Generic ε-Removal Algorithm for Weighted Transducers

Mehryar Mohri

報告人:陳柏含

指導教授:陳嘉平

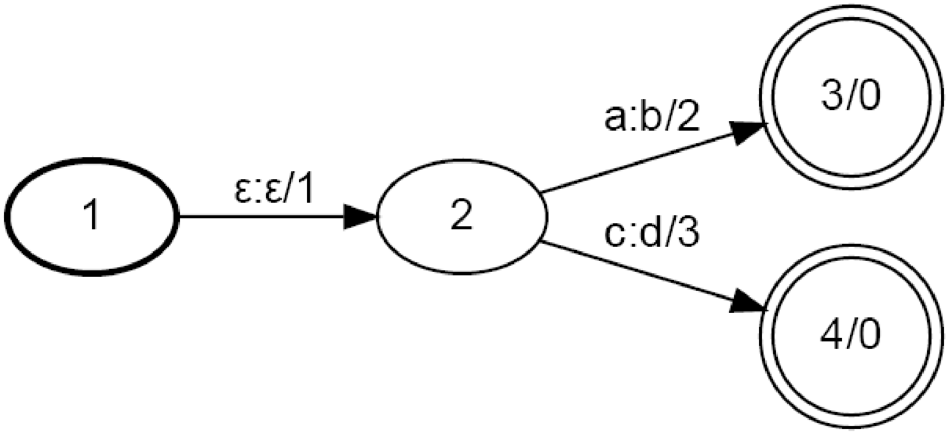
Introduction

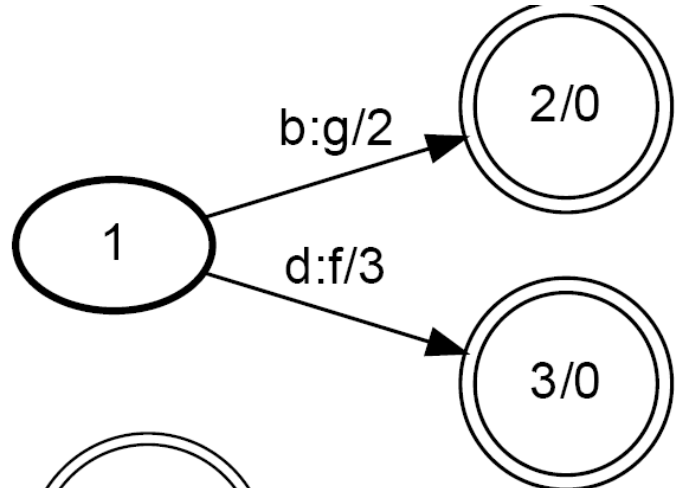
- It present a new generic ε-removal algorithm for weighted automata and transducers.

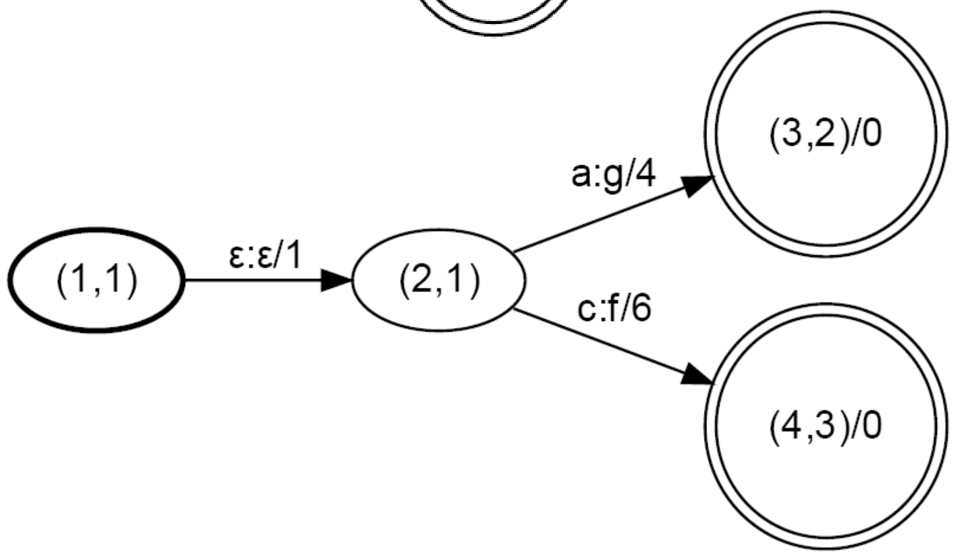
- It is preferable to remove empty strings in general they introduce a delay in using composition algorithm or other algorithms.

- This algorithm is often mixed with other optimization algorithm such as determinization.

