Transducteurs finis à états pondérés

M1 INFO

Marie Tahon
MCF, dpt. Informatique

janvier 2018

PLAN DE LA SECTION ACTUELLE

1. Introduction
2. Les transducteurs finis à états
3. Relation reconnaissable et transducteurs
4. Algorithmes importants pour les WFSR

Introduction

Les machines à états finis sont généralement classées en deux catégories:

- Les accepteurs (ou reconnaisseurs) analyse une chaîne de symboles donnée en entrée, et l'accepte si elle est conforme à la spécification décrite par l'automate.
  - Propriétés intéressantes: exécution rapide d'un AFD, modularité et traitement des exceptions aisé grâce aux opérations ensemblistes, à la concaténation et la fermeture de Kleene.
  - Puissance limitée: mémoire limitée, certaines chaînes de symboles ne peuvent être décrites par un automate.
- Les transducteurs traduisent une chaîne de symboles en une autre, suivant l'algorithme codé dans l'automate.
  - Perte de certaines propriétés des automates (ex: exécution rapide), héritage de certaines opérations seulement.
  - Puissance plus importante: mémoire limitée, plus grande expressivité

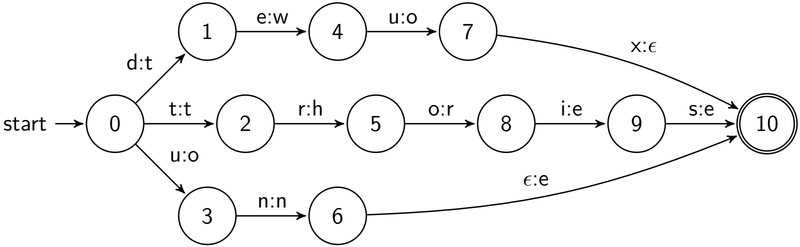
PLAN DE LA SECTION ACTUELLE

Introduction

Un transducteur est un dispositif recevant un message sous une certaine forme et le transformant en une autre.

C'est un automate fini qui non seulement reconnaît un ensemble régulier de chaîne mais encore la traduit dans une autre chaîne appartenant à un autre langage régulier.

C'est un graphe orienté étiqueté où chaque transition est étiquetée par un (ou aucun) symbole à reconnaître et un (ou aucun) symbole utilisé pour la traduction.



Transducteur fini à états: définitions

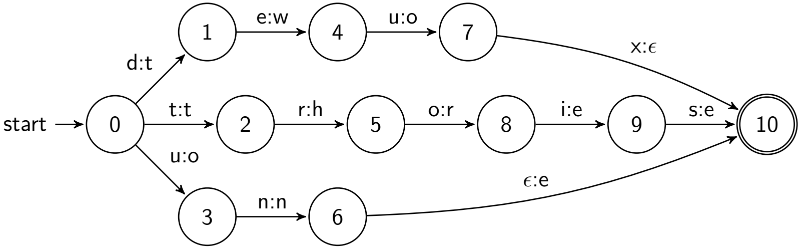
Un transducteur fini, noté FST (Finite State Transducer) est défini par un 6-uplet $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta)$, où:

- Σ_i représente l'alphabet d'entrée
- Σ_o représente l'alphabet de sortie.
- Q est un ensemble fini d'états ($Q \neq \emptyset$)
- $I \subseteq Q$ est l'ensemble des états initiaux (généralement $I = \{q_0\}$)
- $F \subseteq Q$ est l'ensemble des états finaux (éventuellement $F = \emptyset$)
- δ est une relation de $Q \times (\Sigma_i \cup \{\epsilon\}) \times (\Sigma_o \cup \{\epsilon\}) \rightarrow Q$ appelée fonction de transition.

Si $\delta : Q \times (\Sigma_i \cup \{\epsilon\}) \times (\Sigma_o \cup \{\epsilon\}) \times \mathbb{K} \rightarrow Q$, alors T est non déterministe
Si $\delta : Q \times \Sigma_i \times \Sigma_o \times \mathbb{K} \rightarrow Q$, alors T est déterministe

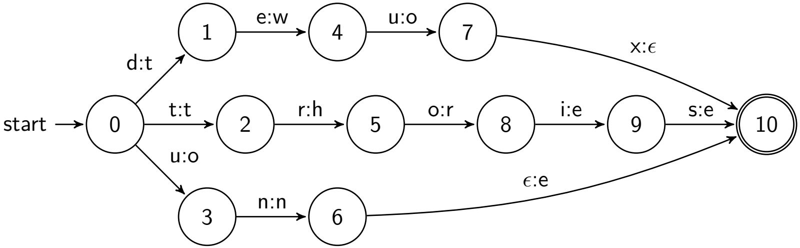
Transducteur fini à états: exemple

- Σ_i
- Σ_o
- Q
- I
- F
- δ



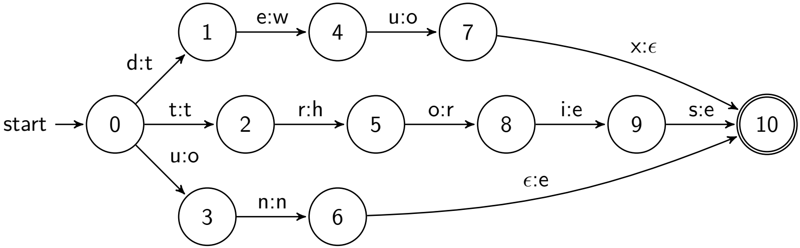
Transducteur fini à états: exemple

- $\Sigma_i = \{d, e, u, x, t, r, o, i, s, n\}$
- Σ_o
- Q
- I
- F
- δ



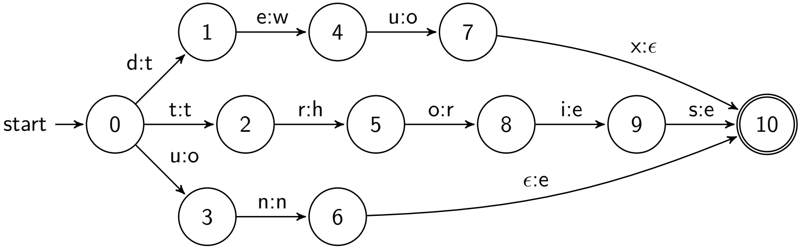
Transducteur fini à états: exemple

- $\Sigma_i = \{d, e, u, x, t, r, o, i, s, n\}$
- $\Sigma_o = \{t, w, o, h, r, e, n\}$
- Q
- I
- F
- δ



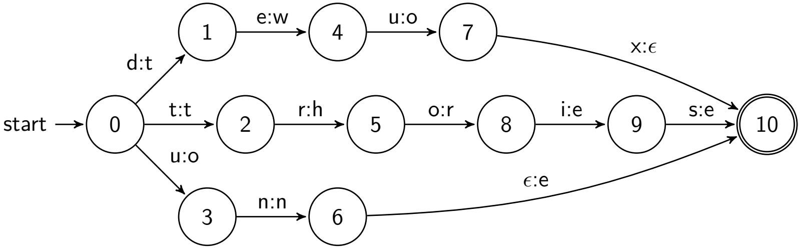
Transducteur fini à états: exemple

- $\Sigma_i = \{d, e, u, x, t, r, o, i, s, n\}$
- $\Sigma_o = \{t, w, o, h, r, e, n\}$
- $Q = \{q_0, q_1, \ldots q_{10}\}$
- I
- F
- δ



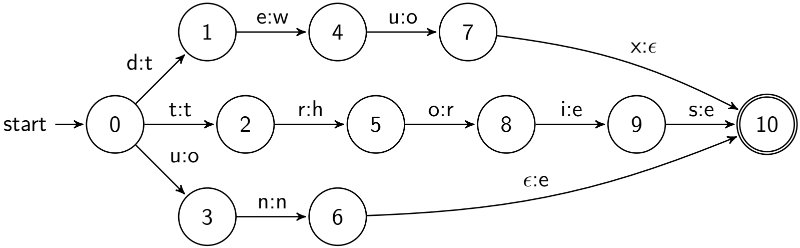
Transducteur fini à états: exemple

- $\Sigma_i = \{d, e, u, x, t, r, o, i, s, n\}$
- $\Sigma_o = \{t, w, o, h, r, e, n\}$
- $Q = \{q_0, q_1, \ldots q_{10}\}$
- $I = \{q_0\}$
- F
- δ



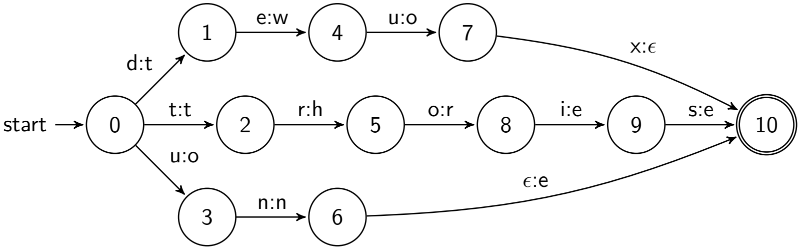
Transducteur fini à états: exemple

- $\Sigma_i = \{d, e, u, x, t, r, o, i, s, n\}$
- $\Sigma_o = \{t, w, o, h, r, e, n\}$
- $Q = \{q_0, q_1, \ldots q_{10}\}$
- $I = \{q_0\}$
- $F = \{q_{10}\}$
- δ



Transducteur fini à états: exemple

- $\Sigma_i = \{d, e, u, x, t, r, o, i, s, n\}$
- $\Sigma_o = \{t, w, o, h, r, e, n\}$
- $Q = \{q_0, q_1, \dots q_{10}\}$
- $I = \{q_0\}$
- $F = \{q_{10}\}$
- $\delta = \{(0, d : t, 1), (1, e : w, 4), (4, u : o, 7), (7, x : \epsilon, 10), \dots\}$



Transducteur fini à états pondérés: définition

De manière générale, on peut définir les transducteurs finis à états pondérés que l'on notera WFST (Weighted Finite State Transducer).

