How do we compare transducers?

Amaldev Manuel

School of Mathematics and Computer Science, IIT Goa

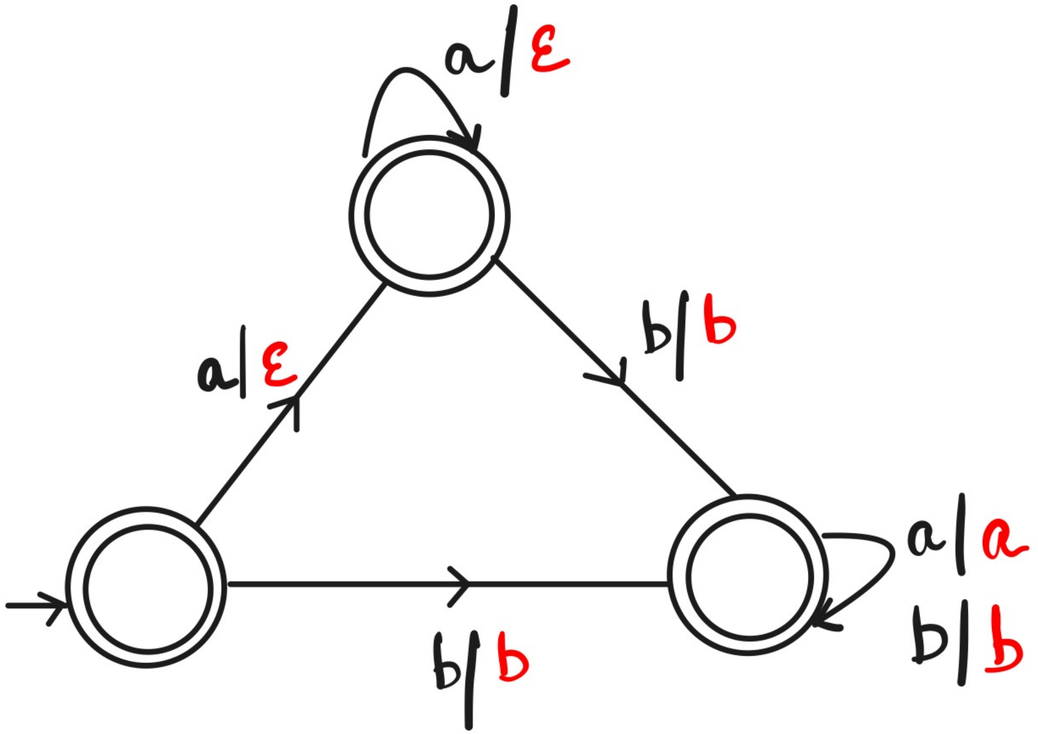
Joint work with **C. Aiswarya** (CMI) and **Saina Sunny** (IIT Goa)

Based on: 1. Edit Distance of Finite State Transducers (ICALP 2024)

2. Deciding Conjugacy of Rational Relation (DLT 2024)

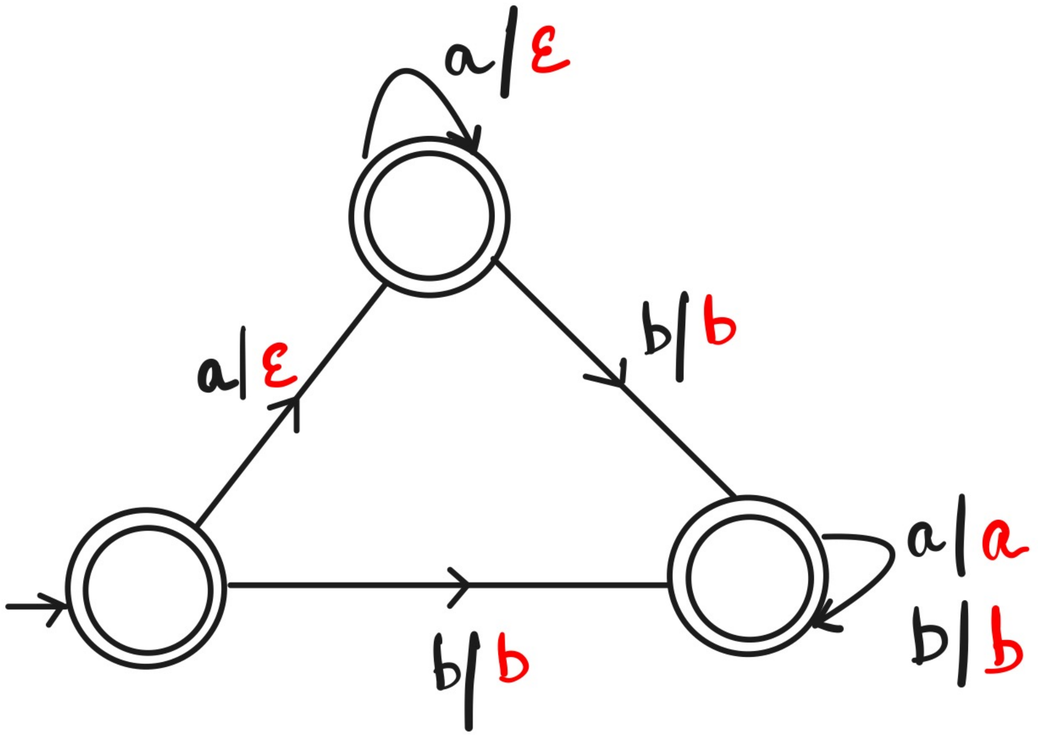
Transducers

- Finite state machines that produce output.



