Applying Formal Methods to Model
Organizations and Structures
in the Real World

Gerard O’ Regan

Department of Computer Science,
Trinity College, Dublin.

August 8, 1997

This thesis is submitted for the Ph.D in Computer Science at the University of Dublin, Trinity College, Department of Computer Science.
Declarations

I, the undersigned, declare that this thesis has not been submitted to this or any other university.

I declare that all of the material contained in this thesis, unless otherwise stated, is entirely my own work.

I declare my consent to the library of Trinity College, Dublin, that I agree that the library may lend or copy this thesis upon request.

Signature________________ Date_____________
Dedication

This thesis is dedicated to the memory of my father, Cornelius O’ Regan.

Late of Shanbally, Co. Cork.

Departed this world on the 25th. of February 1997.

May he rest in peace.
Abstract

Formal methods are generally employed for the formal specification of computer systems. The formal specification of such a system is an unambiguous statement of the requirements of the system, and is expressed in a formal mathematical notation.

The objective of this thesis is to demonstrate that formal methods may be successfully employed to build mathematical models of aspects of the real world. In particular, this thesis uses the method and notation of the Irish school of VDM to construct various models of the real world. The actual method of construction and evaluation of the models is considered.

This thesis aims to model aspects of organizations and beliefs in a formal way. In particular, this involves the construction of formal models of organizations, hierarchies and belief systems. The model itself then serves as a formal means of testing hypotheses or ideas about some aspects of the world. The modeller employs mathematical techniques to assist in model construction and evaluation.

The modelling exhibited in this thesis are developed using the constructive techniques of $VDM^{\bullet}$. Consequently, the models themselves may be implemented in some programming language if this is required. However, this thesis stresses the importance of studying models for the sake of the models themselves and is not overly concerned with implementation issues for a particular machine architecture. The exploration of these constructive models is a mechanism of shaping aspects of the natural world; models with desirable properties may be implemented in the world. Thus constructive models which currently do not represent an aspect of the existing world may be created in the existing world.

The evaluation of a particular model takes the form of model exploration. All models are adequate at explaining some things and inadequate at explaining other things. The thesis stresses the importance of assessing the adequacy or otherwise of each individual model. In this way, an informed decision may be made as to whether a particular model is a suitable representation of a particular system. The advantage that a formal model serves as a representation of a particular aspect of the real world is that it may be referred to in any dispute on the actual properties of the particular aspect of the real world.

In summary, the key objective of this thesis is to demonstrate that a formal specification language may be employed to model aspects of the world. This thesis explains how such models of the world may be built and evaluated. The key conclusion is that such mathematical models offer a valuable means for the examination of aspects of the world. The said models enable a formal statement of some aspects of the world to be expressed. Further properties may be derived via model interrogation and mathematical reasoning.
Acknowledgements

I am deeply indebted to my supervisor, Dr. Mícheál Mac An Airchinnigh, for his friendship, and for his support and encouragement over the last four years. His sharp and penetrating mind and critical attention to detail has instilled greater discipline and rigour into my own work. Furthermore, his enthusiasm and in depth knowledge of the field, and his ability to express complex ideas in a comprehensible form directly assisted my own growth of knowledge of the field. Finally, I have met few individuals with so much charisma, wit and understanding of fellow human beings.

The formal methods group at Trinity College, Dublin proved to be a fertile forum for the exchange of views on the various approaches to the discipline. I am deeply indebted to all members of the group, past and present. In particular, I must thank Dr. Andrew Butterfield for sharing his in depth knowledge of formal methods and functional programming languages with the group. I must thank Dr. Hugh Gibbons for his expertise in formal logic and other areas which he shared with the group. I am grateful to Alexis Donnelly who shared his in depth knowledge of Π Calculus and CCS with the group. Gradamir Starovic shared his knowledge of distributed systems. I must also thank Eoin McDonnell for sharing his knowledge of Z and B, Dara Gallagher for sharing his knowledge of Miranda and Constructive Logic. Finally, I must express a special thanks to Arthur Hughes who read an earlier draft of this thesis, and for providing constructive comments.

Hans-Juergen Kugler, formerly of K& M Technologies provided generous funding through the SCORE research project. I must thank Professor John Byrne for arranging valuable funding which enabled me to continue with my research work in Trinity. Finally, I would like to express thanks to the academic and non-academic staff of the Computer Science Department at Trinity College Dublin.
# Contents

1 Introduction and Background ................................................. 1
   1.1 Introduction .......................................................... 1
   1.2 The Vienna Development Method ...................................... 3
   1.3 The Irish school of VDM ............................................. 5
   1.4 Notation in the Thesis .............................................. 6
   1.5 Models and Modelling ............................................... 9
   1.6 Goals of the Thesis ............................................... 10
   1.7 Applications of This Thesis ...................................... 12
   1.8 Organization of the Thesis ..................................... 13
   1.9 Summary ............................................................. 13

2 The Religion Model .......................................................... 15
   2.1 Introduction .......................................................... 15
   2.2 The Religion Model ................................................ 16
      2.2.1 The Basic Model .............................................. 17
      2.2.2 Evaluating The Model ...................................... 18
      2.2.3 Formation of New Religions ................................ 25
      2.2.4 The Churches of a Religion ................................ 28
      2.2.5 Heresy and Heretics ....................................... 31
      2.2.6 Conclusions .................................................. 32
   2.3 Specific Religions .................................................. 33
      2.3.1 Judaism ......................................................... 33
      2.3.2 The Christian Religions ................................... 39
      2.3.3 Islam .......................................................... 43
      2.3.4 Hinduism ....................................................... 44
   2.4 Generic Features of Modelling ..................................... 46
   2.5 Summary ............................................................. 50

3 The Stock Exchange Model .................................................. 51
   3.1 Introduction .......................................................... 51
6 The Model of Music

6.1 Introduction ................................................................. 129
6.2 Basic Model of Music ....................................................... 130
   6.2.1 Elementary properties of Model .............................. 131
   6.2.2 Operations on the Model ....................................... 133
   6.2.3 Real Time and the Basic Model .............................. 135
6.3 Multi-Note Model .......................................................... 136
6.4 Orchestra Model ........................................................... 141
   6.4.1 Exploring the model ........................................... 143
   6.4.2 Operations on Orchestra Model .............................. 146
   6.4.3 Criticisms of the Model ...................................... 146
6.5 Constructive Time in Music ............................................ 147
6.6 Elementary Temporal Logic in VDM$^*$ ............................... 149
6.7 Summary ................................................................. 150

7 Conclusions ........................................................................ 151

7.1 Applications of This Thesis .............................................. 154

A Unique Sequences ................................................................ 155

A.1 Introduction ................................................................. 155
A.2 A Study of $(\Sigma^*_1, \Diamond, \Lambda)$ ................................ 156
A.3 Unique Sequences, Sets and $(\tilde{\Sigma}^*_1, \Diamond', \Lambda)$ ....... 158
A.4 Isomorphism of $\tilde{\Sigma}^*_1$ and $P\Sigma$ ............................ 162
A.5 Unique Sequences and $(P\Sigma_\cap, \Sigma)$ ........................ 165
A.6 Canonical Representation of Sequences ............................ 167
A.7 Multiset Representation via Sequences ............................ 170
A.8 Refinement of Sequences into Arrays ............................... 174
A.9 Summary ................................................................. 181

B Indexed Structures ................................................................ 182

B.1 Introduction ................................................................. 182
B.2 Quasi Indexed Monoids .................................................... 184
   B.2.1 Application of Quasi Indexed Groups ....................... 187
B.3 Inherited Operators on Indexed Monoids ............................ 188
B.4 Operators on Quasi Indexed Monoids ............................... 190
B.5 Applications of Operators ............................................. 190
B.6 (Quasi-) Indexed Semi-Rings and Rings ............................ 191
B.7 Indexed Semi-ring ........................................................ 192
   B.7.1 Applications of Indexed Semi Rings ......................... 193
Chapter 1

Introduction and Background

1.1 Introduction

Computer Science is a practical discipline, and is especially concerned with building systems to solve some practical problem. There is evidence [57] that the quality of a software product is largely determined by the quality of the software development and maintenance processes used to build it. The normal software development lifecycle involves specification, design and implementation of the proposed system. In particular, the specification details the requirements which the proposed system must satisfy. The implementation of the specification, i.e., the detailed code, must satisfy the properties of the specification.

Specifications may be presented either formally or informally. The latter are worded in natural language and are subject to the ambiguities inherent in natural language. This means that the original intentions of the author of the specification may be misunderstood in the implementation. Secondly, it is a well accepted in the discipline of software engineering that there are considerable economic benefits if defects are detected as early as possible in the software development life cycle. In particular, the requirements stage is the earliest phase of the lifecycle, and is the foundation stone from which the subsequent phases are built. Consequently, the importance of early identification and elimination of defects from this phase cannot be understated.

The accepted approach to identifying defects within their phase in traditional software engineering is to employ Fagan Inspections [23], [28]. The approach of a Fagan inspection basically is to hold a meeting of the key personnel impacted by a requirement document, design document or source code. In theory, the attendees at the inspection are experts in their area, and are in a position collectively to identify all defects within a requirements or design document or source code. Once the code is implemented it is then subject to rigorous testing by the testing groups to verify that the functionality of the system is as defined in the requirements, and to identify any defects which escaped Fagan detection. Most mature software organizations maintain quantitative measurements [24], [43] in order
to assess the quality of the resulting software. Further background information on quality management is described in [16] and [56].

However, it is clear from this that any analysis performed on a specification written in a natural language is informal. The problem with an informal approach is that it is error prone. This may result in incomplete/missing or mutually incompatible requirements. The limitations of the informal approach has led to a growing interest in the academic and business community in more rigorous and structured methods for software development. This had led to a growing interest in formal methods and formal specification of requirements by the academic community. Indeed, the U.K. Ministry of Defence standards, Def. Standard 00-55 [1] and 00-56 [2], mandate the use of formal methods for safety critical software.

Formal specifications, in contrast to informal techniques, are the use of mathematical notation to describe in a precise way the properties which an information system must have ([66], chapter 1) without unduly constraining the way in which these properties are achieved. It is important to distinguish between the 'how' and the 'what' in formal methods. In general, a formal specification describes what the system must do, as distinct from saying how it is to be done. This abstraction away from implementation enables questions about what the system does to be answered, independently of the implementation, i.e., the detailed code. Furthermore, the unambiguous nature of mathematical notation avoids the problem of ambiguity inherent in a natural language description of a particular system.

The formal specification thus becomes the key reference point for the different parties concerned with the construction of the system. This includes determining customer requirements, program implementation, testing of results, and program documentation. It follows that the formal specification is a valuable means of promoting a common understanding for all those concerned with the system. The very use of a formal notation [54] enforces developers to think and express themselves precisely. Furthermore, the formal specification is compact and captures the essentials in a suitable abstraction. Formal Methods may be applied to real industrial applications, for example, several examples of real industrial applications are presented in [34]. For example, the important problem of feature interaction detection in telecommunications is considered in [14]. The industrial take-up of formal methods is described in [13], and an approach to selling formal methods to industry is described in [68].

The Vienna Development Method proposes a step wise development approach to software development. The formal specification of the requirements is the initial specification. This specification is successively refined until eventually the implemented code is produced. Background material on reification and refinement steps may be found in [4]. Each refinement step has associated proof obligations to demonstrate that the refined specification is a valid refinement of the original specification. Each proof obligation requires mathematical proof. Background information on the discrete mathematics and logic employed in formal
methods may be found in [8], [17], [18], [29], [30], [48] and [60]. Specialized calculi have been developed for concurrency and for the specification of properties of mobile processes. These include CSP, [35], CCS [50], and Π Calculus, [51].

The use of formal methods to precisely state the requirements of the system to be implemented, and the stepwise development approach are the two key areas to which formal methods have been applied. A formal specification describes in a precise way the properties which a particular system must satisfy. It is important to note that there is a definite system in mind; the role of the formal specification is to precisely state what the requirements of the system actually are, and to fully understand the implications of the chosen requirements.

This thesis aims to demonstrate that formal methods may be applied to model aspects of the real world. In particular, the primary goal of this thesis is to show that formal methods may be successfully employed to model aspects of organizations, hierarchies and belief systems. The method and specification language of the Irish school of the VDM [45] is employed to demonstrate this.

The secondary goal of the thesis is to identify any mathematical structures which may prove useful in modelling itself. The appendices of this thesis consider several mathematical structures which may be useful in modelling. The precise goals of the thesis are stated in Section 1.6. First, background information on the Vienna Development Method (VDM) and the variant which is employed in this thesis, i.e., VDM\textsuperscript{•}, is presented.

### 1.2 The Vienna Development Method

VDM dates from the work done at the IBM research laboratory in Vienna. The aim of this research group was to specify the semantics of the PL/1 programming language. This was achieved by employing the Vienna Definition Language (VDL), and adopted an operational semantic approach ([10], chapter 1). This consists of specifying the semantics of a language in terms of a hypothetical machine which interprets the programs of that language. Later work led to the Vienna Development Method (VDM) with its specification language, Meta IV. In particular, this concerned itself with the denotational semantics of programming languages, i.e., (cf., Chapter 1 of [10]), [63], a mathematical object (set, function, etc.) is associated with each phrase of the language. The mathematical object is the denotation of the phrase.

VDM is a ‘model oriented approach’. This means that an explicit model of the state of an abstract machine is given, and operations are defined in terms of this state. Operations may act on the system state, taking inputs, and producing outputs and a new system state.

Operations are defined in a precondition, postcondition style. Each operation has an
associated proof obligation to ensure that if the precondition is true, then the operation preserves the system invariant. The initial state itself is, of course, required to satisfy the system invariant. VDM uses keywords to distinguish different parts of the specification, e.g., preconditions, postconditions are introduced by the keywords pre and post respectively. In keeping with the philosophy that formal methods specifies what a system does as distinct from how, VDM employs postconditions to stipulate the effect of the operation on the state. The previous state is distinguished by employing ‘Hooked variables’, e.g., \( v^- \), and the post condition specifies the new state (defined by a logical predicate relating the pre-state to the post-state) from the previous state.

VDM is more than its specification language Meta IV (termed VDM-SL in the standardization of VDM). It is, in fact, a development method, with rules to verify the steps of development. In particular, rules are provided which enable the executable specification, i.e., the detailed code, to be obtained from the initial specification via refinement steps. Thus we have a sequence \( S = S_0, S_1, ..., S_n = E \) of specifications, where \( S \) is the initial specification, and \( E \) is the final (executable) specification. Each refinement step has associated proof obligations to demonstrate that the more concrete model preserves the properties of the more abstract model. In this way, there is mathematical proof that the implemented code satisfies the requirements, i.e., the initial specification, or most abstract specification. Retrieval functions enable a return from a more concrete specification, to the more abstract specification.

The initial specification consists of an initial state, a system state and a set of operations. The system state is a particular domain, where a domain is built out of primitive domains such as the set of natural numbers, etc., or constructed from primitive domains using domain constructors such as Cartesian product, disjoint union, etc. A Domain invariant predicate may further constrain the domain, and a type in VDM reflects a domain obtained in this way. Thus a type in VDM is more specific than the signature of the type, and thus represents values in the domain defined by the signature, which satisfy the domain invariant. In view of this approach to types, it is clear that VDM types may not be ‘statically typed checked’.

VDM specifications are structured into modules, with a module containing the module name, parameters, types, operations etc. Partial functions are the norm in Computer Science, and formal methods in particular. The problem is that many functions, especially recursively defined functions can be undefined, or fail to terminate for some arguments in their domain. VDM addresses partial functions by employing non-standard logical operators, \( \texttt{viz} \) the logic of partial functions (LPFs [3], page 23), which can deal with undefined operands. In this three valued logic \( \texttt{true} \lor \bot = \bot \lor \texttt{true} = \texttt{true} \); essentially, the result of the Boolean operation is true if at least one of the operands is true, irrespective of whether the other operand is defined or not.
1.3 The Irish school of VDM

The Irish school of the VDM [46] is a variant of classical VDM, and is characterized by its constructive approach, classical mathematical style, and its terse notation. In particular, this method combines the 'what' and the 'how' of formal methods in that, its terse specification style stipulates in concise form what the system should do, and furthermore, the fact that its specifications are constructive (or functional) means that the 'how' is included with the 'what'.

However it is important to qualify this by stating that the 'how' presented by VDM is not directly executable on a particular hardware platform, as several of the mathematical data types and operations have no corresponding equivalent in high level programming languages, or functional languages. Thus a conversion or reification of the specification into a functional or higher level language must take place to ensure a successful execution. Furthermore, it should be noted that the fact that a specification is constructive is no guarantee that it is a good implementation strategy, for example, the construction itself may be naive. This issue is considered in pages 135-7 of [45], the example considered is an efficient construction of the Fibonacci series.

The Irish School follows a similar development methodology as in standard VDM, and is, of course, a 'model oriented approach'. The initial specification is presented, with initial state and operations defined. The operations are presented with preconditions, however no postcondition is necessary as the operation is 'functionally', i.e., explicitly constructed. Each operation has as associated proof obligation, if the precondition for the operation is true and the operation is performed, then the system invariant remains true after the operation. The proof of invariant preservation normally takes the form of 'constructive proofs'. This is especially the case for 'existence proofs', in that the philosophy of the school is to go further than to provide a theoretical proof of existence, rather the aim is to exhibit existence constructively.

Because of this emphasis on constructive existence, the school avoids the existential quantifier of predicate calculus. In fact, reliance on logic in proof is kept to a minimum, emphasis instead is placed on equational reasoning. This is due to the philosophical belief that discrete mathematics is closer to the intuitive level of engineers than predicate logic.

Special emphasis is placed in the method of the school in studying algebraic structures and their morphisms. In particular, structures with 'nice' algebraic properties are sought; such a structure includes the monoid, which is a simple algebraic set with a binary operation such that the closure and associativity properties hold, and the structure possesses a unit element. The monoid structure arises quite frequently in computer science,
for example, words are constructed from alphabets, the null word is the unit, the binary operation is concatenation. Once the abstract structure has been studied and understood then any re-occurrences of the structure may use the existing results and properties of the structure. The concept of isomorphism is powerful, reflecting the fact that two structures are essentially identical, apart from a re-labelling, and thus the modeller has the choice of working with either structure, depending on which is more convenient.

The school has been influenced by the work of Lakatos and Polya [41], [62]. In particular the latter advocated a style of problem solving characterized by solving a complex problem by first considering an easier sub-problem, and considering several examples, which generally leads to a clearer insight into solving the main problem. This philosophy has influenced the standard development steps of VDM as proposed in [46]. These include partitioning or subdividing, i.e., introducing a partition domain $K$ into the original model $(X \mapsto Y)$, yielding $K \mapsto (X \mapsto Y)$. The approach of ‘splitting’ involves splitting a domain $(X \mapsto Y)$ into two connected domains $(X \mapsto S)$ and $(S \mapsto Y)$. Parameterizing of the domain $(X \mapsto Y)$ involves converting the domain to two domains $(P \mapsto X)$ and $(P \mapsto Y)$. Finally, joining involves the join of two distinct domains.

Lakatos’ approach to mathematical discovery (cf., [41]) is characterized by heuristic methods. In particular a primitive conjecture is proposed; if global counter-examples to the statement of the conjecture are discovered, then the corresponding ‘hidden lemma’ for which this global counter-example is a local counter-example is identified and added to the statement of the primitive conjecture. The process repeats, until no more global counter-examples are found. This approach takes a sceptical position on absolute truth and on the absolute certainty that a proof is correct.

Partial functions are the norm in VDM and as in standard VDM, the problem is that recursively defined functions may be undefined, or fail to terminate for some arguments in their domain. The logic of partial functions (LPFs) is avoided, instead care is taken with recursive definitions to ensure termination is achieved for each argument. This is achieved by ensuring that the recursive argument is strictly decreasing in each recursive invocation. Other approaches have been advocated where the ⊥ symbol is used in the Irish School to represent undefined, unavailable or do not care may take part in Boolean operations.

1.4 Notation in the Thesis

The importance of a good notation as a tool of thought has been identified in [37]. The notation of the Irish school of VDM is employed in this thesis. The previous section has provided a brief introduction to the school, and the distinctions between it and classical VDM. The philosophy and method of the Irish school is detailed in [45], [46], and [47]. This section provides a brief introduction to the notation employed in VDM, further details
may be found in [15].

The school places emphasis on terseness in notation, and avoids verbosity. This is stressed in [46], where it is observed that verbosity hinders the discovery of theorems and the carrying out of proofs. The notation of the Irish school is very striking and consists of symbols in both the Greek and Roman alphabets. Lowercase letters of the Greek alphabet are used for representing partial maps, sequences, etc. Uppercase Roman letters are used for representing domains, lowercase Roman letters represent elements of a domain.

Thus \( \mu \in (X \rightarrow Y) \) indicates that \( \mu \) is a partial map. The notation \((X \rightarrow Y)\) is the domain of all partial maps between set \( X \) and set \( Y \). In classical VDM, \((X \rightarrow Y)\) represents all total functions between set \( X \) and set \( Y \). In the Irish school, total functions are regarded as a special case of partial functions. The domain of a partial map \( \mu \) is given by \( \text{dom} \mu \) and the range of \( \mu \) is given by \( \text{rng} \mu \). The map \( \mu \) is one to one if \( |\text{dom} \mu| = |\text{rng} \mu| \).

The structure \( X^* \) represents the domain of sequences of elements of \( X \). The corresponding representation in the English school of VDM is \( \text{seq} \) of \( X \). The empty sequence is represented by \( \Lambda \). Individual sequences may be represented by \( \tau_1, \tau_2 \in X^* \). The sequence concatenation operation is \( \circ \), and the concatenation of two sequences \( \tau_1, \tau_2 \) is given by \( \tau_1 \circ \tau_2 \). The singleton sequence element is represented by \( \langle \sigma \rangle \).

Set theoretical operations are as for classical mathematics. The union of two sets \( S_1, S_2 \) is given by \( S_1 \cup S_2 \). Similarly, set intersection, set difference operations, etc. are as in classical mathematics. The disjoint union of two sets \( X_1, X_2 \) is given by \( X_1 + X_2 \). Suppose \( X_1, X_2, \ldots, X_n \) are sets, then the Cartesian product is given by \( X_1 \times X_2 \times \ldots \times X_n \). The powerset of a set \( X \), i.e., the set of all subsets of \( X \) is given by \( \mathcal{P} X \).

The powerset of a set \( X \), i.e., \( \mathcal{P} X \) includes the emptyset \( \emptyset \). \( \mathcal{P} X \) represents the power set of the set \( X \) excluding the emptyset. The structures \((X \rightarrow \mathcal{P} Y)\) and \((X \rightarrow \mathcal{P} Y)\) arise frequently in modelling. The distributed union operation may be explained as follows. Suppose \( \mu \in (X \rightarrow \mathcal{P} Y) \), then \( \text{rng} \mu = \{S_1, S_2, \ldots, S_n\} \). Then \( \cup/ \circ \text{rng} \mu = S_1 \cup S_2 \cup \ldots \cup S_n \).

Curried notation is used extensively in functional programming languages. The idea is to treat a function of \( n \) arguments as the application of \( n \) single argument functions [25], [59]. It is named after H.B. Curry and is due to Schönfinkel [64]. Currying is employed in the Irish school of VDM.

The monoid is a fundamental structure which arises quite frequently in Computer Science. The structure is well behaved, and consists of a set with a binary operation such that closure and associativity properties hold. Furthermore, the structure has a unit element \( u \).

**Definition**

A **monoid** \((M, *, u)\) is a non empty set \( M \) with a binary operation \( * \), and an identity element \( u \) such that the following laws hold.

- \( m_1 * m_2 \in M \) for all \( m_1, m_2 \in M \)
\( (m_1 * m_2) * m_3 = m_1 * (m_2 * m_3) \), for all \( m_1, m_2, m_3 \in M \).

\[ m * u = u * m = m \text{ for all } m \in M. \]

A structure \( M \) which satisfies the closure and associativity properties is a semi-group. Every monoid is ipso facto a semi-group. A monoid is commutative if \( m_1 * m_2 = m_2 * m_1 \) for all \( m_1, m_2 \in M \). A subset \( N \) of a monoid \((M, *, u)\) is termed a submonoid of \( M \) if and only if (iff) \( N \) contains the identity \( u \) and \( N \) is closed under the binary operation \(*\).

**Definition**

A Group \((G, *, 1)\) is a monoid such that every element of \( G \) is invertible, i.e., for every \( g \in G \) there is a corresponding unique \( g^{-1} \) such that

\[ g * g^{-1} = g^{-1} * g = 1 \]

Groups arise less frequently than monoids in practice. Other algebraic structures which may arise are semi-rings, rings, integral domains, division rings, fields and vector spaces. For further details on the above, any standard textbook on modern algebra, e.g., [3], [33], [55] should be consulted.

A homomorphism is a mapping from one structure to another which preserves structure. A homomorphism from the monoid \((M, *, u)\) to the monoid \((P, +, v)\) is a map \( h : M \to P \) such that:

\[
\begin{align*}
  h(m_1 * m_2) &= h(m_1) + h(m_2) \\
  h(u) &= v
\end{align*}
\]

A monomorphism is a one to one homomorphism; an epimorphism is an onto homomorphism; an isomorphism is a bijective homomorphism. An endomorphism is a homomorphism from \((M, *, u)\) to \((M, *, u)\). A bijective endomorphism is an automorphism. Homomorphisms, endomorphisms, etc. enable simplification of complex VDM\(^*\) expressions. For example, the fact that a particular map \( \kappa \) is an endomorphism means that it can be applied separately to two arguments in the following expression:

\[ \kappa(m_1 * m_2) = \kappa(m_1) * \kappa(m_2) \]

This eases the evaluation of complex expressions.

Indexed Monoids inherit monoidal properties from the underlying base monoid. Suppose \((M, *, u)\) is a monoid, and \( I \) is an arbitrary index set, then \((I \mapsto M, \circ, \theta)\) is a monoid. The indexed structure inherits its behaviour from the base monoid. Suppose
\( \mu_1, \mu_2 \in (I \mapsto M, \otimes, \theta) \), then \( \mu_1 \otimes \mu_2 \) represents the operation on the indexed structure. For example, consider the monoid \((\mathcal{P}S, \cup, \emptyset)\), then the structure \((I \mapsto \mathcal{P}S, \emptyset, \theta)\) is a monoid. The semi-group \(M'\) is obtained from the monoid \(M\) by the deletion of the unit element \(u\). There is a corresponding indexed structure \((I \mapsto M', \otimes', \theta)\).

In fact, the bag structure is an indexed monoid. The base monoid is \((\mathbb{N}, +, 0)\), and the index set is \(I\), thus the indexed structure, i.e., the standard bag, is of the form \((I \mapsto \mathbb{N}, \oplus, \theta)\). Bag addition takes the form of \(\mu_1 \oplus \mu_2\), where \(\mu_1, \mu_2 \in (I \mapsto \mathbb{N}, \oplus, \theta)\).

### 1.5 Models and Modelling

A model is a representation of the physical world. However, the model is not the reality; for example, the model of a bridge is unlikely to include the colour of the bridge. Typically, models are mathematical representations of the physical world. Models are simplifications of the reality; consequently, they do not include all aspects of the reality.

In particular, it is important that all the key attributes of the reality should be included in the model. For example, the model of the Tacoma Narrows Bridge did not include aerodynamic forces. This had a major influence on the eventual collapse of the bridge.

It follows that it is necessary to explore the consequences of a model, and to determine if the model is an adequate representation of the reality. Occasionally, there may be more than one model to explain the reality. For example, Ptolemy's Cosmological Model and the Copernican Model. Both models are adequate at explaining aspects of navigation. In fact, [40] the Copernican model was less accurate than Ptolemy's model, until the former was revised by Kepler.

Occam's Razor (or the 'Principle of Parsimony') [45] is a key principle underlying modelling. The principle is stated as follows: 'Entia non sunt multiplicanda praeer necessitate'; this essentially means that the number of entities required to explain anything should be kept to a minimum. The implication of this principle is that the modeller should seek the simplest model with the least number of assumptions. The principle is attributed to the medieval philosopher William of Ockham.

The key application of Occam's Razor in practice is to remove all superfluous concepts which are not needed to explain the phenomena. The net result is a crisp and simpler model. In theory, this should reduce the likelihood of introducing inconsistencies and errors into the model. Such a model captures the essence of the reality.

In summary, a model is an abstraction or simplification of reality. Model exploration enables an informed decision to be made on the adequacy or otherwise of the model. The model should be kept as simple as possible.
1.6 Goals of the Thesis

Formal methods are generally employed for the formal specification of computer systems. The formal specification of such a system is an unambiguous statement of the requirements of the system, and is expressed in a formal mathematical notation.

The objective of this thesis is to demonstrate that formal methods may be successfully employed to build mathematical models of systems other than computer systems, and in particular, to develop models of aspects of the real world. In particular, this thesis uses the method and notation of VDM* to construct various models of aspects of the real world. For example, the thesis demonstrates that formal models of organizations, hierarchies and belief systems may be constructed in this manner. The key point is that such models are not developed for implementation purposes; indeed, implementation may not be meaningful for several of these domains.

The importance of the said models is that they serve as representations of these domains, and may be referred to in any dispute on the actual properties of the domain. Furthermore, the model serves as a means of formal derivation of further properties of the particular aspect of the real world. The very fact that there is a model means that there is a rigorous means of gaining a more detailed understanding of the real world. The actual method of construction and evaluation of the models is considered. In general, a model is good at explaining some aspects of the world, and weak at explaining other aspects. The most adequate model is chosen as a representation of the aspect of the world.

This thesis considers the problem of modelling the domain of religions of the world in a formal way. It is believed that the model of religion which is developed in this thesis is original. The modelling is concerned with the problem of representing the structure of the organized religions of the world, and in capturing the essential beliefs of a religion. It is accepted that religions of the world are far too complex to capture exactly; however, the model developed in this thesis demonstrates that important aspects of religion may be captured formally. Furthermore, model interrogation enables additional properties of religion to be determined. The study of the model of religion indicated that the model itself is generic, and may be applied to model the hierarchical structure of companies, political institutions, or a university. This realization justified an examination of the classical bill of material structure [12], as bills of materials are abstractions of hierarchies. Furthermore, the representation of beliefs and facts about a religion resembles a catalogue of information about the religion. This suggested an examination of the catalogue from the classical model of the file system in [10]. Consequently, the model of the file system is examined in detail.

This particular use of formal methods to model a domain for which there is no intention to implement, and for which implementation is not meaningful is believed to be an original application of formal methods. The importance of this approach is that it enables a more detailed understanding of these domains to be gained and suggests that it may be valid to
employ formal methods to model aspects of the social sciences.

The model of the stock exchange presented in this thesis is believed to be original. The model considers the problem of modelling companies registered on the stock exchange, the shareholders of the company, and the financial operations involved in share sale or purchase. This importance of this model is that it demonstrates that formal methods may be employed to model aspects of the structure and operations of an organization in the real world. The stock exchange is one particular instantiation of organizations.

The thesis aims to model aspects of organizations and beliefs in a formal way. In particular, this involves the construction of formal models of organizations, hierarchies and belief systems. The model itself then serves as a formal means of testing hypotheses or ideas about some aspects of the world. This involves the formulation of questions which are then answered in terms of the model. This thesis adopts the philosophy of $VDM^*$, and aims to provide constructive specifications and models. In some cases, there are some deviations from the Irish school of the VDM in both notation and the emphasis of constructive models. Any deviations from the notation employed in the Irish school are explained.

The minor deviations from the notation in the Irish school are justified by the views on notation held by those involved in the pioneering work in VDM. In particular, they argued [11] that in view of the mathematical nature of VDM, and the fact that there are several open areas of research which need to be addressed, that VDM would continue to evolve. Consequently, this thesis considers the minor deviations from the standard notation of the Irish school to be justifiable.

In general, the models presented in this thesis are constructive. Consequently, the models themselves may be implemented in some programming language if this is required. However, this thesis stresses the importance of studying models for the sake of the models themselves and is not concerned with implementation issues for a particular machine architecture. The fact that a model is constructive indicates that it may be implemented or created in the real world. This suggests that the constructive models may be employed to shape the world. In this thesis, almost all of the models which are presented are constructive.

It is a key requirement to determine how adequate a particular model is of an aspect of the world. In general, the evaluation of a particular model takes the form of model exploration. The thesis stresses the importance of assessing the adequacy or otherwise of each individual model. In this way, an informed decision may be made as to whether a particular model is a suitable representation of a particular system.

The secondary objective of this thesis is to identify mathematical structures which may be useful in the modelling itself. Several mathematical structures which may be useful in modelling are presented in the appendices to this thesis. It is intended that such structures should have practical applications in modelling.
In summary, the key objective of this thesis is to demonstrate that a formal specification language may be employed to model aspects of the world. This thesis explains how such models of the world may be built and evaluated. The key conclusion is that such mathematical models offer a valuable means for the examination of aspects of the world. The said models enable a formal statement of some aspects of the world to be expressed. Further properties may be derived via model interrogation and mathematical reasoning.

1.7 Applications of This Thesis

There are several potential applications of this thesis to computer science. The actual construction and evaluation of models as exhibited in this thesis may be interpreted as modelling for the sake of modelling. This approach may be applicable to determining the appropriate requirements for a particular project. In general, requirement exploration concerns itself with exploring various requirements for the proposed system, i.e., before the explicit requirements are actually detailed. In fact, determining the appropriate requirements for a proposed system is a non-trivial task. In fact, it is often the case that further desirable requirements which require implementation only become apparent at a late stage during the software project. Such an occurrence has a corresponding effect on the quality of the resulting software, since design documents, coding and test plans are affected at a late stage in the project.

The modelling exhibited in this thesis may be applied to assist in determining the appropriate requirements for a proposed system. There may be several candidate sets of requirements, $R_1, R_2, \ldots, R_n$, where each $R_i$ represents a set of requirements. Suppose $M_1, M_2, \ldots, M_n$ are adequate models for $R_1, R_2, \ldots, R_n$ respectively, then the exploration of the properties of these models serves as a rigorous and methodical means of determining the most appropriate model $M_j$ and consequently, the most appropriate set of requirements $R_j$.

Each model is explored for evaluation and understanding purposes, in order to determine the most appropriate choice of model for the requirements of a proposed system. Thus modelling for the sake modelling serves practical purposes in this case.

It is important to distinguish between the use of formal methods in developing these models of various sets of requirements and prototyping as these are quite distinct. The latter ([49], page 104), is used for building a working system early in the life cycle of the project, and is normally an interpreted language implementation. The exact requirements for the system is determined by an iterative process. The initial prototype and its properties are studied. From this study, further desirable properties are identified, and the prototype is revised accordingly. The exact user requirements are achieved via successive prototype approximations [67].
However, the key differences between the model exploration using formal methods and prototyping is that for the latter there is a definite system in mind, and secondly, the prototype itself is a usable model, i.e., implemented on a computer. The former is a mathematical model and is not subject to implementation constraints. Its main purpose is for exploration of properties and understanding the domain.

Modelling for the sake of modelling exhibited here in this thesis may be applicable to identifying potential new products or systems. This is since the models presented here are constructive; it follows that they may be implemented on some machine architecture. Thus a formal means of understanding the implications of a new product may be obtained in this way. The aim is to identify potential systems which may be worthy of eventual implementation. Many of these possible systems may be of little practical interest, as they may have minimal applicability to the world. However, the exploration may yield models with desirable properties which may then be successfully implemented.

1.8 Organization of the Thesis

The rest of the thesis is organized as follows. Chapter 2 presents the model of religions of the world. The model demonstrates that aspects of religion, a domain which is unrelated to computer science may be successfully captured via VDM*. Models of aspects of specific religions of the world are developed. Finally, it is noted that the religion model has a generic nature which may be successfully applied to other domains. In particular, the religion model may be adapted to models of hierarchies, including modelling political parties and a university model.

Chapter 3 demonstrates that a model of an industrial organization may be successfully developed in the Irish school. In particular, a model of the stock exchange is developed. This model demonstrates that aspects of the stock exchange, may be understood via a formal model. Chapter 6 demonstrates how models of music and culture may be developed.

Chapter 4 examines the bill of material structure, i.e., an abstraction of a hierarchy in detail. Chapter 5 considers a model of the file system, i.e., an abstraction of a catalogue of information in detail. The secondary goal of this thesis, as explained in Section 1.6 is to identify mathematical structures which are useful in the modelling itself. In particular, the appendices contain a detailed examination of these structures.

1.9 Summary

The formal specification of the requirements of a proposed system is the main application to which formal methods is applied. This thesis demonstrates that formal methods may be adapted to mathematical modelling of aspects of the real world. In particular, the thesis
employs the method and notation of the Irish school of VDM to develop constructive models of organizations, hierarchies and belief systems.

The models serve as representations of these domains, and may be referred to in any dispute on the actual properties of the domain. The method of construction and evaluation of the models is considered. Each model is adequate at explaining some things about the world, and inadequate at explaining other things. The fact that a model serves as a representation of an aspect of the world enables further properties and a more detailed understanding of the world to be gained.

In general, the models presented in this thesis are constructive. Consequently, the models themselves may be implemented in some programming language if this is required. However, this thesis stresses the importance of studying models for the sake of the models themselves and is not concerned with implementation issues. The fact that a model is constructive indicates that it may be created in the real world.

The secondary objective of this thesis is to identify mathematical structures which may be useful in the modelling itself. These are presented in the appendices to the thesis. It is intended that such structures may be practically applied in modelling.

In summary, the thesis demonstrates that a formal specification language may be employed to model aspects of the world. This actual construction and evaluation of the models is discussed. The key conclusion is that such mathematical models offer a rigorous means for examining aspects of the world.
Chapter 2

The Religion Model

2.1 Introduction

The objective of this chapter is to demonstrate that formal methods may be employed to develop a formal model of aspects of religions of the world. The actual construction and evaluation of the model is considered. The model aims to capture the essential structure and beliefs of several of the major religions of the world at an abstract level. It is believed that the model of religion which is developed in this chapter is original. The model is constructive indicating that it may be implemented. However, there is no intention to implement the model. The purpose of the model is to demonstrate that a model developed in VDM may serve as a formal representation of aspects of religion.

The adequacy or otherwise of the formal model of religion is considered. The key requirement for a model to be adequate is that it should accurately reflect the known properties of the domain, and enable further properties to be deduced. An adequate model of religion then serves as an unambiguous reference point for this particular domain, and may be referred to in any dispute on the exact properties of the domain. The properties of the formal model are determined by model interrogation, and the use of the formal model is a particularly terse and economic means of recording these properties.

Inadequacies in a model are identified via model interrogation. These inadequacies may be addressed either by extensions to the model or by a complete replacement of the model. The inadequacies discovered in the model indicate the limitations of the model. The evaluation of the religion model indicates that it is adequate at explaining aspects of religion; in particular, it provides a good abstract representation of the hierarchical structure and belief system of religions. There are several inadequacies in the model; for example, the model does not consider the problem of modelling conscience and free will. Secondly, the existence or otherwise of God may not be determined from the model. Finally, the basic model does not explain the Christian concept of the Trinity. The model may be extended or revised to address any identified inadequacies. Consequently, the building of
an appropriate model for religion is achieved via model evolution. At each evolving step there is an evaluation of the model. The intention is that the final model is sufficiently adequate to be chosen as a representation for the particular domain of religion.

The advantages gained by developing such a formal model is that it serves as an explicit and precise description of aspects of this particular domain. The model enables properties of religions of the world to be deduced, and the model serves as a means of formally encoding properties of this particular domain. It is accepted that the domain of religion is too complex to capture exactly in a formal model; however, the model indicates that important aspects of religion may be captured formally. A formal model of religion serves as a means of testing hypotheses or ideas about some aspect of religion.

The study of the religion domain demonstrates generic aspects to modelling. In particular, at the abstract modelling level there is a close relationship between the model of religion as presented here, a model of politics, a model of a university, and a model of a company. The generic aspect of the religion model indicates that at a sufficiently abstract level there is a close relationship between these domains. The key similarity is the hierarchical structure of these diverse organizations. The identification of generic domains avoids duplication of effort; proofs need to be conducted for the generic model only, the dedicated models automatically satisfying the proofs.

2.2 The Religion Model

The search for a meaning to existence is fundamental to mankind. One particular explanation to existence is in the belief in a particular deity; the appeasement of the deity being important in some cultures. The archaeological remains from the ancient civilizations, in particular, the temples of Karnak, Luxor, and Abu Simbel from ancient Egypt demonstrate that organized religion has been part of society for several millennia. The objective here is to develop a formal model of religion.

The theistic approach (i.e., belief in a deity) may be divided into mono-theistic or poly-theistic religions. The Christian religion complicates matters by its dogma of the Trinity, the stipulation being that the religion is mono-theistic, the three of the Trinity being one in some sense. The original version of the ancient Egyptian religion [22] was mono-theistic, poly-theism being introduced over its history. The schism by Akhenaten [22] was an attempt to re-introduce the traditional mono-theistic values of the ancient Egyptian religion.

An organized religion has an associated code of ethical and moral behaviour, and customs and rituals for birth, marriage, death, etc. These rituals may vary considerably between the organized religions, with polygamy valid for Muslims, but not for Christians; divorce is valid in Judaism but not for Roman Catholics; a married clergy is allowed for
2.2.1 The Basic Model

The basic model of religion considered in this section provides a formal representation of the structure of religion. The following assumptions are made in the model:

1. Every person is a member of at most one religion (Slav serves one master constraint).
2. Agnosticism and atheism are valid religions, and are distinguished by the former having no beliefs, and the latter rejecting the belief in a deity.
3. Each religion has an associated set of gods, and an associated set of beliefs.
4. The churches of a religion are physical structures where worship is performed.
5. Each church belongs to a particular religion, and is in a specific physical location.
6. The worship in churches is performed by the priests, and the priests have the same religious affiliation as the church.

The basic model of religions of the world is defined as follows:

\[ \alpha : \text{Per}_Jd \mapsto \text{Rel}_Jd \]  
\[ \alpha_1 : \text{Per}_Jd \mapsto \text{B} \]  
\[ \beta : \text{Rel}_Jd \mapsto \mathcal{P}\text{God}_Jd \]  
\[ \gamma : \text{Church}_Jd \mapsto \text{Rel}_Jd \]  
\[ \delta : \text{Church}_Jd \mapsto \mathcal{P}\text{Priest}_Jd \]  
\[ \epsilon : \text{Church}_Jd \mapsto \text{Loc}_Jd \]  
\[ \mu : \text{Rel}_Jd \mapsto \mathcal{P}\text{Bel}_Jd \]  

Notes
1. \( \alpha \) models individual membership of religions.
2. An individual may be a member of at most one religion.
3. \( \alpha_1 \) models whether the member of the particular religion is alive.
4. \( \beta \) models the gods that a particular religion believes in.
5. Different religions may share beliefs in common gods.
6. \( \gamma \) models the religious affiliation of individual churches.
7. \( \delta \) models the assignment of priests to churches.
8. A priest may be assigned to several churches.
Note that $\text{Priest}\, Id = \text{Per}\, Id$.
9. $\epsilon$ models the physical location of a church.
10. $\mu$ models the beliefs that the individual religions profess.

The invariant for the religion model is defined as follows:

\[
\text{Inv}_{\text{Rel}} : (\text{Per}\, Id \mapsto \text{Rel}\, Id) \times (\text{Per}\, Id \mapsto B) \times (\text{Rel}\, Id \mapsto \mathcal{P}\, \text{God}\, Id) \\
\times (\text{Church}\, Id \mapsto \text{Rel}\, Id) \times (\text{Church}\, Id \mapsto \mathcal{P}\, \text{Priest}\, Id) \times \\
(\text{Church}\, Id \mapsto \text{Loc}\, Id) \times (\text{Rel}\, Id \mapsto \mathcal{P}\, \text{Bel}\, Id) \mapsto B
\]

\[
\text{Inv}_{\text{Rel}}[\alpha, \alpha_1, \beta, \gamma, \delta, \epsilon, \mu] \triangleq \\
\text{dom } \alpha \subseteq \text{dom } \alpha_1 \\
\land \text{rng } \alpha \subseteq \text{dom } \mu \\
\land (\text{dom } \beta = \text{dom } \mu) \lor \text{dom } \mu = \text{dom } \beta \cup \mu^{-1}(\emptyset) \\
\land \text{dom } \delta = \text{dom } \gamma = \text{dom } \epsilon \\
\land (I \mapsto \mathcal{P} \alpha) \land [S] \delta = \alpha[S] \gamma
\]

where $S = \emptyset[\delta^{-1}(\emptyset)] \text{dom } \delta$

**Comment 2.1** There is a slight abuse of notation in the last line of the invariant. The signature of $(I \mapsto \mathcal{P} \alpha) \land [S] \delta$ is $(\text{Church}\, Id \mapsto \mathcal{P}\, \text{Rel}\, Id)$, whereas the signature of $\gamma$ is $(\text{Church}\, Id \mapsto \text{Rel}\, Id)$.

**Notation 2.1** The effect of the functional $(f \mapsto g)$ applied to $\mu$ is to apply $f$ to the domain of $\mu$ and $g$ to the range of $\mu$. It is required that $f$ be one to one.

This is the final formulation of the invariant for the basic model of religion. However, the invariant evolved as the model and an understanding of the model and problem domain evolved. The modelling enabled a deeper understanding of the problem domain to be gained. This resulted in further evolution of the model and the constraints on the model. Essentially, the formulation of the model and the invariant were symbiotic. The steps involved in the formulation of the invariant are described later.

### 2.2.2 Evaluating The Model

In this section the model of religion is evaluated to determine its adequacy. The objective is to determine the effectiveness of the model to capture the essential properties of aspects of religion. The evaluation of the religion model consists of determining the effectiveness of the model in answering pertinent questions on aspects of religion.
It should be noted that $\alpha_1$ models whether a person is alive in the world. The model is updated with each addition and departure from the human race. Thus $\text{dom} \, \alpha_1$ is partitioned into a living and a non-living part. The former is given by $\text{Liv} = \alpha_1^{-1}(1)$ or $\emptyset$ depending on whether there is any existing human life on the planet. The latter is given by $\preceq \|\text{Liv}\| \alpha_1$. The present membership of a religion is of special interest, as distinct from its previous membership, whether living or not.

Religions may be revised with doctrinal and dogma changes. Such updates may be in response to a crisis within the religion, for example, schisms which sometimes occur, or may reflect the response of a religion concerned with ensuring that its dogma is optimal for the welfare of its members. The evaluation of the religion model takes the form of model exploration and model interrogation. The exploration of the model yields several elementary properties of the domain of religion.

This section determines if answers to the following relevant questions may be obtained in terms of the model of religion.

**Model Evaluation Questions**

1. What religion does person $p$ have?
2. Who are the members of religion $r$?
3. What are the different religions of the world?
4. What are the *atheistic* religions of the world?
5. Who are the *atheists* in the world?
6. Which religions are *monotheistic*?
7. Which religions are *polytheistic*?
8. What *gods* are believed in?
9. Which religions believe in god $g$?
10. What churches does religion $r$ have?
11. What priests are assigned to church $c$?
12. What churches does city (or location) $l$ have?
13. What churches does religion $r$ have in location $l$?
14. Does the model address heresy?

The adequacy or otherwise of the basic model of religion is judged by the effectiveness of the model in answering the above questions. Each question may be answered by defining operations on the model. If the model answers these questions reasonably effectively then it may be chosen as a representation of aspects of religion. It is not intended that the model should aim to explain all aspects of religion, as the domain of religion is far too complex to capture formally. The intention is to demonstrate that the model is effective.
at capturing aspects of the domain of religion.

Several operations and their pre-conditions are presented. The operations provide the mechanism whereby the individual questions may be answered. The preconditions indicate the constraints which must hold before the operations may be invoked.

**Question 2.1 What religion (if any) is p a member of?**

This question is answered in the model by defining an operation which determines the particular religion that p is a member of. The $\text{Rel\_Type}$ operation determines the religion of p from $\alpha$, provided p is a member of some religion. The latter constraint indicates the precondition for the operation.

$$\text{Rel\_Type} : \text{Per\_Jd} \mapsto (\text{Per\_Jd} \mapsto \text{Rel\_Jd}) \mapsto \text{Rel\_Jd}$$

$$\text{Rel\_Type}[p] \alpha \triangleq \alpha(p)$$

$$\text{pre}_{\text{Rel\_Type}} : \text{Per\_Jd} \mapsto (\text{Per\_Jd} \mapsto \text{Rel\_Jd}) \mapsto \mathbf{B}$$

$$\text{pre}_{\text{Rel\_Type}}[p] \alpha \triangleq \chi[p] \alpha$$

The next operation to be considered is the $\text{Mem\_Rel}$ operation, which determines the current members of religion $r$. This information is important for the individual religions, and enables membership growth or contraction figures to be determined. The operation determines the current existing membership for the individual religions.

**Question 2.2 Who are the members of religion $r$?**

This is given by the $\text{Mem\_Rel}$ operation, and is defined as follows:

$$\text{Mem\_Rel} : \text{Rel\_Jd} \mapsto (\text{Rel\_Jd} \mapsto \mathcal{P}\text{Per\_Jd}) \mapsto \mathcal{P}\text{Per\_Jd}$$

$$\text{Mem\_Rel}[r] \alpha^{-1} \triangleq <[\text{Liv}]\alpha^{-1}(r)$$

The precondition must ensure that $r$ is actually a religion of the world. There is no guarantee that the religion will have any current members as $r$ may be an extinct religion, or a newly created religion.

$$\text{pre}_{\text{Mem\_Rel}} : \text{Rel\_Jd} \mapsto (\text{Rel\_Jd} \mapsto \mathcal{P}\text{Per\_Jd}) \mapsto \mathbf{B}$$

$$\text{pre}_{\text{Mem\_Rel}}[r] \alpha^{-1} \triangleq \chi[r] \alpha^{-1}$$

The $\text{Mem\_Rel}$ operation enables the current membership size of religion $r$ to be immediately determined.
\[ N_{mr\ Rel}[r] \alpha^{-1} \triangleq |Mem_{\ Rel}[r] \alpha^{-1}| \]

The \textit{Mem\_Rel} operation may be employed to determine all of those who are a member of some organized religion of the world. This may include membership of the atheist or agnostic religions. It is given by the \textit{Mem\_Rels} operation.

\[
\text{Mem\_Rels} : (RelId \mapsto \mathcal{P}PerId) \mapsto \mathcal{P}PerId \\
\text{Mem\_Rels}[\alpha^{-1}] \triangleq \mathcal{Liv} \circ \cup / \circ \text{rng} \alpha^{-1}
\]

\[
\text{Pop\_Rels} : (RelId \mapsto \mathcal{P}PerId) \mapsto \mathbb{N} \\
\text{Pop\_Rels}[\alpha^{-1}] \triangleq |\text{Mem\_Rels}[\alpha^{-1}]|
\]

\textbf{Practiced and Extinct Religions}

This objective of this section is to determine if the model can adequately distinguish between religions which are currently practised in the world and the extinct religions of the world. The second objective of this section is to determine if the model may adequately distinguish between \textit{atheism} and \textit{agnosticism}. This enables the list of religions of the world which are neither agnostic nor atheistic to be determined.

\textbf{Question 2.3 What are the different religions in the world?}

This question may be sub-divided as follows:

1. What are the active religions practised in the world today?
2. What are the extinct religions of the world?
3. What is the distinction between agnosticism and atheism in the model?
4. What are the religions of the world, i.e., either practised or extinct, including atheism and agnosticism?

\textbf{Note 1:} \texttt{rng} \circ \mathcal{L}[\alpha] = \{r_1, r_2, \ldots, r_n\} is the list of practised religions in the world.

\textbf{Note 2:} \texttt{rng} \backslash \texttt{rng} \circ \mathcal{L}[\alpha] = \{r'_1, r'_2, \ldots, r'_k\} is a list of extinct religions of the world, however, this list may be incomplete.

The complete list of extinct religions of the world is given by:

\[ \text{dom} \mu \backslash \text{rng} \circ \mathcal{L}[\alpha] \]
Note 3: \( \text{dom } \beta \) is a list of all religions in the world both current or extinct, but does not include agnosticism.

Note 4: \( \text{dom } \mu \) represents all religions in the world both current and past, including both the atheism and agnosticism.

Note 5: The distinction between agnosticism and atheism is the following. For agnosticism it is inappropriate to state a position on belief in a deity; consequently, agnostic religions are not modelled by \( \beta \), whereas atheism is modelled in \( \beta \) by the explicit rejection of belief in a deity. Agnosticism is modelled in \( \mu \) as a religion which has no beliefs, and is not modelled by \( \beta \). Agnosticism is represented in \( \mu \) by an entry of the form \([a_g] \rightarrow \emptyset\).

Note 6: There are \( |\text{rng } \circ [L\text{iv}] \alpha| \) religions practised in the world today.

## Building the Invariant

The invariant was presented without explanation in Section 2.2.1. The final form of the invariant was built in stages, as the model and the understanding of the problem domain evolved. The initial formulation of the invariant is the following.

\[
\text{Inv}_{\text{Rel}}[\alpha, \beta, \mu] \triangleq \\
\text{rng } \alpha \subseteq \text{dom } \mu \\
\land \text{dom } \beta \subseteq \text{dom } \mu \\
\land \text{rng } \alpha \subseteq \text{dom } \beta \cup \{a_g\}
\]

Note
1. The set of previously or currently practised religions is a subset of all listed religions.
2. The list of all religions which profess or reject a belief in a deity is a subset of all listed religions.
3. The set of previously or currently practised religions is a subset of the religions which profess or reject a belief in a deity and the agnostic religion.

However, since there may be several agnostic religions in the model, the correct formulation of the invariant is the following:

\[
\text{Inv}_{\text{Rel}}[\alpha, \beta, \mu] \triangleq \\
\text{rng } \alpha \subseteq \text{dom } \mu \\
\land \text{dom } \beta \subseteq \text{dom } \mu \\
\land \text{rng } \alpha \subseteq \text{dom } \beta \cup \mu^{-1}(\emptyset)
\]
Atheism

QUESTION 2.4 Which current or past religions have rejected belief in a deity, i.e., what are the current or past atheistic religions?

This question is given by the \( \text{Ath}_{\text{Rel}} \) operation, which is defined using inverse images as follows:

\[
\text{Ath}_{\text{Rel}} : (\mathcal{P} \text{God}_{\text{Id}} \mapsto \mathcal{P} \text{Rel}_{\text{Id}}) \mapsto \mathcal{P} \text{Rel}_{\text{Id}}
\]

\[
\text{Ath}_{\text{Rel}}[\beta^{-1}] \triangleq \\
\chi[\emptyset] \beta^{-1} \\
\mapsto \beta^{-1}(\emptyset) \\
\mapsto \emptyset
\]

The number of atheist religions currently or previously in the world is given by \(|\text{Ath}_{\text{Rel}}[\beta^{-1}]|\).

QUESTION 2.5 Who are the current atheists in the world?

This question is of importance to religious orders as it is a measure of their success if the number of atheists in the world is low. By measuring this over a time interval, trends in religious belief may be derived. The \( \text{Aths}_{\text{Wld}} \) operation gives the atheists in the world.

\[
\text{Aths}_{\text{Wld}} : (\text{Rel}_{\text{Id}} \mapsto \mathcal{P} \text{Per}_{\text{Id}}) \times (\mathcal{P} \text{God}_{\text{Id}} \mapsto \mathcal{P} \text{Rel}_{\text{Id}}) \mapsto \mathcal{P} \text{Per}_{\text{Id}}
\]

\[
\text{Aths}_{\text{Wld}}[\alpha^{-1}, \beta^{-1}] \triangleq \\
\llbracket \text{Liv} \rrbracket \circ \alpha^{-1} / \circ \mathcal{P} \alpha^{-1}(\llbracket \text{dom} \alpha^{-1} \rrbracket \beta^{-1}(\emptyset))
\]

Consequently, the number of atheists in the world today is given by:

\[
|\text{Aths}_{\text{Wld}}[\alpha^{-1}, \beta^{-1}]|
\]

Mono-theism and Poly-theism

This section determines the effectiveness of the model in differentiating between religions which are mono-theistic or poly-theistic. Christianity and Islam are mono-theistic religions, i.e., both religions believe in a single god; the Christian form allowing the division of the one God into three, i.e., the Trinity. Many religions through the ages have professed beliefs in a collection of deities, i.e., polytheism, for example, the ancient Egyptian religion, Celtic, ancient Greek and Roman mythology, etc.

QUESTION 2.6 Which current or past religions are mono-theistic?
This is given by the $Mon_{\text{Rels}}$ operation, which applies a cardinality functional to determine the number of gods believed in by the particular religion.

\[
Mon_{\text{Rels}} : (\text{Rel} \, I \, d \mapsto \mathcal{P} \, \text{God} \, I \, d) \mapsto \mathcal{P} \, \text{Rel} \, I \, d.
\]

\[
Mon_{\text{Rels}}[\beta] \triangleq \\
1 \in \text{rng}
\left(\mathcal{I} \mapsto ||\beta||\right)
\quad \mapsto ((\mathcal{I} \mapsto ||\beta||)^{-1}(1))
\quad \mapsto 0
\]

**Question 2.7** Which current or past religions are poly-theistic?

This is given by the $Pol_{\text{Rels}}$ operation, defined as follows:

\[
Pol_{\text{Rels}} : (\text{Rel} \, I \, d \mapsto \mathcal{P} \, \text{God} \, I \, d) \mapsto \mathcal{P} \, \text{Rel} \, I \, d.
\]

\[
Pol_{\text{Rels}}[\beta] \triangleq \\
\left\langle \left[\text{Ath}_{\text{Rels}}[\beta^{-1}]\right] \circ \left[\text{Mon}_{\text{Rel}}[\beta]\right] \text{dom} \beta \right\rangle
\]

**Question 2.8** What gods are currently or previously believed in?

This is given by $\cup / \circ \text{rng} \beta$.

**Question 2.9** What religions believe in $g$?

This is given by the $\text{Rels} \, Gd$ operation, and takes the form of converting $\beta$ from $\text{Rel} \, I \, d \mapsto \mathcal{P} \, \text{God} \, I \, d$ to $\beta_r : \text{Rel} \, I \, d \mapsto \mathcal{P}' \, \text{God} \, I \, d$; this may then be inverted to yield $\beta_r^{-1}$ of the form $\text{God} \, I \, d \mapsto \mathcal{P}' \, \text{Rel} \, I \, d$. The relational inversion algorithm is described in [47].

\[
\text{Rels} \, Gd : \text{God} \, I \, d \mapsto (\text{Rel} \, I \, d \mapsto \mathcal{P} \, \text{God} \, I \, d) \mapsto \mathcal{P} \, \text{Rel} \, I \, d
\]

\[
\text{Rels} \, Gd[\beta][g] \triangleq \\
\text{Let } \beta_r = \langle \left[\text{Ath}_{\text{Rels}}[\beta^{-1}]\right] \beta \text{ in}
\quad \mapsto \beta_r^{-1}(g)
\]

The precondition for this operation must ensure $\chi[\beta][\beta_r^{-1}]$.

**Notation 2.2** The notation $\mathcal{P}'X$ denotes the subsets of a set $X$ excluding the emptyset. For $\mu \in (X \mapsto \mathcal{P}'Y)$ we can form the inverse $\mu^{-1} \in (Y \mapsto \mathcal{P}'X)$ constructively as described in [47].
Overlapping and similar Religions

The aim of this section is to determine how effective the model is in identifying similarities in doctrine and belief for the various religions. It is quite usual for two different religions to exhibit some similarities in doctrine and deities. Two religions $r_1$ and $r_2$ may be interested in extent to which their beliefs overlap. There are several possibilities:

- $\mu(r_1) \cap \mu(r_2) = \emptyset$, then religions $r_1$ and $r_2$ have distinct doctrines, with no common ground between them.
- $\mu(r_1) \cap \mu(r_2) \neq \emptyset$ then religions $r_1$ and $r_2$ have common ground in their doctrines.
- $\mu(r_1) = \mu(r_2)$ then $r_1$ and $r_2$ have identical dogmas, however, they may differ on belief in deities.
- $\beta(r_1) \cap \beta(r_2) = \emptyset$ then religions $r_1$ and $r_2$ have no common deities.
- $\beta(r_1) \cap \beta(r_2) \neq \emptyset$ then religions $r_1$ and $r_2$ share belief in at least one common god.
- $\beta(r_1) = \beta(r_2)$ then religions $r_1$ and $r_2$ believe in exactly the same deities.
- $\mu(r_1) = \mu(r_2)$ and $\beta(r_1) = \beta(r_2)$ then $r_1$ and $r_2$ have identical religious beliefs, and identical gods, i.e., $r_1$ and $r_2$ (apart from a difference in name) are essentially identical.

2.2.3 Formation of New Religions

The objective of this section is to determine the effectiveness of the model for the formation of a new religion. The latter involves the assertion of the precise dogma and deities which the religion believes in. The religion may be mono-theistic, poly-theistic, agnostic, or atheist. The distinction between the various religions within the model has been previously explained.

$$ Crea\_Rel : (Rel_J \times \mathcal{P}God_J \times \mathcal{P}Bel_J) \mapsto (Rel_J \mapsto \mathcal{P}God_J \times (Rel_J \mapsto \mathcal{P}Bel_J) \times (Rel_J \mapsto \mathcal{P}Bel_J) $$

$$ Crea\_Rel[r, gs, bs] (\beta, \mu) \triangleq $$

- $bs = \emptyset$
- $\mapsto (\beta, \mu \cup [r \mapsto bs])$
- $\mapsto (\beta \cup [r \mapsto gs], \mu \cup [r \mapsto bs])$

The operation distinguishes between agnostic religions, and religions which take a definite position on deities. This is achieved by the distinction $bs = \emptyset$; the latter refers to
agnostic religions. The precondition must stipulate that the religion is not an existing religion.

\[
\text{pre}_{\text{Cre}a_{\text{Rel}}} : \text{Rel}_I \vdash (\text{Rel}_I \mapsto \mathcal{P}\text{God}_I) \times (\text{Rel}_I \mapsto \mathcal{P}\text{Bel}_I) \vdash B \\
\text{pre}_{\text{Cre}a_{\text{Rel}}}[r](\beta, \mu) \triangleq \\
\neg \chi[r] \beta \\
\land \neg \chi[r] \mu
\]

It has been remarked that religious dogma is subject to change and reformulation; this may take the form of the introduction of new deities for the religion, or an update of the particular doctrine of the religion. The \( \text{Upd}_{\text{Rel}} \) operation updates the religion accordingly:

\[
\text{Upd}_{\text{Rel}} : (\text{Rel}_I \times \mathcal{P}\text{God}_I \times \mathcal{P}\text{Bel}_I) \vdash (\text{Rel}_I \mapsto \mathcal{P}\text{God}_I) \times (\text{Rel}_I \mapsto \mathcal{P}\text{Bel}_I) \\
\text{Upd}_{\text{Rel}}[r, gs, bs](\beta, \mu) \triangleq \\
bs = \emptyset \\
\mapsto (\downarrow[r] \beta, \mu \uparrow [r \mapsto bs]) \\
\mapsto (\downarrow[r] \beta \sqcup [r \mapsto gs], \mu \uparrow [r \mapsto bs])
\]

**Notation 2.3** The effect of the \( \downarrow[S] \) operation applied to \( \beta \) is to remove all elements of \( S \) from the domain of \( \beta \).

There is a distinction between agnosticism and other religions as before. The precondition stipulates that the religion is an existing religion, i.e., it must appear in \( \beta \) and \( \mu \) if it is a non-agnostic religion, and just \( \mu \) if it is an agnostic religion.

\[
\text{pre}_{\text{Upd}_{\text{Rel}}} : \text{Rel}_I \vdash (\text{Rel}_I \mapsto \mathcal{P}\text{God}_I) \times (\text{Rel}_I \mapsto \mathcal{P}\text{Bel}_I) \vdash B \\
\text{pre}_{\text{Cre}a_{\text{Rel}}}[r](\beta, \mu) \triangleq \\
\chi[r] \beta \\
\lor \chi[r] \mu
\]

The formation of a religion may follow from some divine inspiration, or may be from some schism in an existing religion. In either case, for the religion to survive it must attract a religious community. Evidence of the popularity of a particular religion is demonstrated by the physical churches for preaching the dogma of the religion. The priests are responsible for preaching the beliefs and values of the religion. There is a need for the \( \text{Join}_{\text{Rel}} \) and \( \text{Chg}_{\text{Rel}} \) operations for joining or changing a particular religion.
Join/Change Religion Operations

The members of a new religion are drawn from those who are not previously members of some religion, or those who decide to change to the new religion. These operations are given by the Join _Rel _and Chg _Rel _operations.

\[\text{Join}_\text{Rel} : (\text{Per}_\text{Jd} \times \text{Rel}_\text{Jd}) \mapsto (\text{Per}_\text{Jd} \mapsto \text{Rel}_\text{Jd}) \mapsto (\text{Per}_\text{Jd} \mapsto \text{Rel}_\text{Jd})\]
\[\text{Join}_\text{Rel}[p, r] \alpha \triangleq \alpha \sqcup [p \mapsto r]\]

This operation requires that \(p\) is not currently a member of some religion; Secondly, the specified religion, i.e., religion \(r\) must be an existing religion. Finally, the person who wishes to join a religion must be living.

\[\pre_{\text{Join}_\text{Rel}} : (\text{Per}_\text{Jd} \times \text{Rel}_\text{Jd}) \mapsto ((\text{Per}_\text{Jd} \mapsto \text{Rel}_\text{Jd}) \times (\text{Per}_\text{Jd} \mapsto \text{B})) \mapsto \text{B}\]
\[\pre_{\text{Join}_\text{Rel}}[p, r](\alpha, \alpha_1) \triangleq \neg \chi[p] \alpha \land \chi[r] \mu \land \alpha_1(p)\]

The effect of the change religion operation, Chg _Rel _for person \(p\) is the removal of \(p\) from membership of \(p\)'s current religion, and recording \(p\) as a member of the new religion. If \(p\) is in fact a priest, then \(p\) is de-assigned from all churches that \(p\) is currently assigned to.

\[\text{Chg}_\text{Rel} : \text{Per}_\text{Jd} \times \text{Rel}_\text{Jd} \mapsto (\text{Per}_\text{Jd} \mapsto \text{Rel}_\text{Jd}) \times (\text{Church}_\text{Jd} \mapsto \mathcal{P}_\text{Priest}_\text{Jd}) \mapsto (\text{Per}_\text{Jd} \mapsto \text{Rel}_\text{Jd}) \times (\text{Church}_\text{Jd} \mapsto \mathcal{P}_\text{Priest}_\text{Jd})\]
\[\text{Chg}_\text{Rel}[p, r](\alpha, \delta) \triangleq (\alpha \uparrow [p \mapsto r], (\mathcal{I} \mapsto \mathcal{A}[p]) \delta)\]

The effect of the range deletion functional \((\mathcal{I} \mapsto \mathcal{A}[p])\) is to ensure that if \(p\) is a priest, then \(p\) is de-assigned from any currently assigned churches. The precondition must ensure that \(p\) is a member of some religion, and the new religion actually exists.

\textbf{Notation 2.4} The effect of the override operation \(\uparrow\) is analogous to the semantics of the assignment statement in imperative programming languages.

\[\pre_{\text{Chg}_\text{Rel}} : \text{Per}_\text{Jd} \times \text{Rel}_\text{Jd} \mapsto (\text{Per}_\text{Jd} \mapsto \text{Rel}_\text{Jd}) \times (\text{Church}_\text{Jd} \mapsto \mathcal{P}_\text{Priest}_\text{Jd}) \times (\text{Per}_\text{Jd} \mapsto \text{B}) \mapsto \text{B}\]
\[\pre_{\text{Chg}_\text{Rel}}[p, r](\alpha, \delta, \alpha_1) \triangleq \chi[p] \alpha \land \chi[r] \mu \land \chi[p] \alpha_1\]
2.2.4 The Churches of a Religion

The objective of this section is to determine the effectiveness of the modelling of churches of a particular religion. The church is a physical structure used by the priests for worship and for preaching to the congregation. A church is in a precise physical locations.

\[ \gamma : \text{Church}_d \mapsto \text{Rel}_d \] enforces the fact that a church is associated with at most one religion. Operations to assign a priest to a church are defined later. A new church or a church belonging to an extinct religion may have no assigned priests.

\[ Crea_{Chc} : (\text{Church}_d \times \text{Rel}_d) \mapsto (\text{Church}_d \mapsto \text{Rel}_d) \mapsto (\text{Church}_d \mapsto \text{Rel}_d) \]

\[ Crea_{Chc}[c, r] \gamma \triangleq \gamma \cup [c \mapsto r] \]

\[ \text{pre}_{\text{Crea}_{\text{Chc}}} : (\text{Church}_d \times \text{Rel}_d) \mapsto (\text{Church}_d \mapsto \text{Rel}_d) \times (\text{Rel}_d \mapsto \mathcal{P}\text{Bel}_d) \mapsto \mathcal{B} \]

\[ \text{pre}_{\text{Crea}_{\text{Chc}}}[c, r](\gamma, \mu) \triangleq \chi[c] \gamma \land \chi[r] \mu \]

**Question 2.10** What churches does religion \( r \) have?

This is obtained from \( \gamma^{-1} : \text{Rel}_d \mapsto \mathcal{P}\text{Church}_d \), and \( \gamma^{-1}(r) \) gives \( r \)'s churches providing \( r \) is an existing religion in the model. The number of churches of a particular existing religion is then given by \( |\gamma^{-1}(r)| \).

If it is required to restrict attendance by a person to one particular church, then the model may include \( \zeta : \text{Per}_d \mapsto \text{Church}_d \); if there are no restrictions, then \( \zeta : \text{Per}_d \mapsto \mathcal{P}\text{Church}_d \) indicates the churches attended by each person. Each church is associated with at most one religion, thus if \( p \) attends no church, then if \( p \) is a member of some religion \( r \), \( p \) may be regarded as a lapsed member of religion \( r \).