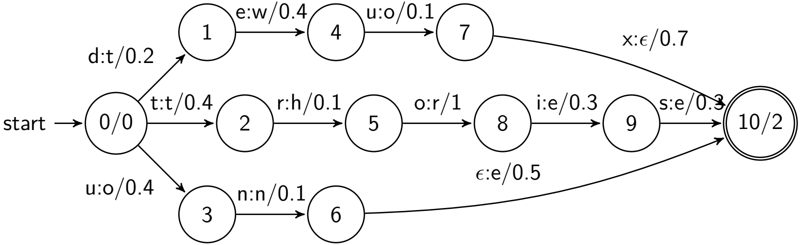
Un WFST est défini par un 8-uplet $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$, où:

- Σ_i représente l'alphabet d'entrée
- Σ_o représente l'alphabet de sortie.
- Q est un ensemble fini d'états ($Q \neq \emptyset$)
- $I \subseteq Q$ est l'ensemble des états initiaux (généralement $I = \{q_0\}$)
- $F \subseteq Q$ est l'ensemble des états finaux (éventuellement $F = \emptyset$)
- δ est une relation de $Q \times (\Sigma_i \cup \{\epsilon\}) \times (\Sigma_o \cup \{\epsilon\}) \times \mathbb{K} \to Q$ appelée fonction de transition.
- $\lambda : I \to \mathbb{K}$ est la fonction de pondération initiale
- $\rho : F \to \mathbb{K}$ est la fonction de pondération finale

Introduction: exemple

Un WFST est défini par un 8-uplet $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$, où:

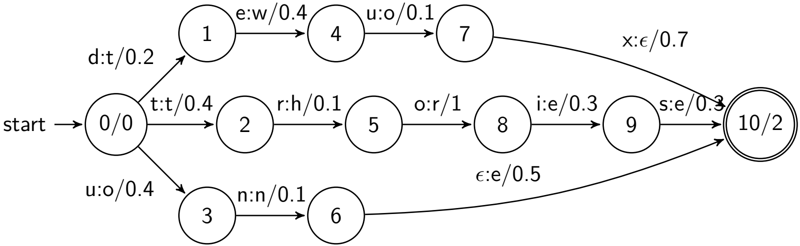
- $\delta : Q \times (\Sigma_i \cup \{\epsilon\}) \times (\Sigma_o \cup \{\epsilon\}) \times \mathbb{K} \to Q$
- $\lambda : I \to \mathbb{K}$
- $\rho : F \to \mathbb{K}$



Introduction: exemple

Un WFST est défini par un 8-uplet $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$, où:

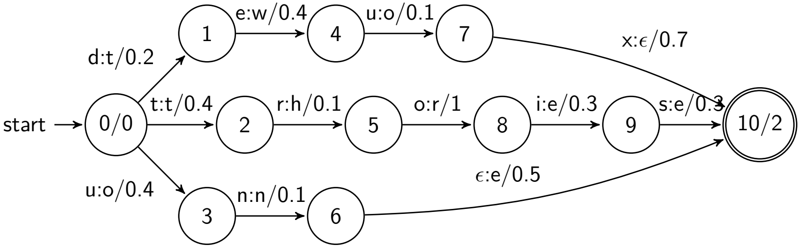
- $\delta : Q \times (\Sigma_i \cup \{\epsilon\}) \times (\Sigma_o \cup \{\epsilon\}) \times \mathbb{K} \rightarrow Q$
  $\delta = (0, d : t, 0.2, 1), (1, e : w, 0.4, 4), \ldots$
- $\lambda : I \rightarrow \mathbb{K}$
- $\rho : F \rightarrow \mathbb{K}$



Introduction: exemple

Un WFST est défini par un 8-uplet $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$, où:

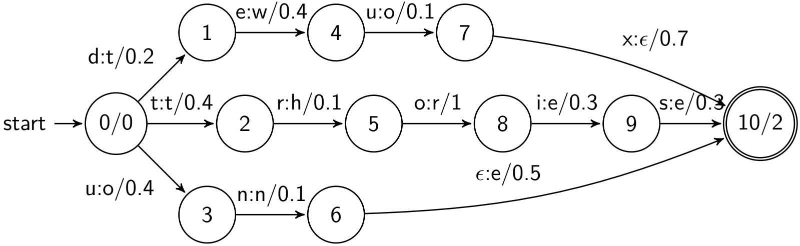
- $\delta : Q \times (\Sigma_i \cup \{\epsilon\}) \times (\Sigma_o \cup \{\epsilon\}) \times \mathbb{K} \rightarrow Q$
  $\delta = (0, d : t, 0.2, 1), (1, e : w, 0.4, 4), \ldots$
- $\lambda : I \rightarrow \mathbb{K}$ Ici $\lambda(q_0) = 0$
- $\rho : F \rightarrow \mathbb{K}$



Introduction: exemple

Un WFST est défini par un 8-uplet $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$, où:

- $\delta : Q \times (\Sigma_i \cup \{\epsilon\}) \times (\Sigma_o \cup \{\epsilon\}) \times \mathbb{K} \to Q$
  $\delta = (0, d : t, 0.2, 1), (1, e : w, 0.4, 4), \ldots$
- $\lambda : I \to \mathbb{K}$ Ici $\lambda(q_0) = 0$
- $\rho : F \to \mathbb{K}$ Ici $\rho(q_{10}) = 2$



Domaine de définition des pondérations

On vient de voir que les fonctions de pondération initiale λ et finale ρ, ainsi que la fonction de transition δ, associent à tout état appartenant à leur ensemble de définition (I ou F), une pondération dans $\mathbb{K}$.

Mathématiquement $\mathbb{K}$ est un demi-anneau défini par le système $(\oplus, \otimes, \overline{0}, \overline{1})$ tel que:

- $(\mathbb{K}, \oplus, \overline{0})$ est un monoïde commutatif:
  - $\overline{0}$ est l'élément neutre de l'opérateur $\oplus$: $\forall x \in \mathbb{K}, x \oplus \overline{0} = x$
  - $\oplus$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \oplus y) \oplus z = x \oplus (y \oplus z)$
  - $\oplus$ est commutatif: $\forall x, y \in \mathbb{K}, x \oplus y = y \oplus x$
- $(\mathbb{K}, \otimes, \overline{1})$ est un monoïde:
  - $\overline{1}$ est l'élément neutre de l'opérateur $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{1} = x$
  - $\otimes$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \otimes y) \otimes z = x \otimes (y \otimes z)$
- $\otimes$ est distributif par rapport à $\oplus$: $\forall x, y, z \in \mathbb{K}, x \otimes (y \oplus z) = (x \otimes y) \oplus (s \otimes z)$
- $\overline{0}$ est absorbant pour $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{0} = \overline{0}$

Domaine de définition des pondérations

On vient de voir que les fonctions de pondération initiale λ et finale ρ, ainsi que la fonction de transition δ, associent à tout état appartenant à leur ensemble de définition (I ou F), une pondération dans $\mathbb{K}$.

Mathématiquement $\mathbb{K}$ est un demi-anneau défini par le système $(\oplus, \otimes, \overline{0}, \overline{1})$ tel que:

- $(\mathbb{K}, \oplus, \overline{0})$ est un monoïde commutatif:
  - $\overline{0}$ est l'élément neutre de l'opérateur $\oplus$: $\forall x \in \mathbb{K}, x \oplus \overline{0} = x$
  - $\oplus$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \oplus y) \oplus z = x \oplus (y \oplus z)$
  - $\oplus$ est commutatif: $\forall x, y \in \mathbb{K}, x \oplus y = y \oplus x$
- $(\mathbb{K}, \otimes, \overline{1})$ est un monoïde:
  - $\overline{1}$ est l'élément neutre de l'opérateur $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{1} = x$
  - $\otimes$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \otimes y) \otimes z = x \otimes (y \otimes z)$
- $\otimes$ est distributif par rapport à $\oplus$: $\forall x, y, z \in \mathbb{K}, x \otimes (y \oplus z) = (x \otimes y) \oplus (s \otimes z)$
- $\overline{0}$ est absorbant pour $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{0} = \overline{0}$

Domaine de définition des pondérations

On vient de voir que les fonctions de pondération initiale λ et finale ρ, ainsi que la fonction de transition δ, associent à tout état appartenant à leur ensemble de définition (I ou F), une pondération dans $\mathbb{K}$.

Mathématiquement $\mathbb{K}$ est un demi-anneau défini par le système $(\oplus, \otimes, \overline{0}, \overline{1})$ tel que:

- $(\mathbb{K}, \oplus, \overline{0})$ est un monoïde commutatif:
  - $\overline{0}$ est l'élément neutre de l'opérateur $\oplus$: $\forall x \in \mathbb{K}, x \oplus \overline{0} = x$
  - $\oplus$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \oplus y) \oplus z = x \oplus (y \oplus z)$
  - $\oplus$ est commutatif: $\forall x, y \in \mathbb{K}, x \oplus y = y \oplus x$
- $(\mathbb{K}, \otimes, \overline{1})$ est un monoïde:
  - $\overline{1}$ est l'élément neutre de l'opérateur $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{1} = x$
  - $\otimes$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \otimes y) \otimes z = x \otimes (y \otimes z)$
- $\otimes$ est distributif par rapport à $\oplus$: $\forall x, y, z \in \mathbb{K}, x \otimes (y \oplus z) = (x \otimes y) \oplus (s \otimes z)$
- $\overline{0}$ est absorbant pour $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{0} = \overline{0}$

Domaine de définition des pondérations

On vient de voir que les fonctions de pondération initiale λ et finale ρ, ainsi que la fonction de transition δ, associent à tout état appartenant à leur ensemble de définition (I ou F), une pondération dans $\mathbb{K}$.

Mathématiquement $\mathbb{K}$ est un demi-anneau défini par le système $(\oplus, \otimes, \overline{0}, \overline{1})$ tel que:

- $(\mathbb{K}, \oplus, \overline{0})$ est un monoïde commutatif:
  - $\overline{0}$ est l'élément neutre de l'opérateur $\oplus$: $\forall x \in \mathbb{K}, x \oplus \overline{0} = x$
  - $\oplus$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \oplus y) \oplus z = x \oplus (y \oplus z)$
  - $\oplus$ est commutatif: $\forall x, y \in \mathbb{K}, x \oplus y = y \oplus x$
- $(\mathbb{K}, \otimes, \overline{1})$ est un monoïde:
  - $\overline{1}$ est l'élément neutre de l'opérateur $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{1} = x$
  - $\otimes$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \otimes y) \otimes z = x \otimes (y \otimes z)$
- $\otimes$ est distributif par rapport à $\oplus$: $\forall x, y, z \in \mathbb{K}, x \otimes (y \oplus z) = (x \otimes y) \oplus (s \otimes z)$
- $\overline{0}$ est absorbant pour $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{0} = \overline{0}$

Domaine de définition des pondérations

On vient de voir que les fonctions de pondération initiale λ et finale ρ, ainsi que la fonction de transition δ, associent à tout état appartenant à leur ensemble de définition (I ou F), une pondération dans $\mathbb{K}$.

