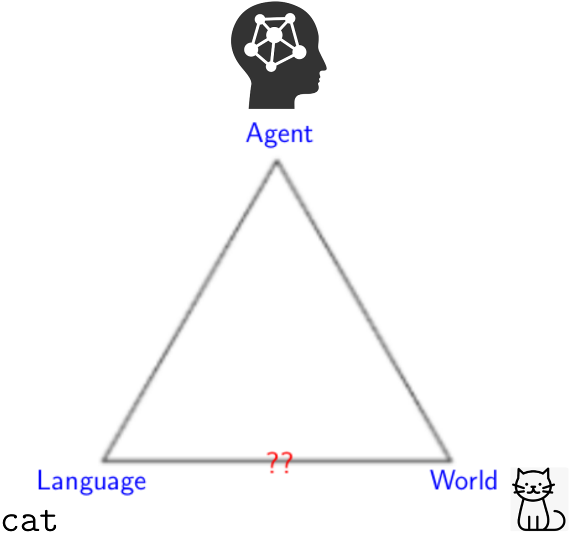
Triadic Temporal Representations & Deformations

Tim.Fernando@tcd.ie

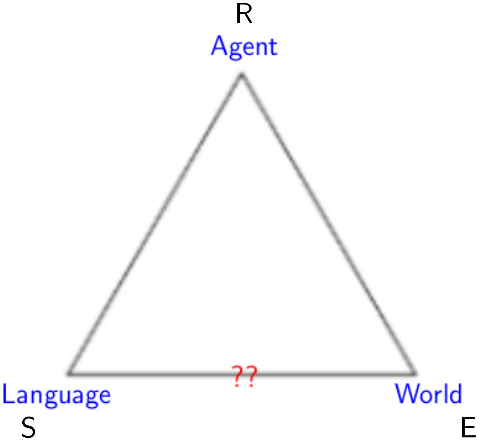
Nancy, 20 June 2023

Triadic Temporal Representations & Deformations



Triadic Temporal Representations
& Deformations

REICHENBACH: tense (S-R) & aspect (R-E)



Triadic Temporal Representations & Deformations

representations $\rightsquigarrow$ patterns

Pattern Theory, formulated by Ulf Grenander, is a mathematical formalism to describe knowledge of the world as patterns.

- Wikipedia

Triadic Temporal Representations & Deformations

representations $\rightsquigarrow$ patterns

Pattern Theory, formulated by Ulf Grenander, is a mathematical formalism to describe knowledge of the world as patterns.

- Wikipedia

the pattern should not merely describe the 'pure' situation that underlies reality but the 'deformed' situation that is actually observed in which the pure pattern may be hard to recognize. This generalizes, for example, Chomsky's idea of the deep structure of an utterance vs. its surface structure, where deep $\sim$ pure and surface $\sim$ deformed.

- Mumford 2019

Triadic Temporal Representations & Deformations

David Bryant Mumford (born 11 June 1937) is an American mathematician known for his work in algebraic geometry and then for research into vision and pattern theory. He won the Fields Medal and was a MacArthur Fellow. In 2010 he was awarded the National Medal of Science.

David Mumford

David Mumford in 2010

pattern ≈ pure situation + deformations

e.g., output ≈ input + noise (Shannon noisy channel)

(1) `Facebook bought Instagram.`

(2) `Facebook owns Instagram.`

(1) `Facebook bought Instagram.` facebook $\xrightarrow{\text{bought}}$ instagram

(2) `Facebook owns Instagram.` facebook $\xrightarrow{\text{owns}}$ instagram

(3) $\boxed{\text{bought}(x, y)} \implies \boxed{\text{owns}(x, y)}$ (Hosseini 2020)

(1) `Facebook bought Instagram.` $\boxed{\text{facebook}} \xrightarrow{\text{bought}} \boxed{\text{instagram}}$

(2) `Facebook owns Instagram.` $\boxed{\text{facebook}} \xrightarrow{\text{owns}} \boxed{\text{instagram}}$

(3) $\boxed{\text{bought}(x, y)} \Longrightarrow \boxed{\text{owns}(x, y)}$ (Hosseini 2020)

(4) $\boxed{\text{buy}(x, y)} \Longrightarrow \boxed{\textsc{Become}(\text{own}(x, y))}$ (Dowty 1979)

(1) `Facebook bought Instagram.` facebook $\xrightarrow{\text{bought}}$ instagram