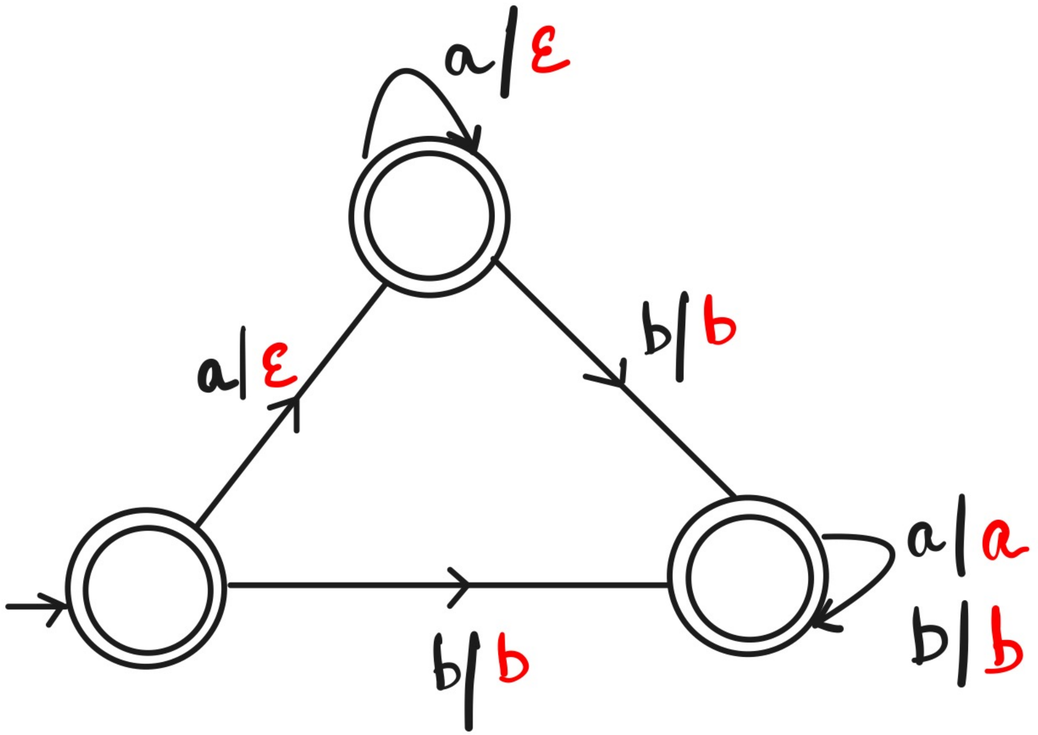
Transducers

- Finite state machines that produce output.
- Functional



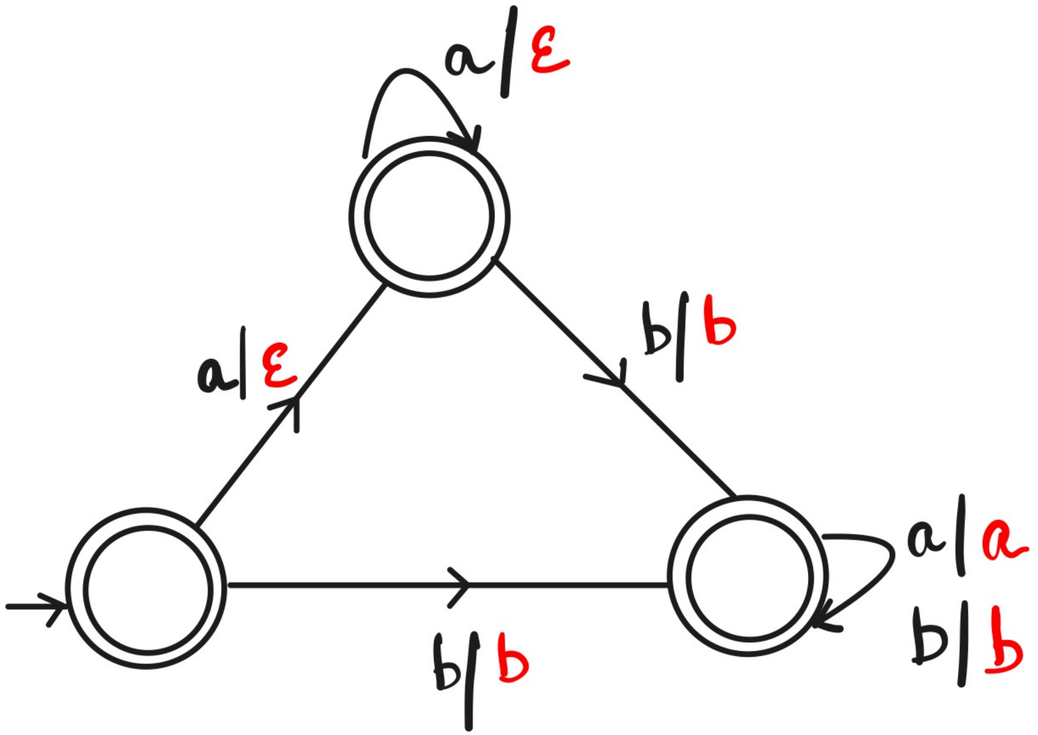
Transducers

- Finite state machines that produce output.
- Functional
  - Sequential (DFA)



Transducers

- Finite state machines that produce output.
- Functional
  - Sequential (DFA)
  - Rational (Unambiguous NFA)



Closeness of transducers

Closeness of transducers

- Functional equivalence

Closeness of transducers

- Functional equivalence
- Relax it : on any input, the respective outputs are “approximately” the same

Closeness of transducers

- Functional equivalence
- Relax it : on any input, the respective outputs are “approximately” the same
- We can convert one output to another by a few “edits”

Closeness of transducers

- Functional equivalence
- Relax it : on any input, the respective outputs are “approximately” the same
- We can convert one output to another by a few “edits”
- Ex: substitute a letter with another

Closeness of transducers

- Functional equivalence
- Relax it : on any input, the respective outputs are “approximately” the same
- We can convert one output to another by a few “edits”
- Ex: substitute a letter with another
- Ex: insert a letter, or delete a letter

Closeness of transducers

- Functional equivalence
- Relax it : on any input, the respective outputs are “approximately” the same
- We can convert one output to another by a few “edits”
- Ex: substitute a letter with another
- Ex: insert a letter, or delete a letter
- The number of such edits needed can indicate some “distance” between two words

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.

- $$d(T_1, T_2) = \begin{cases} sup\ \{d(T_1(w), T_2(w)) \mid w \in dom(T_1)\} & \text{if } dom(T_1) = dom(T_2) \\ \infty & otherwise \end{cases}$$

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.
- $$d(T_1, T_2) = \begin{cases} sup\ \{d(T_1(w), T_2(w)) \mid w \in dom(T_1)\} & \text{if } dom(T_1) = dom(T_2) \\ \infty & otherwise \end{cases}$$
- T_1 and T_2 are close if $d(T_1, T_2)$ is finite.

Some metrics

Some metrics

- Discrete metric $d(u,v) = \begin{cases} 0 & if\ u = v \\ \infty & otherwise \end{cases}$ (Equivalence Problem)

Some metrics

- Discrete metric $d(u,v) = \begin{cases} 0 & if\ u = v \\ \infty & otherwise \end{cases}$ (Equivalence Problem)
- Hamming Distance $d(u,v) = \begin{cases} \infty & if\ |u| \neq |v| \\ |\{i \mid u_i \neq v_i\}| & otherwise \end{cases}$

Some metrics

- Discrete metric $$d(u,v) = \begin{cases} 0 & if\ u = v \\ \infty & otherwise \end{cases}$$ (Equivalence Problem)

- Hamming Distance $$d(u,v) = \begin{cases} \infty & if\ |u| \neq |v| \\ |\{i \mid u_i \neq v_i\}| & otherwise \end{cases}$$

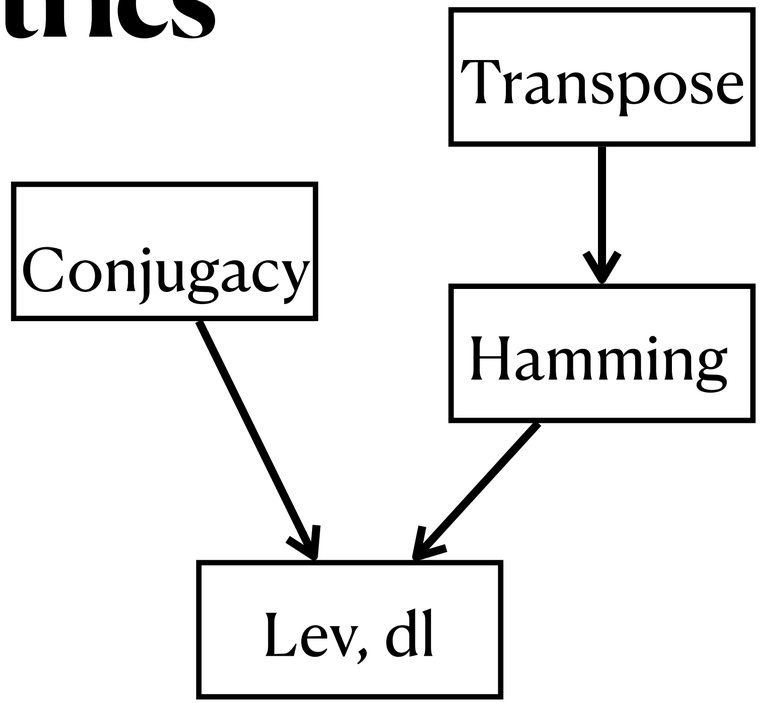
- Levenshtein Edit Distance

$d(u,v) =$ Minimum number of insertions, deletions and substitutions required to convert u to v

Some metrics

- Discrete metric
- Hamming Distance
- Levenshtein Edit Distance - ins,del,sub
- Conjugacy distance - cyclic shifts
- Transposition distance - adjacent transpositions
- Damerau-Levenshtein Edit Distance - insertion, deletions, transpositions and letter-to-letter substitutions

Some metrics



- Discrete metric
- Hamming Distance
- Levenshtein Edit Distance
- Conjugacy distance - cyclic shifts
- Transposition distance - adjacent transpositions
- Damerau-Levenshtein Edit Distance - insertion, deletions, transpositions and letter2letter substitutions

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.
- $$d(T_1, T_2) = \begin{cases} sup\ \{d(T_1(w), T_2(w)) \mid w \in dom(T_1)\} & \text{if } dom(T_1) = dom(T_2) \\ \infty & otherwise \end{cases}$$

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.
- $$d(T_1, T_2) = \begin{cases} sup\ \{d(T_1(w), T_2(w)) \mid w \in dom(T_1)\} & \text{if } dom(T_1) = dom(T_2) \\ \infty & otherwise \end{cases}$$
- Given T_1, T_2 is $d(T_1, T_2)$ computable?