Mathématiquement $\mathbb{K}$ est un demi-anneau défini par le système $(\oplus, \otimes, \overline{0}, \overline{1})$ tel que:

- $(\mathbb{K}, \oplus, \overline{0})$ est un monoïde commutatif:
  - $\overline{0}$ est l'élément neutre de l'opérateur $\oplus$: $\forall x \in \mathbb{K}, x \oplus \overline{0} = x$
  - $\oplus$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \oplus y) \oplus z = x \oplus (y \oplus z)$
  - $\oplus$ est commutatif: $\forall x, y \in \mathbb{K}, x \oplus y = y \oplus x$
- $(\mathbb{K}, \otimes, \overline{1})$ est un monoïde:
  - $\overline{1}$ est l'élément neutre de l'opérateur $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{1} = x$
  - $\otimes$ est associatif: $\forall x, y, z \in \mathbb{K}, (x \otimes y) \otimes z = x \otimes (y \otimes z)$
- $\otimes$ est distributif par rapport à $\oplus$: $\forall x, y, z \in \mathbb{K}, x \otimes (y \oplus z) = (x \otimes y) \oplus (s \otimes z)$
- $\overline{0}$ est absorbant pour $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{0} = \overline{0}$

Domaine de définition des pondérations: exemples

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Booléen	$\{0,1\}$	ET	OU	0	1

Semi-anneaux très largement utilisés pour le traitement automatique de la parole.

- Booléen: avec ce type de semi-anneau, les machines non pondérées et pondérées sont équivalentes.
- Probabilité: les poids représentent des nombres réels entre 0 et 1.
- Log: bonne stabilité numérique, les pondérations correspondent aux $-\log$ probabilités
- Tropical: pratique pour les algorithmes de recherche de plus court chemin, l'approximation est aussi plus rapide (Viterbi)

Domaine de définition des pondérations: exemples

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Booléen	$\{0,1\}$	ET	OU	0	1
Probabilité	$\mathbb{R}_+ \cup \{+\infty\}$	$+$	$\times$	0	1

Semi-anneaux très largement utilisés pour le traitement automatique de la parole.

- Booléen: avec ce type de semi-anneau, les machines non pondérées et pondérées sont équivalentes.
- Probabilité: les poids représentent des nombres réels entre 0 et 1.
- Log: bonne stabilité numérique, les pondérations correspondent aux $-$ log probabilités
- Tropical: pratique pour les algorithmes de recherche de plus court chemin, l'approximation est aussi plus rapide (Viterbi)

Domaine de définition des pondérations: exemples

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Booléen	$\{0,1\}$	ET	OU	0	1
Probabilité	$\mathbb{R}_+ \cup \{+\infty\}$	$+$	$\times$	0	1
Log	$\mathbb{R} \cup \{-\infty, +\infty\}$	$\oplus_{log}$	$+$	$+\infty$	0

Semi-anneaux très largement utilisés pour le traitement automatique de la parole.

- Booléen: avec ce type de semi-anneau, les machines non pondérées et pondérées sont équivalentes.
- Probabilité: les poids représentent des nombres réels entre 0 et 1.
- Log: bonne stabilité numérique, les pondérations correspondent aux $-\log$ probabilités
- Tropical: pratique pour les algorithmes de recherche de plus court chemin, l'approximation est aussi plus rapide (Viterbi)

Domaine de définition des pondérations: exemples

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Booléen	$\{0, 1\}$	ET	OU	0	1
Probabilité	$\mathbb{R}_+ \cup \{+\infty\}$	$+$	$\times$	0	1
Log	$\mathbb{R} \cup \{-\infty, +\infty\}$	$\oplus_{log}$	$+$	$+\infty$	0
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

Semi-anneaux très largement utilisés pour le traitement automatique de la parole.

- Booléen: avec ce type de semi-anneau, les machines non pondérées et pondérées sont équivalentes.
- Probabilité: les poids représentent des nombres réels entre 0 et 1.
- Log: bonne stabilité numérique, les pondérations correspondent aux $-\log$ probabilités
- Tropical: pratique pour les algorithmes de recherche de plus court chemin, l'approximation est aussi plus rapide (Viterbi)

Domaine de définition des pondérations: Log

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Log	$\mathbb{R} \cup \{-\infty, +\infty\}$	$\oplus_{log}$	$+$	$+\infty$	0

$$\forall x, y \in \mathbb{K}, x \oplus_{log} y = -\log\left(e^{-x} + e^{-y}\right)$$

- $(\mathbb{K}, \oplus, \overline{0})$ est un monoïde commutatif:
- $(\mathbb{K}, \otimes, \overline{1})$ est un monoïde
- $\otimes$ est distributif par rapport à $\oplus$:
  $\forall x, y, z \in \mathbb{K}, x \otimes (y \oplus z) = (x \otimes y) \oplus (s \otimes z)$
- $\overline{0}$ est absorbant pour $\otimes$: $\forall x \in \mathbb{K}, x \otimes \overline{0} = \overline{0}$

Domaine de définition des pondérations: exemples

le semi-anneau tropical: simple mais pas du tout intuitif

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

- $5 \oplus 3$
- $(3 \otimes 4) \oplus 6$
- $1 \otimes 5$
- $3 \oplus \overline{1}$
- $3 \otimes \overline{1}$

Domaine de définition des pondérations: exemples

le semi-anneau tropical: simple mais pas du tout intuitif

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

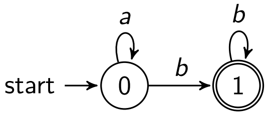
- $5 \oplus 3 = 3$
- $(3 \otimes 4) \oplus 6 = 7 \oplus 6 = 6$
- $1 \otimes 5 = 6$
- $3 \oplus \overline{1} = 3 \oplus 0 = 0$
- $3 \otimes \overline{1} = 3 \otimes 0 = 3$

Cas particuliers des WFST

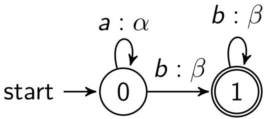
La définition donnée pour les WFST est très générale. En ajoutant des contraintes, on retrouve les définitions suivantes:

- FST non pondéré: $\mathbb{K} = \{1\}$ tous les arcs sont équi-pondérés.
- Automate: $\Sigma_o = \emptyset$ pas d'étiquette de sortie.

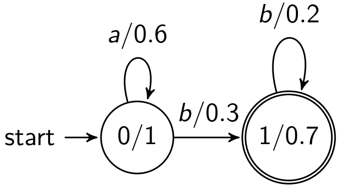
Automate (FSA)



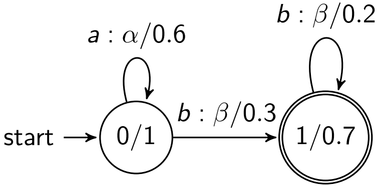
Transducteur (FST)



Automate pondéré (WFSA)

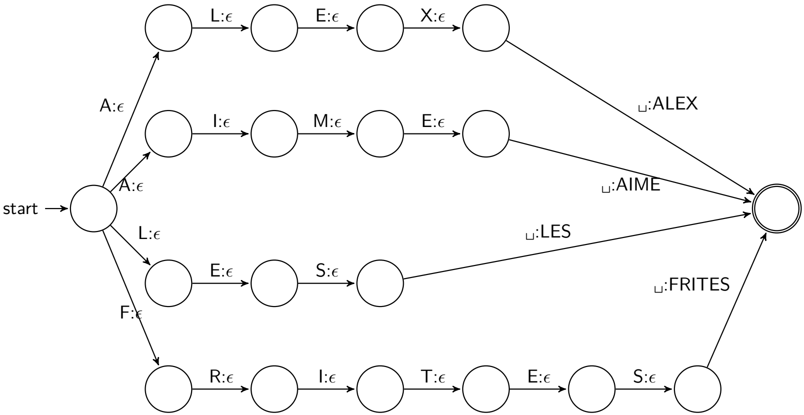


Transducteur pondéré (WFST)



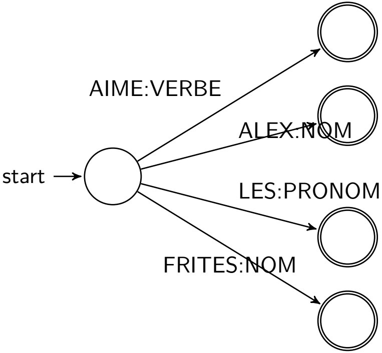
Exemple 1: FST

Chunking de mots



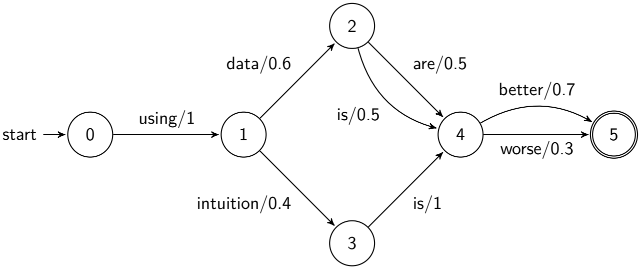
Exemple 2: FST

POS tagger pour la syntaxe grammaticale



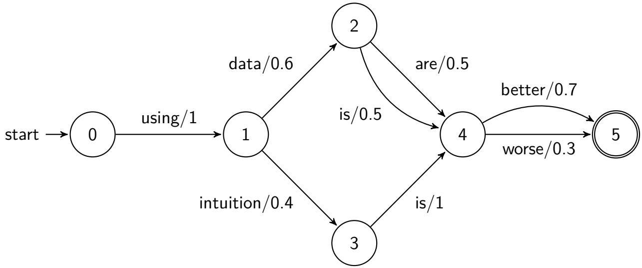
Exemple 3: WFSA

Grammaire (semi-anneau: probabilités réelles)

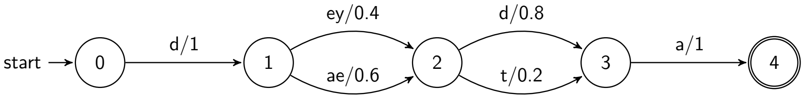


Exemple 3: WFSA

Grammaire (semi-anneau: probabilités réelles)

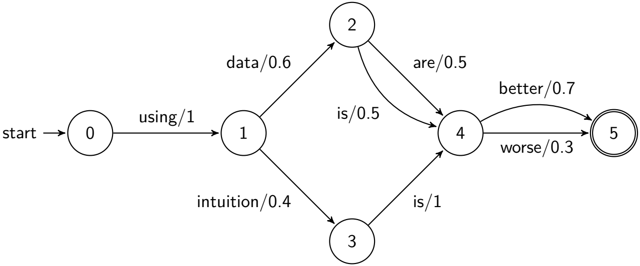


Lexique mot/prononciation “data” (semi-anneau: probabilités réelles)

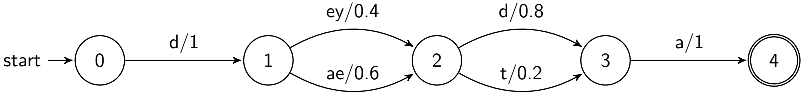


Exemple 3: WFSA

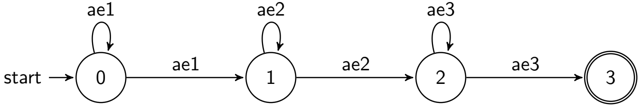
Grammaire (semi-anneau: probabilités réelles)



Lexique mot/prononciation “data” (semi-anneau: probabilités réelles)

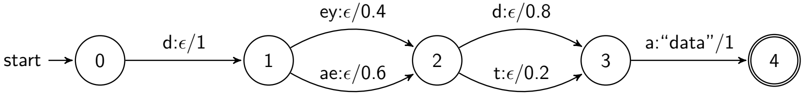


Modèle acoustique de /ae/ (phonème/paramètres acoustiques d'un HMM)



Exemple 4: WFST

Transducteur pour la prononciation d'un mot:



PLAN DE LA SECTION ACTUELLE

Relation reconnaissable

Définitions:

- Une relation est reconnaissable si il existe un transducteur qui la réalise.
  - les mots $u_o \in \Sigma_o^*$ et $u_i \in \Sigma_i^*$ sont en relation si il existe un calcul réussi qui lit u_o et écrit u_i.
- Deux transducteurs sont dits équivalents si ils réalisent la même relation
- Remarques:
  - Un transducteur ne définit pas un langage mais une relation binaire (ensemble de paires de chaînes).
  - La notion de calcul et de calcul réussi est inchangée. Sauf qu'un calcul n'est plus associé à une chaîne mais une paire de chaîne.