(2) `Facebook owns Instagram.` facebook $\xrightarrow{\text{owns}}$ instagram

(3) $\boxed{\text{bought}(x, y)} \implies \boxed{\text{owns}(x, y)}$ (Hosseini 2020)

(4) $\boxed{\text{buy}(x, y)} \implies \boxed{\textsc{Become}(\text{own}(x, y))}$ (Dowty 1979)

(5) $\boxed{\neg\text{own}(x, y)} \xrightarrow{\text{buy}(x,y)} \boxed{\text{own}(x, y)}$ transition

(1) `Facebook bought Instagram.` $\boxed{\text{facebook}} \xrightarrow{\text{bought}} \boxed{\text{instagram}}$

(2) `Facebook owns Instagram.` $\boxed{\text{facebook}} \xrightarrow{\text{owns}} \boxed{\text{instagram}}$

(3) $\boxed{\text{bought}(x, y)} \implies \boxed{\text{owns}(x, y)}$ (Hosseini 2020)

(4) $\boxed{\text{buy}(x, y)} \implies \boxed{\textsc{Become}(\text{own}(x, y))}$ (Dowty 1979)

(5) $\boxed{\neg\text{own}(x, y)} \xrightarrow{\text{buy}(x,y)} \boxed{\text{own}(x, y)}$ finite automaton?

(6) $\boxed{\text{buy}(x, y)} \implies \boxed{\text{pay-for}(x, y)}$ ... open-ended

(1) `Facebook bought Instagram.` $\boxed{\text{facebook}} \xrightarrow{\text{bought}} \boxed{\text{instagram}}$

(2) `Facebook owns Instagram.` $\boxed{\text{facebook}} \xrightarrow{\text{owns}} \boxed{\text{instagram}}$

(3) $\boxed{\text{bought}(x, y)} \Longrightarrow \boxed{\text{owns}(x, y)}$ (Hosseini 2020)

(4) $\boxed{\text{buy}(x, y)} \Longrightarrow \boxed{\textsc{Become}(\text{own}(x, y))}$ (Dowty 1979)

(5) $\boxed{\neg\text{own}(x, y)} \xrightarrow{\text{buy}(x,y)} \boxed{\text{own}(x, y)}$ finite automaton?

(6) $\boxed{\text{buy}(x, y)} \Longrightarrow \boxed{\text{pay-for}(x, y)}$ . . . open-ended

Proposal: extract finite automata from knowledge graphs

(1) `Facebook bought Instagram.` $\boxed{\text{facebook}} \xrightarrow{\text{bought}} \boxed{\text{instagram}}$

(2) `Facebook owns Instagram.` $\boxed{\text{facebook}} \xrightarrow{\text{owns}} \boxed{\text{instagram}}$

(3) $\boxed{\text{bought}(x, y)} \implies \boxed{\text{owns}(x, y)}$ (Hosseini 2020)

(4) $\boxed{\text{buy}(x, y)} \implies \boxed{\textsc{Become}(\text{own}(x, y))}$ (Dowty 1979)

(5) $\boxed{\neg\text{own}(x, y)} \xrightarrow{\text{buy}(x,y)} \boxed{\text{own}(x, y)}$ finite automaton?

(6) $\boxed{\text{buy}(x, y)} \implies \boxed{\text{pay-for}(x, y)}$ ... open-ended

Proposal: extract finite automata from knowledge graphs,

allowing for refinements and alternatives

Deformations: institution as triad (Goguen)

Talk Outline

§1 Transitions from finite automata

§2 Strings as compressed models

§3 Granularity: sigs & reducts

§4 Deformations: institution as triad (Goguen)

Talk Outline

§1 Transitions from finite automata

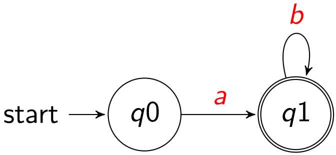
§2 Strings as compressed models

§3 Granularity: sigs & reducts

§4 Deformations: institution as triad (Goguen)

