Finite-state transducers and regular relations

A finite-state transducer (FST) is a finite automaton with the labels on its transitions doubled and allowed to be $\epsilon$, for a transition table $\delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times (\Sigma \cup \{\epsilon\}) \times Q$ with

$$\delta(q, x, x', q')$$ written $q \xrightarrow{x:x'} q'$.

The fst $\langle \rightarrow, F \rangle$ computes the relation

$$\{(x_1 \ldots x_n, x'_1 \ldots x'_n) \in \Sigma^* \times \Sigma^* | (\exists q_1 \ldots q_n) q_0 \xrightarrow{x_1:x'_1} q_1 \xrightarrow{x_2:x'_2} q_2 \cdots \xrightarrow{x_n:x'_n} q_n \in F\}.$$
Finite-state transducers and regular relations

A finite-state transducer (FST) is a finite automaton with the labels on its transitions doubled and allowed to be $\epsilon$, for a transition table $\delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times (\Sigma \cup \{\epsilon\}) \times Q$ with

$$\delta(q, x, x', q') \text{ written } q \xrightarrow{x:x'} q'$$

The fst $\langle \rightarrow, F \rangle$ computes the relation

$$\{(x_1 \ldots x_n, x'_1 \ldots x'_n) \in \Sigma^* \times \Sigma^* \mid (\exists q_1 \ldots q_n) \ q_0 \xrightarrow{x_1:x'_1} q_1 \xrightarrow{x_2:x'_2} \ldots \xrightarrow{x_n:x'_n} q_n \in F \}.$$  

N.B. $x_1 \ldots x_n$ and $x'_1 \ldots x'_n$ may have different lengths, as an $x_i$ and/or $x'_i$ can be $\epsilon$ (of length 0).
Finite-state transducers and regular relations

A finite-state transducer (FST) is a finite automaton with the labels on its transitions doubled and allowed to be $\epsilon$, for a transition table $\delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times (\Sigma \cup \{\epsilon\}) \times Q$ with

$$\delta(q, x, x', q')$$

written $q \xrightarrow{x:x'} q'$. The fst $\langle \rightarrow, F \rangle$ computes the relation

$$\{(x_1 \ldots x_n, x'_1 \ldots x'_n) \in \Sigma^* \times \Sigma^* \mid (\exists q_1 \ldots q_n) q_0 \xrightarrow{x_1:x'_1} q_1 \xrightarrow{x_2:x'_2} q_2 \cdots \xrightarrow{x_n:x'_n} q_n \in F\}.$$  

N.B. $x_1 \ldots x_n$ and $x'_1 \ldots x'_n$ may have different lengths, as an $x_i$ and/or $x'_i$ can be $\epsilon$ (of length 0).  

A relation between strings is regular if it is computed by some fst.
Prolog exercise

Define

\[
\text{fst}(+\text{Input}, +\text{Trans}, +\text{Final}, ?\text{Output})
\]

\textbf{Caution:} consider \text{fst} computing \(1 \times 1^+\) given by

\[\text{Trans} = [[q0, [], 1, q0], [q0, 1, 1, q1]]\]
\[\text{Final} = [q1]\]

That is, the length \(n\) of a pair \((x_1 \ldots x_n, x'_1 \ldots x'_n)\) no longer bounds a run computing it.
Some regular relations

1. The \textit{factor} relation

\[ s \text{ hasFactor } s' \iff (\exists u, v) \ s = us'v \]
Some regular relations

1. The *factor* relation

   \[ s \text{ hasFactor } s' \iff (\exists u, v) \ s = us'v \]

2. The *accepting runs of* a finite automaton \( \to, F \)

   \[
   \{ \langle a_1 a_2 \cdots a_n, q_1 q_2 \cdots q_n \rangle \mid q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \cdots \xrightarrow{a_n} q_n \in F \}
   \]

   mixing symbols/actions \( a_i \) with states/situations \( q_i \)
Some regular relations

1. The factor relation

\[ s \text{ hasFactor } s' \iff (\exists u, v) \ s = us'v \]

2. The accepting runs of a finite automaton \[ \rightarrow, F \]

\[ \{ \langle a_1a_2\cdots a_n, q_1q_2\cdots q_n \rangle \mid q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \cdots \xrightarrow{a_n} q_n \in F \} \]

mixing symbols/actions \( a_i \) with states/situations \( q_i \)