Relation reconnaissable

Définitions:

- Une relation est reconnaissable si il existe un transducteur qui la réalise.
  - les mots $u_o \in \Sigma_o^*$ et $u_i \in \Sigma_i^*$ sont en relation si il existe un calcul réussi qui lit u_o et écrit u_i.
- Deux transducteurs sont dits équivalents si ils réalisent la même relation
- Remarques:
  - Un transducteur ne définit pas un langage mais une relation binaire (ensemble de paires de chaînes).
  - La notion de calcul et de calcul réussi est inchangée. Sauf qu'un calcul n'est plus associé à une chaîne mais une paire de chaîne.

Relation reconnaissable

Définitions:

- Une relation est reconnaissable si il existe un transducteur qui la réalise.
  - les mots $u_o \in \Sigma_o^*$ et $u_i \in \Sigma_i^*$ sont en relation si il existe un calcul réussi qui lit u_o et écrit u_i.
- Deux transducteurs sont dits équivalents si ils réalisent la même relation
- Remarques:
  - Un transducteur ne définit pas un langage mais une relation binaire (ensemble de paires de chaînes).
  - La notion de calcul et de calcul réussi est inchangée. Sauf qu'un calcul n'est plus associé à une chaîne mais une paire de chaîne.

Relation reconnaissable

Définitions:

- Une relation est reconnaissable si il existe un transducteur qui la réalise.
  - les mots $u_o \in \Sigma_o^*$ et $u_i \in \Sigma_i^*$ sont en relation si il existe un calcul réussi qui lit u_o et écrit u_i.
- Deux transducteurs sont dits équivalents si ils réalisent la même relation
- Remarques:
  - Un transducteur ne définit pas un langage mais une relation binaire (ensemble de paires de chaînes).
  - La notion de calcul et de calcul réussi est inchangée. Sauf qu'un calcul n'est plus associé à une chaîne mais une paire de chaîne.

Relation reconnaissable

Définitions:

- Une relation est reconnaissable si il existe un transducteur qui la réalise.
  - les mots $u_o \in \Sigma_o^*$ et $u_i \in \Sigma_i^*$ sont en relation si il existe un calcul réussi qui lit u_o et écrit u_i.
- Deux transducteurs sont dits équivalents si ils réalisent la même relation
- Remarques:
  - Un transducteur ne définit pas un langage mais une relation binaire (ensemble de paires de chaînes).
  - La notion de calcul et de calcul réussi est inchangée. Sauf qu'un calcul n'est plus associé à une chaîne mais une paire de chaîne.

Exemple de relation pour la traduction français-anglais:

$$\{(un, a), (un, one), (gratuit, free), (libre, free)\}$$

Ainsi relation $\neq$ fonction: une chaîne peut avoir plusieurs traductions différentes et deux chaînes différentes peuvent avoir la même traduction.

Exécution d'un transducteur fini

Un FST peut donner lieu à plusieurs exécutions:

- étant donné une chaîne d'entrée, déterminer sa ou ses traductions en sortie
- étant donné une chaîne de sortie, déterminer la ou les entrées dont la sortie est la traduction
- étant donné une paire de chaîne, déterminer si cette paire appartient à la relation
- exhiber une ou plusieurs paires de la relation

Exécution d'un transducteur fini

Un FST peut donner lieu à plusieurs exécutions:

- étant donné une chaîne d'entrée, déterminer sa ou ses traductions en sortie
- étant donné une chaîne de sortie, déterminer la ou les entrées dont la sortie est la traduction
- étant donné une paire de chaîne, déterminer si cette paire appartient à la relation
- exhiber une ou plusieurs paires de la relation

Exécution d'un transducteur fini

Un FST peut donner lieu à plusieurs exécutions:

- étant donné une chaîne d'entrée, déterminer sa ou ses traductions en sortie
- étant donné une chaîne de sortie, déterminer la ou les entrées dont la sortie est la traduction
- étant donné une paire de chaîne, déterminer si cette paire appartient à la relation
- exhiber une ou plusieurs paires de la relation

Exécution d'un transducteur fini

Un FST peut donner lieu à plusieurs exécutions:

- étant donné une chaîne d'entrée, déterminer sa ou ses traductions en sortie
- étant donné une chaîne de sortie, déterminer la ou les entrées dont la sortie est la traduction
- étant donné une paire de chaîne, déterminer si cette paire appartient à la relation
- exhiber une ou plusieurs paires de la relation

Relations régulières

Soit deux alphabets Σ_o et Σ_i, une relation régulière est une relation définie de la façon suivante:

- $\emptyset$ est une relation régulière
- $\{(\epsilon : \epsilon)\}$ est une relation régulière
- $\forall x \in \Sigma_o, y \in \Sigma_i$, $\{(x : y)\}$ est une relation régulière
- Si R et S sont des relations régulières, alors les relations suivantes sont aussi régulières:
  - $R \cup S$
  - RS
  - $R^\star$
- aucune autre relation n'est régulière.

Théorème de Kleene généralisé

Définition (rappel): Une relation est reconnaissable si il existe un transducteur qui la réalise.

Théorème de Kleene généralisé: Une relation est reconnaissable si et seulement si elle est régulière.

Corollaire: une relation est régulière si et seulement si il existe un transducteur fini qui la réalise.

Théorème de Kleene généralisé

Définition (rappel): Une relation est reconnaissable si il existe un transducteur qui la réalise.

Théorème de Kleene généralisé: Une relation est reconnaissable si et seulement si elle est régulière.

Corollaire: une relation est régulière si et seulement si il existe un transducteur fini qui la réalise.

Un transducteur peut donc être vu à la fois comme:

- un automate sur $\Sigma_o \times \Sigma_i$ (auquel s'applique les opérations de suppression des ϵ, de déterminisation, de minimisation, ...)
- une relation régulière sur $\Sigma_o^* \times \Sigma_i^*$

Opérations unaires sur les relations

Soit R une relation de $\Sigma_o^* \times \Sigma_i^*$, on définit:

- L'inverse (ou réciproque) de R est $R^{-1} = \{(v, u) | u \in \Sigma_o^*, v \in \Sigma_i^*\}$
- La restriction de R au langage L se note: $\sigma_L(R) = \{(u, v) \in R | u \in L)\}$
- La projection de R sur
  - les entrées (domaine) est $\text{dom}(R) = \{u \in \Sigma_o^* | \exists v \in \Sigma_i^*, (u, v) \in R\}$
  - les sorties (image) est $\text{imag}(R) = \{v \in \Sigma_i^* | \exists u \in \Sigma_i^*, (u, v) \in R\}$
  - Remarque: $\text{dom}(R)^{-1} = \text{imag}(R)$
- La relation transposée de R est ${}^tR = \{({}^tu, {}^tv) | (u, v) \in R\}$
  - soit $u = a_1 a_2 \ldots a_n$, le miroir ou transposé de u est ${}^tu = a_n \ldots a_2 a_1$
- La relation complémentaire de R est $\overline{R} = \{(u, v) \in \Sigma_o^* \times \Sigma_i^* | (u, v) \notin R\}$

Opérations unaires sur les relations

Soit R une relation de $\Sigma_o^* \times \Sigma_i^*$, on définit:

- L'inverse (ou réciproque) de R est $R^{-1} = \{(v, u) | u \in \Sigma_o^*, v \in \Sigma_i^*\}$
- La restriction de R au langage L se note:
  $\sigma_L(R) = \{(u, v) \in R | u \in L)\}$
- La projection de R sur
  - les entrées (domaine) est $\mathrm{dom}(R) = \{u \in \Sigma_o^* | \exists v \in \Sigma_i^*, (u, v) \in R\}$
  - les sorties (image) est $\mathrm{imag}(R) = \{v \in \Sigma_i^* | \exists u \in \Sigma_i^*, (u, v) \in R\}$
  - Remarque: $\mathrm{dom}(R)^{-1} = \mathrm{imag}(R)$
- La relation transposée de R est ${}^tR = \{({}^tu, {}^tv) | (u, v) \in R\}$
  - soit $u = a_1 a_2 \dots a_n$, le miroir ou transposé de u est ${}^tu = a_n \dots a_2 a_1$
- La relation complémentaire de R est
  $\overline{R} = \{(u, v) \in \Sigma_o^* \times \Sigma_i^* | (u, v) \notin R\}$

Opérations unaires sur les relations

Soit R une relation de $\Sigma_o^* \times \Sigma_i^*$, on définit:

- L'inverse (ou réciproque) de R est $R^{-1} = \{(v, u) | u \in \Sigma_o^*, v \in \Sigma_i^*\}$
- La restriction de R au langage L se note: $\sigma_L(R) = \{(u, v) \in R | u \in L)\}$
- La projection de R sur
  - les entrées (domaine) est $\mathrm{dom}(R) = \{u \in \Sigma_o^* | \exists v \in \Sigma_i^*, (u, v) \in R\}$
  - les sorties (image) est $\mathrm{imag}(R) = \{v \in \Sigma_i^* | \exists u \in \Sigma_i^*, (u, v) \in R\}$
  - Remarque: $\mathrm{dom}(R)^{-1} = \mathrm{imag}(R)$
- La relation transposée de R est ${}^tR = \{({}^tu, {}^tv) | (u, v) \in R\}$
  - soit $u = a_1 a_2 \ldots a_n$, le miroir ou transposé de u est ${}^tu = a_n \ldots a_2 a_1$
- La relation complémentaire de R est $\overline{R} = \{(u, v) \in \Sigma_o^* \times \Sigma_i^* | (u, v) \notin R\}$

Opérations unaires sur les relations

Soit R une relation de $\Sigma_o^* \times \Sigma_i^*$, on définit:

- L'inverse (ou réciproque) de R est $R^{-1} = \{(v,u) | u \in \Sigma_o^*, v \in \Sigma_i^*\}$
- La restriction de R au langage L se note: $\sigma_L(R) = \{(u,v) \in R | u \in L)\}$
- La projection de R sur
  - les entrées (domaine) est $\mathrm{dom}(R) = \{u \in \Sigma_o^* | \exists v \in \Sigma_i^*, (u,v) \in R\}$
  - les sorties (image) est $\mathrm{imag}(R) = \{v \in \Sigma_i^* | \exists u \in \Sigma_i^*, (u,v) \in R\}$
  - Remarque: $\mathrm{dom}(R)^{-1} = \mathrm{imag}(R)$
- La relation transposée de R est ${}^tR = \{({}^tu, {}^tv) | (u,v) \in R\}$
  - soit $u = a_1 a_2 \ldots a_n$, le miroir ou transposé de u est ${}^tu = a_n \ldots a_2 a_1$
- La relation complémentaire de R est $\overline{R} = \{(u,v) \in \Sigma_o^* \times \Sigma_i^* | (u,v) \notin R\}$

Opérations unaires sur les relations

Soit R une relation de $\Sigma_o^* \times \Sigma_i^*$, on définit:

- L'inverse (ou réciproque) de R est $R^{-1} = \{(v, u) | u \in \Sigma_o^*, v \in \Sigma_i^*\}$
- La restriction de R au langage L se note: $\sigma_L(R) = \{(u, v) \in R | u \in L)\}$
- La projection de R sur
  - les entrées (domaine) est $\mathrm{dom}(R) = \{u \in \Sigma_o^* | \exists v \in \Sigma_i^*, (u, v) \in R\}$
  - les sorties (image) est $\mathrm{imag}(R) = \{v \in \Sigma_i^* | \exists u \in \Sigma_i^*, (u, v) \in R\}$
  - Remarque: $\mathrm{dom}(R)^{-1} = \mathrm{imag}(R)$
- La relation transposée de R est ${}^tR = \{({}^tu, {}^tv) | (u, v) \in R\}$
  - soit $u = a_1 a_2 \ldots a_n$, le miroir ou transposé de u est ${}^tu = a_n \ldots a_2 a_1$
- La relation complémentaire de R est $\overline{R} = \{(u, v) \in \Sigma_o^* \times \Sigma_i^* | (u, v) \notin R\}$

Opérations binaires sur les relations

Soit R_1 et R_2 deux relations de $\Sigma_o^* \times \Sigma_i^*$, on définit:

Opérations régulières (déjà vues!):

- L'union: $R_1 \cup R_2 = \{(u,v) \in \Sigma_o^* \times \Sigma_i^* | (u,v) \in R \text{ ou } (u,v) \in S\}$
- La concaténation:
  $R_1 R_2 = \{(u_1 u_2, v_1 v_2) \in \Sigma_o^* \times \Sigma_i^* | (u_1, v_1) \in R_1 \text{ et } (u_2, v_2) \in R_2\}$
- La fermeture transitive de R est $R^\star = \bigcup_{n \geq 1} R^n$

Opérations binaires sur les relations

Soit R_1 et R_2 deux relations de $\Sigma_o^* \times \Sigma_i^*$, on définit:

Opérations régulières (déjà vues!):

- L'union: $R_1 \cup R_2 = \{(u, v) \in \Sigma_o^* \times \Sigma_i^* | (u, v) \in R \text{ ou } (u, v) \in S\}$
- La concaténation:
  $R_1 R_2 = \{(u_1 u_2, v_1 v_2) \in \Sigma_o^* \times \Sigma_i^* | (u_1, v_1) \in R_1 \text{ et } (u_2, v_2) \in R_2\}$
- La fermeture transitive de R est $R^\star = \bigcup_{n \geq 1} R^n$

Nouveauté spécifique aux transducteurs:

- La composition est $R_2 \circ R_1 = \{(u, v) | \exists z : (u, z) \in R_1, (z, v) \in R_2\}$
  - dans le cas de la composition, on peut avoir des alphabets d'entrée et de sortie différents
  - par contre $\Sigma_o^1 = \Sigma_i^2$,

Opérations régulières

Rappel: opération régulière = opération close sur les relations régulières.

- Opérations unaires:
  - Inverse R^{-1}
  - Transposition ${}^{t}R$
  - Projection $\mathrm{dom}(R)$ et $\mathrm{imag}(R)$

- Opérations binaires:
  - Union $R_1 \cup R_2$
  - Concaténation $R_1 \cdot R_2$
  - Fermeture de Kleene $R^\star$
  - Composition: $R_1 \circ R_2$

- Ne sont pas des opérations régulières (contrairement aux opérations sur les automates):
  - Complémentaire $\bar{R}$
  - Différence $R_1 \setminus R_2$
  - Intersection $R_1 \cap R_2$

PLAN DE LA SECTION ACTUELLE

Opérations spécifiques aux transducteurs

Il existe un grand nombre d'opérations qui permette de combiner les transducteurs (pondérés ou non). Parmi celles-ci, quelques unes sont spécifiques aux transducteurs.

- Composition
- opérations d'optimisation:
  - Suppression des ϵ-transitions
  - déterminisation
  - minimisation
  - weight-pushing: normalisation de la distribution des poids suivant les chemins des WFST (ou WFSA). Permet d'améliorer les performances de certaines applications (TAL).
- recherche de plus court chemin (nécessaire pour d'autres opérations)
- intersection, différence
- émondage (pruning): suppression des états inutiles.
- synchronisation
- etc...

Composition: définition

On note un WFST $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$
Et la fonction de transition telle que: $e = (q, a, p, r) \in \delta$ et $a = a_i : a_o$
<u>Définition</u>: $R_2 \circ R_1 = \{(u, v) | \exists z : (u, z) \in R_1, (z, v) \in R_2\}$

<u>Théorème</u>:
La composition de deux relations régulières est régulière.

Composition: définition

On note un WFST $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$
Et la fonction de transition telle que: $e = (q, a, p, r) \in \delta$ et $a = a_i : a_o$
Définition: $R_2 \circ R_1 = \{(u, v) | \exists z : (u, z) \in R_1, (z, v) \in R_2\}$

Théorème:
La composition de deux relations régulières est régulière.

Démonstration:
Soit R_1 et R_2 2 relations régulières, reconnues respectivement par T_1 et T_2. On construit un transducteur $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$ pour $R_1 \circ R_2$ tel que:

- $T(x, y) = \bigoplus_z T_1(x, z) \otimes T_2(z, y)$
- Pour deux relations $(q_1, a : b, w_1, r_1) \in \delta_1$ et $(q_2, b : c, w_2, r_2) \in \delta_2$
- La nouvelle relation devient: $((q_1, q_2), a : c, w_1 \otimes w_2, (r_1, r_2))$

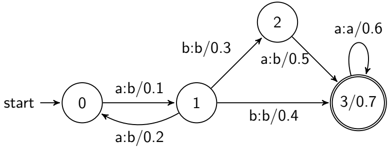
Composition: algorithme

On note un WFST $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$
Et la fonction de transition telle que: $e = (q, a_i : a_o, w, r) \in \delta$

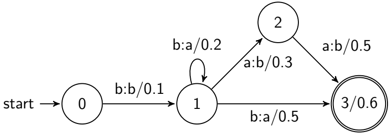
```
Data: T_1, T_2
𝒬 ← I_1 × I_2;
𝒦 ← I_1 × I_2;
while 𝒦 ≠ ∅ do // Construction des états de proche en proche
    q = (q_1, q_2) ← Head(𝒦);
    Dequeue(𝒦);
    if q ∈ I_1 × I_2 then // Construction des états et poids initiaux
        I ← I ∪ {q};
        λ(q) ← λ_1(q_1) ⊗ λ_2(q_2);
    end
    if q ∈ F_1 × F_2 then // Construction des états et poids finaux
        F ← F ∪ {q};
        ρ(q) ← ρ_1(q_1) ⊗ ρ_2(q_2);
    end
    // Combinaison des relations partant de q_1 et q_2
    // telles que l'étiquette de sortie de e_1 soit égale à l'étiquette d'entrée de e_2
    for each (e_1, e_2) ∈ δ_1(q_1) × δ_2(q_2) such that a_o^1 = a_i^2 do
        if r = (r_1, r_2) ∈ Q then
            Q ← Q ∪ {r};
            Enqueue(𝒦, r);
        end
    end
    δ ← δ ∪ {(q, a_i^1 : a_o^2, w_1 ⊗ w_2, r)}// Augmentation de la relation;
end
return T1 ∘ T_2
```

Composition: exemple

T_1: WFST avec le semi-anneau probabilité



T_2: WFST avec le semi-anneau probabilité



δ_1	q_1	i:o	w_1	r_1
q_0	0	a:b	0.1	1
	1	a:b	0.2	0
	1	b:b	0.3	2
	1	b:b	0.4	3
	2	a:b	0.5	3
q_f	3	a:a	0.6	3

δ_2	q_2	i:o	w_2	r_2
q_0	0	b:b	0.1	1
	1	b:a	0.2	1
	1	a:b	0.3	2
	1	a:b	0.4	3
	2	b:a	0.5	3
q_f	3			

Composition: exemple

δ_1	q_1	i:o	w_1	r_1
q_0	0	a:**b**	0.1	1
	1	a:b	0.2	0
	1	b:b	0.3	2
	1	b:b	0.4	3
	2	a:b	0.5	3
q_f	3	a:a	0.6	3

δ_2	q_2	i:o	w_2	r_2
q_0	0	**b**:b	0.1	1
	1	b:a	0.2	1
	1	a:b	0.3	2
	1	a:b	0.4	3
	2	b:a	0.5	3
q_f	3			

$T_1 \circ T_2$

(q_1, q_2)	i:o	$w_1 \otimes w_2$	(r_1, r_2)
$\{0^1, 0^2\}$	a:b	$0.1 \times 0.1 = 0.01$	$\{1^1, 1^2\}$

Composition: exemple

δ_1	q_1	i:o	w_1	r_1
q_0	0	a:b	0.1	1
	1	a:**b**	0.2	0
	1	b:**b**	0.3	2
	1	b:**b**	0.4	3
	2	a:b	0.5	3
q_f	3	a:a	0.6	3

δ_2	q_2	i:o	w_2	r_2
q_0	0	b:b	0.1	1
	1	**b**:a	0.2	1
	1	~~a:b~~	0.3	2
	1	~~a:b~~	0.4	3
	2	b:a	0.5	3
q_f	3			

$T_1 \circ T_2$

(q_1, q_2)	i:o	$w_1 \otimes w_2$	(r_1, r_2)
$\{0^1, 0^2\}$	a:b	$0.1 \times 0.1 = 0.01$	$\{1^1, 1^2\}$
$\{1^1, 1^2\}$	a:a	$0.2 \times 0.2 = 0.04$	$\{0^1, 1^2\}$
$\{1^1, 1^2\}$	b:a	$0.3 \times 0.2 = 0.06$	$\{2^1, 1^2\}$
$\{1^1, 1^2\}$	b:a	$0.4 \times 0.2 = 0.08$	$\{3^1, 1^2\}$