(Distance)

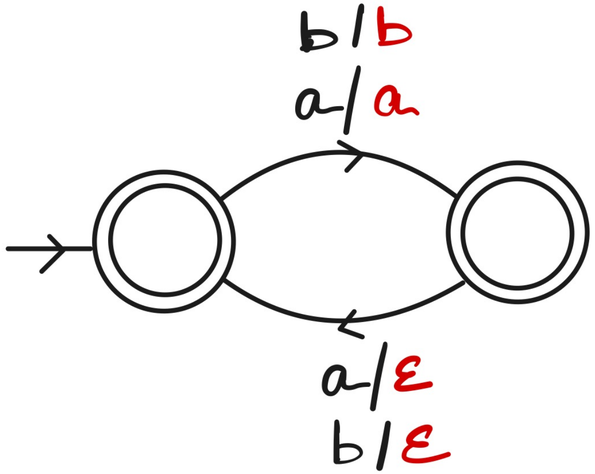
Closeness of transducers

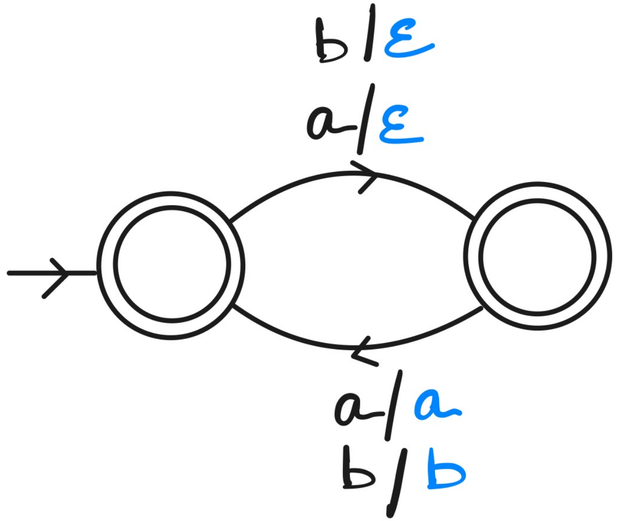
- Let d be a metric on words. Lift it to transducers.
- $$d(T_1, T_2) = \begin{cases} sup\ \{d(T_1(w), T_2(w)) \mid w \in dom(T_1)\} & \text{if } dom(T_1) = dom(T_2) \\ \infty & otherwise \end{cases}$$
- Given T_1, T_2 is $d(T_1, T_2)$ computable? (Distance)
- Given T_1, T_2 is $d(T_1, T_2)$ finite? (Closeness)

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.
- $$d(T_1, T_2) = \begin{cases} sup\ \{d(T_1(w), T_2(w)) \mid w \in dom(T_1)\} & \text{if } dom(T_1) = dom(T_2) \\ \infty & otherwise \end{cases}$$
- Given T_1, T_2 is $d(T_1, T_2)$ computable? (Distance)
- Given T_1, T_2 is $d(T_1, T_2)$ finite? (Closeness)
- Given T_1, T_2 and $k \in \mathbb{N}$, is $d(T_1, T_2)$ at most k? (k-closeness)

Unbounded Edit Distance - Example



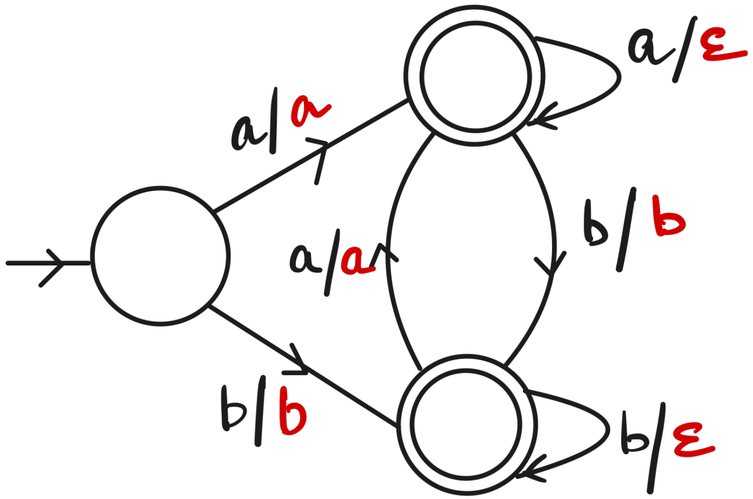


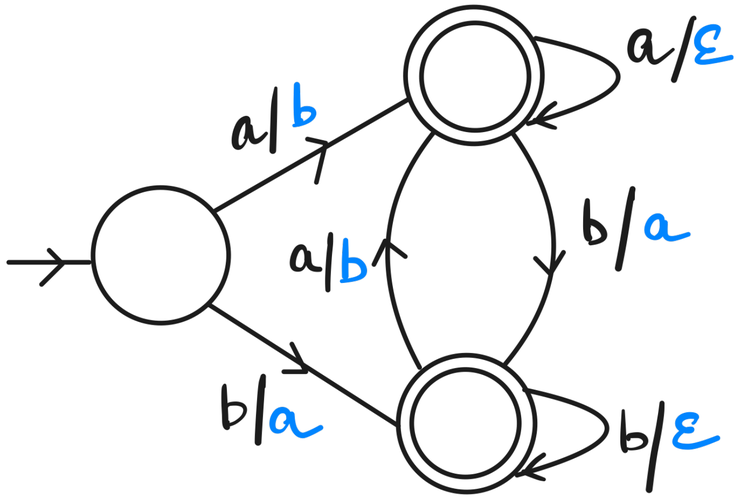
Output letters at odd positions

Output letters at even positions

$$(ab)^n \rightarrow (a^n, b^n)$$

Bounded Lev Edit Distance - Example





- For each block of a, output a
- For each block of b, output b

- For each block of a, output b
- For each block of b, output a

$$aaabbbabbba \rightarrow (ababa, babab)$$

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.
- $$d(T_1, T_2) = \begin{cases} sup\ \{d(T_1(w), T_2(w)) \mid w \in dom(T_1)\} & \text{if } dom(T_1) = dom(T_2) \\ \infty & otherwise \end{cases}$$
- Given T_1, T_2 is $d(T_1, T_2)$ computable? (Distance)
- Given T_1, T_2 is $d(T_1, T_2)$ finite? (Closeness)
- Given T_1, T_2 and $k \in \mathbb{N}$, is $d(T_1, T_2)$ at most k? (k-closeness)

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.

Distance is computable iff closeness and k-closeness are decidable. (for integer-valued metrics)

- Given T_1, T_2 is $d(T_1, T_2)$ computable? (Distance)
- Given T_1, T_2 is $d(T_1, T_2)$ finite? (Closeness)
- Given T_1, T_2 and $k \in \mathbb{N}$, is $d(T_1, T_2)$ at most k? (k-closeness)

k-closeness is decidable

k-closeness is decidable

- Check if domains are the same. Yes: continue; No: not k-close

k-closeness is decidable

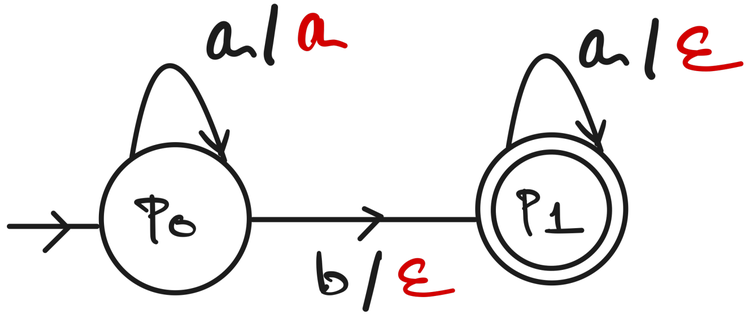
- Check if domains are the same. Yes: continue; No: not k-close
- Construct a cartesian product automaton of given two transducers.

