Regular Splicing Languages

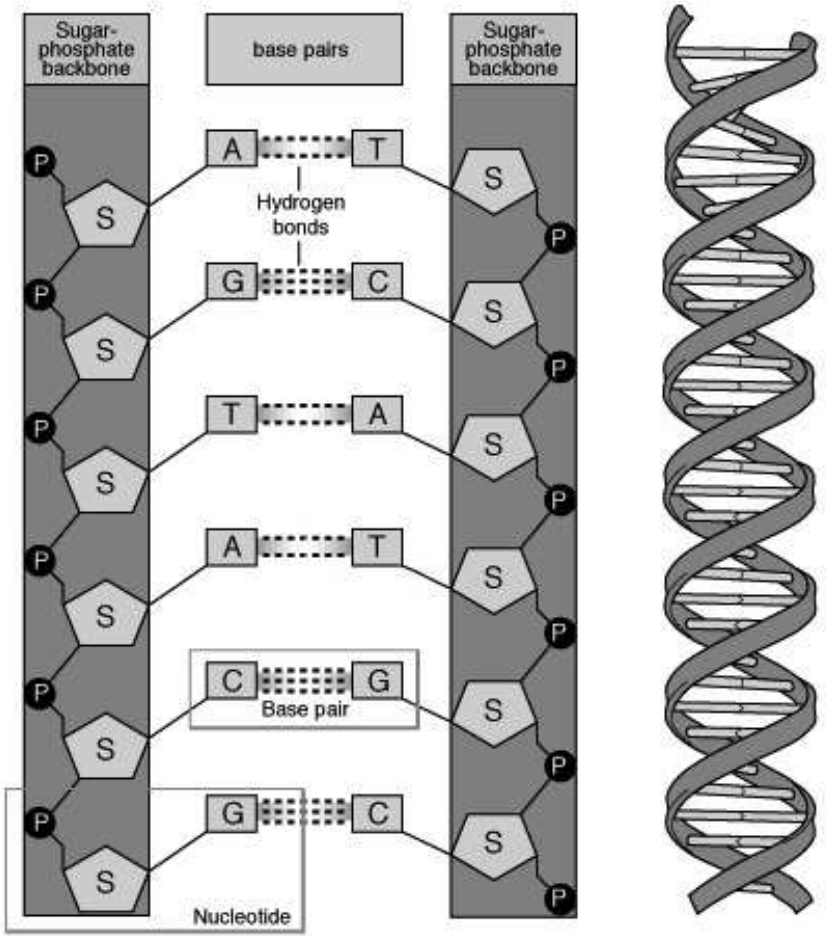
Nataša Jonoska

Department of Mathematics
University of South Florida
jonoska@math.usf.edu

Outline

- Biological motivation
- Mathematical setup
- Examples and prior results
- Regular splicing languages
  - regular splicing languages must have a constant
  - it is decidable whether a given regular language is splicing
- Open (characterization, circular splicing languages)

Structure of DNA



Can represent DNA as words over alphabet of pairs $\{A/T, T/A, C/G, G/C\}$.

. . . AGGCTTACGCGATATTGCC . . .
. . . TCCGAATGCGCTATAACGG . . .

Can represent DNA as words over alphabet of pairs $\{A/T, T/A, C/G, G/C\}$.

. . . AGGCTTACGCGATATTGCC . . .
. . . TCCGAATGCGCTATAACGG . . .

. . . NNNNAGGCTTACGCGATATTGCCNNNN . . .
. . . NNNNTCCGAATGCGCTATAACGGNNNN . . .

Can represent DNA as words over alphabet of pairs $\{A/T, T/A, C/G, G/C\}$.

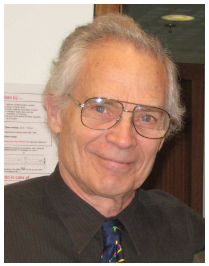
. . . AGGCTTACGCGATATTGCC . . .
. . . TCCGAATGCGCTATAACGG . . .

. . . NNNNAGGCTTACGCGATATTGCCNNNN . . .
. . . NNNNTCCGAATGCGCTATAACGGNNNN . . .

Molecules are read from 5' to 3'

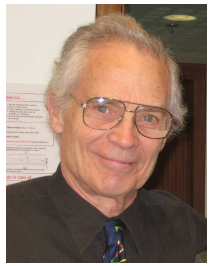
. . . NNNNAGGCTTACGCGATATTGCCNNNN . . .

Splicing


Tom Head, Formal Language Theory and DNA *Bul. Math Biology* **49** (1987).

Actions of Restriction Endonucleases (type II)

Splicing



Tom Head, Formal Language Theory and DNA *Bul. Math Biology* **49** (1987).

Actions of Restriction Endonucleases (type II)

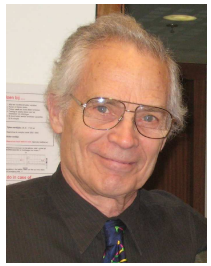
Taq I

.... NN TC GA NN...
.... NN AG CT NN...

Sci NI

.... NN CG CG NN...
.... NN GC GC NN...

Splicing



Tom Head, Formal Language Theory and DNA *Bul. Math Biology* **49** (1987).

Actions of Restriction Endonucleases (type II)

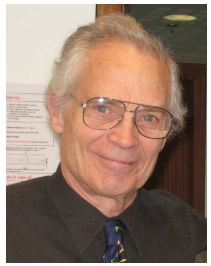
Taq I

CGANN...
.... NNT TNN...
.... NNAGC

Sci NI

CGCNN...
.... NNG GNN...
.... NNCGC

Splicing



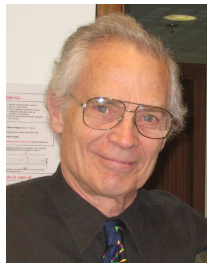
Tom Head, Formal Language Theory and DNA *Bul. Math Biology* **49** (1987).

Actions of Restriction Endonucleases (type II)

.... NN TCGC NN...
.... NN AGCG NN...

.... NN GCGA NN...
.... NN CGCT NN...

Splicing

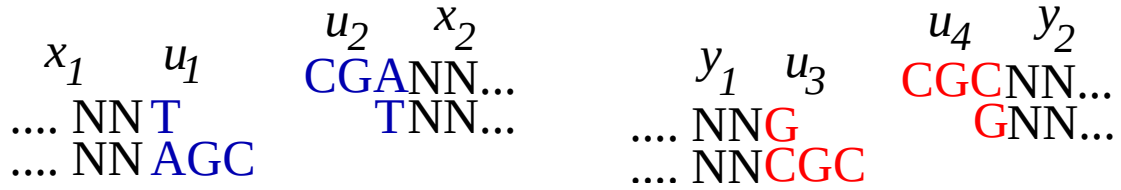

Tom Head, Formal Language Theory and DNA *Bul. Math Biology* **49** (1987).

Actions of Restriction Endonucleases (type II)

Question: Given a set of DNA molecules, restriction endonucleases and a ligase, what are the sequences of the resulting molecules?

Formalism

(due to G. Păun)



Enzymes specify restriction rules: enzyme e_1 recognizes a site u_1u_2 and enzyme e_2 recognizes a site u_3u_4 with the same overhang. A splicing rule is a four-tuple

$$r = (u_1, u_2)(u_3, u_4)$$

Formalism

(due to G. Păun)

x_1 u_1 u_2 x_2 y_1 u_3 u_4 y_2

.... NN T CGA NN... .... NN G CGC NN...

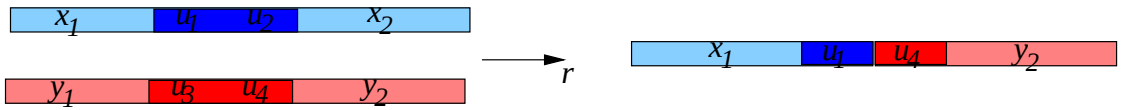
.... NN AGC T NN... .... NN CGC G NN...

