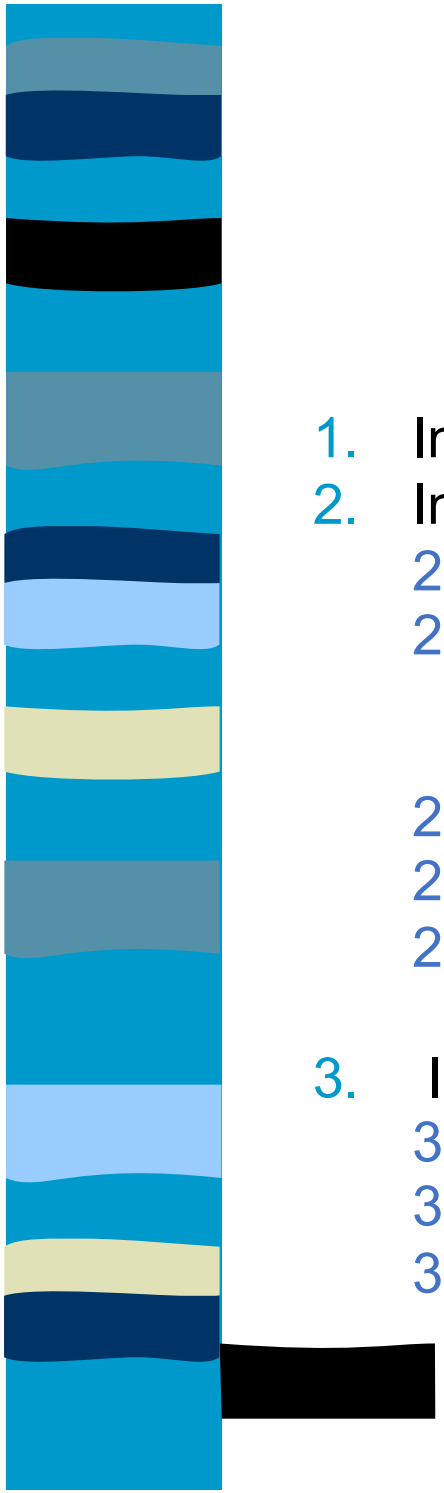


Communication Langagière

Ingénierie des langues et de la parole

1. Introduction générale
2. Ingénierie des langues
   - 2.1. Représentation et codage des textes
   - 2.2. Applications du TALN :
     - 2.2.1. Dictionnaire et étiquetage de surface
     - 2.2.2. Traduction automatique statistique
   - 2.3. Introduction à l'apprentissage profond
   - 2.4. Encodeur/Décodeur
   - 2.5. BERT
3. Ingénierie de la parole
   - 3.1. Rappels de traitement numérique du signal
   - 3.2. Codage et représentation de la parole
   - 3.3. Approches auto-supervisées

Les mots et leur structure

- Aspect morpho-lexical
- Notion de mot
  - Représentation d'un dictionnaire (ou lexique) dans le monde numérique
- Morphologie
  - Structure des mots
- Estimer automatiquement le rôle grammatical (ou syntaxique) d'un mot
  - Etiquetage de surface (*Part-of-speech Tagging POS*)

La Morphologie Lexicale

–B. SCHWISCHAY, Introduction à la lexicologie (hiver 2001/02) Dernière mise à jour : 18-11-01

https://www.home.uni-osnabrueck.de/bschwisc/archives/formation.htm

- Mots Simples : non décomposable
  - *boire*
- Mots Construits : décomposable en éléments significatifs plus petits (mots – morphèmes)
  - ***boisson***, *buvable, buvard, buvette, buveur, imbu,* ***imbuvable***, ***pourboire***

La Morphologie Lexicale

- Construction par **composition**
  - ***pourboire*** < *pour* (préposition) + *boire*

- Construction par **dérivation,** c.-à-d. par adjonction d'un suffixe ou d'un préfixe à l'un des radicaux de ce verbe

La Morphologie Lexicale

- dérivation suffixale
  - ***boisson***

    < *(je) bois + -son* « (ce) qui subit l'action (exprimé par la base) ». « Liquide qui se boit. » Cf. *cuisson, nourrisson*.

  - ***buvable***

    < (nous) buv-(ons) + -able « possibilité ». « Qui peut se boire. » Cf. abordable, faisable.

- dérivation préfixale
  - ***imbuvable***

    < im- (in1-) « élément négatif » + buvable. « Qui n'est pas buvable. » Cf. imbattable, inabordable

La Morphologie Lexicale

- Construction par **dérivation impropre**
  - ***sourire (v.) ® sourire (n.)***
- Construction par **troncation**
  - **auto**[mobile], **radio**[phonie] et **radio**[graphie], **fac**[ulté], **catho**[lique], **cinéma**[tographe], **cine**[ma]

La Morphologie Lexicale

- Construction par **siglaison**
  - **Sigle** *V. T. T. Vélo Tout-Terrain*
  - **Acronyme** *T. I. R. « Transit International Routier »*
  - **Sigle et Acronyme comme base dérivée**

    P. D. G. n. « président-directeur général » ® pédégère n. f. « femme qui exerce les fonctions de P. D. G. »

Lexique

- Définition
  - Liste de mots (ou formes) permettant de
- les identifier dans un texte
- les classer
  - Cette liste est dépendante du domaine d'application (tâche)
  - Chaque mot non présent dans le lexique est considéré comme un mot « hors vocabulaire »