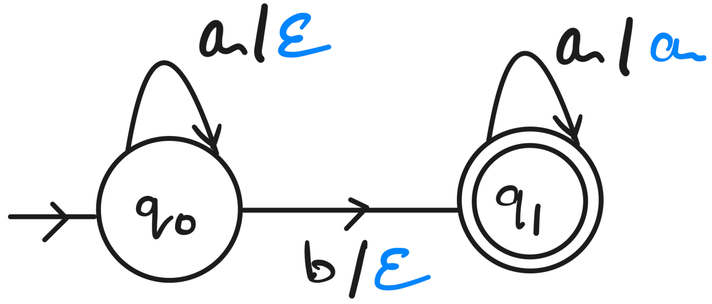
k-closeness is decidable

- Construct a cartesian product automaton of given two transducers.

k-closeness is decidable

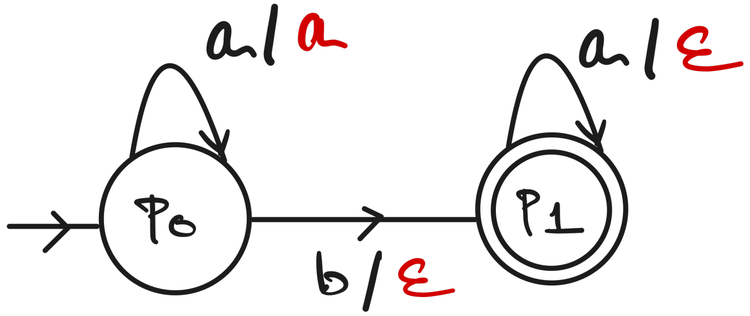
- Construct a cartesian product automaton of given two transducers.



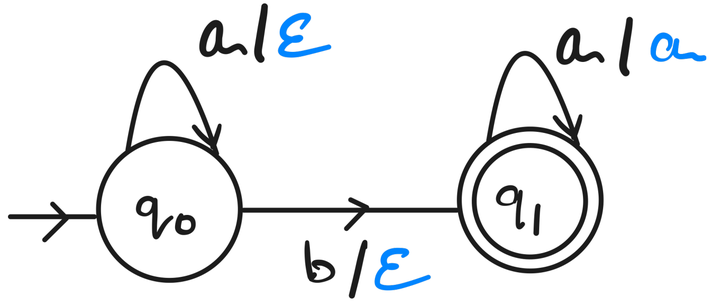


k-closeness is decidable

- Construct a cartesian product automaton of given two transducers.



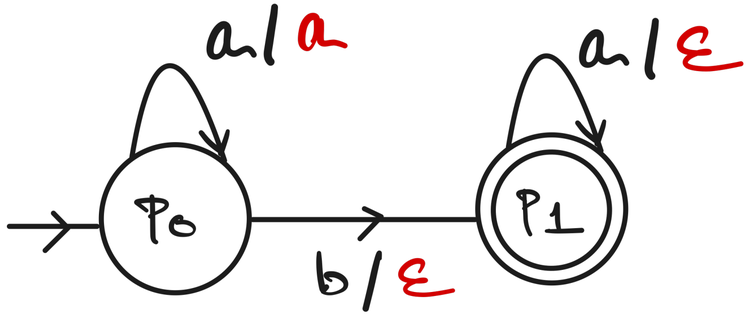
Output a's before b



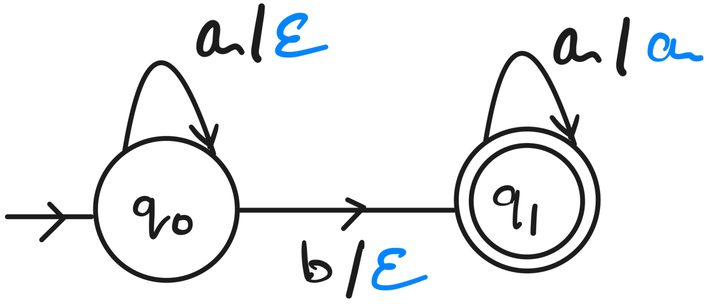
Output a's after b

k-closeness is decidable

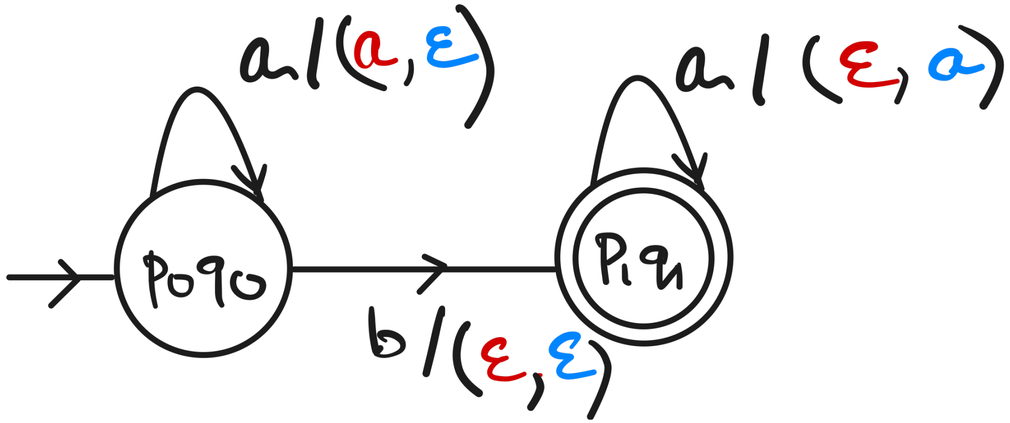
- Construct a cartesian product automaton of given two transducers.



Output a's before b



Output a's after b



$$aaba \rightarrow (a, \epsilon) \cdot (a, \epsilon) \cdot (\epsilon, \epsilon) \cdot (\epsilon, a) = (aa, a)$$

k-closeness is decidable

- Check if domains are the same. Yes: continue; No: not k-close
- Construct a cartesian product automaton of given two transducers.

k-closeness is decidable

- Check if domains are the same. Yes: continue; No: not k-close
- Construct a cartesian product automaton of given two transducers.
- Check if loops generate words of the same length. Yes: continue; No: not k-close

k-closeness is decidable

- Check if domains are the same. Yes: continue; No: not k-close
- Construct a cartesian product automaton of given two transducers.
- Check if loops generate words of the same length. Yes: continue; No: not k-close
- Compute the delay.

k-closeness is decidable

- Check if domains are the same. Yes: continue; No: not k-close
- Construct a cartesian product automaton of given two transducers.
- Check if loops generate words of the same length. Yes: continue; No: not k-close
- Compute the delay.
- Start with budget k. Non-deterministically do edits, update the budget and residues appropriately. Budget is not allowed to be negative.

k-closeness is decidable

- Check if domains are the same. Yes: continue; No: not k-close
- Construct a cartesian product automaton of given two transducers.
- Check if loops generate words of the same length. Yes: continue; No: not k-close
- Compute the delay.
- Start with budget k. Non-deterministically do edits, update the budget and residues appropriately. Budget is not allowed to be negative.
- Check if the language accepted is the domain. Yes: k-close; No: not k-close.

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.