Enzymes specify restriction rules: enzyme e_1 recognizes a site u_1u_2 and enzyme e_2 recognizes a site u_3u_4 with the same overhang. A splicing rule is a four-tuple

$$r = (u_1, u_2)(u_3, u_4)$$

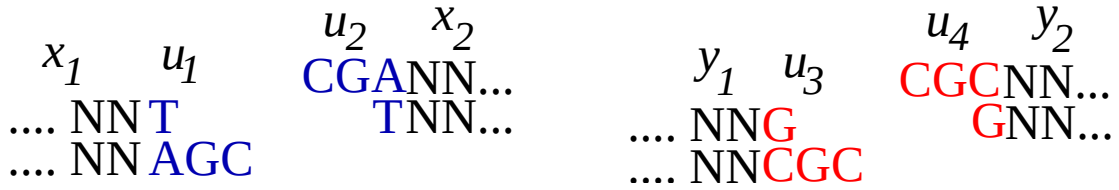
Splicing operation:

$$x_1u_1u_2x_2 \quad y_1u_3u_4y_2 \quad \rightarrow_r \quad x_1u_1u_4y_2$$



Formalism

(due to G. Păun)



Enzymes specify restriction rules: enzyme e_1 recognizes a site u_1u_2 and enzyme e_2 recognizes a site u_3u_4 with the same overhang. A splicing rule is a four-tuple

$$r = (u_1, u_2)(u_3, u_4)$$

Splicing operation:

$$(v_1, v_2) \rightarrow_r w$$

$$\begin{aligned} v_1 &= x_1u_1u_2x_2 \\ v_2 &= y_1u_3u_4y_2 \end{aligned} \qquad \rightarrow_r \qquad w = x_1u_1u_4y_2$$

Language generation

For a language L and a set of rules R we define

$$\begin{aligned} \sigma_R(L) = \sigma(L) = & \quad \{w \mid (v_1, v_2) \rightarrow_r w \text{ for some } v_1, v_2 \in L, r \in R\} \\ \sigma^{i+1}(L) = & \quad \sigma^i(L) \cup \sigma(\sigma^i(L)) \end{aligned}$$

A splicing system is a triple $H = (A, I, R)$ where A is an alphabet, $I \subseteq A^*$ is a set of initial words, R is a set of splicing rules.

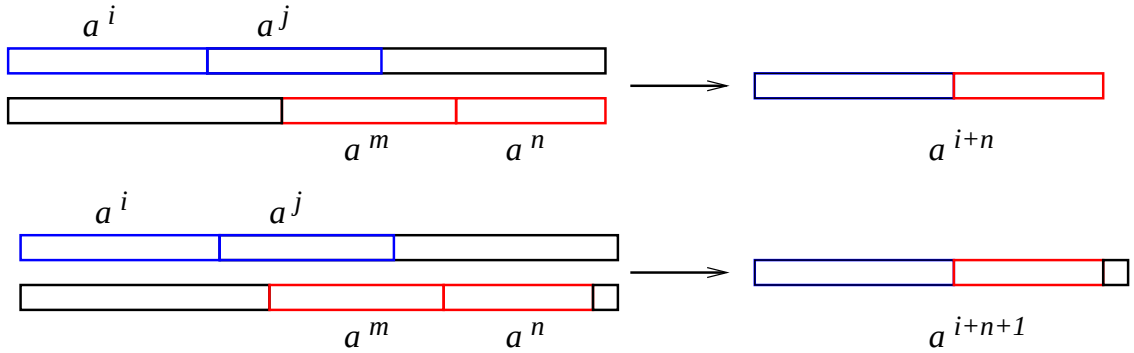
Then

$$L(H) = \cup_0^\infty \sigma^i(I)$$

L is a splicing language if $L = L(H)$ for some splicing system H.

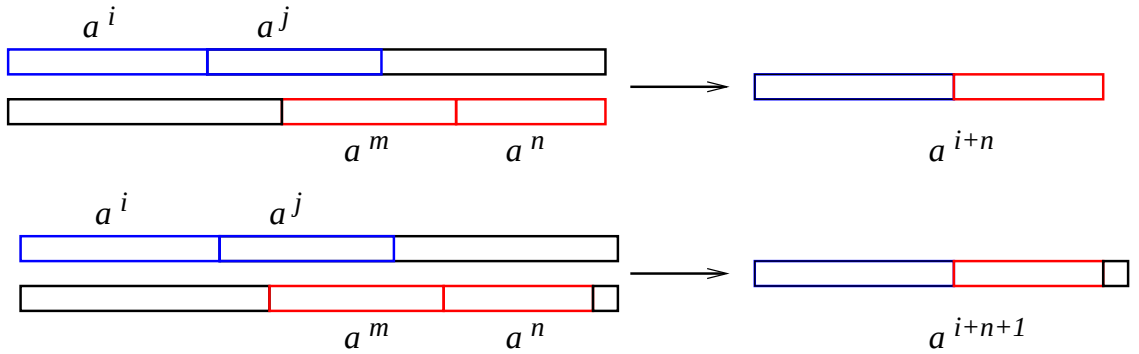
simple examples

- Language $(aa)^*$ is not splicing.
  Consider a rule $r = (a^i, a^j)(a^m, a^n)$ and a word $v = a^s$ for large s. Then $(v, v) \to_r w$ and $(v, v) \to_r wa$.



simple examples

- Language $(aa)^*$ is not splicing.
  Consider a rule $r = (a^i, a^j)(a^m, a^n)$ and a word $v = a^s$ for large s. Then $(v, v) \to_r w$ and $(v, v) \to_r wa$.



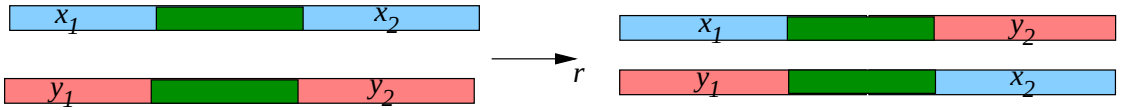
- Language $b(aa)^*$ is splicing.
  Consider $H = (A, I, R)$ with $I = \{b, baa\}$ and a splicing rule $r = (baa, 1)(b, aa)$.

$$baa| + b|aa \to baaaa, \quad baa| + b|aaaa \to baaaaa \ldots$$

Constants of a Language

A word $w \in A^+$ is a *constant of a language* L if w is a factor of some word in L and for all words x_1, x_2, y_1, y_2 in A^* we have:

$$\begin{array}{ll} x_1 w x_2 & \in L \\ y_1 w y_2 & \in L \end{array} \Rightarrow \begin{array}{ll} x_1 w y_2 & \in L \\ y_1 w x_2 & \in L \end{array}$$



Fact: (A. De Luca, A. Restivo'80s) If there is a $p > 0$ such that for all $x \in A^p$ x is a constant for L or $x \notin L$, then L is strictly locally testable.

That is: If L is extendable, factor closed and there is a $p > 0$ such that for all $x \in A^p$ x is a constant for L or $x \notin L$, then L is the factor language of a SFT.

History (brief and not complete)

For finite splicing systems (alphabet, initial strings, rules are all finite):

- T. Head (1987): if splicing is done without context (i.e., $u_1u_2 = u_3u_4$ for every rule $r \in R$) then the class of splicing languages coincides with strictly locally testable languages.
- K. Culik, T. Harju (1991): every splicing language is regular.
- D. Pixton (1996): another elegant proof of “splicing languages are regular" (construction of the finite state automaton accepting the splicing language.)

History (brief / not complete) cont.

- P. Bonizzoni (with C. De Felice, G. Mauri, R. Zizza) (2003–2010): characterized languages that can be generated by reflexive splicing rules: (for every rule $r = (u_1, u_2)(u_3, u_4)$ in R, both $(u_1, u_2)(u_1, u_2)$ and $(u_3, u_4)(u_3, u_4)$ are rules in R).
  and by symmetric splicing rules: (for every rule $r = (u_1, u_2)(u_3, u_4)$ in R, the rule $(u_3, u_4)(u_1, u_2)$ is also in R).

  this characterization relies on constants in the splicing language

- E. Goode, D. Pixton (2007): characterized reflexive splicing languages through syntactic monoids and gave an example of a splicing language that is not reflexive.

Open problems (since 1987)

Problem 1: Is it decidable whether a regular language is splicing? If yes, how hard it is?

Open problems (since 1987)

Problem 1: Is it decidable whether a regular language is splicing? If yes, how hard it is?

Problem 2: Characterize regular splicing languages.

Open problems (since 1987)

Problem 1: Is it decidable whether a regular language is splicing? If yes, how hard it is?

Problem 2: Characterize regular splicing languages.

Conjecture: Every finite splicing system generates a language with a constant.

Recent results

Conjecture: Every finite splicing system generates a language with a constant.

Bonizzoni & J. 2011 Conjecture is true.

The proof relies on the structure of the minimal deterministic finite state automaton for a splicing language.

Recent results

Conjecture: Every finite splicing system generates a language with a constant.