**Question 2.11** What priests are assigned to church \( c \)?

This relationship is modelled by \( \delta : \text{Church}_d \mapsto \mathcal{P}\text{Priest}_d \). A church may have several priests, and a priest may serve several churches. If the requirement is to enforce that a priest serves at most one church, then \( \delta \) is of the form \( \text{Priest}_d \mapsto \text{Church}_d \). The former definition of \( \delta \) is used in the model. The priests assigned to \( c \) is given by \( \delta(c) \), subject \( \chi[c] \delta \). The newly created or extinct churches are given by \( \delta^{-1}(\emptyset) \), subject to \( \chi[\emptyset] \delta^{-1} \).

**Priests and Churches**

It is clear that in order for a priest to be assigned to a church, the church must exist, and the priest’s religion must be that of the church.
\[
\text{pre}_{\text{Asgn Chc}} : (\text{Church Id} \times \text{Priest Id}) \leftrightarrow (\text{Church Id} \mapsto \mathcal{P} \text{Priest Id}) \times \\
(\text{Per Id} \mapsto \text{Rel Id}) \times (\text{Church Id} \mapsto \text{Rel Id}) \mapsto \mathcal{B}
\]

\[
\text{pre}_{\text{Asgn Chc}} [c, p] (\delta, \alpha, \gamma) \triangleq \\
\chi[c] \delta \land \chi[p] \gamma \land \alpha_1(p) \land \chi[p] \alpha \land \gamma(c) = \alpha(p)
\]

\[
\text{Asgn Chc} : (\text{Church Id} \times \text{Priest Id}) \leftrightarrow (\text{Church Id} \mapsto \mathcal{P} \text{Priest Id}) \times \\
(\text{Per Id} \mapsto \text{Rel Id}) \times (\text{Church Id} \mapsto \text{Rel Id}) \mapsto (\text{Church Id} \mapsto \mathcal{P} \text{Priest Id})
\]

\[
\text{Asgn Chc}[c, p] (\delta, \alpha, \gamma) \triangleq \delta \ominus [c \mapsto \{p\}]
\]

**Notation 2.5** The \( \ominus \) is an indexed monoid operation. The indexed operation \( \odot \) is inherited from the base monoid \( (M, \ast, u) \) yielding the structure \( (X \mapsto M', \odot, \theta) \).

**Question 2.12** What churches is \( p \) assigned to?

This is given by the \( \text{Chs Prt} \) operation defined as follow:

\[
\text{Chs Prt} : \text{Priest Id} \mapsto (\text{Church Id} \mapsto \mathcal{P} \text{Priest Id}) \mapsto \mathcal{P} \text{Church Id}
\]

\[
\text{Chs Prt}[p] \delta \triangleq \\
\text{Let } \delta_r = (\mathcal{I} \mapsto \chi[p]) \delta \text{ in } \\
\chi[\text{TRUE}] \delta_r^{-1} = \\
\mapsto \delta_r^{-1}(\text{TRUE}) = \\
\mapsto \emptyset
\]

**Revisiting the invariant**

The study of the above operations has enabled a more detailed knowledge of the problem domain to be gained. For example, all priests must have the same religion as the church to which they have been assigned, and secondly, each time a church is built it must have a collection (possibly empty) of assigned priests. The following constraints are added to the invariant:

\[
\text{dom } \delta = \text{dom } \gamma \\
\land (\mathcal{I} \mapsto \mathcal{P} \alpha) \triangleq [S] \delta = [S] \gamma \\
\text{where } S = \neg [\delta^{-1}(\emptyset)] \text{dom } \delta
\]

The \( \text{Create Chc} \) operation did not enforce these constraints, and is updated accordingly:

\[
\text{Create Chc} : (\text{Church Id} \times \text{Rel Id}) \mapsto (\text{Church Id} \mapsto \text{Rel Id}) \times (\text{Church Id} \mapsto \mathcal{P} \text{Priest Id}) \\
\mapsto (\text{Church Id} \mapsto \text{Rel Id}) \times (\text{Church Id} \mapsto \mathcal{P} \text{Priest Id})
\]

\[
\text{Create Chc}[c, r] (\gamma, \delta) \triangleq (\gamma \sqcup [c \mapsto r], \delta \sqcup [c \mapsto \emptyset])
\]
\( \text{pre}_{\text{Createchurch}} : (\text{Church}_\mathcal{Id} \times \text{Rel}_\mathcal{Id}) \mapsto (\text{Church}_\mathcal{Id} \mapsto \text{Rel}_\mathcal{Id}) \times \\
(\text{Rel}_\mathcal{Id} \mapsto \mathcal{P}\text{Bel}_\mathcal{Id}) \mapsto B \\
\text{pre}_{\text{Createchurch}}[c, r](\gamma, \mu) \triangleq \neg \chi[c] \gamma \land \neg \chi[r] \delta \land \chi[r] \mu \)

**Church Locations**

A church is a physical structure in a specific geographical location. The location of a church is given by \( \epsilon : \text{Church}_\mathcal{Id} \mapsto \text{Loc}_\mathcal{Id} \). The \( \text{Createchurch} \) operation and the invariant must be updated accordingly. The location of church \( c \) is given by \( \epsilon(c) \), subject to \( \chi[c] \epsilon \). The updates to the invariant are as follow:

\[
\text{dom } \delta = \text{dom } \gamma = \text{dom } \epsilon \\
\land (I \mapsto \mathcal{P}\alpha) \ll [S] \delta = [S] \gamma \quad \text{where } S = \ll [\delta^{-1}(\emptyset)] \text{dom } \delta
\]

**Question 2.13** What churches does city (or location) \( l \) have?

This is given by the inverse image \( \epsilon^{-1} : \text{Loc}_\mathcal{Id} \mapsto \mathcal{P}\text{Church}_\mathcal{Id} \), and is \( \epsilon^{-1}(l) \), subject to \( \chi[l] \epsilon^{-1} \).

**Question 2.14** What churches does religion \( r \) have in location \( l \)?

This question is answered in two stages:

- The religion of each church \( c \) in location \( l \) is determined and given by \( \gamma(c) \).
- The list of churches is filtered to yield those which belong to religion \( r \) in location \( l \), as required.

\( \text{Getchurches} : (\text{Loc}_\mathcal{Id} \times \text{Rel}_\mathcal{Id}) \mapsto (\text{Church}_\mathcal{Id} \mapsto \text{Rel}_\mathcal{Id}) \times (\text{Loc}_\mathcal{Id} \mapsto \mathcal{P}\text{Church}_\mathcal{Id}) \mapsto \\
\mathcal{P}\text{Church}_\mathcal{Id} \\
\text{Getchurches}[l, r](\gamma, \epsilon^{-1}) \\
\mapsto [\gamma^{-1}(r)] \epsilon^{-1}(l) \)

The precondition for the \( \text{Getchurches} \) operation must ensure that location \( l \) is a valid location in the model, and each church specified at location \( l \) is an existing church.

\( \text{pre}_{\text{Getchurches}} : (\text{Loc}_\mathcal{Id} \times \text{Rel}_\mathcal{Id}) \mapsto (\text{Church}_\mathcal{Id} \mapsto \text{Rel}_\mathcal{Id}) \times \\
(\text{Loc}_\mathcal{Id} \mapsto \mathcal{P}\text{Church}_\mathcal{Id}) \mapsto B \\
\text{pre}_{\text{Getchurches}}[l, r](\gamma, \epsilon^{-1}) \\
\quad \quad \quad \text{dom } \epsilon \subseteq \text{dom } \gamma \land \chi[l] \epsilon^{-1} \)

**Question 2.15** Are there any churches of religion \( r \) in location \( l \)?

30
This question is given by the predicate $Is\_Chs$ defined by:

$$Get\_Chs[l, r](\gamma, \epsilon^{-1}) \neq \emptyset$$

### 2.2.5 Heresy and Heretics

The objective of this section is to determine if heresy and heretics are captured in the model.

**Question 2.16** *How are heresy and heretics identified in the model?*

The word heretic is derived from the Greek ‘herias’, αἱρεσία meaning ‘to choose’ or ‘choice’ or School of thought, and reflects the philosophical approach of choosing the beliefs which are appropriate for the particular view held on religion. This original meaning of the word heresy is addressed in the model by the $Crea\_Rel$ operation, which enables new religions to be created. This enables a religion with exactly one member to be created, with exactly the religious beliefs of the particular individual.

The modern meaning of heresy is to describe a person who is a member of some particular religion, usually the Catholic Church, who holds religious beliefs which are incompatible with the religion. The Concise Oxford defines heresy as a ‘belief or practice contrary to the orthodox doctrine of the Christian Church’. In order to address the modern meaning of heresy in the model it is necessary to extend the model to include personal religious beliefs, as distinct from the beliefs of the particular religion. Heresy is then viewed as a deviation between an individual’s personal beliefs and the dogma of the religion. The model facilitates personal beliefs via the extension $\nu : Per\_Id \mapsto P\_Bel\_Id$.

The established religions have typically regarded heretics and heresy with deep suspicion. For this reason, it is important for a religion to identify heretics, thus preventing potential schisms from taking place. This is achieved by the inquisitional like operation $Is\_Herc$, which judges a person $p$ to be a heretic, if $p$’s personal religious beliefs fail to be a superset of the beliefs of the religion.

$$Is\_Herc : Per\_Id \mapsto (Per\_Id \mapsto P\_Bel\_Id) \times (Rel\_Id \mapsto P\_Bel\_Id) \times (Per\_Id \mapsto Rel\_Id) \mapsto B$$

$$Is\_Herc[p](\nu, \mu, \alpha) \triangleq$$

Let $r = \alpha(p)$ in

- $\mu(r) \subseteq \nu(p)$

  $\mapsto \text{FALSE}$

  $\mapsto \text{TRUE}$
the precondition for the $I_{\text{Her}}c$ operation requires that $p$ and $p$’s religion exist:

\[
\text{pre}_{I_{\text{Her}}c} : \text{Per} \ J d \times (\text{Per} \ J d \leftrightarrow \mathcal{P} \text{Bel} \ J d) \times (\text{Rel} \ J d \leftrightarrow \mathcal{P} \text{Bel} \ J d) \times \\
(\text{Per} \ J d \leftrightarrow \text{Rel} \ J d) \mapsto B
\]

\[
\text{pre}_{I_{\text{Her}}c}[p](\nu, \mu, \alpha) \triangleq \\
\neg \chi[p, \alpha] \lor \neg \chi[p, \nu] \\
\mapsto \text{FALSE}
\]

Let \( r = \alpha(p) \) in

\[
\chi[r, \mu] \\
\mapsto \text{TRUE}
\]

\[
\mapsto \text{FALSE}
\]

The Invariant - final form

It has been noted that the modelling and invariant evolve as a deeper understanding is gained of the problem domain. The model commenced with the examination of religion and gods, and the invariant was formulated at that level. The study proceeded to examine religions and beliefs, churches and priests, etc. Further structure and constraints were placed on the model as a more in depth understanding of the religion domain was gained.

The final form of the invariant is stated without comments below:

\[
\text{Inv}_{\text{Rel}}[\alpha, \alpha_1, \beta, \gamma, \delta, \epsilon, \mu] \triangleq \\
\text{dom} \alpha \subseteq \text{dom} \alpha_1 \\
\land \text{rng} \alpha \subseteq \text{dom} \mu \\
\land (\text{dom} \beta = \text{dom} \mu) \lor \text{dom} \mu = \text{dom} \beta \cup \mu^{-1}(\emptyset) \\
\land \text{dom} \delta = \text{dom} \gamma = \text{dom} \epsilon \\
\land (I \mapsto \mathcal{P} \alpha) \triangleleft [S] \delta = \alpha \| S \| \gamma \\
\text{where } S = \text{dom} \delta^{-1}(\emptyset) \cap \text{dom} \delta
\]

2.2.6 Conclusions

The basic model of religion is an adequate representation of aspects of the structure of religion. The model serves as a mechanism for formally encoding properties of religion, and properties of religion may be deduced via model interrogation. The modelling enabled a detailed understanding of the domain of religion to be gained. Further desirable properties became evident from the exploration of the model. This has led to the identification of several possible extensions to the basic model.

It is evident that many elementary properties of religion may be immediately deduced from the model; for example, the model is good at distinguishing between monotheistic
and polytheistic religions. Distinctions between agnosticism, atheism and theism are clear from the model. The model addresses aspects of the structure of the individual religions including the fact that each religion has an associated set of theological beliefs, including the belief in a set of deities. The model captures the fact that in general, an organized religion has churches where worship is performed by priests.

The model does not capture conscience or free will. Furthermore, the explanation of the modern meaning of heresy is slightly naive in the model. The model does not consider the issue of whether beliefs are consistent; in fact, the exact definition of a belief is not made in the model.

The naming problem with gods is not considered in the model. In Greek mythology, the god Poseidon, is the god of the sea; in Roman mythology, Neptune is the corresponding god. The model may be extended such that each \( g \in God_{Jd} \) has an associated set of names, i.e., \( \xi : God_{Jd} \rightarrow \mathcal{P}God_{Name} \). Finally, the model does not provide a mechanism to prove or disprove the existence of God.

### 2.3 Specific Religions

This objective of this section is to model specific religions of the world. The study considers several of the major religions, including Judaism, Catholicism, Eastern Orthodox, Protestantism, Islam, and Hinduism.

The study commences with an examination of Judaism, an ancient religion, and parent of Christianity. The Christian religions are then examined, then Islam. Finally Hinduism is studied. The focus of the examination takes the form of studying several important beliefs of customs in the religion. This includes modelling marriage, sin, absolution, etc.

#### 2.3.1 Judaism

Judaism is the parent of Christianity with Yahweh the God of Israel in the Old Testament. While pre-Christian Judaism is associated exclusively with the Old Testament, modern Judaism (cf., Chapter 11 of [31]) is derived from the synod of Jamnia circa 100 A.D. This synod cited four criteria for determining authenticity of Jewish scripts; this included conformity with the Pentateuch viz Genesis, Exodus, Leviticus, Numbers and Deuteronomy, not written after the time of Esdras, written in Hebrew and written in Palestine. These last two conditions rule out texts written in Aramaic and during the Jewish Diaspora. Thus the Septuagint dating from the third century B.C. was replaced by the Hebrew Bible. The Talmud reflects modern Jewish civil and ceremonial laws but stresses the importance of studying the Torah, i.e., Mosaic Law.
Fundamental in Judaism is the belief in the coming of a future messiah; such a messiah (cf., Page 256 of [31]) being a wise but mortal human being who will restore the Jewish people to their godly inheritance and unite all people in allegiance to the God of Israel. The covenant by God binds the Jewish people, i.e., the Chosen people to keep his commandments; obedience and good conduct will be rewarded by an afterlife of happiness i.e., Eden; otherwise an afterlife of misery, i.e., Gehenna will be the reward. Prayer and observance of the Sabbath both at home and by attending the synagogue are important for the devout Jew as are the various festivals, for example Rosh Hashanah, Yom Kippur, Passover, Purim (feast of Lots), etc. The God of Israel is strictly mono-theistic and Judaism flatly rejects the Christian concept of the Trinity.

The objective of this section is to demonstrate that aspects of Judaic marriages may be modelled formally. Judaic marriages take place between persons of opposite gender, with polygamy strictly forbidden on pain of excommunication. The two genders are opposites and the concept of an inverse alphabet of a free group as discussed in Appendix C is used here. The model for Judaic marriage is the following:

$$Gen = \{F, F\}$$

$$\alpha_1 : Per.Jd \mapsto Per.Jd$$

$$\alpha_2 : Per.Jd \mapsto Gen$$

$$\alpha_3 : Per.Jd \mapsto B$$

$$\alpha_2$$ models the gender relation; and the free set concept discussed in Appendix C is used here, thus $$F$$ represents the female gender, and $$\bar{F}$$ represents the male gender; in this way, the gender requirements for a valid Judaic marriage between $$p_1$$ and $$p_2$$ are satisfied if $$\{\alpha_2(p_1)\} \cup_s \{\alpha_2(p_2)\} = \emptyset$$. The notation $$\cup_s$$ is not part of the notation of the Irish school, and is described in Appendix C. $$\alpha_3$$ exhibits the relationship between living or non-living, $$\alpha_3^{-1}(\text{TRUE})$$ gives all those who are currently alive. Finally, conditions must be placed on $$\alpha_1$$ as polygamy is forbidden. The initial invariant for Judaic marriage ($$\text{Kiddushin}$$) is formulated as follows:

$$\text{Inv}_\text{JudMar} : (Per.Jd \mapsto Per.Jd) \times (Per.Jd \mapsto Gen) \times (Per.Jd \mapsto B) \mapsto B$$

$$\text{Inv}_\text{JudMar}[\alpha_1, \alpha_2, \alpha_3] \triangleq$$

$$\text{dom} \alpha_2 = \text{dom} \alpha_3 \land \text{dom} \alpha_1 \subseteq \text{dom} \alpha_2$$

$$\land \forall p \in \text{dom} \alpha_1$$

$$\{\alpha_2(p)\} \cup_s \{\alpha_2(\alpha_1(p))\} = \emptyset$$

$$\land \forall p_1, p_2 \in \text{dom} \alpha_1$$

34
\( (\alpha_1(p_1) = \alpha_1(p_2)) \Rightarrow p_1 = p_2 \)
\( \forall p \in \text{dom} \alpha_1 \)
\( \alpha_1(p) \in \text{dom} \alpha_1 \land \alpha_1^2(p) = p \)

**Notation 2.6** The notation \( \cup_\ast \) refers to the union of free sets operation, a structure identified in the study of free groups in Appendix C.

The invariant ensures that a marriage is composed of two people of opposite genders and the relationship is symmetric, i.e., if \( b \) is married to \( g \) then \( g \) is married to \( b \). Furthermore, polygamy is prohibited. The following elementary properties may be immediately derived from the invariant.

**Lemma 2.1** \( \text{rng} \alpha_1 = \text{dom} \alpha_1 \)

**Proof**
This follows from the last part of the invariant which gives us that \( \text{rng} \alpha_1 \subseteq \text{dom} \alpha_1 \). Furthermore, the idempotent property for \( \alpha_1 \) gives us that given \( p \in \alpha_1 \) then \( \alpha_1^2(p) = p \), thus given \( p \in \alpha_1 \) it follows that \( \exists q \in \text{rng} \alpha_1 \) such that \( \alpha_1(q) = p \). Thus \( \text{dom} \alpha_1 \subseteq \text{rng} \alpha_1 \) and thus \( \text{dom} \alpha_1 = \text{rng} \alpha_1 \).

**Lemma 2.2** Given \( p \in \alpha_1 \) then \( \alpha_1(p) \neq p \). That is, nobody may be married to themselves, and so \( \alpha_1 \) when considered as a relation is irreflexive.

**Proof**
This is immediate from the invariant; however, let us suppose there is a person \( p \) such that \( \alpha_1(p) = p \), then we have \( \alpha_2(p) = \alpha_2(\alpha_1(p)) \) and thus \( \{ \alpha_2(p) \} \cup_\ast \{ \alpha_2(\alpha_1(p)) \} \neq \emptyset \) by definition of \( \cup_\ast \). This contravenes the invariant and thus there is no such \( p \).

**Lemma 2.3** When \( \alpha_1 \) is viewed as a relation it is in fact a symmetric relation, i.e., \( \alpha_1(p) = q \) then \( \alpha_1(q) = p \).

**Proof**
We have \( \text{dom} \alpha_1 \subseteq \text{dom} \alpha_2 \) and so given \( p \in \text{dom} \alpha_1 \) with \( \alpha_1(p) = q \) we have \( q \in \text{dom} \alpha_1 \). Thus we must show \( \alpha_1(q) = p \). However, \( \alpha_1(q) = \alpha_1(\alpha_1(p)) = \alpha_1^2(p) = p \) by the idempotent constraint on the invariant and so the result follows.

Next we proceed to the definition of the \( \text{Jud}_{\text{Mrg}} \) operator, an operation which takes two people of the Jewish faith and joins them in matrimony.
\[ \text{Jud}_\text{Mr}g : (\text{Per}_J d \times \text{Per}_J d) \mapsto (\text{Per}_J d \mapsto \text{Per}_J d) \mapsto (\text{Per}_J d \mapsto \text{Per}_J d) \]
\[ \text{Jud}_\text{Mr}g[p_1, p_2] \alpha_1 \triangleq \alpha_1 \cup [p_1 \mapsto p_2, p_2 \mapsto p_1] \]

the precondition must ensure that the parties who plan to marry are of different genders and not previously married. Furthermore both parties to the marriage must be living.

\[ \text{pre}_\text{Jud}_\text{Mr}g : (\text{Per}_J d \times \text{Per}_J d) \mapsto (\text{Per}_J d \mapsto \text{Per}_J d) \times (\text{Per}_J d \mapsto \text{Gen}) \times (\text{Per}_J d \mapsto \text{B}) \mapsto \text{B} \]
\[ \text{pre}_\text{Jud}_\text{Mr}g[p_1, p_2](\alpha_1, \alpha_2, \alpha_3) \triangleq \]
\[ p_1 \in \alpha_2 \land p_2 \in \alpha_2 \]
\[ \land \{\alpha_2(p_1)\} \cup \{\alpha_2(p_2)\} = \emptyset \]
\[ \land p_1 \not\in \alpha_1 \land p_2 \not\in \alpha_1 \]
\[ \land \alpha_3(p_1) \land \alpha_3(p_2) \]

Given the \( \text{Jud}_\text{Mr}g \) we must prove that the operation preserves the invariant, we state and prove this as a theorem.

**Theorem 2.1** \( \text{pre}_\text{Jud}_\text{Mr}g[p_1, p_2](\alpha_1, \alpha_2, \alpha_3) \land \alpha' = \text{Jud}_\text{Mr}g[p_1, p_2] \alpha_1 \Rightarrow \text{Inv}_\text{Mar}[\alpha'_1, \alpha_2, \alpha_3] \)

**Proof**

Clearly \( \text{dom} \alpha'_1 \subseteq \text{dom} \alpha_2 \). In order to show that \( \forall p \in \alpha'_1 \{\alpha_2(p)\} \cup \{\alpha_2(\alpha'_1(p))\} = \emptyset \) is equivalent to demanding that \( \{\alpha_2(p_1)\} \cup \{\alpha_2(p_2)\} = \emptyset \) which is given by the precondition.

To show that \( \forall q_1, q_2 \in \alpha'_1 \) whenever \( \alpha'_1(q_1) = \alpha'_1(q_2) \) then \( q_1 = q_2 \). If \( q_1, q_2 \in \text{dom} \alpha_1 \) then the property is inherited since \( \alpha_1 \) satisfies the invariant. Suppose \( q_1 \in \alpha_1, q_2 \in \{p_1, p_2\} \) then since \( p_1, p_2 \not\in \alpha_1 \) and \( \alpha'_1(q) = \alpha_1(q) \) for all \( q \in \text{dom} \alpha_1 \). Thus \( \alpha'_1(q_1) \in \text{dom} \alpha_1 \) and \( \alpha'_1(q_2) \in \{p_1, p_2\} \). Then as these sets are disjoint it follows that \( \alpha'_1(q_1) \neq \alpha'_1(q_2) \). The final case is \( q_1 = p_1 \) and \( q_2 = p_2 \). Then since \( \alpha'_1(q_1) = q_2 \) and \( \alpha'_1(q_2) = q_1 \) this case does not arise.

Finally to show the last part of the invariant is preserved we are obliged to show that \( \forall p \in \alpha'_1 \) then \( \alpha'_1(p) \in \alpha'_1 \) and \( \alpha'_1^2(p) = p \). If \( p \in \alpha_1 \) then this property is inherited as \( \alpha_1 \) satisfies the invariant. Thus we must show that for \( p \in \{p_1, p_2\} \) the property holds. But this is immediate from the definition of \( \alpha'_1 \).

**Comment 2.2** We note that our model does not fully reflect marriage in Judaism since it is quite legitimate to re-marry if a spouse dies; furthermore divorce is quite legitimate in Judaism, with the marriage contract (ketubah) stipulating the support the wife will receive from the husband if the marriage terminates in a divorce. Thus the divorce operation must be specified, and the invariant updated accordingly.
The effect of the divorce operation will be to remove any trace of the marriage that existed between the two partners. The effect of a remarriage by the partner of the deceased is treated differently; in this case we still wish a record of the marriage to be maintained; in addition the new marriage must be recorded. However, the model must carefully distinguish this remarriage from polygamy which can only take place if the spouses are living. The approach taken with a remarriage is to remove the symmetric property of the marriage relation. Thus if \((p, q)\) are recorded as married and \(p\) is deceased and \(q\) decides to marry \(r\) then the effect of the \textit{Jud\_Mrg} operation will be to remove the expression \([q \mapsto p]\) which indicates that \(q\) is married to \(p\). However the relationship \([p \mapsto q]\) which indicates that \(p\) is married to \(q\) remains. The marriage relationship \([q \mapsto r, r \mapsto q]\) is then updated. In the case of \(p\) divorcing \(q\) or vice versa the effect is the removal of \([p \mapsto q, q \mapsto p]\) from the marriage relation. However we note the marriage relation is still a function.

\[
\text{Inv\_JudMar} : (\text{Per\_Id} \mapsto \text{Per\_Id}) \times (\text{Per\_Id} \mapsto \text{Gen}) \times (\text{Per\_Id} \mapsto \text{B}) \mapsto \text{B}
\]

\[
\text{Inv\_JudMar}[\alpha_1, \alpha_2, \alpha_3] \triangleq
\begin{align*}
\text{dom } \alpha_2 &= \text{dom } \alpha_3 \land \text{dom } \alpha_1 \subseteq \text{dom } \alpha_2 \\
\forall p \in \text{dom } \alpha_1 &\quad \{\alpha_2(p)\} \cup \{\alpha_2(\alpha_1(p))\} = \emptyset \\
\forall p_1, p_2 \in \text{dom } \alpha_1 &\quad \alpha_3(p_1) \land \alpha_3(p_2) \land (\alpha_1(p_1) = \alpha_1(p_2)) \Rightarrow p_1 = p_2 \\
\forall p \in \text{dom } \alpha_1 &\quad \alpha_1(p) \in \text{dom } \alpha_1 \land (\alpha_3(p) \Rightarrow \alpha_3^2(p) = p)
\end{align*}
\]

This invariant is slightly more involved, the main difference being that for every person \(p\) recorded as being in the married state to \(q\) say, if two living people \(p_1, p_2\) have the same spouse \(q\), then \(p_1\) is identical to \(p_2\). Furthermore, \(q\) is recorded as married to \(p\). The following elementary properties may be derived.

**Lemma 2.4** \(\text{rng } \alpha_1 \subseteq \text{dom } \alpha_1\)

This follows from the last part of the invariant. Equality may fail to hold which may be seen by considering the example \(b_1, b_2\) and \(g\) where \(b_1\) who is now deceased was previously married to \(g\) who has now re-married \(b_2\). This is represented by \(\alpha_1 = [b_1 \mapsto g, b_2 \mapsto g, g \mapsto b_2]\). In this case \(\text{rng } \alpha_1 = \{g, b_2\}\) whereas \(\text{dom } \alpha_1 = \{b_1, b_2, g\}\) and equality fails to hold.

**Question 2.17** Who is the present spouse of person \(p\)? Who are the deceased spouses of person \(p\)?

The present spouse is given by \(\alpha_1(p)\) subject to \(p \in \alpha_1\). The previous spouses are given by \(\pm \alpha_1(p) \circ \alpha_1^{-1}(p)\) subject to \(p \in \alpha_1^{-1}\).
The divorce operation of $p_1, p_2$ is obtained by removing the marriage relationship between $p_1$ and $p_2$. It is specified as follows:

$$
\text{Jud\_Div} : (\text{Per\_Jd} \times \text{Per\_Jd}) \mapsto (\text{Per\_Jd} \mapsto \text{Per\_Jd}) \mapsto (\text{Per\_Jd} \mapsto \text{Per\_Jd})
$$

$$
\text{Jud\_Div} [p_1, p_2] \alpha_1 \triangleq \{ [p_1, p_2] \} \alpha_1
$$

The precondition for the $\text{Jud\_Div} [p_1, p_2]$ operation must ensure that $p_1$ and $p_2$ are actually recorded as married; furthermore both spouses must be living.

$$
\text{pre}_\text{Jud\_Div} : (\text{Per\_Jd} \times \text{Per\_Jd}) \mapsto (\text{Per\_Jd} \mapsto \text{Per\_Jd}) \mapsto \text{B}
$$

$$
\text{pre}_\text{Jud\_Div} [p_1, p_2] \alpha_1 \triangleq
\begin{align*}
& p_1 \in \alpha_1 \wedge p_2 \in \alpha_1 \\
& \wedge \alpha_3(p_1) \wedge \alpha_3(p_2) \\
& \wedge \alpha_1(p_1) = p_2 \wedge \alpha_1(p_2) = p_1
\end{align*}
$$

Given the $\text{Jud\_Div}$ operation we have a corresponding proof obligation which we state as the following lemma.

**Lemma 2.5** $\text{pre}_\text{Jud\_Div} [p_1, p_2] \alpha_1 \wedge \alpha' = \text{Jud\_Div} [p_1, p_2] \alpha_1 \Rightarrow \text{Invl\_JudMar} [\alpha'_1, \alpha_2, \alpha_3]$

The $\text{Jud\_Mrg}$ operation is similar to before, the difference being that both parties to the marriage may both be recorded as married with deceased spouses. We specify the operation as follows:

$$
\text{Jud\_Mrg} : (\text{Per\_Jd} \times \text{Per\_Jd}) \mapsto (\text{Per\_Jd} \mapsto \text{Per\_Jd}) \mapsto (\text{Per\_Jd} \mapsto \text{Per\_Jd})
$$

$$
\text{Jud\_Mrg} [p_1, p_2] \alpha_1 \triangleq
\begin{align*}
& p_1 \not\in \alpha_1, p_2 \not\in \alpha_1 \mapsto \alpha_1 \cup [p_1 \mapsto p_2, p_2 \mapsto p_1] \\
& p_1 \in \alpha_1, p_2 \not\in \alpha_1 \mapsto (\alpha_1 \cup [p_1 \mapsto p_2]) \cup [p_2 \mapsto p_1] \\
& p_1 \not\in \alpha_1, p_2 \in \alpha_1 \mapsto (\alpha_1 \cup [p_1 \mapsto p_2]) \cup [p_2 \mapsto p_1] \\
& p_1 \in \alpha_1, p_2 \in \alpha_1 \mapsto (\alpha_1 \cup [p_1 \mapsto p_2, p_2 \mapsto p_1])
\end{align*}
$$

The precondition must ensure that if $p_1$ or $p_2$ are recorded as married, then their spouses are deceased, in addition to the normal constraints for marriage.

$$
\text{pre}_\text{Jud\_Mrg} : (\text{Per\_Jd} \times \text{Per\_Jd}) \mapsto (\text{Per\_Jd} \mapsto \text{Per\_Jd}) \times (\text{Per\_Jd} \mapsto \text{Gen}) \times \\
(\text{Per\_Jd} \mapsto \text{B}) \mapsto \text{B}
$$

$$
\text{pre}_\text{Jud\_Mrg} [p_1, p_2] (\alpha_1, \alpha_2, \alpha_3) \triangleq
\begin{align*}
& p_1 \in \alpha_2 \wedge p_2 \in \alpha_2 \\
& \wedge \{ \alpha_2(p_1) \} \cup \{ \alpha_2(p_2) \} = \emptyset
\end{align*}
$$

38
$$\wedge \alpha_3(p_1) \wedge \alpha_3(p_2)
\wedge (p_1 \in \alpha_1 \Rightarrow \neg \alpha_3(\alpha_1(p_1))) \wedge (p_2 \in \alpha_1 \Rightarrow \neg \alpha_3(\alpha_1(p_2)))$$

Given the $\text{Jud}_Mrg$ operation we have a corresponding proof obligation which we state as the following lemma.

**Lemma 2.6** $\text{pre}_Mrg[p_1,p_2] \alpha_1 \wedge \alpha' = \text{Jud}_Mrg[p_1,p_2] \alpha_1 \Rightarrow \text{Inv}_Mrg[\alpha_1', \alpha_2, \alpha_3]$

### 2.3.2 The Christian Religions

Christianity derives from Judaism a religion with divinely inspired prophets who predicted the coming of a messiah; Christians believe that Christ was the predicted messiah who was God in human form, i.e., incarnate and later crucified and rose from the dead. Christians take solace in the resurrection of Christ, believing that if they are followers of the teaching of Christ and believe in Him, then death itself has no victory as they too will be resurrected. The Christian concept of God is centred on the concept of the Trinity and this dogma is firmly stated in the Nicean Creed. Elements of this creed [31] have been controversial between the eastern wing of Catholicism, i.e., the Eastern Orthodox, and the western wing, i.e., the apostolic church in Rome. The eastern and western wings finally separated in 1204 A.D. following the sacking of Constantinople by the Crusaders; however the schism was apparent for at least 200 years before official separation. A good account of the rise of Christianity is provided in [27].

The sacrament of Baptism is used to initiate people into the Catholic faith and also to remit *original sin*. The sacrament of penance in Catholicism involves a confession of sins committed followed by absolution for the sins committed. Eastern orthodox perform this ceremony with the confessor facing east. The sacrament of Eucharist (*or sacrifice of Eucharist*) has various interpretations by the Christian churches with Lutherans believing in consubstantiation but not transubstantiation, whereas Catholics believing in transubstantiation. The Catholic Church and Eastern Orthodox express devotion to the saints and to the Blessed Virgin; Protestants reject this as idolatry and contrary to the first commandment. The Eastern Orthodox rejects the Immaculate Conception.

In view of the fact that the final destination for a Christian is determined by the life lead on earth it is important to represent the nature of *sin* and absolution for sins committed. Since Christians believe in a one to one correspondence between humans and souls, i.e., every Christian has exactly one soul, we shall model the soul and its location, i.e., limbo, hell, etc.

$$\text{Loc} = \text{Hell} | \text{Heaven} | \text{Earth} | \text{Limbo} | \text{Purgatory}$$

39
\[
\begin{align*}
\beta_1 : \text{Per} &\rightarrow (\text{Sin} \rightarrow N_1) \\
\beta_2 : \text{Per} &\rightarrow \text{Soul} \\
\beta_3 : \text{Per} &\rightarrow B \\
\beta_4 : \text{Soul} &\rightarrow (\text{Sin} \rightarrow N_1) \\
\beta_5 : \text{Sin} &\rightarrow Q^+ \\
\beta_6 : \text{Soul} &\rightarrow \text{Loc}
\end{align*}
\] (2.11)

(2.12)

(2.13)

(2.14)

(2.15)

(2.16)

**Notation 2.7** The notation \( Q^+ \) represents the positive rationals. It is not part of the notation of the Irish school of VDM.

Each sin committed has a certain weight associated with it, indicating the gravity of the sin; this relationship is captured by \( \beta_5 \). Each person has an associated list of sins which dynamically changes depending on frequency of confessions. Even though absolution for sins confessed is obtained by penance, we shall associate an accumulated sin list with each soul. In this way we have a measure as to how good a person this soul was. This relationship is captured by \( \beta_4 \). \( \beta_6 \) indicates where the soul is; if the person is of this world the location is the planet earth; otherwise the location may be heaven or hell, etc. The invariant is as follows:

\[
\begin{align*}
\text{Inv}_{\text{Soul}}[\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6] &\triangleq \\
\text{dom} \beta_1 = \text{dom} \beta_2 = \text{dom} \beta_3 \\
\wedge \text{rng} \beta_2 = \text{dom} \beta_4 = \text{dom} \beta_6 \\
\wedge (\beta_2(p_1) = \beta_2(p_2)) \Rightarrow p_1 = p_2 \\
\forall p \in \text{dom} \beta_3 \ (\beta_3(p) \Rightarrow \beta_6(\beta_2(p)) = \text{Earth})
\end{align*}
\]

The invariant ensures that each person has exactly one soul, and if the person is living then the soul is located on earth. Next we specify the \( \text{Conf} \) operation; this operation enables the confessant to obtain absolution for sins committed.

\[
\text{Conf} : (\text{Per} \times \text{Sin}) \rightarrow (\text{Per} \rightarrow (\text{Sin} \rightarrow N_1)) \times (\text{Soul} \rightarrow (\text{Sin} \rightarrow N_1)) \rightarrow (\text{Per} \rightarrow (\text{Sin} \rightarrow N_1)) \times (\text{Soul} \rightarrow (\text{Sin} \rightarrow N_1))
\]

Let \( \kappa_1 = \beta_1(p), \kappa_4 = \beta_4(\beta_2(p)) \) in

\[
\begin{align*}
\kappa_1' &\mapsto \llbracket S \rrbracket \kappa_1 \\
\kappa_4' &\mapsto \kappa_4 \circ \llbracket S \rrbracket \kappa_1 \\
(\beta_1 \upharpoonright [p \mapsto \kappa_1], \beta_4 \upharpoonright [p \mapsto \kappa_4'])
\end{align*}
\]

40
The precondition for the \( Conf \) operator stipulates that the person whose sins are to be confessed is living. Furthermore, the confessant is restricted to confessing only those sins which have been committed.

\[
pre_Conf : (Per J d \times P Sin J d) \mapsto (Per J d \mapsto (Sin J d \mapsto N_1)) \times (Soul J d \mapsto (Sin J d \mapsto N_1)) \mapsto B
\]

\[
pre_Conf [p, S] (\beta_1, \beta_4) \triangleq
p \in \beta_1 \land \beta_3(p)
\land S \subseteq \text{dom} \beta_1(p)
\]

The associated proof obligation of invariant preservation is stated in the following lemma.

**Lemma 2.7** \( pre_Conf [p, S] (\beta_1, \beta_4) \land Conf [p, S] \Rightarrow Inv_Soul [\beta_1', \beta_2, \beta_3', \beta_4', \beta_5, \beta_6] \) where \( \beta_1', \beta_4' \) are as defined by the \( Conf \) operator.

**Proof**

Examination of the invariant and the \( Conf \) operation shows that the theorem is clearly true, and no proof is required.

Next we present the \( Judge \) operation, an operation which determines whether a newly deceased will enter heaven, hell, purgatory, etc. We shall assume that there are constants \( M_p, M_h \) which reflect the point system for entering purgatory and hell respectively. If we let \( S_p \) represent the accumulated sin value of person \( p \), then \( p \) enters purgatory if \( M_p \leq S_p < M_h \). Similarly, \( p \) enters hell if \( S_p \geq M_h \). Finally, if \( S_p < M_p \) we distinguish between entering heaven and limbo by those who have un-absolved original sin \( (S_o) \) enter limbo; whereas those who have absolved original sin enter heaven.

\[
Judge : Per J d \mapsto (Per J d \mapsto (Sin J d \mapsto N_1)) \times (Soul J d \mapsto (Sin J d \mapsto N_1)) \times (Sin J d \mapsto Q^+) \mapsto Loc
\]

\[
Judge [p] (\beta_1, \beta_4, \beta_5) \triangleq
S_p \mapsto \Sigma((\beta_1(p) \oplus \beta_4(\beta_2(p))) \otimes \beta_3)
M_p \leq S_p < M_h \mapsto Purgatory
M_h \leq S_p \mapsto Hell
0 \leq S_p < M_p
S_o \in \beta_1(p)
\mapsto Limbo
\mapsto Heaven
\]
Note
The Judge operation as presented considers the sins previously committed and confessed by the person as well as unconfessed sins in order to determine the final destination for the person. The $\otimes \oplus$ operation is the indexed ring multiplication operation, which acts on common domain elements on the two maps to be multiplied. It is defined in Appendix B.

Modelling: The Trinity

Christians believe in one God; they believe that this one God is manifested in three forms, God the father, God the Son and God the Holy Spirit. Christians reject that this belief is polytheism, instead the three persons of the Trinity represent one and the same Divinity. We note the second of the Trinity became man and spread the Word of God on earth. Furthermore, we note that the second of the Trinity is seated at the right hand of the Father and yet is identical with the Father. The Holy Spirit proceeds from the Father and the Son, as stated in the Nicean Creed; this procession from the Father and the Son, in particular the latter, i.e., the Filioque, is a source of the schism between the Eastern and Western wings of Christianity.

To model the Trinity we must capture the humanity of Christ and additionally that the three Gods, Father, Son and Holy Spirit are in fact identical. We let $\zeta_1$ represent the relationship between gods which are present in incarnate form, for example Krishna may be regarded as an incarnation of the Hindu god Vishnu. We let $\zeta_2$ be the relationship of god aliasing, i.e., each god may have several distinct but identical manifestations.

$$\zeta_1 : \text{Per}_Jd \mapsto \text{God}_Jd$$
$$\zeta_2 : \text{God}_Jd \mapsto \mathcal{P'}\text{God}_Jd$$

We let $R$ denote the relation corresponding to $\zeta_2$; we stipulate that $R$ is in fact an equivalence relation, and yields a partition of $\text{dom}\zeta_2$. We define the invariant as follows:

$$\text{Inv}_{\text{Alias}}[\zeta_1, \zeta_2] \triangleq$$

\begin{align*}
\text{rng} \zeta_1 & \subseteq \text{dom} \zeta_2 \\
\cup / \text{rng} \zeta_2 & = \text{dom} \zeta_2 \\
\forall g \in \zeta_2 \ g \in \zeta_2(g) \\
\forall g_1, g_2 \in \zeta_2 \\
& \quad g_2 \in \zeta_2(g_1) \Rightarrow g_1 \in \zeta_2(g_2) \\
\forall g_1, g_2, g_3 \in \zeta_2 \\
& \quad (g_2 \in \zeta_2(g_1) \land g_3 \in \zeta_2(g_2)) \Rightarrow g_3 \in \zeta_2(g_1)
\end{align*}
The Trinity will be defined in terms of an equivalence class of this equivalence relation; we define the equivalence class of a god as follows:

\[ Cl(g) = \{x : x \in \zeta_2 | gRx\} \]

**Lemma 2.8** \( \text{dom} \zeta_2 \) is partitioned by the \( Cl(g) \).

**Proof**
This is a well known result, (cf., Chapter 2 of [3]) for details.

Letting \( Cl(G) \) represent the equivalence class of the Christian God; we have \( |Cl(G)| = 3 \); furthermore we have \( \text{rng} \zeta_1 \cap Cl(G) \neq \emptyset \) as Christ became man. In fact, the precise formulation is \( |\text{rng} \zeta_1 \cap Cl(G)| = 1 \)

### 2.3.3 Islam

This religion was founded by Mohammed in the seventh century and is a monotheistic religion. This absolute God, Lord and creator is Allah, the sole master of mankind. The final judgement will be a terrible vengeance on the ungodly; man’s only hope is Islam (meaning submission to God.). The Koran (Qur’an) meaning recitation is the word of Allah as revealed through the prophet Mohammed. The teaching of Mohammed was initially rejected by the population of Mecca, and he fled to Medina. Later he conquered Mecca by force and the concept of the \( \text{jihad} \) or holy war to convert unbelievers is fundamental in Islam. God is portrayed in places in the Koran as ready to forgive; in other places he punishes. Christ is considered a prophet in the Koran, and it is emphasized repeatedly that God could not have a son.

Hours of prayers are announced by a caller (muezzin). Fasting takes place during the ninth lunar month (Ramadan), and is strictly observed. This involves absence from food and drink between sun-rise and sun-set. Polygamy is allowed for Moslem men, but not for Moslem women. Furthermore Moslem men are allowed to marry Christians or Jews, however this is forbidden to Moslem women. Women play a subservient role in Moslem society, and it is typical for a veil to be worn for modesty purposes. We shall consider the problem of modelling Islamic marriages.

\[
\begin{align*}
\gamma_1 : \text{Per}_d & \mapsto \mathcal{P}' \text{Per}_d \\
\gamma_2 : \text{Per}_d & \mapsto \text{Gen} \\
\gamma_3 : \text{Per}_d & \mapsto \text{B}
\end{align*}
\]
The invariant must stipulate that no woman may be married to more than one man (the standard slave serves one master constraint). Furthermore, any marriage which takes place must be between opposite genders.

\[ \text{Inv}_{\text{IslMar}} : (\text{Per}_J d \mapsto \mathcal{P}' \text{Per}_J d) \times (\text{Per}_J d \mapsto \text{Gen}) \times (\text{Per}_J d \mapsto B) \mapsto B \]

\[ \text{Inv}_{\text{IslMar}}[\gamma_1, \gamma_2, \gamma_3] \triangleq \]

\[ \text{dom} \gamma_2 = \text{dom} \gamma_3 \land \text{dom} \gamma_1 \subseteq \text{dom} \gamma_2 \]

\[ \land \cup \circ \text{rng} \gamma_1 \subseteq \text{dom} \gamma_1 \]

\[ \land \forall p \in \gamma_1 \{ \gamma_2(p) \} \cup, \mathcal{P}_\gamma_2(\gamma_1(p)) = \emptyset \]

\[ \land \forall p_1, p_2 \in \gamma_1 \]

\[ (\gamma_2(\gamma_1(p_1)) = \gamma_2(\gamma_1(p_2)) = F) \land (\gamma_1(p_1) = \gamma_1(p_2)) \Rightarrow p_1 = p_2 \]

The \text{Isl}_\text{Mrg} \text{ operation distinguishes between the marriage a Moslem man may legally have and the marriage a Moslem woman may legally have. This distinction is made in the precondition; the \text{Isl}_\text{Mrg} \text{ operation is specified as follows:}

\[ \text{I}_\text{sl}_\text{Mrg} : (\text{Per}_J d \times \text{Per}_J d) \mapsto (\text{Per}_J d \mapsto \mathcal{P}' \text{Per}_J d) \times (\text{Per}_J d \mapsto \text{Gen}) \times (\text{Per}_J d \mapsto B) \mapsto (\text{Per}_J d \mapsto \mathcal{P}' \text{Per}_J d) \]

\[ \text{I}_\text{sl}_\text{Mrg}[p_1, p_2](\gamma_1, \gamma_2, \gamma_3) \triangleq \]

\[ \mapsto (\gamma_1 \circ [p_1 \mapsto \{ p_2 \}]) \circ [p_2 \mapsto \{ p_1 \}] \]

\[ \text{pre}_\text{I}_\text{sl}_\text{Mrg} : (\text{Per}_J d \times \text{Per}_J d) \mapsto (\text{Per}_J d \mapsto \mathcal{P}' \text{Per}_J d) \times (\text{Per}_J d \mapsto \text{Gen}) \times (\text{Per}_J d \mapsto B) \mapsto B \]

\[ \text{pre}_\text{I}_\text{sl}_\text{Mrg}[p_1, p_2](\gamma_1, \gamma_2, \gamma_3) \triangleq \]

\[ \gamma_2(p_1) \cup, \gamma_2(p_2) = \emptyset \]

\[ \land \gamma_3(p_1) \land \gamma_3(p_2) \]

\[ \land \gamma_2(p_1) = F \Rightarrow p_1 \not\in \gamma_1 \]

\[ \land \gamma_2(p_2) = F \Rightarrow p_2 \not\in \gamma_1 \]

We must show that the Islamic marriage operation preserves the invariant; i.e., we have the following proof obligation stated as a lemma.

**Lemma 2.9** \text{pre}_\text{I}_\text{sl}_\text{Mrg}[p_1, p_2](\gamma_1, \gamma_2, \gamma_3) \land \gamma'_1 = \text{I}_\text{sl}_\text{Mrg}[p_1, p_2] \Rightarrow \text{Inv}_{\text{IslMar}}[\gamma'_1, \gamma_2, \gamma_3]$

### 2.3.4 Hinduism

Hinduism is a polytheistic religion, the three main gods being Brahma the creator, Vishnu the protector and Siva the destroyer. There are other incarnations of Vishnu in Krishna and Rama. Estimates suggest that there may well be in excess of 30 million minor deities
in the Hindu religion. The religion is characterized by its belief in re-incarnation and the caste system. The source of early Hinduism is in the four Vedas, which represent the ideas in the eternal mind. The Vedas eventually were surpassed by the Brahmins, which were commentaries on the Vedas and the ritualism as performed by the Brahmins (or priests) became dominant. Reacting against the rise of the Brahmins are the forest treatises (Aranyakas) as recited by hermits culminating with the Upanishads (which contains the fundamentals of Hindu philosophy). Its fundamental philosophy is that Atman = the Brahman the supreme divinity and Atman is the eternal portion of the Brahman which abides in every living human being.

A more popular form of Hinduism (Smriti) or tradition developed in response to the rise of Buddhism. This included the Sutras and the Hindu epics, e.g., the Ramayana. Writings include the Bhagavad Gita (song of the Blessed One) which has been called [31] ‘India’s favourite bible’. This forms part of a larger epic, the Mahabharata. The fundamental feature of Hinduism which we wish to capture here is re-incarnation and the caste system. We shall consider the relationship between persons, souls and the caste system as follows:

\[ Q^\infty = \{-\infty\} \cup Q \cup \{+\infty\} \]

\[ H_{co}(Q^\infty \times Q^\infty) : H_{co}(X,Y) = X < Y \mapsto \{x | x \in Q^\infty \land x \geq X \land x < Y\}, \emptyset \]

Notation 2.8 The notation \( Q^\infty \) is not part of the notation of the Irish school. It is used in defining \( H_{co}(Q^\infty \times Q^\infty) \), the individual caste intervals.

\[ \delta_1 : Per_J \mapsto Soul_J \]
\[ \delta_2 : Per_J \mapsto B \]
\[ \delta_3 : Soul_J \mapsto Cast_J \]
\[ \delta_4 : Per_J \mapsto (Deed_J \mapsto N_1) \]
\[ \delta_5 : Deed_J \mapsto Q \]
\[ \delta_6 : Cast_J \mapsto H_{co}(Q^\infty \times Q^\infty) \]
\[ \delta_7 : Soul_J \mapsto Q^\infty \]

We must stipulate that every living Hindu has a unique soul; this soul may have been present in various deceased Hindus but at any one moment of time this soul is present in at most one Hindu. An individual’s caste is determined by his karma; i.e., the sum of his actions in previous existence. Thus it is appropriate to model this as a relationship between souls and caste. Each living person performs deeds of varying merit; the Brahmins, the priestly caste perform the religious ceremonies; the warrior caste, the Kshatriyas
generally performing deeds of valour; the merchant caste, the Vaśyās performing business; the Sudrās corresponding to the servant class. Finally there are the untouchable Pariahs. In practice movement between castes is rare; a person is born into a particular caste based on actions performed in a previous existence. Mathematically the caste system is a total ordering. We specify the invariant for the Hindu religion as follows:

$$\text{Inv}_{\text{Hin}}: \text{(Per}_{J}d \mapsto \text{Soul}_{J}d) \times \text{(Per}_{J}d \mapsto B) \times (\text{Soul}_{J}d \mapsto \text{Cast}_{J}d) \mapsto B$$

$$\text{Inv}_{\text{Hin}}[\delta_{1}, \delta_{2}, \delta_{3}, \delta_{4}, \delta_{5}, \delta_{6}] \triangleq$$

$$\begin{align*}
\text{dom} \delta_{1} = \text{dom} \delta_{2} \land (\text{rng} \delta_{1} = \text{dom} \delta_{3}) \\
\land \forall p_{1}, p_{2} \in \delta_{1}, \delta_{1}(p_{1}) = \delta_{1}(p_{2}) \Rightarrow (\neg \delta_{2}(p_{1}) \lor \neg \delta_{2}(p_{2})) \\
\land \forall c_{1} \neq c_{2} \in \delta_{6}, \delta_{6}(c_{1}) \cap \delta_{6}(c_{2}) = \emptyset \\
\land \forall c \in \delta_{6}, \delta_{6}(c) \neq \emptyset \\
\land \forall / \text{rng} \delta_{6} = Q^{\infty}
\end{align*}$$

It is unusual for a change in a person's caste to occur, however, if the deeds performed are exceptional a caste promotion may be warranted. If the actions performed are exceptional then a caste demotion may be warranted. The \text{Chk\_Cast} operation is defined as follows:

$$\text{Chk\_Cast}[p](\delta_{1}, \delta_{4}, \delta_{5}, \delta_{6}) \triangleq \delta_{3} (\delta_{1}(p)).$$

The operation \text{Upd\_Cast} updates the caste of the Hindu if the results of the deeds performed in previous or present lifetimes justify it. The operation is not presented here.

**Question 2.18** What are the previous re-incarnations of soul \textit{s}? In what living person (if any) does soul \textit{s} abide?

Previous incarnations of soul \textit{s} is given by $\llbracket\delta_{2}^{-1}(\text{FALSE})\rrbracket \delta_{1}^{-1}(s)$ subject to $s \in \delta_{1}^{-1}$. The current living person in which soul \textit{s} abides is given by $\llbracket\delta_{2}^{-1}(\text{TRUE})\rrbracket \delta_{1}^{-1}(s)$.

**Question 2.19** Given person \textit{p}, what is person \textit{p}'s caste?

This is given by $\delta_{3} \circ \delta_{1}(p)$ subject to $p \in \delta_{1} \land \delta_{1}(p) \in \delta_{3}$. Furthermore, \textit{p} must be a living person, i.e., $\delta_{2}(p)$.

### 2.4 Generic Features of Modelling

The objective of this section is to demonstrate that the basic model of religion developed in Section 2.2.1 may be applied to domains other than religion. This section identifies similar-
ities between the dedicated model of religion, and models of the structure of organizations including politics, companies and a university.

For example, the adaptation of the religion model to a political system involved the realization that political parties may be considered as religions, and that political beliefs are analogous to religious beliefs. Furthermore, the gods of a religion may be identified as elected members of parliament, thus each party has an associated collection of MPs (gods). Each individual has an associated set of personal political beliefs, thus a political schism may be considered analogous to heresy. From a conceptual view the underlying behaviour exhibited between religions and political parties is very similar at the abstract level.

Similarly, the model of religion may be adapted to a large company with directors and sub-offices. In this case, the companies themselves are considered to be religions, the particular company ethos is identified with religious belief, sub offices of companies are identified with churches, managers are identified with the high priests, gods are identified as the directors. In this case the gods themselves have a finite life span and are subject to removal from their position. Consequently, the company environment reflects a more volatile environment than the religion model itself.

Finally, the model of religion may be adapted to model the structure of a university. The universities themselves are identified as religions, the individual academic tradition corresponding to religious belief, the deans may be modelled as gods, departments as churches and staff and heads of department as the high priests. The environment is subject to less volatility than the political model or the company model.

This suggests that the religion model is generic. It may be tailored to model the organizational structure of companies, universities and politics. The adaptation is not exact, however, the behaviour of these models is close to the behaviour of the religion model at this level of abstraction. Once a generic model has been studied and well understood, properties of the dedicated models which are derived from the generic model are immediately available. The advantage of this is that duplication of effort is minimized.

**Political Systems**

This section demonstrates that a model of the structure of politics may be derived from the religion model. The domain names of the religion model are replaced by the corresponding names for the political domain.

\[
\alpha : \text{Per} Jd \leftrightarrow \text{Par} Jd \tag{2.29}
\]

\[
\alpha_1 : \text{Per} Jd \leftrightarrow \text{B} \tag{2.30}
\]

\[
\beta : \text{Par} Jd \leftrightarrow \mathcal{PM} Jd \tag{2.31}
\]

\[
\gamma : \text{ParOff} Jd \leftrightarrow \text{Par} Jd \tag{2.32}
\]
\[ \delta : ParOff Jd \mapsto \mathcal{P}ParWrk Jd \] (2.33)
\[ \epsilon : ParOff Jd \mapsto Loc Jd \] (2.34)
\[ \mu : Par Jd \mapsto \mathcal{P}Pol_Bel Jd \] (2.35)

**Question 2.20** What are the similarities and differences between the two models?

One key difference is that the MPs, i.e., the gods themselves in terms of the original model, are mortal. Furthermore, the MP must be a member of the party for which it is an MP (i.e. god). In the original model, the gods are immortal.

The operation to determine all the gods is equivalent in the political model to determining all the MPs in the parliament. The party offices correspond to the churches. They are responsible for recruiting new members to the political ideology. In practice, the different party offices of the same party tend to be concerned more with their own local god (or MP) for their constituency, which may differ from the original model in that all gods may be assigned an equal weight in an individual church. The political environment is volatile in that gods are subject to re-election, thus gods may be removed from office.

The creation of a new political party corresponds to the creation of a new religion, the new political party may or may not have gods depending on whether the party has been created by a breakaway group of MPs. Heresy corresponds to political dissonance, unresolved heresy results in a schism resulting in the creation of new political parties, political agnosticism corresponds to a political ideology without beliefs. A newly created political party may initially have no members.

The \( \text{Upd}_\text{Par} \) operation is an operation which updates the gods (MPs) in a political party. The precondition for this operation requires the extra constraint that the gods (i.e., MPs) are members of the party. The operation also updates the political beliefs of a party. The precondition for the operation is presented as follow:

\[ \text{pre}_\text{Upd}_\text{Par} : (Par Jd \mapsto \mathcal{P}MP Jd) \times (Par Jd \mapsto \mathcal{P}Per Jd) \times Par Jd \times \mathcal{P}MP Jd \mapsto B \]
\[ \text{pre}_\text{Upd}_\text{Par}(\beta, \alpha^{-1}, p_t, ms) \triangleq \]
\[ \chi[\beta], \beta \]
\[ \land \chi[\alpha^{-1}, \alpha^{-1} \land ms \subseteq \alpha^{-1}(p_t) \]

The \( \text{Create}_\text{Par} \) operation is presented as follows:

\[ \text{Create}_\text{Par} : (Par Jd \mapsto \mathcal{P}MP Jd) \times (Par Jd \mapsto \mathcal{P}Bel Jd) \times Par Jd \times \mathcal{P}MP Jd \times \mathcal{P}Bel Jd \mapsto (Par Jd \mapsto \mathcal{P}MP Jd) \times (Par Jd \mapsto \mathcal{P}Bel Jd) \]
\[ \text{Create}_\text{Par}(\beta, \mu, p_t, m, bs) \triangleq (\beta \sqcup [p_t \mapsto m], \mu \sqcup [p_t \mapsto bs]) \]
The Model of a Company

This section demonstrates that the model of religion may be adapted to model the organizational structure of a company.

\[ \alpha : Per.Id \mapsto Com.Id \]  (2.36)
\[ \alpha_1 : Per.Id \mapsto B \]  (2.37)
\[ \beta : Com.Id \mapsto PDir.Id \]  (2.38)
\[ \gamma : Off.Id \mapsto Com.Id \]  (2.39)
\[ \delta : Off.Id \mapsto PStaff.Id \]  (2.40)
\[ \epsilon : Off.Id \mapsto Loc.Id \]  (2.41)
\[ \mu : Com.Id \mapsto PCom.Bel.Id \]  (2.42)

**Question 2.21** What are the similarities and differences between the religion model and the company model?

The company directors (i.e., the gods) are mortal as before. The environment is highly volatile with directors removed from office for poor performance. The performance of a god is not considered in the religion model. The director must be a member of the company (or party). The sub-offices correspond to the churches with the high priests corresponding to the managers and the staff of the sub-offices.

The *Chg_Job* operation corresponds to the change religion operation. The model is not suitable for consultants who may work with several companies at one time, as the slave serves one master constraint is enforced in the invariant. An employee of a company may work in more than one sub-office. The model does not consider the level of remuneration of the employee.

The Model of a University

This section demonstrates that the model of religion may be adapted to model the organizational structure of a university.

\[ \alpha : Per.Id \mapsto Univ.Id \]  (2.43)
\[ \alpha_1 : Per.Id \mapsto B \]  (2.44)
\[ \beta : Univ.Id \mapsto PDean.Id \]  (2.45)
\[ \gamma : Dep.Id \mapsto Univ.Id \]  (2.46)
\[ \delta : Dep.Id \mapsto P.Lect.Id \]  (2.47)
\[ \epsilon : \text{Dep Id} \mapsto \text{Loc Id} \quad (2.48) \]
\[ \mu : \text{Univ Id} \mapsto \mathcal{P}\text{Univ Bel Id} \quad (2.49) \]

**Question 2.22** What are the similarities and differences between the religion model and the university model?

The gods (i.e., deans) are mortal as before. The high priests include lecturers, heads of department, etc. The ethos and tradition of a university is represented by its belief system, heresy may correspond to opposing philosophical positions.

### 2.5 Summary

The objective of this chapter is to demonstrate that formal methods may be employed to develop a formal model of aspects of religion. The basic model of religion developed in this chapter has demonstrated that the organizational structure of aspects of religion may be captured at an abstract level. The model has clearly demonstrated how elementary facts such as the deities which a particular religion believes in, and the individual beliefs which a particular religion has may be encoded in the model.

The evaluation of the model consisted of evaluation questions. The adequacy of the model is judged by its effectiveness in answering the evaluation questions. The intention of the model is to model aspects of religion. It is accepted that the domain of religion is far too complex for any formal statement to capture exactly.

The model of specific religions of the world demonstrates how aspects of the beliefs of individual religion may be formally encoded. The models have demonstrated how aspects of legitimate marriages for Judaism and Islam may be modelled. Furthermore, the problem of modelling the Trinity, sin and absolution has been considered for the Christian religions. Finally, aspects of the caste system have been considered for the Hindu religion.

Finally, the basic model of religion is generic. It may be adapted to model the organizational structure of politics, a company or a university. Such a model minimizes duplication of effort in proof and model exploration.
Chapter 3

The Stock Exchange Model

3.1 Introduction

Chapter 2 has demonstrated that VDM* may be employed to develop a model of aspects of religion. The examination of the religion domain indicated that the model is generic, and may be tailored to model the structure and beliefs of organizations other than religion. This includes modelling the organizational structure of a company, a political institution or a university.

A stock exchange is one particular instantiation of an organization. This chapter demonstrates how aspects of a stock exchange may be modelled in the Irish school of VDM. The model which is presented in this chapter is believed to be original. The model of the stock exchange is constructive, indicating that it may be implemented, if this is desired. However, the model is not developed for implementation purposes. The objective instead is to demonstrate that properties of the stock exchange may be formally encoded in a model. Furthermore, the model itself serves as a rigorous mechanism for obtaining a detailed understanding of the properties of the domain and from which further properties of the stock exchange may be derived.

The advantage of a formal model of aspects of the stock exchange is that it serves as a precise and terse means of describing aspects of the behaviour of the stock exchange. The fact that the model is formal ensures that a common understanding of the stock exchange may be shared by all interested parties. This is to be contrasted with an informal description of a stock exchanges. An informal description is prone to misinterpretation due to the ambiguities inherent in natural language.

The adequacy of the stock exchange model is judged by its effectiveness in modelling the behaviour of existing stock exchanges. The model (in common with most models of the real world) is good at explaining some aspects of the stock exchange and weak in explaining other aspects. The adequacy or otherwise of the model indicates the limitations of the model. The limitations of the model enables an informed decision to be made on
whether the model may serve as a representation of the domain.

3.2 The Stock Exchange Model

The model of the stock exchange developed in this chapter provides a formal representation of aspects of the structure and properties of the stock exchange. The following assumptions are made in the model.

1. A Stock Exchange is considered to be a collection of companies, where each company has a collection of shares, and each share has an associated share price.

2. The shares in individual companies are mutually disjoint.

3. The shares in a company are owned by the shareholders, who hold mutually disjoint shareholdings. Investors may hold shareholdings in several companies, each shareholding being mutually disjoint.

4. There is one dealer, who acts as the intermediary for the buyers and sellers. Dealer commission is not considered in the model.

5. Basic financial operations are considered.

6. Dividends are paid on a pro-rata basis.

The environment is predatory; details on the theory and working of the exchange is given in [21], [42], [58]. Three conceptual views of the stock exchange model are considered in this chapter. They are the following:

- **The Exchange Viewpoint**
  This is the global picture of the stock exchange, and consists of a collection of companies, each with a unique collection of shares. Each company has an associated share price.

- **The Investor Viewpoint** This is the viewpoint of an investor with shareholdings in several companies.

- **The Company Viewpoint** A company has several shareholders, each with different share allocations.
3.3 Stock Exchange - Exchange Viewpoint

The stock exchange viewpoint is the global picture of the exchange. The appropriate model at this level is initially considered to be the following:

$$\mu : C \mapsto (\mathcal{P}' S \times \mathbb{Q}^+)$$  \hspace{1cm} (3.1)

Each company listed on the stock exchange has a non empty collection of shares associated with it; moreover these collections of shares are mutually disjoint. Furthermore, each company has an associated share price.

The invariant for this choice of the initial model must ensure that no share belongs to more than one company, (the standard slave serves one master problem), and the share price must be positive. A clumsy formulation of the invariant is given by:

$$\text{Inv}_X \text{chg} : C \mapsto (\mathcal{P}' S \times \mathbb{Q}^+) \mapsto B$$

$$\text{Inv}_X \text{chg} [\mu] \triangleq$$

$$\{c_i, c_j \in \text{dom} \mu \land c_i \neq c_j \} \Rightarrow \pi_1 \circ \mu (c_i) \cap \pi_1 \circ \mu (c_j) = \emptyset$$

However, [36] explains how this formulation may be improved upon. This is achieved by noting that $C \mapsto \mathcal{P}' S$ is invertible and yields $S \mapsto \mathcal{P}' C$ which gives the relationship between the companies which hold a particular share. Since, it is required that no share should belong to two companies, the constraint may be stated by applying a cardinality functional ($I \mapsto \text{card}$) to $\mu^{-1}$ and then insisting that the range of this map is $\{1\}$ for a non trivial exchange $\mu$

$$\text{Inv}_X \text{chg} : C \mapsto (\mathcal{P}' S \times \mathbb{Q}^+) \mapsto B$$

$$\text{Inv}_X \text{chg} [\mu] \triangleq$$

$$\mu \neq \emptyset \Rightarrow \text{rng} \circ (I \mapsto \text{card}) \circ ((I \mapsto \pi_1) \mu)^{-1} = \{1\}$$

The advantage of this invariant is that no reference is made to individual companies in the stock exchange in the statement of the invariant. However, the statement of the invariant is slightly complex, and for simplicity purposes, an alternate form of the initial model is considered.

$$\kappa : S \mapsto C$$  \hspace{1cm} (3.2)

$$\tau : C \mapsto \mathbb{Q}^+$$  \hspace{1cm} (3.3)

The choice of this model immediately enforces the constraint that a slave should serve
exactly one master. Consequently, the corresponding invariant is trivial.

\[ \text{Inv}_{X \text{chg}} : (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \mapsto B \]
\[ \text{Inv}_{X \text{chg}}[\kappa, \tau] \triangleq \text{rng} \kappa = \text{dom} \tau \]

The less complex invariant means easier proofs of invariant preservation. This demonstrates the importance of working with the appropriate model for a system.

The shares in a particular company \( c \) is then given by \( \kappa^{-1}(c) \), where \( \kappa^{-1} : C \mapsto \mathcal{P}'S \) is the inverse image of \( \kappa \). Inverse images partition the domain space into clusters of shares corresponding to the individual companies.

### 3.3.1 Stock Exchange Operations

The operations present at this level of the stock exchange include the following:

\[ CreaX | AddC | DelC | ValC | ValX | Iss | Adj \]

These operations are defined as follows, first the \( CreaX \) operation is defined:

\[ CreaX \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \]
\[ CreaX \triangleq (\theta, \theta) \]

A liberal view is taken to the addition of a company to the exchange; a company may be added to the exchange provided it is not already present, and its shares are disjoint from the shares in other companies listed on the exchange.

\[ AddC : C \times \mathcal{P}'S \times \mathbb{Q}^+ \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \]
\[ AddC[c, S_c, q][\kappa, \tau] \triangleq \text{Let } \kappa_c = \lambda s : S_c \cdot c \text{ in} \]
\[ (\kappa \sqcup \kappa_c, \tau \sqcup [c \mapsto q]) \]

\[ \text{pre}_{AddC} : C \times \mathcal{P}'S \times \mathbb{Q}^+ \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \mapsto B \]
\[ \text{pre}_{AddC}[c, S_c, q][\kappa, \tau] \triangleq (\text{dom} \kappa \cap S_c) = \emptyset \]
\[ \land \neg \chi[c] \tau \]
\[ \land q \geq 0 \]
Listed companies which experience terminal financial difficulties are removed from the exchange. The \(\text{DelC} \) operation is the inverse of the \(\text{AddC} \) operation.

\[
\text{DelC} : C \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto (S \mapsto C) \times (C \mapsto Q^+)
\]

\[
\text{DelC}[c](\kappa, \tau) \triangleq \\
\text{Let } S_c = \kappa^{-1}(c) \text{ in} \\
\mapsto (\exists [S_c], \kappa, c\tau)
\]

\[
\text{pre_DelC} : C \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto B
\]

\[
\text{pre_DelC}[c](\kappa, \tau) \triangleq \chi[c] \tau
\]

The importance of a company is typically judged by its stock market value, which is obtained from the \(\text{ValC} \) operation. This value is computed by determining the number of shares in the company, and multiplying by the corresponding share price of the company.

\[
\text{ValC} : C \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto Q^+
\]

\[
\text{ValC}[c](\kappa, \tau) \triangleq \\
\mapsto |\kappa^{-1}(c)| \ast \tau(c)
\]

\[
\text{pre_ValC} : C \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto B
\]

\[
\text{pre_ValC}[c](\kappa, \tau) \triangleq \chi[c] \tau
\]

The value of the stock exchange is given by the \(\text{ValX} \) operation. This operation determines the accumulated value of each company listed on the exchange. The value of the stock exchange is a measure of its international importance.

\[
\text{ValX} : (S \mapsto C) \times (C \mapsto Q^+) \mapsto Q^+
\]

\[
\text{ValX}(\kappa, \tau) \triangleq \\
\mapsto |\lambda c : \{\text{dom} \tau\} \bullet \text{ValC}[c](\kappa, \tau)|
\]

\textbf{Note:} The bag cardinality operation (\(||\)) is employed in the \(\text{ValX} \) operation.

\[
\text{pre_ValX} : (S \mapsto C) \times (C \mapsto Q^+) \mapsto B
\]

\[
\text{pre_ValX}(\kappa, \tau) \triangleq \text{TRUE}
\]

It is reasonable for a company to occasionally require an increase in shares from its previous or initial share allocation. This requirement is addressed by the \(\text{Iss} \) operations which issues additional shares to the company.
\( \text{Iss} : C \times \mathcal{P}'S \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto (S \mapsto C) \times (C \mapsto Q^+) \)  
\( \text{Iss}[c, S_c](\kappa, \tau) \triangleq \)  
\[ \text{Let } \kappa_c = \lambda s : S_c \cdot c \text{ in } \]  
\[ \kappa \uplus \kappa_c, \tau \]

\( \text{pre-Iss} : C \times \mathcal{P}'S \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto \mathcal{B} \)  
\( \text{pre-Iss}[c, S_c](\kappa, \tau) \triangleq \)  
\[ (\text{dom } \kappa \cap S_c) = \emptyset \]  
\[ \land \chi[c]^{\tau} \]

The \( \text{Iss} \) operation is used to issue extra shares only. The allocation to all shareholders with the exception of the dealer remain unchanged. The \( \text{Adj} \) operation amends the share price of the company.