Composition







Definition of WFSA

- Weighted finite-state automaton $A=(\Sigma,Q, I, F, E, \lambda, \rho)$ over the semiring $\mathbb{K}$

  - Σ : input alphabet

  - Q : a finite set of states

  - I : a set of initial state, $I \subseteq Q$

  - F: a set of final state, $F \subseteq Q$

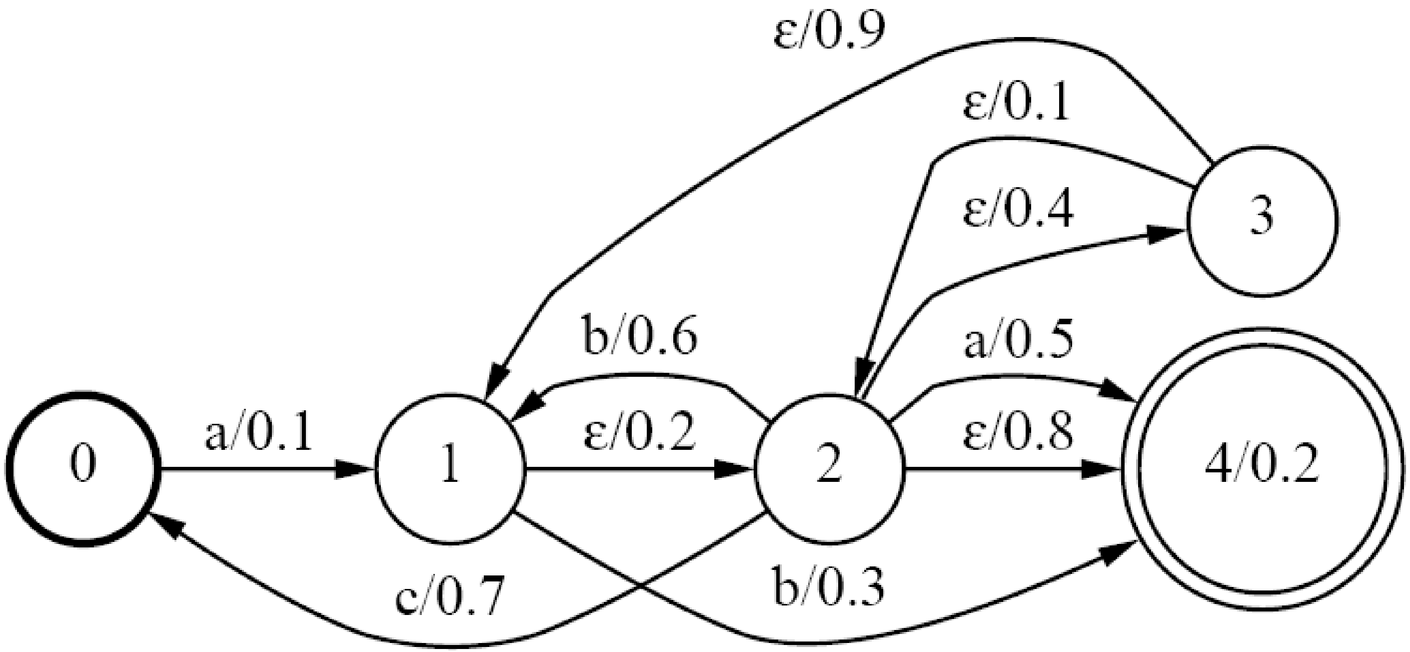
  - E: a finite set of transitions, $E \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times \mathbb{K} \times Q$

  - λ : initial state weight function $\lambda : I \to \mathbb{K}$

  - ρ : final state weight function $\rho : F \to \mathbb{K}$

- $A=(\Sigma, Q, I, F, E, \lambda, \rho)$ is a weighted automaton over semiring $\mathbb{K}$ with ε-transitions.
- A transition of a transducer is an ε-transition when both input and output label are ε.

ε-Transitions Automaton



Preliminary

- For p, q in Q, the ε-distance from p to q in the automaton A is denoted by $d[p,q]$:

$$d[p,q] = \bigoplus_{\pi \in P(p,q),\ i[\pi]=\varepsilon} w[\pi]$$

  - $P(p,q)$: set of paths from p to q
  - π: path $\pi = e_1 e_2 \ldots e_k,\ e_i \in E,\ i = 1 \ldots k$
  - $w[\pi]$: weight of path π

$$w[\pi] = \bigotimes_{i=1}^{k} w[e_i]$$

  - $i[e_i]$: input label of e_i
  - $i[\pi] = i[e_1] \cdots i[e_k]$

ε-Removal Algorithm

- The algorithm works on two steps.