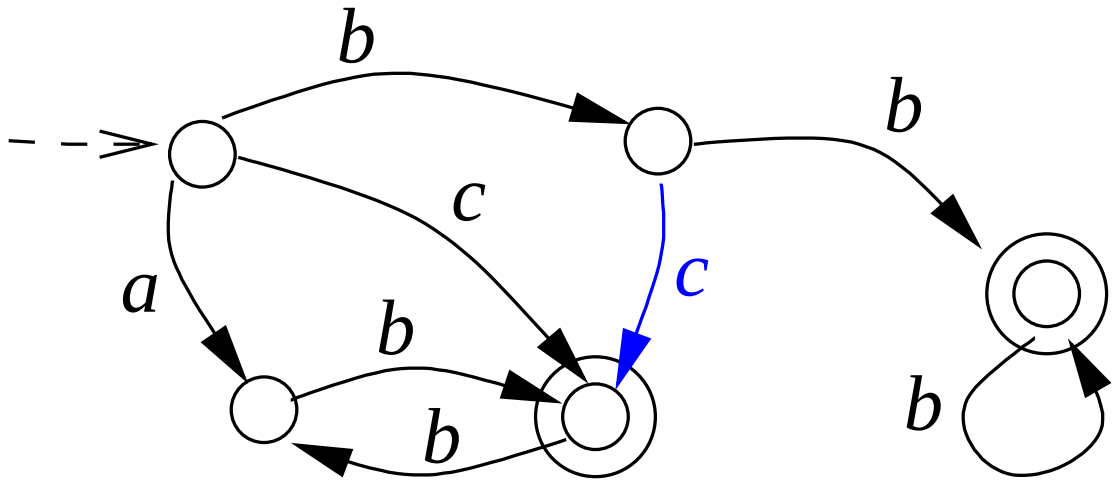
Bonizzoni & J. 2011 Conjecture is true.

The proof relies on the structure of the minimal deterministic finite state automaton for a splicing language.

Problem 1: Is it decidable whether a regular language is splicing?

L. Kari & Kopetcki 2012 It is decidable whether a regular language is splicing. The proof uses the size of the syntactic monoid of the regular language to bound the spice sites.

Examples



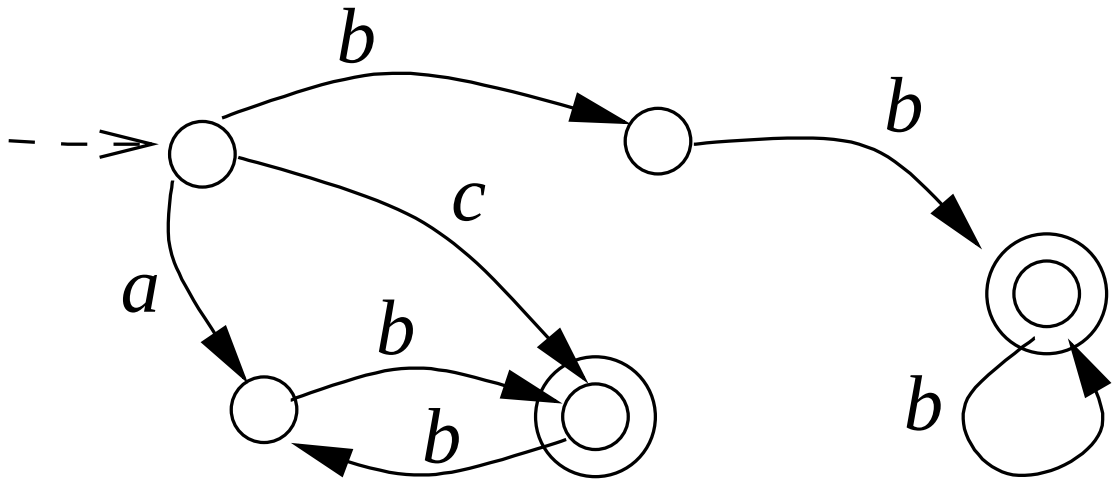
reflexive splicing language:

initial strings: $ab, abbb, c, cbb, bc, bcbb$

rules: $r_1 = (abbb, 1)(ab, bb)$; $r_2 = (cbb, 1)(c, bb)$;
$r_3 = (b, c)(a, b)$; $r_4 = (b, c)(c, b)$

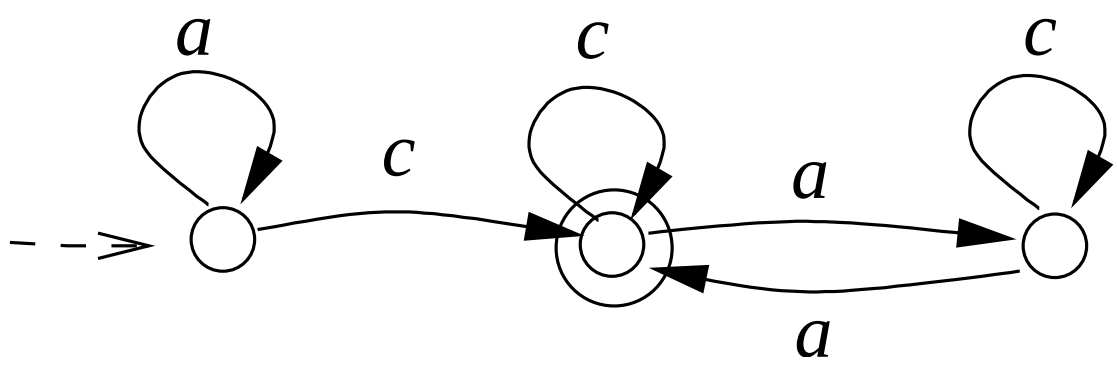
(note: adding reflexive rules doesn't change the language).

Examples



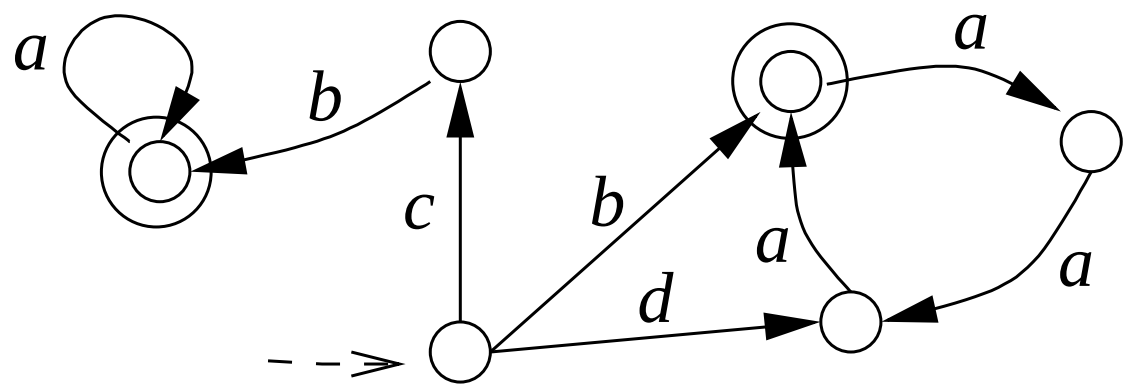
non-splicing language with constants

Examples



non-splicing language without a constant

Examples



non-reflexive splicing language;

initial strings: $ba^3, b, cba, cb, daa^3, da$

splicing rules: $r_1 = (cba, 1)(cb, a)$, $r_2 = (daa^3, 1)(da, 1)$, $r_3 = (b, a^3)(da, 1)$

Automata

A deterministic finite state automaton (DFA) is a 5-tuple $\mathcal{A} = (Q, A, \delta, q_0, T)$, where

- Q is a finite set of states,
- A is a finite alphaet
- $q_0 \in Q$ is the initial state,
- $T \subseteq Q$ is the set of terminal (final) states,
- δ is the transition function $\delta : Q \times A \to Q$.

$L(\mathcal{A})$ is the language recognized by $\mathcal{A}$

L is regular iff there is DFA $\mathcal{A}$ such that $L = L(\mathcal{A})$

For each regular L there is a unique minimal DFA (mDFA).

Notation

$qw = \delta(q, w)$ when δ is understood.

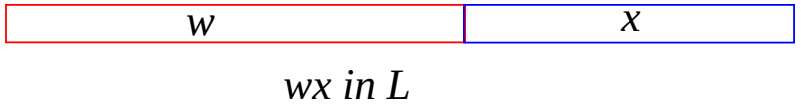
Similarly, $Qw =$ the image of the set Q under the map $w : Q \to Q$ defined with $w(q) = qw$.

$F(L)$ is the set of all factors of words in L, (x is a factor of word w iff $w = zxy$ for $z, y \in A^*$).