Composition: exemple

δ_1	q_1	i:o	w_1	r_1
q_0	0	a:**b**	0.1	1
	1	a:b	0.2	0
	1	b:b	0.3	2
	1	b:b	0.4	3
	2	a:b	0.5	3
q_f	3	a:a	0.6	3

δ_2	q_2	i:o	w_2	r_2
q_0	0	b:b	0.1	1
	1	**b**:a	0.2	1
	1	~~a:b~~	0.3	2
	1	~~a:b~~	0.4	3
	2	b:a	0.5	3
q_f	3			

$T_1 \circ T_2$

(q_1, q_2)	i:o	$w_1 \otimes w_2$	(r_1, r_2)
$\{0^1, 0^2\}$	a:b	$0.1 \times 0.1 = 0.01$	$\{1^1, 1^2\}$
$\{1^1, 1^2\}$	a:a	$0.2 \times 0.2 = 0.04$	$\{0^1, 1^2\}$
$\{1^1, 1^2\}$	b:a	$0.3 \times 0.2 = 0.06$	$\{2^1, 2^2\}$
$\{1^1, 1^2\}$	b:a	$0.4 \times 0.2 = 0.08$	$\{3^1, 3^2\}$
$\{0^1, 1^2\}$	a:a	$0.1 \times 0.2 = 0.002$	$\{1^1, 1^2\}$

Composition: exemple

δ_1	q_1	i:o	w_1	r_1
q_0	0	a:b	0.1	1
	1	a:b	0.2	0
	1	b:b	0.3	2
	1	b:b	0.4	3
	2	a:**b**	0.5	3
q_f	3	a:a	0.6	3

δ_2	q_2	i:o	w_2	r_2
q_0	0	b:b	0.1	1
	1	**b**:a	0.2	1
	1	~~a:b~~	0.3	2
	1	~~a:b~~	0.4	3
	2	b:a	0.5	3
q_f	3			

$T_1 \circ T_2$

(q_1, q_2)	i:o	$w_1 \otimes w_2$	(r_1, r_2)
$\{0^1, 0^2\}$	a:b	$0.1 \times 0.1 = 0.01$	$\{1^1, 1^2\}$
$\{1^1, 1^2\}$	a:a	$0.2 \times 0.2 = 0.04$	$\{0^1, 1^2\}$
$\{1^1, 1^2\}$	b:a	$0.3 \times 0.2 = 0.06$	$\{2^1, 2^2\}$
$\{1^1, 1^2\}$	b:a	$0.4 \times 0.2 = 0.08$	$\{3^1, 3^2\}$
$\{0^1, 1^2\}$	a:a	$0.1 \times 0.2 = 0.02$	$\{1^1, 1^2\}$
$\{2^1, 1^2\}$	a:a	$0.5 \times 0.2 = 0.1$	$\{3^1, 1^2\}$

Composition: exemple

δ_1	q_1	i:o	w_1	r_1
q_0	0	a:b	0.1	1
	1	a:b	0.2	0
	1	b:b	0.3	2
	1	b:b	0.4	3
	2	a:b	0.5	3
q_f	3	a:**a**	0.6	3

δ_2	q_2	i:o	w_2	r_2
q_0	0	b:b	0.1	1
	1	~~b:a~~	0.2	1
	1	**a**:b	0.3	2
	1	**a**:b	0.4	3
	2	b:a	0.5	3
q_f	3			

$T_1 \circ T_2$

(q_1, q_2)	i:o	$w_1 \otimes w_2$	(r_1, r_2)
$\{0^1, 0^2\}$	a:b	$0.1 \times 0.1 = 0.01$	$\{1^1, 1^2\}$
$\{1^1, 1^2\}$	a:a	$0.2 \times 0.2 = 0.04$	$\{0^1, 1^2\}$
$\{1^1, 1^2\}$	b:a	$0.3 \times 0.2 = 0.06$	$\{2^1, 2^2\}$
$\{1^1, 1^2\}$	b:a	$0.4 \times 0.2 = 0.08$	$\{3^1, 3^2\}$
$\{0^1, 1^2\}$	a:a	$0.1 \times 0.2 = 0.02$	$\{1^1, 1^2\}$
$\{2^1, 1^2\}$	a:a	$0.5 \times 0.2 = 0.1$	$\{3^1, 1^2\}$
$\{3^1, 1^2\}$	a:b	$0.6 \times 0.3 = 0.18$	$\{3^1, 2^2\}$
$\{3^1, 1^2\}$	a:b	$0.6 \times 0.4 = 0.24$	$\{3^1, 3^2\}$

Composition: exemple

δ_1	q_1	i:o	w_1	r_1
q_0	0	a:b	0.1	1
	1	a:b	0.2	0
	1	b:b	0.3	2
	1	b:b	0.4	3
	2	a:b	0.5	3
q_f	3	~~a:a~~	0.6	3

δ_2	q_2	i:o	w_2	r_2
q_0	0	b:b	0.1	1
	1	b:a	0.2	1
	1	a:b	0.3	2
	1	a:b	0.4	3
	2	~~b:a~~	0.5	3
q_f	3			

$T_1 \circ T_2$

(q_1, q_2)	i:o	$w_1 \otimes w_2$	(r_1, r_2)
$\{0^1, 0^2\}$	a:b	$0.1 \times 0.1 = 0.01$	$\{1^1, 1^2\}$
$\{1^1, 1^2\}$	a:a	$0.2 \times 0.2 = 0.04$	$\{0^1, 1^2\}$
$\{1^1, 1^2\}$	b:a	$0.3 \times 0.2 = 0.06$	$\{2^1, 2^2\}$
$\{1^1, 1^2\}$	b:a	$0.4 \times 0.2 = 0.08$	$\{3^1, 3^2\}$
$\{0^1, 1^2\}$	a:a	$0.1 \times 0.2 = 0.02$	$\{1^1, 1^2\}$
$\{2^1, 1^2\}$	a:a	$0.5 \times 0.2 = 0.1$	$\{3^1, 1^2\}$
$\{3^1, 1^2\}$	a:b	$0.6 \times 0.3 = 0.18$	$\{3^1, 2^2\}$
$\{3^1, 1^2\}$	a:b	$0.6 \times 0.4 = 0.24$	$\{3^1, 3^2\}$
$\{3^1, 2^2\}$	$\emptyset$		
$\{3^1, 2^2\}$	$\emptyset$		

Composition: exemple

δ_1	q_1	i:o	w_1	r_1
q_0	0	a:b	0.1	1
	1	a:b	0.2	0
	1	b:b	0.3	2
	1	b:b	0.4	3
	2	a:b	0.5	3
q_f	3	~~a:a~~	0.6	3

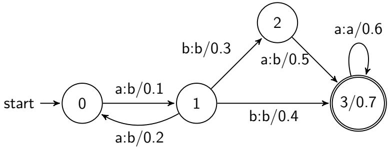
δ_2	q_2	i:o	w_2	r_2
q_0	0	b:b	0.1	1
	1	b:a	0.2	1
	1	a:b	0.3	2
	1	a:b	0.4	3
	2	b:a	0.5	3
q_f	3	$\emptyset$		

$T_1 \circ T_2$

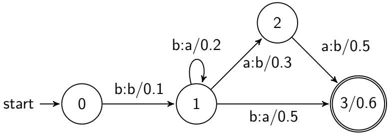
(q_1, q_2)	i:o	$w_1 \otimes w_2$	(r_1, r_2)
$\{0^1, 0^2\}$	a:b	$0.1 \times 0.1 = 0.01$	$\{1^1, 1^2\}$
$\{1^1, 1^2\}$	a:a	$0.2 \times 0.2 = 0.04$	$\{0^1, 1^2\}$
$\{1^1, 1^2\}$	b:a	$0.3 \times 0.2 = 0.06$	$\{2^1, 2^2\}$
$\{1^1, 1^2\}$	b:a	$0.4 \times 0.2 = 0.08$	$\{3^1, 3^2\}$
$\{0^1, 1^2\}$	a:a	$0.1 \times 0.2 = 0.02$	$\{1^1, 1^2\}$
$\{2^1, 1^2\}$	a:a	$0.5 \times 0.2 = 0.1$	$\{3^1, 1^2\}$
$\{3^1, 1^2\}$	a:b	$0.6 \times 0.3 = 0.18$	$\{3^1, 2^2\}$
$\{3^1, 1^2\}$	a:b	$0.6 \times 0.4 = 0.24$	$\{3^1, 3^2\}$
$\{3^1, 3^2\}$	$\emptyset$		

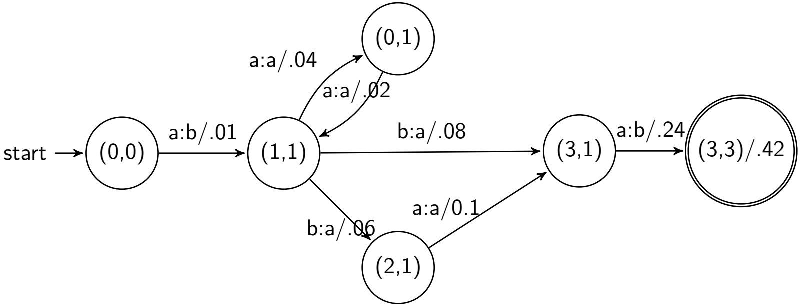
Composition: exemple

T_1: WFST avec le semi-anneau probabilité



T_2: WFST avec le semi-anneau probabilité





Composition avec des ϵ-transitions

$i_1 : o_1$	$i_2 : o_2$	$i : o$
b:ϵ	ϵ:e	b:e
ϵ	ϵ:e	ϵ:e
b:ϵ	ϵ	b:ϵ

Les ϵ transitions doivent ensuite être supprimées par fermeture arrière ou avant suivant les algorithmes vus précédemment sur les automates.

NB: tout automate possède un automate équivalent sans transitions. Par contre ce n'est pas le cas des transducteurs où les transitions spontanées ne peuvent pas être systématiquement supprimées.

Déterminisation

Théorème (rappel): pour tout AFN A défini sur Σ, il existe un AFD A' équivalent à A. Si A possède n états, A' possède au plus 2^n états.

La déterminisation consiste à ne construire que les états accessibles à partir de I, de proche en proche.