Labelled transition $q \xrightarrow{a} q'$

	automata		
q	state		
a	symbol		



$$q0 \xrightarrow{a} q1 \in F$$

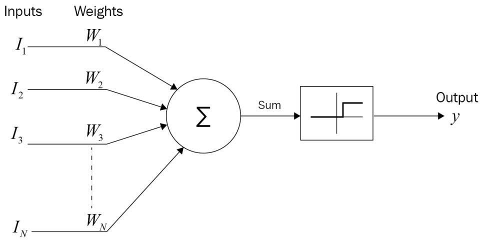
$$q0 \xrightarrow{a} q1 \xrightarrow{b} q1 \in F$$

$$\vdots$$

regular expression ab^*

Labelled transition $q \xrightarrow{a} q'$

	automata	Kleene 1956 (nerve nets)	
q	state	$(v_1, \ldots, v_m)$	
a	symbol	$\{$active input cells$\}$	



$$I_1, I_2 \cdots I_N = q, a$$

$(v_1 \ldots v_m)$ records the values v_i of m *inner cells*

output $q' = (y_1 \ldots y_m)$

Labelled transition $q \xrightarrow{a} q'$

	automata	Kleene 1956 (nerve nets)	action languages
q	state	$(v_1, \ldots, v_m)$	fluent, value
a	symbol	{active input cells}	(elementary) action

Gelfond & Lifschitz 1998 . . . symbolic AI (J. McCarthy)

Labelled transition $q \xrightarrow{a} q'$

	automata	Kleene 1956 (nerve nets)	action languages
q	state	$(v_1, \ldots, v_m)$	fluent, value
a	symbol	{active input cells}	(elementary) action

Gelfond & Lifschitz 1998 ... symbolic AI (J. McCarthy)
- *fluent* (Newton), inertia (frame problem)
- *action signature* $(\mathbf{V}, \mathbf{F}, \mathbf{A})$

  names $\mathbf{A}$ for actions + state information $\mathbf{V}, \mathbf{F}$

$$\boxed{\neg\mathrm{own}(x,y)} \xrightarrow{\mathrm{buy}(x,y)} \boxed{\mathrm{own}(x,y)}$$

$$\boxed{(\mathrm{own}(x,y),0),\ \mathrm{buy}(x,y)}\boxed{(\mathrm{own}(x,y),1)}$$

Talk Outline

§1 Transitions from finite automata

§2 Strings as compressed models

§3 Granularity: sigs & reducts

§4 Deformations: institution as triad (Goguen)

Strings in Reichenbach

Simple Past: $E \approx R \quad R < S$

$$\boxed{E,R} \;\&\; \boxed{R}\boxed{S} \;=\; \boxed{E,R}\boxed{S}$$

String sets in Reichenbach

Simple Past: $E \approx R \quad R < S$

$$\boxed{E,R} \;\&\; \boxed{R}\boxed{S} \;=\; \boxed{E,R}\boxed{S}$$

Posterior Past: $R < E \quad R < S$

$$\boxed{R}\boxed{E} \;\&\; \boxed{R}\boxed{S} \;=\; \boxed{R}\underbrace{(\boxed{E,S} + \boxed{E}\boxed{S} + \boxed{S}\boxed{E})}_{\text{trichotomy}}$$

String sets in Reichenbach and Allen

Simple Past: $E \approx R \quad R < S$

$$\boxed{E,R} \;\&\; \boxed{R}\boxed{S} \;=\; \boxed{E,R}\boxed{S}$$

Posterior Past: $R < E \quad R < S$

$$\boxed{R}\boxed{E} \;\&\; \boxed{R}\boxed{S} \;=\; \boxed{R}\underbrace{(\boxed{E,S} + \boxed{E}\boxed{S} + \boxed{S}\boxed{E})}_{\text{trichotomy}}$$

$$\boxed{l}\boxed{r} \;\&\; \boxed{l'}\boxed{r'} \;=\; \text{13 Allen relations}$$

String sets in Reichenbach and Allen

X *meets* Y	m	mi	XXXYYY
X *overlaps* Y	o	oi	XXX YYY
X *during* Y	d	di	XXX YYYYYY
X starts Y	s	si	XXX YYYYY

$$\boxed{l}\boxed{r} \ \& \ \boxed{l'}\boxed{r'} \ = \ \text{13 Allen relations}$$

$$\mathtt{m}(x,y) \ \& \ \mathtt{d}(y,z) \ \leadsto \ \{\mathtt{s},\mathtt{o},\mathtt{d}\}(x,z)$$

$$\boxed{x}\boxed{y} \ \& \ \boxed{z}\boxed{y,z}\boxed{z} \ = \ (\epsilon + \boxed{x} + \boxed{z})\boxed{x,z}\boxed{y,z}\boxed{z}$$