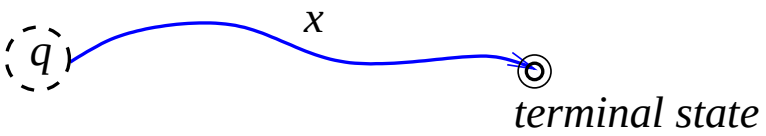
Notation

right context of a word $w \in A^*$ with respect to a language L is

$$\mathcal{R}_L(w) = \{ x \in A^* \mid wx \in L \}$$



right context of a state in $\mathcal{A}$ is $\mathcal{R}_{\mathcal{A}}(q) = \{ x \in A^* \mid qx \in T \}$.



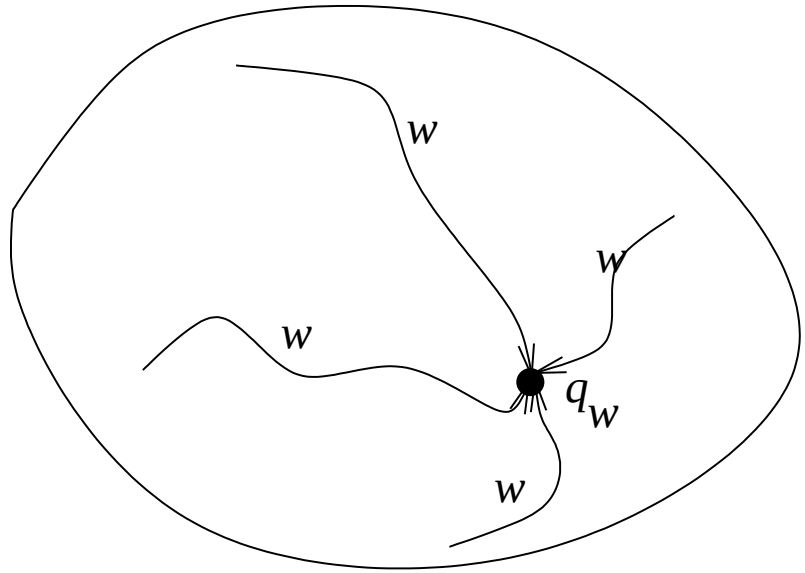
In mDFA $\mathcal{A} = (Q, A, \delta, q_0, T)$ for L,

$$q_0 w = q \text{ iff } \mathcal{R}_L(w) = \mathcal{R}_{\mathcal{A}}(q).$$

In fact, $\mathcal{R}_L(w) = \mathcal{R}_{\mathcal{A}}(q)$ iff $\mathcal{R}_L(wa) = \mathcal{R}_{\mathcal{A}}(qa)$ for all $a \in A$, and therefore $\mathcal{R}_{\mathcal{A}}(q) = \mathcal{R}_{\mathcal{A}}(q')$ implies $q = q'$.

Synchronization

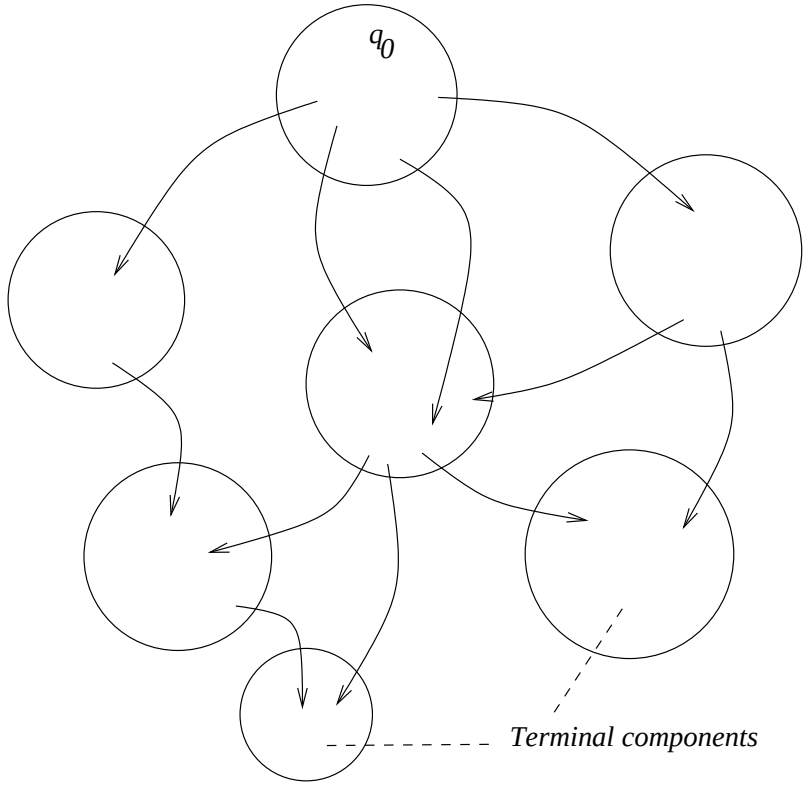
w is a synchronizing word there is a state q_w such that every path in the automaton with label w terminates in q_w, we say that q_w is a synchronizing state, synchronized by w



Fact: A word $w \in A^+$ is a constant of L if and only if w is synchronizing for mDFA of L.

Transitive components

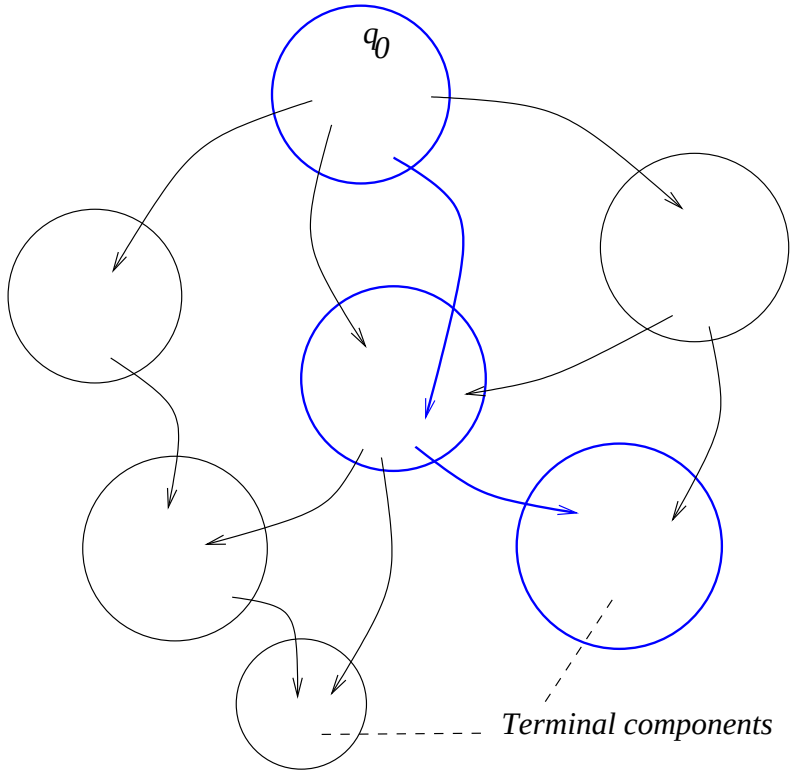
C is a transitive component for an automaton if it is strongly connected.
It is terminal transitive if no transitions leave the component.



Transitive components

C is a transitive component for an automaton if it is strongly connected.
It is terminal transitive if no transitions leave the component.

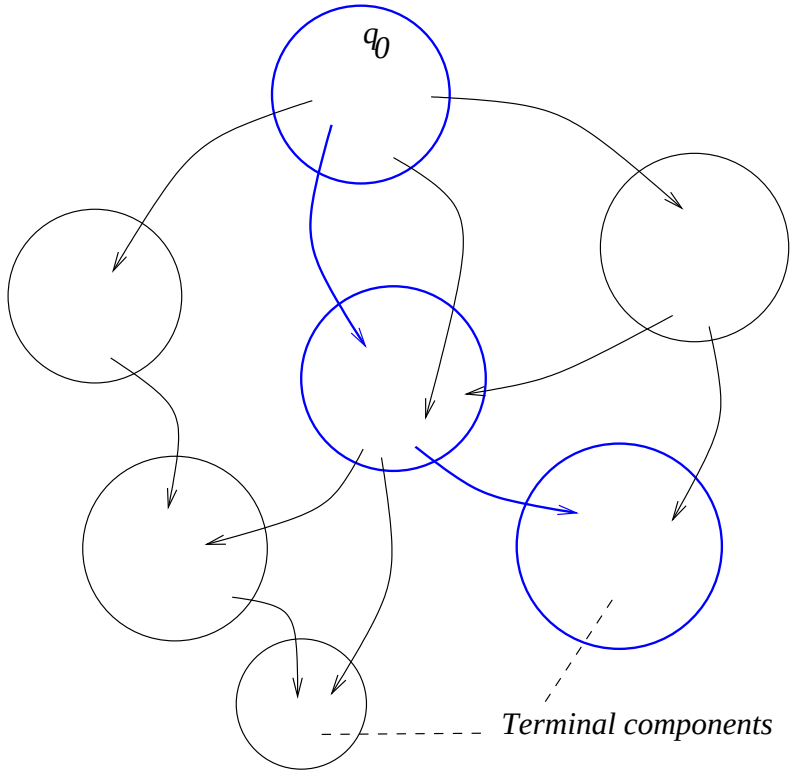
path-automaton within a FSA



Transitive components

C is a transitive component for an automaton if it is strongly connected.
It is terminal transitive if no transitions leave the component.