Distance is computable iff closeness and k-closeness are decidable. (for integer-valued metrics)

- Given T_1, T_2 is $d(T_1, T_2)$ computable? (Distance)
- Given T_1, T_2 is $d(T_1, T_2)$ finite? (Closeness)
- Given T_1, T_2 and $k \in \mathbb{N}$, is $d(T_1, T_2)$ at most k? (k-closeness)

closeness is decidable

For Hamming and transposition

closeness is decidable

For Hamming and transposition

- Output lengths must be equal for all words.

closeness is decidable

For Hamming and transposition

- Output lengths must be equal for all words.
  - Loops generate words of same length

closeness is decidable

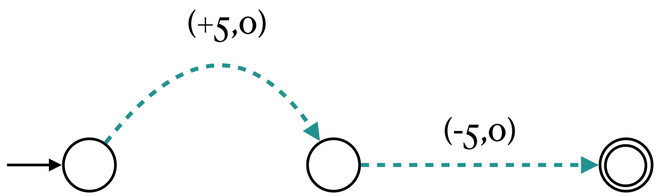
For Hamming and transposition

- Output lengths must be equal for all words.
  - Loops generate words of same length
  - Delay between partial outputs is bounded, and depends only on the state.

closeness is decidable

For Hamming and transposition

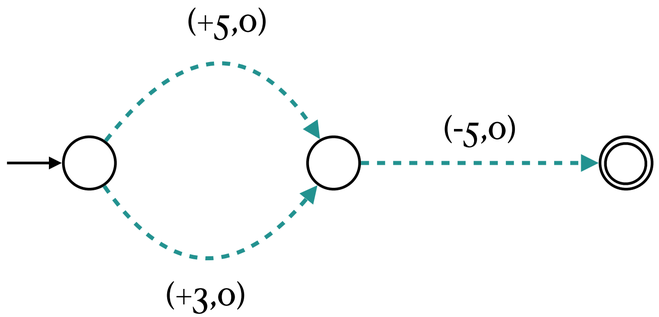
- Output lengths must be equal for all words.
  - Loops generate words of same length
  - Delay between partial outputs is bounded, and depends only on the state.



closeness is decidable

For Hamming and transposition

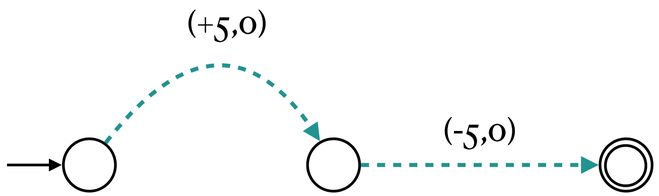
- Output lengths must be equal for all words.
  - Loops generate words of same length
  - Delay between partial outputs is bounded, and depends only on the state.



closeness is decidable

For Hamming and transposition

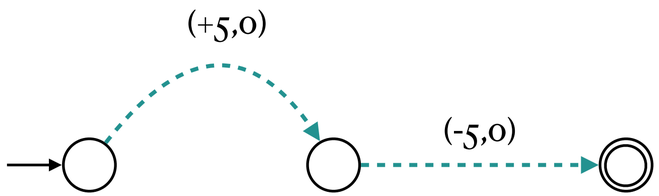
- Output lengths must be equal for all words.
  - Loops generate words of same length
  - Delay between partial outputs is bounded, and depends only on the state.



closeness is decidable

For Hamming and transposition

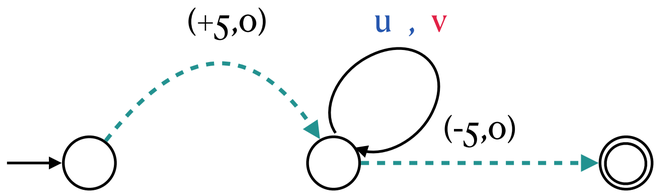
- Output lengths must be equal for all words.
  - Loops generate words of same length
  - Delay between partial outputs is bounded, and depends only on the state.
- Check if all pairs in a loop (modulo the delay on border) are equal (up to some length depending on the delay).



closeness is decidable

For Hamming and transposition

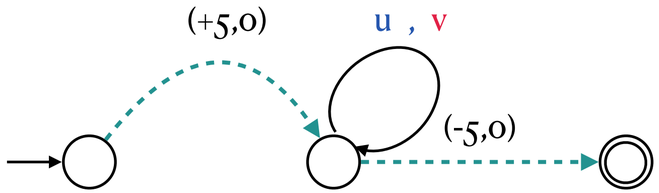
- Output lengths must be equal for all words.
  - Loops generate words of same length
  - Delay between partial outputs is bounded, and depends only on the state.
- Check if all pairs in a loop (modulo the delay on border) are equal (up to some length depending on the delay).

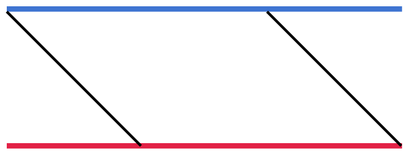


closeness is decidable

For Hamming and transposition

- Output lengths must be equal for all words.
  - Loops generate words of same length
  - Delay between partial outputs is bounded, and depends only on the state.
- Check if all pairs in a loop (modulo the delay on border) are equal (up to some length depending on the delay).

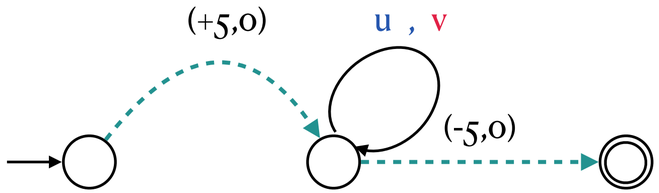


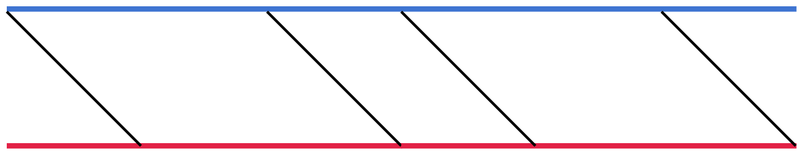


closeness is decidable

For Hamming and transposition

- Output lengths must be equal for all words.
  - Loops generate words of same length
  - Delay between partial outputs is bounded, and depends only on the state.
- Check if all pairs in a loop (modulo the delay on border) are equal (up to some length depending on the delay).





Closeness of transducers

- Let d be a metric on words. Lift it to transducers.

Distance is computable iff closeness and k-closeness are decidable. (for integer-valued metrics)

- Given T_1, T_2 is $d(T_1, T_2)$ computable? (Distance)
- Given T_1, T_2 is $d(T_1, T_2)$ finite? (Closeness)
- Given T_1, T_2 and $k \in \mathbb{N}$, is $d(T_1, T_2)$ at most k? (k-closeness)

Closeness is decidable

For Levenshtein

Closeness is decidable

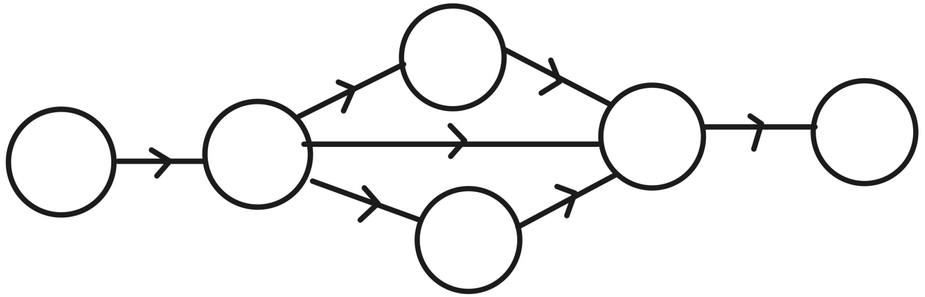
For Levenshtein

- Consider the cartesian product

Closeness is decidable

For Levenshtein

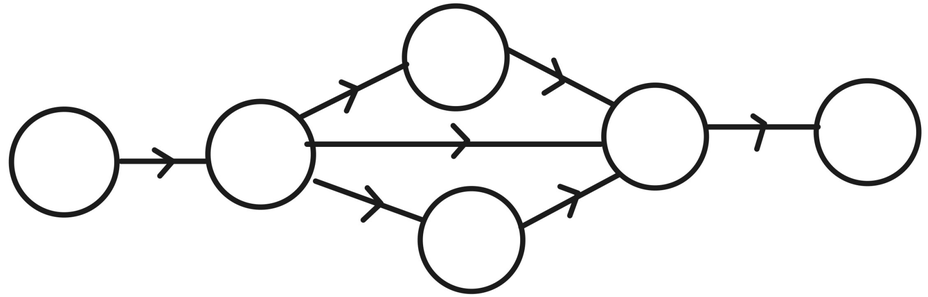
- Consider the cartesian product
- If it is a DAG -> bounded