- The first step consists of computing for state *p* of the input automaton *A* its ε–closure denoted by *C*[p]

$$C[p] = \{(q,w) : q \in \varepsilon[p],\ d[p,q] = w \in \mathbb{K} - \{\bar{0}\}\}$$

  - $\varepsilon[p]$: the set of states reachable from *p* via a path labeled with ε

ε-Removal Algorithm(cont.)

- The second step consists of modifying the outgoing transitions of each state p :
  - remove the outgoing transitions labeled with ε
  - add to $E[p]$ non-ε-transitions leaving each state q which belongs to $\varepsilon[p]$
    - $E[p]$: the set of transitions leaving p

Pseudocode of the Second Step

ϵ-removal(A)

for each $p \in Q$

do $E[p] \leftarrow \{e \in E[p] : i[e] \neq \epsilon\}$

for each $(q, w) \in C[p]$

do $E[p] \leftarrow E[p] \cup \{(p, a, w \otimes w', r) : (q, a, w', r) \in E[q], a \neq \epsilon\}$

if $q \in F$

then **if** $p \notin F$

then $F \leftarrow F \cup \{p\}$

$\rho[p] \leftarrow \rho[p] \oplus (w \otimes \rho[q])$

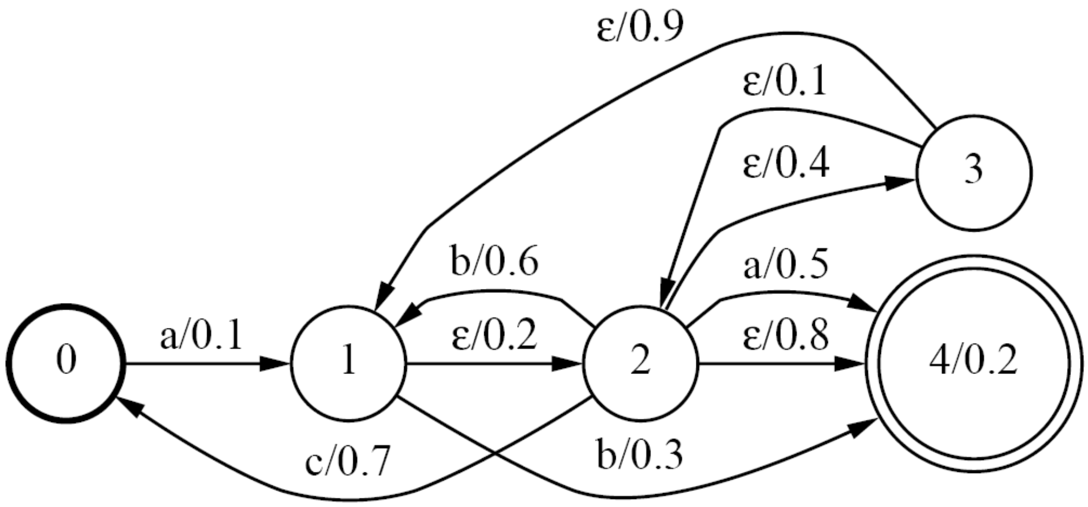
State p is a final state if some state $q \in \epsilon[p]$ is final and the final weight $\rho[p]$ is then:

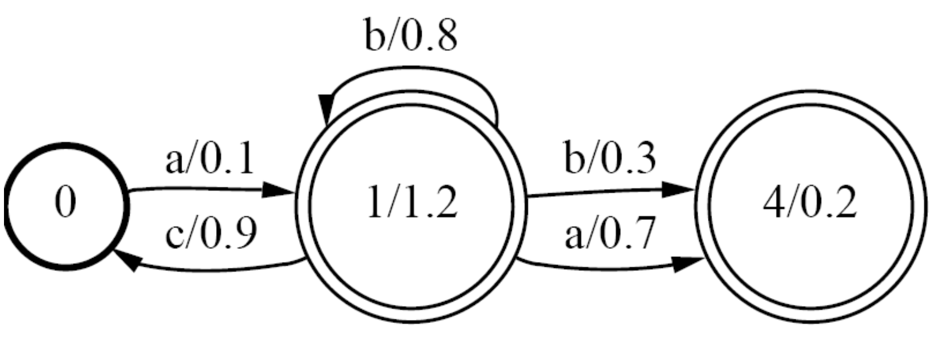
$$\rho[p] = \bigoplus_{q \in \epsilon[p] \cap F} d[p, q] \otimes \rho[q]$$

Post-processing

- After removing ε's at each state p, some states may become inaccessible if they could be reached by ε-transitions originally.
  - those states can be removed by depth-first search of the automaton

Example of ε-Removal Algorithm





Analysis of Resulting Automaton

- The size of resulting automaton is affected by two factors:
  - the number of states in the original automaton whose incoming transitions are all labeled with the empty string
  - the total number of non-ε-transitions of the states that can be reached from each state q