\( \text{Adj} : C \times Q^+ \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto (S \mapsto C) \times (C \mapsto Q^+) \)  
\( \text{Adj}[c, q](\kappa, \tau) \triangleq (\kappa, \tau \uplus [c \mapsto q]) \)

\( \text{pre-Adj} : C \times Q^+ \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto \mathcal{B} \)  
\( \text{pre-Adj}[c, q](\kappa, \tau) \triangleq \chi[c]^{\tau} \)

This operation is invoked frequently in volatile trading conditions.

### 3.3.2 Preservation of Invariant

Several of the operations presented for the initial model of the stock exchange transform the exchange. There is a corresponding proof obligation to demonstrate that the transformed structure remains a valid stock exchange. There is no proof obligation required for the \( \text{ValC} \) and \( \text{ValX} \) operations.

**Lemma 3.1** \( \text{pre-AddC}[c, S_c, q](\kappa, \tau) \land (\kappa', \tau') = \text{AddC}[c, S_c, q](\kappa, \tau) \Rightarrow \text{Inv}_\text{Xchg}\[\kappa', \tau'] \).  

**Proof**  
\[ \text{rng } \kappa' \]  
\[ = \text{rng } (\kappa \uplus \kappa_c) \]  
\[ = \text{rng } (\kappa \uplus \lambda s : S_c \cdot c) \]  
\[ = \text{rng } \kappa \uplus \{c\} \]  
\[ = \text{dom } \tau \uplus \{c\} \]  
\[ = \text{dom } (\tau \uplus [c \mapsto q]) \]  
as required.
Lemma 3.2 \( \text{pre}_{\text{DelC}}[c][\kappa, \tau] \land (\kappa', \tau') = \text{DelC}[c][\kappa, \tau] \Rightarrow \text{Inv}_{X}\text{chg}[\kappa', \tau'] \).  

Proof  
\[
\begin{align*}
\text{rng } \kappa' & = \text{rng } \circ \text{pre}_c^{-1} \circ \kappa \\
& = \text{pre}_c \circ \text{rng } \kappa \\
& = \text{pre}_c \circ \text{dom } \tau \\
& = \text{dom } \text{pre}_c \circ [c] \tau \\
& = \text{dom } \tau' \\
& \text{as required.}
\end{align*}
\]

Lemma 3.3 \( \text{pre}_{\text{Iss}}[c, S_c][\kappa, \tau] \land (\kappa', \tau') = \text{Iss}[c, S_c][\kappa, \tau] \Rightarrow \text{Inv}_{X}\text{chg}[\kappa', \tau'] \).  

Proof  
\[
\begin{align*}
\text{rng } \kappa' & = \text{rng } (\kappa \cup \kappa_c) \\
& = \text{rng } (\kappa \cup \lambda s : S_c \bullet c) \\
& = \text{rng } \kappa \cup \{c\} \\
& = \text{rng } \kappa \\
& = \text{dom } \tau \\
& = \text{dom } \tau' \\
& \text{as required.}
\end{align*}
\]

Lemma 3.4 \( \text{pre}_{\text{Adj}}[c, q][\kappa, \tau] \land (\kappa', \tau') = \text{Adj}[c, q][\kappa, \tau] \Rightarrow \text{Inv}_{X}\text{chg}[\kappa', \tau'] \).  

Proof  
\[
\begin{align*}
\text{rng } \kappa' & = \text{rng } \kappa \\
& = \text{dom } \tau \\
& = \text{dom } \tau' \\
& \text{as required.}
\end{align*}
\]

This completes the study of the stock exchange from the exchange viewpoint. The next section considers the exchange from the investor viewpoint; this includes modelling share ownership and investment value.

Amended Add and Del Operations

For reasons which will become clear in the next section, it is necessary to amend the Add and Del operations to ensure that when a company is added to the exchange, all shares in
the company are initially placed with the dealer. Furthermore, whenever a company is to be
deleted from the exchange, it is necessary to ensure that any reference to the company and
its shares is removed from the exchange. The operations are presented without explanation.

\[
AddC : C \times P' S \times Q^+ \mapsto (S \mapsto C) \times (C \mapsto Q^+) \times (P \mapsto (S \mapsto C)) \mapsto \\
(S \mapsto C) \times (C \mapsto Q^+) \times (P \mapsto (S \mapsto C))
\]

\[
AddC[c, S_c, q][\kappa, \tau, \phi] \triangleq \\
\text{Let } \kappa_c = \lambda s : S_c \cdot c \text{ in} \\
\mapsto (\kappa \sqcup \kappa_c, \tau \sqcup [c \mapsto q], \phi \upharpoonright [d \mapsto \phi(d) \sqcup \kappa])
\]

\[
\text{pre}_\text{AddC} : C \times P' S \times Q^+ \mapsto (S \mapsto C) \times (C \mapsto Q^+) \mapsto B
\]

\[
\text{pre}_\text{AddC}[c, S_c, q][\kappa, \tau] \triangleq \\
(\text{dom} \kappa \cap S_c) = \emptyset \land \chi[c][\tau] \land \chi[d][\phi]
\]

\[
DelC : C \mapsto (S \mapsto C) \times (C \mapsto Q^+) \times (P \mapsto (S \mapsto C)) \mapsto \\
(S \mapsto C) \times (C \mapsto Q^+) \times (P \mapsto (S \mapsto C))
\]

\[
\text{DelC}[c][\kappa, \tau, \phi] \triangleq \\
\text{Let } S_c = \kappa^{-1}(c) \text{ in} \\
\mapsto (\llbracket S_c \rrbracket \kappa, \llbracket c \rrbracket \tau, (I \mapsto \llbracket S_c \rrbracket \phi))
\]

\[
\text{pre}_\text{DelC} : C \mapsto (S \mapsto C) \times (C \mapsto Q^+) \times (P \mapsto (S \mapsto C)) \mapsto B
\]

\[
\text{pre}_\text{DelC}[c][\kappa, \tau, \phi] \triangleq \chi[c][\tau]
\]

### 3.4 Stock Exchange - Investor Viewpoint

The shares in the companies are owned by individual investors. An investor may own
shares in several companies, and is effectively a mini stock exchange. The key constraint
is that all shares held by investors are mutually disjoint, and shares in different companies
held by an investor are mutually disjoint. Furthermore, the original stock exchange model
is gained by merging the holdings of the individual investors.

The initial investor model proved to be difficult to work with, in particular, the cor-
responding invariant resulted in several tedious proof obligations. The initial model was
subsequently re-examined, and a simpler form proposed. The initial model and its invariant
are defined as follows.

\[
\phi : P \mapsto (C \mapsto P' S)
\]

\[
\text{Inv_Invest} : C \mapsto (P' S \times Q^+) \times P \mapsto (C \mapsto P' S) \mapsto B
\]

58
\[ I_{\text{Inv\_Inst}}[\mu, \phi] \triangleq \]
\[ \cup / \circ \mathcal{P}_1 \circ \text{rng } \mu = \cup / \circ \mathcal{P}_\text{rng} \circ \text{rng } \phi \]
\[ \land \pi_1 \circ \mu(c_i) = I_{\text{Inv\_Shs\_Cmp}}[\gamma][c_i] \]
\[ \land \cup / \circ \mathcal{P}_\text{dom} \circ \text{rng } \phi = \text{dom } \mu \]
\[ \land (\phi(p_i))(c_j) \cap (\phi(p_k))(c_j) = \emptyset \forall p_i, p_k \in \text{dom } \phi, \forall c_j, c_i \in \text{dom } \circ \phi(p_i), \text{dom } \circ \phi(p_k) \]
\[ \text{where } c_j \neq c_i \lor (p_i \neq p_k) \]

Proof obligations are directly influenced by invariant complexity. In particular, the initial formulation of the invariant proved to be extremely tedious to work with. Secondly, it makes direct reference to elements of the structure by name; it is preferable to avoid this. Finally, the \( I_{\text{Inv\_Shs\_Cmp}} \) operation needs to be explicitly defined for a full understanding of the invariant. Further examination yielded a simpler model and invariant for the investor model. It is defined as follows.

\[ \phi : P \mapsto (S \mapsto C) \] (3.5)

\[ I_{\text{Inv\_Inst}} : (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \times (P \mapsto (S \mapsto C)) \mapsto B \]

\[ I_{\text{Inv\_Inst}}[\kappa, \tau, \phi] \triangleq \]
\[ \cup / \circ \text{rng } \phi = \kappa \]
\[ \land (\text{rng } \phi \neq \emptyset) \Rightarrow \text{rng} \circ (I \mapsto \text{card}) \circ ([I \mapsto \text{dom } \phi]^{-1} = \{1\} \]

The use of the bag union operation (\( \cup \)) yields a structure of the form \( S \mapsto \mathcal{P}'C \), rather than \( S \mapsto C \) which is the signature of \( \kappa \). The requirement here is that \( \cup \) should effectively function as the \( \cup \) operation, i.e., all map elements in the range have disjoint domains.

The map domains must be guaranteed to be disjoint in order for the \( \cup \) operation to be applied directly. Otherwise, the result of the operation is undefined. Experience suggests that it is preferable to avoid an undefined value. The \( \cup \) operation ensures that an undefined value does not result.

Strictly speaking, map elements in the range of \( \phi \) should be converted from \( S \mapsto \mathcal{P}'C \) to \( S \mapsto \mathcal{P}'C \), to ensure that the \( \cup \) operation may be successfully applied. Adhering to this fully tends to obscure, rather than clarify, and in this instance notation is abused.

The second part of the invariant is adapted from [36]. It ensures that no share belongs to more than one investor. The invariant ensures the following properties are satisfied.

- The merging of the investors’ shareholding yields the stock exchange model.
- The shares in an individual company is composed of the shares of the individual investors in that company.
- All shares owned in individual companies by the individual investors are mutually disjoint.
3.4.1 Investor Operations

Share purchase or sale proceeds through a special investor or intermediary termed the dealer. The dealer is assumed to be prepared to engage in buying or selling operations at any time. Furthermore, the dealer is assumed to own all shares in all companies when the exchange is created. Whenever a company is added to the exchange, the shares in the company are allocated to the dealer. Share ownership then changes dynamically as investors buy or sell shares.

The following operations are considered appropriate for the investor model.

\[ CreaI|Buy|Sell|ValI|ValX|AddI|List \]

The \( CreaI \) is the first operation to consider, its effect is to make the dealer the sole owner of all shares in all companies on the exchange.

\[ CreaI : (S \mapsto C) \mapsto (P \mapsto (S \mapsto C)) \]
\[ CreaI[k] \triangleq [d \mapsto k] \]

\( \text{pre}_\text{CreaI} : (S \mapsto C) \mapsto B \)
\[ \text{pre}_\text{CreaI}[k] \triangleq \text{TRUE} \]

The effect of the \( \text{Buy} \) operation is to amend an investor’s shareholding in a company, and reduce the dealer’s shareholding by the corresponding amount. If the result of the operation leaves the dealer with an empty shareholding in the company, the dealer is effectively removed as a shareholder in the company.

\[ \text{Buy} : (P \times C \times \mathcal{P}(S) \mapsto (P \mapsto (S \mapsto C)) \mapsto (P \mapsto (S \mapsto C)) \]
\[ \text{Buy}[p, c, S_c][\phi] \triangleq \]
\begin{itemize}
  \item \text{Let } \phi'_p = \phi(p) \sqcup \lambda s : S_c \bullet c \text{ in}
  \item \text{Let } \phi'_d = \phi[S_c][\phi(d)] \text{ in}
    \begin{itemize}
      \item \( \phi \upharpoonright [p \mapsto \phi'_p] \upharpoonright [d \mapsto \phi'_d] \)
    \end{itemize}
\end{itemize}

\( \text{pre}_\text{Buy} : (P \times C \times \mathcal{P}(S) \mapsto (P \mapsto (S \mapsto C)) \mapsto B \)
\[ \text{pre}_\text{Buy}[p, c, S_c][\phi] \triangleq \]
\begin{itemize}
  \item \( \chi[p] \phi \land \chi[d] \phi \)
  \item \( \land S_c \subseteq \phi(d) \)
  \item \( \land \mathcal{P}(\phi(d))(S_c) = \{c\} \)
  \item \( \land S_c \neq \emptyset \land d \neq p \)
\end{itemize}
Note: The dealer must have sufficient shares of the particular company in order to ensure that a successful Buy operation can take place.

The Sell operation is the inverse of the Buy operation. This allows an investor to dispose of all or part of the shareholding held in a company. The model assumes that the dealer is prepared to engage in share purchase operations at any time. The Sell operation is defined as follows:

\[ Sell : (P \times C \times P^U S) \mapsto (P \mapsto (S \mapsto C)) \mapsto (P \mapsto (S \mapsto C)) \]
\[ Sell[p, c, S_c] \phi \triangleq \]
\[ \text{Let } \phi_d = \phi(d) \uplus \lambda s : S_c \cdot c \text{ in} \]
\[ \text{Let } \phi_p = \forall [\lbrack S_c \rbrack \phi(p)] \text{ in} \]
\[ \mapsto (\phi \uparrow \lbrack p \mapsto \phi'_p \rbrack) \uparrow [d \mapsto \phi'_d] \]

\[ \text{pre}_\text{Sell} : (P \times C \times P^U S) \mapsto (P \mapsto (S \mapsto C)) \mapsto B \]
\[ \text{pre}_\text{Sell}[p, c, S_c] \phi \triangleq \]
\[ \chi[p] \phi \land \chi[d] \phi \]
\[ \land S_c \subseteq \phi(p) \]
\[ \land \mathcal{P}(\phi)(S_c) = \{c\} \]
\[ \land S_c \neq \emptyset \land d \neq p \]

The last two operations in this section are the AddI operation which adds an investor to the model, and the List operation which lists the investors in the model.

\[ AddI : P \mapsto (P \mapsto (S \mapsto C)) \mapsto P \mapsto (S \mapsto C) \]
\[ AddI[p] \phi \triangleq \phi \uplus \lbrack p \mapsto \emptyset \rbrack \]

\[ \text{pre}_\text{AddI} : P \mapsto (P \mapsto (S \mapsto C)) \mapsto B \]
\[ \text{pre}_\text{AddI}[p] \phi \triangleq \neg \chi[p] \phi \]

\[ \text{List} : (P \mapsto (S \mapsto C)) \mapsto \mathcal{P} P \]
\[ \text{List}[\phi] \triangleq \text{dom } \phi \]

\[ \text{pre}_\text{List} : (P \mapsto (S \mapsto C)) \mapsto B \]
\[ \text{pre}_\text{List}[\phi] \triangleq \text{TRUE} \]
3.4.2 Valuation Operations

It is likely that the investor will have shareholdings in several companies. Consequently, it is required to determine the total value of the shareholdings held. This is given by the $ValI$ operation.

$$ValI : P \mapsto (P \mapsto (S \mapsto C)) \times (C \mapsto Q) \mapsto Q^+$$

$$ValI[p](\phi, \tau) \triangleq Val\tau \circ \phi(p)[0]$$

$$\text{pre}_ValI : P \mapsto (P \mapsto (S \mapsto C)) \times (C \mapsto Q) \mapsto B$$

$$\text{pre}_ValI[p](\phi, \tau) \triangleq \chi[p] \phi$$

$$Val : (S \mapsto Q^+) \mapsto Q \mapsto Q$$

$$Val[\theta]q \triangleq q$$

$$Val[[s \mapsto q] \sqcup \mu]q' \triangleq Val[\mu](q + q')$$

The $ValX$ operation presented in Section 3.3.1 determines the current value of the stock exchange. The value of the exchange is determined by obtaining the accumulated value of all companies listed on the exchange. An alternate formulation may be derived from the $ValI$ operation. The value of the exchange is then the accumulated value of the individual investor portfolios. The $ValX$ operation is presented as follows:

$$ValX : (P \mapsto (S \mapsto C)) \times (C \mapsto Q^+) \mapsto Q^+ \mapsto Q$$

$$ValX[\theta, \tau]q \triangleq q$$

$$ValX[\phi, \tau]q \triangleq$$

Let $p \in \text{dom} \phi$ in

$$ValX[[s[p]] \phi, \tau] (ValI[p](\phi, \tau) + q)$$

$$\text{pre}_ValX : (P \mapsto (S \mapsto C)) \times (C \mapsto Q^+) \mapsto B$$

$$\text{pre}_ValX[\phi, \tau] \triangleq \text{TRUE}$$

3.4.3 Preservation of Invariant

The valuation operations do not transform the stock exchange, and thus no proof of invariant preservation is required for the $ValI$ and $ValX$ operations. Each proof of invariant preservation has two parts; each part is proved separately.

**Lemma 3.5** $\text{pre}_\text{Create}\[\kappa\] \land \phi' = \text{Create}[\kappa] \Rightarrow \text{InvInvest}[\kappa', \tau', \phi']$. 

62
Proof

The first part of the proof of invariant preservation requires proof that $\forall \phi \in \text{rng} \phi = \kappa$.  

The second part of the proof is to show $\text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ [(\mathcal{I} \mapsto \text{dom}) \phi]^{-1} = \{1\}$

\[
\begin{align*}
\text{rng} \phi' \\
&= \text{rng} [d \mapsto \kappa] \\
&= \{ \kappa \}
\end{align*}
\]

$\forall \phi \in \text{rng} \phi = \kappa$ as required.

\[
\begin{align*}
\text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ [(\mathcal{I} \mapsto \text{dom}) \phi]^{-1} \\
&= \text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ [d \mapsto \text{dom} \kappa]^{-1} \\
&= \text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ [\lambda s : \text{dom} \kappa \cdot \{d\}] \\
&= \text{rng} \circ [\lambda s : \text{dom} \kappa \cdot \{d\}] \\
&= \text{rng} \circ [\lambda s : \text{dom} \kappa \cdot 1] \\
&= \{1\}
\end{align*}
\]

as required.

Lemma 3.6  \text{pre-} \text{Buy} [\llbracket p, c, S_c \rrbracket \phi \land \phi' = \text{Buy}[\llbracket p, c, S_c \rrbracket \phi \Rightarrow \text{Inv Jinvst} [\kappa, \tau, \phi']]$

Proof

It is required to prove $\forall \phi \in \text{rng} \phi = \kappa$ and $\text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ [(\mathcal{I} \mapsto \text{dom}) \phi]^{-1} = \{1\}$

\[
\begin{align*}
\text{rng} \phi' \\
&= \text{rng} \llbracket \{d, p\} \rrbracket \phi \cup \{\phi_d, \phi_p\}
\end{align*}
\]

$\forall \phi \in \text{rng} \phi = \kappa$ as required.
\[\text{rng } \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom}) \phi']^{-1} \]
\[= \text{rng } \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom})(\epsilon[d, p] \phi \cup [p \mapsto \phi'_p]) \cup [d \mapsto \phi'_d])^{-1} \]
\[= \text{rng } \circ (I \mapsto \text{card}) \circ [\lambda q : \epsilon[d, p]\text{dom }\phi \bullet \text{dom }\phi(q) \cup [p \mapsto \text{dom }\phi'_p] \cup [d \mapsto \text{dom }\phi'_d])^{-1} \]
\[= \text{rng } \circ (I \mapsto \text{card}) \circ [(\lambda q : \epsilon[d, p]\text{dom }\phi \bullet \text{dom }\phi(q)]^{-1} \cup (\lambda s : \epsilon[S_c] \text{dom }\phi_d \bullet \{p\}) \cup (\lambda s : \epsilon[S_c] \text{dom }\phi_d \bullet \{p\}) \]
\[= \{1\} \cup \{1\} \cup \{1\} \]
\[= \{1\} \]
as required.

**Note** There are hidden lemmas in this proof.

**Lemma 3.7** \(\text{pre}_{\text{Sell}}[p, c, S_c] \phi \land \phi' = \text{Sell}[p, c, S_c] \phi \Rightarrow \text{Inv}_M \text{nst}[\kappa, \tau, \phi']\).

**Proof**

The proof of invariant preservation is similar to that of the \text{Buy} operation.
\[\text{rng } \phi'\]
\[= \text{rng} \in \{[d, p] \phi \cup \{\phi_d', \phi'_p\}\}
\[\text{\textcircled{1}} \circ \text{rng } \phi' \]
\[= \text{\textcircled{1}} \circ \text{rng} \in \{[d, p] \phi \cup \{\phi_d', \phi'_p\}\}
\[= \text{\textcircled{1}} \circ \text{rng} \in \{[d, p] \phi \cup \{S_c \phi(p) \cup \phi(d) \cup \lambda s : S_c \bullet c\}\}
\[= \text{\textcircled{1}} \circ \text{rng} \in \{[d, p] \phi \cup \{S_c \phi(p) \cup \lambda s : S_c \bullet c\} \cup \phi(d)\}
\[= \text{\textcircled{1}} \circ \text{rng} \in \{[d, p] \phi \cup \phi(p) \cup \phi(d)\}
\[= \kappa\]
as required.

The proof for the second part of the invariant is similar to the proof of the \text{Buy} operation, and is not presented here.

**Lemma 3.8** \(\text{pre}_{\text{AddI}}[p] \phi \land \phi' = \text{AddI}[p] \phi \Rightarrow \text{Inv}_M \text{nst}[\kappa, \tau, \phi']\).

**Proof**
\[\text{rng } \phi'\]
\[= \text{rng} (\phi \cup [p \mapsto \theta])\]
\[= \text{rng} \phi \cup \theta\]

64
\( \emptyset / \circ \text{rng } \emptyset \)
\[ = \emptyset / \circ \text{rng } \emptyset \emptyset / \emptyset \theta \]

\[ = \kappa \]

as required.

\[ \text{rng } \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom}) \phi^{-1}] \]
\[ = \text{rng } \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom})(\phi \sqcup [p \mapsto \theta])]^{-1} \]
\[ = \text{rng } \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom}) \phi \sqcup [p \mapsto \emptyset]]^{-1} \]
\[ = \text{rng } \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom}) \phi]^{-1} \]
\[ = \{1\} \]

as required.

\subsection{3.5 The Company Viewpoint}

The shares in an individual company are owned by the shareholders in the company. The key operations for a company are to know who its shareholders are, and to pay dividends on a pro rata basis. A naive predator prey company take-over operation is presented.

The initial model and invariant for the company viewpoint proved to be difficult to work with. In particular, the invariant resulted in several tedious proof obligations. The initial model was subsequently re-examined, and a simpler form proposed. The initial model and its invariant are defined as follows.

\[ \psi : C \mapsto (P \mapsto \mathcal{P}'S) \] (3.6)

\[ \text{Inv}_C \text{Mdl} : C \mapsto (\mathcal{P}'S \times Q^+) \times C \mapsto (P \mapsto \mathcal{P}'S) \mapsto \mathbb{B} \]

\[ \text{Inv}_C \text{Mdl}[\mu, \psi] \triangleq \]

\[ \text{dom } \psi = \text{dom } \mu \]

\[ \forall cp \in \text{dom } \mu \]

\[ \cup / \circ \text{rng } \circ \psi(cp) = \pi_1 \mu(cp) \]

\[ \forall cp_i, cp_j \in \text{dom } \psi \text{ and } \forall p_k, p_l \in \text{dom } \psi(cp_i), \text{dom } \psi(cp_j) \]

\[ \psi(cp_i)(p_k) \cap \psi(cp_j)(p_l) = \emptyset \text{ except where } cp_i = cp_j \land p_k = p_l \]

\[ \forall cp_i, cp_j \in \text{dom } \psi \text{ where } cp_i \neq cp_j \]

\[ \cup / \circ \text{rng } \circ \psi(cp_i) \cap \cup / \circ \text{rng } \circ \psi(cp_j) = \emptyset. \]

\[ \land \cup / \circ \mathcal{P} \text{rng } \circ \text{rng } \psi = \cup / \circ \mathcal{P} \pi_1 \circ \text{rng } \mu \]

The invariant is excessively complex, leading to very tedious proof obligations. A re-examination of the model for the company viewpoint yielded the following simpler form.
\[ \psi : C \mapsto (S \mapsto P) \]  

\[ \text{Inv}_{\mathcal{CMd}} : (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \times C \mapsto (S \mapsto P) \mapsto \mathcal{B} \]

\[ \text{Inv}_{\mathcal{CMd}}[\kappa, \tau, \psi] \triangleq \]

\[ (\mathcal{I} \mapsto \text{dom})\psi = \kappa^{-1} \]

\[ \land (\kappa \neq \emptyset) \Rightarrow \text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ [(\mathcal{I} \mapsto \text{dom})\psi]^{-1} = \{1\} \]

The invariant ensures that the following properties are satisfied.

1. The merging of the shareholdings of the different companies yields the stock exchange model.

2. The shares in an individual company is composed of the shares of the individual investors in that company.

3. All shares owned in individual companies by individual investors are mutually disjoint.

### 3.5.1 Company Operations

Several of the operations in the company model are concerned with administrative management for the company. Dividends are paid on an annual or biannual basis based on the earning per share. In order for a quoted company to fulfil its obligations it is necessary to maintain records of who the shareholders are, and the shareholding size held. Dividends may then be paid on a pro rata basis.

The company model is very similar to the investor model. Operations which transform the company model should require a similar transformation of the investor model. These operations include \textit{Create}, \textit{Buy}, \textit{Sell}, and the \textit{Tko} ver operation. This section presents updates to the company model only. However, the investor model should be updated accordingly.

The following operations are considered appropriate for the company viewpoint.

\[ \text{Create}|\text{ListC}|\text{NmrC}|\text{ListS}|\text{TShsC}|\text{Earns}|\text{Divs}|\text{DivS}|\text{Buy}|\text{Sell}|\text{ListI}|\text{Tko}ver \]

The companies listed on the stock exchange is given by the \textit{ListC} operation. The number of companies listed on the exchange is given by the \textit{NmrC} operation.
ListC : C \hookrightarrow (S \hookrightarrow P) \hookrightarrow \mathcal{P} C

ListC[\psi] \triangleq \text{dom} \psi

Nm r C : C \hookrightarrow (S \hookrightarrow P) \hookrightarrow \mathbb{N}
Nm r C[\psi] \triangleq |\text{dom} \psi|

The shareholders in a company is given by the ListS operation. The investors in the exchange is given by the ListI operation.

ListS : C \hookrightarrow (C \hookrightarrow (S \hookrightarrow P)) \hookrightarrow \mathcal{P} P
ListS[c] \psi \triangleq \text{rng} \circ \psi(c)

\text{pre}_\text{ListS} : C \hookrightarrow (C \hookrightarrow (S \hookrightarrow P)) \hookrightarrow \mathbb{B}
\text{pre}_\text{ListS}[c] \psi \triangleq \chi[c] \psi

ListI : C \hookrightarrow (S \hookrightarrow P) \hookrightarrow \mathcal{P} P
ListI[\psi] \triangleq \cup / \circ \mathcal{P} \text{rng} \circ \text{rng} \psi

3.5.2 Company Performance

The total number of shares is determined from the TShsC operation, or from the exchange model from $|\kappa^{-1}(c)|$

TShsC : C \hookrightarrow (C \hookrightarrow (S \hookrightarrow P)) \hookrightarrow \mathbb{N}
TShsC[c] \psi \triangleq |\text{dom} \circ \psi(c)|

\text{pre}_\text{TShsC} : C \hookrightarrow (C \hookrightarrow (S \hookrightarrow P)) \hookrightarrow \mathbb{B}
\text{pre}_\text{TShsC}[c] \psi \triangleq \chi[c] \psi

Earning per share is determined from overall company performance, i.e., profitability per unit share, where performance is based on a specific accounting period. The dividend to be paid to the shareholders in the company is based on the earning per share, generally the company retains a certain proportion of the profits for reinvestment, and the remainder is paid to the shareholders on a pro rata basis.

E ar n s : (C \times \mathbb{Q}) \hookrightarrow (C \hookrightarrow (S \hookrightarrow P)) \hookrightarrow \mathbb{Q}
E ar n s[c, e] \psi \triangleq \frac{e}{TShsC[c] \psi}

67
\[ \text{pre} \_ \text{Earnings} : (C \times Q) \mapsto (C \mapsto (S \mapsto P)) \mapsto B \]
\[ \text{pre} \_ \text{Earnings}[c, e] \psi \triangleq \chi[e] \psi \]

The dividend per share is calculated from the total dividend the company plans to pay, and is given by the \text{Divs} operation. The dividend per shareholder is then determined from the shareholding size held by the individual shareholder, and is given by the \text{Divs} operation.

\[ \text{Divs} : (C \times Q^+) \mapsto (C \mapsto (S \mapsto P)) \mapsto Q^+ \]
\[ \text{Divs}[c, r] \psi \triangleq r \times S_{\text{divs}}[c] \psi \]

\[ \text{pre} \_ \text{Divs} : (C \times Q^+) \mapsto (C \mapsto (S \mapsto P)) \mapsto B \]
\[ \text{pre} \_ \text{Divs}[c, r] \psi \triangleq \chi[c] \psi \]

\[ \text{Divs} : (C \times P \times Q^+) \mapsto (C \mapsto (S \mapsto P)) \mapsto Q^+ \]
\[ \text{Divs}[c, p, r] \psi \triangleq \left( (\psi(c))^{-1}(p) \right) \times \text{Divs}[c, r] \psi \]

\[ \text{pre} \_ \text{Divs} : (C \times P \times Q^+) \mapsto (C \mapsto (S \mapsto P)) \mapsto B \]
\[ \text{pre} \_ \text{Divs}[c, p, r] \psi \triangleq \chi[c] \psi \land p \in \psi(c) \]

### 3.5.3 Buy - Sell Operations

These operations are analogous to the investor model. They involve the \text{Create}, \text{Buy} and \text{Sell} operations which are tailored for the company model. The investor model should be updated accordingly to maintain consistency.

\[ \text{Create} : (S \mapsto C) \mapsto (C \mapsto (S \mapsto P)) \]
\[ \text{Create}[\kappa] \triangleq \lambda c : \{ \text{rng} \kappa \} \ast \lambda s : \{ \kappa^{-1}(c) \} \ast d \]

\[ \text{Buy} : (C \times P \times P'S) \mapsto (C \mapsto (S \mapsto P)) \mapsto (C \mapsto (S \mapsto P)) \]
\[ \text{Buy}[c, p, S] \psi \triangleq \]
\[ \text{Let } \psi_c' = \psi(c) \uparrow \lambda s : S_c \ast p \text{ in } \]
\[ \mapsto \psi \uparrow [c \mapsto \psi_c'] \]

\[ \text{pre} \_ \text{Buy} : (C \times P \times P'S) \mapsto (C \mapsto (S \mapsto P)) \mapsto B \]
\[ \text{pre} \_ \text{Buy}[c, p, S] \psi \triangleq \]
\[ \chi[c] \psi \land \chi[d] \text{rng} \psi(c) \]
\[ \land S_c \subseteq \text{dom} \psi(c) \]
\[ \land \mathcal{P} \psi(c)(S_c) = \{ d \} \]

68
\[ p \neq d \]

\[ Sell : (C \times P \times \mathcal{P}(S)) \mapsto (C \mapsto (S \mapsto P)) \mapsto (C \mapsto (S \mapsto P)) \]

\[ Sell[c, p, S_c] \psi \triangleq \]

Let \( \psi'_c = \psi(c) \uparrow \lambda s : S_c \bullet d \) in

\( \mapsto \psi \uparrow [c \mapsto \psi'_c] \)

\[ \text{pre}_\text{Sell} : (C \times P \times \mathcal{P}(S)) \mapsto (C \mapsto (S \mapsto P)) \mapsto B \]

\[ \text{pre}_\text{Sell}[c, p, S_c] \psi \triangleq \]

\[ \chi[c] \psi \land \chi[p] \text{rng} \psi(c) \land S_c \subseteq \text{dom} \psi(c) \land \mathcal{P} \psi(c)(S_c) = \{p\} \land p \neq d \]

### 3.5.4 The Company Take-over Operation

The next operation to be considered is a predator prey operation in which a company is taken over by another. The operation is quite involved, and has three possible cases; shares being issued on a one to one basis, a one to many basis, and a many to one basis. The \( \text{TkOvr} \) operations involves the following:

1. A merging of shareholders and shares of both companies must take place. The shareholders in the company that is to be taken over must receive shares in the company which is to take them over.

2. The shares which the shareholders receive is directly related to the shareholding previously held, and the share price of both companies.

3. The company being taken over must be removed from the exchange.

The definition of the take-over operation is presented in three stages. First a naive take-over operation is presented, the assumption made is that the share price of both companies is equal. This enables the operation to be defined easily, since there is no necessity to issue further shares, and the shareholders in the company being taken over become shareholders in the take-over company.

\[ \text{TkOvr}_1 : C \times C \mapsto (S \mapsto C) \times (C \mapsto Q^+) \times (C \mapsto (S \mapsto P)) \mapsto (C \mapsto (S \mapsto P)) \]

\[ \text{TkOvr}_1[t_c, c](\kappa, \tau, \psi) \triangleq \]

Let \( \psi' = \exists[c] \psi \uparrow [t_c \mapsto \psi(t_c) \sqcup \psi(c)] \) in

Let \( \kappa' = \kappa \uparrow \lambda s : \{\kappa^{-1}(c)\} \bullet t_c \) in

69
Let \( \tau' = \llbracket [c] \rrbracket \tau \) in 
\( \mapsto (\kappa', \tau', \psi') \)

\[ \text{pre}_{TkOvr_1} : C \times C \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \times (C \mapsto (S \mapsto P)) \mapsto (C \mapsto (S \mapsto P)) \mapsto B \]
\[ \text{pre}_{TkOvr_1}[t_c, c][\kappa, \tau, \psi] = \chi[c] \psi \wedge \chi[t_c] \psi \\
\wedge \kappa = \tau(t_c) \]

**Comment 3.1 (Tackling an Easier Sub-problem)** The \( \text{TkOvr} \) definition is correct when the share prices of both companies are equal. The general case is more involved, requiring the issuing of new shares in the take-over company, equivalent in value to the company being taken over. The shares are then distributed on a pro rata basis to the shareholders of the company being taken over.

**Comment 3.2 (m shares for 1 share)** If the quotient \( \frac{\tau(t_c)}{\tau(t_c)} \) is greater than 1, then several shares in the take-over company will be received for each share in the company being taken over.

\[ \text{TkOvr}_2 : C \times C \times \mathbb{N} \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \times (C \mapsto (S \mapsto P)) \mapsto (C \mapsto (S \mapsto P)) \]
\[ \text{TkOvr}_2[t_c, c, m][\kappa, \tau, \psi] = \]
\[ \text{Let } S_c = \text{dom } \psi(c) \text{ in } \]
\[ \text{Let } N_s = |S_c| \times m \text{ in } \]
\[ \text{Let } Shs \in \llbracket [v] / \circ \text{rng } (I \mapsto \text{dom } \psi) \rrbracket S \wedge |Shs| = N_s \text{ in } \]
\[ \text{Let } \psi' = \llbracket [c] \psi \rrbracket t_c \mapsto \psi(t_c) \cup \lambda s : Shs \bullet \psi_s(f(s)) \text{ in } \]
\[ \text{Let } \kappa' = \llbracket [k^{-1}(c)] \rrbracket k \cup \lambda s : Shs \bullet t_c \text{ in } \]
\[ \text{Let } \tau' = \llbracket [c] \rrbracket \tau \text{ in } \]
\( \mapsto (\kappa', \tau', \psi') \)

The definition of the take-over operation makes reference to an undefined function \( f : Shs \mapsto S_c; \) this function associates \( m \) shares with each share in the original company. The function is many to one, i.e., \( m \) to one.

\[ \text{pre}_{TkOvr_2} : C \times C \times \mathbb{N} \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \times (C \mapsto (S \mapsto P)) \mapsto B \]
\[ \text{pre}_{TkOvr_2}[t_c, c, m][\kappa, \tau, \psi] = \chi[c] \wedge \chi[t_c] \wedge m > 1 \]

**Comment 3.3 (1 share for m shares)** If the quotient \( \frac{\tau(t_c)}{\tau(t_c)} \) is less than 1, then several shares in the company being taken over will be required for a single share in the take-over company.

\[ \text{TkOvr}_3 : C \times C \times \mathbb{N} \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \times (C \mapsto (S \mapsto P)) \mapsto (C \mapsto (S \mapsto P)) \]
\[ \text{TkOvr}_3[t_c, c, m][\kappa, \tau, \psi] = \]

70
Let $S_c = \text{dom} \psi_c$ in
Let $N_s = \left\lceil \frac{|S_c|}{m} \right\rceil$ in
Let $Shs \in \comprehend{\mathbb{C}} / \circ \text{rng} (\mathcal{I} \mapsto \text{dom} \psi) S \cap |Shs| = N_s$ in
Let $\psi' = \comprehend{\mathbb{C}} / \circ \text{rng} (\mathcal{I} \mapsto \psi(t_c) \cap \lambda s : Shs \bullet \psi_c(g^{-1}(s)))$ in
Let $\kappa' = \comprehend{\mathbb{C}} / \circ \text{rng} (\mathcal{I} \mapsto \psi(t_c) \cap \lambda s : Shs \bullet t_c$ in
Let $\tau' = \comprehend{\mathbb{C}} / \circ \text{rng} (\mathcal{I} \mapsto \psi(t_c) \cap \lambda s : Shs \bullet t_c$ in
\[
\mapsto (\kappa', \tau', \psi')
\]

The definition of the take-over operation makes reference to an undefined functions $g : S_c \mapsto Shs$. This function associates $m$ shares in the original company with a single share in the take-over company, and is many to one (i.e., $m$ to one) as before. The use of $\psi_c(g^{-1}(s))$ is an abuse of notation; the correct formulation is $\psi_c(s')$, where $s' \in g^{-1}(s)$.

The precondition for the operation requires that all shareholders in the company to be taken over have some multiple of $m$ shares.

\[
\begin{align*}
\text{pre}_T kOvr3 : C \times C \times \mathbb{N} & \mapsto (S \mapsto C) \times (C \mapsto \mathbb{Q}^+) \times (C \mapsto (S \mapsto P)) \mapsto \mathbb{B} \\
\text{pre}_T kOvr3[t_c, c, m] & (\kappa, \tau, \psi) \triangleq \\
& \chi[c] \kappa \land \chi[t_c] \kappa \\
& \land m > 1 \\
& \mathcal{P}_{\text{card}} \circ \text{rng} \circ (\psi(c))^{-1} \subseteq mN
\end{align*}
\]

### 3.5.5 Invariant preservation

**Lemma 3.9** $\psi = \text{Create}\mathbb{C}[\kappa] \Rightarrow \text{Inv}_C\text{Mdl}[\kappa, \tau, \psi].$

**Proof**
There are two parts to the invariant. Each part is proved separately.

\[
\psi = \lambda c : \{\text{rng} \kappa\} \bullet \lambda s : \{\kappa^{-1}(c)\} \bullet d \\
\Rightarrow (\mathcal{I} \mapsto \text{dom} \psi) = \lambda c : \{\text{rng} \kappa\} \bullet \{\kappa^{-1}(c)\} \\
= \kappa^{-1}
\]
as required.

\[
\text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ [(\mathcal{I} \mapsto \text{dom} \psi)]^{-1} \\
= \text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ [(\mathcal{I} \mapsto \text{dom} \lambda c : \{\text{rng} \kappa\} \bullet \lambda s : \kappa^{-1}(c) \bullet d)]^{-1} \\
= \text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ (\lambda c : \{\text{rng} \kappa\} \bullet \kappa^{-1}(c) \bullet d)^{-1} \\
= \text{rng} \circ (\mathcal{I} \mapsto \text{card}) \circ (\lambda c : \{\text{rng} \kappa\} \bullet \kappa^{-1}(c))^{-1} \\
= \{1\}
\]

71
as required (where \( \kappa \neq \theta \)).

**Lemma 3.10**  
\( \text{pre}_{\text{Buy}}[c, p, S_c] \psi \land \psi' = \text{Buy}[c, p, S_c] \psi \Rightarrow \text{Inv}_C \text{Mdl}[\kappa, \tau, \psi'] \).

**Proof**

\[
(I \mapsto \text{dom}) \psi' \\
= (I \mapsto \text{dom})(\psi \uparrow [c \mapsto \psi'_c \triangleright \lambda s : S_c \bullet p]) \\
= (I \mapsto \text{dom})(\llbracket c \rrbracket \psi \cup [c \mapsto \psi'_c \triangleright \lambda s : S_c \bullet p]) \\
= (I \mapsto \text{dom})(\llbracket c \rrbracket \psi \cup [c \mapsto \text{dom}(\psi'_c \triangleright \lambda s : S_c \bullet p)]) \\
= \llbracket \psi_c \rrbracket (I \mapsto \text{dom}) \psi \cup [c \mapsto \llbracket S_c \rrbracket \text{dom} \psi_c \cup S_c] \\
= \llbracket \psi_c \rrbracket (I \mapsto \text{dom}) \psi \cup [c \mapsto \text{dom} \psi_c] \\
= (I \mapsto \text{dom}) \psi \\
= \kappa^{-1}
\]

as required.

\[
\text{rng} \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom}) \psi']^{-1} \\
= \text{rng} \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom})(\llbracket c \rrbracket \psi \cup [c \mapsto \psi'_c])]^{-1} \\
= \text{rng} \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom})(\llbracket c \rrbracket \psi \cup [c \mapsto \text{dom} \psi'_c])]^{-1} \\
= \text{rng} \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom})(\llbracket c \rrbracket \psi \cup [c \mapsto \text{dom}(\psi'_c \triangleright \lambda s : S_c \bullet p)])^{-1} \\
= \text{rng} \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom})(\llbracket c \rrbracket \psi \cup [c \mapsto \text{dom} \psi_c])]^{-1} \\
= \text{rng} \circ (I \mapsto \text{card}) \circ [(I \mapsto \text{dom}) \psi]^{-1} \\
= \{1\}
\]

as required.

**Lemma 3.11**  
\( \text{pre}_{\text{Sell}}[c, p, S_c] \psi \land \psi' = \text{Sell}[c, p, S_c] \psi \Rightarrow \text{Inv}_C \text{Mdl}[\kappa, \tau, \psi'] \).

**Proof**

\[
(I \mapsto \text{dom}) \psi' \\
= (I \mapsto \text{dom})(\psi \uparrow [c \mapsto \psi_c \triangleright \lambda s : S_c \bullet d]) \\
= (I \mapsto \text{dom})(\llbracket c \rrbracket \psi \cup [c \mapsto \psi_c \triangleright \lambda s : S_c \bullet d]) \\
= (I \mapsto \text{dom})(\llbracket c \rrbracket \psi \cup [c \mapsto \text{dom}(\psi_c \triangleright \lambda s : S_c \bullet d)]) \\
= \llbracket \psi_c \rrbracket (I \mapsto \text{dom}) \psi \cup [c \mapsto \llbracket S_c \rrbracket \text{dom} \psi_c \cup S_c] \\
= \llbracket \psi_c \rrbracket (I \mapsto \text{dom}) \psi \cup [c \mapsto \text{dom} \psi_c] \\
= (I \mapsto \text{dom}) \psi \\
= \kappa^{-1}
\]

as required.

The proof of the second part of the invariant is similar to the corresponding proof of invariant satisfaction for the Buy operation.
3.6 Financial Operations

Share purchase or sale involve a financial payment by one of the parties in the transaction. The buyer of the shares pays the dealer the market value of the shares; this involves crediting the dealers account, and debiting the buyers account, in effect a simple banking operation. The key constraint is that sufficient funds must be in the buyer’s account in order to complete the financial transaction.

Banking models have been presented elsewhere in the literature; for example, [69], page 281. The objective here is to present a simple banking model which may then be applied to model the financial operations in the stock exchange.

\[
\alpha : P \mapsto A \quad |\text{rng} \alpha| = |\text{dom} \alpha| \quad (3.8)
\]

\[
\beta : A \mapsto Q \quad (3.9)
\]

\[
\gamma : A \mapsto Q^* \quad (3.10)
\]

\[
\text{Inv}_{\text{Bnk}}(P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto B
\]

\[
\text{Inv}_{\text{Bnk}}[\alpha, \beta, \gamma] 
\]

\[
\text{rng} \alpha = \text{dom} \beta = \text{dom} \gamma
\]

\[
\text{and} (I \mapsto \gamma) \gamma = \beta
\]

The relationship between customers and accounts is given by \( \alpha \). There is a restriction to exactly one account per customer in the model. The relationship between accounts and balances is given by \( \beta \). Finally, \( \gamma \) models the transaction history of the accounts. The current balance in each account must equal the accumulated sum of the transactions on the account.

3.6.1 Banking Operations

\[
\text{Ops} = \text{Crea}|\text{Add}|\text{Rem}|\text{Dep}|\text{Wrtd}|\text{List}|\text{Trns}
\]

\[
\text{Crea} \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*)
\]

\[
\text{Crea} \triangleq (\theta, \theta, \theta)
\]

The \text{Crea} operation creates an empty banking system; the \text{Add} operation is used to open up a customer account in the system.

\[
\text{Add} : (P \times A \times Q^+) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*)
\]

73
\[ Add[p, a, b] (\alpha, \beta, \gamma) \triangleq (\alpha \cup \{ p \mapsto a \}, \beta \cup \{ a \mapsto b \}, \gamma \cup \{ a \mapsto (b) \}) \]

\[ \text{pre}_{\text{Add}} : (P \times A \times Q) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto B \]
\[ \text{pre}_{\text{Add}}[p, a, b] (\alpha, \beta, \gamma) \triangleq \]
\[ \neg \chi[p] \alpha \]
\[ \land \neg \chi[a] \beta \]
\[ \land \neg \chi[a] \gamma \]

The remove operation \( Rem \) is the inverse of the \( Add \) operation, and its effect is to close a customer's account, and to remove the customer from the bank.

\[ Rem : P \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \]
\[ Rem[p](\alpha, \beta, \gamma) \triangleq (\varepsilon[p] \alpha, \varepsilon[\alpha(p)] \beta, \varepsilon[\alpha(p)] \gamma) \]

\[ \text{pre}_{\text{Rem}} : P \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto B \]
\[ \text{pre}_{\text{Rem}}[p] (\alpha, \beta, \gamma) \triangleq \chi[p] \alpha \]

The effect of the \( Dep \) and \( Wrd \) operations on a customer’s account is to make a deposit or withdrawal on the account. The account balance is updated by the amount of the transaction, and the transaction amount is appended to the transaction history of the account.

\[ Dep : (P \times Q^+) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \]
\[ Dep[p, q](\alpha, \beta, \gamma) \triangleq (\alpha, \beta \oplus [\alpha(p) \mapsto q], \gamma \ominus [\alpha(p) \mapsto \{ q \}]) \]

\[ \text{pre}_{\text{Dep}} : (P \times Q^+) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto B \]
\[ \text{pre}_{\text{Dep}}[p, q] (\alpha, \beta, \gamma) \triangleq \chi[p] \alpha \]

Note: The \( \ominus \) operation is an indexed concatenation operation. The indexed monoid inherits its behaviour from the underlying base monoid and is defined in [47]. Extensions to more general indexed structures are defined in Appendix B of this thesis. These extensions are believed to be original.

\[ Wrd : (P \times Q^+) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \]
\[ Wrd[p, q](\alpha, \beta, \gamma) \triangleq (\alpha, \beta \ominus [\alpha(p) \mapsto q], \gamma \ominus [\alpha(p) \mapsto \{ q \}]) \]

\[ \text{pre}_{\text{Wrd}} : (P \times Q^+) \mapsto (P \mapsto A) \times (A \mapsto Q) \times (A \mapsto Q^*) \mapsto B \]
\[ \text{pre}_{\text{Wrd}}[p, q] (\alpha, \beta, \gamma) \triangleq \]

74
\[
\chi[p]_\alpha \\
\beta(\alpha(p)) \geq q
\]

\[\text{List} : (P \mapsto A) \mapsto \mathcal{P} \]
\[\text{List}[\alpha] \triangleq \text{dom } \alpha\]

\[\text{Trans} : P \mapsto (P \mapsto A) \times (A \mapsto Q^*) \mapsto Q^*\]
\[\text{Trans}[p](\alpha, \gamma) \triangleq \gamma(\alpha(p))\]

### 3.6.2 Preservation of Invariant

Several of the operations including \textit{Crea}, \textit{Add}, \textit{Rem}, \textit{Dep}, \textit{Wrd} transform the banking system; there is an obligation to demonstrate that the invariant is preserved in each case.

**Lemma 3.12** \(\text{pre}_\text{Crea} \land \text{Crea} \Rightarrow \text{Inv}_\text{Bnk}[\theta, \theta, \theta]\)

**Proof**

This is immediate since \(\text{rng } \theta = \text{dom } \theta\) and \((\mathcal{I} \mapsto^+ \theta) \theta = \theta\).

**Lemma 3.13** \(\text{pre}_\text{Add}[p, a, b](\alpha, \beta, \gamma) \land \text{Add}[p, a, b](\alpha, \beta, \gamma) \Rightarrow \text{Inv}_\text{Bnk}[\alpha', \beta', \gamma']\)

**Proof**

\[\text{rng } \alpha'\]
\[= \text{rng } (\alpha \cup [p \mapsto a])\]
\[= \text{rng } \alpha \cup \{a\}\]
\[= \text{dom } \beta \cup \{a\}\]
\[= \text{dom } \gamma \cup \{a\}\]
\[= \text{dom } (\beta \cup [a \mapsto b])\]
\[= \text{dom } (\gamma \cup [a \mapsto b])\]

\[(\mathcal{I} \mapsto^+ \gamma')\]
\[= (\mathcal{I} \mapsto^+ \gamma \cup [a \mapsto b])\]
\[= (\mathcal{I} \mapsto^+ \gamma \cup (\mathcal{I} \mapsto^+ [a \mapsto b]))\]
\[= \beta \cup (\mathcal{I} \mapsto^+ [a \mapsto b])\]
\[= \beta \cup [a \mapsto b]\]
\[= \beta'\]

as required.

**Lemma 3.14** \(\text{pre}_\text{Rem}[p](\alpha, \beta, \gamma) \land \text{Rem}[p](\alpha, \beta, \gamma) \Rightarrow \text{Inv}_\text{Bnk}[\alpha', \beta', \gamma']\)
Proof

\[ \text{rng } \alpha' \]
\[ = \text{rng } \alpha \]
\[ = \{ \alpha(p) \} \text{rng } \alpha \]
\[ = \{ \alpha(p) \} \text{dom } \beta \]
\[ = \text{dom } \{ \alpha(p) \} \beta \]
\[ = \text{dom } \beta' \]
\[ = \text{dom } \gamma' \]

\[ \{ \alpha(p) \} \beta \cup \{ \alpha(p) \beta (\alpha(p)) \} \]
\[ = \beta \]
\[ = (I \mapsto ^+ /) \gamma' \]
\[ = (I \mapsto ^+ /)(\{ \alpha(p) \} \beta \cup \{ \alpha(p) \beta (\alpha(p)) \}) \]
\[ = (I \mapsto ^+ /)(\{ \alpha(p) \} \beta \cup (I \mapsto ^+ /)(\alpha(p) \beta (\alpha(p)) \gamma(q))) \]
\[ = (I \mapsto ^+ /)(\{ \alpha(p) \} \beta \cup (I \mapsto ^+ /)(\alpha(p) \beta (\alpha(p)) \gamma(q))) \]
\[ = (I \mapsto ^+ /)(\{ \alpha(p) \} \beta \cup (I \mapsto ^+ /)(\alpha(p) \beta (\alpha(p)) + q)) \]
\[ = \beta' \]

Lemma 3.15 \( \text{pre}_\text{Dep}[p,q](\alpha, \beta, \gamma) \wedge \text{Dep}[p,q](\alpha, \beta, \gamma) \Rightarrow \text{Inv}_\text{Bnk}[\alpha', \beta', \gamma'] \)

Proof

Clearly \( \text{rng } \alpha' = \text{dom } \beta' = \text{dom } \gamma' \).

\[ (I \mapsto ^+ /) \gamma' \]
\[ = (I \mapsto ^+ /)(\{ \alpha(p) \} \beta \cup (I \mapsto ^+ /)(\alpha(p) \beta (\alpha(p)) \gamma(q))) \]
\[ = (I \mapsto ^+ /)(\{ \alpha(p) \} \beta \cup (I \mapsto ^+ /)(\alpha(p) \beta (\alpha(p)) + q)) \]
\[ = (I \mapsto ^+ /)(\{ \alpha(p) \} \beta \cup (\{ \alpha(p) \beta (\alpha(p)) \gamma(q))) \]
\[ = (I \mapsto ^+ /)(\{ \alpha(p) \} \beta \cup (\{ \alpha(p) \beta (\alpha(p)) \gamma(q))) \]
\[ = \beta' \]

Lemma 3.16 \( \text{pre}_\text{Wrd}[p,q](\alpha, \beta, \gamma) \wedge \text{Wrd}[p,q](\alpha, \beta, \gamma) \Rightarrow \text{Inv}_\text{Bnk}[\alpha', \beta', \gamma'] \)

Proof

The proof is similar to the \( \text{Dep} \) operation.
3.6.3 Financial operations in Stock Exchange

The purchases or sale of shares involves payment by one counterparty to the to the other counterparty. The Buy and Sell operations model share purchase and sale, and must include functionality for payment for the shares. The operations are sketched as follows.

\[
\text{Buy}[[c, p, S_c](\psi, \alpha, \beta, \gamma)]
\]

\[
\text{Let } q = |S_c| * \tau(c) \text{ in }
\]

\[
\psi'_c \mapsto \psi_c \upharpoonright \lambda s : S_c \cdot p
\]

\[
(\alpha', \beta', \gamma') \mapsto \text{Dep}[d, q] \circ \text{Wrd}[p, q](\alpha, \beta, \gamma)
\]

\[
\psi' \mapsto \psi \upharpoonright [c \mapsto \psi'_c]
\]

\[
\mapsto (\psi', \alpha', \beta', \gamma')
\]

The Sell operation is sketched similarly.

3.7 Summary

The objective of this chapter is to demonstrate that formal methods may be employed to model aspects of the structure and behaviour of an organization in the real world. A stock exchange is one particular instantiation of an organization. This chapter demonstrates how a model of a stock exchange may be constructed.

The properties and behaviour of aspects of the stock exchange are formally encoded in the model. This enables a highly concise and precise statement of the behaviour and properties of the stock exchange to be made. The fact that the model is constructive indicates that it may be implemented if this is required. However, since stock exchanges already exist in the real world the model is not developed for implementation purposes. Instead, the objective is to demonstrate that aspects of the behaviour of a stock exchange may be encoded in a formal model.

The model considers the problem of modelling companies registered on the stock exchange, the shareholders in the company, and the financial operations in the exchange. Several assumptions are made in the model, for example, it is assumed that there is one dealer who is the intermediary for all share purchase or sale operations. The model is adequate at explaining part of the behaviour of the stock exchange, and is inadequate at explaining other aspects. The following summarizes the strengths and weaknesses of the model.

- The model demonstrates that aspects of the properties and behaviour of a stock exchange may be formally encoded in a model.
• The three conceptual viewpoints in the model provide a good abstract representation of the organizational structure and behaviour of the stock exchange.

• The model captures aspects of the behaviour of companies and shareholders and explains the effect of share purchase or sale operations.

• The limitation to one dealer is unrealistic. In practice, several stockbroking firms exist in the market place.

• There is no theory or explanation within the model as to why share prices change. Furthermore, the model does not assist in predicting future share prices.

• The profit or loss incurred by an investor on share purchase or sale operations is not considered.

• Broker commission is not considered.

• The financial operations associated with share purchase and sale are sketched.

• The definition of the company take-over operation demonstrates the validity of the approach of solving a complex problem by first solving several easier sub-problems.

• The initial formulation of the model and invariant lead to quite involved proof obligations. The re-formulation to a simpler model and invariant had a corresponding effect on the complexity of the proof obligations. This demonstrates the importance of working with the right model.
Chapter 4

Bill of Materials

4.1 Introduction

Chapter 2 has demonstrated that formal methods may be employed to model aspects of the organizational structure for the domain of religion. The study of the religion model indicated a generic aspect to modelling. It was concluded that the religion model may be adapted to model the hierarchical structure of companies, political institutions and a university. Chapter 3 demonstrates that VDM* may be employed to develop a model of a stock exchange. The stock exchange is one particular instantiation of an organization.

The fact that the religion model has proved to be effective in modelling the hierarchical structure of organizations indicates the importance of gaining a more detailed understanding of hierarchies. Consequently, the objective of this chapter is to examine the essential structure of hierarchies. The standard abstraction of hierarchies is the bill of material structure. The formal specification of the bill of materials [12] is a classic in formal methods. An alternate approach to the specification of the bill of material is described in terms of the annihilator function described in [46].

This chapter provides a more complete presentation of the material in [46]. In particular, this chapter places the annihilator function on a sounder mathematical platform, and it is proved that a structure is a bill of material if and only if \( A_\mu = \theta \).

The chapter includes a number of results which are believed to be new. This includes the shrinking operation \( \beta \) which shrinks a bill of material but preserves it as a bill of material structure. This operation is derived from the annihilator function. Secondly, the problem of joining two bill of materials is considered. The key constraint in joining two bill of materials is that they should be consistent. This is considered in Section 4.6.

The formulation of an invariant has an effect on the complexity of the proof obligations. This has been remarked upon previously in Chapter 3. The classic Björner invariant is compared against the annihilator function to determine the effectiveness in simplifying the proof obligations.

79
The style of proof exhibited in this chapter is a mixture of the constructive style of the Irish school and informal mathematical proof. The advantages of informal proofs is that they are easier to read than their formal counterparts. However, they suffer from the disadvantage that they are error prone. The use of the informal proofs in this chapter is a deviation from the VDM*. However, it is often difficult to be absolutely certain that a proof is correct. The argument against absolute certainty in mathematical proof has been made by Lakatos in [41]. Similarly, the re-examination of a theorem on parts explosion presented originally in [46] demonstrates that the statement of the theorem is incorrect.

4.2 The Annihilator Function

The classic invariant for the bill of material as adopted from [12] is given by an all recorded part, and a prohibition of cycles part, defined as follows:

\[ Inv_{BO}M : (X \mapsto \mathcal{P}X) \mapsto B \]
\[ inv_{BO}M[\gamma] \triangleq \]
\[ (\forall S \in \text{rng} \gamma)(S \subseteq \text{dom} \gamma) \]
\[ \land (\forall x \in \text{dom} \gamma)((x \not\in \text{Parts}(x, \gamma))) \]

the \( \text{Parts} \) operation determined the sub-parts of an assembly, the constraint in the invariant is that no assembly may contain itself as a sub-assembly. The \( \text{Parts} \) operation is defined as follows:

\[ \text{Parts} : X \times (X \mapsto \mathcal{P}X) \mapsto \mathcal{P}X \]
\[ \text{Parts}(x, \gamma) \triangleq \]
\[ \text{Let } S = \{x' | x' \in \text{dom} \gamma : x' \in \gamma(x) \]  
\[ \lor (\exists x^\prime \in S, x' \in \gamma(x^\prime)) \} \text{ in } S \]

MacAnAirechinnigh [45] observed that for a non empty bill of material the all recorded part of the invariant can be more accurately expressed by the constraint:

\[ \cup/\text{rng} \gamma \subset \text{dom} \gamma \]  

(4.1)

The original invariant specified by Bjørner employed the improper subset, however, the proper subset is more appropriate and accurate if dealing with a non-empty bill of material, as equality cannot arise in this case. The stronger constraint which must be satisfied for non-trivial bill of materials is important in real specifications, where total accuracy and precision is essential.
The re-examination by MacAnAirchinnigh of the original bill of material specification in [46] identified the annihilator function, which allows an elegant stipulation of the bill of material invariant. Before presenting this material, an elementary property of bills of material is presented. The property states that a non-empty bill of material has at least one basic part; essentially this is equivalent to the statement that a non-empty tree has at least one leaf. The proof provided is non-constructive, based on informal mathematical reasoning.

**Lemma 4.1** If $\gamma \in X \mapsto \mathcal{P}X$ is a bill of material, $\gamma \neq \emptyset$ then $\exists x \in \text{dom } \gamma$ such that $\gamma(x) = \emptyset$, i.e., $\gamma$ has at least one basic part.

**Proof** (Informal + formal)

*Informal* This is obvious since a bill of material consists of either parts or assemblies. These assemblies are themselves composed of either sub-assemblies or basic parts, and so on. Ultimately, we must reach one basic part.

However, the informal proof begs the question; its argument is based on assemblies and parts, rather than the mathematical model of a bill of material. The formal proof uses the properties of the mathematical model of a bill of material to derive the result. The structure of the proof is to assume that there is no basic part in the bill of material structure, and to derive a contradiction from this, i.e., *reductio ad absurdum*.

*Formal* Suppose $\gamma$ is a bill of material, such that $\gamma(x) \neq \emptyset$, $\forall x \in \text{dom } \gamma$, i.e., there are no basic parts. Then consider arbitrary $x \in \text{dom } \gamma$ and consider the sequence $S_n$ defined by:

$$
S_1 = \gamma(x)
$$

$$
S_2 = S_1 \cup \gamma(S_1) = \gamma(x) \cup \gamma^2(x)
$$

$$
S_3 = S_2 \cup \gamma(S_2) = \gamma(x) \cup \gamma^2(x) \cup \gamma^3(x)
$$

$$
\vdots
$$

$$
S_n = S_{n-1} \cup \gamma(S_{n-1}) = \gamma(x) \cup \gamma^2(x) \cup \gamma^3(x) \cup \ldots \cup \gamma^n(x)
$$

Then $S_n$ is a monotonic increasing sequence which is bounded above by $\text{dom } \gamma$.

Thus the sequence $S_n$ is convergent, and converges to $S$, where $S = \{x_1, x_2, \ldots, x_m\}$ say, and $S = S \cup \gamma(S)$. Thus $\gamma(S) \subseteq S \Rightarrow \gamma(\{x_1, x_2, \ldots, x_m\}) \subseteq (\{x_1, x_2, \ldots, x_m\})$, in fact, $S$ is simply the transitive closure of $x$ where we consider $\gamma$ as a relation, i.e., $S = \{z | x R^+ z\}$.

The transitive closure of a binary relation $R$ is given by $R^+ = \cup_{n=1}^{\infty} R^n$. It follows that:

$$
\gamma(x_1) \subseteq \{x_1, x_2, \ldots, x_m\}
$$

$$
\gamma(x_2) \subseteq \{x_1, x_2, \ldots, x_m\}
$$

$$
\vdots
$$

$$
\gamma(x_m) \subseteq \{x_1, x_2, \ldots, x_m\}
$$

81
Consider \( x_i \in \gamma(x_1, x_2, \ldots, x_m) \) then we shall produce a sequence of subparts of \( x_n \) that violates the prohibition of cycles property of the invariant; this will follow since \( S \) is finite.

\[
\begin{align*}
x_i & \in \gamma(x_k) \text{ say} \\
x_i & \in \gamma(x_i) \text{ is impossible as this violates the invariant.} \\
x_k & \in \gamma(x_i) \text{ is impossible since this violates the invariant.}
\end{align*}
\]

Thus \( x_i \in \gamma(x_i) \) where \( x_i \neq x_i \) and \( x_i \neq x_k \).

Considering \( \gamma(x_i) \) we obtain,

\[
\begin{align*}
x_i & \in \gamma(x_i) \text{ is impossible, since } x_i \in \gamma(x_i) \text{ since this would violate the prohibition of cycles.} \\
x_k & \in \gamma(x_i) \text{ is impossible, since we have } x_i \in \gamma(x_i) \land x_i \in \gamma(x_k) \text{ and thus } x_i \text{ is a sub-part of } x_k \text{ and thus the prohibition of cycles would be violated.} \\
x_i & \in \gamma(x_i) \text{ is impossible}
\end{align*}
\]

Thus an \( x_m \) distinct from \( x_i, x_k, x_l \) is needed such that \( x_m \in \gamma(x_l) \).

Proceeding in this manner, we first observe that for \( n \) elements, \( x_i, x_k, x_l, \ldots, x_n \), an element \( x_{n+1} \) is needed, distinct from \( x_i, x_k, x_l, \ldots, x_n \) and such that \( x_{n+1} \notin \gamma(x_i), x_{n+1} \notin \gamma(x_k), \ldots \) etc. and such that \( x_{n+1} \in \gamma(x_n) \).

Since \( x_1, x_2, x_3, \ldots, x_m \) is finite it means that eventually \( x_{n+1} = x_j \) where \( x_j \) is some previous member in the sequence. This means that \( x_j \in \gamma(x_n) \) and by the construction of the sequence \( x_i, x_k, x_l, \ldots \) we have \( x_n \) is a sub-part of \( x_j \).

This contradicts the fact that \( \gamma \) is a bill of material, thus implying our original assumption that \( \gamma(x) \neq \emptyset \forall x \in \text{dom } \gamma \) is false, thus \( \exists y \) such that \( \gamma(y) = \emptyset \).

**Note**

Notation is abused in the proof; \( \gamma \) operates on an element \( x \in X \); however, in the proof \( \gamma \) is applied to elements and sets. Strictly speaking, \( \mathcal{P} \gamma \) should be employed.

### 4.2.1 The shrinking operator: \( \beta \)

In this section a shrinking operation \( \beta \) is defined; the annihilator function is then defined as the limit of successive applications of the shrinking operator. \( \beta \) is defined as follows:

\[
\beta : (X \mapsto \mathcal{P}X) \mapsto (X \mapsto \mathcal{P}X)
\]

\[
\beta(\theta) \triangleq \theta
\]
\[
\begin{align*}
\beta(\gamma) & \triangleq \\
\gamma^{-1}(0) & = \emptyset \\
\mapsto & \gamma \\
\mapsto (I & \mapsto \mathbb{S}[\gamma^{-1}(0)]) \circ \mathbb{S}[\gamma^{-1}(0)]
\end{align*}
\]

\(\beta\) has the interesting property in that it shrinks a bill of material but preserves the property that the transformed structure is a bill of material. Furthermore, a partial ordering \(\ll\) may be defined on bill of materials, and it will be proved that \(\beta(\mu) \ll \mu\), with respect to the partial ordering.

**Theorem 4.1** Suppose \(\gamma\) is a bill of material and \(\beta\) is defined as above, then \(\beta(\gamma)\) is a bill of material.

**Proof**

Let \(\gamma' = \beta(\gamma)\), then from the definition of \(\beta\), it is clear that \(\gamma'\) is one of the following:

\[
\begin{align*}
\gamma' & = \emptyset, \text{ clearly a bill of material.} \\
\gamma' & = \gamma, \text{ clearly a bill of material.} \\
\gamma' & = (I \mapsto \mathbb{S}[\gamma^{-1}(0)]) \circ \mathbb{S}[\gamma^{-1}(0)]
\end{align*}
\]

Suppose \(\gamma'\) is not a bill of material, then either (a) or (b) of Björner’s invariant fails to hold. Suppose part (a) fails to hold, then \(\exists S \in \text{rng} \gamma'\) such that \(\neg (S \subseteq \text{dom} \gamma')\). Thus \(\exists x \in \text{rng} \gamma'\) such that \(x \notin \text{dom} \gamma'\).

\(\gamma\) is a bill of material, hence for \(x \in \text{rng} \gamma\) we have \(x \in \text{dom} \gamma\). Clearly \(x \in \text{rng} \gamma' \Rightarrow x \in \text{rng} \gamma\). By the construction of \(\gamma'\), it is clear that from \(x \notin \text{dom} \gamma' \land x \in \text{dom} \gamma\) we can deduce that \(x \in \gamma^{-1}(0)\), that is, \(\gamma(x) = \emptyset\). However, by the construction of \(\gamma'\), \(\gamma(x) = \emptyset \Rightarrow (x \notin \text{rng} \gamma')\) which is a contradiction; thus part (a) of the invariant must be satisfied.

Suppose the prohibition of cycle part of the invariant fails to hold for \(\gamma'\), then there is an \(x \in \text{dom} \gamma'\) such that \(x \in \text{Parts}(x, \gamma')\). By the construction of \(\gamma'\) it is clear that \(\text{Parts}(x, \gamma') \subseteq \text{Parts}(x, \gamma)\). Thus \(x \in \text{Parts}(x, \gamma') \Rightarrow x \in \text{Parts}(x, \gamma)\), which is a contradiction, since \(\gamma\) is a bill of material.

Thus if \(\gamma\) is a bill of material then \(\beta(\gamma)\) is a bill of material; inductively \(\beta^n(\gamma)\) is a bill of material.

**Theorem 4.2** Suppose \(\gamma\) is a bill of material and \(\beta\) is defined as above, then \(\beta^n(\gamma)\) is a bill of material.

83
**Proof (by induction)**

The basis case is \( n = 0 \) where we have \( \beta^0(\gamma) = \gamma \) and \( \gamma \) is a bill of material.

Suppose \( \beta^k(\gamma) \) is a bill of material, for some \( k \), to show \( \beta^{k+1} \) a bill of material.

By previous theorem, given that \( \beta^k \) is a bill of material it follows that \( \beta(\beta^k(\gamma)) \) is a bill of material; thus \( \beta^{k+1}(\gamma) \) is a bill of material. Thus by induction we have \( \beta^n \) is a bill of material \( \forall n \in \mathbb{N} \).