Dans le cas des automates ou transducteurs pondérés, la détermination n'est pas toujours possible

Déterminisation: algorithme

On note un WFST $T = (\Sigma_i, \Sigma_o, Q, I, F, \delta, \lambda, \rho)$
Et la fonction de transition telle que: $e = (q, a_i : a_o, w, r) \in \delta$

Data: T
$i' \leftarrow \{(i, \lambda(i)) | i \in I\}; \lambda'(i') \leftarrow \bar{1}$ // Initialisation états et poids initiaux;
$\mathcal{Q} \leftarrow \{i'\}$;
while $\mathcal{K} \neq \emptyset$ **do** // Construction des états de proche en proche
  $p' \leftarrow \text{Head}(\mathcal{Q})$;
  $\text{Dequeue}(\mathcal{Q})$;
  for *each* $a_i \in i(\delta(Q(p')))$ **do** // Boucle sur les étiquettes d'entrée ayant un état d'origine dans p'
    $w' \leftarrow \oplus\{v \otimes w | (p, v) \in p', (p, a, w, r) \in \delta\}$ // poids associé;
    $r' \leftarrow \left\{ \left(r, \oplus\{w'^{-1} \otimes (v \otimes w) | (p, v) \in p', (p, a, w, r) \in \delta\} \right) \right\}$ // états destination;
    $\delta' \leftarrow \delta' \cup \{(p', a, w', r')\}$ // Augmentation de la relation;
    if $q' \notin Q'$ **then**
      $Q' \leftarrow Q' \cup \{r'\}$;
      if $Q(r') \cap F \neq \emptyset$ **then** // Si l'état destination fait partie des états finaux de T
        $F' \leftarrow F' \cap \{r'\}$ // Augmentation des états finaux;
        $\rho'(r') \leftarrow \oplus\{w \otimes \rho(r) | (r, v) \in r', r \in F\}$ // Calcul du poids final;
      end
      $\text{Enqueue}(\mathcal{K}, r')$;
    end
  end
end
return T'

Les états jumeaux

Définition 1:
Soit un automate $\mathcal{A}$ pondéré sur un semi-anneau $\mathbb{K}$ et $q, p \in Q$.
q et q' sont dits frères si:

- il existe $x \in \Sigma^\star$ tel que $I \xrightarrow{x} q$ ET $I \xrightarrow{x} p$
- il existe $y \in \Sigma^\star$ tel que $q \xrightarrow{y} q$ ET $p \xrightarrow{y} p$ (boucle)

Définition 2:
q et q' sont dits jumeaux si:

$$w\left(q \xrightarrow{y} q\right) = w\left(p \xrightarrow{y} p\right)$$

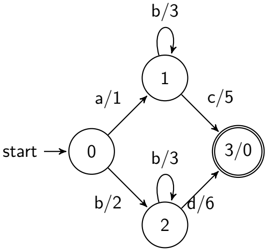
Propriété: $\mathcal{A}$ possède la propriété des jumeaux (twin property) si tous les états frères sont jumeaux.

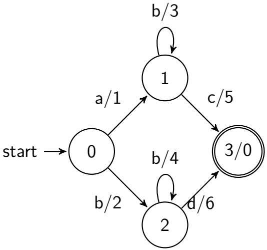
Théorème: Si $\mathcal{A}$ possède la propriété des jumeaux et $\mathbb{K} =$ Tropical, alors T peut être déterminisé.

Rq: extension possible aux transducteurs: $x = a_i : a_o$

Les états jumeaux

Deux automates pondérés définis sur le semi-anneau tropical.





Pour les deux automates:

- $q_0 \xrightarrow{a} q_1$ ET $q_0 \xrightarrow{a} q_2$
- $q_1 \xrightarrow{b} q_1$ ET $q_2 \xrightarrow{b} q_2$

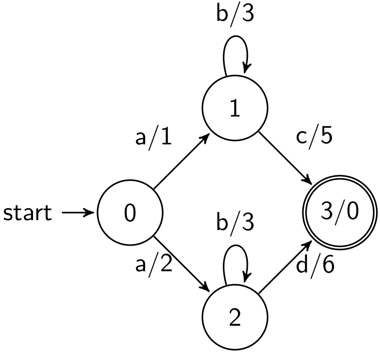
Les états 1 et 2 sont frères.

- à gauche: $w\left(q_1 \xrightarrow{b} q_1\right) = w\left(q_2 \xrightarrow{b} q_2\right) = 3 \Rightarrow$ états jumeaux
- à droite: $w\left(q_1 \xrightarrow{b} q_1\right) = 3 \neq w\left(q_2 \xrightarrow{b} q_2\right) = 4 \Rightarrow$ états pas jumeaux

A gauche, tous les états frères sont jumeaux (+ anneau tropical) donc l'automate est déterminisable.
A droite, les états frères ne sont pas jumeaux donc l'automate ne peut pas être déterminisé.

Déterminisation d'un WFST: exemple

Nom	Ensemble	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0



δ	q	i	p	r
q_0	0	a	1	1
	0	a	2	2
	1	b	3	1
	1	c	5	3
	2	b	3	2
	2	d	6	3
q_f	3			

Déterminisation d'un WFST: exemple

Nom	Ensemble	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

On construit de proche en proche de puis q0, les états accessibles résultant en général à des automates complets ayant moins que 2n états

δ	q	i	p	r
q_0	0	a	1	1
	0	a	2	2
	1	c	5	3
	1	b	3	1
	2	b	3	2
	2	d	6	3
q_f	3			

δ'	q	i	p	r
q_0	{0,0}	a	p'=1	q'={(1,0),(2,1)}

$$p' = \oplus\left\{(\bar{1} \otimes 1), (\bar{1} \otimes 2)\right\} = \min\{(0+1), (0+2)\} = 1$$

$$q' = \left\{\left(1, \oplus\{1^{-1} \otimes (\bar{1} \otimes 1)\}\right), \left(2, \oplus\{1^{-1} \otimes (\bar{1} \otimes 2)\}\right)\right\}$$
$$= \{(1, \min\{-1 + (0+1)\}), (2, \min\{-1 + (0+2)\})\} = \{(1,0), (2,1)\}$$

Déterminisation d'un WFST: exemple

Nom	Ensemble	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

On construit de proche en proche de puis q0, les états accessibles résultant en général à des automates complets ayant moins que 2n états

δ	q	i	p	r
q_0	0	a	1	1
	0	a	2	2
	1	b	3	1
	1	c	5	3
	2	b	3	2
	2	d	6	3
q_f	3			

δ'	q	i	p	r
q_0	{0,0}	a	p'=1	q'={(1,0),(2,1)}
	{(1,0),(2,1)}	b	p'=3	q'={(1,0),(2,1)}

$$p' = \oplus\{(0 \otimes 3), (1 \otimes 3)\} = \min\{(0+3), (1+3)\} = 3$$

$$q' = \left\{\left(1, \oplus\{3^{-1} \otimes (0 \otimes 3)\}\right), \left(2, \oplus\{3^{-1} \otimes (1 \otimes 3)\}\right)\right\}$$

$$= \{(1, \min\{-3 + (0+3)\}), (2, \min\{-3 + (1+3)\})\} = \{(1,0), (2,1)\}$$

Déterminisation d'un WFST: exemple

Nom	Ensemble	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

On construit de proche en proche de puis q0, les états accessibles résultant en général à des automates complets ayant moins que 2n états

δ	q	i	p	r
q_0	0	a	1	1
	0	a	2	2
	1	b	3	1
	1	c	5	3
	2	b	3	2
	2	d	6	3
q_f	3			

δ'	q	i	p	r
q_0	{0,0}	a	p'=1	q'={(1,0),(2,1)}
	{(1,0),(2,1)}	b	p'=3	q'={(1,0),(2,1)}
	{(1,0),(2,1)}	c	p'=5	q'={(3,0)}

$$p' = \oplus\{(0 \otimes 5)\} = \min\{(0+5)\} = 5$$

$$q' = \left\{\left(3, \oplus\{5^{-1} \otimes (0 \otimes 5)\}\right)\right\}$$

$$= \{(3, \min\{-5 + (0+5)\})\} = \{(3, 0)\}$$

Déterminisation d'un WFST: exemple

Nom	Ensemble	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

On construit de proche en proche de puis q0, les états accessibles résultant en général à des automates complets ayant moins que 2n états

δ	q	i	p	r
q_0	0	a	1	1
	0	a	2	2
	1	c	5	3
	1	b	3	1
	2	b	3	2
	2	d	6	3
q_f	3			

δ'	q	i	p	r
q_0	{0,0}	a	p'=1	q'={(1,0),(2,1)}
	{(1,0),(2,1)}	b	p'=3	q'={(1,0),(2,1)}
	{(1,0),(2,1)}	c	p'=5	q'={(3,0)}
	{(1,0),(2,1)}	d	p'=7	q'={(3,0)}

$$p' = \oplus\{(1 \otimes 6)\} = \min\{(1+6)\} = 7$$

$$q' = \left\{\left(3, \oplus\{7^{-1} \otimes (1 \otimes 6)\}\right)\right\}$$

$$= \{(3, \min\{-7 + (1+6)\})\} = \{(3, 0)\}$$

Déterminisation d'un WFST: exemple

Nom	Ensemble	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0

On construit de proche en proche de puis q0, les états accessibles résultant en général à des automates complets ayant moins que 2n états

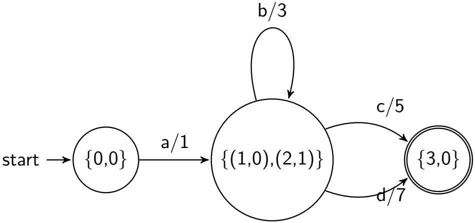
δ	q	i	p	r
q_0	0	a	1	1
	0	a	2	2
	1	c	5	3
	1	b	3	1
	2	b	3	2
	2	d	6	3
q_f	3			

δ'	q	i	p	r
q_0	{0,0}	a	p'=1	q'={(1,0),(2,1)}
	{(1,0),(2,1)}	b	p'=3	q'={(1,0),(2,1)}
	{(1,0),(2,1)}	c	p'=5	q'={(3,0)}
	{(1,0),(2,1)}	d	p'=7	q'={(3,0)}
q_f	{(3,0)}	$\emptyset$	p'=0	q'=$\emptyset$

$$
\begin{aligned}
p' &= 0 \\
q' &= \left\{ \left(3, \oplus\{7^{-1} \otimes (1 \otimes 6)\}\right) \right\} \\
&= \emptyset
\end{aligned}
$$

Déterminisation: exemple

δ'	q	i	p	r
q_0	{0,0}	a	1	{(1,0),(2,1)}
	{(1,0),(2,1)}	b	3	{(1,0),(2,1)}
	{(1,0),(2,1)}	c	5	{(3,0)}
	{(1,0),(2,1)}	d	7	{(3,0)}
q_f	{(3,0)}		0	



Suppression des ϵ-transitions

- Il existe un algorithme de suppression des ϵ-transitions qui produit un WFST équivalent sans transitions du type $\epsilon : \epsilon$
- Il est préférable de supprimer les transitions avant de faire des opérations d'optimisation (mais après la déterminisation)
- Rappel: les ϵ-transitions accroissent le temps d'optimisation, et peuvent causer des problèmes plus ou moins sérieux lors de l'application des algorithmes.