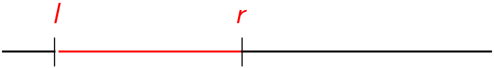
X *meets* Y: compression

Stative	delete stutters	XXXYYY	$\rightsquigarrow$ $\boxed{x}\boxed{y}$
		X *meets* Y	

X *meets* Y: compression

Stative delete stutters XXXYYY $\rightsquigarrow$ $\boxed{x}\boxed{y}$

X *meets* Y

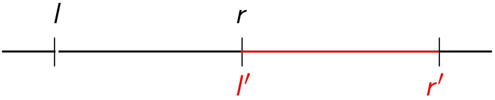


Transition $\boxed{l}\boxed{r}$

X *meets* Y: compression

Stative   delete stutters   XXXYYY $\rightsquigarrow$ $\boxed{x}\boxed{y}$

X *meets* Y



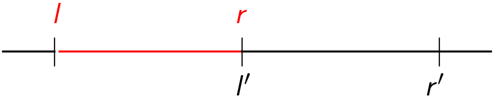
Transition $\boxed{l}\boxed{r}$

$\boxed{l}\boxed{r, l'}\boxed{r'}$

X *meets* Y: compression two ways

Stative delete stutters XXXYYY $\rightsquigarrow$ $\boxed{x}\boxed{y}$

X *meets* Y



Transition $\boxed{l}\boxed{r}$

$\boxed{l}\boxed{r, l'}\boxed{r'}$ $\rightsquigarrow$ $\boxed{l}\boxed{r}\boxed{}$

delete $\square$ (S-words, Durand & Schwer 2008)

No change: $q \xrightarrow{\square} q$

no time without change (Aristotle)

String as model: *no time without change* (Aristotle)

$$\boxed{l}\boxed{r, l'}\boxed{r'} \text{ as } \langle\{1,2,3\}, [\![S]\!], [\![P_l]\!], [\![P_r]\!], [\![P_{l'}]\!], [\![P_{r'}]\!]\rangle$$

$$[\![S]\!] := \{(1,2),(2,3)\}$$

$$[\![P_l]\!] := \{1\}, \quad [\![P_r]\!] := \{2\}, \quad [\![P_{l'}]\!] := \{2\}, \quad [\![P_{r'}]\!] := \{3\}$$

$$ntwoc_{A,V} := \forall i \; (\bigvee_{a \in A} P_a(i) \vee \bigvee_{u \in \sum V} \delta_u(i))$$

$$\delta_u(i) := P_u(i) \wedge \neg\exists j(iSj \wedge P_u(j))$$

String as model: *no time without change* (Aristotle)

$$\boxed{l}\boxed{r,l'}\boxed{r'} \quad \text{as} \quad \langle\{1,2,3\}, [\![S]\!], [\![P_l]\!], [\![P_r]\!], [\![P_{l'}]\!], [\![P_{r'}]\!]\rangle$$

$$[\![S]\!] \ := \ \{(1,2),(2,3)\}$$

$$[\![P_l]\!] := \{1\}, \quad [\![P_r]\!] := \{2\}, \quad [\![P_{l'}]\!] := \{2\}, \quad [\![P_{r'}]\!] := \{3\}$$

$$ntwoc_{A,V} \ := \forall i \ (\bigvee_{a \in A} P_a(i) \vee \bigvee_{u \in \sum V} \delta_u(i))$$

$$\delta_u(i) \ := \ P_u(i) \wedge \neg\exists j(iSj \wedge P_u(j))$$

J.A. Wheeler

John Archibald Wheeler (July 9, 1911 – April 13, 2008) was an American theoretical physicist. He was largely responsible for reviving interest in general relativity in the United States after World War II. Wheeler also worked with Niels Bohr in explaining the basic principles behind nuclear fission. Together with Gregory Breit, Wheeler developed the concept of the Breit–Wheeler process. He is best known for popularizing the term "black hole,"[1] as to objects with gravitational collapse already predicted during the early 20th century, for inventing the terms "quantum foam", "neutron moderator", "wormhole" and "it from bit", and for hypothesizing the "one-electron universe". Stephen Hawking referred to him as the "hero of the black hole story".[2]

John Archibald Wheeler

Wheeler before the Hermann Weyl-Conference 1985 in Kiel, Germany

J.A. Wheeler: *it from bit*

every it — every particle, every field of force, even the spacetime continuum itself — derives its function, its meaning, its very existence entirely — even if in some contexts indirectly — from the apparatus-elicited answers to yes-or-no questions, binary choices, bits.