3. The diagonal \( \Delta_L \) of a regular language \( L \)

\[ \Delta_L := \{(s, s) \mid s \in L\} \]
Some regular relations

1. The factor relation

\[ s \text{ hasFactor } s' \iff (\exists u, v) \ s = us'v \]

2. The accepting runs of a finite automaton \( \rightarrow, F \)

\[ \{ \langle a_1a_2\cdots a_n, q_1q_2\cdots q_n \rangle \mid q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \cdots \xrightarrow{a_n} q_n \in F \} \]

mixing symbols/actions \( a_i \) with states/situations \( q_i \)

3. The diagonal \( \Delta_L \) of a regular language \( L \)

\[ \Delta_L := \{ (s, s) \mid s \in L \} \]

4. A-string-meronym \( \geq_A \) on \( (2^A)^* \)

\[ \alpha_1 \cdots \alpha_n \geq_A x_1 \cdots x_n \iff x_i = \epsilon \ \text{or} \ x_i \subseteq \alpha_i \ \text{for} \ 1 \leq i \leq n \]

for \( \alpha_1 \cdots \alpha_n \in (2^A)^* \).
Some closure properties

1. If $R$ is regular, so is its inverse $R^{-1}$. 

2. If $R$ and $R'$ are regular, so are their union $R \cup R'$ and relational composition $R ; R' := \{(s, s') \mid \exists s_0 \in R s R s_0 \text{ and } s_0 R' s' \}$.

3. The restriction $R_L$ of a regular relation $R$ to a regular language $L$ 

$$R_L := \{(s, s') \in R \mid s \in L\}$$

If $R$ is a regular relation, then its image 

$$\{s' \mid \exists s \in R s R s' \}$$

is regular including $L \cap L' = \text{image} (\Delta L; \Delta L')$ and the Peirce product 

$$R^{-1} L := \{s \mid \exists s' \in L s R s' \} = \text{image} (R^{-1} L)$$.
Some closure properties

1. If $R$ is regular, so is its inverse $R^{-1}$.
2. If $R$ and $R'$ are regular, so are its union $R \cup R'$ and *relational composition*

\[
R; R' := \{(s, s') \mid (\exists s_0) \ sRs_0 \text{ and } s_0R's'\}
\]
Some closure properties

1. If $R$ is regular, so is its inverse $R^{-1}$.
2. If $R$ and $R'$ are regular, so are its union $R \cup R'$ and relational composition
   \[ R; R' := \{(s, s') \mid (\exists s_0) sRs_0 \text{ and } s_0R's'\} \]

3. The restriction $R_L$ of a regular relation $R$ to a regular language $L$
   \[ R_L := \{(s, s') \in R \mid s \in L\} \]
Some closure properties

1. If $R$ is regular, so is its inverse $R^{-1}$.

2. If $R$ and $R'$ are regular, so are its union $R \cup R'$ and relational composition

   $$R; R' := \{(s, s') \mid (\exists s_0) \ sRs_0 \text{ and } s_0R's'\}.$$ 

3. The restriction $R_L$ of a regular relation $R$ to a regular language $L$

   $$R_L := \{(s, s') \in R \mid s \in L\}$$

If $R$ is a regular relation, then its image $\{s' \mid (\exists s) \ sRs'\}$ is regular including

$$L \cap L' = \text{image}(\Delta_L; \Delta_{L'})$$

and the Peirce product

$$R^{-1}L := \{s \mid (\exists s' \in L) \ sRs'\} = \text{image}(R^{-1}L)$$
Regular relations are not Boolean-closed

Regular relations are *not* closed under intersection —

\[ \{\langle 0^n, 1^n2^m \rangle \mid n \geq 0, m \geq 0\} \quad \text{and} \quad \{\langle 0^n, 1^m2^n \rangle \mid n \geq 0, m \geq 0\} \]

are regular, but their intersection has image \( \sum_{n \geq 0} 1^n2^n \).
Regular relations are not Boolean-closed

Regular relations are not closed under intersection —

\[ \{ \langle 0^n, 1^n2^m \rangle \mid n \geq 0, m \geq 0 \} \text{ and } \{ \langle 0^n, 1^m2^n \rangle \mid n \geq 0, m \geq 0 \} \]

are regular, but their intersection has image \( \sum_{n \geq 0} 1^n2^n \).