The annihilator function is introduced in [46], page 201 and is an elegant approach to defining necessary and sufficient conditions for a structure \( \gamma \) to be a bill of material. However, there is no proof provided in [46] that the annihilator constraint identified is mathematically correct. The result is far from obvious and is essential for the validity of the arguments in [46]. The annihilator function is defined and mathematically justified here.

\[
\mathcal{A} : (X \mapsto \mathcal{P}X) \mapsto (X \mapsto \mathcal{P}X)
\]

\[
\mathcal{A}(\theta) \triangleq \theta \\
\mathcal{A}(\gamma) \triangleq \\
\gamma_0^{-1} = \emptyset \\
\mapsto \gamma \\
\mapsto \mathcal{A}(((\mathcal{I} \mapsto \downarrow[\gamma_0^{-1}]) \circ \downarrow[\gamma_0^{-1}]) (\gamma))
\]

It follows from the definition of the annihilator function that given a bill of material \( \gamma \), then \( \mathcal{A}\gamma = \beta^n(\gamma) \) for some \( n \in \mathbb{N} \). This is clear since the effect of the annihilator function is to produce a partially ordered sequence, \( P_\gamma = \gamma = \beta^0(\gamma) \gg \beta^1(\gamma) \gg \beta^2(\gamma) \gg \ldots \) where the partial ordering \( \gg \) is defined by \( f \gg g \iff \text{dom } f \supseteq \text{dom } g \land f(x) \supseteq g(x) \ \forall x \in \text{dom } g \).

The sequence \( P_\gamma \) is monotonic decreasing and is bounded below by \( \theta \), thus \( \beta^n(\gamma) = \beta^k(\gamma) \) \( \forall k \geq n \), \( P_\gamma \) has limit \( \mu = \beta^n(\gamma) \) say.

The next theorem demonstrates that the annihilator function may be used in specifying the invariant for a bill of materials, and the investigation of the properties of the annihilator function demonstrates that \( \gamma \) is a bill of materials if and only if \( \mathcal{A}(\gamma) = \theta \).

**Theorem 4.3** Suppose \( \gamma \) is a bill of material and \( \mathcal{A} \) is defined as above, then \( \mathcal{A}(\gamma) = \theta \), i.e., \( \mathcal{A}\gamma = \theta \) is a necessary condition for a structure \( \gamma \) to be a bill of material.

**Proof**

Suppose \( \mathcal{A}(\gamma) \neq \theta \), then \( \mathcal{A}(\gamma) = \mu \) where \( \mu = \beta^n(\gamma) \) for some \( n \in \mathbb{N} \), and where \( \beta^k(\gamma) = \mu \) \( \forall k \geq n \). By Theorem 4.2 \( \beta^n(\gamma) = \mu \) is a bill of materials. Thus if \( \mu \neq \theta \) then by Lemma 4.1 it follows that \( \exists y \in \text{dom } \mu \) such that \( \mu(y) = \emptyset \). Thus \( \beta(\mu) \neq \mu \) by the definition of \( \beta \); this is impossible since \( \mu \) is a fixed point of \( \mathcal{A} \), and thus \( \mu = \theta \) and \( \mathcal{A}(\gamma) = \theta \).
Theorem 4.4 Suppose \( \gamma : X \mapsto \mathcal{P}X \) and \( A(\gamma) = \emptyset \). Then \( (\gamma) \) is a bill of materials; i.e., \( A(\gamma) = \emptyset \) is a sufficient condition for a structure to be a bill of material.

Proof

This result is given by the contrapositive; i.e., \( \neg \gamma \) a bill of material \( \Rightarrow A(\gamma) \neq \emptyset \).

Suppose \( \gamma \) is not a bill of materials, then either part\((a)\) or part\((b)\) of Björner’s invariant is violated; suppose part\((a)\) is violated.

Then \( \exists S \in \text{rng} \gamma \) such that \( \neg S \subseteq \text{dom} \gamma \), thus \( \exists x \in \text{rng} \gamma \) such that \( \neg x \subseteq \text{dom} \gamma \).

Now \( x \in \text{rng} \gamma \wedge x \notin \text{dom} \gamma \Rightarrow x \in \text{rng} \beta(\gamma) \), which is immediate from the definition of \( \beta \).

Hence, inductively we have \( x \in \text{rng} \gamma \wedge x \notin \text{dom} \gamma \Rightarrow x \in \text{rng} \beta^n(\gamma) \). As \( A(\gamma) = \beta^n(\gamma) \) for some \( n \), it is clear that \( x \in \text{rng} A(\gamma) \), and thus \( A(\gamma) \neq \emptyset \) as required.

Suppose part\((b)\) of the invariant is violated, then \( \exists x \in \text{dom} \gamma \) such that \( x \in \text{Parts}(x,\gamma) \).

Clearly \( \gamma(x) \neq \emptyset \) and \( x \in \gamma(y) \) say, thus \( \gamma(y) \neq \emptyset \). Similarly, we get \( y \in \gamma(z) \) say, where \( \gamma(z) \neq \emptyset \); in this way a sequence \( y, z, t, \ldots, r \) is constructed such that:

\[
\begin{align*}
  x & \in \gamma(y) \wedge \gamma(y) \neq \emptyset \\
  y & \in \gamma(z) \wedge \gamma(z) \neq \emptyset \\
  z & \in \gamma(q) \wedge \gamma(q) \neq \emptyset \\
  & \vdots \\
  r & \in \gamma(x) \wedge \gamma(x) \neq \emptyset
\end{align*}
\]

Thus from the sequence above and the definition of \( \beta \), we get \( x \in \text{Parts}(x,\beta(\gamma)) \) Inductively we get \( x \in \text{Parts}(x,\beta^n(\gamma)) \). Thus \( x \in \text{Parts}(x,A(\gamma)) \) which gives us that \( A(\gamma) \neq \emptyset \), as required.

\( \gamma \) is a bill of materials if and only if \( A(\gamma) = \emptyset \)

Question 4.1 Suppose \( \gamma_1,\gamma_2 : X \mapsto \mathcal{P}X \) where \( \gamma_1 \ll \gamma_2 \) where \( \gamma_1,\gamma_2 \) may not be bill of materials is \( A \) is monotonic on these general maps, i.e., is \( A(\gamma_1) \ll A(\gamma_2) \)?

The hypothesis is false and a counter example is easily found:

Example

\[\gamma_1 = \begin{bmatrix}
  x_1 & \mapsto & \{x_2, x_3, x_4\} \\
  x_3 & \mapsto & \{x_2, x_4\} \\
  x_4 & \mapsto & \emptyset
\end{bmatrix} \quad \gamma_2 = \begin{bmatrix}
  x_1 & \mapsto & \{x_2, x_3, x_4\} \\
  x_3 & \mapsto & \{x_2, x_4\} \\
  x_2 & \mapsto & \emptyset \\
  x_4 & \mapsto & \emptyset
\end{bmatrix}\]
Then $\gamma_1 \ll \gamma_2$; however, $\mathcal{A}(\gamma_1) \gg \mathcal{A}(\gamma_2)$ as:

$$\mathcal{A}\gamma_1 = \begin{bmatrix}
    x_1 & \mapsto & \{x_2, x_3\} \\
    x_3 & \mapsto & \{x_2\} \\
    \theta & 
\end{bmatrix}$$

$$\mathcal{A}\gamma_2 = \begin{bmatrix}
    \theta
\end{bmatrix}$$

4.3 Operations on Bill of Materials

There are two key objectives here; firstly, the proof in [46] that the $\text{Ent}$ operation preserves the invariant is quite complex; an alternate more readable proof is presented. Secondly, many operations are sketched or left as exercises in [46], the actual details are presented here. The outstanding pre-conditions are stated and proofs of invariant preservation presented. In fact, the $\text{Add}$ operation is significantly more involved than [46] suggests; this is since the operation must ensure that no cycles are introduced. This section considers the abstract model of the bill of materials; i.e., $\mu \in (X \mapsto \mathcal{P}X)$; the more concrete form $X \mapsto (X \mapsto N_1)$ is considered later. The retrieve function $\mathcal{R}_{10}$ is given by $\mathcal{R}_{10} = (L \mapsto \text{dom})$.

The create operation $\text{Create}$ is given by $\text{Create} \triangleq \theta$. The first operation that is of interest is the $\text{Ent}_0$ operation which forms a new assembly by introducing a new part name $x$ such that the parts of $x$ are already part of the bill of material structure; thus this operation is used to build a bill of material from the ground up.

$\text{Ent}_0 : X \times \mathcal{P}X \mapsto (X \mapsto \mathcal{P}X) \mapsto (X \mapsto \mathcal{P}X)$

$\text{Ent}_0[x, S] \gamma_0 \triangleq \gamma_0 \cup [x \mapsto S]$

The precondition for the operation specifies that the part name $x$ is a new assembly and that the set $S$ contains the names of existing sub-assemblies, thus the precondition has the form:

$\text{pre}_{\text{Ent}_0} : X \times \mathcal{P}X \mapsto (X \mapsto \mathcal{P}X) \mapsto \mathcal{B}$

$\text{pre}_{\text{Ent}_0}[x, S] \gamma_0 \triangleq x \notin \gamma_0 \land S \subset \text{dom} \gamma_0$

The $\text{Ent}_0$ operation has the following proof obligation:

$\text{pre}_{\text{Ent}_0}[x, S] \gamma_0 \land \text{Ent}_0[x, S] \gamma_0 \Rightarrow \mathcal{A}(\text{Ent}_0[x, S] \gamma_0) = \theta$

The proof of this result in [46] is excessively complex; the aim here is to present a simpler and more readable proof of invariant preservation.

86
Theorem 4.5 \( \mathcal{A}(\gamma) = \beta^n(\gamma) = \theta. \)

Proof

Suppose \( \mathcal{A}(\gamma) = \beta^n(\gamma) = \theta. \) It is required to demonstrate that \( \mathcal{A}(\gamma) = \mathcal{A}(\gamma \cup [x \mapsto S]) = \theta. \)

\[
\beta(\gamma \cup [x \mapsto S]) = \beta(\gamma) = \beta^i(\gamma) \cup \beta([x \mapsto \gamma^{-1}_0]S).
\]

Thus \( \beta(\gamma \cup [x \mapsto S]) = \beta(\gamma) \cup \beta([x \mapsto \gamma^{-1}_0]S). \) This leads to an expression for \( \beta^n(\gamma \cup [x \mapsto S]) \) via an inductive argument.

\[
\beta^n(\gamma \cup [x \mapsto S]) = \beta^n(\gamma) \cup \beta([x \mapsto \gamma_0^0(\gamma_0^0)^{-1} \cup \beta_1(\gamma_0^1)^{-1} \cup ... \cup \beta^{-1}(\gamma_0^{-1})^{-1} S])
\]

The basis case for the inductive argument is \( n = 1, \) which has been proved above; the inductive step assumes the result is true for \( n = k, \) i.e., we assume that \( \beta^k(\gamma \cup [x \mapsto S]) = \beta^k(\gamma) \cup \beta([x \mapsto \gamma_0^0(\gamma_0^0)^{-1} \cup \beta_1(\gamma_0^1)^{-1} \cup ... \cup \beta^{-1}(\gamma_0^{-1})^{-1} S]). \)

Then the proof for \( n = k + 1 \) is derived by applying \( \beta \) to \( \beta^k(\gamma \cup [x \mapsto S]), \) and we get:

\[
\beta^{k+1}(\gamma \cup [x \mapsto S]) = \beta(\beta^k(\gamma \cup [x \mapsto S])) = \beta(\beta^k(\gamma) \cup \beta([x \mapsto \gamma_0^0(\gamma_0^0)^{-1} \cup \beta_1(\gamma_0^1)^{-1} \cup ... \cup \beta^{-1}(\gamma_0^{-1})^{-1} S])).
\]

Thus \( \beta^n(\gamma \cup [x \mapsto S]) = \beta^n(\gamma) \cup \beta([x \mapsto \gamma_0^0(\gamma_0^0)^{-1} \cup \beta_1(\gamma_0^1)^{-1} \cup ... \cup \beta^{-1}(\gamma_0^{-1})^{-1} S]) \) as required.

\( \mathcal{A}(\gamma) = \beta^n(\gamma) = \theta, \) thus the basic parts and sub-assemblies in the bill of materials \( \gamma \) is given by:

\[
\text{dom} \gamma = \beta_0^0(\gamma_0^0)^{-1} \cup \beta_1(\gamma_0^1)^{-1} \cup ... \cup \beta^{-1}(\gamma_0^{-1})^{-1}
\]

87
This follows since the bill of material $\gamma$ is annihilated by successive applications of the shrinking operator $\beta$; thus each element $y \in \gamma$ is in exactly one of $\{\beta^0(\gamma)_{\emptyset}, \beta^1(\gamma)_{\emptyset}, \ldots, \beta^{n-1}(\gamma)_{\emptyset}\}$

The precondition guarantees that $S \subseteq \text{dom} \gamma$:

$$
\beta([x \mapsto \emptyset]) = \beta([x \mapsto \text{dom} \gamma]) = \beta([x \mapsto \emptyset]) = \theta
$$

$$
\mathcal{A}(\gamma \cup [x \mapsto S]) = \beta^n(\gamma \cup [x \mapsto S]) = \beta^n(\gamma) \cup \beta([x \mapsto \emptyset]) = \theta \cup \theta = \theta
$$

thus the proof that $\text{Ent}_0[x, S] \gamma$ preserves the invariant is complete; however, the proof is quite involved.

$$
\text{pre}_\text{Ent}_0[x, S]^{\gamma_0} \wedge \text{Ent}_0[x, S]^{\gamma_0} \Rightarrow \mathcal{A}(\text{Ent}_0[x, S]^{\gamma_0}) = \theta
$$

The **Remove operation**

This operation is defined in [46], however, no proof of invariant preservation is provided; for completeness, the proof of invariant preservation is presented here. The $\text{Rem}_0$ operation is defined as follows:

$$
\text{Rem}_0 : X \mapsto (X \mapsto \mathcal{P}X) \mapsto (X \mapsto \mathcal{P}X)
$$

$$
\text{Rem}_0[x]^{\gamma_0} \triangleq \mathcal{P}[x]^{\gamma_0}
$$

$$
\text{pre}_\text{Rem}_0 : X \mapsto (X \mapsto \mathcal{P}X) \mapsto \text{B}
$$

$$
\text{pre}_\text{Rem}_0[x]^{\gamma_0} \triangleq x \in \text{dom} \gamma_0 \land x \notin \text{rng} \gamma_0
$$

The proof obligation for the $\text{Rem}_0$ operation is the following:

$$
\text{pre}_\text{Rem}_0[x]^{\gamma_0} \land \text{Rem}_0[x]^{\gamma_0} \Rightarrow \mathcal{A}(\text{Rem}_0[x]^{\gamma_0}) = \theta
$$

This result is proved by contradiction; suppose $\mathcal{A}(\text{Rem}_0[x]^{\gamma_0}) = \mu$ where $\mu \neq \theta$, then it will be shown that $\mathcal{A}(\gamma) \neq \theta$, which contradicts the fact that $\gamma$ is a bill of materials.
\[ \gamma = \mathbb{4}[x] \gamma \cup [x \mapsto \gamma(x)] \]

The precondition states that \( x \not\in \mathcal{U} / \circ \text{rng} \gamma \), thus \( x \not\in \mathcal{U} / \circ \text{rng} \circ \mathbb{4}[x] \gamma \). Theorems 4.3 and 4.4 enable the Björner invariant to be interchanged with the annihilator formulation of the invariant; thus either part \( a \) or part \( b \) of the Björner invariant fails. Suppose part \( a \) fails, then we have:

\[ \exists S \subseteq \mathcal{U} / \circ \text{rng} \circ (\mathbb{4}[x] \gamma) \) and \( \neg(S \subseteq \text{dom} \circ \mathbb{4}[x] \gamma) \). Then \( \exists y \in \mathcal{U} / \circ \text{rng} \circ \mathbb{4}[x] \gamma \) such that \( y \not\in \text{dom} \circ \mathbb{4}[x] \gamma \). However, this implies that \( y \in \text{rng} \gamma \), and furthermore \( y \not\in \text{dom} \gamma \), since \( \text{dom} \gamma = \text{dom} \circ \mathbb{4}[x] \gamma \cup \{x\} \) and \( y \neq x \). Thus we have a contradiction as \( \gamma \) is a bill of materials, thus part \( a \) of Björner’s invariant must hold.

Suppose part \( b \) of the Björner invariant fails; then \( \exists y \in \mathbb{4}[x] \gamma \) such that \( y \in \text{Parts}(y, \mathbb{4}[x] \gamma) \). Clearly \( z \in \text{Parts}(y, \mathbb{4}[x] \gamma) \Rightarrow z \in \text{Parts}(y, \gamma) \), thus we have \( y \in \text{Parts}(y, \gamma) \) which is a contradiction, since \( \gamma \) is a bill of material. Thus part \( b \) of invariant holds and the desired result follows.

\[ \text{pre} \text{Rem}_0[x] \gamma_0 \land \text{Rem}_0[x] \gamma_0 \Rightarrow \mathcal{A}(\text{Rem}_0[x] \gamma_0) = \theta \]

**The Add operation**

The precondition for the Add operation is significantly more involved than [46] suggests, and its definition is sketched in [46]. The fundamental problem with amending the definition of sub-assembly \( x \) to include another sub-assembly \( y \) is that such an amendment may lead to the introduction of cycles, thereby invalidating the bill of materials invariant. The solution to this problem is to place stringent constraints on the precondition to ensure that cycles will not be introduced by the operation. A crude approach of achieving this is to include invariant satisfaction as part of the precondition, however, this approach is rejected as the precise constraints guaranteed to preserve the invariant are sought.

In order to gain some insight into the problem, an example is considered which has the effect of introducing cycles. This enables a deduction of the appropriate formulation of the precondition to ensure that the Add operation preserves the invariant.

\[
\gamma_0 = \begin{bmatrix}
x_1 & \{x_3, x_4\} \\
x_2 & \emptyset \\
x_3 & \emptyset \\
x_4 & \{x_2\} \\
x_5 & \emptyset \\
x_6 & \{x_1\}
\end{bmatrix}
\]

\[
\gamma_0 = \begin{bmatrix}
x_1 & \{x_3, x_4\} \\
x_2 & \emptyset \\
x_3 & \emptyset \\
x_4 & \{x_2, x_6\} \\
x_5 & \emptyset \\
x_6 & \{x_1\}
\end{bmatrix}
\]
where \( \gamma_0 \) is obtained from \( \text{Add}_0[x_4, x_6] \gamma_0 \) operation, the effect of the operation being to change \( [x_4 \mapsto \{x_2\}] \) to \( [x_4 \mapsto \{x_2, x_6\}] \), giving a structure which is not a bill of materials since it contains cycles. The problem with the \( \text{Add}_0[x_4, x_6] \) is that part \( x_6 \) is a sub-assembly of part \( x_4 \); however, \( x_1 \) is a sub-assembly of \( x_6 \), and \( x_4 \) is a sub-assembly of \( x_1 \), thus transitively \( x_4 \) is a sub-assembly of \( x_6 \). Thus it must be ensured that whenever \( x_6 \) is to be added as a sub-assembly to \( x_4 \) that \( x_4 \) is not a part of \( x_6 \) or equivalently \( x_4 \) is not in the transitive closure of \( x_6 \). This is certainly a necessary condition for the \( \text{Add}_0 \) operation to preserve the invariant, in fact it is a sufficient condition.

\[
\text{Add}_0 : X \times X \mapsto (X \mapsto \mathcal{P} X) \mapsto (X \mapsto \mathcal{P} X) \\
\text{Add}_0[x, x'] \gamma_0 \triangleq \gamma_0 \cup [x \mapsto \{x'\}]
\]

\[
\text{pre}_{\text{Add}_0} : X \times X \mapsto (X \mapsto \mathcal{P} X) \mapsto \mathbb{B} \\
\text{pre}_{\text{Add}_0}[x, x'] \gamma_0 \triangleq x \in \gamma_0 \land x' \in \gamma_0 \land x' \notin \gamma_0(x) \land \gamma_0(x) \neq \emptyset \land \neg x' R^+ x.
\]

The precondition ensures that \( x \) and \( x' \) are already in the bill of material. It prohibits \( x' \) from already being a sub-part of \( x \), and \( x \) being promoted from a basic part to a sub-assembly. Finally, it states that \( x' \), the part being added to the sub-assembly \( x \) must not contain \( x \) as a part. The next theorem proves that the constraints in the precondition are sufficient to preserve the invariant.

**Theorem 4.6** \( \text{pre}_{\text{Add}_0}[x, x'] \gamma_0 \land \text{Add}_0[x, x'] \gamma_0 \Rightarrow \mathcal{A}(\text{Add}_0[x, x'] \gamma_0) = \emptyset \)

**Proof**

Suppose not, then either part (a) or part (b) of Björner’s invariant fails. Suppose part (a) fails, then \( \exists S \subseteq ^1 \circ \text{rng} \gamma_0 \) and \( \neg (S \subseteq \text{dom} \gamma_0) \), where \( \gamma_0 \) denotes \( \gamma_0 \cup [x \mapsto \{x'\}] \). However, it is clear from the construction of \( \gamma_0 \) that we have:

\[
\text{dom} \gamma_0 = \text{dom} \gamma_0
\]

\[
^1 \circ \text{rng} \gamma_0 = ^1 \circ \text{rng} \gamma_0
\]

Thus \( \exists S \subseteq \text{rng} \gamma_0 \) such that \( \neg (S \subseteq \text{dom} \gamma_0) \), which is a contradiction since \( \gamma_0 \) is a bill of materials.

Suppose part (b) of the invariant fails. Then \( \exists y \in \gamma_0 \) such that \( y \in \text{Parts}(y, \gamma_0) \). Thus \( \exists z \) such that \( y \in \gamma_0(z) \) and \( y R^+_x z \). Since \( x \) is the only part to change definition the sequence from \( y \) to \( z \) to \( y \) must involve \( x \), and thus \( x' \), otherwise the original invariant is violated. Thus we have:

\[
90
\]
\[ yR^+x \quad \text{and} \quad xR^+y \]
\[ yR^+x \quad \text{and} \quad xR^1x' \quad \text{and} \quad x'R^+y \]

\( x'R^+y \) and \( yR^+x \) \( \Rightarrow \) \( x'R^+x \). However, \( x'R^+x \) is a contradiction of the precondition for the \( Add_0 \) operation, thus the second part of the invariant must hold and the proof is complete.

The Erase operation

The \( Eras_0 \) is the inverse operation of the \( Add_0 \) operation and removes a part or sub-assembly \( x' \) of \( x \), provided \( x \) and \( x' \) are recorded as part of the bill of materials, and the removal of \( x' \) from \( x \) does not cause \( x \) to become a basic part.

\[
\begin{align*}
Eras_0 : X \times X & \mapsto (X \mapsto P X) \mapsto (X \mapsto P X) \\
\text{pre}_Eras_0 : X \times X & \mapsto (X \mapsto P X) \mapsto B \\
\text{pre}_Eras_0[x, x'] & \gamma_0 \triangleq x \in \gamma_0 \land x' \in \gamma_0(x) \land \exists\, \gamma_0(x) \neq \emptyset \\
\text{Theorem 4.7} & \quad \text{pre}_Eras_0[x, x'] \gamma_0 \land Eras_0[x, x'] \gamma_0 \Rightarrow A(Eras_0[x, x'] \gamma_0) = \theta.
\end{align*}
\]

Proof

Clearly part \((a)\) of Björner’s invariant is satisfied. Suppose part \((b)\) fails, then \( \exists y \) such that \( y \in R^+_0 y \). Then as the definition of \( x \) is all that has changed in the bill of materials, it follows that \( y \in R^+_0 x \) and \( x \in R^+_0 y \). However, we have:

\[
\begin{align*}
y \in R^+_0 x & \Rightarrow yR^+ x \quad (4.4) \\
x \in R^+_0 y & \Rightarrow xR^+ y \quad (4.5)
\end{align*}
\]

Thus it follows that \( yR^+y \) and the original invariant is violated, which is a contradiction, since \( \gamma_0 \) is a bill of materials. Thus the \( Eras_0 \) operation preserves the invariant.

4.4 Model 1 of Bill of Materials

The natural starting point for modelling bills of materials is given by:

\[
\gamma_1 \in BOM_1 = X \mapsto (X \mapsto N_1)
\]

the abstract model \( X \mapsto \mathcal{P}X \) has the advantage of being easier to work with, however, the abstract model is given by the retrieval function \( R_{10} = (I \mapsto \text{dom}) \), and thus \( \gamma_0 = R_{10}(\gamma_1) \).

Furthermore, the invariant for \( \gamma_1 \) is given by:

91
\[ \mathcal{A}(\{I \mapsto \text{dom}\} \gamma_1) = \theta \]  

(4.6)

The retrieval function is not one to one [46], page 214, as two distinct bills of materials may map to the same abstract bill of materials, \( \gamma_0 \). The numerical information for each basic part or sub-assembly associated with a bill of materials is effectively the well known bag structure.

Bill of material operations for the abstract model have been studied in Section 4.3; in order to demonstrate that the corresponding concrete operations, are faithful to the abstract operations, there are proof obligations to demonstrate that each operation is a valid refinement. This involves proving that the well known commuting diagram property holds; for example, in the case of the \( \text{Ent}_1 \) operation, the proof obligation is as follows:

\[ \text{Ent}_0[x, S] \circ \mathcal{R}_{10}(\gamma_1) = \mathcal{R}_{10} \circ \text{Ent}_1[x, \beta] \gamma_1 \]  

(4.7)

where \( S = \text{dom} \beta \). The \( \text{Ent}_1 \) operation and precondition are defined as follows:

\[
\begin{align*}
\text{Ent}_1 : X \times (X \mapsto N_1) \mapsto (X \mapsto (X \mapsto N_1)) \mapsto (X \mapsto (X \mapsto N_1)) \\
\text{Ent}_1[x, \beta] \gamma_1 \equiv \gamma_1 \cup [x \mapsto \beta] \\
\text{pre}_{\text{Ent}_1} : X \times (X \mapsto N_1) \mapsto (X \mapsto (X \mapsto N_1)) \mapsto \text{B} \\
\text{pre}_{\text{Ent}_1}[x, \beta] \gamma_1 \equiv \text{pre}_{\text{Ent}_0}[x, \text{dom} \beta](I \mapsto \text{dom}) \gamma_1
\end{align*}
\]

The proof obligation is to demonstrate that Equation 4.7 holds.

\[
\begin{align*}
\text{Ent}_1[x, \beta] & = \gamma_1 \cup [x \mapsto \beta] \\
\mathcal{R}_{10} \circ \text{Ent}_1[x, \beta] & = (I \mapsto \text{dom}) (\gamma_1 \cup [x \mapsto \beta]) \\
& = (I \mapsto \text{dom}) \gamma_1 \cup (I \mapsto \text{dom}) [x \mapsto \beta] \\
& = \gamma_0 \cup [x \mapsto \text{dom} \beta] \\
& = \gamma_0 \cup [x \mapsto S] \\
\mathcal{R}_{10}(\gamma_1) & = (I \mapsto \text{dom}) \gamma_1 = \gamma_0 \\
\text{Ent}_0[x, S] \circ \mathcal{R}_{10}(\gamma_1) & = \text{Ent}_0[x, S] \gamma_0 \\
& = \gamma_0 \cup [x \mapsto S]
\end{align*}
\]

Thus the proof obligation for the \( \text{Ent}_1 \) operation is complete; in fact, every concrete operation \( \text{op}_1 \) has a corresponding proof obligation; it is required to demonstrate the commuting diagram property:

\[ \text{op}_0[\_] \circ \mathcal{R}_{10}(\gamma_1) = \mathcal{R}_{10} \circ \text{op}_1[\_] \]  

(4.8)

92
4.5 Fallacious Proofs

MacAnAirchinnigh [46], page 215 considers the problem of determining the parts explosion of a newly added assembly to a bill of material. The formula and its proof presented in [46] seem excessively complicated, an attempt is made to simplify the proof here. The statement of the theorem in [46] is correct, however, the definition of the \( \beta_i \) used in the statement of the theorem is incorrect.

This demonstrates the importance of re-examining proofs and specifications; such a re-examination providing further justification to the correctness of the theorem, or actually determining flaws in a proof. Lakatos [41] argues that one may never be absolutely certain of the correctness of a proof, and argues against the absolute truth of mathematical proofs. However, re-doing a proof presents further evidence of correctness of the proof.

In this section the original statement of the theorem in [46] is corrected, and an alternate proof presented.

**Theorem 4.8** \( Prts[x] \circ \text{Ent}_1[x, \beta] \gamma = \Sigma_{i=1}^k m_i \otimes \beta_i \)

The symbols used in the equation represent the following: \( Prts[x] \) represents the parts explosion of the newly added assembly \( x \), i.e., it yields the set of basic parts that \( x \) is composed of; \( \beta = [x_1 \mapsto m_1, x_2 \mapsto m_2, \ldots, x_k \mapsto m_k] \) represents the definition of the assembly \( x \); \( \Sigma \) represents bag summation; \( \otimes \) represents an endomorphism on a bag, i.e., it represents the calculation of a multiple of the bag. \( \beta_i \) is defined as follows in [46].

\[
\beta_i = \begin{cases} 
\beta_{x_i} & \text{if } i \text{ is a basic part not in another assembly.} \\
\beta_{x_i} \otimes Prts[x_j] \gamma & \text{if } i \text{ is a basic part in another assembly } x_j, i \neq j \\
Prts[x_i] \gamma & \text{otherwise}
\end{cases}
\]

where \( \beta_{x_i} \) denotes the singleton bag \( [x_i \mapsto 1] \).

**Comment 4.1** The definition of \( \beta_i \) in [46] is incorrect, in particular, it is unnecessary to consider whether a basic part is in another assembly. The following example demonstrates the formulation is actually incorrect.

**Example**

Consider the addition of a new assembly \( x_4 \) to the following bill of materials \( \gamma \), where \( \beta = [x_1 \mapsto 2, x_3 \mapsto 3, x_5 \mapsto 2] \), and \( \gamma, \bar{\gamma} \) are defined as follows:

\[
\gamma = \begin{bmatrix} 
  x_1 & \mapsto & [x_2 \mapsto 3, x_3 \mapsto 4, x_5 \mapsto 6] \\
  x_2 \mapsto \theta \\
  x_3 \mapsto \theta \\
  x_5 \mapsto \theta 
\end{bmatrix} \quad \bar{\gamma} = \begin{bmatrix} 
  x_1 & \mapsto & [x_2 \mapsto 3, x_3 \mapsto 4, x_5 \mapsto 6] \\
  x_2 \mapsto \theta \\
  x_3 \mapsto \theta \\
  x_4 \mapsto [x_1 \mapsto 2, x_3 \mapsto 3, x_5 \mapsto 2] \\
  x_5 \mapsto \theta 
\end{bmatrix}
\]

93
Then \( \text{Prts}[x_4] \circ \text{Ent}_1[x_4, \beta] \gamma = \text{Prts}[x_4] \gamma \). This calculation yields \( [x_2 \mapsto 6, x_3 \mapsto 8, x_5 \mapsto 14] \)

When calculating \( \sum_{i=1}^{k} m_i \otimes \beta_i \) according to the definition in [46] we observe that basic parts \( x_5 \) and \( x_2 \) occur as basic parts of sub-assembly \( x_1 \) thus applying the formula as presented we get:

\[
2 \otimes \text{Prts}[x_1] \gamma + 3 \otimes ([x_3 \mapsto 1] \oplus \text{Prts}[x_1] \gamma) + 2 \otimes ([x_5 \mapsto 1] \oplus \text{Prts}[x_1] \gamma)
\]

It is immediately obvious that this formula cannot be correct as \( \text{Prts}[x_1] \) is calculated and included three times in the formula. Thus the computed sum exceeds the correct value; in fact, it yields \( [x_2 \mapsto 21, x_3 \mapsto 31, x_5 \mapsto 44] \) which is incorrect.

However, the theorem is correct if the definition of the \( \beta_i \) is amended such that whenever a basic part occurs no cross checking is made with other sub-assemblies in which the part appears. The correct definition of the \( \beta_i \) is stated as follows:

\[
\beta_i = \begin{cases} 
\beta_{x_i} & \text{if } x_i \text{ a basic part} \\
\text{Prt}[x_i] \gamma & \text{otherwise}
\end{cases}
\]

**Theorem 4.9** \( \text{Prt}[x] \circ \text{Ent}_1[x, \beta] \gamma = \sum_{i=1}^{k} m_i \otimes \beta_i \) where \( \beta_i \) is the amended definition.

**Proof**

It is clear from the parts explosion algorithm [46], page 214, that \( \text{Prt}[y] \gamma = \text{Prt}[y] \gamma \) where \( x \neq y \). This follows since the definition of \( \gamma \) agrees with \( \gamma \) except for \( x \), and \( x \) does not appear as a part of any assembly. Furthermore, the algorithm extracts the basic parts from the corresponding bag of all parts of \( x \).

In fact, \( \text{Prt}[x] \gamma \) may be determined by a restriction to the basic parts of the following expression:

\[
\text{Prt}[x] \gamma = \Delta[\gamma^{-1}(\theta)](m_1 \otimes \beta_{x_1}) \oplus m_1 \otimes \text{Prt}[x_1] \gamma \\
\oplus (m_2 \otimes \beta_{x_2}) \oplus m_2 \otimes \text{Prt}[x_2] \gamma \\
\vdots \\
\oplus (m_k \otimes \beta_{x_k}) \oplus m_k \otimes \text{Prt}[x_k] \gamma)
\]

From the above formulation the theorem follows immediately, as if \( x_i \) is a basic part then \( \text{Prt}[x_i] \gamma = \theta \) thus \( m_i \otimes \text{Prt}[x_i] \gamma = \theta \). Thus for a basic part \( x_i \) the expression \((m_i \otimes \beta_{x_i}) \oplus m_i \otimes \text{Prt}[x_i] \gamma \) yields \( m_i \otimes \beta_{x_i} \) as desired.
Similarly if \( x_i \) is not a basic part then \( (m_i \otimes \beta_{x_i}) \) is omitted when the restriction to basic parts takes place, thus the expression \( (m_i \otimes \beta_{x_i}) \oplus m_i \otimes Prts[x_i] \gamma \) yields \( m_i \otimes Prts[x_i] \gamma = m_i \otimes Prts[x_i] \gamma \) as desired.

The theorem is proved if the expression above can be justified. This is achieved by studying the parts explosion algorithm in [46].

\[
Prts[x] \gamma \\
= Prts_1[\beta]\gamma, \theta \\
= Prts_1[x_1 \mapsto m_1] \sqcup \beta'(\gamma, \theta) \\
= Prts_1[\beta'] \circ Prts_{m_1} [\gamma(x_1)](\gamma, [x_1 \mapsto m_1]) \\
= Prts_1[\beta'] \circ Prts_{m_1} [\gamma(x_1)](\gamma, \theta) \oplus ([x_1 \mapsto m_1]) \\
= Prts_1[\beta'] \circ Prts_{m_1} [\gamma(x_1)](\gamma, \theta) \oplus (m_1 \otimes \beta_{x_1}) \\
= Prts_1[\beta'] \circ (m_1 \otimes Prts_1[x_1](\gamma, \theta)) \oplus (m_1 \otimes \beta_{x_1}) \\
= Prts_1[\beta'](\gamma, \theta) \oplus (m_1 \otimes Prts_1[x_1] \gamma) \oplus (m_1 \otimes \beta_{x_1})
\]

Similarly, we have \( \beta' = [x_2 \mapsto m_2] \sqcup \beta' \) and we get \( Prts_1[\beta'](\gamma, \theta) = Prts_1[\beta'](\gamma, \theta) \oplus (m_2 \otimes Prts_1[x_2] \gamma) \oplus (m_2 \otimes \beta_{x_2}) \). Clearly, by proceeding in this manner the result and theorem follows.

4.6 Joining Bills of Materials

The Ent_0 operation is used to build bills of materials from the ground up; an obvious generalization is to consider an operation to join two existing bills of materials. It is immediate from considering elementary examples that a problem of consistency may arise when joining two existing bills of materials. The problem is that given two bills of materials, there is the possibility that they may have overlapping assembly definitions, such definitions may be mutually inconsistent. It is thus important to ensure that bills of materials are not joined naively; a stringent precondition must be defined which guarantees that the joined structure is a bill of material. Two bills of materials are termed consistent if they contain no incompatible definitions of parts or assemblies.

Example

\[
\gamma_1 = \begin{bmatrix}
  x_1 & \mapsto & \{x_2\} \\
  x_2 & \mapsto & \emptyset
\end{bmatrix} \quad \gamma_2 = \begin{bmatrix}
  x_1 & \mapsto & \{x_3\} \\
  x_3 & \mapsto & \emptyset
\end{bmatrix}
\]

Thus it is immediately seen that it is not meaningful to join \( \gamma_1 \) and \( \gamma_2 \), as these contain mutually incompatible definitions of part \( x_1 \). Thus an operation to join \( \gamma_1 \) and \( \gamma_2 \) must
examine overlapping parts and assemblies, in order to ensure that these definitions are consistent. The precondition must prohibit a join operation taking place when incompatibilities exist between $\gamma_1$ and $\gamma_2$.

$$\text{pre}_0 \Join: (X \mapsto \mathcal{P}X) \times (X \mapsto \mathcal{P}X) \mapsto \mathcal{B}$$

$$\text{pre}_0(\gamma_1, \gamma_2) \triangleq$$

$$\lbrack \gamma_1 \cap \gamma_2 \rbrack \gamma_1 = \lbrack \gamma_1 \cap \gamma_2 \rbrack \gamma_2$$

$$\land \mathcal{A}(\gamma_1) = \emptyset$$

$$\land \mathcal{A}(\gamma_2) = \emptyset$$

**Note 1:** $\gamma_1 \cap \gamma_2$ denotes the map elements $\gamma_1$ and $\gamma_2$ have in common. This may be reformulated as: $\forall x \in \text{dom} \; \gamma_1 \cap \text{dom} \; \gamma_2 \; \gamma_1(x) = \gamma_2(x)$

**Note 2:** It is clear that the precondition as defined above is a necessary condition for the joined structure to be a bill of material; in fact, the condition is sufficient. The operation to join two bills of material is then given simply by the override operator of the $\text{VDM}^*$ operator calculus.

$$\Join_0: (X \mapsto \mathcal{P}X \times X \mapsto \mathcal{P}X) \mapsto X \mapsto \mathcal{P}X$$

$$\Join_0(\gamma_1, \gamma_2) \triangleq \gamma_1 \uparrow \gamma_2$$

**Lemma 4.2** Given consistent $\gamma_1$ and $\gamma_2$, the structure $\gamma_1 \uparrow \gamma_2$ maintains the consistency of $\gamma_1$ and $\gamma_2$, and furthermore, $\lbrack \gamma_1 \rbrack(\gamma_1 \uparrow \gamma_2) = \gamma_1$ and $\lbrack \gamma_2 \rbrack(\gamma_1 \uparrow \gamma_2) = \gamma_2$.

**Proof**

The precondition for the $\Join_0$ operation ensures the following properties hold for $\gamma_1$ and $\gamma_2$:

$$\gamma_1 = \lbrack \gamma_2 \rbrack \gamma_1 \cup \lbrack \gamma_1 \rbrack \gamma_2$$

$$\gamma_2 = \lbrack \gamma_1 \rbrack \gamma_2 \cup \lbrack \gamma_2 \rbrack \gamma_1$$

$$\gamma_1 \uparrow \gamma_2 = \lbrack \gamma_2 \rbrack \gamma_1 \cup \gamma_2$$

$$= \lbrack \gamma_2 \rbrack \gamma_1 \cup \lbrack \gamma_1 \rbrack \gamma_2 \cup \lbrack \gamma_2 \rbrack \gamma_1$$

This is exactly analogous to $A \cup B = (A - B) \cup (A \cap B) \cup (B - A)$, i.e., $A \cup B$ is composed of the union of three disjoint sets, we note the precondition gives us:

$$\lbrack \gamma_2 \rbrack \gamma_1 = \lbrack \gamma_1 \rbrack \gamma_2$$

In fact, for consistent bills of materials $\gamma_1 \uparrow \gamma_2 = \gamma_2 \uparrow \gamma_1$, i.e., for consistent bills of materials the $\uparrow$ operation is commutative:

96
\[
\begin{align*}
\gamma_1 \uparrow \gamma_2 &= 4[\gamma_2] \gamma_1 \sqcup 4[\gamma_1] \gamma_2 \sqcup \llbracket \gamma_1 \rrbracket \gamma_2 \\
&= 4[\gamma_1] \gamma_2 \sqcup 4[\gamma_2] \gamma_1 \sqcup \llbracket \gamma_1 \rrbracket \gamma_2 \\
\gamma_1 \uparrow \gamma_2 &= 4[\gamma_2] \gamma_1 \sqcup 4[\gamma_2] \gamma_1 \sqcup \llbracket \gamma_2 \rrbracket \gamma_1 \\
&= 4[\gamma_1] \gamma_2 \sqcup \gamma_1 \\
&= \gamma_2 \uparrow \gamma_1
\end{align*}
\]

Thus \( \gamma_1 \uparrow \gamma_2 \) is commutative for consistent bills of materials.

Furthermore, \( \llbracket \gamma_1 \rrbracket (\gamma_1 \uparrow \gamma_2) = \gamma_1 \) as:

\[
\begin{align*}
\llbracket \gamma_1 \rrbracket (\gamma_1 \uparrow \gamma_2) &= \llbracket \gamma_1 \rrbracket (\downarrow \downarrow \gamma_2) \\
&= \llbracket \gamma_1 \rrbracket (\downarrow \downarrow \gamma_2) \uparrow \llbracket \gamma_1 \rrbracket \gamma_2 \sqcup \llbracket \gamma_2 \rrbracket \gamma_1 \\
\llbracket \gamma_1 \rrbracket (\gamma_1 \uparrow \gamma_2) &= \llbracket \gamma_1 \rrbracket \circ (\downarrow \downarrow \gamma_2) \gamma_1 \sqcup \llbracket \gamma_1 \rrbracket \circ (\downarrow \downarrow \gamma_2) \gamma_2 \sqcup \llbracket \gamma_2 \rrbracket \circ (\downarrow \downarrow \gamma_2) \gamma_1 \\
&= (\downarrow \downarrow \gamma_2) \gamma_1 \sqcup \theta \sqcup \downarrow \downarrow \gamma_2 \gamma_1 \\
&= \gamma_1
\end{align*}
\]

Similarly \( \llbracket \gamma_2 \rrbracket (\gamma_1 \uparrow \gamma_2) = \gamma_2 \).

**Note 1:** Thus whenever two consistent bill of material structures are joined by the \( \uparrow \) operation, then this operation is commutative, i.e., \( \gamma_1 \uparrow \gamma_2 = \gamma_2 \uparrow \gamma_1 \). Furthermore, \( \gamma_1 \) and \( \gamma_2 \) may be retrieved from the super structure via a domain restriction operation. However, the objective remains to prove that such structure is in fact a bill of material.

**Note 2:** The proof of the lemma employed the fact that the \( \llbracket \rrbracket \) operator is an endomorphism, i.e., it is distributive over \( \sqcup \).

**Theorem 4.10** Given two consistent bills of materials \( \gamma_1 \) and \( \gamma_2 \), then \( \gamma = \text{Join}_0(\gamma_1, \gamma_2) \) is a bill of material.

**Proof** (Using Björner’s invariant)

Two proofs of this result are presented; the first uses the Björner invariant formulation; the second uses the annihilator form of the invariant. This enables a comparison to be made on the effectiveness and usability of both invariants; in fact, the proof using the annihilator function approach is more involved than the corresponding proof using Björner’s invariant.

Suppose \( \text{Join}_0(\gamma_1, \gamma_2) \) is not a bill of material, and suppose part (a), i.e., the all recorded part of the invariant fails; then \( \exists x \in \text{dom} \gamma \) such that \( x \notin \text{dom} \gamma \).

\[
x \in \text{rng}(\text{Join}_0(\gamma_1, \gamma_2)) \land x \notin \text{dom}(\text{Join}_0(\gamma_1, \gamma_2))
\]

97
⇒ \( x \in \text{rng} (\gamma_1 \uparrow \gamma_2) \land x \notin \text{dom} (\gamma_1 \uparrow \gamma_2) \)
⇒ \((x \in \text{rng} \gamma_1 \lor x \in \text{rng} \gamma_2) \land (x \notin \text{dom} \gamma_1 \land x \notin \text{dom} \gamma_2) \)
⇒ \((x \in \text{rng} \gamma_1 \lor x \in \text{rng} \gamma_2) \land x \notin \text{dom} (\gamma_1 \land \gamma_2) \)
⇒ \((x \in \text{rng} \gamma_1 \land x \notin \text{dom} \gamma_1) \lor (x \in \text{rng} \gamma_2 \land x \notin \text{dom} \gamma_2) \)
⇒ \((x \in \text{rng} \gamma_1 \land x \notin \text{dom} \gamma_1 \land x \notin \text{dom} \gamma_2) \lor (x \in \text{rng} \gamma_2 \land x \notin \text{dom} \gamma_1 \land x \notin \text{dom} \gamma_2) \)
⇒ \((x \in \text{rng} \gamma_1 \land x \notin \text{dom} \gamma_1) \lor (x \in \text{rng} \gamma_2 \land x \notin \text{dom} \gamma_2) \)

However, this is a contradiction since \( \gamma_1 \) and \( \gamma_2 \) are bills of materials. Thus the all recorded part of the invariant must be satisfied.

Suppose part \((b)\) of the invariant fails to hold, then \( \exists x \in \text{dom} \gamma \) such that \( x \in \text{Parts}(x, \gamma) \).
By the consistency of \( \gamma_1 \) and \( \gamma_2 \) it is clear that,
\[
\begin{align*}
x \in \text{dom} \gamma_1 & \Rightarrow \text{Parts}(x, \gamma_1 \uparrow \gamma_2) = \text{Parts}(x, \gamma_1) \\
x \in \text{dom} \gamma_2 & \Rightarrow \text{Parts}(x, \gamma_1 \uparrow \gamma_2) = \text{Parts}(x, \gamma_2)
\end{align*}
\]

In either case we have a contradiction since \( \gamma_1 \) and \( \gamma_2 \) are bills of materials.

Thus we have proved (using Björner’s invariant) that provided two bills of material are consistent then the joined structure given by the \( \uparrow \) operation is a bill of material. Next we prove the result for the annihilator formulation. We first prove that the shrinking operator \( \beta \) is an endomorphism for consistent bills of materials.

**Lemma 4.3** Suppose \( \gamma_1 \) and \( \gamma_2 \) are consistent bill of materials, then \( \beta(\gamma_1 \uparrow \gamma_2) = \beta(\gamma_1) \uparrow \beta(\gamma_2) \) and \( \beta(\gamma_1) \) is consistent with \( \beta(\gamma_2) \).

**Proof**
Suppose \( \gamma_1, \gamma_2 \) are consistent, we first show that \( \beta(\gamma_1) \) and \( \beta(\gamma_2) \) are consistent. Let \( x \in \text{dom} \beta(\gamma_1) \cap \text{dom} \beta(\gamma_2) \), thus we must show that \( \beta(\gamma_1)(x) = \beta(\gamma_2)(x) \).

\[
\begin{align*}
\beta(\gamma_1)(x) &= \langle [\gamma_1^{-1}(\emptyset)] \gamma_1(x) \\
\beta(\gamma_2)(x) &= \langle [\gamma_2^{-1}(\emptyset)] \gamma_2(x)
\end{align*}
\]

Thus all that is required is to prove the following property:
\[
\langle [\gamma_1^{-1}(\emptyset)] \gamma_1(x) = \langle [\gamma_1^{-1}(\emptyset)] \gamma_2(x)
\]

It is clear that \( x \in \text{dom} \beta(\gamma_1) \cap \text{dom} \beta(\gamma_2) \Rightarrow x \in \text{dom} \gamma_1 \cap \text{dom} \gamma_2 \), and thus \( \gamma_1(x) = \gamma_2(x) \) by the consistency of \( \gamma_1 \) and \( \gamma_2 \), thus it follows:
Thus $\beta(\gamma_1)$ and $\beta(\gamma_2)$ are consistent.

Next, we prove that $\beta(\gamma_1 \uparrow \gamma_2) = \beta(\gamma_1) \uparrow \beta(\gamma_2)$. The proof is via an analysis of three cases:

1. $x \in \text{dom } \gamma_1 \land x \in \text{dom } \gamma_2$
2. $x \in \text{dom } \gamma_1 \land x \not\in \text{dom } \gamma_2$
3. $x \not\in \text{dom } \gamma_1 \land x \in \text{dom } \gamma_2$

**Case (i) $x \in \text{dom } \gamma_1 \land x \in \text{dom } \gamma_2$**

$$\beta(\gamma_1 \uparrow \gamma_2)(x) = \beta(\gamma_1)(x)$$

Similarly, since $\gamma_1(x) = \gamma_2(x)$ (by consistency of $\gamma_1$ and $\gamma_2$), it follows:

$$\beta(\gamma_1 \uparrow \gamma_2)(x) = \beta(\gamma_2)(x)$$

It follows that $\beta(\gamma_1 \uparrow \gamma_2) = \beta(\gamma_1) \uparrow \beta(\gamma_2)$ for all $x \in \text{dom } \gamma_1 \cap \text{dom } \gamma_2$, and thus case (i) is proved.

**Case (ii) $x \in \text{dom } \gamma_1 \land x \not\in \text{dom } \gamma_2$**

$$\beta(\gamma_1 \uparrow \gamma_2)(x) = \beta(\gamma_1)(x)$$

Clearly $x \not\in \beta(\gamma_2)$ since $x \not\in \gamma_2$, and so $\beta(\gamma_1 \uparrow \gamma_2)(x) = \beta(\gamma_1)(x) = (\beta(\gamma_1) \uparrow \beta(\gamma_2))(x)$. Thus
\[ \beta(\gamma_1 \uparrow \gamma_2) = \beta(\gamma_1) \uparrow \beta(\gamma_2) \]

for all \( x \) such that \( x \in \text{dom} \gamma_1, x \notin \text{dom} \gamma_2 \).

Case (iii) is similar, and is not elaborated.

Thus all three cases of the lemma are proved.

\[ \beta(\gamma_1 \uparrow \gamma_2) = \beta(\gamma_1) \uparrow \beta(\gamma_2) \]

This result may be extended via an inductive argument to yield the following:

\[ \beta^n(\gamma_1 \uparrow \gamma_2) = \beta^n(\gamma_1) \uparrow \beta^n(\gamma_2) \]

**Lemma 4.4** Suppose \( \gamma_1 \) and \( \gamma_2 \) are consistent bill of materials, then \( \beta^n(\gamma_1 \uparrow \gamma_2) = \beta^n(\gamma_1) \uparrow \beta^n(\gamma_2) \); furthermore \( \beta^n(\gamma_1) \) and \( \beta^n(\gamma_2) \) are consistent.

Lemma 4.3 has demonstrated that for consistent \( \gamma_1 \) and \( \gamma_2 \) that \( \beta(\gamma_1 \uparrow \gamma_2) = \beta(\gamma_1) \uparrow \beta(\gamma_2) \), and furthermore \( \beta(\gamma_1) \) is consistent with \( \beta(\gamma_2) \). Suppose this is true for \( n = k \), then \( \beta^k(\gamma_1 \uparrow \gamma_2) = \beta^k(\gamma_1) \uparrow \beta^k(\gamma_2) \) and \( \beta^k(\gamma_1) \) is consistent with \( \beta^k(\gamma_2) \) then it follows:

\[
\beta^{k+1}(\gamma_1 \uparrow \gamma_2) = \beta^k(\gamma_1 \uparrow \gamma_2) \\
= \beta(\beta^k(\gamma_1 \uparrow \gamma_2)) \\
= \beta(\beta^k(\gamma_1) \uparrow \beta^k(\gamma_2)) \\
= \beta(\beta^k(\gamma_1)) \uparrow \beta(\beta^k(\gamma_2)) \\
= \beta^{k+1}(\gamma_1) \uparrow \beta^{k+1}(\gamma_2)
\]

and the lemma is proved.

**Theorem 4.11** Suppose \( \gamma_1 \) and \( \gamma_2 \) are consistent bills of materials, then \( \text{Join}_0(\gamma_1, \gamma_2) \) is a bill of material.

**Proof** (Annihilator version)

It is required is to prove that \( \mathcal{A}(\gamma_1 \uparrow \gamma_2) = \theta \). Since \( \gamma_1 \) is a bill of materials it follows that \( \beta^{n_1}(\gamma_1) = \theta \) for some \( n_1 \); similarly \( \beta^{n_2}(\gamma_2) = \theta \) for some \( n_2 \). Let \( n = \max(n_1, n_2) \), then we claim \( \beta^n(\gamma_1 \uparrow \gamma_2) = \theta \). Lemma 4.4 demonstrates that for consistent \( \gamma_1, \gamma_2 \) that \( \beta^n(\gamma_1 \uparrow \gamma_2) = \beta^n(\gamma_1) \uparrow \beta^n(\gamma_2) \). Thus we have:

\[
\mathcal{A}(\gamma_1 \uparrow \gamma_2) = \beta^n(\gamma_1 \uparrow \gamma_2) \\
= \beta^n(\gamma_1) \uparrow \beta^n(\gamma_2) \\
= \mathcal{A}(\gamma_1) \uparrow \mathcal{A}(\gamma_2) \\
= \theta \uparrow \theta = \theta
\]

thus we have \( \mathcal{A}(\gamma_1 \uparrow \gamma_2) = \theta \) as required.

**Conclusion 4.1** The annihilator function is a very nice theoretical construct, however proofs involving the annihilator function are slightly more involved than a corresponding proof using the equivalent Bjørner invariant.
4.6.1 Simulation of $Ent_0$ operation by $Join_0$

It has been demonstrated that a bill of materials is formed from the ground up; in particular, the $Ent_0$ operation builds the bill of materials from the ground up. It has been proved that the $Ent_0\{x, S\}^\gamma_0$ preserves the invariant. The objective here is to demonstrate that the $Join_0$ operation may be used to simulate the $Ent_0$ operation; this is achieved by joining the minimal consistent bill of material with $\gamma$ containing $[x \mapsto S]$.

It is required to demonstrate that a minimal consistent bill of materials with $\gamma$ may be constructed. This is achieved by the $Span_x[x] \gamma$ operation, which takes a bill of material $\gamma$ and a new assembly $[x \mapsto S]$, and creates the minimal bill of material containing $[x \mapsto S]$. The operation and its precondition are defined as follows:

$$Span_x : X \mapsto \mathcal{P}X \mapsto (X \mapsto \mathcal{P}X) \mapsto (X \mapsto \mathcal{P}X)$$

$$Span_x[S]\gamma \triangleq Span[S, \gamma][x \mapsto S]$$

$$pre\_Span_x : X \mapsto \mathcal{P}X \mapsto (X \mapsto \mathcal{P}X) \mapsto B$$

$$pre\_Span_x[S]\gamma \triangleq x \not\in \gamma \land S \subseteq \gamma$$

The $Span$ operation creates the minimal bill of material containing $S$, which is distinct from the minimal bill of material containing $[x \mapsto S]$, and is defined as follows:

$$Span : \mathcal{P}X \times (X \mapsto \mathcal{P}X) \mapsto (X \mapsto \mathcal{P}X) \mapsto (X \mapsto \mathcal{P}X)$$

$$Span[\emptyset, \gamma]\mu \triangleq \mu$$

$$Span[\{y\} \uplus S', \gamma]\mu \triangleq$$

$$\begin{array}{l}
y \in \mu \\
\Rightarrow Span[S', \gamma]\mu
\end{array}$$

Let $V_y = \{z | y R^+ z \text{ for some } z \in \gamma\}$ in

$$\Rightarrow Span[S', \gamma] \circ Span[V_y, \gamma](\mu \uplus [y \mapsto \gamma(y)])$$

Note

$Span_x[S]\gamma$ may also be constructed via the intersection of all bills of material consistent with $\gamma$ that contain the assembly $[x \mapsto S]$. This approach is analogous to finding the smallest subgroup $H$ of a group $G$ that contains the set $S$.

Theorem 4.12 $pre\_Span_x[S]\gamma \land \mu = Span_x[S]\gamma \Rightarrow \mathcal{A}\mu = \emptyset$. Furthermore, $\mu$ is the minimal bill of material consistent with $\gamma$ that contains $S$.

Proof
Suppose $\mu$ is not a bill of material, then either part (a) or (b) of Björner's invariant fails; suppose part (a) fails. Then $\exists y \in \text{rng} \mu$ such that $y \notin \text{dom} \mu$. Clearly $y \neq x$ as $x \notin \text{rng} \mu$.

Now $y \in \text{rng} \mu \Rightarrow y \in \mu(z)$ for some $z \in \mu$. Thus $y \in V_z$ where $V_z = \{w | zR^+_\mu w \text{ for some } w \in \mu\}$ However $y \in V_z$ as generated by $\mu$ means $y \in W_z$ where $W_z = \{w | zR^+w \text{ for some } w \in \gamma\}$, i.e., $W_z$ is the equivalent $V_z$ generated by $\gamma$; this follows since $\text{dom } \phi_{\gamma} \subseteq \text{dom } \gamma$ and $\mu(z) = \gamma(z)$, for all $z \in \phi_{\gamma}(x); \mu$; thus $\mu$ is consistent with $\gamma$.

Thus we have $y \in W_z$, however from the definition of $\text{Span}$ this ensures $y \in \text{dom } \mu$ which is a contradiction. Thus part (a) of the invariant must hold.

Suppose part (b) fails to hold, then there is some $y \in \mu$ such that $y \in \text{Parts}(y, \gamma)$, i.e., $y \in V_y = \{z | yR^+_\mu z \text{ for some } z\}$. However, as in part (a), $V_y = W_y$ where $W_y$ is as before. Thus $yR^+y$ in $\gamma$, which is a contradiction since $\gamma$ is a bill of material.

**Theorem 4.13** $\text{Ent}_0[x, S][\gamma] = \text{Join}(\gamma, \text{Span}_x[S, \gamma])$

**Proof**

Theorem 4.12 has demonstrated that $\text{Span}_x[S][\gamma]$ produces a bill of material which is consistent with $\gamma$. Furthermore, Theorem 4.11 has proved that the join of two consistent bills of materials yields another bill of material. Clearly $\text{dom } \text{Ent}_0[x, S][\gamma] = \{x\} \cup \text{dom } \gamma$. Furthermore, $\{x\} \cup S \subseteq \text{dom } \text{Span}_x[S][\gamma]$ and $\text{dom } \text{Join}(\gamma, \text{Span}_x[S][\gamma]) = \text{dom } \gamma \cup \text{dom } \text{Span}_x[S][\gamma] = \{x\} \cup \text{dom } \gamma$.

Clearly, $(\text{Ent}_0[x, S][\gamma])(x) = S = \text{Join}(\gamma, \text{Span}_x[S, \gamma])(x)$. Similarly, for $y \neq x$ where $y \in \gamma$, we have $\text{Ent}_0[x, S][\gamma](y) = \gamma(y)$. Since $\text{Join}(\gamma, \text{Span}_x[S, \gamma])$ is consistent with $\gamma$ we have as required that $\text{Join}(\gamma, \text{Span}_x[S, \gamma])(y) = \gamma(y)$

**4.7 Summary**

The objective of this chapter is to obtain a more detailed understanding of the bill of materials structure. The key motivation for this study is the realization that bill of materials are abstractions of hierarchies. The study of the model of religion and the stock exchange has demonstrated that formal methods may successfully model the hierarchical structure of organizations. Consequently, it is appropriate to gain a more detailed understanding of the abstract structure of a hierarchy.

The theoretical annihilator function in [46] has been placed on a sounder mathematical platform. The examination of the annihilator function led to the identification of the
shrinking operation $\beta$. The importance of this operation is that it shrinks a bill of material, but preserves it as a bill of material. This result is believed to be new.

The results on joining bill of materials is believed to be new. The key constraint in joining two bill of materials is that they should be consistent. This requires that all common elements in the two bill of materials (whether parts or assemblies) should have identical definitions.

The relationship between invariants and proof obligation complexity has been considered in this chapter. The Björner invariant and the annihilator formulation of the invariant were evaluated to determine if the proof obligations are less complex with the latter. The results (on the limited examples considered) suggest that the Björner invariant is slightly easier to work with than the theoretical annihilator function.
Chapter 5

The File System

5.1 Introduction

The specification of a file system is one of the earliest specifications in the VDM literature [10]. In recent years, [32] the $\mathcal{Z}$ community have developed a formal specification of the UNIX file system. The original VDM specification was used as an example in [45] to demonstrate the effectiveness of the operator calculus of the Irish school of VDM, and to demonstrate that constructive mathematical proof may be employed instead of formal logic.

The study of the religion model in Chapter 2 has demonstrated that formal methods may be successfully employed to models aspects of the hierarchical structure of organizations and beliefs. The original VDM model of the file system introduced catalogues, where each file has an associated catalogue of information on its pages. The model of an organization must record relevant information on the organization, i.e., a catalogue of information must be maintained.

Consequently, the objective of this chapter is to present a simpler and more readable model of the file system. This enables an understanding of the abstraction of a catalogue to be gained. Finally, the model includes file aliasing.

5.2 Abstract Model of File System

The initial model of the file system in [10] is incorrect. The error in the specification is an excessively liberal invariant which places no constraints on what a file may be. This problem is identified in [45], and a solution to the problem is presented. However, the problem with the invariant presented in [45] is that it is excessively complex, and may be simplified further. The principle of ‘Ockham’s Razor’, i.e., ‘Entia non sunt multiplicanda praeter necessitatem’, suggests that the simplest model should be chose. It is also important
to choose the simplest invariant as the proofs of invariant preservation are simplified.

\[ F_{S_0} = F_n \mapsto FILE \]  \hspace{1cm} (5.1)
\[ FILE = P_n \mapsto P_g \]  \hspace{1cm} (5.2)

The file system is considered to be a map from file names, \( F_n \) to files \( FILE \). A file is a map from page names, \( P_n \), to pages, \( P_g \). The invariant places no constraints on the domain \( F_{S_0} \). Thus anything with this signature is considered to be a file.

\[ inv_{F_{S_0}} \triangleq TRUE \]

The problem with the invariant is evident from considering the example:

\[ \phi = [f_1 \mapsto [p_{n_1} \mapsto pg_1, p_{n_2} \mapsto pg_2], f_2 \mapsto [p_{n_2} \mapsto pg_2, p_{n_3} \mapsto pg_3]] \]

This allows two distinct files, \( f_1, f_2 \) to share the same page \( pg_2 \), and also to share the same page name \( p_{n_2} \). This behaviour is not consistent with that expected of a file system, where pages and page names are disjoint among files, with the exception of file aliasing. 

MacAnAirchinnigh [45] presents a corrected invariant.

\[ inv_{F_{S_0}} : F_{S_0} \mapsto B \]
\[ inv_{F_{S_0}}(\phi) \triangleq \]
\[ \land / \circ Prng \circ \text{rng} \phi = \lor / Prng \circ \text{rng} \phi \]
\[ \land / \circ \text{Pj} \circ Prng \circ \text{rng} \phi = \lor / Prng \circ \text{rng} \phi \]
\[ \land / \circ \text{Dom} \circ \text{rng} \phi = \lor / \text{Dom} \circ \text{rng} \phi \]
\[ \land / \circ (\text{Dom} \circ \circ) = (\lor)^{-1} \circ \text{rng} \circ (\lor)^{-1} \circ \text{Pj} \circ \text{rng} \phi \]

This invariant is incomprehensible to those who are unfamiliar with the background material in [45]. The invariant is excessively complex and unreadable, in essence, it states that the set of all pages in the file system is partitioned, and the set of all page names in the file system is partitioned. The last part of the invariant specifies that a one to one correspondence property should be satisfied.

The use of operator overloading within the invariant contributes to the readability difficulties; in particular, the application of the operator \( j(e) \) means the construction of a singleton bag \( [e \mapsto 1] \) in one instance, and the construction of a singleton sequence \( \langle e \rangle \) in another instance.
The final part of the invariant is unnecessary as it is a tautology; i.e., \(^/ \circ (\text{dom } = \langle - \rangle^{-1} \circ \text{rng } - \rangle \circ \mathcal{P}j \circ \text{rng } \phi)\) is a tautology, and consequently is not required in the invariant, and by Ockham’s Razor should not be present.

This requirement essentially states that \(\text{dom } \mu = \mu^{-1} \circ \text{rng } \mu\) (a strict formulation is \(\text{dom } \mu = \mathcal{U} / \circ \mathcal{P} \mu^{-1} \circ \text{rng } \mu\)) for each map \(\mu \in \text{rng } \phi\). However, this requirement holds in general, for all maps \(\mu\), as can be seen by.

**Lemma 5.1** \(\text{dom } \mu = \mu^{-1} \circ \text{rng } \mu\).

**Proof**
Let \(x \in \text{dom } \mu\), then \(\mu(x) \in \text{rng } \mu\), and thus \(x \in \mu^{-1}(\mu(x))\), and so \(\text{dom } \mu \subseteq \mu^{-1} \circ \text{rng } \mu\).

Suppose \(x \in \mu^{-1} \circ \text{rng } \mu\), then \(\mu(x) \in \text{rng } \mu\), and thus \(x \in \text{dom } \mu\), so \(\mu^{-1} \circ \text{rng } \mu \subseteq \text{dom } \mu\).

**Comment 5.1 (Simplification of Invariant)** If an invariant is complex and unreadable, it should be re-examined. A simpler form may exist.

**The Invariant**
A simpler form of the invariant may be derived from the above. In particular, The simplified invariant is presented as follows:

\[
\text{inv}_F S_0 : F S_0 \mapsto B \\
\text{inv}_F S_0(\phi) \triangleq \\
\begin{align*}
+ / \circ \text{card } \circ \text{rng } \circ \mathcal{P}j \circ \text{rng } \phi &= |\mathcal{U} / \circ \mathcal{P} \text{rng } \circ \text{rng } \phi| \\
\wedge^+ / \circ \text{card } \circ \text{dom } \circ \mathcal{P}j \circ \text{rng } \phi &= |\mathcal{U} / \circ \mathcal{P} \text{dom } \circ \text{rng } \phi|
\end{align*}
\]

**Note**
The notation \(j\) converts an arbitrary element \(s\) to a sequence \(\langle s \rangle\). The notation \(\mathcal{P}j\) converts a set \(S = \{s_1, ..., s_n\}\) to a sequence \(\langle s_1 \rangle \cap ... \cap \langle s_n \rangle\).

It is required to demonstrate that this simplified invariant is equivalent to the more complex form, this is achieved by showing the simplified form is at least as strong as the complex form, and vice versa.

**Lemma 5.2** \(+ / \circ \text{card } \circ \text{rng } \circ \mathcal{P}j \circ \text{rng } \phi = |\mathcal{U} / \circ \mathcal{P} \text{rng } \circ \text{rng } \phi| \Rightarrow (\text{inv}_F S_0(\phi) \Rightarrow \wedge^+ / \circ \mathcal{P} \text{rng } \circ \text{rng } \phi = \mathcal{U} / \mathcal{P} \text{rng } \circ \text{rng } \phi)\)

**Proof**
The result is achieved by the contrapositive, i.e., \((p \Rightarrow q) \iff (\neg q \Rightarrow \neg p)\). A well known result from set theory, is that for finite sets \(S_1, S_2, |S_1 \cup S_2| \leq |S_1| + |S_2|\), with equality
only when $S_1 \cap S_2 = \emptyset$.

Let $\mathcal{P} \circ \text{rng} \phi = \{S_1, S_2, \ldots, S_k\}$, then $\Delta / \mathcal{P} \circ \text{rng} \phi = S_1 \triangle S_2 \ldots \triangle S_k$, and $\cup / \mathcal{P} \circ \text{rng} \phi = S_1 \cup S_2 \ldots \cup S_k$.

Suppose $\Delta / \mathcal{P} \circ \text{rng} \phi \not\subseteq \cup / \mathcal{P} \circ \text{rng} \phi$.

Then $S_1 \triangle S_2 \ldots \triangle S_k \subset S_1 \cup S_2 \ldots \cup S_k$, $\Rightarrow S_i \cap S_j \neq \emptyset$, for some $S_i, S_j$ where $i \neq j$.

Consider $S'_i = \bigcup_{k \neq j} S_k$, and $S'_j = S_j$, then $S'_i \cap S'_j \neq \emptyset$, and thus:

$$|\cup / \mathcal{P} \circ \text{rng} \phi| = |\bigcup_{k \neq j} S_k| + |S_j| \leq \sum_{k \neq j} |S_k| + |S_j| = |S_1| + |S_2| + \ldots + |S_k| \leq \binom{\cup / \mathcal{P} \circ \text{rng} \phi}{}$$

Thus $|\cup / \mathcal{P} \circ \text{rng} \phi| \neq \cup / \mathcal{P} \circ \text{rng} \phi$, and the result follows.

**Lemma 5.3** $(\cup / \mathcal{P} \circ \text{rng} \phi) = |\cup / \mathcal{P} \circ \text{rng} \phi| \Rightarrow |\cup / \mathcal{P} \circ \text{rng} \phi| = |\cup / \mathcal{P} \circ \text{rng} \phi|$.

**Proof**

The operation $j$ is overloaded in the statement of the lemma; in one instance it refers to sequence construction, the second case is singleton bag construction.

This result is achieved by using the contrapositive as before. Suppose $|\cup / \mathcal{P} \circ \text{rng} \phi| \not= |\cup / \mathcal{P} \circ \text{rng} \phi|$.

Then there is $p_\phi \in \cup / \mathcal{P} \circ \text{rng} \phi$ such that $p_\phi \in \text{rng} \mu_1$ and $p_\phi \in \text{rng} \mu_2$ where $\mu_1 \neq \mu_2$ and $\mu_1, \mu_2 \in \text{rng} \phi$.

Then clearly $\langle S_1 \rangle, \langle S_2 \rangle$, where $S_1, S_2$ are the corresponding sequences formed from $\mu_1, \mu_2$ respectively both contain $p_\phi$, and $S_1 \neq S_2$.