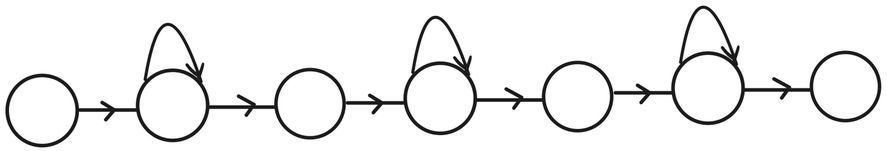


Closeness is decidable

For Levenshtein

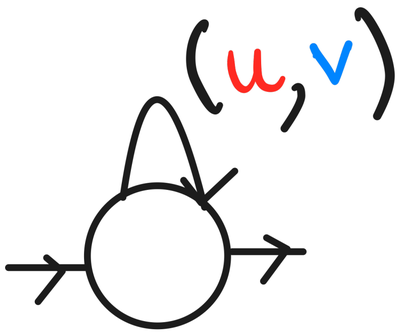
- Consider the cartesian product
- If it is a DAG -> bounded
- If it has simple cycles



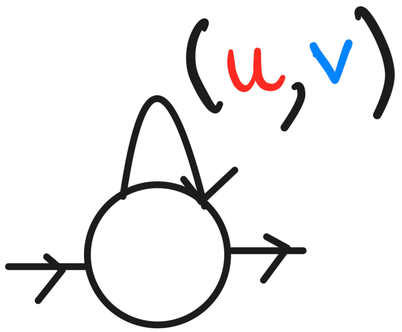


if $u = xy$ and $v = yx$

if $u = xy$ and $v = yx$



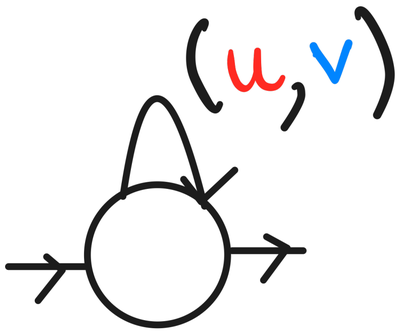
if $u = xy$ and $v = yx$



$$x\,y\,x\,y\,\cdots\,x\,y\,x\,y$$

$$y\,x\,y\,x\,\cdots\,y\,x\,y\,x$$

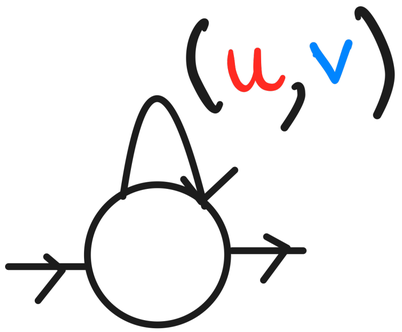
if $u = xy$ and $v = yx$

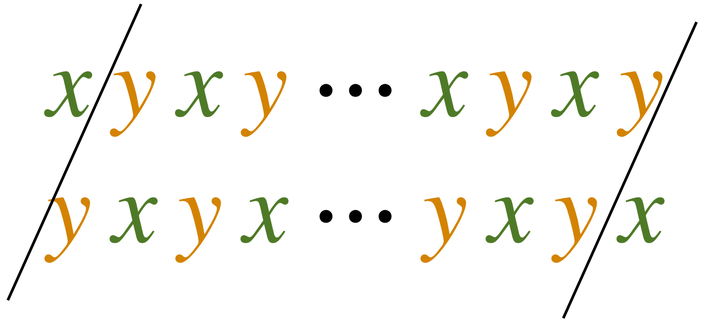


$$x\,y\,x\,y\cdots x\,y\,x\,y$$
$$y\,x\,y\,x\cdots y\,x\,y\,x$$

Conjugates

- u and v are conjugates if $u = xy$ and $v = yx$

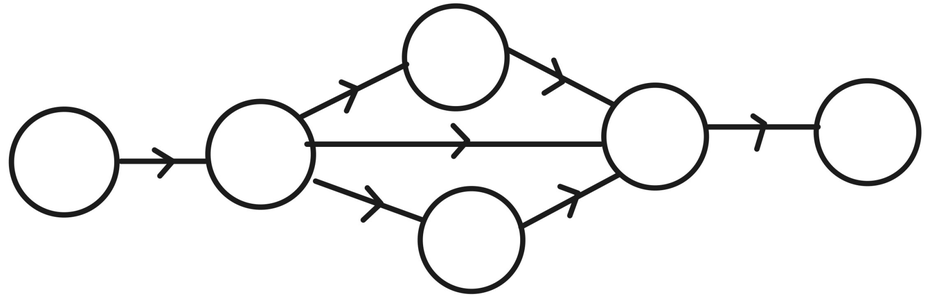


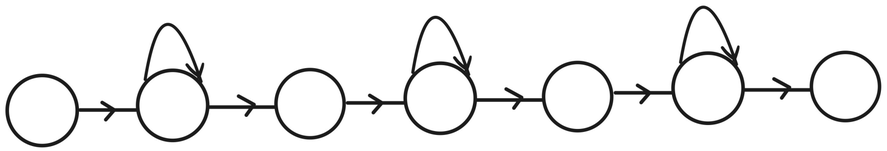


Closeness is decidable

For Levenstein

- Consider the cartesian product
- If it is a DAG -> bounded
- If it has simple cycles
- Check if each of them is conjugate
- If there are multiple cycle?

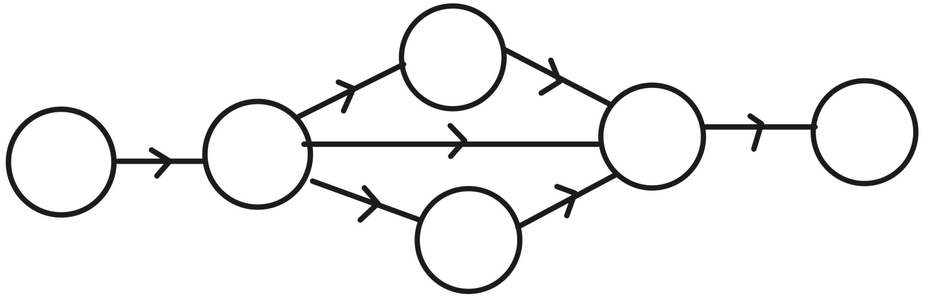


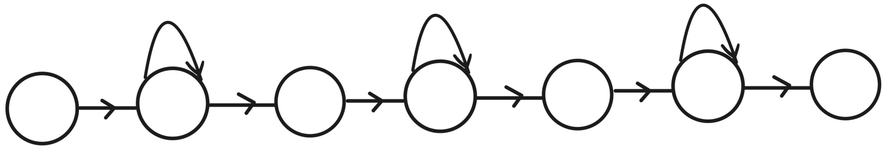


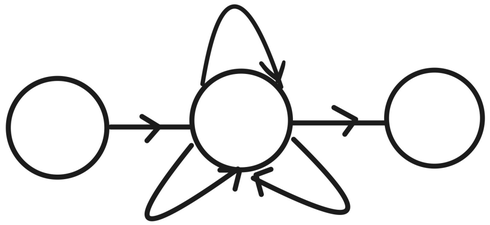
Closeness is decidable

For Levenstein

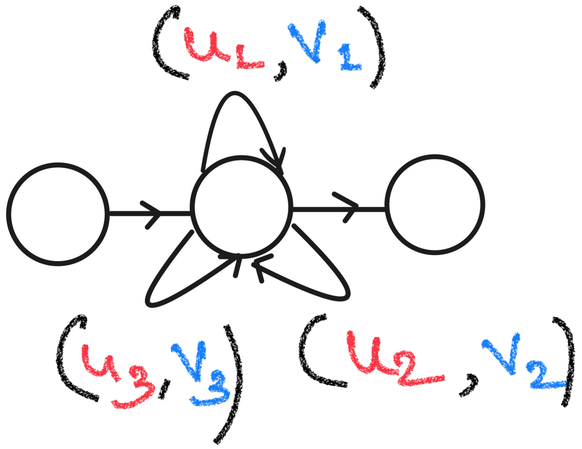
- Consider the cartesian product
- If it is a DAG -> bounded
- If it has simple cycles
- Check if each of them is conjugate
- If there are multiple cycle?