Les ϵ transitions doivent ensuite être supprimées par fermeture arrière ou avant suivant les algorithmes vus précédemment sur les automates.

NB: tout automate possède un automate équivalent sans transitions. Par contre ce n'est pas le cas des transducteurs où les transitions spontanées ne peuvent pas être systématiquement supprimées (en particulier les transitions du type $\epsilon : a$ ou $a : \epsilon$.

Algorithme de suppression des ϵ-transitions

Nous n'allons pas détailler ici l'algorithme complet, mais en voici les idées principales:

- Copier tous les états et les transitions non nulles dans le WFST équivalent.
- Déterminer la fermeture de Kleene de chacun des états d'origine qui sont accessible via des transitions nulles.
- Générer les nouveaux arcs (transitions non nulles) à partir de la fermeture de Kleene. Ajouter des états finaux quand c'est nécessaire.
- Supprimer les états qui ne sont pas co-accessibles.

Algorithme de suppression des ϵ-transitions

Nous n'allons pas détailler ici l'algorithme complet, mais en voici les idées principales:

- Copier tous les états et les transitions non nulles dans le WFST équivalent.
- Déterminer la fermeture de Kleene de chacun des états d'origine qui sont accessible via des transitions nulles.
- Générer les nouveaux arcs (transitions non nulles) à partir de la fermeture de Kleene. Ajouter des états finaux quand c'est nécessaire.
- Supprimer les états qui ne sont pas co-accessibles.

NB: Le choix du semi-anneau affecte la complexité de l'algorithme:

- Le semi-anneau tropical est le plus rapide car il avantage l'approximation de viterbi pour le calcul du plus court chemin.
- Les autres semi-anneaux entraînent une augmentation exponentielle de la complexité.
- Dans le cas de FST non pondérés, la complexité augmente au carré.

Weight-pushing

Nous n'allons pas détailler ici l'algorithme complet, mais en voici les idées principales:

- Le choix de la distribution du poids total dans le WFST n'a aucune conséquence sur la fonction du WSFT
- Par contre, il a fort un impact sur la performance du WFST dans plusieurs applications, notamment en TAL.
- L'algorithme de *weight-pushing* permet de normaliser la distribution des poids suivant les chemins du WSFT.

Chemin le plus court

Soit un WSFT T défini sur le semi-anneau $\mathbb{K}$. Pour chaque état $q \in Q$, on pose $p \to q_f$ un chemin qui part de l'état q et arrive à l'état final $q_f \in F$. On définit la distance:

$$d(q) = \bigoplus_{\pi : p \to q_f} (w(\pi) \otimes \rho(r(\pi))$$

où $r(\pi)$ est un état final obtenu avec le chemin π et $\rho(r(\pi)$, son poids.

NB: la somme n'est définie que si $\mathbb{K}$ est un semi-anneau $k - \mathit{ferm}$é (ex: tropical, probabilité)

L'algorithme consiste à pousser les poids de chaque chemin au plus proche des états initiaux.

Algorithme pour le weight-pushing

L'algorithme consiste à déterminer les distances les plus proches pour chaque état q puis a re-pondérer les transitions, états initiaux et finaux.

$$\forall e \in \delta \text{ telle que } d(p(e)) \neq \overline{0} : w'(e) = d(p(e))^{-1} \otimes w(e) \otimes d(r(e))$$
$$\forall q \in I : \lambda(q) = \lambda(q) \otimes d(q)$$
$$\forall q \in F \text{ tel que } d(q) \neq \overline{0} : \rho'(q) = d(q)^{-1} \otimes \rho(q)$$

Weight-pushing: exemple

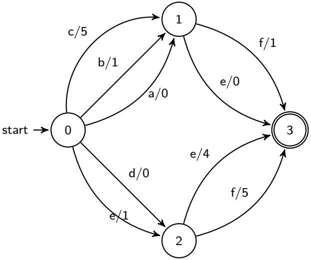
$$d(q) = \bigoplus_{\pi:p\to q_f} (w(\pi) \otimes \rho(r(\pi)))$$

$$\forall e \in \delta \text{ telle que } d(p(e)) \neq \bar{0} : w'(e) = d(p(e))^{-1} \otimes w(e) \otimes d(r(e))$$

$$\forall q \in I : \lambda(q) = \lambda(q) \otimes d(q)$$

$$\forall q \in F \text{ tel que } d(q) \neq \bar{0} : \rho'(q) = d(q)^{-1} \otimes \rho(q)$$

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0



On détermine: $d(0) = 0$; $d(1) = 0$;

e	w'	λ'	ρ'
0		0	
$0 \xrightarrow{c} 1$	5		

$$d(1) = \bigoplus \left\{ \left(w(1 \xrightarrow{e} 3) \otimes \rho(r(1 \xrightarrow{e} 3)) \right), \left(w(1 \xrightarrow{f} 3) \otimes \rho(r(1 \xrightarrow{f} 3)) \right) \right\} = \min\{(0+0), (1+0)\} = 0$$

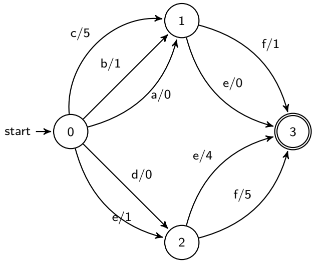
$$w'(0 \xrightarrow{c} 1) = d(0)^{-1} \otimes w(0 \xrightarrow{c} 1) \otimes d(1) = 0 + 5 + 0 = 5$$

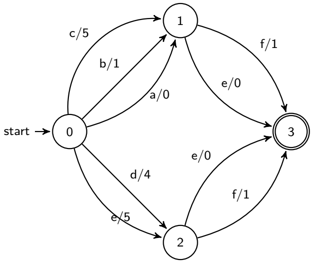
Weight-pushing: exemple

$$d(q) = \bigoplus_{\pi : p \to q_f} (w(\pi) \otimes \rho(r(\pi))$$

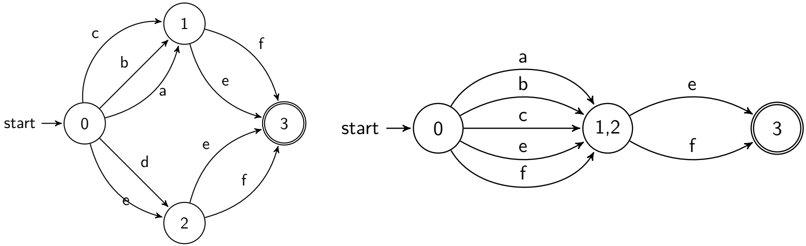
$$\forall e \in \delta \text{ telle que } d(p(e)) \neq \bar{0} : w'(e) = d(p(e))^{-1} \otimes w(e) \otimes d(r(e))$$
$$\forall q \in I : \lambda(q) = \lambda(q) \otimes d(q)$$
$$\forall q \in F \text{ tel que } d(q) \neq \bar{0} : \rho'(q) = d(q)^{-1} \otimes \rho(q)$$

Nom	$\mathbb{K}$	$\oplus$	$\otimes$	$\bar{0}$	$\bar{1}$
Tropical	$\mathbb{R}_+ \cup \{+\infty\}$	min	$+$	$+\infty$	0





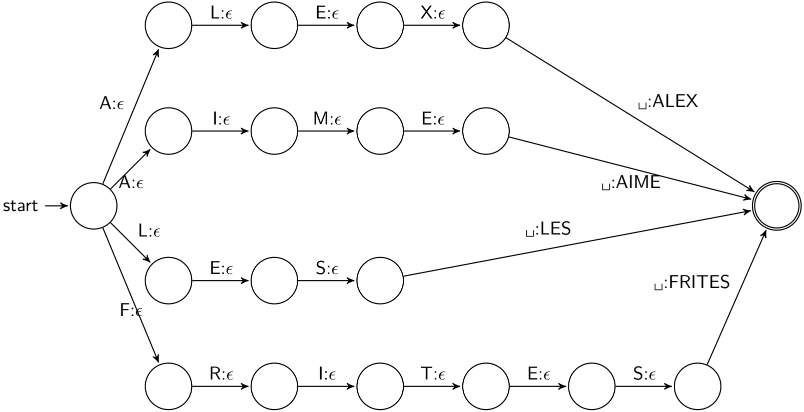
Minimisation

- Un WSFT déterministe est dit minimal lorsqu'il possède un nombre minimal d'états.
- La détermination d'un WFST minimal équivalent passe par la recherche d'états équivalents
    - Il faut d'abord normaliser les poids via un algo de *weight-pushing*
    - Deux états d'un WFST déterministe sont équivalents si il existe des chemins identiques (mêmes étiquettes, mêmes poids) partant de ces états vers un état final.
    - Ces états équivalents sont regroupés pour former un nouvel état.
- 

Application TAL: exemple

Étant donné une phrase, le but est de générer une suite qui donne la nature grammaticale (POS) de chaque mot de la phrase. Par exemple, si la phrase est "Alex aime les frites" alors le but est de générer la séquence NOM VERBE ARTICLE NOM. Pour cela, on définit deux transducteurs.

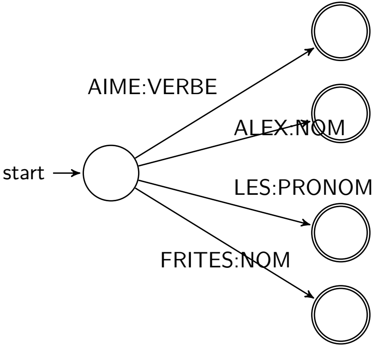
Premier transducteur L pour le lexique:



Application TAL: exemple

Étant donné une phrase, le but est de générer une suite qui donne la nature grammaticale (POS) de chaque mot de la phrase. Par exemple, si la phrase est "Alex aime les frites" alors le but est de générer la séquence NOM VERBE ARTICLE NOM. Pour cela, on définit deux transducteurs.

Second transducteur *POS* pour la syntaxe grammaticale:



Le transducteur composé

$$T = D \circ POS$$

est également un transducteur qui prendra en entrée une séquence de lettres et écrira en sortie une séquence de POS correspondant aux mots de la phrase.

Conclusion

Les transducteurs finis offrent un modèle plus puissant et plus flexible que les automates finis.

- Perte des propriétés de clôture des automates finis pour les opérations d'intersection et différence et d'optimisation automatique (déterminisation, minimalisation)
- On les utilise pour définir un calcul comme une succession d'étapes élémentaires, chaque étape étant réalisée par un transducteur fini: les cascades de transducteurs.
- Limitations: les WFST sont limités par les propriétés des langages (et relations) réguliers. Notamment certains langages ne peuvent pas être représentés $a^n b^n$