Consequently, $(\cup / \mathcal{P} \circ \text{rng} \phi) > |\cup / \mathcal{P} \circ \text{rng} \phi|$, as required.

The two lemmas have demonstrated that the simplified form of the invariant is as strong as the original complex form. In fact, both invariants are equivalent. The pages and page names of the physical files partition the page space, and page name space respectively. This is similar (but not equivalent) to the effect of an inverse image of a map, yielding a partition on the domain.
Properties of File System

The model of the file system is of the form $\phi : Fn \mapsto (Pn \mapsto Pg)$. Applying the functionals $(I \mapsto \text{dom})$ and $(I \mapsto \text{rng})$ to $\phi$ yields the relationship between file names and page names, and file names and pages, respectively.

$$\phi_n : Fn \mapsto \mathcal{P}Pn = (I \mapsto \text{dom})\phi$$
$$\phi_p : Fn \mapsto \mathcal{P}Pg = (I \mapsto \text{rng})\phi$$

The key point to note is the following:

$$(\forall fn, fm \in \phi_n)(\phi_n(fn) = \phi_n(fm)) \lor (\phi_n(fn) \cap \phi_n(fm) = \emptyset)$$
$$(\forall fn, fm \in \phi_p)(\phi_p(fn) = \phi_p(fm)) \lor (\phi_p(fn) \cap \phi_p(fm) = \emptyset)$$

File System Operations

The next stage is to present the operations for the initial model, there are five of them. These operations have been adopted from [10] and [45] to address file aliasing.

$$Cmd_0 = \text{Create}_0|\text{Erase}_0|\text{Put}_0|\text{Get}_0|\text{Delete}_0$$

These operations respectively create, erase, add a page to a file, get a page from a file, and delete a page from a file. The corresponding proof obligation for each operation is simplified, by employing the less complex form of the invariant.

The Create operation

$$\text{pre}_\text{Create}_0 : Fn \mapsto (Fn \mapsto \text{FILE}) \mapsto B$$
$$\text{pre}_\text{Create}_0[fn] \phi \triangleq \chi[fn] \phi$$

$$\text{Create}_0 : Fn \mapsto (Fn \mapsto \text{FILE}) \mapsto (Fn \mapsto \text{FILE})$$
$$\text{Create}_0[fn] \phi \triangleq \phi \cup [fn \mapsto \theta]$$

Lemma 5.4 $\text{pre}_\text{Create}_0[fn] \phi \land \phi' = \text{Create}_0[fn] \phi \Rightarrow Inv_{FS_0}[\phi'].$

Proof

$$\text{pre}_\text{Create}_0[fn] \phi \land \phi' = \cup_\phi \circ \text{Prng} \circ \text{rng} (\phi \cup [fn \mapsto \theta])$$
$$= \cup_\phi \circ \text{Prng} \circ \text{rng} (\phi \cup \emptyset)$$
$$= \cup_\phi \circ \text{Prng} \circ \text{rng} \phi$$
\[ +/ \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P} \circ \text{rng} \phi = +/ \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P} \circ \text{rng} \phi \]

The argument for $\mathcal{P}\text{dom} \circ \text{rng} \phi'$ is similar.

**The Erase operation**

\[
\text{pre}_{\text{Eras}_0} : \text{Fn} \mapsto (\text{Fn} \mapsto \text{FILE}) \mapsto \text{B}
\]

\[
\text{pre}_{\text{Eras}_0}[[\text{fn}]\phi \triangleq \neg \chi[[\text{fn}]\phi)
\]

\[
\text{Eras}_0 : \text{Fn} \mapsto (\text{Fn} \mapsto \text{FILE}) \mapsto (\text{Fn} \mapsto \text{FILE})
\]

\[
\text{Eras}_0[[\text{fn}]\phi \triangleq \text{Inv}_{FS_0}[[\phi']]
\]

**Comment 5.2 (Aliases)** The definition of the file system permits aliases, thus several file names may represent the same physical file. Thus the deletion of a file from the file system may have no impact on the pages and page names in the system, if an alias for the file remains. Otherwise the page names and physical page names are removed from the system. There are consequently two cases for the proof of invariant preservation.

**Comment 5.3 (Aliases and Empty files)** The first reification of the file system in the literature [10] prohibits aliases by employing one to one maps. However, immediately on inspection of the $\text{Cre}_{a_1}$ operation in [10], it is evident that if the operation is invoked twice in succession, then the one to one constraints of the file system are violated. Furthermore, the limitation to a maximum of one empty file in the file system is too restrictive and unrealistic.

**Lemma 5.5** $\text{pre}_{\text{Eras}_0[[\text{fn}]\phi \wedge \phi'] = \text{Eras}_0[[\text{fn}]\phi \Rightarrow \text{Inv}_{FS_0}[[\phi']]$}

**Proof**

There are two cases depending on whether the file to be deleted has aliases. Suppose $|\phi^{-1} \circ \phi(\text{fn})| > 1$, then $\text{fn}$ has aliases, i.e., $\phi(\text{fm}) = \phi(\text{fn}) = \mu$ for some $\text{fm} \neq \text{fn}$. Then $\text{rng} \circ \text{Eras}_0[[\text{fn}]\phi = \text{rng} \phi' = \text{rng} \circ \text{pre}[[\text{fn}]\phi = \text{rng} \phi$, since $\text{fm} \in \text{pre}[[\text{fn}]\phi$. Thus $\mathcal{P}\text{rng} \circ \text{rng} \phi' = \mathcal{P}\text{rng} \circ \text{rng} \phi$, and thus:

\[
+/- \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P} \circ \text{rng} \phi = +/- \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P} \circ \text{rng} \phi
\]

\[
= |/- \circ \mathcal{P}\text{rng} \circ \text{rng} \phi|
\]

\[
= |/- \circ \mathcal{P}\text{rng} \circ \text{rng} \phi|
\]

Suppose $|\phi^{-1} \circ \phi(\text{fn})| = 1$, then $\text{fn}$ has no aliases, and $\text{rng} \phi' = \text{rng} \circ \text{pre}[[\text{fn}]\phi = \text{pre}[[\phi(\text{fn})]\text{rng} \phi$. Thus $\phi(\text{fn}) \cap \text{rng} \circ \text{pre}[[\text{fn}]\phi = \emptyset$.  

109
\[ = \cup / \circ \mathcal{P}_{\text{rng}} \circ \text{rng} \downarrow [fn] \phi \]

\[ = \cup / \circ \mathcal{P}_{\text{rng}} \circ \text{rng} \phi' \]

Thus \( \cup / \circ \mathcal{P}_{\text{rng}} \circ \text{rng} \phi' \cap \text{rng} \phi(fn) = \emptyset \).

\[ |\cup / \circ \mathcal{P}_{\text{rng}} \circ \text{rng} \phi'| + |\text{rng} \phi(fn)| = \]

\[ \geq |\cup / \circ \mathcal{P}_{\text{rng}} \circ \text{rng} \phi' \cup \text{rng} \phi(fn)| \quad \text{(Disjoint union)} \]

\[ = |\cup / \circ \mathcal{P}_{\text{rng}} \circ \text{rng} \phi| \]

Thus \( + / \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P} \circ \text{rng} \phi' = |\cup / \circ \mathcal{P}_{\text{rng}} \circ \text{rng} \phi'| \) as required.

The proof for \( |\mathcal{P}_{\text{dom}} \circ \text{rng} \phi'| \) is similar.

**The Put operation**

This operation adds a page to an existing file, however, care is needed as the file may have aliases. If this is the case, then all aliases must be updated. Failure to do this will result in invariant violation, as the addition of \((pn_3, pg_3)\) to file \( f_1 \) in the following example demonstrates:

\[ \phi = [f_1 \mapsto [pn_1 \mapsto pg_1, pn_2 \mapsto pg_2], f_2 \mapsto [pn_1 \mapsto pg_1, pn_2 \mapsto pg_2]] \]

\( f_2 \) is an alias of \( f_1 \); suppose a naive put operation, \( \text{Put}[[fn_1, pn_3, pg_3] \phi] \), is defined, its effect being to update \( f_1 \), but not \( f_2 \).

\[ \phi' = [f_1 \mapsto [pn_1 \mapsto pg_1, pn_2 \mapsto pg_2, pn_3 \mapsto pg_3], f_2 \mapsto [pn_1 \mapsto pg_1, pn_2 \mapsto pg_2]] \]

Then \( \phi' \) violates the invariant; this is since \( f_1, f_2 \) represent two distinct files which share common pages. The precondition for the naive put operation is presented below in page 435 of [45] as follows:

\[ \text{pre}_{\text{Put}_0} : Fn \times Pn \times Pg \mapsto (Fn \mapsto (Pn \mapsto Pg)) \mapsto B \]

\[ \text{pre}_{\text{Put}_0}[fn, pn, pg] \phi \triangleq \]

\[ \chi[fn] \phi \]

\[ \land pg \in \cup / \circ \mathcal{P}_{\text{rng}} \circ \text{rng} \phi \Rightarrow pg \in \text{rng} \phi(fn) \]

\[ \land pn \in \cup / \circ \mathcal{P}_{\text{dom}} \circ \text{rng} \phi \Rightarrow pn \in \phi(fn) \]

The precondition for the \( \text{Put}_0 \) operation in [45] is incomplete, as it should additionally ensure that the page name \( pn \), is not present in any other file which is not an alias of \( fn \).
This has been corrected here to ensure that the page name \( pn \), and the physical page \( pg \), to be added, do not appear in any other file which is not an alias of \( fn \).

**Comment 5.4 (Aliases)** The system designer is faced with a fundamental design choice with aliasing, whether these should be permitted or forbidden. Either choice has a corresponding effect on the model. The approach taken here is to recognize aliasing as a legitimate requirement in a file system, thus operations, their preconditions and system invariants have been tailored to permit aliasing.

The naive \( \text{Put}_0 \) operation is presented as follows, and is correct if \( fn \) has no aliases, i.e., \( |\phi^{-1} \circ \phi(fn)| = 1 \).

\[
\text{Put}_0 : Fn \times Pn \times Pg \rightarrow (Fn \mapsto (Pn \mapsto Pg)) \mapsto (Fn \mapsto (Pn \mapsto Pg))
\]

\[
\text{Put}_0[fn, pn, pg] \phi \triangleq \\
\chi[pn] \phi(fn) \\
\mapsto \phi \uparrow [fn \mapsto \phi(fn) \uparrow [pn \mapsto pg]] \\
\mapsto \phi \uparrow [fn \mapsto \phi(fn) \cup [pn \mapsto pg]]
\]

**Lemma 5.6** \( \text{pre} \cdot \text{Put}_0[fn, pn, pg] \phi \land \mathcal{E} = \text{Put}_0[fn, pn, pg] \phi \Rightarrow \text{Inv} \cdot F S_0[\mathcal{E}] \)

**Proof**

\( |\phi^{-1} \circ \phi(fn)| = 1 \), thus there are two cases, \( pn \in \phi(fn) \) and \( pn \notin \phi(fn) \). Suppose \( pn \in \phi(fn) \), then \( \mathcal{E} = \phi \uparrow [fn \mapsto \mu] \), where \( \mu = \phi(fn) \uparrow [pn \mapsto pg] \). Clearly \( \text{rng} \mu \subseteq \text{rng} \phi(fn) \cup \{ pg \} \). Combining the precondition and the invariant, it follows that:

\[
\text{rng} \mu \cap \cup / \circ P \text{rng} \circ \text{rng} \not<_s [fn] \phi \\
\subseteq (\text{rng} \phi(fn) \cup \{ pg \}) \cap \cup / \circ P \text{rng} \circ \text{rng} \not<_s [fn] \phi \\
= \text{rng} \phi(fn) \cap \cup / \circ P \text{rng} \circ \text{rng} \not<_s [fn] \phi \cup \{ pg \} \cap \cup / \circ P \text{rng} \circ \text{rng} \not<_s [fn] \phi \\
= \emptyset \cup \emptyset \\
= \emptyset.
\]

\[
\begin{align*}
P \text{rng} \circ \text{rng} \mathcal{E} & = P \text{rng} \circ \text{rng} \not<_s [fn] \phi \cup [fn \mapsto \mu] \\
& = P \text{rng} \circ (\text{rng} \not<_s [fn] \phi \cup \text{rng} \mu).
\end{align*}
\]

\[
\begin{align*}
\cup / \circ \text{card}^* \circ \text{rng}^* \circ P j \circ \text{rng} \not<_s [fn] \phi + \text{rng} \mu \\
= \cup / P \text{rng} \circ \text{rng} \not<_s [fn] \phi \cup \text{rng} \mu \\
\end{align*}
\]

\[
\begin{align*}
\cup / \circ \text{card}^* \circ \text{rng}^* \circ P j \circ \text{rng} \not<_s [fn] \phi \\
& = \cup / P \text{rng} \circ (\text{rng} \not<_s [fn] \phi \cup [fn \mapsto \mu]).
\end{align*}
\]
Suppose \( pn \notin \phi(fn) \), then \( \phi' = \phi \upharpoonright [fn \mapsto \mu] \) where \( \mu = \phi(fn) \cup [pn \mapsto pg] \). Furthermore, \( \text{rng} \mu \cap \uparrow \mathcal{P} \text{rng} \circ \text{rng} \phi = \emptyset \), as before. The remainder of the proof is exactly as before. The proof for \( |\mathcal{P}\text{dom} \circ \text{rng} \phi| \) is similar.

There is a hidden lemma in the proof, its statement and proof are as follows.

**Lemma 5.7** \(+/\circ \text{card}^{*} \circ \text{rng}^{*} \circ \mathcal{P} \circ \text{rng} \phi = |\uparrow \mathcal{P} \text{rng} \circ \text{rng} \phi| \Rightarrow +/\circ \text{card}^{*} \circ \text{rng}^{*} \circ \mathcal{P} \circ \text{rng} \phi = \uparrow |\mathcal{P} \text{rng} \circ \text{rng} \phi| = |\mathcal{P} \text{rng} \circ \text{rng} \phi|\)

**Proof**
The proof is by the contrapositive, suppose \(+/\circ \text{card}^{*} \circ \text{rng}^{*} \circ \mathcal{P} \circ \text{rng} \phi = \uparrow \mathcal{P} \text{rng} \circ \text{rng} \phi \neq |\mathcal{P} \text{rng} \circ \text{rng} \phi|\) then

\[\Rightarrow +/\circ \text{card}^{*} \circ \text{rng}^{*} \circ \mathcal{P} \circ \text{rng} \phi = \uparrow |\mathcal{P} \text{rng} \circ \text{rng} \phi|\]

which is the required result.

**The Careful Put operation**

The problem with the definition of the \( \text{Put}_0 \) operation is that it is invalid if the file system contains aliases, as the invariant is violated. The key to the correction of the definition, is to ensure that each alias of \( fn \) is updated with the new page. In order for two file names to be aliases, they must refer to the same non empty physical file.

**Definition 5.1 (Aliases)** \( fn \) is an alias of \( fn \) if \( \phi(fn) = \phi(fm) \) and \( \phi(fn) \neq \emptyset \).

\( \text{Put}_0 : Fn \times Pn \times Pg \mapsto (Fn \mapsto (Pn \mapsto Pg)) \mapsto (Fm \mapsto (Pm \mapsto Pg)) \)

\( \text{Put}_0[fn, pn, pg] \phi = \)

<table>
<thead>
<tr>
<th>( \phi(fn) \neq \emptyset )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A \mapsto \phi^{-1} \circ \phi(fn) )</td>
</tr>
<tr>
<td>( A \mapsto {fn} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( \chi[pn] \phi(fn) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mu \mapsto \phi(fn) \upharpoonright [pn \mapsto pg] )</td>
</tr>
<tr>
<td>( \mu \mapsto \phi(fn) \cup [pn \mapsto pg] )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( \phi_A \mapsto \lambda a : A \bullet \mu )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mapsto \phi \upharpoonright \phi_A )</td>
</tr>
</tbody>
</table>

There is no change to the precondition for the operation, the associated proof of invariant preservation is as follows. If \( \phi(fn) = \emptyset \), then as the physical file is empty, and contains
no pages, it is not considered to have any aliases, irrespective of whether there are other empty files. The other empty files are considered to represent distinct files. It is assumed that there is a \( M_{k_{\mathcal{A}}}[f_1, f_2] \) operation which makes \( f_1 \) an alias of \( f_2 \) provided certain conditions hold.

**Lemma 5.8** \( \preceq \text{Put}_0 \llbracket fn, pn, pg \rrbracket \phi \land \phi' = \text{Put}_0 \llbracket fn, pn, pg \rrbracket \phi \Rightarrow \text{Inv}_FSO[\phi'] \)

**Proof**

If \( |\phi^{-1} \circ \phi(fn)| = 1 \), i.e., \( fn \) has no aliases, then \( \phi_A = [fn \mapsto \mu] \) and the Put operation is as before. Suppose \( |\phi^{-1} \circ \phi(fn)| \neq 1 \), there are two cases \( \phi(fn) = \theta \) and \( \phi(fn) \neq \theta \). Suppose \( \phi(fn) = \theta \), then \( \phi_A = [fn \mapsto \mu] \), and \( \text{rng} \mu = \{pg\} \),

\[
\begin{align*}
\text{\text{\textbackslash /}} & / \text{Prng} \circ \text{rng} \phi' \\
= & \text{\text{\textbackslash /}} / \text{Prng} \circ \text{rng} (\exists [fn] \phi \sqcup [fn \mapsto \mu]) \\
= & \text{\text{\textbackslash /}} / \text{Prng} \circ \text{rng} \sqcup [fn] \phi \sqcup \{pg\} \\
= & \text{\text{\textbackslash /}} / \text{Prng} \circ \text{rng} \sqcup [fn] \phi \sqcup \{pg\} \\
= & \text{\text{\textbackslash /}} / \text{Prng} \circ \text{rng} \phi'
\end{align*}
\]

Suppose \( \phi(fn) \neq \theta \), then all aliases of \( fn \) are updated, giving:

\[
\begin{align*}
\text{\text{\textbackslash /}} & / \text{Prng} \circ \text{rng} \phi' \\
= & \text{\text{\textbackslash /}} / \text{Prng} \circ \text{rng} (\exists [A] \phi \sqcup \phi_A) \\
= & \text{\text{\textbackslash /}} / \text{Prng} \circ \text{rng} \sqcup [A] \phi \sqcup (\text{Prng} \circ \text{rng} \phi_A) \\
= & \text{\text{\textbackslash /}} / \text{Prng} \circ \text{rng} \sqcup [A] \phi \sqcup (\text{Prng} \circ \text{rng} \phi_A) \\
= & \text{\text{\textbackslash /}} / \text{Prng} \circ \text{rng} \phi'
\end{align*}
\]

The proof for \( |\text{Pdom} \circ \text{rng} \phi'| \) is similar.

**Comment 5.5 (Hidden Lemmas)** There may appear to be several assumptions made in the proof, or unstated hidden lemmas. The key point is that these assumptions may be justified, if demanded. The heavy constructive machinery in VDM* is available, if a full elaboration of the proof is desired. However, for readability and ease of understanding, the proof should concentrate on the presentation of the essential details, and should be as short and concise as possible.
The following hidden lemma is used above; it is stated without proof.

\[ \text{LEMMA } 5.9 + / \circ \text{card} \ast \circ \text{rng} \circ \mathcal{P} j \circ \text{rng} \phi = |^1 / \mathcal{P} \text{rng} \circ \text{rng} \phi | \Rightarrow + / \circ \text{card} \ast \circ \text{rng} \ast \circ \mathcal{P} j \circ \text{rng} \subset [A] \phi = |^1 / \mathcal{P} \text{rng} \circ \text{rng} \subset [A] \phi | \text{ where } A = \phi^{-1}(\phi(fn)). \]

**The Get operation**

The *Get* operation is a lookup operation, which retrieves the physical page corresponding to the page name. The operation does not transform the file system, and thus no proof of invariant preservation is necessary.

\[ \text{Get}_0 : Fn \times Pn \mapsto (Fn \mapsto (Pn \mapsto Pg)) \mapsto Pg \]
\[ \text{Get}_0[fn, pn] \phi \triangleq \phi(fn)(pn) \]

\[ \text{pre}_0 \text{Get}_0 : Fn \times Pn \mapsto (Fn \mapsto (Pn \mapsto Pg)) \mapsto B \]
\[ \text{pre}_0 \text{Get}_0[fn, pn] \phi \triangleq \chi[fn] \phi \land \chi[pn] \phi(fn) \]

**The Del operation**

The *Del* operation is the inverse of the *Put* operation, the definition on Page 439 of [45] is correct if there are no aliases. The corresponding definition which ensures that all aliases are correctly updated is given by:

\[ \text{Del}_0 : Fn \times Pn \mapsto (Fn \mapsto (Pn \mapsto Pg)) \mapsto (Fn \mapsto (Pn \mapsto Pg)) \]
\[ \text{Del}_0[fn, pn] \phi \triangleq \]
\[ A \mapsto \phi^{-1} \circ \phi(fn) \]
\[ \mu \mapsto \chi[pn] \phi(fn) \]
\[ \phi_A \mapsto \lambda a : A \bullet \mu \]
\[ \mapsto \phi \uparrow \phi_A \]

The precondition ensures \( fn \) is recorded in the file system, and \( pn \) is a page name of the file.

\[ \text{pre}_0 \text{Del}_0 : Fn \times Pn \mapsto (Fn \mapsto (Pn \mapsto Pg)) \mapsto B \]
\[ \text{pre}_0 \text{Del}_0[fn, pn] \phi \triangleq \chi[fn] \phi \land \chi[pn] \phi(fn) \]

\[ \text{LEMMA } 5.10 \text{ pre}_0 \text{Del}_0[fn, pn] \phi \land \phi = \text{Del}_0[fn, pn] \phi \Rightarrow \text{Inv}_0 FS_0[\phi] \]

114
Proof
\[
\begin{align*}
&\cap / \mathcal{P} \text{rng } \cap \text{rng } \phi \\
&=\cap / \cap \mathcal{P} \text{rng } \cap \left(\phi \cup \phi_A\right) \\
&=\cap / \cap \mathcal{P} \text{rng } \cap \left[\phi \cup \left(\text{rng } \cap [pn] \phi(fn)\right)\right]
\end{align*}
\]

\[
\begin{align*}
&\cap / \text{card } \cap \text{rng } \cap \mathcal{P} \text{ rng } \phi' \\
&=\cap / \text{card } \cap \text{rng } \cap \mathcal{P} \text{ rng } \phi + |\text{rng } \cap [pn] \phi(fn)| \\
&=\cap / \text{rng } \cap \left[\phi \cup \left(\text{rng } \cap [pn] \phi(fn)\right)\right] \\
&=\cap / \text{rng } \cap \left(\phi \cup \phi_A\right) \\
&=\cap / \text{rng } \phi'
\end{align*}
\]

The proof for \(|\mathcal{P} \text{dom } \cap \text{rng } \phi'| is similar.

This completes the study of the initial model of the file system. The conclusions from the study include the following:

**Conclusion 5.1 (Invariant Complexity and Proof)** The complexity of a proof obligation of invariant preservation is directly related to the complexity of the operation and the invariant. In particular, a simpler equivalent form of the latter, in general, leads to a simpler proof of invariant preservation for the operation.

**Conclusion 5.2 (Proof Legibility)** The purpose of a proof in formal methods is to provide convincing verification of the correctness of part of a specification. Programmers are trained to program, not to be professional mathematicians or logicians. If a proof provided by a programmer, is to be peer checked, it is essential that the proof is well structured and readable. The more terse and elegant the proof, the more amenable it is to verification by peers.

**Conclusion 5.3 (Communicating Proof)** The purpose of a proof in formal methods is to provide justification of correctness. If the proof is presentable in a mixture of natural language and technical language, it is communicable to an audience of other programmers. A proof which is not communicable to peers is unlikely to gain widespread acceptability, and is suitable for dedicated specialist use only.

### 5.3 Model 1 of File System

The first level of reification in the file system is the introduction of catalogues, \(\text{CTLG}\), with one catalogue per file. This structure maintains a list of file names and the corresponding page directory for each file. The page directory records where the pages are stored, and is
composed of the user oriented page names, with the corresponding system page addresses. The \( Pgs \) structure is similar to a disk, and records the relationship between these system page addresses, and the corresponding physical pages on disk. The definition of the semantic domain in [45] is adapted from [10]. The definition is very rigid, effectively prohibiting aliases from occurring by employing one to one constraints. However, as [45] observes, the invariant is violated if two successive create operations are invoked. Furthermore, the logical implication of the invariant is that there is a maximum of one empty file in the file system at any one time, which is an unrealistic limitation.

Two isomorphic forms of the semantic domain are presented in [45]. The objective is to present an easier retrieval function, by combining the two retrieval functions \( R_{10} \circ R_{11} \). However, this approach is un-necessary as a retrieval function may be constructed directly without difficulty. The objective here is to present the model as in [45], and to evaluate its suitability as a refinement from model 0. Following this assessment, the model is amended, and the corresponding operations presented. There then follows two proof obligations for each operator, firstly that the operator should preserve the invariant for model 1, and secondly that the operation is a valid refinement of the corresponding model 0 definition.

\[
\begin{align*}
\phi \in FS_1 & = CTLG_1 \times DIRS_1 \times Pgs_1 \\
\kappa \in CTLG_1 & = Fn \mapsto Dn \quad |\text{dom } \kappa| = |\text{rng } \kappa| \\
\tau \in DIRS_1 & = Dn \mapsto DIR_1 \quad |\text{dom } \tau| = |\text{rng } \tau| \\
\varpi \in Pgs_1 & = Pa \mapsto Pg \\
\delta \in DIR_1 & = Pn \mapsto Pa \quad |\text{dom } \delta| = |\text{rng } \delta|
\end{align*}
\]

**Comment 5.6 (Aliases)** The cardinality constraints indicate one to one mappings, however this in effect prohibits file aliasing. The original file system permitted file aliasing, thus for the refinement to be adequate, it should permit aliasing also.

The invariant for \( FS_1 \) in [10] is specified as follows:

\[
\begin{align*}
Inv_{FS_1} : FS_1 & \mapsto B \\
Inv_{FS_1}[\kappa, \tau, \varpi] & \triangleq \\
\text{rng } \kappa & = \text{dom } \tau \\
\land \cup / \{ \text{rng } \delta | \delta \in \text{rng } \tau \} & = \text{dom } \varpi \\
\land (\forall pa \in \text{dom } \varpi)(\exists dn \in \text{dom } \tau)(pa \in \text{rng } \tau(dn))
\end{align*}
\]

**Comment 5.7 (Pages and Page Addresses)** The invariant stipulates that page addresses should be partitioned by the files, with no two files sharing the same page address. However, there are no constraints placed on the mapping between page addresses and physical pages, i.e., \( \varpi \). In fact, a one to one constraint is necessary.
These inadequacies are resolved by an amendment to the model to permit aliasing, and to strictly forbid any two files which are not aliases, from sharing a physical page. The amended model is defined as follows.

\[
\begin{align*}
\phi \in FS_1 & = \text{CTLG}_1 \times \text{DIRS}_1 \times Pg_{s_1} \\
k \in \text{CTLG}_1 & = F_n \mapsto D_n \quad |\text{dom } k| = |\text{rng } k| \\
\tau \in \text{DIRS}_1 & = D_n \mapsto D_R \\
\varpi \in Pg_{s_1} & = P_a \mapsto P_g \quad |\text{dom } \varpi| = |\text{rng } \varpi| \\
\delta \in \text{DIR}_1 & = P_n \mapsto P_a \quad |\text{dom } \delta| = |\text{rng } \delta|
\end{align*}
\]

The changes made to the model may appear superficial. However, the removal of the one to one constraints on \(\tau\), ensures aliasing is permitted. The placing of constraints on \(\varpi\), combined with the existing one to one constraints on \(\delta\) ensure that two files which are not aliases do not share a common physical page.

The corresponding invariant is presented as follows:

\[
\begin{align*}
\text{Inv}_{FS_1} : FS_1 & \mapsto B \\
\text{Inv}_{FS_1} \llbracket k, \tau, \varpi \rrbracket & \triangleq \\
\text{rng } k & = \text{dom } \tau \\
\land \cup / \circ \text{Prng } \circ \text{rng } \tau & = \text{dom } \varpi \\
\land \lor / \circ \text{card } ^* \circ \text{rng } ^* \circ \text{Pj } \circ \text{rng } \tau & = \cup / \circ \text{Prng } \circ \text{rng } \tau \\
\land \lor / \circ \text{card } ^* \circ \text{dom } ^* \circ \text{Pj } \circ \text{rng } \tau & = \cup / \circ \text{Pdom } \circ \text{rng } \tau
\end{align*}
\]

**Comment 5.8 (Invariant Complexity)** The invariant is more involved than the corresponding model 0 definition. This seems to be an unavoidable consequence of the increased manipulation required in the more detailed model.

**The Retrieval Function \(\mathcal{R}_{1,0}\)**

The objective of the retrieval function is to enable a return from the more concrete specification to the more abstract representation. This is to ensure that the concrete representation validly reflects the abstract, and the commuting diagram property is satisfied.

\[
\mathcal{R}_{n+1, n} \circ Op_{n+1} = Op_n \circ \mathcal{R}_{n+1, n}
\]

\[
\begin{align*}
\mathcal{R}_{1,0} : FS_1 & \mapsto FS_0 \\
\mathcal{R}_{1,0} \llbracket k, \tau, \varpi \rrbracket & \triangleq \\
(\mathcal{I} \mapsto \varpi, \circ)(\tau \circ k)
\end{align*}
\]

117
The effect of the functional \( f \mapsto g \) on the function \( h \) is to apply \( f \) to elements in the domain of the function \( h \), and to apply \( g \) to elements in the range of the function \( h \). The operation \((\circ, \circ)\) acts as \( g \) here, the effect on each \( \delta_i \in \text{rng} \tau \circ \kappa \) being \( \circ \circ \delta_i \), yielding a composite map with signature \( Pn \mapsto Pg \). The effect on each \( fn \in Fn \) by \( I \) is \( fn \). The overall effect of the operation, is a conversion of the concrete file system to \( Fn \mapsto (Pn \mapsto Pg) \), i.e., the initial model of the file system.

**The \( \text{Crea}_1 \) Operation**

\[
\text{pre}_\text{Crea}_1 : Fn \mapsto FS_1 \mapsto B
\]

\[
\text{pre}_\text{Crea}_1[fn](\kappa, \tau, \varpi) \triangleq \neg \chi[fn] \kappa
\]

\[
\text{Crea}_1 : Fn \mapsto FS_1 \mapsto FS_1
\]

\[
\text{Crea}_1[fn](\kappa, \tau, \varpi) \triangleq
\]

Let \( dn \in \mathcal{S}[\tau]Dn \) in

\[
\mapsto (\kappa \sqcup [fn \mapsto dn], \tau \sqcup [dn \mapsto \theta], \varpi)
\]

**Lemma 5.11** \( \text{pre}_\text{Crea}_1[fn] \phi \land \text{Inv}_FS_1[\phi'] \land \phi' = \text{Crea}_1[fn] \phi \Rightarrow \text{Inv}_FS_1[\phi'] \) where \( \phi \) denotes \((\kappa, \tau, \varpi)\) and \( \phi' \) denotes \((\kappa', \tau', \varpi')\).

**Proof**

There are four parts to the invariant; each part is proved separately.

1. \( \text{rng} (\kappa \sqcup [fn \mapsto dn]) \)
   \[
   = \text{rng} \kappa \cup \{dn\}
   = \text{dom} \tau \cup \{dn\}
   = \text{dom} \tau \cup \text{dom} [dn \mapsto \theta]
   = \text{dom} (\tau \sqcup [dn \mapsto \theta])
   \]

2. \( \cup / \circ \mathcal{P} \text{rng} \circ \text{rng} (\tau \sqcup [dn \mapsto \theta]) \)
   \[
   = \cup / \circ \mathcal{P} \text{rng} \circ \text{rng} \tau \sqcup \emptyset
   = \cup / \circ \mathcal{P} \text{rng} \circ \text{rng} \tau
   = \text{dom} \varpi
   \]

3. \( + / \circ \text{card}^* \circ \text{rng} \circ \mathcal{P} j \circ \text{rng} (\tau \sqcup [dn \mapsto \theta]) \)
   \[
   = + / \circ \text{card}^* \circ \text{rng} \circ \mathcal{P} j \circ \text{rng} \tau
   = |\cup / \circ \mathcal{P} \text{rng} \circ \text{rng} \tau|
   = |\cup / \circ \mathcal{P} \text{rng} \circ \text{rng} (\tau \sqcup [dn \mapsto \theta])|
   \]

4. \( + / \circ \text{card}^* \circ \text{dom}^* \circ \mathcal{P} j \circ \text{rng} (\tau \sqcup [dn \mapsto \theta]) = |\cup / \circ \mathcal{P} \text{rng} \circ \text{rng} (\tau \sqcup [dn \mapsto \theta])| \) is similar to Case 3.
Lemma 5.12 \( R_{1,0} \circ Crea_1[fn] \phi = Crea_0 \circ R_{1,0}[\phi] \).

Proof
\[
R_{1,0} \circ Crea_1[fn](\kappa, \tau, \omega)
= R_{1,0}(\kappa \sqcup [fn \mapsto dn], \tau \sqcup [dn \mapsto \theta], \omega)
= (I \mapsto \omega, \circ)((\tau \sqcup [dn \mapsto \theta] \circ (\kappa \sqcup [fn \mapsto dn]))
= (I \mapsto \omega, \circ)((\tau \circ \kappa) \sqcup ([dn \mapsto \theta] \circ [fn \mapsto dn]))
= (I \mapsto \omega, \circ)(\tau \circ \kappa) \sqcup ((I \mapsto \omega, \circ)([fn \mapsto \theta]))
= R_{1,0}[\kappa, \tau, \omega] \sqcup ([fn \mapsto \theta])
= Crea_0[fn] \circ R_{1,0}[\kappa, \tau, \omega]
\]

The \text{Eras}_1 Operation
\[
pre_{\text{Eras}_1} : Fn \mapsto FS_1 \mapsto B
pre_{\text{Eras}_1}[fn](\kappa, \tau, \omega) \triangleq \chi[fn] \kappa
\]

\text{Eras}_1 : Fn \mapsto FS_1 \mapsto FS_1
\[
Eras_1[fn](\kappa, \tau, \omega) \triangleq \\
|\tau^{-1} \circ \tau(\kappa(fn))| = 1
\mapsto (\# fn| \kappa, \# \kappa(fn)|\tau, \# \text{rng} \tau(\kappa(fn))|\omega)
\mapsto (\# fn| \kappa, \# \kappa(fn)|\tau, \omega)
\]

Lemma 5.13 \( pre_{\text{Eras}_1}[fn] \phi \land Inv_{FS_1}[\phi] \land \phi' = Eras_1[fn] \phi \Rightarrow Inv_{FS_1}[\phi'] \).

Proof

1. \text{rng} \kappa = \text{dom} \tau
   \[
   \Rightarrow \# \kappa(fn)|\text{rng} \kappa = \# \kappa(fn)|\text{dom} \tau
   \Rightarrow \text{rng} \triangleq fn| \kappa = \text{dom} \triangleq \kappa(fn)|\tau
   \text{as required.}
   \]

2. It is required to show that \( ^{\cup}\text{Prng} \circ \text{rng} \tau' = \text{dom} \circ \tau' \). There are two cases to consider; suppose \( |\tau^{-1} \circ \tau(\kappa(fn))| = 1 \), then

\[
^{\cup}\text{Prng} \circ \text{rng} \triangleq \kappa(fn)|\tau
= ^{\cup}\text{Prng} \circ \# \kappa(fn)|\tau \text{rng} \tau
= \# \text{rng} \tau(\kappa(fn))|^{\cup}\text{Prng} \circ \text{rng} \tau
\]

119
Suppose $|\tau^{-1} \circ \tau(k(f(n)))| > 1$, then

$\uparrow / \mathcal{P} \text{rng} \circ \text{rng} \preceq \llbracket k(f(n)) \rrbracket \tau$

$\Rightarrow \uparrow / \mathcal{P} \text{rng} \circ \text{rng} \tau$

as required.

3. It is required to show that $\uparrow / \circ \text{card} ^* \circ \text{rng} ^* \circ \mathcal{P} j \circ \text{rng} \tau' = \uparrow / \mathcal{P} \text{rng} \circ \text{rng} \tau'$. There are two cases; suppose $|\tau^{-1} \circ \tau(k(f(n)))| = 1$, then

$|\uparrow / \circ \mathcal{P} \text{rng} \circ \text{rng} \preceq \llbracket k(f(n)) \rrbracket \tau| = |\uparrow / \circ \mathcal{P} \text{rng} \circ \text{rng} \tau|$

Thus $|\uparrow / \mathcal{P} \text{rng} \circ \text{rng} \preceq \llbracket k(f(n)) \rrbracket \tau| = \uparrow / \circ \text{card} ^* \circ \text{rng} ^* \circ \mathcal{P} j \circ \text{rng} \preceq \llbracket k(f(n)) \rrbracket \tau$ as required.

Suppose $|\tau^{-1} \circ \tau(k(f(n)))| > 1$, then

$|\uparrow / \circ \mathcal{P} \text{rng} \circ \text{rng} \preceq \llbracket k(f(n)) \rrbracket \tau|$

$\Rightarrow |\uparrow / \circ \mathcal{P} \text{rng} \circ \text{rng} \tau|$ as required.

4. $\uparrow / \circ \text{card} ^* \circ \text{rng} ^* \circ \mathcal{P} j \circ \text{rng} (\llbracket k(f(n)) \rrbracket \tau) = \uparrow / \mathcal{P} \text{rng} \circ \text{rng} \preceq \llbracket k(f(n)) \rrbracket \tau$ is similar to Case 3.

**Lemma 5.14** $\mathcal{R}_{1,0} \circ \text{Eras}_1 [f n] \phi = \text{Eras}_0 \circ \mathcal{R}_{1,0} [\phi]$.

**Proof**

There are two cases as before, suppose $|\tau^{-1} \circ \tau(k(f(n)))| = 1$, then

$\mathcal{R}_{1,0} \circ \text{Eras}_1 [f n] \phi$
Suppose \(|\tau^{-1} \circ \tau(\kappa(fn))| > 1\), then

\[
\mathcal{R}_{10} \circ \text{Eras}_0[fn] \circ \mathcal{R}_{10}[\kappa, \tau, \varpi] = \mathcal{R}_{10}(\mathcal{I} \mapsto \varpi, \circ) \circ \mathcal{R}_{10}[\kappa, \tau, \varpi]
\]

**The Put\(_1\) Operation**

\[
\text{pre}_\text{Put}_1 : (Fn \times Pn \times Pg) \mapsto FS_1 \mapsto B
\]

\[
\chi[fn] \kappa
\]

\[
\land pg \notin \text{rng } \varpi
\]

\[
\land (pm \in \cup /\text{dom } \circ \text{rng } \tau) \Rightarrow pm \in \tau(\kappa(fn))
\]

\[
\text{Put}_1 : (Fn \times Pn \times Pg) \mapsto FS_1 \mapsto FS_1
\]

\[
\chi[pm] \tau(\kappa(fn))
\]

\[
\mapsto (\kappa, \tau, \varpi \upharpoonright [\tau(\kappa(fn))(pn) \mapsto pg])
\]

\[
\neg \chi[pm] \tau(\kappa(fn))
\]

\[
\tau(\kappa(fn)) \neq \emptyset
\]

\[
A \mapsto \tau^{-1} \circ \tau(\kappa(fn))
\]

\[
A \mapsto \{fn\}
\]

Let \(pa \in \mathcal{A}[\varpi]Pa\) in

\[
\mu \mapsto \tau(\kappa(fn)) \sqcup \{pn \mapsto pa\}
\]

\[
\tau_A \mapsto \lambda a : A \bullet \mu
\]

\[
\mapsto (\kappa, \tau \upharpoonright \tau_A, \varpi \sqcup \{pa \mapsto pg\})
\]
Lemma 5.15 \( \text{pre}_1[f_n,p_n,p_g] \wedge \text{Inv}_1[F_1[\phi] \wedge \phi = F_1[F_1[\phi] \wedge \phi] \Rightarrow \text{Inv}_1[F_1[\phi]] \).

Proof

1. It is required to show that \( \text{rng } \kappa' = \text{dom } \tau' \). There are two cases, the only case requiring proof is \( \tau' = \tau' \upharpoonright \tau_A \).

\[
\begin{align*}
\text{dom } (\tau' \upharpoonright \tau_A) \\
= \text{dom } (\phi[\tau_A \upharpoonright \tau_A]) \\
= \text{dom } (\phi[\tau_A \upharpoonright \tau_A]) \cup \text{dom } \tau_A \\
= \text{dom } \tau \\
= \text{rng } \kappa
\end{align*}
\]

2. It is required to show that \( \cup / \text{Prng } \circ \text{rng } \tau' = \text{dom } \omega' \). There are two cases, suppose \( \chi [p_n] \tau (\kappa (f_n)) \), then

\[
\begin{align*}
\cup / \text{Prng } \circ \text{rng } \tau' \\
= \cup / \text{Prng } \circ \text{rng } \tau \\
= \text{dom } \omega \\
= \text{dom } (\omega \upharpoonright [\tau (\kappa (f_n)) (p_n \mapsto p_g)]) \text{ as required.}
\end{align*}
\]

Suppose \( \tau \chi [p_n] \tau (\kappa (f_n)) \), then

\[
\begin{align*}
\cup / \text{Prng } \circ \text{rng } \tau' \\
= \cup / \text{Prng } \circ \text{rng } (\tau' \upharpoonright \tau_A) \\
= \cup / \text{Prng } \circ \text{rng } (\phi[\tau_A \upharpoonright \tau_A]) \\
= \cup / \text{Prng } \circ \text{rng } \phi [\tau_A \upharpoonright (\text{rng } \circ \text{rng } \tau_A)] \\
= \cup / \text{Prng } \circ \text{rng } \phi [\tau_A \upharpoonright \text{rng } \tau (\kappa (f_n)) \cup \{p_n \mapsto p_a\}] \\
= \cup / \text{Prng } \circ \text{rng } \phi [\tau_A \upharpoonright \text{rng } \tau (\kappa (f_n)) \cup \{p_a\}] \\
= \cup / \text{Prng } \circ \text{rng } \phi \cup \{p_a\} \\
= \text{dom } \omega \cup \{p_a\} \\
= \text{dom } (\omega \cup [p_a \mapsto p_g])
\end{align*}
\]

3. It is required to show that \( \cup / \text{card } \circ \text{rng } \circ \text{Prng } \circ \text{rng } \tau' = [\cup / \text{Prng } \circ \text{rng } \tau'] \). There are two cases, suppose \( \chi [p_n] \tau (\kappa (f_n)) \), then

122
\[+/\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \tau' =+\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \tau = |^\cup/\text{Prng} \circ \text{rng} \tau| = |^\cup/\text{Prng} \circ \text{rng} \tau'| \text{ as required.} \]

Suppose \(\neg \chi[[\text{pn}]]\tau(\kappa(fn))\), then

\[|^\cup/\text{Prng} \circ \text{rng} (\tau \uparrow \tau_A)|\]
\[= |^\cup/\text{Prng} \circ \text{rng} \tau \cup \{\text{pa}\}|\]
\[= |^\cup/\text{Prng} \circ \text{rng} \tau| + |\{\text{pa}\}|\]
\[= |^\cup/\text{Prng} \circ \text{rng} \triangleq [A[\tau \cup \text{rng} \circ \text{rng}(\kappa(fn))] + |\{\text{pa}\}|\]
\[= |^\cup/\text{Prng} \circ \text{rng} \triangleq [A[\tau] + |\text{rng} \tau(\kappa(fn))] + |\{\text{pa}\}|\]
\[=+\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \triangleq [A[\tau] + |\text{rng} \tau(\kappa(fn))] + |\{\text{pa}\}|\]
\[=+\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \triangleq [A[\tau] + |\text{rng} \tau(\kappa(fn))] \cup |\{\text{pa}\}|\]
\[=+\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \triangleq [A[\tau] + |\text{rng} \tau(\kappa(fn)) \cup \text{rng} [\text{pn} \mapsto \text{pa}]]\]
\[=+\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \triangleq [A[\tau] + |\text{rng} \tau(\kappa(fn)) \cup \text{rng} [\text{pn} \mapsto \text{pa}]]\]
\[=+\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \triangleq [A[\tau] + |\text{rng} \mu|\]
\[=+\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \triangleq [A[\tau] + |\text{rng} \lambda a : A \bullet \mu|\]
\[=+\circ \text{card}^\ast \circ \text{rng}^\ast \circ \mathcal{P}j \circ \text{rng} \tau'\]

4. \[+\circ \text{card}^\ast \circ \text{dom}^\ast \circ \mathcal{P}j \circ \text{rng} \triangleq [\kappa(fn)] \tau = |^\cup/\text{Prng} \circ \text{rng} \triangleq [\kappa(fn)\tau]|\] is similar to Case 3.

**Lemma 5.16** \(\Re_{1,0} \circ \text{Put}_1[fn, \text{pn}, pg]_\phi = \text{Put}_0 \circ \Re_{1,0}[\kappa, \tau, \varpi]\).

**Proof**

There are two cases to consider, suppose \(\chi[[\text{pn}]]\tau(\kappa(fn))\), then:

\[\Re_{1,0} \circ \text{Put}_1[fn, \text{pn}, pg]_\phi\]
\[= (\mathcal{I} \mapsto \varpi', \circ)(\tau' \circ \kappa')\]
\[= (\mathcal{I} \mapsto \varpi', \circ)(\tau \circ \kappa)\]
\[= (\mathcal{I} \mapsto \varpi \uparrow [\tau(\kappa(fn))(\text{pn} \mapsto \text{pg}), \circ)(\tau \circ \kappa)\]
\[= \lambda b : \text{dom} \kappa \bullet \varpi \uparrow [\tau(\kappa(fn))(\text{pn} \mapsto \text{pg}]) \circ (\tau \circ \kappa)(fn)\]
\[\lambda b : \text{dom} \kappa \bullet \varpi \circ \tau(\kappa(fn))\]
\[\lambda b : \text{dom} \kappa \bullet \varpi \circ \tau(\kappa(fn))\]
\[\lambda b : \text{dom} \kappa \bullet \varpi \circ \tau(\kappa(fn))\]
\[\lambda b : +[A]\text{dom} \kappa \bullet \varpi \circ \tau(\kappa(fn))\]
\[\lambda b : A \bullet (\varpi \uparrow [\tau(\kappa(fn))(\text{pn} \mapsto \text{pg}]) \circ \tau(\kappa(fn)))]\]

123
\[
\begin{align*}
\lambda b : \models [A] \dom k \bullet \varpi \circ \tau(k(fn)) \\
\lambda b : A \bullet (\varpi \uparrow [pa \mapsto pg]) \circ \tau(k(fn)))
\end{align*}
\]
\[
\begin{align*}
\lambda b : \models [A] \dom k \bullet \varpi \circ \tau(k(fn)) \\
\lambda b : A \bullet (\varpi \circ \tau(k(fn)) \uparrow [pa \mapsto pg] \circ [pn \mapsto pa])
\end{align*}
\]
\[
\begin{align*}
\lambda b : \models [A] \dom k \bullet \varpi \circ \tau(k(fn)) \\
\lambda b : A \bullet (\varpi \circ \tau(k(fn)) \uparrow [pn \mapsto pg])
\end{align*}
\]
\[
\begin{align*}
(\mathcal{I} \mapsto \varpi, o)(\tau \circ k) \uparrow \lambda a : A \bullet (\varpi \circ \tau(k(fn)) \uparrow [pn \mapsto pg])
= Put_0 \circ (\mathcal{I} \mapsto \varpi, o)(\tau \circ k)
= Put_0 \circ \mathcal{R}_{1,0}[k, \tau, \varpi]
\end{align*}
\]

Suppose \( \neg \chi[\text{fn}] \tau(k(fn)) \), then:

\[
\begin{align*}
\mathcal{R}_{1,0} \circ Put_1[fn, pm, pg] \phi \\
= (\mathcal{I} \mapsto \varpi', o)(\tau' \circ k') \\
= (\mathcal{I} \mapsto \varpi', o)(\tau \uparrow \tau_A \circ k) \\
= (\mathcal{I} \mapsto \varpi \sqcup [pa \mapsto pg], o)(\tau \uparrow \tau_A \circ k) \\
= \lambda b : \dom k \bullet (\varpi \sqcup [pa \mapsto pg], o)(\tau \uparrow \tau_A \circ k) \\
\end{align*}
\]
\[
\begin{align*}
\lambda b : \models [A] \dom k \bullet (\varpi \sqcup [pa \mapsto pg], o)(\tau \uparrow \tau_A(k(b))) \\
\lambda b : A \bullet (\varpi \sqcup [pa \mapsto pg], o)(\tau_A(k(b))) \\
\lambda b : \models [A] \dom k \bullet \varpi \circ \tau(k(b)) \\
\lambda b : A \bullet (\varpi \sqcup [pa \mapsto pg], o)(\tau(k(b)) \sqcup [pn \mapsto pa]) \\
= (\mathcal{I} \mapsto \varpi, o)(\tau \circ k) \uparrow \lambda a : A \bullet (\varpi \sqcup [pa \mapsto pg], o)(\tau(k(fn)) \sqcup [pn \mapsto pa]) \\
= (\mathcal{I} \mapsto \varpi, o)(\tau \circ k) \uparrow \lambda a : A \bullet (\varpi \circ \tau(k(fn)) \sqcup [pa \mapsto pg], o)(\tau(k(fn)) \sqcup [pn \mapsto pa]) \\
= (\mathcal{I} \mapsto \varpi, o)(\tau \circ k) \uparrow \lambda a : A \bullet (\varpi \circ \tau(k(fn)) \sqcup [pn \mapsto pg]) \\
= Put_0 \circ (\mathcal{I} \mapsto \varpi, o)(\tau \circ k) \\
= Put_0 \circ \mathcal{R}_{1,0}[k, \tau, \varpi]
\end{align*}
\]

Comment 5.9 The proof is tedious, and contains some hidden lemmas. However, the structure of the proof is reasonably clear.

The \( \text{Del}_1 \) Operation

\[
\begin{align*}
\text{pre}_{\text{Del}_1} : (Fn \times Pn) \mapsto FS_1 \mapsto B \\
\text{pre}_{\text{Del}_1}[fn, pm](k, \tau, \varpi) \triangleq \\
\chi[fn]k \wedge \chi[pm]\tau(k(fn))
\end{align*}
\]
\[
\begin{align*}
\text{Del}_1 : (Fn \times Pn) \mapsto FS_1 \mapsto FS_1 \\
\text{Del}_1[fn, pm](k, \tau, \varpi) \triangleq \\
A \mapsto \tau^{-1} \circ \tau(k(fn))
\end{align*}
\]

124
\[
\mu \triangleright \{pn\} \tau(\kappa(fn)) \\
\tau_A \mapsto \lambda a : A \cdot \mu \\
\mapsto (\kappa, \tau \uparrow \tau_A, \triangleleft [\tau(\kappa(fn))(pn)]\omega)
\]

**Lemma 5.17** \(\text{pre}_D e l_1[fn, pn] \land Inv_{FS_1}[\phi] \land \phi = Del_1[fn, pn] \phi \Rightarrow Inv_{FS_1} [\phi].\)

**Proof**

1. \(\text{dom} (\tau \uparrow \tau_A)\)
   
   \[\text{dom} (\triangleleft [\tau_A] \tau \cup \tau_A)\]
   
   \[= \text{dom} (\triangleleft [\tau_A] \tau) \cup \text{dom} \tau_A\]
   
   \[= \text{dom} \tau\]
   
   \[= \text{rng} \kappa\]

2. It is required to show that \(\uparrow / \Prng \circ \text{rng} \tau' = \text{dom} \omega'.\)
   
   \[\uparrow / \Prng \circ \text{rng} (\tau \uparrow \tau_A)\]
   
   \[= \uparrow / \Prng \circ (\triangleleft [\tau(\kappa(fn))] \text{rng} \tau) \cup \triangleleft [pn] \tau(\kappa(fn))\]
   
   \[= \triangleleft [\text{rng} \tau(\kappa(fn))] \uparrow / \Prng \circ \text{rng} \tau \cup \text{rng} \triangleleft [pn] \tau(\kappa(fn))\]
   
   \[= \triangleleft [\text{rng} \tau(\kappa(fn))] \uparrow / \Prng \circ \text{rng} \tau \cup \triangleleft [\tau(\kappa(fn))(pn)] \text{rng} \tau(\kappa(fn))\]
   
   \[= \triangleleft [\tau(\kappa(fn))(pn)] \uparrow / \Prng \circ \text{rng} \tau\]
   
   as required.

3. It is required to show that \(\uparrow / \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P}_j \circ \text{rng} \tau' = \uparrow / \Prng \circ \text{rng} \tau'.\)
   
   \[\mid \uparrow / \circ \Prng \circ \text{rng} \triangleleft [A] \tau \mid + \text{rng} \tau(\kappa(fn))\]
   
   \[= \mid \uparrow / \circ \Prng \circ \text{rng} \triangleleft [A] \tau \cup \text{rng} \tau(\kappa(fn))\]
   
   \[= \mid \uparrow / \circ \Prng \circ \text{rng} \triangleleft [A] \tau \cup \text{rng} \circ \text{rng} \lambda a : A \cdot \tau(\kappa(fn))\]
   
   \[= \mid \uparrow / \Prng \circ \text{rng} \tau\]

   \[= \uparrow / \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P}_j \circ \text{rng} \tau\]
   
   \[= \uparrow / \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P}_j \circ \text{rng} \triangleleft [A] \tau \mid + \text{rng} \tau(\kappa(fn))\]

   Thus \(\uparrow / \circ \text{card}^* \circ \text{rng}^* \circ \mathcal{P}_j \circ \text{rng} \triangleleft [A] \tau = \mid \uparrow / \circ \Prng \circ \text{rng} \triangleleft [A] \tau\).

\[\uparrow / \Prng \circ \text{rng} (\tau \uparrow \tau_A)\]

\[= \triangleleft [\text{rng} \tau(\kappa(fn))] \uparrow / \Prng \circ \text{rng} \tau \cup \triangleleft [\tau(\kappa(fn))(pn)] \text{rng} \tau(\kappa(fn))\]

125
\[
\begin{align*}
\text{Lemma 5.18 } & \quad \mathcal{R}_1 \circ \text{Del}_1[f(n), pn] \phi = \text{Del}_0[f(n), pn] \circ \mathcal{R}_1[\phi].
\end{align*}
\]

\textbf{Proof}

\[
\begin{align*}
\mathcal{R}_1 \circ \text{Del}_1[f(n), pn] \phi &= \mathcal{R}_1(f(n), \tau \uparrow \tau_A, \|\tau(f(n))(pn)\|) \\
&= (\mathcal{I} \mapsto \|\tau(f(n))(pn)\|, o)(\tau \uparrow \tau_A \circ \kappa) \\
&= (\mathcal{I} \mapsto \|\tau(f(n))(pn)\|, o)(\tau \uparrow \tau_A \circ \kappa) \\
&= (\mathcal{I} \mapsto \tau, o)(\tau \circ \kappa \uparrow \tau_A \circ \kappa_A) \\
&= (\mathcal{I} \mapsto \tau, o)(\tau \circ \kappa \uparrow \tau_A \circ \kappa_A) \\
&= \mathcal{R}_1[f(n), pn] \circ \mathcal{R}_1[\phi] \\
&= \text{Del}_0[f(n), pn] \circ \mathcal{R}_1[\phi].
\end{align*}
\]

There are several hidden lemmas in the proof, for clarity purposes they are stated and proved below, rather than obscuring the structure of the proof.

\textbf{Lemma 5.19 } (\tau \uparrow \tau_A \circ \kappa = (\tau \circ \kappa) \uparrow (\tau_A \circ \kappa_A) \text{ where } \text{rng} \kappa = \text{dom} \tau \text{ and } \text{dom} \tau_A \subseteq \text{dom} \tau, \text{ and } \kappa_A = \text{rng} \tau_A.

\textbf{Proof}

Suppose \( x \in \kappa \) such that \( \kappa(x) \in \tau, \kappa(x) \in \tau_A \), then \( \tau \uparrow \tau_A \circ \kappa(x) = \tau_A(\kappa(x)) = \tau(\kappa(x)) \uparrow \tau_A(\kappa(x)) \). The other case is similar.

\textbf{Lemma 5.20 } (\mathcal{I} \mapsto \tau, o)((\tau \uparrow \tau_A \circ \kappa)(f) = (\mathcal{I} \mapsto \tau, o)((\tau \uparrow \tau_A \circ \kappa)(f).

\textbf{Proof (Informal)}

Suppose \( \text{pa} = \tau(f(n))(pn) \), then
\[\cup \circ \mathcal{P}_\text{rng} \circ \text{rng} \tau = \text{dom} \varpi\]
\[\cup \circ \mathcal{P}_\text{rng} \circ \text{rng} (\tau \uparrow \tau_A) = \text{dom} \varpi' = 4 \| \text{pa} \| \text{dom} \varpi.\]

Thus \(\varpi'\) and \((\tau \uparrow \tau_A) \circ \kappa)(f)\) are composable, for any \(f \in \text{Fn}\). Clearly \(\text{pa} \notin \text{dom} \varpi'\), and thus \(\varpi' \cup [\text{pa} \mapsto \varpi(\text{pa})]\) and \((\tau \uparrow \tau_A) \circ \kappa)(f)\) are composable. Since \(\text{pa} \notin \text{rng}((\tau \uparrow \tau_A) \circ \kappa)(f)\), it follows:

\[\varpi \circ ((\tau \uparrow \tau_A) \circ \kappa)(f) = \varpi' \circ ((\tau \uparrow \tau_A) \circ \kappa)(f)\]

Since this is true for every \(f \in \text{dom} \kappa\), the result follows.

**Lemma 5.21** \((f \mapsto g)(\mu_1 \uparrow \mu_2) = (f \mapsto g)\mu_1 \uparrow (f \mapsto g)\mu_2\)

**Proof**

Let \(x \in \mu_1 \uparrow \mu_2\), suppose \(x \in \mu_1\) and \(x \in \mu_2\), then

\[\begin{align*}
(f \mapsto g)(\mu_1 \uparrow \mu_2)(x) &= \begin{cases} 
[f(x) \mapsto g(\mu_2(x))] & \text{if } x \in \mu_1 \\
[f(x) \mapsto g(\mu_1(x))] \uparrow [f(x) \mapsto g(\mu_2(x))] & \text{if } x \in \mu_2
\end{cases} \\
&= ((f \mapsto g)\mu_1 \uparrow (f \mapsto g)\mu_2)(x)
\end{align*}\]

The other cases are similar.

**Lemma 5.22** \(\phi_A = (I \mapsto \varpi, \circ)(\tau_A \circ \kappa_A)\)

**Proof**

\(A\) is the set of aliases of \(\text{fn}\) in model one of the file system. Let \(a \in A\), then \(\tau(\kappa(a)) = \tau(\kappa(\text{fn}))\), and since \(\varpi\) is one to one \(\varpi \circ \tau(\kappa(a)) = \varpi \circ \tau(\kappa(\text{fn}))\). Thus \(A \subseteq \text{dom} \phi_A\). Reversing the argument gives \(\text{dom} \phi_A \subseteq A\), and thus \(\text{dom} \phi_A = A\). Thus \(\phi_A = (I \mapsto \varpi, \circ)(\tau_A \circ \kappa_A)\).

**The Get\(_1\) Operation**

\[\text{pre}_{\text{Get}_1} : (\text{Fn} \times \text{Pn}) \mapsto \text{FS}_1 \mapsto \text{B}\]

\[\begin{align*}
\chi[\text{fn}] &\land \chi[\text{pn}] \tau(\kappa(\text{fn})) \\
\end{align*}\]

\[\begin{align*}
\text{Get}_1 : (\text{Fn} \times \text{Pn}) \mapsto \text{FS}_1 \mapsto \text{Pg}\]

\[\begin{align*}
\varpi(\tau(\kappa(\text{fn}))(\text{pn})) \\
\end{align*}\]

There is no necessity to prove that the invariant is preserved, as this is a lookup operation, and does not transform the state. The only proof required, is that the operation is a valid refinement of \(\text{Get}_0\).
Lemma 5.23 $\mathcal{R}_{1,0} \circ \text{Get}_1[f_n, pn][\phi] = \text{Get}_0 \circ \mathcal{R}_{1,0}[\phi]$.

Proof
$\mathcal{R}_{1,0} \circ \text{Get}_1[f_n, pn](\kappa, \tau, \varpi)$
$= \mathcal{R}_{1,0}\varpi(\tau(\kappa(f_n))(pn))$
$= \varpi(\tau(\kappa(f_n))(pn))$
$= [f_n \mapsto \varpi(\tau(\kappa(f_n))))(f_n)(pn)$
$= ((\mathcal{I} \mapsto \varpi, \circ)(\tau \circ \kappa))(f_n)(pn)$
$= \text{Get}_0[f_n, pn][\mathcal{I} \mapsto \varpi, \circ)(\tau \circ \kappa)$
$= \text{Get}_0[f_n, pn] \circ \mathcal{R}_{1,0}[\kappa, \tau, \varpi]$

5.4 Summary

The work involved in the study of the file system presents a certain view on the role and applicability of formal methods in software engineering. In particular, the step wise refinement approach of methodologies such as VDM is unlikely to gain widespread appeal unless significant progress is made in proof automation. The model of the file system demonstrates the complexity of providing proofs by hand.

Consequently, given the explosion of proof obligations for invariant satisfaction and reification validity it is essential that usable tools are available in order for formal methods to gain widespread appeal in the Computer Science community. Furthermore, there is an additional difficulty with step wise refinement in that it effectively requires the programmer to rewrite the same operation ad nauseum.

The niche area for formal methods is requirements analysis. An initial model of some system is proposed. This model is a representation of the proposed system, independent of a particular implementation. This initial model may then be thoroughly explored to determine its properties, and thus its suitability. The model is a mathematical object and is unambiguous, thus is a clear representation of the requirements of the proposed system. The model then serves as an unambiguous reference point, which the future implemented system (developed formally or informally) must satisfy.
Chapter 6

The Model of Music

6.1 Introduction

Chapter 2 has demonstrated that formal methods may be employed to model the organizational structure of religion. It was concluded that the religion model is generic and may be adapted to model the hierarchical organizational structure of companies, political institutions, or a university. This justified a detailed examination of the bill of material structure, i.e., an abstraction of hierarchies. The fact that the religion model encodes or catalogues information suggested an examination of catalogues from the classical model of a file system.

The objective of the final chapter of this thesis is to present further examples of the applications of formal methods to model aspects of the real world. This chapter develops a model of aspects of music and musical compositions using the notation of the Irish school of VDM. This involves modelling the structure of a musical composition. Since musical compositions are generally intended to be performed by an orchestra, the model considers the problem of orchestras and musical instruments.

The standard approach to the specification of a musical compositions is to employ the five line stave symbol system. Musical notes are then written on either the treble or bass clef in a particular key signature. Each note has a specific pitch and time duration. The speed of the composition (or tempo) is given by the metronome setting. However, mood changes within a composition may have a corresponding effect on the tempo; for example, a switch from adagio to allegro. The conductor implements the metronome setting via the speed at which the baton is waved. Consequently, the standard specification of musical compositions is directly executable by an orchestra. In view of the fact that the standard approach to musical specification is unambiguous, the question which immediately arises is what is the added advantage of presenting an alternate approach to musical specification?

The main justification for developing this alternate specification style for musical composition is to demonstrate the applicability of formal methods to modelling aspects of
music. Secondly, this style is closer to that of electronic musical composition, where [38] musical composition is available to all including those with minimal musical education. The key question is whether this particular musical specification style is adequate. The standard specification style does not adequately reflect the musical massage which sometimes takes place in recording studios to achieve special effects, (cf., Lennon's 'Strawberry Fields Forever', Page 67 of [38]).

The initial model of music considers a composition to be a sequence of notes, where each note lasts for a specified time period. The second model of music considers a composition to be a collection of multi-notes. The final model of a musical composition considers a composition to be made up of individual instruments, where each instrument plays a sequence of notes, where each note is for a specified time duration.

Finally, the study of the domain of music highlights the need for temporal operators, either discrete or continuous to express various musical properties. This involves detailing temporal operators for discrete or continuous time intervals.

6.2 Basic Model of Music

The objective of this section is to present a simple model of musical composition. This initial model assumes that a musical composition is simply a sequence of notes, where a single note is played at any moment of time. This model does not consider issues such as harmony, chords, movements, temporal aspects, bars, etc. The occurrence of $f$ in the key of $G$ is represented as the pitch $f^\#$, the key signature is ignored. The initial model assumes that an instrument is capable of playing exactly one note at a time. This is a limitation of the model as several instruments including harpsichords, classical guitars, pianos and organs may play several notes at once.

The model assumes that a musical composition consists of a sequence of notes, where each note is for a specific time period and has an associated pitch and volume. Each composition has a composer, a metronome setting and is in a particular time. Rhythm is modelled as the physical time that a particular note is to be played. The metre or time a composition is in and the metronome setting are recorded in the model, but are only used to determine the number of bars in the composition, assuming a constant metronome setting.

The effect of a metronome is to translate logical time, as exhibited by crotchets, quavers, etc., into the standard specification style of music, to actual physical time duration.

The following details the basic mathematical structures used in the model of musical compositions. The notation $\mathbb{R}^+$ and $\mathbb{Q}^+$ refer to the positive reals and positive rationals respectively. The notation is not part of the Irish school of VDM.

\[ \text{Note} = \mathbb{R}_0 \]  

(6.1)

130
\[ \text{Len} = \mathbb{R}^+ \]  \hspace{2cm} (6.2)
\[ \text{Vol} = \mathbb{R}_0 \]  \hspace{2cm} (6.3)
\[ \text{Met} = \mathbb{N}^+ \]  \hspace{2cm} (6.4)
\[ \text{Time} = \mathbb{Q}^+ \]  \hspace{2cm} (6.5)

A composition is created by exactly one composer. Composition names are not unique, for example, there are two distinct compositions termed ‘Ave Maria’. The basic model of music considers a composition to be a work for a single instrument. Consequently, the composition is a sequence of simple notes, i.e., there are no chords, and each note has a physical time duration. The basic model is defined as follows:

\[
\begin{align*}
\alpha : & \text{Cmp}_Jd \mapsto \text{Cp}_Jd \times \text{Met} \times \text{Time} \\
\beta : & \text{Cmp}_Jd \mapsto (\text{Note} \times \text{Vol} \times \text{Len})^* 
\end{align*}
\]

\[ \text{Inv}_\text{Mus} : (\text{Cmp}_Jd \mapsto \text{Cp}_Jd \times \text{Met} \times \text{Time}) \times (\text{Cmp}_Jd \mapsto (\text{Note} \times \text{Vol} \times \text{Len})^*) \mapsto \text{B} \]

\[ \text{Inv}_\text{Mus}[\alpha, \beta] \triangleq \]
\[ \text{dom} \alpha = \text{dom} \beta \\
\forall c \in \beta \\
\beta(c) \neq \Lambda \\
\forall c \in \beta \\
\text{elems} \circ \pi^* \beta(c) \neq \{0\} \]

The invariant ensures that every composition is non-trivial, i.e., it ensures that a composition consists of at least one note which is distinct from a rest note. A rest note is assumed to be a note which has a zero volume. The model places no restrictions on composition length. For example, in this model it is valid for a composition to last for an infinitesimally period or time or extend for several years.

A composer is expected to produce an original composition. If two composers write the same musical composition then a form of plagiarism is suspected. Of course, plagiarism may be suspected if two musical compositions are very similar, without being identical. Plagiarism may be prohibited by adding an an extra constraint to the invariant to forbid duplicate compositions by different composers.

### 6.2.1 Elementary properties of Model

This section determines the adequacy of the basic model of music by model exploration and model interrogation. The adequacy of the model is then judged by its ability to encode basic properties of music. The following model evaluation questions are considered.
Model Evaluation Questions

1. Who are the living/dead composers?
2. Who is the composer of composition c?
3. What is the total length of composition c?
4. What is the logical length of composition c?
5. How many bars of music are there in composition c?

**Question 6.1** *Who are the living/dead composers?*

The composers are given by $P\pi_1 \circ \text{rng} \alpha$

Many compositions, for example, the folk songs of many countries in which the music has been preserved via the oral tradition, have no known composers. The composer-id in this case may be considered to be a special composer-id representing an unknown composer.

**Question 6.2** *Who is the composer of composition c?*

This is given by $\pi_1 \alpha(c)$ subject to $\chi[c] \alpha$

**Question 6.3** *What is the total length of composition c?*

This is given by the $\text{Len}_Cmp$ operation which is defined below. The exact definition of this operation is understood by noting that $\beta(c) = \tau = \langle (n_1, v_1, l_1), (n_2, v_2, l_2), ..., (n_k, v_k, l_k) \rangle$. Consequently, by applying the sequence function operation $\pi_3^*$ to $\tau$ we get:

$$\pi_3^* \tau = \langle l_1, l_2, ..., l_k \rangle$$

Thus $\text{Len}_Cmp[\beta(c)] \triangleq / \circ \pi_3^* \beta(c)$ gives the length of composition c. Thus $\text{Len}_Cmp[c] \beta$ gives the physical time that elapses from the start of the composition to the end of the composition.