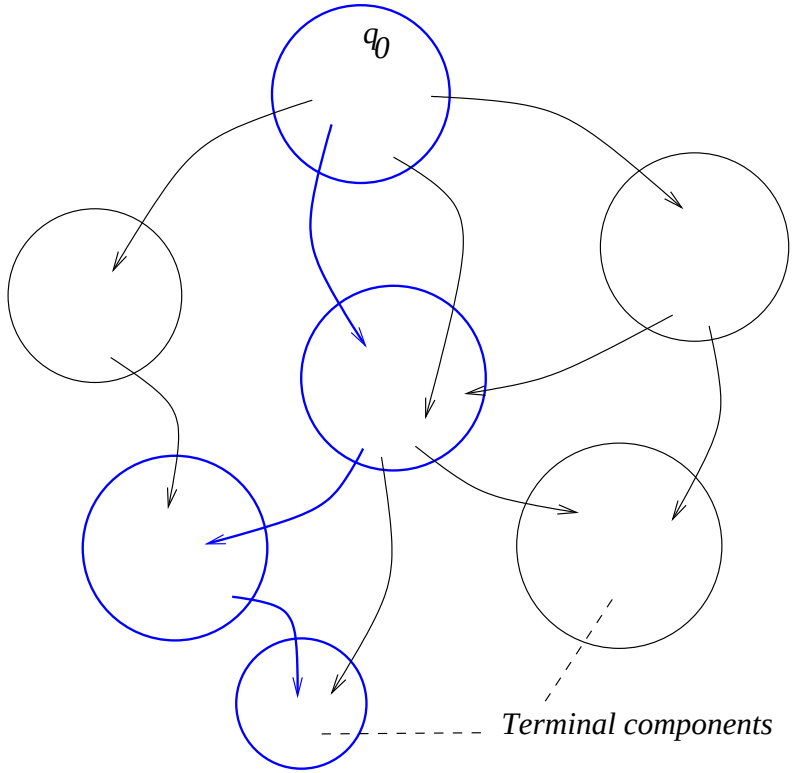
path-automaton within a FSA



Transitive components

C is a transitive component for an automaton if it is strongly connected.
It is terminal transitive if no transitions leave the component.

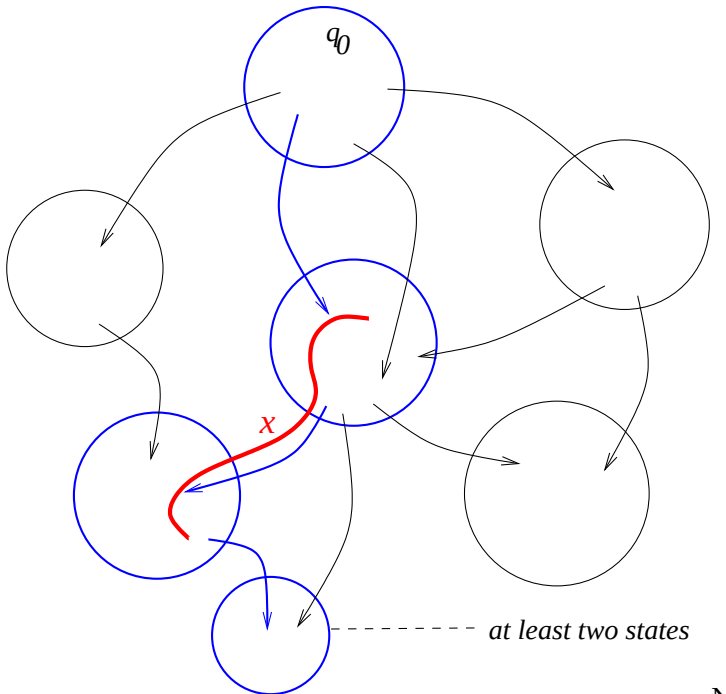
path-automaton within a FSA



Proposition
Let L be a regular language, $x \in F(L)$ and $\mathcal{A}$ be an mDFA for L. At least one of the two cases holds:

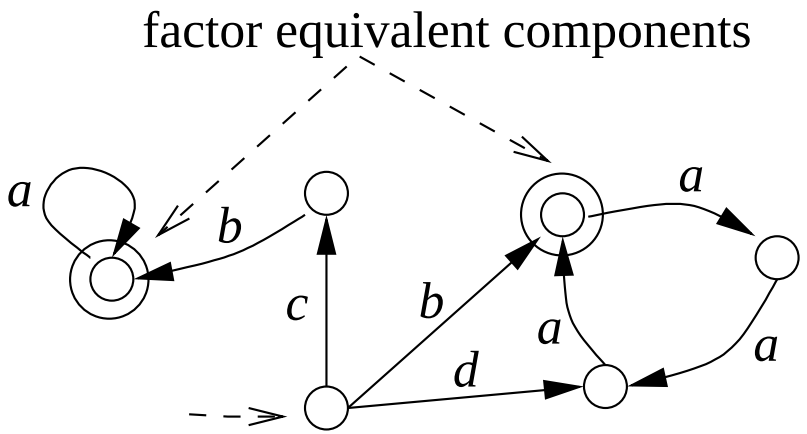
(i) x is a factor of a constant for L,

(ii) there is a path-automaton containing a path labeled x having a non-trivial terminal transitive component with two non-zero states.



Transitive Components

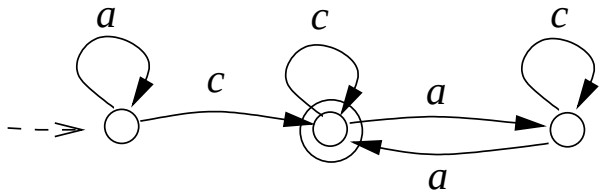
$L(C) =$ set of all labels of paths in C. Two transitive components C and C' are factor equivalent if $L(C) = L(C')$



FACT: Given a deterministic path-automaton $\mathcal{A}$ with C_T its terminal component.Then one of the following holds:

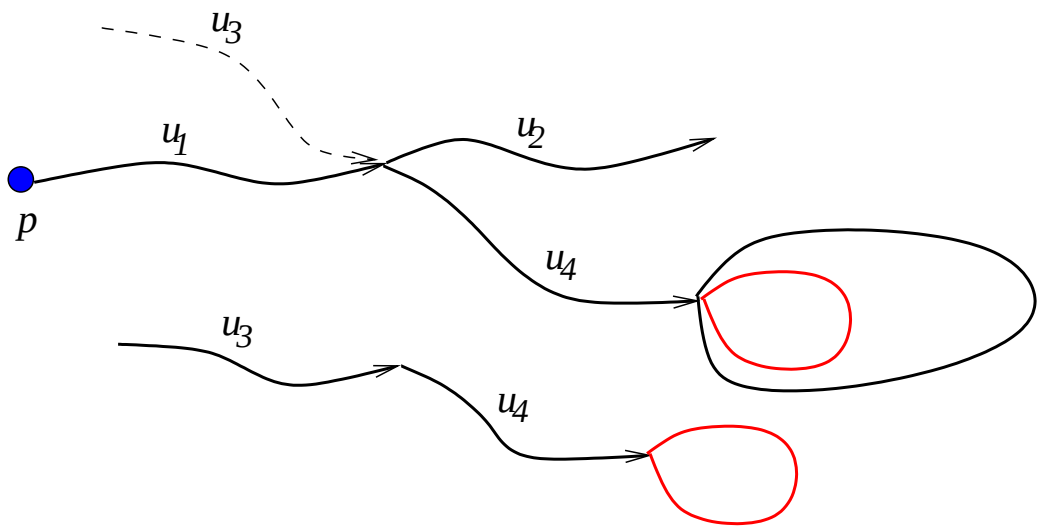
(a) $\mathcal{A}$ has a synchronizing word, or,

(b) $F(L(\mathcal{A})) \subseteq L(C_T)$.