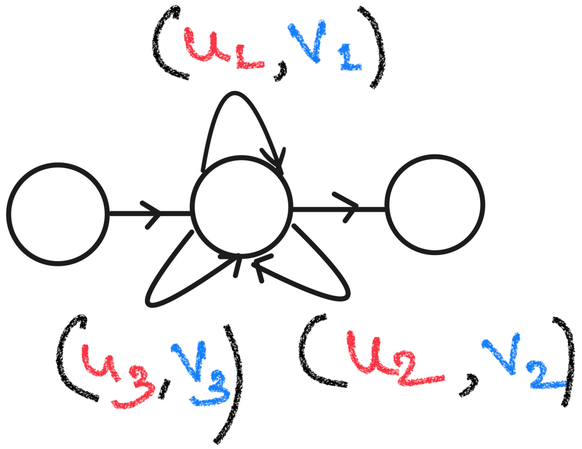


More on conjugates

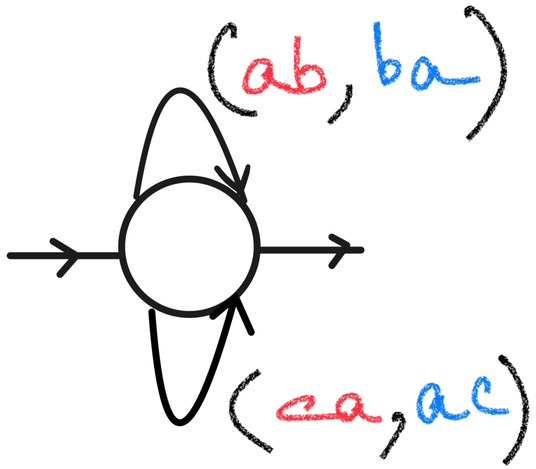


where each u_i and v_i are conjugates

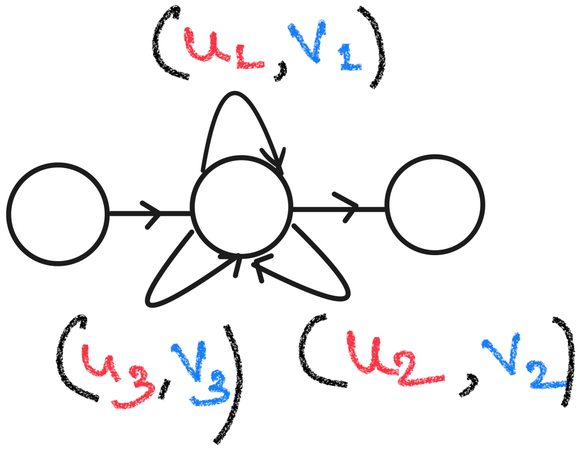
More on conjugates



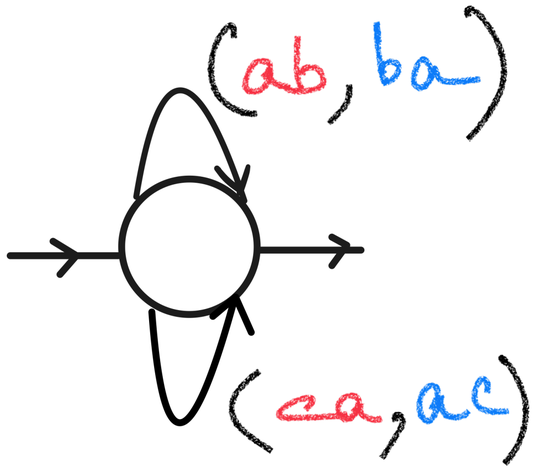
where each u_i and v_i are conjugates



More on conjugates

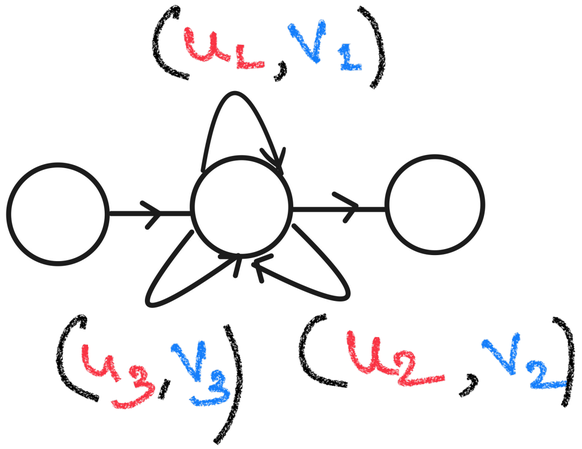


where each u_i and v_i are conjugates



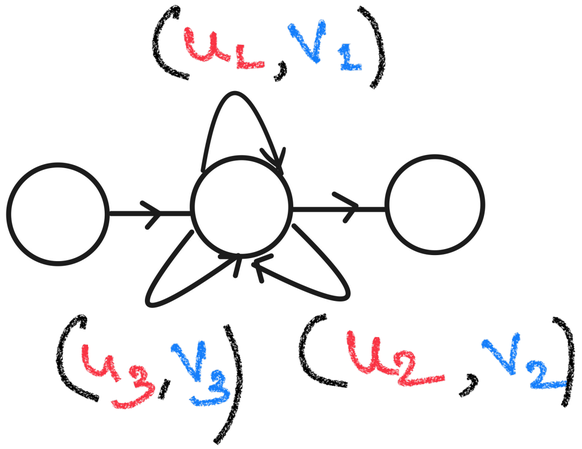
$(abca, baac)$

More on conjugates



where each u_i and v_i are conjugates

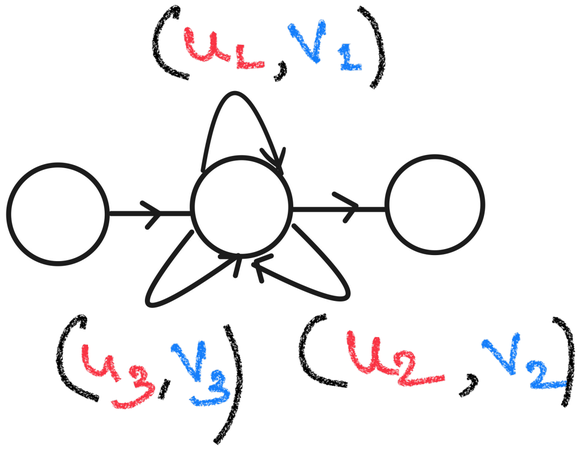
More on conjugates



where each u_i and v_i are conjugates

- $G = \{(u_i, v_i) \mid i \in I\}$

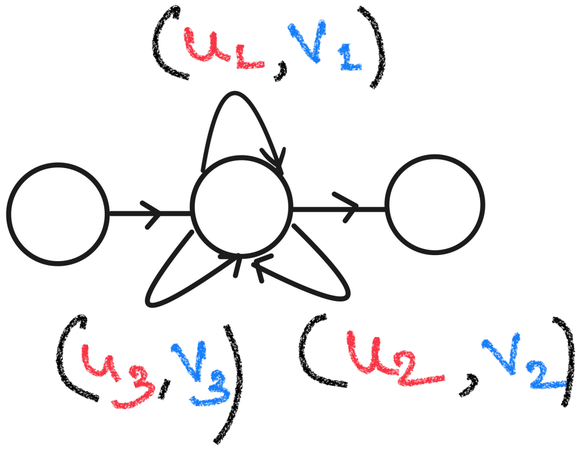
More on conjugates



where each u_i and v_i are conjugates

- $G = \{(u_i, v_i) \mid i \in I\}$
- Is $\langle G \rangle$ conjugate?