$\text{Len}_Cmp : (\text{Note} \times \text{Vol} \times \text{Len})^* \rightarrow \mathbb{R}^+$

$\text{Len}_Cmp[\beta(c)] \triangleq / \circ \pi_3^* \beta(c)$

subject to the precondition $\chi[c] \beta$

**Question 6.4** *What is the logical length of composition c?*

The logical length of a composition is its length expressed in terms of a number of crotchets. The metronome setting combined with the physical time of the composition enables the equivalent number of crotchets of the composition to be derived. The metronome
setting indicates the number of crotchets to be played in one minute of time, for example, 96 in a minute. It is assumed that the physical length of each note is expressed in terms of minutes. The logical length of the composition is calculated as follows:

\[ \text{Len}_Cmp[\beta(c)] \] gives the total composition duration in terms of minutes. Furthermore, \( \pi_2 \circ \alpha(c) \) gives the metronome setting, i.e., the number of crotchets per minute. Then the \( \text{Log}_\text{Len} \) operation is defined as follows:

\[
\text{Log}_\text{Len} : \text{Cmp}_\text{Id} \mapsto ((\text{Cmp}_\text{Id} \mapsto (\text{Cp}_\text{Id} \times \text{Met} \times \text{Time}) \times (\text{Cmp}_\text{Id} \mapsto (\text{Note} \times \text{Vol} \times \text{Len})^*)) \mapsto \mathbb{R}^+ \\
\text{Log}_\text{Len}[c](\alpha, \beta) \triangleq \text{Len}_\text{Cmp}[\beta(c)] \ast \pi_2 \circ \alpha(c)
\]

The precondition is \( \chi[c] \alpha \).

**Question 6.5** How many bars of music are there in composition \( c \)?

The number of bars in composition \( c \) is calculated from the logical length of the composition as given by \( \text{Log}_\text{Len}[c](\alpha, \beta) \). The particular time the composition is in enables the equivalent number of bars to be calculated. It is assumed that the metronome setting remains constant throughout the composition as otherwise the \( \text{Len}_\text{Cmp}[\beta(c)] \) operation is invalid.

Consider a piece of music with logical length 15 crotchets in \( \frac{6}{8} \) time. This is equivalent to 5 bars of music and is obtained by converting \( \frac{6}{8} \) to \( \frac{3}{4} \) and dividing the 15 crotchets by 3, as there is the equivalence of three crotchets per bar. Similarly, if the composition is in \( \frac{3}{8} \) time this is equivalent to \( \frac{3}{4} \), i.e., \( \frac{3}{2} \) crotchets per bar and in this case it represents 10 bars of music.

In general, if a piece is in \( \frac{n}{m} \) time it is converted to its equivalent number of crotchets per bar, i.e., \( \frac{n}{4} = \frac{2}{4} = \frac{n}{4} \) and so \( x = 4 \ast \frac{n}{4} \). The number of bars is then calculated by \( \text{Log}_\text{Len}[c](\alpha, \beta)/x \). This operation is expressed as follows:

\[
\text{Log}_\text{Bar}[c]\alpha \triangleq 4 \ast \pi_3 \circ \alpha(c)
\]

\[
\text{Nmr}_\text{Bars}[c]\alpha \triangleq \frac{\text{Log}_\text{Len}[c](\alpha, \beta)}{\text{Log}_\text{Bar}[c]\alpha}
\]

These operations are subject to \( \chi[c] \alpha \) and \( \chi[c] \beta \).

### 6.2.2 Operations on the Model

This section presents some elementary operations for the addition, update, or removal of compositions from the model. These are given by the \textit{Add/Upd/Rem} operations.
\[ \text{Add}\_\text{Cmp} : \text{Cmp}\_I d \times \text{Cp}\_I d \times \text{Met} \times \text{Time} \times (\text{Note} \times \text{Vol} \times \text{Len})^* \mapsto ((\text{Cmp}\_I d \mapsto (\text{Cp}\_I d \times \text{Met} \times \text{Time})) \times (\text{Cmp}\_I d \mapsto (\text{Note} \times \text{Vol} \times \text{Len})^*)) \]

\[ \text{Add}\_\text{Cmp}[c, c_p, m, T, \xi](\alpha, \beta) \triangleq (\alpha \cup [c \mapsto (c_p, m, T)], \beta \cup [c \mapsto \xi]) \]

\[ \text{pre}\_\text{Add}\_\text{Cmp}[c, c_p, m, T, \xi](\alpha, \beta) \triangleq \]
\[ \neg \chi[c] \alpha \land \neg \chi[c] \beta \]
\[ \land \xi \neq \Lambda \]
\[ \land \text{elems} \circ \pi_2^* \xi \neq \{0\} \]

This operation has the following proof obligation to ensure that the invariant is preserved; no proof of the lemma is required.

**Lemma 6.1** \( \text{pre}\_\text{Add}\_\text{Cmp}[c, c_p, m, T, \xi](\alpha, \beta) \land (\alpha', \beta') = \text{Add}\_\text{Cmp}[c, c_p, m, T, \xi](\alpha, \beta) \Rightarrow \text{Inv}\_\text{Mus}[\alpha', \beta'] \)

\[ \text{Rem}\_\text{Cmp}[c](\alpha, \beta) \triangleq (\forall \mathcal{C} \alpha \land \forall \mathcal{C} \beta) \]

The proof obligation for the \( \text{Rem}\_\text{Cmp}[c] \) is stated in the following lemma. The result is clearly true, and no proof is provided.

**Lemma 6.2** \( \text{pre}\_\text{Rem}\_\text{Cmp}[c](\alpha, \beta) \land (\alpha', \beta') = \text{Rem}\_\text{Cmp}[c](\alpha, \beta) \Rightarrow \text{Inv}\_\text{Mus}[\alpha', \beta'] \)

Finally, the \( \text{Upd}\_\text{Cmp} \) operation, an operation which allows a composer to perform updates to the composition is presented. It is very similar to the \( \text{Add}\_\text{Cmp}[c] \) operation.

\[ \text{Upd}\_\text{Cmplb}, m, T, \xi](\alpha, \beta) \triangleq (\alpha \uparrow [c \mapsto (\pi_1 \circ \alpha(c), m, T)], \beta \uparrow [c \mapsto \xi]) \]

\[ \text{pre}\_\text{Upd}\_\text{Cmp}[c, m, T, \xi](\alpha, \beta) \triangleq \]
\[ \chi[c] \alpha \land \chi[c] \beta \]
\[ \land \xi \neq \Lambda \]
\[ \land \text{elems} \circ \pi_2^* \xi \neq \{0\} \]

The proof obligation is stated in the following lemma.

**Lemma 6.3** \( \text{pre}\_\text{Upd}\_\text{Cmp}[c, m, T, \xi](\alpha, \beta) \land (\alpha', \beta') = \text{Upd}\_\text{Cmp}[c, m, T, \xi](\alpha, \beta) \Rightarrow \text{Inv}\_\text{Mus}[\alpha', \beta'] \)

134
6.2.3 Real Time and the Basic Model

The basic model allows a performance to be made by following the sequence of notes, where each note has an individual pitch, volume and time duration. However, a digitalized recording may include the requirement to examine an individual note at a random part of the recording. This involves converting the model to a representation in the form:

$$
\zeta: [0, L] \mapsto (Note \times Vol)
$$

(6.8)

where \( L \) represents \( \text{Len} \cdot \text{Cmp} \cdot [\beta(c)] \). The required form is described in pictorial form as follows:

```
Pitch

|
|
|   ******
| ****  ***
|    ***  ******
| ****      ******
|   ****
|      ******
|        ****
|           **

----------------------------------------------------------
/                        Time
/                        /

Volume

Melody :-  *****
```

This section presents an operation to convert the sequence representation of a composition to a representation of the form \([0, L] \mapsto (Note \times Vol)\). The approach taken is to represent a note \( n \) at volume \( v \) for length \( t \) commencing at time \( T \) as the half open, half closed interval \([T, t + T)\). This means note \( n \) commences at time \( T \) and terminates an infinitesimally small time before \( T + t \).

It should be noted that this representation of time is inherently non-constructive and is effectively a deviation from the Irish school of VDM. The final part of this chapter considers constructive time. The \( Trn_X \# \) operation is defined as follows:
\[ T_{\text{Trn} \cdot \text{Xlt}} : (\text{Note} \times \text{Vol} \times \text{Len})^* \mapsto \mathbb{R} \mapsto (\text{Note} \times \text{Vol}) \]
\[ T_{\text{Trn} \cdot \text{Xlt}}[\xi] \triangleq T_{\text{Trn} \cdot \text{Xlt}}[\xi] \theta \]

\[ T_{\text{Trn} \cdot \text{Xlt}} : (\text{Note} \times \text{Vol} \times \text{Len})^* \mapsto (\mathbb{R} \mapsto (\text{Note} \times \text{Vol})) \mapsto (\mathbb{R} \mapsto (\text{Note} \times \text{Vol})) \]
\[ T_{\text{Trn} \cdot \text{Xlt}}[\Lambda][\zeta] \triangleq \zeta \]
\[ T_{\text{Trn} \cdot \text{Xlt}}[\sigma] \sim \tau] \zeta \triangleq \]

Let \( p = \pi_1 \sigma, v = \pi_2 \sigma, t = \pi_3 \sigma \) in

Let \( T = \sup \circ \text{dom} \zeta \quad \zeta \neq \emptyset \) in

Let \( \zeta_{(T; T+t)} = \lambda r : [T; T + t) \cdot (p, v) \) in

\[ T_{\text{Trn} \cdot \text{Xlt}}[\tau](\zeta \sqcup \zeta_{(T; T+t)}) \]

The notation \( \sup \) indicates the supremum.

**COMMENT 6.1** The operation is non-constructible. However, it expresses the behaviour of time in music in an adequate manner. Constructive time for music is considered later in this chapter.

### 6.3 Multi-Note Model

The basic model is adequate at representing instruments such as the flute, oboe, etc., i.e., instruments which play exactly one note at a time. However, other instruments, for example, the piano or classical guitar may play several notes at once. It is clear that such instruments are not represented in the model. The basic model is limited to compositions which consist of exactly one note played at each instant of time. Furthermore, the basic model does not consider harmony.

The approach taken to resolve this limitation is to consider an alternate representation of a composition. The basic model considers a composition to be a sequence of notes. The model in this section considers a composition to be a sequence of sequences of notes. Each sequence represents an individual counterpoint melody, and each counterpoint melody (including rest notes) is of the same length. This, in effect, allows several notes to be played at once. Consequently, the invariant must ensure that each sequence (even if it is composed of many rest notes) is exactly the same length.

\[ \alpha : \text{Cmp} \cdot \text{Id} \mapsto \text{Cp} \cdot \text{Id} \times \text{Met} \times \text{Time} \quad (6.9) \]
\[ \beta : \text{Cmp} \cdot \text{Id} \mapsto ((\text{Note} \times \text{Vol} \times \text{Len})^*)^* \quad (6.10) \]

The definition of the model is similar to that of the basic model defined in Section 6.2. The only difference is a sequence of sequences for the multi-note model.
The behaviour of the model is described in pictorial form above. Three counterpoint melodies are played by a single instrument. For example, a classical guitar has six strings, and each string may play various notes. Consequently, a classical guitar may play a maximum of six notes at any one time. This is represented by a sequence of parallel counterpoint melodies.

The invariant for the multi-note model is more involved. The fact that every every musical sequence in the sequence of sequences has the same physical time duration must be stipulated.

\[ \text{Inv}_C\mu\text{us} : (CmpId \mapsto CmpId \times Met \times Time) \times (CmpId \mapsto ((\mu Note \times Vol \times Len)^*)^*) \mapsto B \]

\[ \text{Inv}_C\mu\text{us}[\alpha, \beta] \triangleq \]

\[ \text{dom } \alpha = \text{dom } \beta \]

\[ \forall c \in \beta \]

\[ \beta(c) \neq \Lambda \]

\[ \neg \chi[[\Lambda] \beta(c)] \]

\[ \forall c \in \beta \]

\[ \forall \xi_1, \xi_2 \in \text{elems } \beta(c) \]
\[ \text{Len}_\text{Cmp}[\xi_1] = \text{Len}_\text{Cmp}[\xi_2] \]
\[ \forall c \in \beta \forall \xi \in \text{elems } \beta(c) \]
\[ \text{elems } \circ \pi^*_\xi \neq \{0\} \]

The invariant ensures that every composition consists of at least one counterpoint melody. Secondly, each counterpoint melody has at least one non-rest note. If there are multiple parallel counterpoint melodies then each one must be of the same length.

The multi-note model may not be considered a direct refinement of the basic model. The problem is that the original composition may not be retrieved from the multi-note model unless additional constraints on the multi-note model. The constraint which is employed to achieve a successful refinement is to assume that one particular counterpoint melody refers to the main melody which is employed in the basic model. Consequently, it is assumed that the head of the multiple counterpoint sequence refers to the main melody.

The retrieval function \( R_{1,0} \) applies to \( \beta_1 \) to yield \( \beta_0 \) as \( \alpha \) is identical in both models. Consequently, \( R_{1,0} \) is of the following form:

\[ R_{1,0}(\alpha_1, \beta_1) = (\alpha_1, (\mathcal{I} \mapsto \text{hd}) \beta) \]

For any operations \( O_{p_1} \) which is defined in the multi-note model 1 there is a proof obligation to demonstrate that concrete operation is a valid refinement of the abstract operation.

\[ R_{1,0} \circ O_{p_1} = O_{p_0} \circ R_{1,0} \]

The Add\_Cmp\_1 operation is exactly verbatim to the Add\_Cmp\_0 operation. However, the precondition is different. The Add\_C\_p\_t operations adds a counterpoint melody to the composition. This operation is specific to the multi-note model.

\[ \text{Add\_Cmp}_1[c, c_p, m, T, \xi](\alpha, \beta) \triangleq (\alpha \sqcup [c \mapsto (c_p, m, T)], \beta \sqcup [c \mapsto \xi]) \]

\[ \text{pre}_{\text{Add\_Cmp}_1}[c, c_p, m, T, \xi](\alpha, \beta) \triangleq \]
\[ \neg \chi[c] \alpha \land \neg \chi[c] \beta \]
\[ \land \xi \land \neg \chi[L] \xi \]
\[ \forall \xi_1, \xi_2 \in \xi \]
\[ \text{Len\_Cmp}[\xi_1] = \text{Len\_Cmp}[\xi_2] \]
\[ \land \forall \xi_j \in \xi \]
\[ \text{elems } \circ \pi^*_\xi_j \neq \{0\} \]

The key signature for the Add\_Cmp\_1 operation is not presented. The operation has following proof obligation to demonstrate that the invariant is preserved.
Lemma 6.4 \( \text{pre}_{\text{Add}_C\text{mp}_1}[c, c_p, m, T, \xi](\alpha, \beta) \land (\alpha', \beta') = \text{Add}_C\text{mp}_1[c, c_p, m, T, \xi](\alpha, \beta) \Rightarrow \text{Inv}_C\text{Mus}[\alpha', \beta'] \)

Proof (Informal)
\[ \text{dom } \alpha = \text{dom } \beta \]
\[ \Rightarrow \text{dom } \alpha \cup \{c\} = \text{dom } \beta \cup \{c\} \]
\[ \Rightarrow \text{dom } \alpha' = \text{dom } \beta' \]

By the construction of \( \alpha', \beta' \) it follows that \( \alpha'(c_i) = \alpha(c_i) \) and \( \beta'(c_i) = \beta(c_i) \) for \( c_i \neq c \). From this it may be immediately deduced that the invariant holds for all \( c_i \) such that \( c \neq c_i \). Thus in order to complete the proof it is required to show that the invariant is true for \( c \). However, this is clear from the precondition to the Add_Cmp operation.

The next operation which is considered here is the \( \text{Add}_C\text{pt}_1 \) operation. This operation adds a counterpoint melody to an existing composition. The counterpoint melody \( \xi \) is a sequence of notes, i.e., \( \xi \in (\text{Note} \times \text{Vol} \times \text{Len})^* \). The \( \text{Add}_C\text{pt}_1 \) operation is defined as follows:

\[ \text{Add}_C\text{pt}_1[c, \xi] \beta \triangleq \beta \cap [c \mapsto (\xi)] \]

\[ \text{pre}_{\text{Add}_C\text{pt}_1}[c, \xi] \beta \triangleq \]
\[ \chi[c] \alpha \land \chi[c] \beta \]
\[ \land (\xi \neq \Lambda) \]
\[ \land \text{Len}_C\text{mp}(\text{hd}(\beta(c))) = \text{Len}_C\text{mp}[\xi] \]
\[ \land \text{elems } \circ \pi x \xi \neq \{0\} \]

The proof obligation of invariant preservation for the Add_Cpt operation is stated in the following lemma.

Lemma 6.5 \( \text{pre}_{\text{Add}_C\text{pt}_1}[c, \xi] \beta \land (\alpha', \beta') = \text{Add}_C\text{pt}_1[c, \xi] \beta \Rightarrow \text{Inv}_C\text{Mus}[\alpha', \beta'] \)

Proof

Clearly, \( \text{dom } \alpha' = \text{dom } \alpha \) and \( \text{dom } \beta' = \text{dom } \beta \). It follows from \( \text{dom } \alpha = \text{dom } \beta \) that \( \text{dom } \alpha' = \text{dom } \beta' \).

Clearly, \( \beta'(c) \) is the only part of the invariant that needs to be validated. The construction of \( \beta' \) ensures that \( \beta'(c) \neq \Lambda \). It is clear that \( \text{elems } (\beta'(c)) = \text{elems } (\beta(c)) \cup \xi \). Since \( \beta \) satisfies the invariant it follows that \( \neg \chi[\Lambda] \beta(c) \). The precondition guarantees that \( \xi \neq \Lambda \), thus it follows by the construction of \( \beta' \) that \( \neg \chi[\Lambda] \beta'(c) \).

The invariant for \( (\alpha, \beta) \) ensures that \( \forall \xi_1 \xi_2 \in \beta(c) \) that \( \text{Len}_C\text{mp}[\xi_1] = \text{Len}_C\text{mp}[\xi_2] \). It is required to demonstrate that this holds for \( (\alpha', \beta') \).

139
The precondition ensures that \( \text{Len}_\text{Cmp}[\xi] = \text{Len}_\text{Cmp}[\text{hd } \beta(c)] \) and \( \text{Len}_\text{Cmp}[\text{hd } \beta(c)] = \text{Len}_\text{Cmp}[\xi'] \) where \( \xi' \in \beta(c) \). It follows that \( \text{Len}_\text{Cmp}[\xi] = \text{Len}[\xi'] \) for all \( \xi' \in \beta(c) \) which is the required result.

Finally, the precondition for \( \text{Add}_\text{Cpt}_1 \) gives that \( \text{elems } \circ \pi_2^c \xi \neq \{0\} \) as required.

The next operation to be considered is the inverse of the \( \text{Add}_\text{Cpt}_1 \) operation. The effect of the \( \text{Rem}_\text{Cpt}_1 \) is to remove a counterpoint melody from a composition provided that at least one composition melody remains after the removal takes place.

\[
\text{Rem}_\text{Cpt}_1 [c, \xi][\beta] \triangleq \beta \uplus [c \mapsto \text{elems } \circ \beta(c)]
\]

\[
\text{pre}_\text{Rem}_\text{Cpt}_1 [c, \xi][\beta] \triangleq \xi \subset \text{elems } \circ \beta(c)
\]

The operation has an associated proof obligation for invariant preservation. This is stated in the following lemma.

**Lemma 6.6** \( \text{pre}_\text{Rem}_\text{Cpt}_1 [c, \xi][\beta \land (\alpha', \beta')] = \text{Rem}_\text{Cpt}_1 [c, \xi][\beta] \Rightarrow \text{Inv}_\text{C Mus}[\alpha', \beta'] \)

**Refinement Proof**

It has been remarked previously that if constraints are placed on \( \beta_1 \) then \( \beta_1 \) may be considered to be a refinement of \( \beta_0 \). The appropriate constraint is that \( \text{hd } \beta_1(c) \) is considered to be the melody of \( \beta_0(c) \).

Consequently, there is a proof obligation to demonstrate that the \( \text{Add}_\text{Cmp}_1 \) operation is a valid refinement of the \( \text{Add}_\text{Cmp}_0 \) operation. This is stated and proved in the following theorem.

**Theorem 6.1** \( R_{1,0} \circ \text{Add}_\text{Cmp}_1 [c, c_p, m, T, \xi](\alpha_1, \beta) = \text{Add}_\text{Cmp}_0 [c, c_p, m, T, \xi'] \circ R_{1,0}(\alpha, \beta) \) where \( \xi' = \text{hd } \xi \).

The definition of the \( \text{Add}_\text{Cmp}_1 \) operation gives that:

\( \text{Add}_\text{Cmp}_1 [c, c_p, m, T, \xi](\alpha, \beta) = (\alpha \sqcup [c \mapsto (c_p, m, T)], \beta \sqcup [c \mapsto \xi]). \)

Thus \( R_{1,0} \circ \text{Add}_\text{Cmp}_1 [c, c_p, m, T, \xi](\alpha, \beta) = R_{1,0}(\alpha', \beta') = (\alpha', (I \mapsto \text{hd }) \beta') \).

By definition of \( \beta' \) it follows that:

\[
(I \mapsto \text{hd }) \beta'
\]

\[
= (I \mapsto \text{hd }) \beta \sqcup [c \mapsto \text{hd } \xi]
\]

\[
= (I \mapsto \text{hd }) \beta \sqcup [c \mapsto \xi']
\]

\( \text{Add}_\text{Cmp}_0 [c, c_p, m, T, \xi'] \circ R_{1,0}(\alpha, \beta) \)
\[ \delta : \text{Ins} \xrightarrow{Jd} \text{Ins} \quad (6.11) \]
\[ \epsilon : \text{Ins} \xrightarrow{Jd} \text{Pr}r \xrightarrow{Jd} \quad (6.12) \]
\[ \zeta : \text{Or} \xrightarrow{Jd} \text{Cnd} \xrightarrow{Jd} \quad (6.13) \]
\[ \eta : \text{Or} \xrightarrow{Jd} \text{PIns} \xrightarrow{Jd} \quad (6.14) \]
\[ \vartheta : \text{Cmp} \xrightarrow{Jd} (\text{Ins} \mapsto (\text{Note} \times \text{Vol} \times \text{Len})^*) \quad (6.15) \]
\[ \vphi : \text{Cmp} \xrightarrow{Jd} \text{Cp} \xrightarrow{Jd} \text{Met} \times \text{Time} \quad (6.16) \]
\[ \kappa : \text{Ins} \mapsto \text{PNote} \quad (6.17) \]
\[ \mu : \text{Ins} \mapsto \text{Or} \_ \text{Sec} \quad (6.18) \]
Note 1
There may be several occurrences of a particular instrument in the orchestra. This, in
effect, means that an orchestra is essentially a bag of instruments. The distinction between
several occurrences of a particular instrument is achieved by introducing an instrument-id
\((Ins.Jd)\) which is distinct from the particular instrument type given by \(Ins\).

Note 2
The use of \(\vartheta : Cmp.Jd \mapsto Ins \mapsto (Note \times Vol \times Len)^*\) models a composition to be a
counterpoint of instruments, where each instrument plays a particular melody. Rest notes
may be part of the melody.

Note 3
The orchestra model is not a valid refinement of the basic model unless an additional
assumption is made. This requires that the melody \(\beta\) in the basic model be obtained from
a specified instrument in the orchestra model.

Note 4
A player may play more than one instrument in the orchestra. However, it is necessary
that there are no overlapping non rest notes.

Note 5
The orchestra is a bag in its most abstract form. This may be modelled by \(\eta_\beta\) of the form
\(Or.Jd \mapsto Ins \mapsto N_1\).

Note 6
The limitation of the orchestra model is that each instrument is limited to playing a single
note at a time.
The orchestra model is represented in pictorial form above. The diagram indicates that the various instruments are playing melodies in parallel, independently of one another. The invariant for the orchestra model is defined as follows:

\[ Inv_{Orc}[\delta, \epsilon, \zeta, \eta, \vartheta, \kappa, \mu] \triangleq \]
\[ \text{dom} \delta = \text{dom} \epsilon \]
\[ \land \text{rng} \delta = \text{dom} \kappa = \text{dom} \mu \]
\[ \land \text{dom} \zeta = \text{dom} \eta \]
\[ \land \text{dom} \vartheta = \text{dom} \varpi \]
\[ \land \forall c \in \text{dom} \vartheta \]
\[ \land \forall i, j \in \text{dom} \vartheta(c) \]
\[ \text{Len}_Cmp[\vartheta(c)(i)] = \text{Len}_Cmp[\vartheta(c)(j)] \]
\[ \land \forall c \in \text{dom} \vartheta, \forall i \in \vartheta(c) \]
\[ \text{elems} \circ \pi^*_i \vartheta(c)(i) \neq \{0\} \]

The invariant ensures that the melody for each instrument in the composition is exactly the same length and has at least one non rest note.

### 6.4.1 Exploring the model

This section determines the adequacy of the orchestra model by model interrogation. The objective is to determine how effective the model is in answering pertinent questions on the domain of music and compositions.

The invariant has placed no constraint on the model to ensure that the notes specified for a particular instrument in the composition are actually playable by the instrument. In general, each instrument operates between a minimum note and a maximum note. Furthermore, within this range only distinct pitches are playable by the instrument. The model is evaluated here to determine if it may decide if a piece of music specified for a particular instrument is actually playable by the instrument.

**Question 6.6** Can instrument i play its portion of composition c?

This question is answered by the \( Chk_{Rng} \) operation.

\[ Chk_{Rng}[c, i](\vartheta, \kappa) \triangleq \]
\[ \text{elems} \circ \pi^*_i \vartheta(c)(i) \subseteq \kappa(i) \]
**Question 6.7** Is the score for composition c valid? That is, is every specified note for every instrument i valid in the score?

This question is answered by the \( \text{Chk}\_\text{Insts} \) operation.

\[
\text{Chk}\_\text{Insts}[c](\vartheta, \kappa) \triangleq \\
\vartheta(c) = \Lambda \\
\quad \leftrightarrow \text{TRUE} \\
\quad \text{Let } i \in \vartheta(c) \text{ in} \\
\quad \text{Let } \vartheta' = \vartheta \uparrow [c \mapsto \forall [i] \vartheta(c)] \text{ in} \\
\quad \text{Chk}\_\text{Rng}[c, i](\vartheta, \kappa) \\
\quad \leftrightarrow \text{Chk}\_\text{Insts}[c](\vartheta', \kappa) \\
\quad \leftrightarrow \text{FALSE}
\]

The \( \text{Chk}\_\text{Insts} \) operation determines if the score written for a collection of instruments is actually valid. A composition is valid if this is the case. The invariant should be updated to ensure that this constraint is adhered to. The next operation determines the instruments played in a particular orchestra.

**Question 6.8** What instruments are played in orchestra \( O \)?

This importance of this question is that it enables an orchestra to determine whether it is capable of playing an individual composition. If the orchestra contains a superset of the set of instruments required for the composition then clearly the orchestra may play the composition.

The instruments in a particular orchestra is given by the \( \text{Ins}\_\text{Orc} \) operation. The model \( \eta : Or\_Jd \mapsto \mathcal{P}\text{Ins}\_Jd \) does not enable the required result to be immediately derived as it yields the individual instrument ids. However, what is required is the individual instruments. This problem may be seen by the following example:

\[
\eta(O) = \{vn_1, vn_2, vl_1\}
\]

This indicates that the orchestra \( O \) has the three instruments as specified. However, it does not yield the required answer which is that there are two instruments, i.e., violins and violas = \{vn, vl\} in orchestra \( O \). This is rectified with the \( \text{Ins}\_\text{Orc} \) operation.

\[
\text{Ins}\_\text{Orc} : Or\_Jd \mapsto (Or\_Jd \mapsto \mathcal{P}\text{Ins}\_Jd) \times (\text{Ins}\_Jd \mapsto \text{Ins}) \mapsto \mathcal{P}\text{Ins}
\]

\[
\text{Ins}\_\text{Orc}[O](\eta, \delta) \triangleq \mathcal{P}\delta(\eta(O))
\]

The precondition for this operation must ensure that the orchestra exists and instruments exist.
\[ \text{pre}_{\text{InsOrc}}[O](\eta, \delta) \triangleq \]
\[ \chi[O]^{\eta} \]
\[ \land \eta(O) \subseteq \text{dom} \delta \]

**Question 6.9** Who are the players in orchestra O?

This is given by the \( \text{PlrsOrc} \) operation and is defined as follows:

\[ \text{PlrsOrc} : \text{Orjd} \mapsto (\text{Orjd} \mapsto \mathcal{P} \text{Insjd}) \times (\text{Insjd} \mapsto \text{Plrdjd}) \mapsto \mathcal{P} \text{Plrdjd} \]

\[ \text{PlrsOrc}[O](\eta, \epsilon) \triangleq \mathcal{P} \epsilon \circ \eta(O) \]

The definition of \( \text{PlrsOrc} \) follows since \( \delta(O) = \{i_1, ..., i_n\} \) gives a list of every instrument in the orchestra. Each instrument is played by exactly one player and this relationship is given by \( \epsilon \). Consequently, the players in the orchestra are given by \( \mathcal{P} \epsilon \circ \eta(O) \). The precondition for the operation is given by:

\[ \text{pre}_{\text{PlrsOrc}}[O](\eta, \epsilon) \triangleq \]
\[ \chi[O]^{\delta} \]
\[ \land \eta(O) \subseteq \text{dom} \epsilon \]

**Question 6.10** What instruments does p play?

It is valid for a player to play more than one instrument in an orchestra. However, it must be ensured that there are no overlapping notes non rest notes. The instruments \( p \) plays is determined via inverse images as follows:

\[ \epsilon^{-1} : \text{Plrdjd} \mapsto \mathcal{P} \text{Insjd} \]

Thus \( \epsilon^{-1}(p) = \{i_1, i_2, ..., i_k\} \). However, the individual instruments are of interest not the instrument ids. This is obtained by combining \( \epsilon^{-1} \) with \( \delta \) to get:

\[ \text{InsPlr} : \text{Plrdjd mapsto}(\text{Insjd} \mapsto \text{Plrdjd}) \times (\text{Insjd} \mapsto \text{Ins}) \mapsto \mathcal{P} \text{Ins} \]

\[ \text{InsPlr}[p](\epsilon, \delta) \triangleq \mathcal{P} \delta(\epsilon^{-1}(p)) \]

The precondition for the operation is given as follows:

\[ \text{pre}_{\text{InsPlr}}[p](\epsilon, \delta) \triangleq \]
\[ \chi[p]^{\epsilon^{-1}} \]
\[ \land \epsilon^{-1}(p) \subseteq \text{dom} \delta \]

145
QUESTION 6.11 Suppose p plays several instruments. Is p able to play its part successfully in the composition; i.e., are there any overlapping non rest notes for the instruments that player p plays?

The Trn_Xlt operation has been presented previously in Section 6.2. Its effect is to convert a single note composition specification to a temporal representation of the form $\psi : R \mapsto (Note \times Vol)$ where $\psi$ is built up using half open half closed intervals.

There is a corresponding operation for the orchestra model which is defined analogously to yield a representation of the form $\psi : R \mapsto (Ins \mapsto (Note \times Vol))$. Thus $\psi$ represents the temporal representation of the composition $c$ of length $T$.

Thus to ensure that there are no overlapping non rest notes involves placing the following non constructive constraint on $\psi$:

$$\forall i_1, i_2 \in Ins_{Plr}[p] (\epsilon, \delta) \forall t \in [0, T) \quad (\pi_2 \circ \psi(t)(i_1) > 0 \land \pi_2 \circ \psi(t)(i_2) > 0) \Rightarrow i_1 = i_2$$

The model may answer other elementary questions such as:

- How many instruments are in the string section?
- What instruments are present in the string section?
- What Orchestra(s) does conductor $c_p$ conduct?

6.4.2 Operations on Orchestra Model

Several operations may be defined in the model. These include Add_Cmp and Rem_Cmp operations which add and remove a composition from the model respectively. Similarly, the Add_Plr operation to add a player to the orchestra, the Add_Ins operation which creates a new instrument, and the AddO_Ins operation which assigns an instrument to the orchestra. Due to space constraints these operations are not presented.

6.4.3 Criticisms of the Model

This section evaluates the effectiveness of the model. The exploration of the model has demonstrated that a formal model of aspects of music may successfully encode several properties of music. However, there are other properties which are not encoded in the model. The main strengths and weaknesses of the model are described as follows:

- The model is adequate at modelling aspects of a musical score for an orchestral composition. The various instruments which contribute to the musical composition are captured in the model.
• There is no explanation as to why a particular piece of music is melodic. Furthermore, there is no explanation as to why several notes played together by different instruments produces harmony whereas other notes played together are dissonant.

• The orchestra model has assumed that each instrument in the orchestra plays a single note at a time. This is unrealistic as the classical guitar or piano may play several notes at once. The model may be extended to address this weakness.

• The structure of a composition is not modelled. For example, sonata form consists of exposition, development and recapitulation. This is not captured in the model.

• The model does not concern itself with metre and the composition is not divided into bars.

• The timbre of a musical instrument is not modelled. The pitch, volume and note duration is modelled, however, the particular characteristic of an instrument, which for example, distinguishes an oboe from a cello is not modelled.

• Musical Plagiarism is inadequately addressed in the model.

6.5 Constructive Time in Music

Section 6.2 has demonstrated that the basic model of music may be converted via the translation operation $Trn_Xlt$ to a temporal representation $\psi : \mathbb{R} \mapsto (Note \times Vol \times Len)$. However, the said representation is inherently non constructible as it is composed of half open and half closed intervals in the reals.

The model of musical compositions may be adapted to the domain of recorded music. Consequently, implementation issues must be considered for this domain. This requires a constructive temporal representation of musical compositions. The approach taken in this section is to consider a smallest possible time duration, for example, $10^{-15}$. Every time period is then assumed to consist of several of these time atoms, and the model of time for music is in fact isomorphic to the natural numbers $\mathbb{N}$. Consequently, each natural number represents a time duration of one time atom.

$$\phi : \mathbb{N} \mapsto InsId \mapsto (Note \times Vol)$$  \hspace{1cm} (6.19)

However, since a composition is of finite time duration, only a finite subset of $\mathbb{N}$ is necessary to model a composition. The start of a composition is represented by $t = 0$, and the end of a composition by $t = T$. At each time instant several notes are played by the individual instruments. Clearly, several of these notes are rest notes and the remainder (in theory) are harmonious.
\[ \phi(t) : \text{Ins} Jd \mapsto (\text{Note} \times \text{Vol}) \]  
\[ \text{rng} \phi(t) = \{(p_1, v_1), (p_2, v_2), \ldots, (p_k, v_k)\} \]

Thus the representation of the notes that are played at time \( t \) have signature:

\[ \phi_t : \mathbb{N} \mapsto \mathcal{P}(\text{Note} \times \text{Vol}) = (\mathcal{I} \mapsto \text{rng}) \phi \]

The notes that have been played before time \( t = n \) are given by:

\[ \nabla_{k=0}^{n-1} : \mathbb{N} \mapsto (\mathbb{N} \mapsto \mathcal{P}(\text{Note} \times \text{Vol})) \longrightarrow \mathcal{P}\mathcal{P}(\text{Note} \times \text{Vol}) \]
\[ \nabla_{k=0}^{n-1}[\mathcal{I}] \phi_t \triangleq \{\phi_t(0), \phi_t(1), \ldots, \phi_t(n-1)\} \]

The notes that will be played after \( t = n \), assuming \( t = T \) is the last time instant in the composition in which a note is played is given by:

\[ \Delta_{k=n+1}^{T} : \mathbb{N} \mapsto (\mathbb{N} \mapsto \mathcal{P}(\text{Note} \times \text{Vol})) \longrightarrow \mathcal{P}\mathcal{P}(\text{Note} \times \text{Vol}) \]
\[ \Delta_{k=n+1}^{T} \phi_t \triangleq \{\phi_t(n+1), \phi_t(n+2), \ldots, \phi_t(T-1), \phi_t(T)\} \]

The remainder of this section considers how the constructive model of time enables crescendos and decrescendos to be modelled. It is quite normal for musical compositions to exhibit variations in sound volume at different parts of the composition. This may vary from pianissimo to fortissimo. The former refers to when music is played extremely softly, and the latter extremely loudly. This is employed by the composer to achieve special dramatic effects. Crescendos and decrescendos present gradual variations in sound volume. The former proceeds from a low volume to a high volume and vice versa for the latter.

The total volume of sound produced at instant \( t = n \) is considered to be the summation of the individual volume of sound produced at this instant by the various instruments. This is defined precisely as follows:

\[ \phi(t) : \text{Ins} Jd \mapsto (\text{Note} \times \text{Vol}) \]
\[ \nu_t = (\mathcal{I} \mapsto \pi_2) \phi(t) : \text{Ins} Jd \mapsto \text{Vol} \]

Consequently, the total volume of music produced at instant \( t = n \) is the summation of the individual volumes and is given by the \( Vlm_t \) operation which is defined as follows:

\[ Vlm_t : (\text{Ins} Jd \mapsto \text{Vol}) \mapsto \text{Vol} \mapsto \text{Vol} \]
\[ Vlm_t[\theta]v \triangleq v \]
\[ Vlm_t[v]v \triangleq v \]

Let \( i \in \text{dom} \nu_t \) in
\[ Vlm_t \mathcal{S} [i] v_t (v + v_t (i)) \]

The \( Vlm_t \) operations gives the volume of sound produced at a particular time instant \( t \). The \( Vlm_t \) operation is one part of the \( Vlm \) operation which gives the volume of sound produced at the various time instants. This operation enables a precise formulation of a crescendo to be made, for example, to model the *sneeze* on interval \([t_1, t_2]\) from the Hary Janos Suite by Kodaly may be modelled as follows:

Given any \( t_i, t_j \in [t_1, t_2] \) where \( t_i \in \mathbb{N} \) and \( t_j \in \mathbb{N} \) and \( t_i < t_j \) then,

\[ V ol(t_i) \leq V ol(t_j) \quad (6.22) \]

Similarly, a decrescendo can be modelled via a monotonic decreasing sequence.

\[ V ol(t_i) \geq V ol(t_j) \quad (6.23) \]

### 6.6 Elementary Temporal Logic in \( VDM^{\oplus} \)

This section considers the problem of modelling time discretely in formal methods, and in \( VDM^{\oplus} \), in particular. Pnueli, [61] defines the following future tense operators:

- \( \Box P \quad \triangleq \quad \text{property always holds in the future.} \)
- \( \Diamond P \quad \triangleq \quad \text{property holds sometime in the future.} \)
- \( \bigcirc P \quad \triangleq \quad \text{property holds at next time instant.} \)

The present is considered to be part of the future in Pnueli’s system. Consequently, the \( \Diamond P \) property is true if it is true precisely at this time instant without ever being true at any future time. The \( \Box \) operator means that the property \( P \) is true now and at all future time instances.

Half opened and half closed intervals have been employed in Section 6.2 for the translation of a musical specification to real time. There are corresponding open and closed discrete intervals.

\[
[i, j] \triangleq \{ k | k \in \mathbb{N} \land i \leq k \land k \leq j \}
\]

\[
[i, j) \triangleq \{ k | k \in \mathbb{N} \land i \leq k \land k < j \}
\]

These discrete intervals enable restricted \( \Box \) and \( \Diamond \) temporal operators to be defined as follows:
\( \square_{[i,j]} P \triangleq \text{property always true in discrete interval } [i,j]. \)

\( \Diamond_{[i,j]} P \triangleq \text{property true somewhere in } [i,j]. \)

Consequently, \( \square_{[i,j]} P \) is essentially a shorthand for

\[
P_i \land P_{i+1} \land \ldots \land P_j
\]

Similarly, \( \Diamond_{[i,j]} P \) is essentially a shorthand for

\[
P_i \lor P_{i+1} \lor \ldots \lor P_j
\]

Note

The notation \( P_k \) indicates the truth value of the predicate \( P \) at time \( t = k \).

### 6.7 Summary

The standard specification style for musical compositions has been employed for several hundred years. It is a terse and economical symbol system for expressing musical compositions. This chapter has presented a formal model of aspects of musical compositions, and has successfully demonstrated that aspects of musical compositions may be formally encoded in a model.

The approach is quite distinct from the traditional five line stave approach to musical specification. The model presented in this chapter is believed to be an original application of formal methods. The model evolved from a simple model where a musical composition is considered to be a sequence of notes, to a multi note model and finally to the orchestra model.

The model was revised in each case due to the perceived inadequacies in the model. The adequacies and inadequacies of the final model are discussed in Section 6.4.3. The exploration of the model demonstrated the importance of time in music, and non-constructive and constructive formulations were presented.

**Conclusion 6.1** *The exploration of time in music is important. The exploration raises questions on the nature of time and whether time is discrete or continuous in formal methods.***

**Conclusion 6.2** *Non constructivism in specifications is useful as an initial statement of the properties of the specification. However, any implementation must resolve any non-constructive parts of the specification.*
Chapter 7

Conclusions

The formal specification of the requirements of a proposed system is the main application to which formal methods has been applied. The formal specification is an unambiguous statement of the requirements of the system, and is expressed in a formal mathematical notation. This thesis has demonstrated that formal methods may be adapted to mathematical modelling of aspects of the real world. In particular, the thesis employed the method and notation of the Irish school of VDM to develop constructive models of organizations, hierarchies and belief systems.

The model serves as a mechanism for formally encoding properties of aspects of organizations, structures and belief systems. The model also serves as a mechanism for understanding and testing hypotheses about aspects of the real world. The key point is that these models were not developed for implementation purposes; indeed, implementation may not be meaningful for several of these domains.

The models serve as representations of these domains, and may be referred to in any dispute on the actual properties of the domain. The method of construction and evaluation of the models is considered. Each model is adequate at explaining some aspects of the world, and inadequate at explaining other aspects. The fact that there is a model of the world enables further properties and a more detailed understanding of the world to be gained.

This thesis considered the problem of modelling the domain of religions of the world in a formal way. It is believed that the model of religion which is presented in this thesis is original. The modelling is concerned with the problem of representing the structure of the organized religions of the world, and in capturing the essential beliefs of a religion. It is accepted that religions of the world are far too complex to capture exactly; however, the model developed in this thesis demonstrates that important aspects of religion may be captured formally. Furthermore, model interrogation enables additional properties of religion to be determined. The study of the model of religion indicated that the model itself is generic, and may be applied to model the hierarchical structure of companies, political institutions, or a university.
This realization justified an examination of the classical bill of material structure [12], as bills of materials are abstractions of hierarchies. This examination presented a formal proof that a bill of material contains at least one basic part and identified a shrinking operator \((\beta)\) which acts as an invariant on a bill of material. The annihilator function \((\mathcal{A})\) may then be defined as the limit of successive applications of \(\beta\). The bill of materials chapter places the annihilator function on a firmer theoretical foundation and a formal proof that a structure \(\mu \in (X \mapsto \mathcal{P}X)\) is a bill of material if and only if \(\mathcal{A}(\mu) = \emptyset\) is presented. The chapter compares the Björner formulation of the invariant for bill of material structures and the annihilator formulation. It is concluded that while the annihilator function is a nice theoretical construct, it is slightly more difficult to work with than the Björner formulation. An examination of the join operation for bill of material structures yields that the operation is sensible only for consistent bill of materials.

The representation of beliefs and facts about a religion resembles a catalogue of information about the religion. This suggested an examination of the catalogue from the classical model of the file system in [10]. Consequently, the model of the file system is examined in detail. This model re-examines the work done by Mac An Airchinnigh in [46] and seeks further simplicity. The chapter has important results for the usability of formal methods; especially, in the difficulty of doing manual proofs, and the limitations of the stepwise development approach of the Vienna Development Method. The chapter notes that the complexity of proof of invariant preservation is directly related to the complexity of the invariant. Thus it is important to work with the appropriate model and invariant. The file system model is extended to include file aliasing.

The model of the stock exchange presented in this thesis is believed to be original. The model considers the problem of modelling companies registered on the stock exchange, the shareholders of the company, and the financial operations involved in share sale or purchase. The importance of this model is that it demonstrates that formal methods may be employed to model aspects of the structure and operations of an organization in the real world. The stock exchange is one particular instantiation of organizations. Three conceptual viewpoints of the stock exchange are presented and financial operations are considered.

The model of music in this thesis demonstrates that formal methods may be applied to model a cultural domain in the real world. The specification style is an alternative to the standard five line stave specification style of written music, and is closer to the domain of recorded music. The model demonstrates that properties of music may be formally encoded in a model. The problem of time is fundamental in music, and this chapter presents several elementary results in modelling time and music.

In general, the models presented in this thesis are constructive. Consequently, the models themselves may be implemented in some programming language if this is required.
However, this thesis stresses the importance of studying models for the sake of the models themselves and is not concerned with implementation issues for a particular machine architecture. The fact that a model is constructive indicates that it may be implemented or created in the real world. This suggests that the constructive models may be employed to shape the world. In this thesis, almost all of the models which are presented are constructive.

It is a key requirement to determine the adequacy of a particular model is of an aspect of the world. In general, the evaluation of a particular model takes the form of model exploration. The thesis stresses the importance of assessing the adequacy or otherwise of each individual model. In this way, an informed decision may be made as to whether a particular model is a suitable representation of a particular system.

The importance of the said models is that they serve as representations of these domains, and may be referred to in any dispute on the actual properties of the domain. Furthermore, the model serves as a means of formal derivation of further properties of the particular aspect of the real world. The very fact that there is a model means that there is a rigorous means of gaining a more detailed understanding of the real world. The actual method of construction and evaluation of the models is considered. In general, a model is good at explaining some aspects of the world, and weak at explaining other aspects. The most adequate model is chosen as a representation of that aspect of the world in which one is interested.

This particular use of formal methods to model a domain for which there is no intention to implement, and for which implementation is not meaningful is believed to be an original application of formal methods. The importance of this approach is that it enables a more detailed understanding of these domains to be gained and suggests that it may be valid to employ formal methods to model aspects of the social sciences.

The secondary objective of this thesis is to identify mathematical structures which may be useful in the modelling itself. Several mathematical structures which may be useful in modelling are presented in the appendices to this thesis. It is intended that such structures should have practical applications in modelling. Several results on unique sequences and multi-sequences are presented. The indexed monoid structure as defined in [47] is extended to general indexed structures. Finally, the free group structure is generalized to the free set and free list structures.

In summary, the key objective of this thesis is to demonstrate that a formal specification language may be employed to model aspects of the world. This thesis explains how such models of the world may be built and evaluated. The key conclusion is that such mathematical models offer a valuable means for the examination of aspects of the world. The said models enable a formal statement of some aspects of the world to be expressed. Further properties may be derived via model interrogation and mathematical reasoning.
7.1 Applications of This Thesis

There are several potential applications of this thesis to computer science. The actual construction and evaluation of models as exhibited in this thesis may be interpreted as modelling for the sake of modelling. This approach may be applicable to determining the appropriate requirements for a particular project. In general, requirement exploration concerns itself with exploring various requirements for the proposed system, i.e., before the explicit requirements are actually detailed. In fact, determining the appropriate requirements for a proposed system is a non-trivial task. It is often the case that further desirable requirements which require implementation only become apparent at a late stage during the software project. Such an occurrence has a corresponding effect on the quality of the resulting software, since design documents, coding and test plans are affected at a late stage in the project.

The modelling exhibited in this thesis may be applied to assist in determining the appropriate requirements for a proposed system. There may be several candidate sets of requirements, \( R_1, R_2, \ldots, R_n \), where each \( R_i \) represents a set of requirements. Suppose \( M_1, M_2, \ldots, M_n \) are adequate models for \( R_1, R_2, \ldots, R_n \) respectively, then the exploration of the properties of these models serves as a rigorous and methodical means of determining the most appropriate model \( M_j \) and consequently, the most appropriate set of requirements \( R_j \). Each model is explored for evaluation and understanding purposes, in order to determine the most appropriate choice of model for the requirements of a proposed system. Thus modelling for the sake modelling serves practical purposes in this case.

Modelling for the sake of modelling exhibited here in this thesis may be applicable to identifying potential new products or systems. This is since the models presented here are constructive; it follows that they may be implemented on some machine architecture. Thus a formal means of understanding the implications of a new product may be obtained in this way. The aim is to identify potential systems which may be worthy of eventual implementation. Many of these possible systems may be of little practical interest, as they may have minimal applicability to the world. However, the exploration may yield models with desirable properties which may then be successfully implemented.
Appendix A

Unique Sequences

A.1 Introduction

The objective of this section is to present a study on the relationship between unique sequences \( \Sigma^*_1 \) under unique sequence concatenation, and set theoretical operations. An equivalence relation is defined on the unique sequence domain, and in fact, the equivalence relation is a congruence. This enables the quotient monoid of unique sequences to be formed. In fact, this quotient monoid is isomorphic to \((\mathcal{P}\Sigma, \cup, \emptyset)\). A similar isomorphism may be derived for \((\mathcal{P}\Sigma, \cap, \Sigma)\).

As the identification is made via equivalence classes, where each equivalence class may contain multiple entries, a canonical sequence representation of a set is desirable. This is achieved by placing a total order on \( \Sigma \) and choosing the unique ordered sequence from each class. This provides a mechanism for the implementation of sets via unique sequences. Set complement operations may also be provided for unique sequences.

The relationship between multisets and general sequences in \( \Sigma^* \) is then examined. The approach is similar to unique sequences, and an equivalence relation is defined as before. The equivalence relation is, in fact, a congruence. The multiset is then identified as the quotient monoid of sequences of \((\Sigma^*, \prec, \Lambda)\) under the congruence equivalence relation. In fact, there is an isomorphism between the multiset (or bag structure) \((X \mapsto N_1, \oplus, \theta)\) and the quotient monoid of \((\Sigma^*, \prec, \Lambda)\) under concatenation.

Finally, the refinement of sequences \( \Sigma^* \) into arrays \((N \mapsto \Sigma)\), and the refinement of maps \((X \mapsto Y)\) into sequences \((X \times Y)^*\) is considered.
A.2 A Study of \( (\Sigma^*_i, \Diamond, \Lambda) \)

Let \( \Sigma^* \) denote the set of all unique sequences, i.e., all those sequences of elements of \( \Sigma^* \) which do not contain duplicate elements. The unique sequence operation is defined in [45] as follows:

\[
(\langle e \rangle \smallsetminus \sigma) \Diamond \tau \overset{\Delta}{=} \begin{cases} 
\sigma \Diamond \tau & \text{if } \chi[\sigma]\tau \\
\langle e \rangle \smallsetminus (\sigma \Diamond \tau) & \text{otherwise.}
\end{cases}
\]

(A.1)

\[
\Lambda \Diamond \sigma \overset{\Delta}{=} \sigma = \sigma \Diamond \Lambda
\]

(A.2)

Theorem A.1 \( (\Sigma^*_i, \Diamond, \Lambda) \) forms a non commutative monoid.

Proof

The proof is tedious, and uses mathematical induction based on the length of the unique sequence. The closure property is proved by induction on \(|s_1|\).

It is required to demonstrate that \( s_1 \Diamond s_2 \in \Sigma^*_i \) for \( s_1, s_2 \in \Sigma^*_i \). The basis case is \(|s_1| = 0\), i.e. \( s_1 = \Lambda \). Then by applying the definition of the \( \Diamond \) operator it follows:

\[
s_1 \Diamond s_2 = \Lambda \Diamond s_2 = s_2 = s_2 \Diamond \Lambda = s_2 \Diamond s_1
\]

Therefore \( s_1 \Diamond s_2 \in \Sigma^*_i \) for \(|s_1| = 0\), and the basis case is proved.

For the inductive step, suppose \( s_1 \Diamond s_2 \in \Sigma^*_i \), for all \( s_1, s_2 \in \Sigma^*_i \) such that \(|s_1| \leq n\). Consider a string \( s_1 \) of length \( n + 1 \).

\[
s_1 \Diamond s_2 = (\langle e \rangle \smallsetminus s) \Diamond s_2, \text{ where } |s| = n. \text{ There are two possible cases from the definition of } \Diamond.
\]

1. \( \chi[e]s_2 \)

Then \( s_1 \Diamond s_2 = s \Diamond s_2 \), and \(|s| = n\). Hence, by the inductive hypothesis \( s \Diamond s_2 \in \Sigma^*_i \).

Thus \( s_1 \Diamond s_2 \in \Sigma^*_i \).

2. \( \neg \chi[e]s_2 \)

Then \( s_1 \Diamond s_2 = (\langle e \rangle \smallsetminus (s \Diamond s_2)) \)

It is clear that \( \neg \chi[e]s_2 \) since \( s_1 = \langle e \rangle \smallsetminus s \in \Sigma^*_i \), and furthermore, it is immediate from the definition of \( \Diamond \), that \( \neg \chi[e]s_1 \) and \( \neg \chi[e]s_2 \Rightarrow \neg \chi[e](s_1 \Diamond s_2) \).

As \(|s| = n\), it follows from the inductive hypothesis that \( s \Diamond s_2 \in \Sigma^*_i \). Hence, \( \langle e \rangle \smallsetminus (s \Diamond s_2) \in \Sigma^*_i \), i.e., \( s_1 \Diamond s_2 \in \Sigma^*_i \) as required.
Hence, the closure property follows by induction.

The proof of the associative property uses an inductive argument based on \(|s_1|\). The basis case is \(|s_1| = 0\) as before, and corresponds to \(s_1 = \Lambda\), and for this case it follows:

\[
(s_1 \diamond s_2) \diamond s_3 = (\Lambda \diamond s_2) \diamond s_3 = s_2 \diamond s_3 = \Lambda \diamond (s_2 \diamond s_3) = s_1 \diamond (s_2 \diamond s_3)
\]

Therefore the associative property is true for the basis case. For the inductive step, suppose that the property holds for all \(s_1, s_2, s_3\) such that \(|s_1| \leq n\). Then consider \(s_1\) of length \(n + 1\), i.e., \(s_1 = (e) - \xi_1\), where \(|\xi_1| = n\). Then \((s_1 \diamond s_2) \diamond s_3) = ((e) \sim \xi_1) \diamond (s_2 \diamond s_3)\), where \(|\xi_1| = n\). There are two cases as before:

1. \(\chi[e] s_2\)

   \[
   (s_1 \diamond s_2) \diamond s_3 = (((e) \sim \xi_1) \diamond s_2) \diamond s_3 = (\xi_1 \diamond s_2) \diamond s_3.
   \]

   Hence, by the inductive hypothesis we have,

   \[
   (\xi_1 \diamond s_2) \diamond s_3 = \xi_1 \diamond (s_2 \diamond s_3)
   \]

   \(e \in s_2 \Rightarrow e \in (\xi_1 \diamond s_2)\) and \(e \in s_2 \diamond s_3\)

   Thus we have \(e \in (\xi_1 \diamond s_2) \diamond s_3\) and \(e \in \xi_1 \diamond (s_2 \diamond s_3)\). Thus we get:

   \[
   (((e) \sim \xi_1) \diamond s_2) \diamond s_3 = (\xi_1 \diamond s_2) \diamond s_3 = \xi_1 \diamond (s_2 \diamond s_3) = ((e) \sim \xi_1) \diamond (s_2 \diamond s_3) = s_1 \diamond (s_2 \diamond s_3)
   \]

   Thus we have \((s_1 \diamond s_2) \diamond s_3 = s_1 \diamond (s_2 \diamond s_3)\) and this case is proved.

2. \(\neg \chi[e] s_2\)

   This case is proved by divide and conquer techniques via two subcases:

   a) \(\chi[e] s_3\) then,

   \[
   (((e) \sim \xi_1) \diamond s_2) \diamond s_3 = ((e) \sim (\xi_1 \diamond s_2)) \diamond s_3 = (\xi_1 \diamond s_2) \diamond s_3\]

   Since \(|e| = n\), the inductive hypothesis holds for \(\xi_1\) and we get:
\[(\xi_1 \diamond s_2) \diamond s_3 = \xi_1 \diamond (s_2 \diamond s_3)\]
\[e \in s_3 \Rightarrow e \in (s_2 \diamond s_3)\]

\[\Rightarrow (s_1 \diamond s_2) \diamond s_3\]
\[= (\xi_1 \diamond s_2) \diamond s_3\]
\[= \xi_1 \diamond (s_2 \diamond s_3)\]
\[= (\langle e \rangle \ominus \xi_1) \diamond (s_2 \diamond s_3)\]
\[= s_1 \diamond (s_2 \diamond s_3)\]

Hence this case follows.

(b) \(\neg \chi \llbracket e \rrbracket s_3\)
\[(s_1 \diamond s_2) \diamond s_3\]
\[= [\langle e \rangle \ominus (\xi_1 \diamond s_2)] \diamond s_3\]
\[= \langle e \rangle \ominus [(\xi_1 \diamond s_2) \diamond s_3]\]
\[= \langle e \rangle \ominus [\xi_1 \diamond (s_2 \diamond s_3)]\]
\[= (\langle e \rangle \ominus \xi_1) \diamond (s_2 \diamond s_3)\]
\[= s_1 \diamond (s_2 \diamond s_3)\]

where we have applied the inductive hypothesis for \(\xi_1\) and also noted that
\(\neg \chi \llbracket e \rrbracket s_2\) and \(\neg \chi \llbracket e \rrbracket s_3\) ensures \(\neg \chi \llbracket e \rrbracket s_2 \diamond s_3\).

This completes the proof that the \(\diamond\) operator is associative. It is clear that \(\Lambda\) is the
identity element of \(\Sigma^*_i\) under \(\diamond\). Hence \((\Sigma^*_i, \diamond, \Lambda)\) forms a monoid.

Note \((\Sigma^*_i, \diamond, \Lambda)\) is non-commutative.

This can be seen immediately, as given \(s_1 = \langle a \rangle\), and \(s_2 = \langle b \rangle\) then \(s_1 \diamond s_2 = (a) \ominus (b)\),
whereas \(s_2 \diamond s_1 = (b) \ominus (a)\). Thus \(s_1 \diamond s_2 \neq s_2 \diamond s_1\), and thus \((\Sigma^*_i, \diamond, \Lambda)\) is non-commutative.

### A.3 Unique Sequences, Sets and \((\tilde{\Sigma}^*_i, \diamond', \Lambda)\)

**Lemma A.1** If \(|\Sigma| = n\) then the maximum length of any sequence \(s \in \Sigma^*_i\) is \(n\), i.e., \(|s| \leq n\)
\(\forall s \in \Sigma^*_i\).

**Proof**
Suppose there is a sequence \(s \in \Sigma^*_i\) such that \(|s| > n\), and let \(s = s_1 \ominus s_2 \ominus \ldots \ominus s_n \ominus s_{n+1}\),
for some \(s_1, s_2, \ldots s_{n+1} \in \Sigma\).
Suppose $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$
Then $s \diamond (\sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_n) = \sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_n$
$\Rightarrow s_i = s_j = \sigma_k$ for some $i, j, k$ where $i \neq j$.
$\Rightarrow s \notin \Sigma^n$
$\Rightarrow |s| \leq n$ for all $s \in \Sigma^n$

A key motivations for studying unique sequences is that they may be adapted to the implementation of sets; this is achieved by a canonical representation of a unique sequence, and is achieved by placing a total order on $\Sigma$, and choosing the representation of a set $S$ to be the unique totally ordered sequence in $\Sigma^*$ containing exactly one occurrence of each element of $S$.

The union and intersection operators in set theory are commutative, however, it has been demonstrated that the unique concatenation operator is not commutative. As it is desired to eventually provide a construction or implementation of sets via unique sequences, then this is meaningful only if an operation $\diamond'$ may be derived from $\diamond$ such that $\diamond'$ is, in fact commutative.

The fundamental result which is proved in the next section is that a quotient monoid of $\Sigma^*$, written as $\Sigma_i^*$ with the new $\diamond'$ operator is isomorphic to $(\mathcal{P}\Sigma, \cup, \emptyset)$. The quotient monoids derived by defining an equivalence relation on $\Sigma_i^*$, the equivalence relation is, in fact a congruence.

$$(\Sigma_i^*, \diamond', \sim) \cong (\mathcal{P}\Sigma, \cup, \emptyset) \tag{A.3}$$

The effect of defining the equivalence relation on $\Sigma_i^*$ is that the equivalence classes of $\Sigma_i^*$ turn out to be isomorphic to the subsets of $\Sigma$, and the equivalence relation is a congruence.

**Equivalence Relation and Equivalence Classes of $\Sigma_i^*$**

Suppose $s_1, s_2 \in \Sigma_i^*$, then $s_1 \sim s_2$ if $s_1 \diamond s_2 = s_2$ and $s_2 \diamond s_1 = s_1$. Informally, $s_1 \sim s_2$ if $s_1$ is a permutation sequence of $s_2$.

**Theorem A.2** The relation $\sim$ is an equivalence relation.

**Proof**

- Reflexive
  $$s_1 \diamond s_1 = s_1 \Rightarrow s_1 \sim s_1 \forall s_1 \in \Sigma_i^*$$
• Symmetric

Suppose \( s_1 \sim s_2 \) then,
\[
s_1 \diamond s_2 = s_2 \text{ and } s_2 \diamond s_1 = s_1
\]
Clearly, this is exactly the requirement for \( s_2 \sim s_1 \).

• Transitive

Suppose \( s_1 \sim s_2 \) and \( s_2 \sim s_3 \) then,
\[
s_1 \diamond s_2 = s_2 \text{ and } s_2 \diamond s_1 = s_1
\]
\[
s_2 \diamond s_3 = s_3 \text{ and } s_3 \diamond s_2 = s_2
\]
Then \( s_1 \diamond s_3 = s_1 \diamond (s_2 \diamond s_3) \)
\[
= (s_1 \diamond s_2) \diamond s_3
\]
\[
= s_2 \diamond s_3 = s_3
\]
\[
s_3 \diamond s_1 = s_3 \diamond (s_2 \diamond s_1)
\]
\[
= (s_3 \diamond s_2) \diamond s_1
\]
\[
= s_2 \diamond s_1
\]
\[
= s_1
\]

Thus \( s_1 \sim s_3 \), and \( \sim \) is transitive. Hence, \( \sim \) is an equivalence relation.

**Question A.1 (Equivalence Classes)** What are the equivalence classes of \( \Sigma^*_i \) under \( \sim \)?

Consider \( s = \sigma_1 \sim \sigma_2 \sim \ldots \sim \sigma_k \), then the equivalence class of \( s \), i.e., \( Cl(s) \) consists of all unique sequences \( s_i \in \Sigma^*_i \), such that \( s \diamond s_i = s_i \) and \( s_i \diamond s = s \).

In fact, it is demonstrated here that the equivalence class of \( s \) consists precisely of those sequences of \( \Sigma^*_i \) which are permutation sequence of \( s \). This may be seen by noting that if \( s_j \) is a permutation sequence of \( s \), then \( s \diamond s_j = s_j \) and \( s_j \diamond s = s \). Thus any permutation sequence of \( s \) is in the equivalence class of \( s \).

Conversely, suppose \( s_j \in Cl(s) \), then it is required to demonstrate that \( s_j \) is a permutation sequence of \( s \); suppose not, i.e., suppose \( s_j \) is not a permutation sequence of \( s \). Then there is an \( e \in \Sigma \) such that, \( \chi[e]s \) and \( \neg \chi[e]s_j \), or vice versa. Suppose the former, then by the definition of \( \diamond \) it follows that \( \chi[e](s \diamond s_j) \) and \( \chi[e](s_j \diamond s) \).
However, since $s_j \in Cl(s)$ it follows that $s \diamond s_j = s_j$, and $s_j \diamond s = s$. This implies that $\chi[e]s_j$ and $\chi[e]s$, which is a contradiction. Similarly, a contradiction is obtained for $\neg \chi[e]s$ and $\chi[e]s_j$

Thus $|s| = |s_j|$, and the equivalence class of $s$ consists precisely of those sequences in $\Sigma^*$, which are permutation sequences of $s$.

**Conclusion A.1** Intuitively the equivalence class of $s$ yields a subset of $\Sigma^*$ where no distinction is made in the order in which the elements of $s$ appear, i.e., the equivalence class of $s$ is effectively the collection of unique permutation sequences of $s$ in $\Sigma^*$.

**Question A.2 (Number of Equivalence Classes of size $k$)** How many equivalence classes of sequences of length $k$ are there in $\Sigma^*$?

Since $|\Sigma| = n$, there are $(\binom{n}{k})$ ways of selecting $k$ elements from $\Sigma$. That is, there are exactly $(\binom{n}{k})$ ways of selecting a unique sequence of length $k$ from $\Sigma^*$, where the order the elements appear in the sequence is ignored.

For each sequence $s$ of length $k$, there are exactly $k!$ elements in the equivalence class of $s$, i.e., there are exactly $k!$ permutations. Thus the total number of unique sequence of length $k$ is given by:

$$\binom{n}{k} \times k!$$

The order of the elements is ignored in the equivalence class of $s$, thus the number of equivalence classes of size $k$ is given by:

$$\binom{n}{k}$$

**Question A.3** How many equivalence classes are there in $\Sigma^*$?

This is given by the total sum of the number of equivalence classes of each size $k$, and is given by:

$$\sum_{0}^{n} \binom{n}{k}$$

The Binomial Theorem is well known, and states that $(1 + x)^n = \sum_{0}^{n} \binom{n}{k} x^k$; thus taking $x$ as 1, it follows:

$$2^n = (1 + 1)^n = \sum_{0}^{n} \binom{n}{k}$$

Therefore, the number of equivalence classes of $\Sigma^*$ under $\sim$ is equal to the number of subsets of $\Sigma$, we note that this is a necessary condition for the isomorphism we shall exhibit between this quotient monoid and $P \Sigma$. 

161
A Congruence Equivalence Relation

A congruence equivalence relation enables the quotient monoid to be constructed. In order to demonstrate that ~ is a congruence equivalence relation, it is required to show that if \( s_1 \sim s'_1 \) and \( s_2 \sim s'_2 \) then:

\[
s_1 \diamond s_2 \sim s'_1 \diamond s'_2
\]

By definition, \( s_1 \sim s'_1 \) implies that \( s_1 \diamond s'_1 = s'_1 \diamond s_1 = s_1 \). Similarly, for \( s_2 \sim s'_2 \), \( s_2 \diamond s'_2 = s'_2 \diamond s_2 = s_2 \)

\[
\Rightarrow (s_1 \diamond s_2) \diamond (s'_1 \diamond s'_2) \\
= (s_1 \diamond (s_2 \diamond s'_1)) \diamond s'_2 \\
= (s_2 \diamond s'_1) \diamond s'_2 \\
= s_2 \diamond (s'_1 \diamond s'_2) \\
= (s'_1 \diamond s'_2)
\]

\[
(s'_1 \diamond s'_2) \diamond (s_1 \diamond s_2) \\
= ((s'_1 \diamond s'_2) \diamond s_1) \diamond s_2 \\
= (s'_1 \diamond (s'_2 \diamond s_1)) \diamond s_2 \\
= s'_1 \diamond ((s'_2 \diamond s_1) \diamond s_2) \\
= (s'_2 \diamond s_1) \diamond s_2 \\
= s'_2 \diamond (s_1 \diamond s_2) \\
= (s_1 \diamond s_2)
\]

Hence, ~ is a congruence equivalence relation.

A.4 Isomorphism of \( \bar{\Sigma_i}^* \) and \( P\Sigma \)

The congruence equivalence relation ~ leads directly to the quotient monoid of \( (\Sigma_i^*, \diamond, \Lambda) \), denoted by \( \Sigma_i/\sim \), or alternatively by \( (\bar{\Sigma_i}^*, \diamond', \Lambda') \). The behaviour of the \( \diamond' \) operator is inherited from the behaviour of the \( \diamond \) operator, the former acts on equivalence classes, whereas the latter acts on arbitrary elements in \( \Sigma_i^* \).

\[
\bar{p} \diamond' \bar{q} = \overline{p \diamond q}
\]

Theorem A.3 \( (\bar{\Sigma_i}^*, \diamond', \Lambda') \) is a commutative monoid.

Proof

It is clear from the properties of congruence equivalence relations that \( (\bar{\Sigma_i}^*, \diamond', \Lambda') \) is a
monoid, i.e., the quotient monoid derived from the congruence equivalence relation ∼.
Thus, it is required to demonstrate that \( (\bar{\Sigma}^*, \ast', \Lambda') \) is a commutative monoid.

It is required to show that for \( s_1, s_2 \in \bar{\Sigma}^* \), that \( s_1 \ast' s_2 = s_2 \ast' s_1 \).

Let \( s_1, s_1' \in \bar{\Sigma}_1 \), then \( s_1 \sim s_1' \).
Let \( s_2, s_2' \in \bar{\Sigma}_2 \), then \( s_2 \sim s_2' \).
Since \( \sim \) is a congruence, \( s_1 \ast s_2 \sim s_1' \ast s_2' \).
It is required to show that \( s_1 \ast s_2 \sim s_2 \ast s_1 \sim s_1' \ast s_2' \sim s_2' \ast s_1' \).

\[
(s_1 \ast s_2) \ast (s_2 \ast s_1) \\
= s_1 \ast (s_2 \ast s_2) \ast s_1 \\
= s_1 \ast (s_2 \ast s_1) \\
= (s_2 \ast s_1)
\]

\[
(s_2 \ast s_1) \ast (s_1 \ast s_2) \\
= s_2 \ast (s_1 \ast s_1) \ast s_2 \\
= s_2 \ast (s_1 \ast s_2) \\
= (s_1 \ast s_2)
\]

Thus \( (s_1 \ast s_2) \sim (s_2 \ast s_1) \)

\[
\Rightarrow s_1 \ast s_2 \sim s_2 \ast s_1 \sim s_1' \ast s_2' \sim s_2' \ast s_1' \\
\Rightarrow s_1 \ast s_2 = s_2 \ast s_1 \\
\]
By definition, \( s_1 \ast' s_2 = \overline{s_1 \ast s_2} \)
\[
\Rightarrow s_1 \ast' s_2 = \overline{s_1 \ast s_2} = \overline{s_2 \ast s_1} = \overline{s_2 \ast' s_1} = s_1 \ast' s_2, \text{ as required.}
\]

Hence, \( (\bar{\Sigma}^*, \ast', \Lambda) \) is a commutative monoid.

**Theorem A.4** \( (\bar{\Sigma}^*, \ast', \Lambda') \) is isomorphic to \((\mathcal{P}\Sigma, \cup, \emptyset)\), i.e., \( (\bar{\Sigma}^*, \ast', \Lambda') \cong (\mathcal{P}\Sigma, \cup, \emptyset) \).