Splicing Revisited

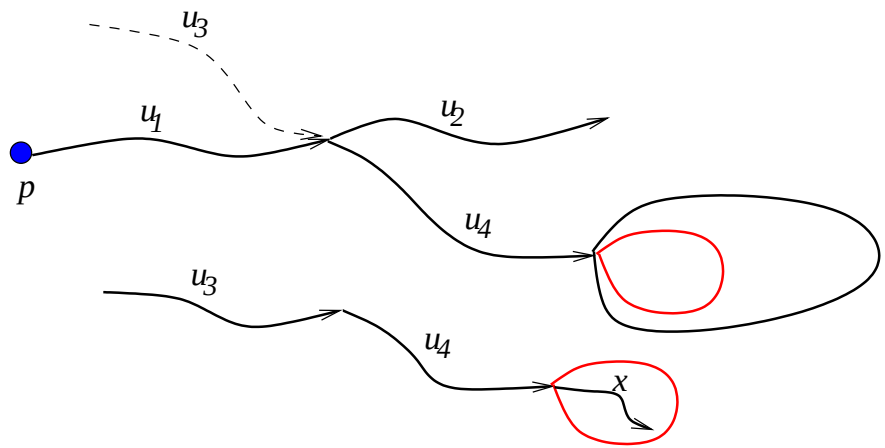
Let $\mathcal{A}$ be the mDFA for a regular splicing language L. The word u_1u_4 is said to be a paste site for a state $p \in Q$ with respect to a splicing rule $r = (u_1, u_2)(u_3, u_4)$ if $\mathcal{R}_L(u_3u_4) \subseteq \mathcal{R}_{\mathcal{A}}(pu_1u_4)$ and $pu_1u_2 \neq \emptyset$.



Splicing Revisited

Let $S = (A, I, R)$ be a finite splicing system and $r = (u_1, u_2)(u_3, u_4)$ be a splicing rule in R. Set $L = L(S)$.

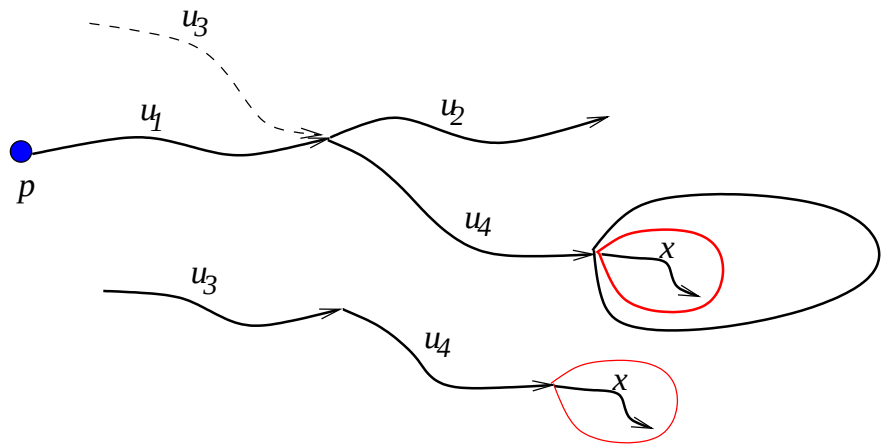
Fact: Let $\mathcal{A}$ be a finite state automaton for L. If u_1u_4 is a paste site at state p for rule r then for every $x \in \mathcal{R}_L(u_1u_4) \cap \mathcal{R}_L(u_3u_4)$ we have that u_1u_4x is a paste site at p for rule $r = (u_1, u_2)(u_3, u_4x)$.



Splicing Revisited

Let $S = (A, I, R)$ be a finite splicing system and $r = (u_1, u_2)(u_3, u_4)$ be a splicing rule in R. Set $L = L(S)$.

Fact: Let $\mathcal{A}$ be a finite state automaton for L. If u_1u_4 is a paste site at state p for rule r then for every $x \in \mathcal{R}_L(u_1u_4) \cap \mathcal{R}_L(u_3u_4)$ we have that u_1u_4x is a paste site at p for rule $r = (u_1, u_2)(u_3, u_4x)$.



Sketch of Proof of Conjecture

L is regular splicing language and $\mathcal{A} = (Q, A, \delta, q_0, T) =$ mDFA for language L.

Sketch of Proof of Conjecture

L is regular splicing language and $\mathcal{A} = (Q, A, \delta, q_0, T) =$ mDFA for language L.

Consider a terminal component C whose factor language is minimal among all terminal components in the automaton.

Sketch of Proof of Conjecture

L is regular splicing language and $\mathcal{A} = (Q, A, \delta, q_0, T) =$ mDFA for language L.

Consider a terminal component C whose factor language is minimal among all terminal components in the automaton.

Take $\mathfrak{C} = \{C_1 = C, C_2, \cdots, C_k\}$ be the set of all terminal components of the automaton that are factor-equivalent to C.

Sketch of Proof of Conjecture

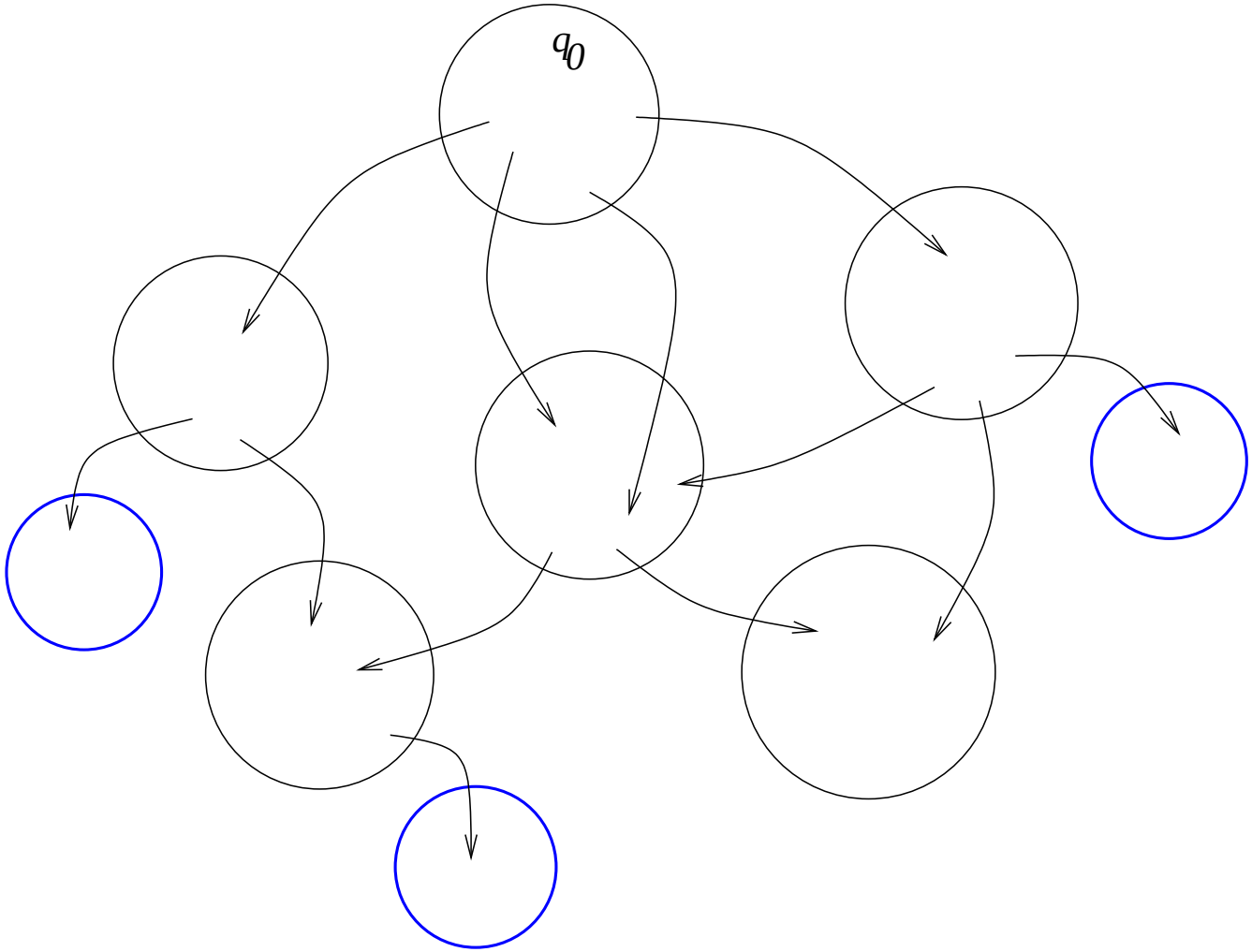
L is regular splicing language and $\mathcal{A} = (Q, A, \delta, q_0, T) =$ mDFA for language L.