- Information, physics, quantum: the search for links, 1990

John Archibald Wheeler (July 9, 1911 – April 13, 2008) was an American theoretical physicist. He was largely responsible for reviving interest in general relativity in the United States after World War II. Wheeler also worked with Niels Bohr in explaining the basic principles behind nuclear fission. Together with Gregory Breit, Wheeler developed the concept of the Breit–Wheeler process. He is best known for popularizing the term "black hole,"[1] as to objects with gravitational collapse already predicted during the early 20th century, for inventing the terms "quantum foam", "neutron moderator", "wormhole" and "it from bit", and for hypothesizing the "one-electron universe". Stephen Hawking referred to him as the "hero of the black hole story".[2]

John Archibald Wheeler

Wheeler before the Hermann Weyl-Conference 1985 in Kiel, Germany

A-compression for $ntwoc_{A,V}$

Theorem. For all $s \in {\mathcal{B}_{A,V}}^*$,

$$s \models ntwoc_{A,V} \iff s = \kappa_A(s)$$

A-compression for $ntwoc_{A,V}$

Theorem. For all $s \in \mathcal{B}_{A,V}{}^*$,

$$s \models ntwoc_{A,V} \iff s = \kappa_A(s)$$

where

$$\kappa_A(s) := \begin{cases} \epsilon & \text{if } s = \epsilon \text{ or } s = \Box \\ s & \text{else if length}(s) = 1 \end{cases}$$

$$\kappa_A(\alpha\,\alpha' s) := \begin{cases} \kappa_A(\alpha' s) & \text{if } \alpha = \Box \text{ or } \alpha = \alpha' \setminus A \\ \alpha\ \kappa_A(\alpha' s) & \text{otherwise} \end{cases}$$

A-compression for $ntwoc_{A,V}$

Theorem. For all $s \in {\mathcal{B}_{A,V}}^*$,

$$s \models ntwoc_{A,V} \iff s = \kappa_A(s)$$

where

$$\kappa_A(s) := \begin{cases} \epsilon & \text{if } s = \epsilon \text{ or } s = \Box \\ s & \text{else if length}(s) = 1 \end{cases}$$

$$\kappa_A(\alpha\,\alpha' s) := \begin{cases} \kappa_A(\alpha' s) & \text{if } \alpha = \Box \text{ or } \alpha = \alpha' \setminus A \\ \alpha\ \kappa_A(\alpha' s) & \text{otherwise} \end{cases}$$

κ_A is computable by a finite-state transducer and for $s \in {\mathcal{B}_{A,V}}^*$,

$$\kappa_A(s) = \begin{cases} d^{\Box}(s & \text{if } V = \emptyset \\ bc(s) & \text{else if } A = \emptyset \end{cases}$$

Talk Outline

§1 Transitions from finite automata

§2 Strings as compressed models

§3 Granularity: sigs & reducts

§4 Deformations: institution as triad (Goguen)

Finite precision: (Act, Val)-sigs

Given: a function Val from variables x to sets $\mathrm{Val}(x)$, and
a set Act of acts.

Finite precision: (Act, Val)-sigs

Given: a function Val from variables x to sets $\mathrm{Val}(x)$, and a set Act of acts.

An (Act, Val)-*sig* is a pair (A, V) of a finite subset A of Act and a fin-blurring V of Val

Finite precision: (Act, Val)-sigs

Given: a function Val from variables x to sets Val(x), and
a set Act of acts.

An (Act, Val)-*sig* is a pair (A, V) of a finite subset A of Act
and a fin-blurring V of Val
a function V whose domain is a finite subset of *dom*(Val) s.t.

$$(\forall x \in dom(\mathrm{Val}))\ V(x) \text{ is a finite partition of } \mathrm{Val}(x).$$

Finite precision: (Act, Val)-sigs

Given: a function Val from variables x to sets $\mathrm{Val}(x)$, and
a set Act of acts.