**Proof**
By definition, a homomorphism from the monoid \((M, \oplus, u)\), to the monoid \((P, \otimes, v)\) is a map \(h : M \mapsto P\) such that:

\[
h(m_1 \oplus m_2) = h(m_1) \otimes h(m_2) \quad (A.4)
\]

\[
h(u) = v \quad (A.5)
\]

163
The proof of the isomorphism result is achieved by defining a map \( h \) from \( \bar{\Sigma}_i^* \) to \( \mathcal{P}\Sigma \). It will be demonstrated that \( h \) is, in fact, a homomorphism, and in fact, \( h \) is an isomorphism. The definition of \( h \) is given in terms of a particular element in the equivalence class; in order for the definition to be sound, the definition must yield precisely the same result, irrespective of the chosen element from the equivalence class. The definition of \( h \) follows:

\[
h(\bar{s}) \triangleq \text{elems}(s_1) \quad s_1 \in \bar{s}
\]

It is clear from the definition of \( h(\bar{s}) \) that it is independent of the particular element chosen from the equivalence class; this follows since every element in the equivalence class is a permutation sequence of a unique sequence, and consequently contains exactly the same elements.

The fact that \( h \) is a homomorphism follows since:

\[
h(\bar{s}_1 \diamond \bar{s}_2) = h(s_1 \diamond s_2) = \text{elems}(s_1 \diamond s_2)
\]

\[
\text{elems}(s_1 \diamond s_2) = \text{elems}(s_1) \cup \text{elems}(s_2)
\]

Clearly, \( h(\bar{s}_1) = \text{elems}(s_1) \) and \( h(\bar{s}_2) = \text{elems}(s_2) \) thus it follows:

\[
h(\bar{s}_1) \cup h(\bar{s}_2) = \text{elems}(s_1) \cup \text{elems}(s_2))
\]

\[
h(\bar{s}_1 \diamond \bar{s}_2) = h(\bar{s}_1) \cup h(\bar{s}_2)
\]

\[
h(\Lambda') = \emptyset
\]

Thus \( h \) is a homomorphism, in fact, \( h \) is an epimorphism. This follows since given a subset \( S \subseteq \Sigma \), and suppose \( S = \{\sigma_1, \sigma_2, ..., \sigma_n\} \), then \( \text{elems}(s) = S \) where \( s \) by \( s = \langle \sigma_1 \circ \sigma_2 \circ ... \circ \sigma_n \rangle \). This implies that \( h(s) = S \), and since \( S \) is arbitrary, it follows that \( h \) is onto.

In order to complete the proof it is required to demonstrate that \( h \) is one to one:

Suppose \( h(\bar{s}_1) = h(\bar{s}_2) \) then

\( \text{elems}(s_1) = \text{elems}(s_2) \), where \( s_1 \in \bar{s}_1 \) and \( s_2 \in \bar{s}_2 \).

Thus \( s_1 \diamond s_2 = s_2 \diamond s_1 \) and \( s_2 \diamond s_1 = s_1 \)

\( \Rightarrow s_1 \sim s_2 \)

\( \Rightarrow \bar{s}_1 = \bar{s}_2 \)

Thus \( h \) is one to one and \( h \) is an isomorphism of \( (\bar{\Sigma}_i^*, \diamond', \Lambda') \) and \( (\mathcal{P}\Sigma, \cup, \emptyset) \), i.e.,

\( (\bar{\Sigma}_i^*, \diamond', \Lambda') \cong (\mathcal{P}\Sigma, \cup, \emptyset) \).

164
Conclusion A.2 (Canonical Unique Sequence) Theorem A.4 demonstrates that for all practical purposes the structure \((\mathcal{P} \Sigma, \cup, \emptyset)\) is identical with the quotient monoid of unique sequences \((\Sigma^*_a, \diamond', \Lambda)\) under unique sequence concatenation. This fact allows an alternate representation of a set to be achieved via unique sequences; in particular, a canonical unique sequence from the equivalence class representing the set to be chosen to represent the set.

A.5 Unique Sequences and \((\mathcal{P} \Sigma, \cap, \Sigma)\)

The previous section proved that the power set of \(\Sigma\) under the set union operation, may be represented via unique sequences, via a specialized concatenation operator. Naturally, it is desirable to represent set intersection by a specialized sequence restriction operation. In order to motivate the definition of the \(\bigodot\) operation, an example which illustrates the expected behaviour of the \(\bigodot\) operation is presented.

\[
\langle a, b, c \rangle \bigodot \langle b, d \rangle \triangleq \langle b \rangle \\
\langle a, b, c \rangle \bigodot \langle d, e \rangle \triangleq \Lambda \\
\langle a, b, c \rangle \bigodot \langle c, a \rangle \triangleq \langle c, a \rangle \\
\langle c, a \rangle \bigodot \langle a, b, c \rangle \triangleq \langle a, c \rangle
\]

The \(\bigodot\) operation is defined on \(\Sigma^*_a\) as follows:

\[
(\langle e \rangle \setminus \sigma) \bigodot \tau \triangleq \sigma \bigodot (\langle e \rangle \setminus \sigma \bigodot \tau) \quad \text{if } \neg \chi[e][\tau] \\
\Lambda \bigodot \sigma \triangleq \Lambda = \sigma \bigodot \Lambda
\]

Theorem A.5 \((\Sigma^*_a, \bigodot)\) forms a non commutative semi-group.

Proof

Closure is clear from the constructive definition, as the resultant sequence formed from \(s_1 \bigodot s_2\) is a subsequence of \(s_1\). The fact that the \(\bigodot\) operation is non-commutative follows from the example presented above. It may be thought that the sequence \(\Sigma_{(,)} = \langle \sigma_1, \sigma_2, \ldots, \sigma_n \rangle\), where \(\Sigma = \{\sigma_1, \ldots, \sigma_n\}\) is the identity element under the \(\bigodot\) operation. However, \(\langle a, b, c \rangle \bigodot \langle c, a \rangle = \langle c, a \rangle \neq \langle a, c \rangle = \langle c, a \rangle \bigodot \langle a, b, c \rangle\) is a counter example to this hypothesis. An inductive argument similar to Theorem A.1 demonstrates that the associative property holds.

The isomorphism between \((\mathcal{P} \Sigma, \cup, \emptyset)\) and \((\Sigma^*_a, \diamond', \Lambda')\) was achieved by defining a congruence equivalence relation, and forming the quotient monoid. The approach for \((\mathcal{P} \Sigma, \cap, \Sigma)\) is similar, the equivalence relation being defined by:
\[ s_1 \sim s_2 \text{ if } s_1 \Diamond s_2 = s_1 \text{ and } s_2 \Diamond s_1 = s_2 \]

where \( s_1, s_2 \in \Sigma^*_i \), and we have the corresponding theorems, as before.

**Theorem A.6 (Congruence Equivalence Relation)** The relation defined by \( \sim \) is a congruence equivalence relation.

**Proof**
This is similar to Theorem A.2.

**Theorem A.7** \((\Sigma^*_i, \Diamond', \Sigma'_i)\) is a commutative monoid.

**Proof**
This is similar to Theorem A.3.

**Theorem A.8** \((\Sigma^*_i, \Diamond', \Sigma'_i)\) is isomorphic to \((P\Sigma, \cap, \Sigma)\), i.e., \((\Sigma^*_i, \Diamond', \Sigma'_i) \cong (P\Sigma, \cap, \Sigma)\).

**Proof**
This is similar to Theorem A.4.

**Conclusion A.3** Thus we have shown that \((P\Sigma, \cap, \Sigma)\) is essentially identical with the quotient monoid of unique sequences \((\Sigma^*_i, \Diamond', \Sigma'_i)\) under unique sequence concatenation. This allows us an alternate means of implementing sets via unique sequences.

**Unique Sequences and Set complements**

Given \( A \) a subset of \( \Sigma \), \( \bar{A} \) or \( A^c \) denotes the complement of \( A \). It is important to develop an equivalent representation of the set complement operator for unique sequences. Thus given a sequence \( \tau \in \Sigma^* \), then we consider \( E_\tau = \text{elems} \tau, E^c_\tau = \Sigma \setminus E_\tau \). Consider a sequence \( \tau^c = \langle \sigma_1, ..., \sigma_r \rangle \) where \( E^c_\tau = \{ \sigma_1, ..., \sigma_r \} \).

Then it is clear from the definition of \( \Diamond \) and \( \Diamond' \) that \( \tau \Diamond \tau^c = \Lambda = \tau^c \Diamond \tau \). However, we note also that \( \tau \Diamond' \psi = \psi \Diamond \tau = \Lambda \) if \( \psi \) is any permutation sequence of \( \tau^c \), if \( \psi \sim \tau^c \). Thus we get \( Cl(\tau) \Diamond' Cl(\tau^c) = \Lambda' \), and \( Cl(\tau) \Diamond Cl(\tau^c) = Cl(\Sigma'_i) \).

**Comment A.1 (Canonical sequence representation of a subset)** A set \( \Sigma \) and any subset \( S \) of \( \Sigma \) is represented by an equivalence class of unique sequences, where each sequence in the equivalence class is a permutation of any other in the class. It is easier to work with a single sequence rather than multiple, and this may be achieved by choosing the canonical sequence from the class. This requires \( \Sigma \) to be ordered (we can define an order if necessary) and the canonical sequence from each class is the unique sequence which is in ascending order in the class.
A.6 Canonical Representation of Sequences

The canonical sequence representation of a set $A$, denoted by $A_{(\cdot)}$, is a more concrete representation of the set, and is closer to implementation. The retrieval function from the concrete to the abstract is given by:

$$R_{1,0} s_{(\cdot)} \triangleq \text{elems } s_{(\cdot)} : s_{(\cdot)} \in \Sigma^*$$  \hspace{1cm} (A.8)

The miscellaneous set theoretical operations in standard set theory require corresponding definitions in the canonical sequence representation. We present the definition of $S_1 \subseteq_{(\cdot)} S_2$ for the canonical representation as follows:

$$A_{(\cdot)} \subseteq_{(\cdot)} B_{(\cdot)} \triangleq \text{elems } A_{(\cdot)} \subseteq \text{elems } B_{(\cdot)}$$

The associated proof obligation is trivial, and is stated as the following lemma:

**Lemma A.2** $R_{1,0}(A_{(\cdot)}) \subseteq R_{1,0}(B_{(\cdot)}) = A_{(\cdot)} \subseteq_{(\cdot)} B_{(\cdot)}$

**Proof**

This is true by definition of $\subseteq_{(\cdot)}$.

**Question A.4** Suppose $A_{(\cdot)}, B_{(\cdot)}$ are the respective canonical representations of the sets $A$ and $B$. What is the canonical representation of $A \cap B$, and is it equal to $A_{(\cdot)} \Diamond B_{(\cdot)}$?

$A_{(\cdot)} \Diamond B_{(\cdot)}$ gives the canonical representation of $A \cap B$. It is clear that the former yields at worst a permutation of the canonical representation of $A \cap B$, thus if the resultant sequence is ordered in ascending order we are done. An examination of the definition of $\Diamond$ in Section A.5 shows that if both sequences are ordered then the resultant sequence is ordered and contains exactly those elements both sequences have in common. This is clear from the definition of $\Diamond$, and may be made more rigorous by an inductive argument.

$$\langle \varepsilon \rangle \cap \sigma \Diamond \tau \triangleq \begin{cases} \sigma \Diamond \tau & \text{if } -\chi[\varepsilon][\tau] \\ \langle \varepsilon \rangle \cap (\sigma \Diamond \tau) & \text{otherwise.} \end{cases}$$  \hspace{1cm} (A.9)

$$A \Diamond \sigma \triangleq A = \sigma \Diamond A$$  \hspace{1cm} (A.10)

**Question A.5** Suppose $A_{(\cdot)}, B_{(\cdot)}$ are the respective canonical representations of the sets $A$ and $B$. What is the canonical representation of $A \cup B$, and is it equal to $A_{(\cdot)} \Diamond B_{(\cdot)}$?

$A_{(\cdot)} \Diamond B_{(\cdot)}$ yields a permutation of the canonical representation of $A \cup B$. This may be seen by the following counter-example which yields a sequence representation of $A \cup B$, which is not ordered.

167
\langle 1, 5, 7 \rangle \diamond \langle 4, 8 \rangle = \langle 1, 5, 7, 4, 8 \rangle \neq \langle 1, 4, 5, 7, 8 \rangle

This presents a problem as it is desirable that the union of two canonical sequences, be the canonical sequence of the union. One approach to solve this is to amend the definition of the $\diamond$ operation in section A.2, which will ensure that the resultant sequence is ordered if both sequences are ordered. This is achieved by the following definition:

\[
\langle e \rangle \smallsetminus (\langle e' \rangle \smallsetminus \tau) \quad \begin{cases} 
\langle e \rangle \smallsetminus (\sigma \diamond \tau) & \text{if } e = e' \\
\langle e \rangle \smallsetminus (\sigma \diamond (\langle e' \rangle \smallsetminus \tau)) & \text{if } e \ll e' \\
\langle e' \rangle \smallsetminus (\langle e \rangle \smallsetminus \sigma) \diamond \tau & \text{if } e' \ll e
\end{cases}
\]

(A.11)

\[
\Lambda \diamond \sigma \triangleq \sigma = \sigma \diamond \Lambda
\]

(A.12)

Comment A.2 This specialized definition of $\diamond$ behaves similarly to the original, with the exception that both sequences are sorted then it is guaranteed that the resultant sequence is sorted. Its effect when working with arbitrary elements in $\Sigma^*$ is that it produces a permutation of whatever the original produces, however there is no guarantee that this permutation is ordered. Thus the isomorphism results of for $(P\Sigma, \cup, \emptyset) = (\tilde{\Sigma}^*, \diamond, \Lambda)$ remain true.

Comment A.3 We note in fact when this operation is applied to unique sequences sorted in ascending order, it is functionally equivalent to a merge of two sorted buffers, as found in several merge-sort algorithms.

Question A.6 Suppose $A_{(\cdot)}$ is the canonical sequence representation of $A$, what is the canonical sequence representation of $A^c$?

A solution to this question is partially sketched in section A.5, where the problem of determining the complement of a sequence $A_{(\cdot)} \in \Sigma^*$ is considered. It is noted that a solution is formed by constructing a unique sequence $A^c_{(\cdot)}$ such that the sequence contains each element in $\Sigma \setminus A$. Obviously, there are multiple unique sequences satisfying these conditions.

Then the canonical sequence is the unique ordered sequence in the equivalence class of $A^c_{(\cdot)}$. However, we may wish to construct the canonical sequence representation of $A^c$, i.e., $A^c_{(\cdot)}$ explicitly.

This is achieved by the $Compl\_Seq$ operation, which produces the complement of the canonical representation of a set. The complement of the sequence is then assured to be in canonical form, provided that the sequence itself is in canonical form.
\[ Cmpl_{\text{Seq}} : \Sigma_1^* \mapsto \Sigma_1^* \]
\[ Cmpl_{\text{Seq}}[A_{\langle \cdot \rangle}] \triangleq Cmpl_{\text{Seq}}[\Sigma_{\langle \cdot \rangle}, A_{\langle \cdot \rangle}] \Lambda \]

where \( \Sigma_{\langle \cdot \rangle} \) is the canonical sequence representation of the set \( \Sigma \). The predicate \( IsCnf \) determines if a sequence \( \psi \in \Sigma_1^* \) is in canonical form. The precondition for the \( Cmpl_{\text{Seq}} \) operation is then specified, using this predicate, follows:

\[
\begin{align*}
IsCnf[\Lambda] & \triangleq \text{TRUE} \\
IsCnf[\langle \sigma \rangle] & \triangleq \text{TRUE} \\
IsCnf[\langle \sigma_1, \sigma_2 \rangle \prec \tau] & \triangleq \\
(\sigma_1 \ll \sigma_2) \land IsCnf[\langle \sigma_2 \rangle \prec \tau]
\end{align*}
\]

\[
\text{pre}_{\text{Cmpl}_{\text{Seq}}} : \Sigma_1^* \mapsto \mathcal{B} \\
\text{pre}_{\text{Cmpl}_{\text{Seq}}}[\psi] \triangleq \\
IsCnf[\psi] \\
\land \text{elems } \psi \subseteq \Sigma
\]

The \( Cmpl_{\text{Seq}}[\Sigma_{\langle \cdot \rangle}, A_{\langle \cdot \rangle}] \psi \) operation, constructs the complement of the sequence by going through every element in \( \Sigma_{\langle \cdot \rangle} \), and removing any elements which are in \( A_{\langle \cdot \rangle} \). It is defined as follows:

\[
\begin{align*}
Cmpl_{\text{Seq}} : (\Sigma_1^* \times \Sigma_1^*) \mapsto \Sigma_1^* \mapsto \Sigma_1^* \\
Cmpl_{\text{Seq}}[\Lambda, \Lambda] \psi & \triangleq \psi \\
Cmpl_{\text{Seq}}[\tau, \Lambda] \psi & \triangleq (\psi \prec \tau) \\
Cmpl_{\text{Seq}}[\langle \sigma \rangle \prec \tau, \langle a \rangle \prec T] \psi & \triangleq \\
& \quad \sigma \ll a \\
& \quad \mapsto Cmpl_{\text{Seq}}[\tau, \langle a \rangle \prec T](\psi \prec \langle \sigma \rangle) \\
& \quad \sigma = a \\
& \quad \mapsto Cmpl_{\text{Seq}}[\tau, T] \psi
\end{align*}
\]

**Comment A.4** A clearer and more natural definition of the \( Cmpl_{\text{Seq}} \) operation is easily obtained from the above, and is described below.

\[ \text{Cmpl}_{\text{Seq}}[A_{\langle \cdot \rangle}] \triangleq [\text{elems } A_{\langle \cdot \rangle}] \Sigma_{\langle \cdot \rangle} \]

It is required to demonstrate that the concrete complement operation is a refinement of the abstract set theoretical complement operation. This is stated in the following lemma.

**Lemma A.3** \( (\mathcal{R}_{1,0} A_{\langle \cdot \rangle})^c = \mathcal{R}_{1,0} \circ \text{Cmpl}_{\text{Seq}}[A_{\langle \cdot \rangle}] \]

169
Proof
\[(R_{1,0} A_{i,j})^c\]
\[= (\text{elems } A_{i,j})^c\]
\[= \Sigma \setminus (\text{elems } A_{i,j})\]
\[= \text{elems } \circ \llbracket \text{elems } A_{i,j} \rrbracket \Sigma_{\cdot i}\]
\[= R_{1,0} \circ \text{Seq}[A_{i,j}]\]

A.7 Multiset Representation via Sequences

Multisets and their applications have been presented elsewhere [7]. The standard representation of multisets in $VDM$ is via bag operations; the bag maintains an accurate record of the number of occurrences of each element in the bag. However, having examined the relationship between sets and unique sequences, it is natural to consider a representation of multisets via sequences. The key difference for multiset representation via sequences in contrast with representation of standard sets via unique sequences, is that more general sequences are appropriate for multiset representation. In fact, sequences with duplicate elements are the norm for multiset representation.

Suppose $\Sigma$ is any alphabet, and $(\Sigma^*, \cdot, \Lambda)$ be the free monoid. In multiset representation, it is important that there should be no distinction in the ordering of the elements. This relationship is exhibited by the equivalence relation $\sim$, defined by:

\[s_1 \sim s_2 \triangleq MPerm[s_1] s_2\]

where $MPerm$ is a predicate which determines if $s_1$ is a permutation of $s_2$. It is defined as follows:

\[MPerm : \Sigma^* \mapsto \Sigma^* \mapsto \mathbb{B}\]
\[MPerm[\Lambda] \sim \Lambda \triangleq \text{TRUE}\]
\[MPerm[\Lambda] s \triangleq \text{FALSE}\]
\[MPerm[s] \sim \Lambda \triangleq \text{FALSE}\]
\[MPerm[\langle \sigma \rangle \cdot \tau] s \triangleq \neg \chi[\sigma] s\]
\[\mapsto \text{FALSE}\]

Let $s = s_1 \sim \langle \sigma \rangle \cdot \tau \sim s_2$ in
\[\mapsto MPerm[\tau](s_1 \sim s_2)\]
Then we claim $\sim$ is an equivalence relation which is in fact a congruence, first we state and prove some elementary lemmas.

**Lemma A.4** $s_1 \sim s_2 \Rightarrow |s_1| = |s_2|.$

**Proof (By Induction)**
The result is clearly true for $|s_1| = 0$, i.e., $s_1 = \Lambda$. Then by definition $s_1 \sim s_2 \Rightarrow s_2 = \Lambda$, i.e., $|s_1| = |s_2|$ and the basis case is established.
Suppose the lemma is true for all $s_1$ such that $|s_1| \leq k$, and consider $s_1 = \langle \sigma \rangle \ominus \tau$, where $|\tau| = k$. Then as $s_1 \sim s_2$, it follows that $\sigma \in s_2$ and $s_2 = s \ominus \langle \sigma \rangle \ominus s'$. By definition, $s_1 \sim s_2$ iff $\tau \ominus s \ominus s'$; however, since $|\tau| = k$ it follows that $|\tau| = |s \ominus s'|$. Thus $|s_1| = |s_2|$ as required.

**Lemma A.5** $s_1 \sim s_2 \Rightarrow \text{elems}(s_1) = \text{elems}(s_2)$.

**Proof (By contradiction)**
Suppose not, then $\exists u \in s_1$ such that $\neg u \in s_2$, or $\exists w \in s_2$ such that $\neg w \in s_1$. For the former, the definition of the $MPerm$ operation states that if $s_1 \sim s_2$ then $u \in s_2$. For the latter case, we have for each $u \in s_1$ that $u \in s_2$. Thus given that $w \in s_2$ such that $\neg w \in s_1$, it follows by the definition of the $MPerms$ operation that $s_1 \sim s_2$ is equivalent to in this case $\Lambda \sim W$, where $w \in W$; however, this is impossible, by the definition of the $MPerms$ operation.

**Lemma A.6** $s_1 \sim s_2$ and $u \in s_1$, then $\text{Occ}[u]s_1 = \text{Occ}[u]s_2$ where the Occ operator determines the frequency, i.e., the number of occurrences of an element $u$ in a sequence.

**Proof (By Induction)**
The basis case is $|s_1| = 1$, and $s_1 = \langle u \rangle = s_2$. Clearly, $\text{Occ}[u]s_1 = \text{Occ}[u]s_2 = 1$. Suppose the lemma is true whenever $|s_1| \leq k$, and consider $s_1 = \langle \sigma \rangle \ominus \tau$. Since $s_1 \sim s_2$, it follows that $s_2 = s \ominus \langle \sigma \rangle \ominus s'$. If $u \neq \sigma$, then $\text{Occ}[u]s_1 = \text{Occ}[u]\tau = \text{Occ}[u](s \ominus s')$, since $\tau \ominus s \ominus s'$, and $|\tau| = k = |s \ominus s'|$.

If $u = \sigma$, then it follows that $\text{Occ}[u]s_1 = 1 + \text{Occ}[u]|\tau| = 1 + \text{Occ}[u]|s \ominus s'|$, since $\tau \ominus s \ominus s'$, and $|\tau| = k$. Thus $\text{Occ}[u]s_1 = \text{Occ}[u]s_2$ as required.

**Theorem A.9** $s_1 \sim s_2$ if and only if $|s_1| = |s_2|$, and $\text{elems}(s_1) = \text{elems}(s_2)$, and $\text{Occ}[u]s_1 = \text{Occ}[u]s_2$, for all $u \in \text{elems}(s_1)$.

Lemma A.4, A.5, A.6 have demonstrated that $s_1 \sim s_2 \Rightarrow |s_1| = |s_2|$, and $\text{elems}(s_1) = \text{elems}(s_2)$, and $\text{Occ}[u]s_1 = \text{Occ}[u]s_2$. The objective here is to prove the converse of the theorem; i.e., it is required to demonstrate that if the properties in the theorem hold, then $s_1 \sim s_2$. This result is proved by the contra positive. That is, $\neg(s_1 \sim s_2) \Rightarrow$ one of the
above properties fail.

An inductive argument is used; the basis case is \(|s_1| = 1\); for this case it is clear that \(s_1 \not\in s_2 \Rightarrow s_1 = \langle u \rangle\), and \(v \in s_2\), where \(u \not\in v\). Thus, the property \(\text{elems}(s_1) = \text{elems}(s_2)\) fails for the basis case.

Suppose the converse of the theorem is true for all \(|s_1| \leq k\), and consider \(s_1 = \langle \sigma \rangle \cap \tau\) where \(|\tau| = k\). Then suppose \(\neg(s_1 \sim s_2)\), then If \(\sigma \not\in s_2\), it is clear that the property \(\text{elems}(s_1) = \text{elems}(s_2)\) fails.

If \(\sigma \in s_2\), then \(s_2 = s \cap \langle \sigma \rangle \cap s'\), and thus \(\neg(\tau \sim s \cap s')\). By the inductive hypothesis, since \(|\tau| = k\) that one of the three properties above fail for \(\tau\) and \(s \cap s'\). By the construction of \(s_1\) and \(s_2\) from \(\tau\) and \(s \cap s'\) respectively, it follows that at least one of the properties fail for \(s_1\) and \(s_2\).

**Theorem A.10 (Equivalence Relation)** The \(MPerm\) relation, \(\sim\), is an equivalence relation, and is, in fact, a congruence.

1. **Reflexive** It is required to show that \(s_1 \sim s_1\); this result is obvious since \(s_1\) is clearly a permutation of itself. The argument may be made more rigorous by induction, since if \(s_1 = \Lambda\), then by definition \(s_1 \sim s_1\), and the basis case is established. Inductively, assume the property is true for \(|s_1| \leq k\), and consider \(s_1 = \langle \sigma \rangle \cap s'_1\) where \(|s'_1| = k\). Then \(s_1 \sim s_1\) iff \(s'_1 \sim s'_1\); clearly, this is true by the inductive hypothesis, since \(|s'_1| = k\), thus the reflexive property holds.

2. **Symmetric** It is required to show that \(s_1 \sim s_2 \Rightarrow s_2 \sim s_1\), which is clear since \(s_1\) is a permutation of \(s_2\) if and only if \(s_2\) is a permutation of \(s_1\). Lemma A.4 has demonstrated that \(s_1 \sim s_2 \Rightarrow |s_1| = |s_2|\); similarly, Lemma A.5 proves that \(\text{elems}(s_1) = \text{elems}(s_2)\), and Lemma A.6 shows that \(\text{Occ}[u]s_1 = \text{Occ}[u]s_2\), for all \(u \in \text{elems}(s_1)\). Clearly, by Theorem A.9 \(s_2 \sim s_1\).

3. **Transitive** It is required to show that if \(s_1 \sim s_2\), and \(s_2 \sim s_3\) then \(s_1 \sim s_3\). By Lemma A.4, it follows that \(|s_1| = |s_2|\) and \(|s_2| = |s_3|\), and consequently \(|s_1| = |s_3|\). Furthermore, by Lemma A.5 \(\text{elems}(s_1) = \text{elems}(s_2)\), and \(\text{elems}(s_2) = \text{elems}(s_3)\), thus \(\text{elems}(s_1) = \text{elems}(s_3)\). Finally, \(\text{Occ}[u]s_1 = \text{Occ}[u]s_2\), and \(\text{Occ}[u]s_2 = \text{Occ}[u]s_3\), thus \(\text{Occ}[u]s_1 = \text{Occ}[u]s_3\) and thus \(\sim\) is transitive.

**Lemma A.7 (Congruence Property)** The equivalence relation \(\sim\) is a congruence.

We must show that if \(s_1 \sim s_2\) and \(s'_1 \sim s'_2\), then \(s_1 \cap s'_1 \sim s_2 \cap s'_2\). We have \(|s_1| = |s_2|\) and \(|s'_1| = |s'_2|\). Thus \(|s_1 \cap s'_1| = |s_1| + |s'_1| - |s_2| + |s'_2| = |s_2 \cap s'_2|\).

172
Similarly, we have \( \text{elems} (s_1) = \text{elems} (s_2) \), and \( \text{elems} (s'_1) = \text{elems} (s'_2) \), thus \( \text{elems} (s_1 \sim s'_1) = \text{elems} (s_1) \cup \text{elems} (s'_1) = \text{elems} (s_2) \cup \text{elems} (s'_2) = \text{elems} (s_2 \sim s'_2) \).

Finally, given \( u \in s_1 \sim s'_1 \), we must show \( \text{Occ}[u](s_1 \sim s'_1) = \text{Occ}[u](s_2 \sim s'_2) \). This is seen by a case analysis:

1. \( u \in s_1 \land u \in s'_1 \). Then \( \text{Occ}[u] s_1 = \text{Occ}[u] s_2 \) and \( \text{Occ}[u] s'_1 = \text{Occ}[u] s'_2 \), thus we get
   \[
   \text{Occ}[u](s_1 \sim s'_1) = \text{Occ}[u] s_1 + \text{Occ}[u] s'_1 = \text{Occ}[u] s_2 + \text{Occ}[u] s'_2 = \text{Occ}[u](s_2 \sim s'_2).
   \]
2. \( u \not\in s_1 \land u \in s'_1 \). This is similar to Case 1, note \( \text{Occ}[u] s_1 = 0 \).
3. \( u \in s_1 \land u \not\in s'_1 \). This is similar to Case 1, note \( \text{Occ}[u] s'_1 = 0 \).

**Question A.7 (Equivalence Classes of \( \Sigma^* \))** What are the equivalence classes of \( \Sigma^* \) under the equivalence relation \( \sim \)?

It is clear from Theorem A.9 that the equivalence class of a sequence \( s \) in \( \Sigma^* \) consists precisely of all those sequences which have exactly the same elements as \( s \), and are exactly the same length as \( s \); finally, the frequency of each element \( u \) in the sequence \( s \) must be precisely the same as the frequency of \( u \), for every sequence \( s' \) in the equivalence class of \( s \). Thus a sequence \( s' \) is in the equivalence class of \( s \) if and only if it is a permutation of \( s \), the permutation being a more sophisticated form as the sequence may contain duplicate elements.

These results have been demonstrated by Lemma A.4, Lemma A.5, Lemma A.6 and Theorem A.9. In effect, the equivalence class of \( s \) consists of an arbitrary sequence constructed from the multi-set \([u_1 : r_1, u_2 : r_2, ..., u_k : r_k]\), or equivalently, the well known bag structure \((\Sigma \mapsto N_1)\). Thus the equivalence class of \( s \) is essentially identical with the multi-set representation of \( s \).

**Question A.8** How many elements are there in the equivalence class of \( s \) under \( \sim \), where \( Cl(s) \) is the collection of all sequences of the form \([u_1 : r_1, u_2 : r_2, ..., u_k : r_k]\)?

We let \( n = r_1 + r_2 + ... + r_k \), where each \( r_i \) gives the occurrence in \( s \) of each \( u_i \). The result is identical to problems in permutations and combinations, the problem being to determine the number of permutations of an object of size \( n \) containing \( r_1 \) of type \( u_1 \), \( r_2 \) of type \( u_2 \), etc., and is given by the formula:

\[
\frac{n!}{r_1!r_2!...r_k!}
\]

**Theorem A.11** The Quotient Monoid \( \Sigma^*/\sim \) is isomorphic to the multiset defined over \( \Sigma \), i.e., \((\Sigma \mapsto N_1)\).

173
Proof (Informal Sketch)

The proof involves constructing a map from $\Sigma^*/ \sim$ to $(\Sigma \mapsto N_1)$, such that the equivalence class of $s$ is associated with exactly one map $\psi$ in $(\Sigma \mapsto N_1)$. Furthermore, any map $\psi \in (\Sigma \mapsto N_1)$ is associated with exactly one equivalence class of $\Sigma^*$. The precise statement of the isomorphism is:

$$(\Sigma^*/ \sim, \sim', \Lambda') \cong (X \mapsto N_1, \oplus, \theta)$$

A.8 Refinement of Sequences into Arrays

This section considers the problem of the implementation of sequences into arrays is considered. The former is an abstract structure and is refined into a mathematical array, i.e., $(N \mapsto \Sigma)$. Constraints are placed on the partial mapping to stipulate that the sequence is finite and contiguous. These constraints are expressed by the following invariant.

$$inv_{\text{Seq}} : \Sigma^* \mapsto B$$
$$inv_{\text{Seq}}[\psi] \triangleq$$
$$\psi = \theta$$
$$\lor (||\text{dom } \psi|| < \infty) \land (\text{dom } \psi = \{1..||\text{dom } \psi||\})$$

Sequence concatenation $\sim$ is then refined to the Join operation, the Rem operation performs removal, and is the more concrete form of $\ll$. Rst is the restriction operator, corresponding to $\llcorner$. Hd and Tail are amended accordingly. These concrete representations are more cumbersome than the abstract sequence definition, as for example, the removal of an element from the concrete form requires reformatting, to ensure that the resulting structure is contiguous and satisfies the invariant.

Comment A.5 The Rem and Rst operations are unique to sequences, and in general do not apply to arrays, which generally reflects a static collection of cells

The retrieval function $R_{1,0}$ is given by:

$$R_{1,0}[\psi] \triangleq \begin{cases} 
\psi = \theta & \mapsto \Lambda \\
\rightarrow \langle \psi(1) \sim \ldots \sim \psi(|\text{dom } \psi|) \rangle \text{ otherwise} \end{cases} \quad (A.13)$$

For each refined operation we have the associated proof obligation that the commuting diagram property holds, e.g., for the Join operation we must show:

$$R_{1,0}(\psi_1) \sim R_{1,0}(\psi_2) = R_{1,0} \circ \text{Join}[\psi_1]\psi_2 \quad (A.14)$$
The \textit{Join} operation is sketched as follows.

$$\text{Join} : (N_1 \mapsto \Sigma) \mapsto (N_1 \mapsto \Sigma) \mapsto (N_1 \mapsto \Sigma)$$

$$\text{Join}[\theta] \psi_2 \triangleq \psi_2$$

$$\text{Join}[\psi_1] \quad \theta \triangleq \psi_1$$

$$\text{Join}[\psi_1] \psi_2 \triangleq$$

$$\psi' \mapsto (\text{Off}_{\text{dom}\psi_1} \mapsto \mathcal{I}) \psi_2$$

$$\mapsto \psi_1 \sqcup \psi'_2$$

The result of \(\text{Off}_n(x)\) is \(x + n\). The effect of the use of \(\text{Off}_n\) is to translate the map to a domain at offset \(n\) from the original map.

**Lemma A.8** The \textit{Join} operation is a refinement of the concatenation operation with respect to the concrete representation of sequences, i.e., Eqn A.14 is valid.

**Proof**

Suppose \(\psi_1 = \theta\), then \(\text{Join}[\psi_1] \psi_2 = \psi_2\), and \(\mathcal{R}_{1,0} \psi_1 = \Lambda\). Thus \(\mathcal{R}_{1,0} \circ \text{Join}[\psi_1] \psi_2 = \mathcal{R}_{1,0} \psi_2 = \Lambda \cap \mathcal{R}_{1,0} \psi_2 = \mathcal{R}_{1,0} \psi_1 \cap \mathcal{R}_{1,0} \psi_2\) as desired.

The argument for \(\psi_2 = \theta\) is identical, thus we suppose \(\psi_1 \neq \theta\) and \(\psi_2 \neq \theta\). Then we get:

$$\mathcal{R}_{1,0}(\psi_1) \cap \mathcal{R}_{1,0}(\psi_2)$$

$$= \langle \psi_1(1) \cap \ldots \cap \psi_1(\text{dom}\psi_1) \rangle \cap \langle \psi_2(1) \cap \ldots \cap \psi_2(\text{dom}\psi_2) \rangle$$

$$= \mathcal{R}_{1,0} \circ \text{Join}[\psi_1] \psi_2$$

The last line of the proof follows from the definition of the \textit{Join} operation.

Next it is required to show that the \(Hd_1\) and \(Tail_1\) operations are refinements of the \(Hd_0\) and \(Tail_0\) operations. The \(Hd_1\) operation is sketched as follows:

$$Hd_1 : (N_1 \mapsto \Sigma) \mapsto \Sigma$$

$$Hd_1[\psi] \triangleq \psi(1)$$

**Lemma A.9** The \(Hd_1\) operation is a refinement of the \(Hd_0\) operation.

**Proof**

In order to demonstrate that \(Hd_1\) is a successful refinement of \(Hd_0\), it is required to show that \(Hd_0 \circ \mathcal{R}_{1,0} \psi = Hd_1 \psi\). The preconditions on \(Hd_0\) and \(Hd_1\) ensure that \(\psi \neq \theta\).
\[ H d_0 \circ R_{1,0} \psi \]
\[ = H d_0 \langle \psi(1) \sim \ldots \sim \psi(|\text{dom}\psi|) \rangle \]
\[ = \psi(1) \]
\[ = H d_1 \psi \]

The \textit{Tail}_1 operation is defined as follows:

\[ \text{Tail}_1 : (N_1 \mapsto \Sigma) \mapsto (N_1 \mapsto \Sigma) \]
\[ \text{Tail}_1[\psi] \triangleq \]
\[ \psi' \mapsto \#[1] \psi \]
\[ \mapsto (\text{Off}_{-1} \mapsto I) \psi' \]

\textbf{Lemma A.10} The \textit{Tail}_1 operation is a refinement of the \textit{Tail}_0 operation.

\textbf{Proof}
From the precondition, we have that \( \psi \neq \emptyset \)

\[ \text{Tail}_0 \circ R_{1,0} \psi \]
\[ = \text{Tail}_0 \langle \psi(1) \sim \ldots \sim \psi(|\text{dom}\psi|) \rangle \]
\[ = \langle \psi(2) \sim \ldots \sim \psi(|\text{dom}\psi|) \rangle \]
\[ = \langle \psi'(1) \sim \ldots \sim \psi'(|\text{dom}\psi| - 1) \rangle \]
\[ \text{where } \psi'(i) = \psi(i + 1), \text{ } 1 \leq i \leq |\text{dom}\psi| - 1 \]
\[ = \text{Tail}_1 \psi \]

The last line of the proof follows from the definition of the \textit{Tail}_1 operation.

The \textit{Rem} operation removes all occurrences of a particular set of elements from the sequence. It is defined as follows:

\[ \text{Rem} : \mathcal{P} \Sigma \mapsto (N_1 \mapsto \Sigma) \mapsto (N_1 \mapsto \Sigma) \]
\[ \text{Rem}[S]\psi \triangleq \]
\[ \text{Let } D = \{ n | \psi(n) \in S \} \]
\[ \text{Let } \psi' = \#[D] \psi \text{ in} \]
\[ \mapsto \text{Cut}_\text{Seq}[\psi'] \]

The \textit{Rem} operation has the following elementary properties, which is stated in the following lemma.

\textbf{Lemma A.11} \( \text{dom} \circ \text{Rem}[S] \psi \subseteq \text{dom} \psi \land \text{rng} \circ \text{Rem}[S] \psi \cap S = \emptyset \). Furthermore \( a \in (\text{rng} \psi \setminus S) \Rightarrow a \in \text{rng} \circ \text{Rem}[S] \psi \).
Proof
These properties are immediate from the construction of $\text{Rem}[S]\psi$.

**Lemma A.12** The Rem operation is a refinement of the $\triangleleft$ operation, i.e., $\mathcal{R}_{1,0} \circ \text{Rem}[S]\psi = \triangleleft[S] \circ \mathcal{R}_{1,0}\psi$.

**Proof** (Informal)
\[
\mathcal{R}_{1,0} \circ \text{Rem}[S]\psi \\
= \mathcal{R}_{1,0}\psi' \\
= \langle \sigma_1, \ldots, \sigma_{\text{dom}\psi}\rangle
\]
where we have each $\sigma_i \in (\text{rng}\psi \setminus S)$

\[
\triangleleft[S] \circ \mathcal{R}_{1,0}\psi \\
= \triangleleft[S]\langle \psi_1, \ldots, \psi(|\text{dom}\psi|) \rangle \\
= \langle \sigma_{j_1}, \ldots, \sigma_{j_{k}} \rangle
\]
where we have each $\sigma_{j_i} \in (\text{rng}\psi \setminus S)$

From the construction of the $\text{Rem}[S]\psi$ operation, and these facts, it follows that the commuting diagram property holds.

The $\text{Rst}$ operation restricts a sequence to all occurrences of a particular set of elements in the sequence. It is defined as follows:

\[
\text{Rst} : \mathcal{P}_\Sigma \mapsto (N_1 \mapsto \Sigma) \mapsto (N_1 \mapsto \Sigma)
\]

\[
\text{Rst}[S]\psi \triangleq
\]

Let $R = \{ n | \psi(n) \in S \}$

\[
\psi' \mapsto \triangleleft[R]\psi \\
\mapsto \text{Cut}_{\text{Seq}}[\psi']
\]

The $\text{Rst}$ operation has the following elementary properties, which are stated in the following lemma.

**Lemma A.13** $\text{dom}\ \text{Rst}[S]\psi \subseteq \text{dom}\psi$ and $\text{rng}\ \text{Rst}[S]\psi \subseteq S$. Furthermore $a \in \text{rng}\psi \cap S \Rightarrow a \in \text{rng} \circ \text{Rst}[S]\psi$.

**Lemma A.14** The $\text{Rst}$ operation is a refinement of the $\triangleleft$ operation.

**Proof**
The proof is similar to Lemma A.12.
Refinement of $X \rightarrow Y$ to $(X \times Y)^*$

The abstract data type $(X \rightarrow Y)$ may not be implemented directly in high level languages. One possible approach to implementing this data type is as the sequence $(X \times Y)^*$, where a stipulation is placed on the sequence, to ensure that for any $x \in X$, there is at most one occurrence of $x$ in the sequence. This requirement is stated by the $Inv\_Map$ invariant.

$$Inv\_Map : (X \times Y)^* \rightarrow B$$

$$Inv\_Map[\xi] \triangleq (|\xi| = |\text{elems} \circ \pi_1^* (\xi)|)$$

Using this more concrete representation for maps, it is necessary to provide refinements of the standard map operations, and prove that these operations preserve the invariant. Furthermore, the retrieval function is given by:

$$R_{1,0} \triangleq \theta \tag{A.15}$$

$$R_{1,0}((x, y)) \triangleq [x \mapsto y] \tag{A.16}$$

$$R_{1,0}((x, y) \circ \xi) \triangleq [x \mapsto y] \sqcup R_{1,0}(\xi) \tag{A.17}$$

The first operation we consider is the $\sqcup$ operation, which is refined by the $Add\_Map[]$ operation. The precondition must stipulate that the representations of both maps, i.e., $\xi_1, \xi_2$ are disjoint. It is defined as follows:

$$\text{pre}_\text{Add\_Map} : (X \times Y)^* \times (X \times Y)^* \rightarrow B$$

$$\text{pre}_\text{Add\_Map}[\xi_1, \xi_2] \triangleq (\text{elems} \circ \pi_1^* (\xi_1) \cap \text{elems} \circ \pi_1^* (\xi_2)) = \emptyset$$

$$Add\_Map : (X \times Y)^* \times (X \times Y)^* \rightarrow (X \times Y)^*$$

$$Add\_Map[\xi_1, \xi_2] \triangleq \xi_1 \circ \xi_2$$

There are two proof obligations associated with the operation, the first is that it preserves the invariant, the second that it is a refinement of the $\sqcup$ operation.

**Lemma A.15** $\text{pre}_\text{Add\_Map}[\xi_1, \xi_2] \land \xi = Add\_Map[\xi_1, \xi_2] \Rightarrow Inv\_Map[\xi]$.

**Proof**

From the precondition we have $\text{elems} \circ \pi_1^* (\xi_1) \cap \text{elems} \circ \pi_1^* (\xi_2) = \emptyset$, and $\xi_1, \xi_2$ satisfy the invariant property, i.e., $|\xi_1| = |\text{elems} \circ \pi_1^* (\xi_1)|$, and similarly for $\xi_2$.

$$|\text{elems} \circ \pi_1^* (\xi)|$$

178
\[ |\text{elems} \circ \pi_1^* (\xi_1 \setminus \xi_2) | \\
= |\text{elems} \circ \pi_1^* (\xi_1) \setminus \pi_1^* (\xi_2) | \\
= |\text{elems} \circ \pi_1^* (\xi_1) \cup \text{elems} \circ \pi_1^* (\xi_2) | \\
= |\text{elems} \circ \pi_1^* (\xi_1) + \text{elems} \circ \pi_1^* (\xi_2) | \\
= |\xi_1| + |\xi_2| \\
= |\xi_1 \setminus \xi_2| \]

Lemma A.16 \( R_{1,0} \circ Add\_Map[\xi_1, \xi_2] = R_{1,0} \xi_1 \sqcup R_{1,0} \xi_2. \)

Proof

The \( Add\_Map \) operation preserves the invariant, next we show that it refines the \( \sqcup \) operation. An inductive argument on \( |\xi| \) is used, the basis case is \( \xi_1 = \Lambda \).

If \( \xi_1 = \Lambda \), then \( Add\_Map[\xi_1, \xi_2] = \xi_2 \), and thus by definition of \( R_{1,0} \), we get

\[
R_{1,0}(Add\_Map[\xi_1, \xi_2]) \\
= R_{1,0}(\xi_2) \\
= \theta \sqcup R_{1,0}(\xi_2) \\
= R_{1,0}(\Lambda) \sqcup R_{1,0}(\xi_2) \\
= R_{1,0}(\xi_1) \sqcup R_{1,0}(\xi_2)
\]

and the basis case is proved.

Suppose the property is true whenever \( |\xi_1| \leq k \), and consider \( |\xi_1| = k + 1 \), where \( \xi_1 = (x, y) \setminus \xi'_1 \). Then \( Add\_Map[\xi_1, \xi_2] = (x, y) \setminus (\xi'_1 \setminus \xi_2) \), and thus by definition of \( R_{1,0} \) we have that:

\[
R_{1,0}(Add\_Map[\xi_1, \xi_2]) \\
= R_{1,0}((x, y) \setminus (\xi'_1 \setminus \xi_2)) \\
= [x \mapsto y] \sqcup R_{1,0}(\xi'_1 \setminus \xi_2). \\
= [x \mapsto y] \sqcup (R_{1,0}(\xi'_1) \sqcup R_{1,0}(\xi_2)).
\]

(Follows from the inductive hypothesis)

\[
= ([x \mapsto y] \sqcup R_{1,0}(\xi'_1)) \sqcup R_{1,0}(\xi_2). \\
= (R_{1,0}((x, y)) \sqcup R_{1,0}(\xi'_1)) \sqcup R_{1,0}(\xi_2). \\
= R_{1,0}(\xi_1) \sqcup R_{1,0}(\xi_2).
\]

The fact that \( Add\_Map \) is a refinement of \( \sqcup \), i.e., the commuting diagram property holds, follows by induction.
The next operation we consider is the $\uparrow$ operation, which is refined by the $Upd_{\text{Map}}$ operation. It is defined as follows:

\[
\text{pre}_\text{Upd_{Map}} : (X \times Y)^* \times (X \times Y)^* \hookrightarrow B
\]

\[
\text{pre}_\text{Upd_{Map}}[\xi_1, \xi_2] \triangleq \text{TRUE}
\]

\[
\text{Upd_{Map}} : (X \times Y)^* \times (X \times Y)^* \hookrightarrow (X \times Y)^*\]

\[
\text{Upd_{Map}}[\Lambda, \xi_2] \triangleq \xi_2
\]

\[
\text{Upd_{Map}}[\langle \sigma \rangle \land \xi_1, \xi_2] \triangleq
\]

\[
\sigma \in \xi_2
\]

\[
\mapsto \text{Upd_{Map}}[\xi_1, \xi_2]
\]

\[
\mapsto \sigma \land \text{Upd_{Map}}[\xi_1, \xi_2]
\]

This operation as before, has two proof obligations associated with it. They are stated as the following lemmas.

**Lemma A.17** $\text{pre}_\text{Upd_{Map}}[\xi_1, \xi_2] \land \xi = \text{Upd_{Map}}[\xi_1, \xi_2] \Rightarrow \text{Inv_{Map}}[\xi]$.

**Proof**

This is similar to the proof of Lemma A.15.

**Lemma A.18** $\mathcal{R}_{1,0} \circ \text{Upd_{Map}}[\xi_1, \xi_2] = \mathcal{R}_{1,0} \uparrow \mathcal{R}_{1,0} \xi_2$.

**Proof**

This is similar to the proof of Lemma A.16.

The $\text{Dom}$ and $\text{Rng}$ operations correspond to the abstract $\text{dom}$ and $\text{rng}$ operations.

\[
\text{Dom} : (X \times Y)^* \hookrightarrow \mathcal{P}X
\]

\[
\text{Dom}[\Lambda] \triangleq \emptyset
\]

\[
\text{Dom}[\xi] \triangleq \text{elems} \circ \pi^*_1(\xi)
\]

\[
\text{Rng} : (X \times Y)^* \hookrightarrow \mathcal{P}Y
\]

\[
\text{Rng}[\Lambda] \triangleq \emptyset
\]

\[
\text{Rng}[\xi] \triangleq \text{elems} \circ \pi^*_2(\xi)
\]

We may also develop the corresponding concrete operations for the $\less{\text{[]}}$ and $\less{\text{[]}}$ operations.

180
A.9 Summary

The main conclusions from this study on the representation of set theory via unique sequences, and the study of general sequences in $\Sigma^*$ and multisets is the following:

- $(\Sigma^*_i, \diamond, \Lambda)$ forms a non-commutative monoid.
- $(\Sigma_i^*, \sqcap, \Lambda')$ is a commutative monoid.
- $(\Sigma_i^*, \triangledown, \Lambda')$ is isomorphic to $(P\Sigma, \cup, \emptyset)$.
- $(\Sigma_i^*, \triangledown, \Sigma_{\setminus})$ forms a non-commutative monoid.
- $(\Sigma_i^*, \triangledown', \Sigma'_{\setminus})$ is a commutative monoid.
- $(\Sigma_i^*, \triangledown', \Sigma'_{\setminus})$ is isomorphic to $(P\Sigma, \cap, \Sigma)$.

- The canonical sequence representation of a set is achieved by placing a total order on $\Sigma$, and choosing the unique ordered sequence from each class. This provides a mechanism for the implementation of sets via unique sequences.

- The Quotient Monoid $\Sigma^*/\sim$ is isomorphic to the multiset (or bag) defined over $\Sigma$, i.e., $(\Sigma \mapsto \mathbb{N}_1)$ where the congruence equivalence relation $\sim$ is a defined in terms of the specialized permutation operation $(MPerm)$ for multi-sequences.

- Sequences $\Sigma^*$ may be reified to arrays $(\mathbb{N} \mapsto \Sigma)$. The sequence operations $\#$, $\triangleleft$, $Hd$ and $Tail$ require refinement for this concrete structure. The concrete definitions are more cumbersome that the abstract operation definitions.

- Map structures $(X \mapsto Y)$ may be reified to tables of the form $(X \times Y)$ or a sequence of the form $(X \times Y)^*$. 

181
Appendix B

Indexed Structures

B.1 Introduction

The study of indexed structures is a natural generalization of the indexed monoid structure, identified in [47]. The key property of indexed structures is the inheritance property; i.e., the indexed structure inherits its behaviour from the underlying base structure, for example, the indexed monoid is a monoid inheriting its monoidal property from the underlying base monoid. In this section generalizations of the base structure are considered; in particular, base structures such as groups, rings, fields, vector spaces are examined. The objective is to extend the base structures to indexed structures.

The study identifies a fundamental problem with the indexed monoid definition in [47], the problem is invertible elements. This leads to the distinction between the indexed monoid and the quasi indexed monoid, the latter addressing invertible elements. If the base monoid is commutative then the (quasi) indexed monoid inherits the commutativity. If \( \Omega \) is a set of operators for the monoid \( M \), then the (quasi) indexed monoid inherits \( \tilde{\Omega} \) as a set of operators. If the base structure is a group then the (quasi) indexed structure is a group, the indexed structure fails to be a group.

Base structures with more than one binary operation are then examined, the objective is to determine if an extension to an indexed structure is meaningful. The definition of indexed multiplication is based on domain intersection of maps, whereas the definition of indexed addition is in terms of domain union of maps. In this way indexed semi-rings may be formed from base sub-rings; quasi indexed rings may be formed from base rings. The indexed field may be defined on total maps only; finally, the quasi indexed vector space may be defined.

The studying and identification of these new structures is conducted for pragmatic purposes. It is intended that such structures should be applicable to formal methods, and ultimately assist in simplification of formal specifications. Several applications of indexed structures are presented including the teleshopping and banking domain.
The example of a bag of marbles containing three red, one green and two blue marbles enables the concept of the indexed monoid notation to be easily grasped. This is represented as \([\text{Rd} \mapsto 3, \text{Gr} \mapsto 1, \text{Bl} \mapsto 2]\). The colours of the marbles represent an index set, and the number of occurrences of a particular marble is represented by the monoid \(N\). Consider the addition two green and one orange marble to the bag, i.e., \([\text{Gr} \mapsto 2, \text{Or} \mapsto 1]\). The updated bag is of the form \([\text{Rd} \mapsto 3, \text{Gr} \mapsto 1 + 2, \text{Bl} \mapsto 2, \text{Or} \mapsto 1]\).

An element is added directly into the bag if it is not already present, otherwise the number of occurrences of the element is added to the number of occurrences of the element currently in the bag. Formally, the indexed monoid consists of a monoid \((M, \ast, u)\) termed the base monoid, and an indexed set \(I\), the structure \((I \mapsto M, \odot, \theta)\) is an indexed monoid where for \(\mu \in (I \mapsto M)\) we have:

\[
\mu \odot [i \mapsto m] \triangleq \begin{cases} 
\mu \sqcup [i \mapsto m] & i \not\in \mu \\
\mu \upharpoonright [i \mapsto \mu(i) \ast m] & i \in \mu
\end{cases}
\]

(B.1)

This enables us to provide the full definition of \(\odot\) on arbitrary \(\mu, \mu'\) as follows:

\[
\mu \odot \theta \triangleq \mu \quad \quad \quad \quad (B.2)
\]

\[
\mu \odot ([i \mapsto m] \sqcup \mu') \triangleq \begin{cases} 
(\mu \sqcup [i \mapsto m]) \odot \mu' & i \not\in \mu \\
(\mu \upharpoonright [i \mapsto \mu(i) \ast m]) \odot \mu' & i \in \mu
\end{cases}
\]

(B.3)

**Comment B.1 (Signature of Indexed Monoid)** This structure yields a monoid which inherits its operations from the underlying base monoid, i.e., if the underlying base monoid is commutative, then the indexed monoid is commutative. However, the definition of the indexed monoid presented here deviates from the signature of the indexed monoid in [47]; in particular, the latter considers the indexed monoid to have signature \((I \mapsto M')\), where \(M' = \{[u] \mid u \in M\}\), and \(u\) is the unit of \(M\). In fact, the signature in [47] is incorrect, unless the \(\odot\) operation is tailored to remove entries of the form \([i \mapsto u]\).

**Lemma B.1** Suppose \(\mu, \mu' \in (I \mapsto M)\) then \(\text{dom} \circ (\mu \odot \mu') = \text{dom} \mu \cup \text{dom} \mu'\).

**Proof (By Induction)**

The proof is by induction based on \(|\mu'|\), where \(|\mu'|\) denotes the cardinality of the map \(\mu'\), or equivalently the cardinality of \(\mu'\) considered as \(\mu' \subseteq (I \times M)\). The basis case is \(|\mu'| = 0\), and corresponds to \(\mu' = \theta\). By definition, \(\mu \odot \theta = \mu\), thus \(\text{dom} \circ (\mu \odot \theta) = \text{dom} \mu = \text{dom} \mu \cup \emptyset = \text{dom} \mu \cup \text{dom} \theta\), as required.

Suppose the result is true for all \(\mu'\) such that \(|\mu'| < k\), and consider \(\mu' = [i \mapsto m] \sqcup \eta\) where \(|\eta| = k - 1\). Then we have \(\text{dom} (\mu \odot \mu') = \text{dom} \circ (\mu \odot ([i \mapsto m] \sqcup \eta))\). There are two cases \(i \in \mu\) and \(i \not\in \mu\). For the former \(\text{dom} (\mu \odot \mu') = \text{dom} \circ (\mu \odot ([i \mapsto \mu(i) \ast m]) \odot \eta\), by the inductive hypothesis, this yields \(\text{dom} \circ (\mu \upharpoonright [i \mapsto \mu(i) \ast m]) \cup \text{dom} \eta\). Clearly, for
\[ i \in \mu \text{ we have } \text{dom}(\mu \upharpoonright [i \mapsto \mu(i) \ast m]) = \text{dom}\mu = \text{dom}\mu \cup \{i\}, \text{ and thus it follows that } \text{dom} \circ (\mu \bigcirc \mu') = \text{dom}\mu \cup \{i\} \cup \text{dom}\eta = \text{dom}\mu \cup \text{dom}\mu' \text{ as desired. The second case where } i \notin \mu \text{ is similar, and is not elaborated.} \]

### B.2 Quasi Indexed Monoids

The definition of the Indexed Monoid given in equation B.1 does not generalize to the definition of the indexed group, even if the underlying structure is a group. The bag of marbles example demonstrates this effectively; suppose \( \beta_1 = [Bl \mapsto 2, Gr \mapsto 3] \), and suppose negative numbers are used to indicate that marbles are owed. For example, \( \beta_2 = [Bl \mapsto -2] \) indicates that the owner of \( \beta_2 \) owes two blue marbles, i.e., there are -2 marbles in the bag.

Consider the addition of \( \beta_1 \) and \( \beta_2 \); this yields \([Bl \mapsto 0, Gr \mapsto 3]\); the addition of \( \beta_1 \) and \( \beta_3 \) yields \([Bl \mapsto 0, Gr \mapsto 0]\) where \( \beta_3 = [Bl \mapsto -2, Gr \mapsto -3] \). An examination of \( \beta_1 \) and \( \beta_3 \) yields that they are effectively inverses of one another, however, when they are added together the result is not the identity element \( \theta \) of \( (I \mapsto M) \).

The required result is that whenever \( \beta_1 \) and \( \beta_3 \) are as defined above, then the result of the bag addition operation is annihilation, i.e., \( \beta_1 \bigcirc' \beta_3 = \theta = \beta_3 \bigcirc' \beta_1 \). This is achieved with the Quasi Indexed Monoid structure, a structure which behaves similarly to the Indexed Monoid, except that whenever the effect of the indexed monoid operation is to yield entries of the form \([x \mapsto u]\), where \( u \) is the unit of \( M \), then such entries are removed.

The signature of the Indexed Monoid structure is \((I \mapsto M)\). This is a deviation from [47], however, the signature of the Quasi Indexed Monoid structure is \( I \mapsto M' \), where \( M' = \{[u]M \} \). In fact, the signature of the quasi indexed monoid presented here matches the signature of the indexed monoid defined in [47].

\[
\begin{align*}
\mu \cup [i \mapsto m] & \quad i \notin \mu \\
\mu \bigcirc' [i \mapsto m] & \triangleq \begin{cases} 
\mu \upharpoonright [i \mapsto \mu(i) \ast m] & i \in \mu \land \mu(i) \ast m \neq u \\
\emptyset & i \in \mu \land \mu(i) \ast m = u
\end{cases} & (B.4)
\end{align*}
\]

The full definition of \( \bigcirc' \) on arbitrary \( \mu, \mu' \) is as follows:

\[
\begin{align*}
\mu \bigcirc' ([i \mapsto m] \cup \mu') & \triangleq (\mu \upharpoonright [i \mapsto m]) \bigcirc' \mu' & i \notin \mu \\
& \quad (\mu \upharpoonright [i \mapsto \mu(i) \ast m]) \bigcirc' \mu' & i \in \mu \land \mu(i) \ast m \neq u \\
& \quad \emptyset & i \in \mu \land \mu(i) \ast m = u & (B.5)
\end{align*}
\]

**Lemma B.2** Suppose \( u \) the identity element of \( M \), and \( \mu, \mu' \in (I \mapsto M') \) then \( u \notin \text{rng} \circ \mu \bigcirc' \mu' \).
Suppose not, then it follows that \((\mu \odot' \mu')(i) = u\) for some \(i \in \text{dom}(\mu \odot' \mu')\). If \(i \in \mu \land i \notin \mu'\) then it follows from the definition of \(\odot'\) that \(\mu(i) = u\), which is a contradiction. The case \(i \notin \mu \land i \in \mu'\) is similar. The remaining case is \(i \in \mu \land i \in \mu'\) and from the definition of \(\odot'\), it follows that \(\mu(i) \ast \mu'(i) = u\), which is a contradiction, since by the definition of \(\odot'\), if \(\mu(i) \ast \mu'(i) = u\) then \(i\) is removed from the resulting map.

**Lemma B.3**  \(\text{dom} \circ (\mu \odot' \mu') \subseteq \text{dom} \mu \cup \text{dom} \mu'.\)

Suppose \(i \in (\mu \odot' \mu')\) then it is clear from the definition of \(\odot'\) that \(i \in \mu \lor i \in \mu',\) or both. Thus \(i \in \text{dom} \mu \cup \text{dom} \mu'.\) Lemma B.1 demonstrates that equality holds for the indexed monoid; however, the counter example \(\mu = [\text{Red} \mapsto 2]\) and \(\mu' = [\text{Red} \mapsto -2]\) demonstrates that \(\text{dom} \circ (\mu \odot' \mu') = \text{dom} \theta = \emptyset \subseteq \text{dom} \mu \cup \text{dom} \mu',\) and thus equality does not hold for the quasi indexed monoid.

**Theorem B.1** \((I \mapsto M', \odot', \theta)\) yields a Monoid, called the Quasi Indexed Monoid.

**Proof**

The closure property follows by observing that the only way by which this property may fail to hold is if \((\mu_1 \odot' \mu_2)(i) = u\) for some \(i \in (\mu_1 \odot' \mu_2).\) However, Lemma B.2 has shown that \(u \notin \text{rng} \circ (\mu \odot' \mu')\) for \(\mu_1, \mu_2 \in (I \mapsto M'),\) and thus the closure property is established.

The proof of associativity is straightforward though tedious, there are seven cases. The approach taken is to consider \((\mu_1 \odot' \mu_2) \odot' \mu_3(i)\) for each \(i \in (\mu_1 \odot' \mu_2) \odot' \mu_3,\) and to demonstrate that this is equal to \(\mu_1 \odot' (\mu_2 \odot' \mu_3)(i)\). It is clear that \(i \in (\mu_1 \odot' \mu_2) \odot' \mu_3 \Rightarrow i \in \mu_1 \lor i \in \mu_2 \lor i \in \mu_3.

1. \(i \in \mu_1, i \in \mu_2, i \in \mu_3\)

Then \((\mu_1 \odot' \mu_2) \odot' \mu_3(i) = (\mu_1(i) \ast \mu_2(i)) \ast \mu_3(i) = \mu_1(i) \ast (\mu_2(i) \ast \mu_3(i)).\) By the definition of \(\odot'\), it is clear that \(\mu_1(i) \ast (\mu_2(i) \ast \mu_3(i)) \neq u \Rightarrow i \in \mu_1 \odot' \mu_2 \odot' \mu_3,\) and thus \((\mu_1 \odot' \mu_2) \odot' \mu_3(i) = \mu_1 \odot' (\mu_2 \odot' \mu_3)(i),\) as required.

2. \(i \in \mu_1, i \in \mu_2, i \notin \mu_3\)

We get \((\mu_1 \odot' \mu_2) \odot' \mu_3(i) = \mu_1(i) \ast \mu_2(i)\) and \(\mu_1(i) \ast \mu_2(i) \neq u.\)

Clearly \(i \in (\mu_2 \odot' \mu_3)\) and \(\mu_2 \odot' \mu_3)(i) = \mu_2(i).\)

Thus \(i \in \mu_1 \odot' (\mu_2 \odot' \mu_3)\) since \(\mu_1(i) \ast \mu_2(i) \neq u,\) and \(\mu_1 \odot' (\mu_2 \odot' \mu_3) = \mu_1(i) \ast \mu_2(i),\)

and associativity holds for Case 2.

The remaining cases are similar.

Finally it is clear that \(\theta\) is the identity.
Theorem B.2 If the underlying base monoid \((M, *, u)\) is a commutative monoid, then the quasi indexed monoid \((I \mapsto M', \otimes', \theta)\) is, in fact, a commutative monoid, i.e., the quasi indexed monoid inherits behaviour from the properties of the underlying base monoid.

Proof
This result follows immediately from a sub-case analysis.

\[
\begin{align*}
i & \in \mu_1 & i & \notin \mu_2 \\
i & \notin \mu_1 & i & \in \mu_2 \\
i & \in \mu_1 \quad i & \in \mu_2
\end{align*}
\]

Theorem B.3 If \((G, *, u)\) is a group then the quasi indexed structure \((I \mapsto G', \otimes', \theta)\) is, in fact, a group, called the ‘Quasi Indexed Group’.

Proof
This is an interesting result, the only group property that must be proved is that given \(\mu \in (I \mapsto G')\), then a \(\mu^{-1} \in (I \mapsto G')\) may be constructed such that \(\mu \otimes' \mu^{-1} = \mu^{-1} \otimes' \mu = \theta\).

Since \(G\) is a group, it follows that for any \(g \in G\) there is a corresponding \(g^{-1} \in G\) such that \(g \ast g^{-1} = g^{-1} \ast g = u\). Thus it is required to construct a \(\mu^{-1}\) from \(\mu\); this is achieved as follows:

\[
\mu = \begin{bmatrix} i_1 \mapsto g_1 \\ i_2 \mapsto g_2 \\ \vdots \\ i_n \mapsto g_n \end{bmatrix} \quad \mu^{-1} = \begin{bmatrix} i_1 \mapsto g_1^{-1} \\ i_2 \mapsto g_2^{-1} \\ \vdots \\ i_n \mapsto g_n^{-1} \end{bmatrix}
\] (B.6)

Then consider \(\mu \otimes' \mu^{-1}\), clearly \(\text{dom} \mu = \text{dom} \mu^{-1}\), and for each \(i \in \text{dom} \mu\) we have \(\mu(i) \ast \mu^{-1}(i) = g_i \ast g_i^{-1} = u\), thus \(i \notin \mu \otimes' \mu^{-1}\) by definition of the \(\otimes'\) operator. Clearly, from the definition of \(\otimes'\), given \(j \notin \text{dom} \mu, j \notin \text{dom} \mu^{-1}\) we have \(j \notin \mu \otimes' \mu^{-1}\), and thus \(\mu \otimes' \mu^{-1} = \theta\). Similarly, \(\mu^{-1} \otimes' \mu = \theta\). Thus given arbitrary \(\mu\) in \((I \mapsto G')\), the corresponding \(\mu^{-1}\) in \((I \mapsto G')\) has been explicitly constructed.

\((I \mapsto G')\) yields a group as \(\mu^{-1} \in (I \mapsto G')\) whenever \(\mu \in (I \mapsto G')\). Thus if we are dealing with a proper subset \(S\) of \((I \mapsto G')\), it is clear that we may then consider the smallest group containing the subset \(S\), it is also clear that this group is obtained by adding extra elements to \(S\) to make it a group. This group is thus generated by the set \(S\).

Note 1: Several examples are considered to demonstrate the applicability of the structure. Consider a bag, represented by a map \(\mu\). The concept of the inverse bag of \(\mu\), represented by \(\mu^{-1}\), is that when the bag \(\mu\) meets its formal inverse, the bag \(\mu^{-1}\), the net effect is
annihilation, yielding the empty bag $\theta$. The map $\mu$ may represent the goods purchased by an individual customer, the map $\mu^{-1}$ represents the adjustment that must be made to a store’s quantity of goods on hand.

**Note 2:** The definition of the $\oplus'$ operator has the effect of removing entries of the form $g \mapsto u$ from the map.

### B.2.1 Application of Quasi Indexed Groups

Consider a banking model of saving accounts, each account has a non zero balance. Several transactions may take place on an account throughout the banking day, however, at the end of the banking day there is a single transaction figure which reflects the updates which have taken place to the customer’s account. If the net transaction is a debit, and the debit amount matches the balance in the account, then it is appropriate to close the account, since a savings account with zero balance is not meaningful. Suppose savings accounts are modelled by $\tau$ as follows:

$$\tau : Acc \mapsto \mathbb{Q}$$

Then the updates that take place to the accounts may be modelled by:

$$\psi : Acc \mapsto \mathbb{Q}$$

thus if the updates are done at the end of the banking day basis, the new balances on the accounts are given by:

$$\tau \oplus' \psi$$

This has the effect of removing accounts from the banking system whenever $\tau(i) + \psi(i) = 0$, which is appropriate for savings accounts. The transactions made to accounts in the bank are represented as follows:

$$\psi_t : Acc \mapsto \mathbb{Q}^*$$

The net changes to be made to the individual accounts is given by $\psi$, and is obtained by considering the transactions made to the individual accounts. The map $\psi$ is given by the $Upd_{Trn}$ operation.

$$Upd_{Trn} : (Acc \mapsto \mathbb{Q}^*) \mapsto (Acc \mapsto \mathbb{Q})$$

$$Upd_{Trn}[\psi_t] \equiv Upd_{Trn}[\psi_t] \theta$$
\[ U_{pd \text{Trn}} : (\text{Acc} \mapsto Q^*) \mapsto (\text{Acc} \mapsto Q) \mapsto (\text{Acc} \mapsto Q) \]
\[ U_{pd \text{Trn}}[\theta] \psi \triangleq \psi \]
\[ U_{pd \text{Trn}}[\psi_t] \psi \triangleq \]

Let \( a \in \psi_t \) in
\( \psi'_t \mapsto \psi(a) \psi_t \)
\( T_a = 0 \)
\( \mapsto U_{pd \text{Trn}}[\psi'_t] \psi \)
\( \mapsto U_{pd \text{Trn}}[\psi'_t] (\psi \uplus [a \mapsto T_a]) \)

Note that the definition of the \( U_{pd \text{Trn}} \) operation omits account entries \( a \) from the update map if the net effect of the debits and credits on the account is 0.