Hence, the complement \( \overline{R} \) of a regular relation \( R \) need not be regular, as

\[ R \cap R' = \overline{R} \cup \overline{R'} \]
TM-actions via finite-state transducers

\[
\text{moveRight}(q, a, q') \quad \vec{t}_a q a \vec{r} \rightsquigarrow \vec{t}_a q' a \vec{r}
\]

\[
\text{moveLeft}(q, a, q') \quad \vec{t}_a' q a \vec{r} \rightsquigarrow \vec{t}_q a' a \vec{r}
\]

\[
\text{write}(q, a, a', q') \quad \vec{t}_q a \vec{r} \rightsquigarrow \vec{t}_q a' a \vec{r}
\]
TM-actions via finite-state transducers

moveRight\((q, a, q')\) \hspace{1cm} \vec{t}qar \leadsto \vec{t}aq'r

moveLeft\((q, a, q')\) \hspace{1cm} \vec{t}a'qar \leadsto \vec{t}q'a'ar

write\((q, a, a', q')\) \hspace{1cm} \vec{t}qar \leadsto \vec{t}q'a'r

tape infinite to the right \hspace{1cm} \vec{t}q \approx \vec{t}q\#

tape infinite to the left \hspace{1cm} q\vec{r} \approx \#q\vec{r}
TM-actions via finite-state transducers

\[
\begin{align*}
\text{moveRight}(q, a, q') & : \quad \vec{t}qar \xrightarrow{\sim} \vec{t}aq'r \\
\text{moveRight}(q, \#, q') & : \quad \vec{t}q \xrightarrow{\sim} \vec{t}\#q' \\
\text{moveLeft}(q, a, q') & : \quad \vec{t}a'qar \xrightarrow{\sim} \vec{t}q'a'ar \\
\text{moveLeft}(q, a, q') & : \quad qar \xrightarrow{\sim} q'\#a'r \\
\text{moveLeft}(q, \#, q') & : \quad \vec{t}aq \xrightarrow{\sim} \vec{t}q'a \\
\text{moveLeft}(q, \#, q') & : \quad q \xrightarrow{\sim} q' \\
\text{write}(q, a, a', q') & : \quad \vec{t}qar \xrightarrow{\sim} \vec{t}q'a'\vec{r} \\
\text{write}(q, \#, a, q') & : \quad \vec{t}q \xrightarrow{\sim} \vec{t}q'a \\
\text{tape infinite to the right} & : \quad \vec{t}q \approx \vec{t}q\# \\
\text{tape infinite to the left} & : \quad q\vec{r} \approx \#q\vec{r}
\end{align*}
\]
TM-actions via finite-state transducers

\[
\begin{align*}
\text{moveRight}(q, a, q') & \quad \vec{l}qar \rightsquigarrow \vec{l}aq'r \\
\text{moveRight}(q, \#, q') & \quad \vec{l}q \rightsquigarrow \vec{l}\#q' \\
\text{moveLeft}(q, a, q') & \quad \vec{l}a'qar \rightsquigarrow \vec{l}q'a'ar \\
\text{moveLeft}(q, a, q') & \quad qar \rightsquigarrow q'\#a'r \\
\text{moveLeft}(q, \#, q') & \quad \vec{l}aq \rightsquigarrow \vec{l}q'a \\
\text{moveLeft}(q, \#, q') & \quad q \rightsquigarrow q' \\
\text{write}(q, a, a', q') & \quad \vec{l}qar \rightsquigarrow \vec{l}q'a'r \\
\text{write}(q, \#, a, q') & \quad \vec{l}q \rightsquigarrow \vec{l}q'a \\
\end{align*}
\]

tape infinite to the right \quad \vec{l}q \approx \vec{l}q\# \\
tape infinite to the left \quad q\vec{r} \approx \#q\vec{r}

For any Turing machine \( M \), \( \text{step}_M \) is regular (finite tuples)
Finite-state approximations

Output extraction $\rightsquigarrow$ via finite-state transducer

\[
\begin{align*}
\text{halt}(q, a) & \quad \vec{I}q\vec{a} \rightsquigarrow \text{unpad}(\vec{I}a\vec{r}) \\
\text{halt}(q, \#) & \quad \vec{I}q \rightsquigarrow \text{unpad}(\vec{I})
\end{align*}
\]

$n$-step approximation of a TM $M$

\[
M_n := \{ (s, s') \mid q_0s \xrightarrow{\text{step}_M^n} s' \}
\]

$\exists x \hspace{0.2cm} q_0s \xrightarrow{\text{step}_M^n} x$ \text{ and } $x \rightsquigarrow s'$

Bounded iterations (time-out clock) are regular

\[
\text{input/output}(M) = \bigcup_{n \geq 0} M_n
\]