An $(\mathrm{Act}, \mathrm{Val})$-*sig* is a pair (A, V) of a finite subset A of Act
and a fin-blurring V of Val
a function V whose domain is a finite subset of $dom(\mathrm{Val})$ s.t.

$$(\forall x \in dom(\mathrm{Val}))\ V(x) \text{ is a finite partition of } \mathrm{Val}(x).$$

$$(A, V) \preceq (A', V') \iff A \subseteq A' \text{ and } V \leq V'$$

Finite precision: (Act, Val)-sigs

Given: a function Val from variables x to sets $\mathsf{Val}(x)$, and
a set Act of acts.

An (Act, Val)-*sig* is a pair (A, V) of a finite subset A of Act
and a fin-blurring V of Val
a function V whose domain is a finite subset of $dom(\mathsf{Val})$ s.t.

$$(\forall x \in dom(\mathsf{Val}))\ V(x) \text{ is a finite partition of } \mathsf{Val}(x).$$

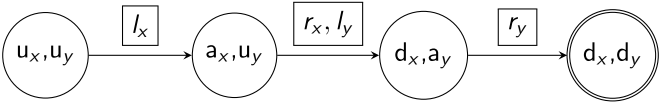
$$(A, V) \preceq (A', V') \iff A \subseteq A' \text{ and } V \leq V'$$

where $\leq$ allows values (cells) to be refined

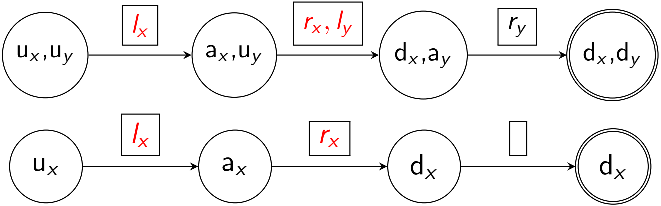
$$V \leq V' \iff (\forall x \in dom(V))\ x \in dom(V') \text{ and } \overbrace{V'(x) \text{ refines } V(x)}^{}$$

$$\text{refines: } (\forall c' \in V'(x))(\exists c \in V(x))\ c' \subseteq c$$

X meets Y, revisited



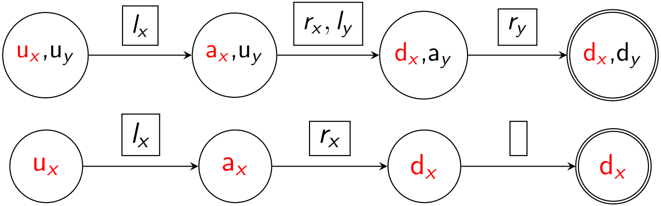
X meets Y, revisited



B-reduct $\quad \rho_B(\alpha_1 \cdots \alpha_n) := (\alpha_1 \cap B) \cdots (\alpha_n \cap B)$

$$\rho_{\{l_x, r_x\}}(\boxed{l_x}\boxed{r_x, l_y}\boxed{r_y}) = \boxed{l_x}\boxed{r_x}\boxed{}$$

X meets Y, revisited



B-reduct $\quad \rho_B(\alpha_1 \cdots \alpha_n) \; := \; (\alpha_1 \cap B) \cdots (\alpha_n \cap B)$

$$\rho_{\{l_x, r_x\}}(\boxed{l_x}\boxed{r_x, l_y}\boxed{r_y}) \;=\; \boxed{l_x}\boxed{r_x}\boxed{}$$

$$\rho_{\{\mathsf{u}_x, \mathsf{a}_x, \mathsf{d}_x\}}(\boxed{\mathsf{u}_x,\mathsf{u}_y}\boxed{\mathsf{a}_x,\mathsf{u}_y}\boxed{\mathsf{d}_x,\mathsf{a}_y}\boxed{\mathsf{d}_x,\mathsf{d}_y}) \;=\; \boxed{\mathsf{u}_x}\boxed{\mathsf{a}_x}\boxed{\mathsf{d}_x}\boxed{\mathsf{d}_x}$$

Talk Outline

§1 Transitions from finite automata

§2 Strings as compressed models

§3 Granularity: sigs & reducts

§4 Deformations: institution as triad (Goguen)

Institution (Goguen & Burstall)

finite automaton	deformation	institution
alphabet (A, V)	blur	$\Sigma \in \mathbf{Sig}$

Joseph Amadee Goguen (/ˈɡoʊɡən/ *GOH-gən*; June 28, 1941 – July 3, 2006) was an American computer scientist. He was professor of Computer Science at the University of California and University of Oxford, and held research positions at IBM and SRI International.