Consider a terminal component C whose factor language is minimal among all terminal components in the automaton.

Take $\mathfrak{C} = \{C_1 = C, C_2, \cdots, C_k\}$ be the set of all terminal components of the automaton that are factor-equivalent to C.

Take the set of all states in these components:
$\mathcal{F} = \{q_1, \cdots, q_n\} = \cup_{i=1}^{k} Q(C_i)$.

Sketch of Proof Cont.



Sketch of Proof Cont.

Choose $\bar{q} \in \mathcal{F}$ with $\bar{q} \in Q(\bar{C})$ for some component $\bar{C} \in \mathfrak{C}$ such that $\mathcal{R}_{\mathcal{A}}(\bar{q})$ is minimal among all $\mathcal{R}_{\mathcal{A}}(p)$ for $p \in \mathcal{F}$.
(we show that $\bar{q}$ is a synchronizing state)

Sketch of Proof Cont.

Choose $\bar{q} \in \mathcal{F}$ with $\bar{q} \in Q(\bar{C})$ for some component $\bar{C} \in \mathfrak{C}$ such that $\mathcal{R}_{\mathcal{A}}(\bar{q})$ is minimal among all $\mathcal{R}_{\mathcal{A}}(p)$ for $p \in \mathcal{F}$.
(we show that $\bar{q}$ is a synchronizing state)

Let $w \in A^*$ be the shortest word such that $q_0 w = \bar{q}$

Sketch of Proof Cont.

Choose $\bar{q} \in \mathcal{F}$ with $\bar{q} \in Q(\bar{C})$ for some component $\bar{C} \in \mathfrak{C}$ such that $\mathcal{R}_{\mathcal{A}}(\bar{q})$ is minimal among all $\mathcal{R}_{\mathcal{A}}(p)$ for $p \in \mathcal{F}$.
(we show that $\bar{q}$ is a synchronizing state)

Let $w \in A^*$ be the shortest word such that $q_0 w = \bar{q}$

Consider a word c that "identifies" $\bar{C}$ (in the sense that if $c \in L(C)$) then $L(\bar{C}) \subseteq L(C)$).
Moreover, we can choose c such that $\bar{q}c = \bar{q}$.

Sketch of Proof Cont.

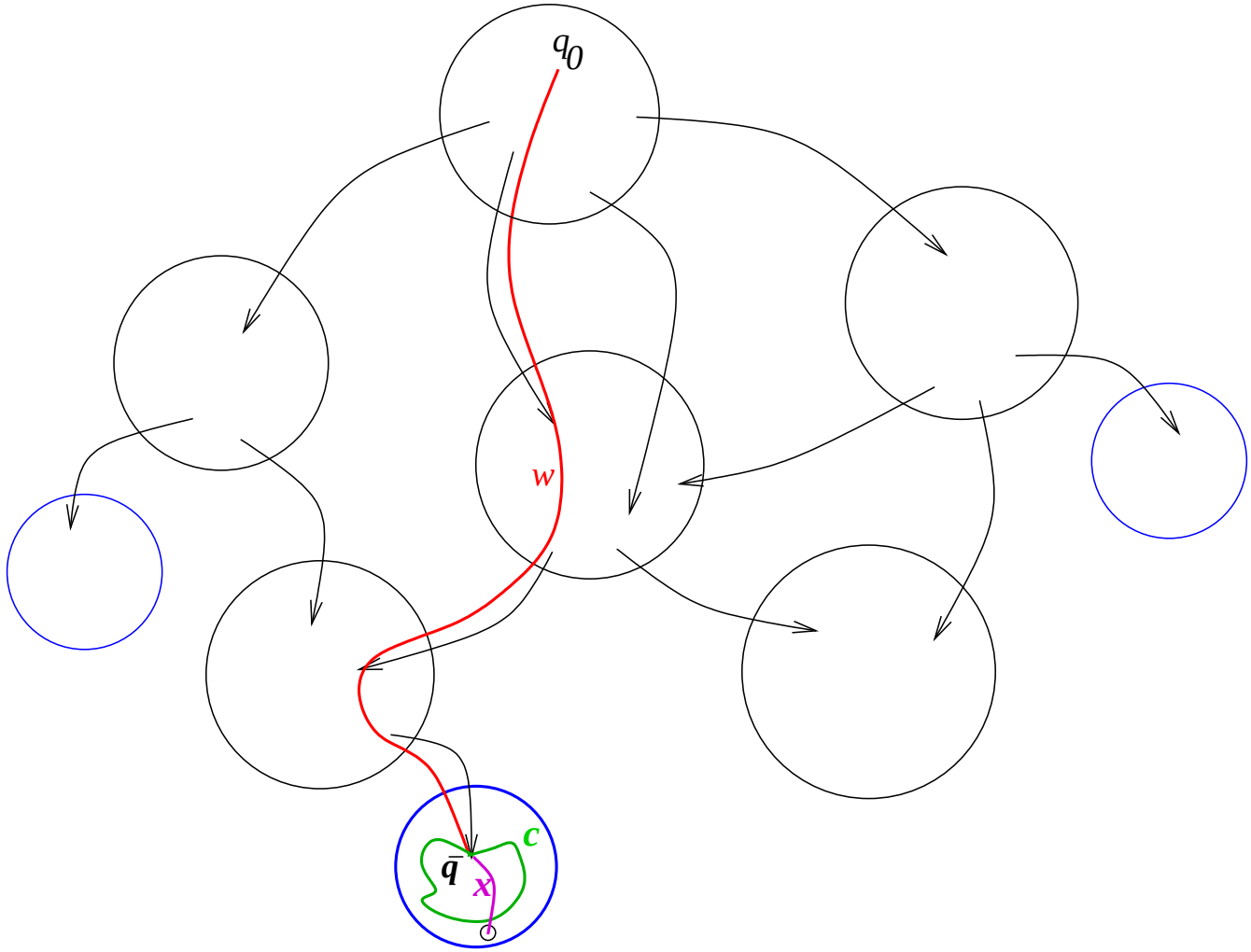
Choose $\bar{q} \in \mathcal{F}$ with $\bar{q} \in Q(\bar{C})$ for some component $\bar{C} \in \mathfrak{C}$ such that $\mathcal{R}_{\mathcal{A}}(\bar{q})$ is minimal among all $\mathcal{R}_{\mathcal{A}}(p)$ for $p \in \mathcal{F}$.
(we show that $\bar{q}$ is a synchronizing state)

Let $w \in A^*$ be the shortest word such that $q_0 w = \bar{q}$

Consider a word c that "identifies" $\bar{C}$ (in the sense that if $c \in L(C)$) then $L(\bar{C}) \subseteq L(C)$).
Moreover, we can choose c such that $\bar{q}c = \bar{q}$.

Then $wc^*x \subseteq L$ for some $x \in A^*$, and there is a rule $r = (u_1, u_2)(u_3, u_4)$ that is used infinite number of times for generation of wc^*x.

Sketch of Proof Cont.



Sketch of Proof Cont.

There must exist infinite number of pairs of words $v'u_1u_2v'' \in L$ and $w'u_3u_4w'' \in L$ such that $v'u_1u_4w'' \in wc^*x$.

Sketch of Proof Cont.

There must exist infinite number of pairs of words $v'u_1u_2v'' \in L$ and $w'u_3u_4w'' \in L$ such that $v'u_1u_4w'' \in wc^*x$.

Moreover: For arbitrarily large i's, it holds that c^i is a factor in the right context of u_3u_4.

Sketch of Proof Cont.

There must exist infinite number of pairs of words $v'u_1u_2v'' \in L$ and $w'u_3u_4w'' \in L$ such that $v'u_1u_4w'' \in wc^*x$.

Moreover: For arbitrarily large i's, it holds that c^i is a factor in the right context of u_3u_4.

We can show that u_1u_4 is a paste site at state p for rule $r = (u_1u_2)(u_3u_4)$ where p is a state in the path-automaton determined by w.

Sketch of Proof Cont.