More on conjugates



where each u_i and v_i are conjugates

- $G = \{(u_i, v_i) \mid i \in I\}$
- Is $\langle G \rangle$ conjugate?
- $\langle G \rangle = \{(u_{i_1} u_{i_2} \cdots u_{i_n}, v_{i_1} v_{i_2} \cdots v_{i_n}) \mid n \geq 1, i_j \in I\}$

Conjugates

- Theorem (Lyndon Schutzenberger)

$uz = zv$ **iff** $u = xy$ **and** $v = yx$ **and** $z \in (xy)^*x$

Conjugates

- Theorem (Lyndon Schutzenberger)

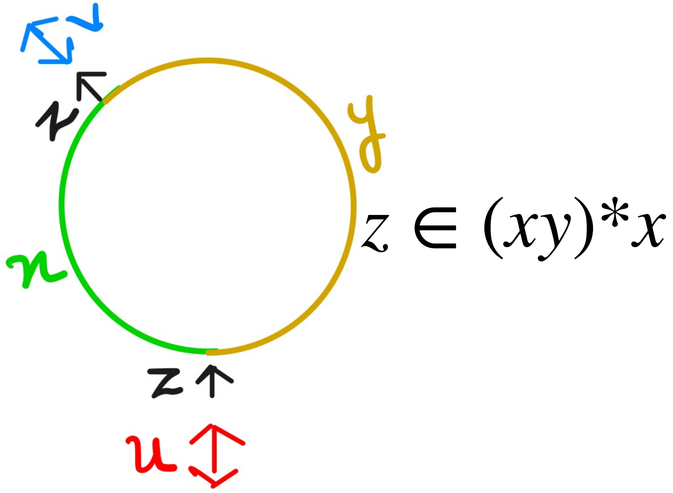
$uz = zv$ **iff** $u = xy$ **and** $v = yx$ **and** $z \in (xy)^*x$

$z'u = vz'$ **iff** $u = xy$ **and** $v = yx$ **and** $z' \in (yx)^*y$

Conjugates

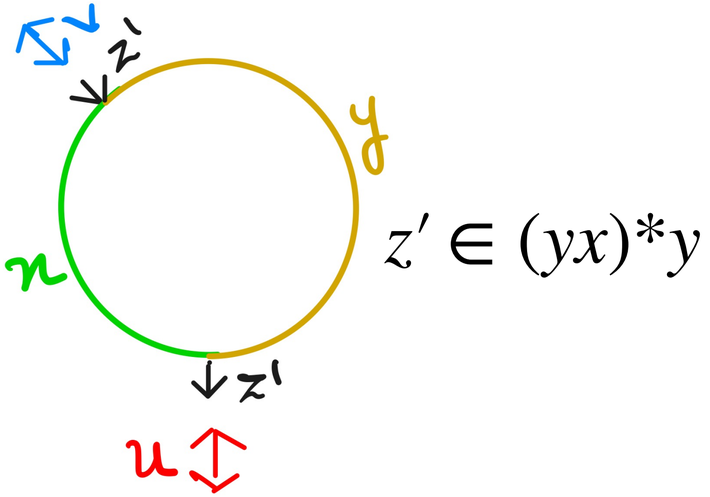
- Theorem (Lyndon Schutzenberger)

$$uz = zv \quad \textbf{iff} \quad u = xy \textbf{ and } v = yx \textbf{ and } z \in (xy)^*x$$
$$z'u = vz' \quad \textbf{iff} \quad u = xy \textbf{ and } v = yx \textbf{ and } z \in y(xy)^*$$



$z \in (xy)^*x$

z is an inner-witness of $(u\ ,\ v\ )$



$z' \in (yx)^*y$

z' is an outer-witness of $(u\ ,\ v\ )$

Conjugates

- Common Witness Theorem

$\langle G \rangle$ **is conjugate iff** $\exists z : \; z \in \cap \, (x_i y_i)^* x_i$

Conjugates

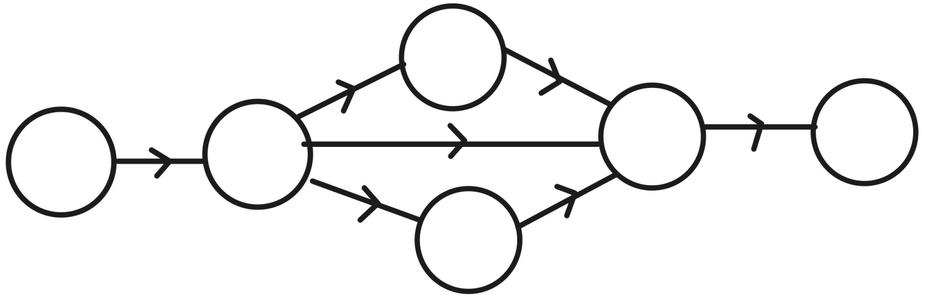
- Common Witness Theorem

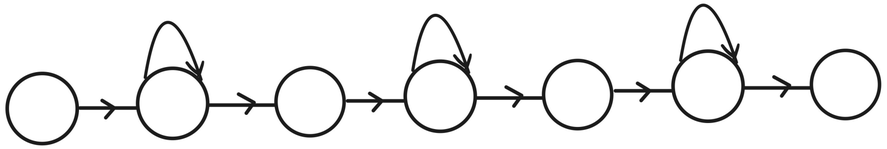
$\langle G \rangle$ **is conjugate iff** $\exists z : \; z \in \cap\,(x_i y_i)^* x_i$ **OR** $z \in \cap\,(y_i x_i)^* y_i$

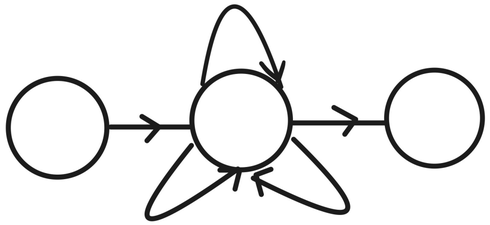
Closeness is decidable

For Levenstein

- Consider the cartesian product
- If it is a DAG -> bounded
- If it has simple cycles
- Check if each of them is conjugate
- If there are multiple cycle?







Closeness is decidable

For Levenstein

Closeness is decidable

For Levenstein

- Get a rational expression from the cartesian product

Closeness is decidable

For Levenstein

- Get a rational expression from the cartesian product
- Express it as a sum of sum-free expressions

Closeness is decidable

For Levenstein

- Get a rational expression from the cartesian product
- Express it as a sum of sum-free expressions
- Structural induction. For star, do a Check conjugacy of $\langle G \rangle$.

Closeness is decidable

For Levenshtein

- Two transducers are close w.r.t. Levenshtein edit distance iff all the cycles in their Cartesian product are conjugates.

Closeness is decidable

For Levenshtein

- Two transducers are close w.r.t. Levenshtein edit distance iff all the cycles in their Cartesian product are conjugates.

Closeness w.r.t. Levenshtein edit distance reduces to deciding conjugacy of rational relations.

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.

Distance is computable iff closeness and k-closeness are decidable. (for integer-valued metrics)

- Given T_1, T_2 is $d(T_1, T_2)$ computable? (Distance)
- Given T_1, T_2 is $d(T_1, T_2)$ finite? (Closeness)
- Given T_1, T_2 and $k \in \mathbb{N}$, is $d(T_1, T_2)$ at most k? (k-closeness)

Closeness of transducers

- Let d be a metric on words. Lift it to transducers.

Distance is computable iff closeness and k-closeness are decidable. (for integer-valued metrics)

- Given T_1, T_2 is $d(T_1, T_2)$ computable? (Distance)
- Given T_1, T_2 is $d(T_1, T_2)$ finite? (Closeness)
- Given T_1, T_2 and $k \in \mathbb{N}$, is $d(T_1, T_2)$ at most k? (k-closeness)

} **Decidable**

Thank you