In the 1960s, along with Lotfi Zadeh, Goguen was one of the earliest researchers in fuzzy logic and made profound contributions to fuzzy set theory.[1][2] In the 1970s Goguen's work was one of the earliest approaches to the algebraic characterisation of abstract data types and he originated and helped develop the OBJ family of programming languages.[3][4] He was author of *A Categorical Manifesto* and founder[5] and Editor-in-Chief of the *Journal of Consciousness Studies*. His development of institution theory impacted the field of universal logic.[6][7]

Joseph A. Goguen

Joseph Goguen in 2004

Institution (Goguen & Burstall)

finite automaton	deformation	institution
alphabet (A, V)	blur	$\Sigma \in \mathbf{Sig}$
string	domain warping	Σ-model

Given $\Sigma \xrightarrow{\sigma} \Sigma'$, $s' \in \mathbf{Mod}(\Sigma')$, ,

$$\mathbf{Mod}(\sigma) : \mathbf{Mod}(\Sigma') \to \mathbf{Mod}(\Sigma)$$

$$\mathbf{Mod}(\sigma)(s') = \kappa_\sigma(s')$$ reduct ; compression

Institution (Goguen & Burstall)

finite automaton	**deformation**	**institution**
alphabet (A, V)	blur	$\Sigma \in \mathbf{Sig}$
string	domain warping	Σ-model
regular expression	superposition	Σ-sentence

Given $\Sigma \stackrel{\sigma}{\to} \Sigma'$, $s' \in \mathbf{Mod}(\Sigma')$, $\varphi \in Sen(\Sigma)$,

$$\mathbf{Mod}(\sigma) : \mathbf{Mod}(\Sigma') \to \mathbf{Mod}(\Sigma)$$

$$\mathbf{Mod}(\sigma)(s') = \kappa_\sigma(s') \quad \text{reduct ; compression}$$

contra $Sen(\sigma) : Sen(\Sigma) \to Sen(\Sigma')$

$$Sen(\sigma)(\varphi) = \langle\sigma\rangle\varphi$$

$$s' \models_{\Sigma'} \langle\sigma\rangle\varphi \iff \kappa_\sigma(s') \models_\Sigma \varphi \qquad \text{satisfaction condition}$$

Institution (Goguen & Burstall)

finite automaton	**deformation**	**institution**
alphabet (A, V)	blur	$\Sigma \in \mathbf{Sig}$
string	domain warping	Σ-model
regular expression	superposition	Σ-sentence

Given $\Sigma \overset{\sigma}{\to} \Sigma'$, $s' \in \mathbf{Mod}(\Sigma')$, $\varphi \in Sen(\Sigma)$,

$$\mathbf{Mod}(\sigma) : \mathbf{Mod}(\Sigma') \to \mathbf{Mod}(\Sigma)$$

$$\mathbf{Mod}(\sigma)(s') = \kappa_\sigma(s') \quad \text{reduct ; compression}$$

contra $Sen(\sigma) : Sen(\Sigma) \to Sen(\Sigma')$

$$Sen(\sigma)(\varphi) = \langle\sigma\rangle\varphi$$

$$s' \models_{\Sigma'} \langle\sigma\rangle\varphi \iff \kappa_\sigma(s') \models_\Sigma \varphi \qquad \text{satisfaction condition}$$

superpose(φ_1, φ_2) as $\langle\sigma_1\rangle\varphi_1 \wedge \langle\sigma_2\rangle\varphi_2$

Institution (Goguen & Burstall)

finite automaton	**deformation**	**institution**
alphabet (A, V)	blur	$\Sigma \in \mathbf{Sig}$
string	domain warping	Σ-model
regular expression	superposition	Σ-sentence
string set	interruption	$[\![\varphi]\!]_\Sigma$

Given $\Sigma \xrightarrow{\sigma} \Sigma'$, $s' \in \mathbf{Mod}(\Sigma')$, $\varphi \in Sen(\Sigma)$,

$$\mathbf{Mod}(\sigma) : \mathbf{Mod}(\Sigma') \to \mathbf{Mod}(\Sigma)$$

$$\mathbf{Mod}(\sigma)(s') \;=\; \kappa_\sigma(s') \qquad \text{reduct ; compression}$$

contra $Sen(\sigma) : \; Sen(\Sigma) \to Sen(\Sigma')$

$$Sen(\sigma)(\varphi) = \langle\sigma\rangle\varphi$$

$$s' \models_{\Sigma'} \langle\sigma\rangle\varphi \iff \kappa_\sigma(s') \models_\Sigma \varphi \qquad \text{satisfaction condition}$$

superpose(φ_1, φ_2) as $\langle\sigma_1\rangle\varphi_1 \wedge \langle\sigma_2\rangle\varphi_2$

$$[\![\varphi]\!]_\Sigma \;:=\; \{s \in \mathbf{Mod}(\Sigma) \mid s \models_\Sigma \varphi\}$$

Inertia & interruption

For $a \in \text{Act}$, let $\text{af}(a)$ be the set of variables that a can affect.