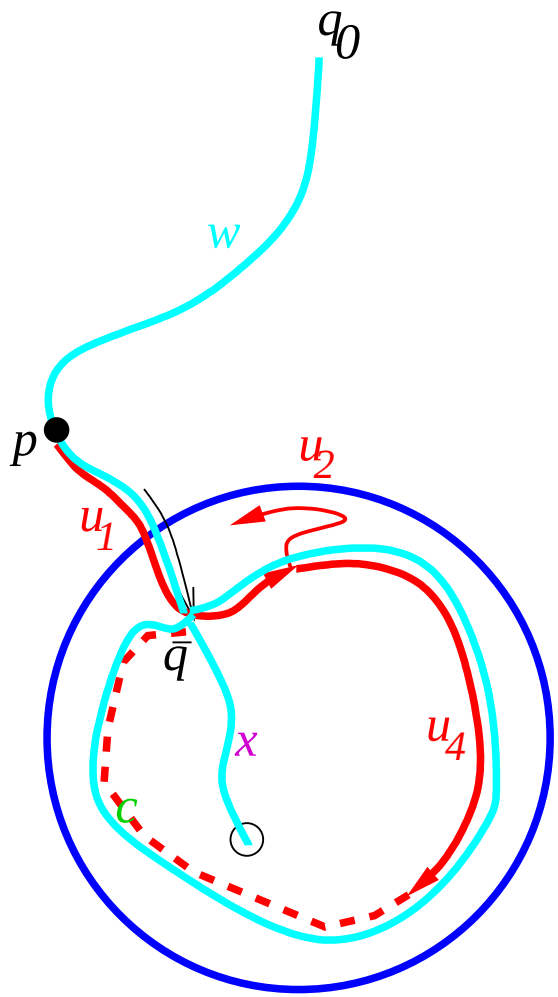
There must exist infinite number of pairs of words $v'u_1u_2v'' \in L$ and $w'u_3u_4w'' \in L$ such that $v'u_1u_4w'' \in wc^*x$.

Moreover: For arbitrarily large i's, it holds that c^i is a factor in the right context of u_3u_4.

We can show that u_1u_4 is a paste site at state p for rule $r = (u_1u_2)(u_3u_4)$ where p is a state in the path-automaton determined by w.

Extend u_4 to $u_4' = u_4\bar{w}c^i$ for a nicely chosen i.

Sketch of Proof Cont.



Sketch of Proof Cont.

Let $\hat{q}$ be a state in $Qu_3u'_4$. ($\hat{q}$ is in component $\hat{C}$)

(we observe that $\bar{q}$ is synchronized by $u_3u'_4$, i.e., $\hat{q} = \bar{q}$)

Sketch of Proof Cont.

Let $\hat{q}$ be a state in Qu_3u_4'. ($\hat{q}$ is in component $\hat{C}$)

(we observe that $\bar{q}$ is synchronized by u_3u_4', i.e., $\hat{q} = \bar{q}$)

By the choice of c, it holds that

$$\mathcal{R}_{\mathcal{A}}(\hat{q}) \subseteq \mathcal{R}_L(u_3u_4') \subseteq \mathcal{R}_{\mathcal{A}}(pu_1u_4' = \bar{q}).$$

If $\hat{q} \in \mathcal{F}$ we are done: $\hat{q} = \bar{q}$.

Sketch of Proof Cont.

Let $\hat{q}$ be a state in Qu_3u_4'. ($\hat{q}$ is in component $\hat{C}$)

(we observe that $\bar{q}$ is synchronized by u_3u_4', i.e., $\hat{q} = \bar{q}$)

By the choice of c, it holds that

$$\mathcal{R}_{\mathcal{A}}(\hat{q}) \subseteq \mathcal{R}_L(u_3u_4') \subseteq \mathcal{R}_{\mathcal{A}}(pu_1u_4' = \bar{q}).$$

If $\hat{q} \in \mathcal{F}$ we are done: $\hat{q} = \bar{q}$.

Observe that

$$L(\hat{C}) = L(\bar{C}) \text{ and } \hat{C} \text{ is terminal}$$

($L(\hat{C}) = L(\bar{C})$ follows from the fact that c “identifies" $\bar{C}$ and $\hat{C}$ must be terminal by properties of path-automata.) □

decidability

Syntactic monoid

Context of x with respect to L is the set

$$C_L(x) = \{(u, v) \mid uxv \in L\}$$

for $x, y \in A^*$ and $L \subseteq A^*$ we have

$$x \sim_L y \Leftrightarrow C_L(x) = C_L(y)$$

The syntactic monoid of L is $S(L) = A^*/\sim_L$.

Fact: L is regular if and only if $S(L)$ is finite.

(very) Brief sketch of decidability

L. Kari & Kopetcki 2012

- A rule r respects L if for all $x, y \in L$, $(x, y) \to_r z$ implies $z \in L$.

  If $(u_1, u_2)(u_3, u_4)$ respects L then so do:

  $$(xu_1, u_2)(u_3, u_4), \quad (u_1, u_2x)(u_3, u_4),$$

  $$(u_1, u_2)(xu_3, u_4), \quad (u_1, u_2)(u_3, u_4x)$$

- Let $n = |S(L)|$. For each $[x] \in S(L)$, $[x] \cap A^{<n} \neq \emptyset$.

(very) Brief sketch of decidability

- If r respects L then each rule $s \in [r]$ respects L where

$$[r] = [u_1] \times [u_2] \times [u_3] \times [u_4]$$

- Consider a splicing language L generated by a splicing system (A, I, R) with

  $\ell =$ max length of words in I and

  $\mu =$ largest component of a rule in R in lexicographic order.

  Set $W_\mu = \{w \in A^* \mid w \leq_{\text{lex}} \mu\}$. Then:

$$(A, L \cap A^{\leq \ell}, \{r \in W_\mu \mid r \text{ respects } L\})$$

  generates L.

(very) Brief sketch of decidability

- Show that if xyz is obtained by series of splicings with $|y| \geq |\mu|$ then the rules and the words involved can be chosen to be of length not significantly larger than $\max\{|x|, |z|\}$.

(very) Brief sketch of decidability

- Show that if xyz is obtained by series of splicings with $|y| \geq |\mu|$ then the rules and the words involved can be chosen to be of length not significantly larger than $\max\{|x|, |z|\}$.
- If L is splicing, then it is splicing with rules that are "nice" (splice sites are shorter than $n^2 + 10n$) over initial strings not longer than $n^2 + 6n$.

(very) Brief sketch of decidability

- Show that if xyz is obtained by series of splicings with $|y| \geq |\mu|$ then the rules and the words involved can be chosen to be of length not significantly larger than $\max\{|x|, |z|\}$.
- If L is splicing, then it is splicing with rules that are "nice" (splice sites are shorter than $n^2 + 10n$) over initial strings not longer than $n^2 + 6n$.
- Decidability: there are finite number of splicing rules with splice/paste sites of length $< n^2 + 10n$. Check all possibilities.... (Pixton's regularity proof provides construction of an automaton for a given splicing system)

Still Open:

Problem 1: How fast we can determine whether a regular language is splicing?

Problem 2: Characterize regular splicing languages.

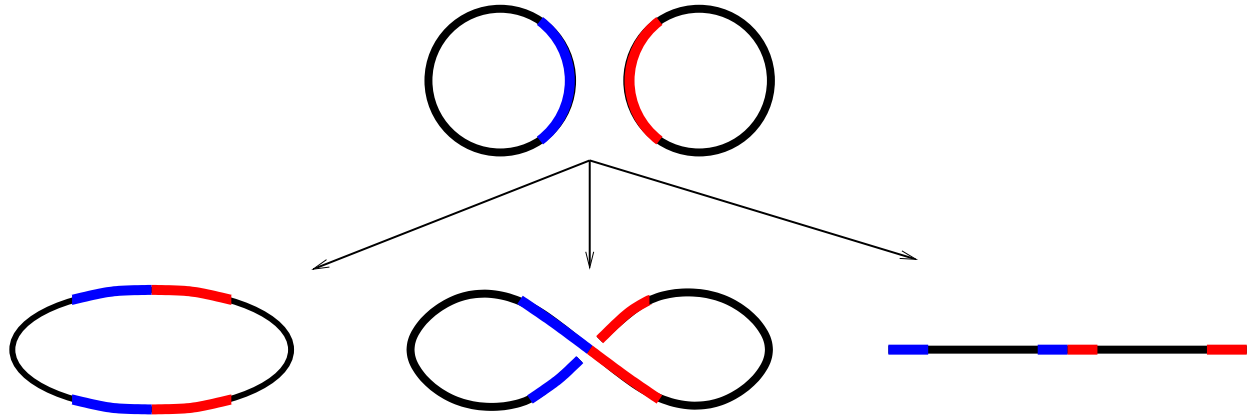
Note: $((aa)^*)^{\text{circ}}$ is splicing, but $(b(aa)^*)^{\text{circ}}$ is not splicing.

Still Open:

Problem 1: How fast we can determine whether a regular language is splicing?

Problem 2: Characterize regular splicing languages.

virtually all questions about characterization of circular languages obtained by splicing are still open



Note: $((aa)^*)^{\text{circ}}$ is splicing, but $(b(aa)^*)^{\text{circ}}$ is not splicing.

THANK YOU!!