### B.3 Inherited Operators on Indexed Monoids

Recall (cf., Page 123 of [45], Page 157 of [55]) that a set \( \Omega \) such that each \( \omega \in \Omega \) is an endomorphism of the monoid \((M, \ast, u)\) is called a set of operators for the monoid \( M \), denoted \((\Omega, (M, \ast, u))\). An endomorphism of \( M \) is a mapping \( \omega : M \mapsto M \) such that:

\[
\omega(m_1 \ast m_2) = \omega(m_1) \ast \omega(m_2) \tag{B.7}
\]
\[
\omega(u) = u \tag{B.8}
\]

Consider a monoid with operators, i.e., \((\Omega, (M, \ast, u))\). This section determines the relationship between operators on the base monoid and inherited operators on the indexed monoid or quasi indexed monoid. The indexed monoid operator \( \bar{\omega} \) may be constructed from the underlying base operator \( \omega \). The behaviour of \( \bar{\omega} \) is as follows:

\[
\bar{\omega}(\mu_1 \boxplus \mu_2) = \bar{\omega}(\mu_1) \boxplus \bar{\omega}(\mu_2) \tag{B.9}
\]
\[
\bar{\omega}(\emptyset) = \emptyset \tag{B.10}
\]

The definition of \( \bar{\omega} \) is as follows,

\[
\bar{\omega}(\emptyset) \triangleq \emptyset \tag{B.11}
\]
\[
\bar{\omega}(\mu \uplus [i \mapsto m]) \triangleq \bar{\omega}(\mu) \uplus [i \mapsto \omega(m)] \tag{B.12}
\]

The proof that \( \bar{\omega} \) is an operator on \((I \mapsto M)\) is presented in Theorem B.4. First an elementary property of \( \bar{\omega} \) is stated.
Lemma B.4 Suppose $\mu \in (I \rightharpoonup M)$ then $\text{dom} \mu = \text{dom} \overline{\omega}(\mu)$.

Theorem B.4 Suppose $\omega$ is an operator on $M$, then $\overline{\omega}$ is an operator on the Indexed Monoid $(I \rightharpoonup M)$, where $\overline{\omega}$ is defined in Equation B.11 and B.12.

Proof

$$\overline{\omega} \left( \mu \rhd [i \mapsto m] \right) = \left\{ \begin{array}{l}
i \in \mu \mapsto \mu \left( i \mapsto \mu(i) \rhd [i \mapsto m] \right) \\
i \not\in \mu \mapsto \mu \cup [i \mapsto m]
\end{array} \right\}$$

$$\overline{\omega} \left( \mu \rhd [i \mapsto m] \right) = \left\{ \begin{array}{l}
i \in \mu \mapsto \overline{\omega} \left( \mu \left( i \mapsto \omega(\mu(i) \rhd [i \mapsto m]) \right) \right) \\
i \not\in \mu \mapsto \overline{\omega}(\mu) \cup [i \mapsto \omega(m)]
\end{array} \right\}$$

$$\overline{\omega} \left( \mu \rhd [i \mapsto m] \right) = \left\{ \begin{array}{l}
i \in \mu \mapsto \overline{\omega}(\mu) \cup [i \mapsto \omega(\mu(i) \rhd [i \mapsto m])] \\
i \not\in \mu \mapsto \overline{\omega}(\mu) \cup [i \mapsto \omega(m)]
\end{array} \right\}$$

This proves the result for the case of $\mu_2$ of the form $[i \mapsto m]$. The next stage is to prove that the result is true for arbitrary maps $\mu_1, \mu_2 \in (I \rightharpoonup M)$. Consider $i \in \mu_1 \rhd \mu_2$; we wish to show that:

$$\overline{\omega} \left( \mu_1 \rhd \mu_2 \right)(i) = (\overline{\omega} \left( \mu_1 \right) \rhd \overline{\omega} \left( \mu_2 \right))(i)$$

We employ divide and conquer techniques, and consider the three cases:

1. $i \in \mu_1 \land i \not\in \mu_2$
2. $i \not\in \mu_1 \land i \in \mu_2$
3. $i \in \mu_1 \land i \in \mu_2$

Then we have $(\mu_1 \rhd \mu_2)(i) = \mu_1(i)$, thus we have $[i \mapsto \mu_1(i)] \in (\mu_1 \rhd \mu_2)$. Thus $(\mu_1 \rhd \mu_2) = \left[ i \mapsto [\mu_1 \rhd \mu_2] \cup [i \mapsto \mu_1(i)] \right]$. Thus we get
\[
\tilde{\omega}(\mu_1 \circ \mu_2) = \tilde{\omega}(\mu_1 \circ \mu_2) \triangleright [i \mapsto \mu_1(i)]
\]
\[
= \tilde{\omega}(\tilde{\omega}(\mu_1 \circ \mu_2)) \triangleright [i \mapsto \omega(\mu_1(i))]
\]
\[
\Rightarrow \tilde{\omega}(\mu_1 \circ \mu_2)(i) = \omega(\mu_1(i)).
\]

Clearly, \(i \in \tilde{\omega}(\mu_1)\) and \(i \notin \tilde{\omega}(\mu_2) \Rightarrow \tilde{\omega}(\mu_1) \circ \tilde{\omega}(\mu_2)(i) = \tilde{\omega}(\mu_1)(i) = \omega(\mu_1(i))\), and thus Case 1 is proved.

The remaining cases are similar.

**Note:** The signature of \(\omega\) is \(M \mapsto M\), whereas the signature of \(\tilde{\omega}\) is \((I \mapsto M) \mapsto (I \mapsto M)\).

### B.4 Operators on Quasi Indexed Monoids

The definition of the \(\tilde{\omega}\) operator needs to be amended in order to preserve the closure property in the quasi indexed monoid structure. The problem which arises is that \(\tilde{\omega}[i \mapsto m] = [i \mapsto \omega(m)]\) may yield an entry of the form \([i \mapsto u]\) which is not in \((I \mapsto M')\). Thus the \(\tilde{\omega}\) definition is amended to ensure that if \(\omega(m) = u\), then \([i \mapsto \omega(m)]\) is removed.

\[
\tilde{\omega}(\theta) \triangleq \theta
\]
\[
\tilde{\omega}(\mu \triangleright [i \mapsto m]) \triangleq \tilde{\omega}(\mu) \triangleright [i \mapsto \omega(m)] \text{ where } \omega(m) \neq u
\]
\[
\tilde{\omega}(\mu \triangleright [i \mapsto m]) \triangleq \tilde{\omega}(\mu) \text{ otherwise}
\]

**Theorem B.5** Suppose \(\omega\) an operator on the base monoid \(M\), then \(\tilde{\omega}\) is an operator on the Quasi Indexed Monoid \((I \mapsto M')\).

**Proof**

This proof is an extension of the proof of Theorem B.4. In particular, there was no requirement in the proof of the previous theorem to distinguish the cases where \(\mu_1(i) \ast \mu_2(i) = u\), or where \(\omega(\mu_1(i)) = u\), or \(\omega(\mu_2(i)) = u\), or where \(\omega(\mu_1(i)) \ast \omega(\mu_2(i)) = u\). Thus it is required to show that \(\tilde{\omega}(\mu_1 \circ' \mu_2) = \tilde{\omega}(\mu_1) \circ' \tilde{\omega}(\mu_2)\) for these cases. The proof is given by a sub-case analysis as before.

### B.5 Applications of Operators

Consider the bag of marbles \([Rd \mapsto 2, Gr \mapsto 3] = [Rd \mapsto 2] \oplus [Gr \mapsto 3]\). Consider an operation \(\omega\) that doubles a number (this is an endomorphism of \((\mathbb{N}, +, 0)\)), \(\tilde{\omega}\) has the effect of doubling the contents of a bag, and is an endomorphism of \((I \mapsto \mathbb{N})\).
\[
\bar{2} [Rd \mapsto 2, Gr \mapsto 3] = [Rd \mapsto 2] \oplus [Gr \mapsto 3]
\]

The operator \(\bar{2}\) is an endomorphism of \((I \mapsto N)\). This indexed monoid with operators may be applied to the simple banking domain. Consider the problem of interest application to savings accounts. This takes the form \((1 + r)\tau(a)\); let \(I_r = (1 + r)\), then \(I_r\tau\) represents the application of interest to the accounts. The fact that \(I_r\) is an operator on \(\tau : (Acc \mapsto Q)\) is a consequence of Theorem B.5, since \(I_r\) is an operator on \(Q\). Clearly, \(I_r(x + y) = (1 + r)(x + y) = (1 + r)x + (1 + r)y\), by the distributive property of \(Q = I_r(x) + I_r(y)\).

### B.6 (Quasi-) Indexed Semi-Rings and Rings

The base structures considered here have several binary operations, for example, base structures such as semi-rings and rings are considered. In these cases a natural extension to the indexed semi-rings and indexed rings is derived. However, an alternate approach is necessary for base structures which are fields and vector spaces.

The generalization of multiplication to an indexed structure requires that a well defined multiplication operation be defined on the indexed structure. The standard indexed operation \(\otimes\) is _inclusive_, i.e., the result of an operation \(\mu_1 \otimes \mu_2\) contains all elements in the domain of both maps. However, intuitively the indexed multiplication operation is _restrictive_, since it is not meaningful to include an element \([x \mapsto m]\), unless \(x\) is in both \(\mu_1\) and \(\mu_2\). Consequently, the definition of the indexed multiplication operation is restricted to those elements that both maps have in common. If \(\mu_1\) and \(\mu_2\) are disjoint, then the empty map \(\theta\) is the result of the operation. That is, since \(\theta\) is the unit (or zero) under indexed addition, the indexed structure formed may include zero divisors.

The \(\otimes^\cap\) operation is quite distinct from \(\otimes, \otimes'\) which have been discussed previously. The \(\otimes\) operator essentially ensures that \(\text{dom} (\mu_1 \otimes \mu_2) = \text{dom} \mu_1 \cup \text{dom} \mu_2\); the \(\otimes'\) performs a conditional union of the domains provided no elements in the structure annihilate one another; however, the \(\otimes^\cap\) operation is quite distinct, in that the product of the two maps \(\mu_1, \mu_2\) is restricted to the map elements that both maps have in common, i.e., all other map elements are ignored.

Finally, distinctions are made between the \(\otimes^\cap\) and the \(\otimes^\cap'\) operations. The latter operation ensures that entries of the form \([i \mapsto u]\) are removed from the result, whereas the former does not. The definition of the indexed and quasi indexed multiplication operations is presented as follows:
Indexed Multiplication Definition

\[
\mu \otimes \cap [i \mapsto r] \triangleq \begin{cases} 
\theta & i \not\in \mu \\
[i \mapsto \mu(i) \times r] & \text{otherwise}
\end{cases}
\]  

(B.16)

This is extended to arbitrary maps \( \mu \) and \( \mu' \) as follows:

\[
\mu \otimes \cap \theta \triangleq \theta
\]

(B.17)

\[
\mu \otimes \cap ([i \mapsto r] \cup \mu') \triangleq \begin{cases} 
\mu \otimes \cap \mu' & i \not\in \mu \\
[i \mapsto (\mu \otimes \cap \mu') \cup [i \mapsto \mu(i) \times r]) & \text{otherwise}
\end{cases}
\]  

(B.18)

The following lemma states an elementary, but fundamental property of the \( \otimes \cap \) operator.

**Lemma B.5** \( \text{dom} \mu \otimes \cap \mu' = \text{dom} \mu \cap \text{dom} \mu' \)

Quasi Indexed Multiplication Definition

The definition of the \( \otimes'^r \) operation is slightly more involved, in order to ensure that no entries of the form \([i \mapsto u]\) are created.

\[
\mu \otimes'^r [i \mapsto r] \triangleq \begin{cases} 
\theta & i \not\in \mu \\
[i \mapsto \mu(i) \times r] & i \in \mu \land \mu(i) \times r \neq u \\
\theta & i \in \mu \land \mu(i) \times r = u
\end{cases}
\]  

(B.19)

This enables us to provide the full definition of \( \otimes'^r \) on arbitrary \( \mu, \mu' \).

\[
\mu \otimes'^r \theta \triangleq \theta
\]

(B.20)

\[
\begin{align*}
\mu \otimes'^r ([i \mapsto r] \cup \mu') & \triangleq \begin{cases} 
\mu \otimes'^r \mu' & i \not\in \mu \\
[i \mapsto (\mu \otimes'^r \mu') \cup [i \mapsto \mu(i) \times r]) & i \in \mu \land \mu(i) \times r \neq u \\
\mu \otimes'^r \mu' & i \in \mu \land \mu(i) \times r = u
\end{cases} \\
\end{align*}
\]  

(B.21)

The following lemma states an elementary, but fundamental property of the \( \otimes'^r \) operator:

**Lemma B.6** \( \text{dom} \mu \otimes'^r \mu' \subseteq \text{dom} \mu \cap \text{dom} \mu' \)

**B.7 Indexed Semi-ring**

**Theorem B.6** Given the base semi ring \((R, +, \times)\) with additive unity \(u\), then the structure \((I \mapsto R, \oplus, \otimes)\) with additive identity \(\theta\), and with \(\oplus\) and \(\otimes\) as defined yields a semi ring called the Indexed Semi-ring.
Proof

It has been previously demonstrated that given \((R, +)\), a semigroup with unity \(u\), then 
\( (I \mapsto R, \oplus, \theta) \) forms an indexed monoid. It is clear from its definition that the \( \otimes^n \) operation is closed, thus it is required to demonstrate associativity of \( \otimes^n \), and furthermore, that the distributive property of \( \otimes^n \) over \( \oplus \) holds.

In order to prove the associativity property it should be noted that \( i \in (\mu_1 \otimes^n \mu_2) \otimes^n \mu_3 \) if and only if \( i \in \mu_1 \land i \in \mu_2 \land i \in \mu_3 \), which is exactly the requirement for \( i \in \mu_1 \otimes^n (\mu_2 \otimes^n \mu_3) \). Thus all that is required is to show that \( (\mu_1 \otimes^n \mu_2) \otimes^n \mu_3(i) = \mu_1 \otimes^n (\mu_2 \otimes^n \mu_3)(i) \). This is equivalent to demanding \( (\mu_1(i) \times \mu_2(i)) \times \mu_3(i) = \mu_1(i) \times (\mu_2(i) \times \mu_3(i)) \), which is immediate from the associativity of the base semi-ring \( \times \) operation.

In order to prove the distributive operation, it should be noted that \( i \in \mu_1 \otimes^n (\mu_2 \oplus \mu_3) \) then \( i \in \mu_1 \land (i \in \mu_2 \lor i \in \mu_3) \). This is equivalent to \( i \in \mu_1 \land i \in \mu_2 \lor i \in \mu_3 \) which is precisely the requirement for \( i \in (\mu_1 \otimes^n \mu_2) \oplus (\mu_1 \otimes^n \mu_3) \). Thus it is required to prove that \( \mu_1 \otimes^n (\mu_2 \oplus \mu_3)(i) = (\mu_1 \otimes^n \mu_2 \oplus \mu_1 \otimes^n \mu_3)(i) \). If \( i \in \mu_2 \land i \in \mu_3 \), then the result is immediate, since this is equivalent to \( \mu_1(i) \times (\mu_2(i) + \mu_3(i)) = (\mu_1(i) \times \mu_2(i)) + (\mu_1(i) \times \mu_3(i)) \), which follows from the distributive property of the base semi ring.

Suppose \( i \in \mu_2 \land i \notin \mu_3 \), then \( (\mu_2 \oplus \mu_3)(i) = \mu_2(i) \), and so \( \mu_1 \otimes^n (\mu_2 \oplus \mu_3)(i) = \mu_1(i) \times \mu_2(i) = \mu_1 \otimes^n \mu_2(i) = (\mu_1 \otimes^n \mu_2 \oplus \mu_1 \otimes^n \mu_3)(i) \) as required. The remaining case \( i \notin \mu_2 \land i \in \mu_3 \) is similar. Similarly, the right distributive law holds, and the theorem is proved.

B.7.1 Applications of Indexed Semi Rings

The tele-shopping domain is considered here. Each customer performs purchases via an electronic basket. The indexed semi-ring is an applicable structure for this domain. The store and electronic baskets are bags, there is an associated price bag, which records the price of each item in the store. The indexed multiplication of the electronic bag by the price bag has the effect of computing the total value purchased of each item in the electronic bag. The total value of the electronic basket may then be computed.

Let \( \mu : Gd_J \mapsto Q \) represent the customer’s electronic basket, and \( \psi : Gd_J \mapsto Q \) represent the relationship between goods and prices in the store. Consider \( \mu \otimes^n \psi \), which yields the following:
\[
\mu = \begin{bmatrix}
g_1 \mapsto n_1 \\
g_2 \mapsto n_2 \\
\vdots \\
g_k \mapsto n_k
\end{bmatrix}
\psi = \begin{bmatrix}
g'_1 \mapsto p'_1 \\
g'_2 \mapsto p'_2 \\
\vdots \\
g'_n \mapsto p'_n
\end{bmatrix}
\]

We note implicitly we have \(\text{dom} \mu \subseteq \text{dom} \psi\), as otherwise the customer would have items in the electronic basket which have no associated price, which is not meaningful. Thus \(\text{dom} \circ \mu \otimes \psi = \text{dom} \mu = \text{dom} \mu \cap \text{dom} \psi\). In general, we have \(\text{dom} \circ \mu \otimes \psi = \text{dom} \mu \cap \text{dom} \psi \subseteq \text{dom} \mu\).

### B.8 Quasi-Indexed Rings

**Theorem B.7** Given the base ring \((R, +, \times)\) with additive unity \(u\), then the structure \((I \mapsto R', \oplus', \otimes')\) with additive identity \(\theta\), and \(\otimes', \otimes''\) as defined previously, is a ring called the Quasi Indexed Ring.

**Proof**

\((R, +)\) is a commutative group and the operation \(\oplus'\) has been shown previously to yield the commutative Quasi Indexed Group \((I \mapsto R', \oplus', \theta)\). Next, we must show \((I \mapsto R', \otimes', \theta)\) is a semi-group, closure is clear from the definition of \(\otimes'\). To show that \(\otimes'\) is associative we note that the only possible elements in \((\mu_1 \otimes' \mu_2) \otimes' \mu_3\) are those in \(((\text{dom} \mu_1) \cap (\text{dom} \mu_2)) \cap (\text{dom} \mu_3) = (\text{dom} \mu_1) \cap ((\text{dom} \mu_2) \cap (\text{dom} \mu_3))\) which are the only possible elements in \(\mu_1 \otimes' (\mu_2 \otimes' \mu_3)\). There are three possible cases:

1. \(i \in (\mu_1 \otimes' \mu_2) \otimes' \mu_3 \land i \notin \mu_1 \otimes' (\mu_2 \otimes' \mu_3)\)

Then the only way this can arise is if either \(\mu_2(i) \times \mu_3(i) = u\) or \(\mu_1(i) \times (\mu_2(i) \times \mu_3(i)) = u\). Recalling for a ring \((R, +, \times)\) that \(ru = u, \forall r \in R\) we have in both cases that \(\mu_1(i) \times (\mu_2(i) \times \mu_3(i)) = u\) thus using the associativity of \(R\) we have \((\mu_1(i) \times \mu_2(i)) \times \mu_3(i) = u\) which is a contradiction, thus case (1) does not arise.

The remaining sub-cases are similar.

From the associativity of \(R\) we have that \((\mu_1(i) \times \mu_2(i)) \times \mu_3(i) = \mu_1(i) \times (\mu_2(i) \times \mu_3(i))\), thus we have that \((\mu_1 \otimes' \mu_2) \otimes' \mu_3(i) = \mu_1 \otimes' (\mu_2 \otimes' \mu_3)(i)\), and thus \(\otimes'\) is associative.

It remains to prove that the left and right distributive laws hold:

\begin{align*}
\mu_1 \otimes' (\mu_2 \oplus' \mu_3) &= (\mu_1 \otimes' \mu_2) \oplus' (\mu_1 \otimes' \mu_3) \quad (B.22) \\
(\mu_2 \oplus' \mu_3) \otimes' \mu_1 &= (\mu_2 \otimes' \mu_1) \oplus' (\mu_3 \otimes' \mu_1) \quad (B.23)
\end{align*}
Suppose \( i \in \mu_1 \otimes \overset{\ominus}{\tau} (\mu_2 \oplus' \mu_3) \) then \( i \in \mu_1 \land i \in \mu_2 \otimes' \mu_3 \).

Now \( i \in \mu_2 \oplus' \mu_3 \) gives \( i \in \mu_2 \lor i \in \mu_3 \). Thus we have \( i \in \mu_1 \land (i \in \mu_2 \lor i \in \mu_3) \), or equivalently we have \( (i \in \mu_1 \land i \in \mu_2) \lor (i \in \mu_1 \land i \in \mu_3) \).

Furthermore, we note that \( i \in (\mu_1 \otimes \overset{\ominus}{\tau} \mu_2) \oplus' (\mu_1 \otimes \overset{\ominus}{\tau} \mu_3) \) yields that
\[
i \in (\mu_1 \otimes \overset{\ominus}{\tau} \mu_2) \lor i \in (\mu_1 \otimes \overset{\ominus}{\tau} \mu_3) \\
\Rightarrow (i \in \mu_1 \land i \in \mu_2) \lor (i \in \mu_1 \land i \in \mu_3)
\]

Thus the elements in \( \mu_1 \otimes \overset{\ominus}{\tau} (\mu_2 \oplus' \mu_3) \) is some subset of \((\mu_1 \cap \mu_2) \cup (\mu_1 \cap \mu_3)\). Similarly, the elements in \((\mu_1 \otimes \overset{\ominus}{\tau} \mu_2) \oplus' (\mu_1 \otimes \overset{\ominus}{\tau} \mu_3)\) are some subset of \((\mu_1 \cap \mu_2) \cup (\mu_1 \cap \mu_3)\). It is required to demonstrate that these elements represent the same subset, and secondly, equality must be proved. It is clear that for \( i \in \mu_1 \otimes \overset{\ominus}{\tau} (\mu_2 \oplus' \mu_3) \) that \( \mu_1(i) \times (\mu_2 \oplus' \mu_3)(i) \neq u \), by the definition of the \( \otimes \overset{\ominus}{\tau} \) operation. This implies that \( \mu_1(i) \neq u \cap (\mu_2 \oplus' \mu_3)(i) \neq u \), yielding the three obvious sub-cases.

Similarly, \( i \in (\mu_1 \otimes \overset{\ominus}{\tau} \mu_2) \oplus' (\mu_1 \otimes \overset{\ominus}{\tau} \mu_3) \) implies \((\mu_1(i) \times \mu_2(i) \neq u) \lor (\mu_1(i) \times \mu_3(i) \neq u)\) where \((i \in \mu_1)\) or \((i \in \mu_3)\), or \((\mu_1(i) \times \mu_2(i)) \lor (\mu_1(i) \times \mu_3(i)) \neq u\) and \((i \in \mu_2)\) and \((i \in \mu_3)\).

Considering the three subcases shows that \( i \in \mu_1 \otimes \overset{\ominus}{\tau} (\mu_2 \oplus' \mu_3) \) if and only \( i \in (\mu_1 \otimes \overset{\ominus}{\tau} \mu_2) \oplus' (\mu_1 \otimes \overset{\ominus}{\tau} \mu_3)\)(\( i \)). Finally, we show that these maps are exactly the same, and this is seen by considering the three cases:

1. \( i \in \mu_1 \land i \in \mu_2 \land i \not\in \mu_3 \)

Then we get \( \mu_1 \otimes \overset{\ominus}{\tau} (\mu_2 \oplus \mu_3)(i) = \mu_1(i) \times \mu_2(i) \\
= (\mu_1 \otimes \overset{\ominus}{\tau} \mu_2)(i) \\
= (\mu_1 \otimes \overset{\ominus}{\tau} \mu_2) \oplus' (\mu_1 \otimes \overset{\ominus}{\tau} \mu_3)(i) \) as required.

The remaining sub-cases are similar.

Thus \( \mu_1 \otimes \overset{\ominus}{\tau} (\mu_2 \oplus' \mu_3) = (\mu_1 \otimes \overset{\ominus}{\tau} \mu_2) \oplus' (\mu_1 \otimes \overset{\ominus}{\tau} \mu_3)(i) \) as required.

### B.9 Miscellaneous Algebraic structures

**Question B.1** Given an integral domain \((\mathbb{Z}, \times, +)\) does the structure \((I \mapsto \mathbb{Z}', \otimes', \oplus')\) form a quasi-integral domain?

The fundamental difference between a ring and an integral domain is that the latter structure has no zero divisors. We note that under our definition of \( \otimes' \) that if two partial maps with disjoint domains are multiplied then the result is the empty map \( \theta \), the \( 0 \) of the structure, i.e., \( \mu \otimes' \mu' = \theta \) where \( \text{dom} \mu \cap \text{dom} \mu' = \emptyset \). Thus the indexed structure fails to form an integral domain.

195
Given a field $(F, +, \times)$ does the structure $(I \rightarrow F', \otimes\!^\prime, \oplus')$ form a quasi indexed field.

Note
This is immediate since a field is an integral domain and we have seen that the indexed structure fails to be an indexed domain and consequently fails to be a field. Alternately this may be seen from the nature of the multiplicative identity $\mathcal{I}_v$ which is a total map of the form $I \mapsto v$ where $v$ is the multiplicative identity of $F$. From the definition of $\otimes\!^\prime$ if a partial map $\mu$ was to have an inverse then $\mu \otimes\!^\prime \mu^{-1}$ must yield the identity $\mathcal{I}_v$, but this is impossible as $\mu$ is a strict partial map, i.e., $\text{dom} \mu \subset \text{dom} \mathcal{I}_v$ and it is clear from the definition of $\otimes\!^\prime$ that $\text{dom}(\mu \otimes\!^\prime \mu') \subset \text{dom} \mu$, thus $\text{dom}(\mu \otimes\!^\prime \mu^{-1}) \subset \text{dom} \mathcal{I}_v$.

B.10 Total Functions - Indexed Structures

The difficulty in obtaining an indexed field of the form $(I \rightarrow F)$ from the base field $F$ is due to the nature of partial maps. If instead we consider total maps, i.e., elements of $F^I$, where this denotes the set of all total functions from $I$ to $F$, then under the operations $\otimes\!$, $\oplus$ $(I \rightarrow F)$ is an indexed field. In order to be convinced of this result we note from Theorem B.6 that all that is required is to show a unity element, and given any element $\mu \in F^I$ then $\exists ! \mu^{-1}$ such that $\mu \otimes\! \mu^{-1} = \mu^{-1} \otimes\! \mu^{-1} = \mathcal{I}$. We note that all elements in $F^I$ are total functions.

Theorem B.8 Given a field $F$, and $F^I$ the set of all total functions from $I$ to $F$ then $(F^I, \otimes\!, \oplus)$ yields a field termed the indexed field.

Proof
We note $\mathcal{I}$ is given by $\lambda i : I \mapsto v$ where $v$ is the multiplicative unit of the field, thus $\mathcal{I}(i) = v$ for all $i \in I$. Given $\mu \in F^I$ we must show that $\mu \otimes\! \mathcal{I} = \mathcal{I} \otimes\! \mu = \mu$. Clearly $\mu \otimes\! \mathcal{I} = \lambda i : I, \mu(i) \times v = \lambda i : I, \mu(i) = \mu$. Similarly we get $\mathcal{I} \otimes\! \mu = \mu$ and so $\mathcal{I}$ is the identity of $F^I$.

Next given $\mu = [i_1 \mapsto f_1, i_2 \mapsto f_2, \ldots, i_n \mapsto f_n]$ then defining $\mu^{-1}$ as $[i_1 \mapsto f_1^{-1}, i_2 \mapsto f_2^{-1}, \ldots, i_n \mapsto f_n^{-1}]$ where $f_i^{-1}$ is the multiplicative inverse of $f_i$ in field $F$ then it is clear that $\mu \otimes\! \mu^{-1} = \mathcal{I} = \mu^{-1} \otimes\! \mu$. Thus the theorem is proved as other parts follow from Theorem B.6.

Note:
It is important to realize that the additive identity of $F^I$ is not $\theta$ the zero of $(I \rightarrow F)$, rather it is given by $\lambda i : I \mapsto u$, where $u$ is the additive unit of $F$.  

196
B.11 Quasi Indexed Vector Spaces

The final structure examined here is the generalization of a vector space to an indexed vector space. A vector space [65] \( V \) over a field \( F \) is a commutative group under vector addition. Furthermore, the operation of a scalar times a vector is defined, and satisfies the following:

\[
\begin{align*}
\alpha (v_1 + v_2) &= \alpha v_1 + \alpha v_2 \\
(\alpha + \beta) v &= \alpha v + \beta v \\
(\alpha \beta) v &= \alpha (\beta v) \\
1 v &= v
\end{align*}
\]

Thus under the operation \( \oplus' \) it is clear that \( (I \mapsto V') \) forms the quasi indexed group. We form the field \( \bar{F} \subseteq F' \) where \( F' \) is as defined in Theorem B.8 and \( \bar{F} = \{ \bar{f} \in F' | \bar{f}(v) = f \} \).

We may regard \( \bar{F} \) as an embedding of \( F \) in \( F' \). Then we have that \( (I \mapsto V', \oplus', \theta) \) is a commutative group, thus we wish to show that the additional vector space properties hold.

**Theorem B.9** Given a vector space \( V \) over a field \( F \), with \( F' \), \( \bar{F} \) as defined previously, the structure \( (I \mapsto V', \oplus', \theta) \), \( (\bar{F}, \otimes') \) is the inherited vector space, termed the quasi indexed vector space.

**Proof**

First we show \( \bar{F} \otimes' (\mu_1 \oplus' \mu_2) = \bar{F} \otimes' \mu_1 \oplus' \bar{F} \otimes' \mu_2 \). Suppose \( i \in \bar{F} \otimes' (\mu_1 \oplus' \mu_2) \). Then we have \( i \in \mu_1 \lor i \in \mu_2 \) or both. Suppose \( i \in \mu_1, i \notin \mu_2 \) then \( (\mu_1 \oplus' \mu_2)(i) = \mu_1(i) \) and so \( \bar{F} \otimes' (\mu_1 \oplus' \mu_2)(i) = \alpha \mu_1(i) = (\bar{F} \otimes' \mu_1 \oplus' \bar{F} \otimes' \mu_2)(i) \). The other cases follow similarly.

Next we show \( (\bar{F} \oplus \bar{F}) \otimes' \mu = (\bar{F} \otimes' \mu) \oplus' (\bar{F} \otimes' \mu) \). Strictly speaking, we should employ different symbols for the \( \oplus \) additive operator of \( \bar{F} \) and the \( \oplus' \) additive operator of \( (I \mapsto V') \). Suppose \( i \in (\bar{F} \oplus \bar{F}) \otimes' \mu \) then \( ((\bar{F} \oplus \bar{F}) \otimes' \mu)(i) = (\alpha + \beta) \mu(i) = \alpha \mu(i) + \beta \mu(i) \). If \( i \in \bar{F} \otimes' \mu \) then \( \alpha \otimes' \mu(i) = \alpha \mu(i) \), otherwise \( \alpha \mu(i) = u \) the additive unity of \( V \), similarly for \( \bar{F} \otimes' \mu \). The result then follows by considering the various cases.

The next requirement is to show \( (\bar{F} \otimes' \bar{F}) \otimes' \mu = \bar{F} \otimes' (\bar{F} \otimes' \mu) \). Clearly, \( \bar{F} \otimes' \bar{F} = \bar{F} \otimes' \bar{F} \), where \( \alpha, \beta \neq u \) the additive identity of \( F \). Given \( i \in (\bar{F} \otimes' \bar{F}) \otimes' \mu \) we have \( ((\bar{F} \otimes' \bar{F}) \otimes' \mu)(i) = (\alpha \otimes' \beta)(i) \). Suppose \( i \in \bar{F} \otimes' (\bar{F} \otimes' \mu) \) then we get \( \alpha \otimes' (\bar{F} \otimes' \mu)(i) = \alpha \beta \mu(i) \). Thus if the domains of these two expressions are equal then the associativity property holds, this involves a consideration of the various cases.

Finally the last requirement is to show \( I \otimes' \mu = \mu \) where \( I \) is the multiplicative unit of \( \bar{F} \), and where \( 1 \) is the multiplicative unit of \( F \). This result is immediate.

**Note:**

Strictly speaking we should employ \( \otimes' \) for multiplication of elements of \( \bar{F} \), and reserve \( \otimes' \)
for operations between \( \mathcal{F} \) and \( (I \to V') \). This is since \( \mathcal{F} \) is a collection of total functions, and includes \( \tilde{a} = \lambda i : I.a \), where \( u \) is the additive identity of \( F \).

### B.12 Summary

The main conclusions on the study of indexed structures is the following.

- **The Indexed Monoid** \((I \to M, \Theta, \Theta)\) inherits its behaviour from the base monoid \((M, *, u)\). Note that the **unit** element \( u \) may be in the range of \( \mu \in (I \to M) \).

- **The Quasi Indexed Monoid** \((I \to M', \Theta', \Theta)\) inherits its behaviour from the base monoid \((M, *, u)\). Note however that its definition ensures that the **unit** element \( u \) is not in the range of \( \mu \in (I \to M') \).

- If the base monoid is commutative then the (quasi) indexed structure formed inherits this commutativity.

- If the base structure is a group, then the quasi indexed structure is a group, the indexed structure fails to be a group as its definition does not enable removal of entries of the form \([i \mapsto u]\), and thus there is no way an element may have an inverse under the indexed definition.

- If \( \Omega \) is a set of operators for the monoid \((M, *, u)\) then the (quasi) indexed monoid inherits \( \Omega \) as an indexed set of operators.

- If the base structure is a semi-ring \((R, +, \times, u)\), then we may form the indexed semi-ring \((I \to R, \Theta, \Theta, \Theta)\), where the \( \Theta \) operator produces a result based on the elements both maps have in common.

- If the base structure is a ring \((R, +, \times, u)\), then we may form the quasi indexed ring \((I \to R', \Theta', \Theta', \Theta)\), where the \( \Theta' \) operator produces a result based on the elements both maps have in common.

- If the base structure is a field \((F, +, \times, u, v)\), then by considering total functions \( F^I \) we may form the indexed field \((F^I, \Theta, \Theta, \Theta, I_u, I_v)\).

- Finally if the base structure is a vector space \((V, F)\), then we may form the quasi indexed vector space \( \{(I \to V', \Theta', \Theta), (\mathcal{F}, \Theta')\} \).

The key point to note from this study is the inheritance of properties from the underlying base structures, and secondly, the derived structures themselves have applications in formal methods.
Appendix C

The Free Group

C.1 Introduction

This section presents a very brief account of structures derived from the free group. A good introduction to group theory is available in Chapter 5, 6 of [3]. The definition of a group has been presented in Chapter 1. The free group is constructed from an arbitrary set $E$, where $E$ is a countable or an uncountable set. The fundamental theorem of the free group states that every group is isomorphic to a quotient group of a free group. The construction presented here is an adaptation of [55], and the words of the free group are described via sequences. The join operation ensures that a reduced word is produced from the join of two reduced words.

Let $E$ be a set, we construct the set $\tilde{E}$, a set with the same cardinality of $E$, and disjoint from $E$, and consider a bijection $b : E \mapsto \tilde{E}$, where for every $x \in E$, the image $b(x)$ is denoted by $\tilde{x}$. The element $\tilde{x}$ is called the formal inverse of $x$ and $\tilde{E}$ is the set of formal inverses of $E$. If an element $x$ of $E$ occurs adjacent to its formal inverse $\tilde{x}$, then the effect is annihilation.

The properties of alphabets and words are familiar from Automata Theory [19]. Given an alphabet $\Sigma$ then $w \in \Sigma^*$ is a word generated by the alphabet $\Sigma$. It is well known that $(\Sigma^*, \cdot, \Lambda)$ forms a monoid, termed the free monoid. The alphabet $\Sigma$ is given by $\Sigma = E \cup \tilde{E}$ for the free group. If adjacent $x$ and $\tilde{x}$ occur in a word of the free group, they are considered to annihilate one another, and thus the words which are of particular interest are the reduced words, i.e., words in which no adjacent $x$, $\tilde{x}$ occur.

Definition Every finite sequence of letters of $E \cup \tilde{E}$ is called a word, i.e., $w \in (E \cup \tilde{E})^*$. Of course, $\Lambda$ is the empty word.
Definition Given a word \( w \) then \( w \) is said to be a reduced word if given any \( x \in w \), then \( x \) does not appear adjacent to its formal inverse \( \bar{x} \) in \( w \).

Reduced words are the norm in the theory of free groups, such a word being denoted by \( w_R \), or usually just \( w \). The set of reduced words in \( (E \cup \bar{E})^* \) is denoted by \( \mathcal{M} \). In the theory of free monoids [45], a new word may be formed from the concatenation of two existing words; i.e., given \( s_1, s_2 \in \Sigma^* \) then \( s_1 \circ s_2 \) is a word. It is desired to derive an analogous operation for two reduced words, however, it must be ensured that the concatenation of two reduced words is again a reduced word. Thus it must be ensured that \( x \) and \( \bar{x} \) do not appear adjacent in \( w \) where \( w = w_1 \circ w_2 \) is the reduced word created by the concatenation of \( w_1 \) and \( w_2 \).

Thus a specialized concatenation binary operation \( \circ \) is defined on \( \mathcal{M} \) as follows; given two reduced words \( w_1, w_2 \) then \( w_1 \circ w_2 \) is defined as the reduced word formed by the juxtaposition of \( w_1 \) and \( w_2 \), and eliminating occurrences of \( x \) and \( \bar{x} \), whenever they are adjacent. This may be defined more concisely in algorithmic form as follows:

\[
\text{join} : (E \cup \bar{E})^* \mapsto (E \cup \bar{E})^* \mapsto (E \cup \bar{E})^* \\
\text{join} [\Lambda] w_2 \equiv w_2 \\
\text{join} [w_1] \Lambda \equiv w_1 \\
\text{join} [s_1 \circ <x>] (<y> \circ s_2) \equiv \\
(x \in E \land y \in \bar{E} \land \bar{x} = y) \\
\lor (x \in \bar{E} \land y \in E \land \bar{y} = x) \\
\Rightarrow \text{join} [s_1] s_2 \\
\Rightarrow s_1 \circ <x> \circ <y> \circ s_2
\]

The infix operator \( \circ \) is used with the understanding that it is identical with the prefix operation \( \text{join} \) as defined above. The structure \( (\mathcal{M}, \circ) \) is examined in order to determine its properties. It is, in fact, a group.

Free Group as a Group

The structure \( (\mathcal{M}, \circ) \) is a group. This result is stated without proof.

Theorem C.1 Given a set \( E \) and \( \bar{E} \) as defined, then the structure \( (\mathcal{M}, \circ) \) forms a group, where \( \mathcal{M} \) denotes the reduced words of \( (E \cup \bar{E})^* \) and \( \circ \) denotes concatenation of reduced words.
Note 1
We shall denote the free group \((\mathcal{M}, \odot)\) by \(\mathcal{L}(E)\), i.e. the free group generated by the set \(E\).

Definition The cardinality of \(E\) is is called the rank of the free group \(\mathcal{L}(E)\).

Note 2
\(\mathcal{L}(E)\) is non commutative for \(|E| > 1\). This is clear since the sequence \(< x, y > \neq < y, x >\)
for \(y \neq x\) and clearly \(< x > \odot < y > = < x, y >\) whereas \(< y > \odot < x > = < y, x >\). (We are of course restricting \(x, y\) such that \(x \in E\) and \(y \in E\).)

Lemma C.1 \(\psi : (E \cup \bar{E}) \mapsto (E \cup \bar{E})\) as defined below is a bijection.

\[
\psi(\sigma) = \begin{cases} 
  b(\sigma) & \sigma \in E \\
  b^{-1}(\sigma) & \sigma \in \bar{E}
\end{cases}
\] (C.1)

where \(b\) has been defined previously in section C.1.

Lemma C.2 Given \(s \in \mathcal{M}\) then \(\text{rev}(s) \in \mathcal{M}\).

Proof
This is easily seen as \(\sigma \in \text{rev} \ s\) iff \(\sigma \in s\), also \(\sigma \tau\) are adjacent in \(s\) iff \(\tau \sigma\) are adjacent in \(\text{rev} \ s\). Thus if \(y \bar{y}\) are adjacent in \(\text{rev} \ s\) then \(y \bar{y}\) are adjacent in \(s\), and thus \(s\) is a reduced word iff \(\text{rev} \ s\) is a reduced word.

Lemma C.3 Given \(\sigma \in (E \cup \bar{E})\) then \(< \sigma > \odot < \psi(\sigma) > = \Lambda\).

Proof
Clearly all that is required in the proof is to note that \(e \in E\) then \(\psi(e) = \bar{e}\) whereas \(e \in \bar{E}\) then \(\psi(e) \in E\). Thus \(\psi(\sigma)\) is the formal inverse of \(\sigma\) and vice-versa, thus \(< \sigma > \odot < \psi(\sigma) > = \Lambda\) by definition of the \(\odot\) operation.

Finally, the fundamental theorem of the free group states that every group is isomorphic to a quotient group of the free group. The proof uses the idea of a set of generators \((P)\) for a group and considers the set \(E\) to be a set equal in cardinality to \(P\). A homomorphism is then constructed from the free group \(\mathcal{L}(E)\) into the group \(G\), this is, in fact, an epimorphism. The kernel of the homomorphism identifies and a well known isomorphism theorem of group theory is invoked to provide the required conclusion.

Theorem C.2 (Fundamental Theorem of Free Groups) Every group is isomorphic to a quotient of a free group.
Note
Given that groups may be finite, countable or uncountable we note that the corresponding alphabet $E$ may be finite, countable or uncountable. We note $\mathbb{Z}$ is generated by $E = \{x\}$ a finite alphabet of one element.

C.2 The Free Set with Applications

This section introduces a structure which is an adaptation of the free group. However, its properties are not as well behaved as the free group. In particular, it is not associative. Consider a set $\Sigma$ and $\bar{\Sigma}$ as before; The set $A \subset (\Sigma \cup \bar{\Sigma})$ to be a free set if:

$$\forall a \in A \quad a \in \Sigma \implies \bar{a} \not\in A$$

$$a \in \bar{\Sigma} \implies \bar{a} \not\in A$$

Then we define the free union of free sets as follows:

$$A \cup_* B \triangleq \{x \mid x \in A, \bar{x} \not\in B \lor x \in B, \bar{x} \not\in A\}$$

Then let $S$ denote the free sets of $\Sigma \cup \bar{\Sigma}$. Then we note the following properties of $S, \cup_*$. 

1. **Closure**: The only way the closure property may fail is if we have $x$ and $\bar{x}$ in $A \cup_* B$, however from definition of $\cup_*$ the only way this can occur is if $x, \bar{x} \in A$ or $x, \bar{x} \in B$ which is impossible since $A, B$ are free sets.

2. **Not Associative**: Consider $A = \{a_1, a_2\}, B = \{\bar{a}_1, a_2\}, C = \{a_1, \bar{a}_2\}$, then we get:

$$A \cup_* B = \{a_2\}, B \cup_* C = \{\}$$

thus we get:

$$(A \cup_* B) \cup_* C = \{a_1\}$$

whereas $A \cup_* (B \cup_* C) = \{a_1, a_2\}$. 

3. **Identity Element**: Clearly $A \cup_* \emptyset = A = \emptyset \cup_* A$.

4. **Inverses**: Suppose $A = \{a_1, a_2, ..., a_n\}$ then defining $\bar{A}$ as $\bar{A} = \{\bar{a}_1, \bar{a}_2, ..., \bar{a}_n\}$ then it is clear $A \cup_* \bar{A} = \emptyset$ the identity of $S$.

5. **Commutative**: Clearly from definition of $\cup_*$ we have $A \cup_* B = B \cup_* A$.

Note The $\cup_*$ operator is commutative. Associativity is defined as left to right, i.e., $A \cup_* B \cup_* C$ is interpreted as $(A \cup_* B) \cup_* C$. 

202
Applications: Modelling Simple Graphs

The free list is considered in section C.3, and is suitable for modelling multigraphs. The
free set structure is suitable for modelling simple graphs, with the nodes \( V \) in the graph
is associated a set of inverse nodes \( \tilde{V} \). Node removal and update operations correspond to
free set union.

\[ \gamma : V \mapsto \mathcal{P}V \tag{C.3} \]

This model associates a set of nodes (possibly empty) with each node in the network; the
model is restricted to ensure that each node \( v' \) connected to node \( v \) is recorded in the
network. The constraint is specified as follows:

\[
\text{Inv\_SimGrp} : (V \mapsto \mathcal{P}V) \mapsto \mathbf{B}
\]

\[
\text{Inv\_SimGrp}[\gamma] \triangleq \\
\cup / \circ \text{rng} \gamma \subseteq \text{dom} \gamma \\
\mapsto \text{TRUE} \\
\mapsto \text{FALSE}
\]

The key operation presented here is the \( \text{Upd\_Node} \) operation which may connect a node
\( v \) to other nodes not previously connected to, or alternately it may remove an existing con-
nection between \( v \) and \( v' \). It is defined via free sets as follows:

\[
\text{Upd\_Node} : (V \mapsto \mathcal{P}V) \mapsto (V \times \mathcal{P}(V \cup \tilde{V})) \mapsto (V \mapsto \mathcal{P}V)
\]

\[
\text{Upd\_Node}[\gamma](v, K) \triangleq \gamma \uparrow [v \mapsto \gamma(v) \cup_* K]
\]

The precondition for this operation requires that the node specified for updating is al-
ready present in the network; secondly if the operation specifies new nodes for \( v \) to connect
to then these nodes must already be present in the network; finally if a node connection is
to be removed then the connection to be removed must already be present.

\[
\text{pre\_Upd\_Node} : (V \mapsto \mathcal{P}V) \mapsto (V \times \mathcal{P}(V \cup \tilde{V})) \mapsto \mathbf{B}
\]

\[
\text{pre\_Upd\_Node}[\gamma](v, K) \triangleq \\
v \in \gamma \\
\land \llbracket V \rrbracket K \subseteq \text{dom} \gamma \\
\land \forall \tilde{w} \in \llbracket \tilde{V} \rrbracket K \\
\land \text{Inv\_FreeSet}[K]
\]

203
Note
A set \( K \subset (\Sigma \cup \bar{\Sigma}) \) is a free set if:

\[
\text{Inv}_\text{FreeSet}[K] \triangleq \forall a \in K \quad a \in \Sigma \quad \Rightarrow \bar{a} \notin K \\
\quad a \in \bar{\Sigma} \quad \Rightarrow \bar{a} \notin K
\]

The \textit{Upd\_Node} operation has an associated proof obligation which ensures that the resultant structure is a network. This proof obligation is stated as the following theorem.

\textbf{Theorem C.3} \( \text{pre}_{\text{Upd\_Node}[\gamma]}(v, K) \land \text{Upd\_Node}[\gamma](v, K) \Rightarrow \text{Inv}_\text{SimGrp}[\text{Upd\_Node}[\gamma](v, K)] \)

### C.3 Free List:- Definition

The free list differs from the free set in that it allows multiple elements. However, it differs from multisets in that there is a specific order for element removal where duplicates of elements are in a list. Essentially, its operation enables multiple deletion and addition of elements to take place in a single operation. For example, the list \( L = [x, y, z, x, x, z, x, \bar{s}] \) is a free list.

We shall consider a list \( L \in (\Sigma \cup \bar{\Sigma})^* \) to be a free list if \( L \) satisfies the following property:

\[
\text{Inv}_\text{FreeList}[A] \triangleq \forall a \in \text{elems}(A) \quad \neg \bar{a} \in \text{elems}(A)
\]

where \( a \in \Sigma \Rightarrow \bar{a} \in \bar{\Sigma} \) and \( \bar{\bar{a}} = a \). The concatenation operation \textit{Conc} ensures whenever that whenever two free lists \( L_1, L_2 \) are joined then \( \text{Conc}[L_1]L_2 \) or equivalently in the infix form \( L_1 \sim L_2 \) is a free list.

\[
\text{Conc} : (\Sigma \cup \bar{\Sigma})^* \mapsto (\Sigma \cup \bar{\Sigma})^* \mapsto (\Sigma \cup \bar{\Sigma})^*
\]

\[
\text{Conc}[\Lambda]L \triangleq L \\
\text{Conc}[L] \Lambda \triangleq L
\]

\[
\text{Conc}[L_1](\langle \sigma \rangle \sim L_2) \triangleq \\
\bar{\sigma} \in \text{elems}(L_1) \\
L_1' \mapsto \text{StkRem}[ar{\sigma}]L_1 \\
\mapsto \text{Conc}[L_1']L_2 \\
\mapsto \text{Conc}[L_1 \sim \langle \sigma \rangle]L_2
\]

where the \textit{StkRem} operation removes the last occurrence of an element \( \sigma \) from a list; it is defined as follows:
\[ \text{Stk Rem} : (\Sigma \cup \bar{\Sigma}) \mapsto (\Sigma \cup \bar{\Sigma})^* \mapsto (\Sigma \cup \bar{\Sigma})^* \]
\[ \text{Stk Rem}[\sigma] \triangleq \Lambda \]
\[ \text{Stk Rem}[\sigma](\tau \triangleq \sigma') \triangleq \]
\[ \quad \mapsto \tau \]
\[ \quad \mapsto (\text{Stk Rem}[\sigma]\tau) \triangleq \sigma' \]

Note:
Denoting the set of free lists by \( \mathcal{L} \) we note that \( (\mathcal{L}, \triangle) \) forms a structure which is not a group since \( \triangle \) is not associative. However the other group properties hold.

Applications: Networks and Graph

Graphs have traditionally been modelled in the Irish school of VDM via models such as \( V \mapsto \mathcal{P}V \) or \( V \mapsto \mathcal{P}^2V \) or simply \( \mathcal{P}^2V \). These models do not successfully address multigraphs, where two nodes may have several connections. The approach to modelling multi-graphs considered here is to consider the application of the free list structure to this problem. The approach taken is to associate a set of inverse nodes \( \mathcal{V} \) with the set of nodes \( V \). Node removal and update operations then correspond to a free list concatenation.

\[
\gamma : V \mapsto V^* \quad \text{(C.4)}
\]

This model associates a list of nodes (possibly empty) with each node in the network; the model is restricted to ensure that each node \( v' \) connected to node \( v \) is already present in the network. This constraint is specified in terms of an image of the graph representation. In particular, \( (I \mapsto \text{elems}) \) is applied to \( \gamma \), and yields \( \gamma_r : V \mapsto \mathcal{P}V \), the constraint is then specified in terms of \( \gamma_r \).

\[
\text{Inv}_\text{MltGrp} : (V \mapsto V^*) \mapsto B
\]
\[
\text{Inv}_\text{MltGrp}[\gamma] \triangleq
\]

Let \( \gamma_r = (I \mapsto \text{elems})\gamma \) in

\[
\cup / \circ \text{rng} \gamma_r \subseteq \text{dom} \gamma_r
\]
\[ \mapsto \text{TRUE} \]
\[ \mapsto \text{FALSE} \]

The elements in the free list are of the form \( (V \cup \mathcal{V})^* \), as distinct from \( V^* \) which is the multi-graph model. However, it is in node removal that the free list structure is used, and constraints are placed on the node removal operation to ensure that the resultant graph after a node removal operation is strictly of the form \( V \mapsto V^* \) and not \( V \mapsto (V \cup \mathcal{V})^* \). Thus it
is ensured that the resultant graph does not contain entries of the form \( v \mapsto [v_1, v_2, \ldots \bar{v}, \ldots v_k] \) where \( \bar{v} \in \bar{V} \).

Next we present the multigraph network operations; the create network operation is immediate, and is given by:

\[
\text{Create}\_\text{Netwk} \mapsto (V \mapsto V^*)
\]

\[
\text{Create}\_\text{Netwk} \triangleq \emptyset
\]

It is assumed that the network is built bottom up; thus, whenever a new node is added to the network it must be ensured that the node is not already present in the network, and furthermore, the nodes to which the new node is to connect must already be present in the network. The \text{Add}\_\text{Node} operation and its precondition are defined as follows:

\[
\text{Add}\_\text{Node} : V \times V^* \mapsto (V \mapsto V^*) \mapsto (V \mapsto V^*)
\]

\[
\text{Add}\_\text{Node}[v, L] \gamma \triangleq \gamma \sqcup [v \mapsto L]
\]

\[
\text{pre}\_\text{Add}\_\text{Node} : V \times V^* \mapsto (V \mapsto V^*) \mapsto \mathbb{B}
\]

\[
\text{pre}\_\text{Add}\_\text{Node}[v, L] \gamma \triangleq
\]

\[
v \notin \gamma
\]

\[
\wedge \text{elems}(L) \subseteq \text{dom} \gamma
\]

The use of the list mechanism ensures that multigraphs are modelled effectively, e.g., consider \( v \mapsto [v_1, v_2, v_1, v_3, v_1] \) then this essentially states that there are three links from \( v \) to \( v_1 \), and a single link from \( v \) to \( v_2 \), and from \( v \) to \( v_3 \). The following proof obligation arises for the \text{Add}\_\text{Node} operation:

\textbf{Theorem C.4} \text{pre}\_\text{Add}\_\text{Node}[v, L] \gamma \wedge \text{Add}\_\text{Node}[v, L] \gamma \Rightarrow \text{Inv}\_\text{MltGrp}[\text{Add}\_\text{Node}[v, L] \gamma]

\textbf{Proof}

Let \( \gamma = \text{Add}\_\text{Node}[v, L] \gamma \) and let \( \gamma_r = (I \mapsto \text{elems}) \gamma \). Then clearly we have \( \gamma_r = (I \mapsto \text{elems}) \gamma \sqcup (I \mapsto \text{elems}) [v \mapsto L] \). Thus we get \( \cup / \circ \text{rng} \gamma_r = \cup / \circ \text{rng} \gamma_r \cup \text{elems}(L) \subseteq \text{dom} \gamma_r \cup \text{elems}(L) \subseteq \text{dom} \gamma_r \) by the precondition. Clearly, \( \text{dom} \gamma_r \subseteq \text{dom} \gamma_r \) and so we have \( \cup / \circ \text{rng} \gamma_r \subseteq \text{dom} \gamma_r \) as required.

Next we present the \text{Upd}\_\text{Node} operation which may connect a node \( v \) to other nodes not previously connected to, alternately it may add an extra connection between an existing link from \( v \) to \( v' \) say; this operation may also remove an existing link from \( v \) to \( v' \). This operation is best captured via the free list as follows:
\[Upd_{\text{Node}} : (V \mapsto V^*) \mapsto (V \times (V \cup \tilde{V})^*) \mapsto (V \mapsto V^*)\]
\[Upd_{\text{Node}}[\gamma](v, K) \triangleq \gamma \upharpoonright [v \mapsto \gamma(v) \setminus K]\]

The precondition for this operation is quite involved; firstly, it is required to ensure that
the node specified for updating is already present in the network; secondly, if the operation
specifies new nodes for \(v\) to connect to, then these must already be present in the network;
finally, if node deletion is to be performed for some of the nodes that \(v\) is connected to then
the effect of the operation must ensure that if \(n\) occurrences of node \(v'\) is to be removed
from node \(v\), then there must be at least \(n\) connections from \(v\) to \(v'\).

\[\text{pre}_{Upd_{\text{Node}}} : (V \mapsto V^*) \mapsto (V \times (V \cup \tilde{V})^*) \mapsto \mathcal{B}\]
\[\text{pre}_{Upd_{\text{Node}}}[\gamma](v, K) \triangleq \]
\[v \in \gamma\]
\[\land \forall \tilde{w} \in \tilde{V} \text{elem}(K)\]
\[\text{Occ}[\tilde{w}]K \leq \text{Occ}[w]\gamma(v)\]
\[\land \text{Inv}\_\text{FreeList}[K]\]

where \(\text{Occ}[\tilde{w}]K\) determines the number of occurrences of \(\tilde{w}\) in \(K\), i.e., the number of
connections from \(v\) to \(w\) to be removed. The precondition stipulates that this must be less
than or equal to the number of connections currently available from \(v\) to \(w\) as given by
\(\text{Occ}[w]\gamma(v)\). The \(\text{Occ}\) operation is defined on an arbitrary alphabet \(\Sigma\) as follows:

\[\text{Occ} : \Sigma \mapsto \Sigma^* \mapsto \mathbb{N}\]
\[\text{Occ}[\sigma]L \triangleq \text{Occ}[\sigma, L]0\]

\[\text{Occ} : (\Sigma \times \Sigma^*) \mapsto \mathbb{N} \mapsto \mathbb{N}\]
\[\text{Occ}[\sigma, \Lambda](n) \triangleq n\]
\[\text{Occ}[\sigma, \langle \sigma' \rangle \setminus L'](n) \triangleq\]
\[\sigma = \sigma'\]
\[\mapsto \text{Occ}[\sigma, L'](n + 1)\]
\[\mapsto \text{Occ}[\sigma, L'](n)\]

The \(Upd_{\text{Node}}\) operation has an associated proof obligation which ensures that the
resultant structure is a network. We state this proof obligation as a theorem.

**Theorem C.5** \(\text{pre}_{Upd_{\text{Node}}}[\gamma](v, K) \land Upd_{\text{Node}}[\gamma](v, K) \Rightarrow\]
\[\text{Inv}\_\text{MltGrp}[Upd_{\text{Node}}[\gamma](v, K)]\]

207
Proof (Informal)
Informally it is clear that the proof is completed if we can show \(\text{elems} \circ \gamma(v) \subseteq \text{dom} \gamma\) as the node \(v\) is the only node directly affected by the \(\text{Upd\_Node}\) operation. Given the precondition for \(\text{Occ}[w]\) it is clear that no \(\bar{w} \in \bar{V}\) is in \(\gamma(v) \subseteq K\). Similarly, when \(w \in \text{elems}(K), w \in V\) then \(w \in \text{elems} \circ \gamma(v) \subseteq K\) with the precondition guaranteeing that \(w \in \text{dom} \gamma\). We proceed to a rigorous proof based on this informal argument.

We let \(\bar{\gamma} = \text{Upd\_Node}[\gamma](v, K)\) and we suppose \(\text{Inv\_MltGrp}\) is violated, then we have some \(w \in \cup/\circ \text{rng} \bar{\gamma}\), and \(w \not\in \text{dom} \bar{\gamma}\). Clearly as \(\text{dom} \bar{\gamma} = \text{dom} \gamma\) we can not have \(w \in \text{elems} \circ \bar{\gamma}(v') = \text{elems} \circ \gamma(v')\) for \(v \neq v'\) as otherwise the original invariant is violated for \(\gamma\). Thus \(w \in \text{elems} \circ \gamma(v) \subseteq K\) and since \(\gamma\) satisfies the invariant we must have \(w \in \text{elems}(K)\).

Suppose \(w \in V\) then we have \(w \in \text{elems}(K) \wedge w \in V\) thus we have \(w \in \llbracket V \rrbracket \text{elems}(K)\); however by the precondition we have \(\llbracket V \rrbracket \text{elems}(K) \subseteq \text{dom} \gamma\) and so \(w \in \text{dom} \gamma\) which is a contradiction. Thus \(w \not\in V\).

Thus \(w \in \bar{V}\), i.e. \(w = \bar{u}, u \in V\) and thus we have \(\bar{u} \in \text{elems}(K)\) and \(\bar{u} \in \gamma(v) \subseteq K\).
By the precondition we have \(\text{Occ}[\bar{u}] K \leq \text{Occ}[u] \gamma(v)\) thus the conditions for Lemma C.4 hold and we get \(\bar{u} \not\in \gamma(v) \subseteq K\) which is a contradiction. Thus the assumption that the invariant is violated yields a contradiction and thus the supposition that the invariant is violated is false and thus the operation therefore preserves the invariant.

Lemma C.4 Suppose \(L\) and \(K\) are free lists and \(w \in \text{elems}(L)\) and \(\bar{w} \in \text{elems}(K)\) with \(\text{Occ}[w] L \geq \text{Occ}[\bar{w}]\) then we have \(\bar{w} \not\in \text{elems} \circ L \subseteq K\).

Proof
Informally from the definition of the \(\subseteq\) operation this result is immediate as each \(\bar{w}\) in \(K\) annihilates exactly one \(w\) in \(L\). Since there are at least as many \(w\)'s as \(\bar{w}\)'s it is not possible for \(\bar{w}\) to be in \(L \subseteq K\). A more rigorous argument is provided by induction.

The basis case is \(n = 1\), i.e. \(\text{Occ}[w] L = 1\) and therefore \(\text{Occ}[\bar{w}] K = 1\) or 0. The result is immediate if \(\text{Occ}[\bar{w}] = 0\) thus we assume \(\text{Occ}[\bar{w}] = 1\). We have defined \(\subseteq\) to be left associative thus we get

\[
L \subseteq K = (L \subseteq K') \subseteq (\langle \bar{w} \rangle \subseteq K_1)
\]

where \(K = K' \subseteq \langle \bar{w} \rangle \subseteq K_1\) and where neither \(w\) nor \(\bar{w}\) are present in \(K'\) or \(K_1\), thus exactly one \(w\) is present in \(L \subseteq K'\). From the definition of \(\subseteq\) we see that \(\bar{w}\) annihilates the only \(w\) in \(L \subseteq K'\) and as there is no \(w\) or \(\bar{w}\) in \(K_1\) we get neither \(w\) nor \(\bar{w}\) are present in \(L \subseteq K\).
We assume the result is true whenever $\text{Occ}[[w]]L = k$ and $\text{Occ}[[\bar{w}]]K \leq k$. We consider $L$ such that $\text{Occ}[[w]]L = k + 1$. Then we have $\text{Occ}[[\bar{w}]]K \leq k + 1$. If $\text{Occ}[[\bar{w}]]K = 0$ then clearly it is impossible for $\bar{w} \in \text{elems} \circ L \sim K$. Thus we suppose $1 \leq \text{Occ}[[\bar{w}]]K \leq k + 1$ and be get:

$$L \sim K = (L \sim K') \sim ((\bar{w} \sim K_1)$$

where $K = K' \sim (\bar{w}) \sim K_1$ and where $w$ is not present in $K'$ or $K_1$, and $\bar{w}$ is not present in $K'$ and $\text{Occ}[[\bar{w}]]K_1 \leq k$. From the definition of $\sim$, we get $\text{Occ}[[w]](L \sim K') \sim (\bar{w}) = k$ and $\text{Occ}[[\bar{w}]]K_1 \leq k$ thus the inductive step ensures that $\bar{w} \notin \text{elems} \circ L \sim K$. The proof thus follows by induction.

**Example**

To ensure that the concept of a free list is fully understood we present the following example. We present a multigraph $\gamma$ followed by an update operation which yields $\bar{\gamma}$.

$$\begin{align*}
\gamma & = \langle v_1 \mapsto \{v_2 \sim v_2 \sim v_3 \sim v_2\} \\
v_2 & \mapsto \{v_2 \sim v_3 \sim v_1 \sim v_3\} \\
v_3 & \mapsto \{v_2 \sim v_1\} \\
v_4 & \mapsto \Lambda
\end{align*}$$

We note that this is a multigraph; there are three connections from $v_1$ to $v_2$ and one from $v_1$ to $v_3$, etc. We consider an update operation on $v_1$ which removes one link from $v_1$ to $v_2$ and adds a link from $v_1$ to $v_4$. In this case the update operation is $\text{Upd\_Node}[v_1, K]_{\gamma}$ where $K$ is defined as:

$$K = [v_2 \sim v_4]$$

and we get $\text{Upd\_Node}[v, K]_{\gamma}$ yields:

$$\bar{\gamma} = \text{Upd\_Node}[[v, K]]_{\gamma} = \begin{align*}
v_1 & \mapsto \{v_2 \sim v_2 \sim v_3 \sim v_4\} \\
v_2 & \mapsto \{v_2 \sim v_3 \sim v_1 \sim v_3\} \\
v_3 & \mapsto \{v_2 \sim v_1\} \\
v_4 & \mapsto \Lambda
\end{align*}$$

Thus the $\text{Upd\_Node}$ operation removes one connection from $v_1$ to $v_2$ and adds a connection from $v_1$ to $v_4$.

**C.4 Summary**

The objective of this appendix is to present a constructive approach to the definition of the free group and to present the free set and free list structures. The free set and free
list were identified during a generalization of the free group, and several applications of these structures have been identified. For example, there is an application of the free group structure to reversible systems. Furthermore, the free group and free set may be applied to model aspects of the scope rules of programming languages, and in modelling simple graphs and multigraphs.

The use of the free list structure is an elegant means of modelling multi-graphs; in particular, the $Upd\_Node$ operation is very concise, allowing extra connections and connection removal to be expressed in a neat algebraic manner. The theoretical foundation of the free list and free set ensures that these results are built from a solid foundation. The free set is useful when dealing with simple graphs.

**Conclusion C.1** The free set and free list are structures which enrich expressibility power, and provide a terse means of modelling connection addition and removal operations in a single operation in graph theory.
## Appendix D

### Glossary of VDM\textsuperscript{\textcopyright} Notation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\mathbf{B})</td>
<td>the Boolean domain, (\mathbf{B} = {\text{False, True}})</td>
</tr>
<tr>
<td>(\mathbf{N})</td>
<td>the Natural Numbers domain, (\mathbf{N} = {0, 1, 2, \ldots})</td>
</tr>
<tr>
<td>(\mathbf{N}_1)</td>
<td>the positive Natural Numbers, (\mathbf{N}_1 = {1, 2, \ldots})</td>
</tr>
<tr>
<td>(\mathbf{Z})</td>
<td>the Integers</td>
</tr>
<tr>
<td>(\mathbf{Q})</td>
<td>the Rational Numbers</td>
</tr>
<tr>
<td>(\mathbf{R})</td>
<td>the Real Numbers</td>
</tr>
<tr>
<td>(X^*)</td>
<td>Finite Sequences over (X)</td>
</tr>
<tr>
<td>(\mathcal{P}S)</td>
<td>the power set of a set (S).</td>
</tr>
<tr>
<td>(\mathcal{P}'S)</td>
<td>the power set of (S) excluding the empty set</td>
</tr>
<tr>
<td>(X^*)</td>
<td>domain of finite sequences over (X)</td>
</tr>
<tr>
<td>(X^*_i)</td>
<td>domain of unique finite sequences over (X)</td>
</tr>
<tr>
<td>(X^+)</td>
<td>domain of finite (non-empty) Sequences over (X)</td>
</tr>
<tr>
<td>(X^n)</td>
<td>domain of (n)-tuples.</td>
</tr>
<tr>
<td>(X \mapsto Y)</td>
<td>Domain of partial functions from (X) to (Y)</td>
</tr>
<tr>
<td>(X \mapsto \mathbf{N}_1)</td>
<td>Bag Domain</td>
</tr>
<tr>
<td>(f[x]y)</td>
<td>Function (f) applied to (curried) (x), applied to (y)</td>
</tr>
<tr>
<td>(\uparrow)</td>
<td>Map Override operator</td>
</tr>
<tr>
<td>(\mu(x))</td>
<td>Map Lookup, returning the element in the range mapped to by (x)</td>
</tr>
<tr>
<td>(I)</td>
<td>The Identity Function</td>
</tr>
<tr>
<td>(\oplus)</td>
<td>Reduction w.r.t. binary operation (\oplus)</td>
</tr>
<tr>
<td>(\land)</td>
<td>Logical And</td>
</tr>
<tr>
<td>(\lor)</td>
<td>Logical Or</td>
</tr>
<tr>
<td>(\neg)</td>
<td>Logical Negation.</td>
</tr>
<tr>
<td>Symbol</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
</tr>
<tr>
<td>$\circ$</td>
<td>Function Composition</td>
</tr>
<tr>
<td>$f$</td>
<td>Mapping function $f$</td>
</tr>
<tr>
<td>$\pi_n$</td>
<td>$n$th Projection Function</td>
</tr>
<tr>
<td>$\text{rng}$</td>
<td>Map Range</td>
</tr>
<tr>
<td>$\text{dom}$</td>
<td>Map domain</td>
</tr>
<tr>
<td>$(f \mapsto g)$</td>
<td>Maps $f$ and $g$ to Domain and Range resp. of a Map</td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>The empty set</td>
</tr>
<tr>
<td>$\Lambda$</td>
<td>The Null Sequence</td>
</tr>
<tr>
<td>$:$</td>
<td>The Sequence ‘Cons’ Operator</td>
</tr>
<tr>
<td>$\text{rev}$</td>
<td>The reverse of a sequence</td>
</tr>
<tr>
<td>$[\ldots]$</td>
<td>Sequence Subrange operator</td>
</tr>
<tr>
<td>$f^*$</td>
<td>Maps $f$ over a Sequence (Kleene Star functor)</td>
</tr>
<tr>
<td>$</td>
<td></td>
</tr>
<tr>
<td>$#$</td>
<td>The sequence length operator</td>
</tr>
<tr>
<td>$\text{elems}$</td>
<td>The elements in a sequence.</td>
</tr>
<tr>
<td>$\in$</td>
<td>The set membership operator.</td>
</tr>
<tr>
<td>$\chi$</td>
<td>The characteristic function membership operator.</td>
</tr>
<tr>
<td>$\sqcup$</td>
<td>Disjoint union operation.</td>
</tr>
<tr>
<td>$\cup$</td>
<td>The set union operation.</td>
</tr>
<tr>
<td>$\cap$</td>
<td>The set intersection operation.</td>
</tr>
<tr>
<td>$\subseteq$</td>
<td>The Subset relation</td>
</tr>
<tr>
<td>$\Rightarrow$</td>
<td>Logical Implication</td>
</tr>
<tr>
<td>$\bowtie$</td>
<td>Sequence Concatenation operator</td>
</tr>
<tr>
<td>$\leftarrow$</td>
<td>Set removal operator</td>
</tr>
<tr>
<td>$\triangleleft$</td>
<td>Set restriction operator</td>
</tr>
<tr>
<td>$\otimes$</td>
<td>Indexed operator</td>
</tr>
<tr>
<td>$\sqcup$</td>
<td>Indexed union operator</td>
</tr>
<tr>
<td>$\perp$</td>
<td>Undefined, or bottom element, or do not care.</td>
</tr>
</tbody>
</table>
Bibliography


214


215


217