An (A, V)-string s is (A, V, af)-*inertial* if for every V-pair u,
any u-change in s occurs with an act in A that can affect u

$$\forall i \forall j \quad (iSj \wedge P_u(i) \wedge \neg P_u(j)) \supset \bigvee_{a \in A_u} P_a(i) \qquad (\dagger)$$

where $A_{(x,c)} = \{a \in A \mid x \in \text{af}(a)\}$.

Inertia & interruption

For $a \in \mathrm{Act}$, let $\mathrm{af}(a)$ be the set of variables that a can affect.

An (A, V)-string s is (A, V, af)-*inertial* if for every V-pair u,
any u-change in s occurs with an act in A that can affect u

$$\forall i \forall j \quad (iSj \wedge P_u(i) \wedge \neg P_u(j)) \supset \bigvee_{a \in A_u} P_a(i) \qquad (\dagger)$$

where $A_{(x,c)} = \{a \in A \mid x \in \mathrm{af}(a)\}$.

Inertia & interruption

For $a \in \text{Act}$, let $\text{af}(a)$ be the set of variables that a can affect.

An (A, V)-string s is (A, V, af)-*inertial* if for every V-pair u, any u-change in s occurs with an act in A that can affect u

$$\forall i \forall j \quad (iSj \wedge P_u(i) \wedge \neg P_u(j)) \supset \bigvee_{a \in A_u} P_a(i) \qquad (\dagger)$$

where $A_{(x,c)} = \{a \in A \mid x \in \text{af}(a)\}$.

Otherwise, s is (A, V, af)-*interrupted*.

The (A, V)-projection of an (A', V', af)-inertial string can be (A, V, af)-interrupted because $(\dagger)$ needs an $a \in A' \setminus A$.

Expand V to V' for $(\dagger)$-converse on *event nuclei* (Moens & Steedman 1988)

`https://web.stanford.edu/~laurik/fsmbook/examples/YaleShooting.html` F & Nairn 2005, IWCS-6 Tilburg 2005

So what?

What is a string assigned a probability by a language model about?

So what?

What is a string assigned a probability by a language model about?

(1) `Facebook bought Instagram.`

So what?

What is a string assigned a probability by a language model about?

(1) `Facebook bought Instagram.`

(2) `Facebook owns Instagram.`

So what?

What is a string assigned a probability by a language model about?

(1) `Facebook bought Instagram.`

(2) `Facebook owns Instagram.`

(7) `Facebook spreads lies.`

So what?

What is a string assigned a probability by a language model about?

(1) `Facebook bought Instagram.`

(2) `Facebook owns Instagram.`

(7) `Facebook spreads lies.`

It is about a process (learning) that can be approximated by semantic representations at bounded granularities.

$$\text{pattern} \approx \underbrace{\text{pure situation}}_{\text{input}} + \underbrace{\text{deformations}}_{\text{noise}} \quad \text{(Shannon channel)}$$

So what?

What is a string assigned a probability by a language model about?

(1) `Facebook bought Instagram.`

(2) `Facebook owns Instagram.`

(7) `Facebook spreads lies.`

It is about a process (learning) that can be approximated by semantic representations at bounded granularities.

$$\text{pattern} \approx \underbrace{\text{pure situation}}_{\text{input}} + \underbrace{\text{deformations}}_{\text{noise}} \quad \text{(Shannon channel)}$$

$$\text{string} \quad \left\{ \begin{array}{ll} \text{blur} & \Delta(\text{alphabet}) \\ \text{domain warp} & \text{compression} \\ \text{superposition} & \text{language} \\ \text{interruption} & \text{constraints} \end{array} \right.$$

Proposal: turn knowledge graphs into finite automata,
to support refinements and alternatives

Proposal: turn knowledge graphs into finite automata,
to support refinements and alternatives

Thank You

☺