Organisation d'un lexique

- Liste de champs de différents types
  - « forme de surface » (le mot lui-même)
    - Ex: livres
  - Étiquette syntaxique (POS)
    - Ex : nom au pluriel
  - Lemme (forme normalisée du mot)
    - Ex : livre/Ns
  - Scores ou probabilités
    - Ex: 2.4e-05
  - Prononciation
    - Ex: [l i v R]

Type abstrait « lexique »

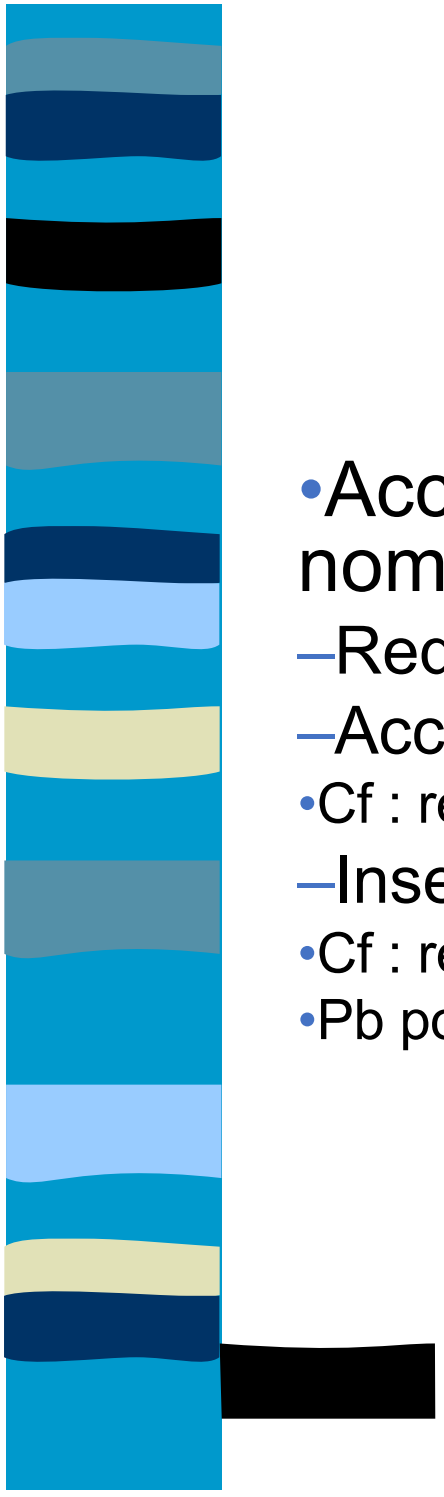
- Méthodes pour un lexique
  - Insertion et suppression
  - Tests d'existence (pour différents champs)
  - Liste du contenu du lexique
  - Extraction d'éléments dont le champ correspond à une certaine requête

Type abstrait « lexique »

- Représentation
  - Listes et tableaux
  - Tables de hachage
  - Arbres
  - Automates d'états finis

Type abstrait « lexique »

- Représentation
  - Listes et tableaux
  - Tables de hachage
  - Arbres
  - Automates d'états finis

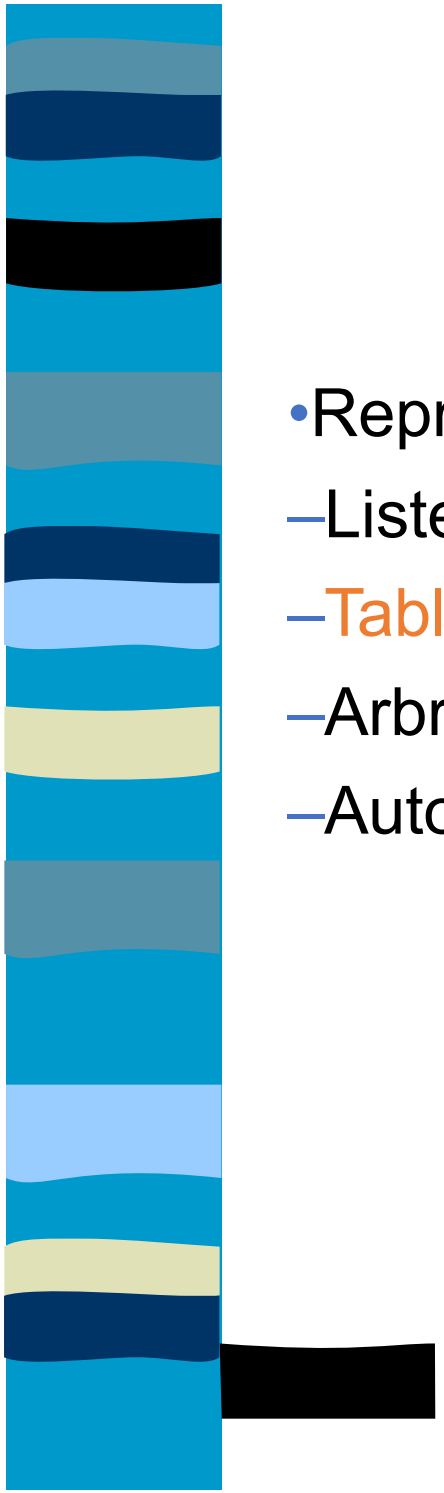


Listes et tableaux

- Accès par valeur : mapping de chaîne vers nombre
  - Requiert que la table soit triée par ordre alphabétique
  - Accès en O(log(N))
    - Cf : recherche dichotomique dans un tableau trié
  - Insertion en O(log(N))
    - Cf : recherche de la position souhaitée puis insertion
    - Pb pour les gros lexiques (>100000 mots)

Type abstrait « lexique »

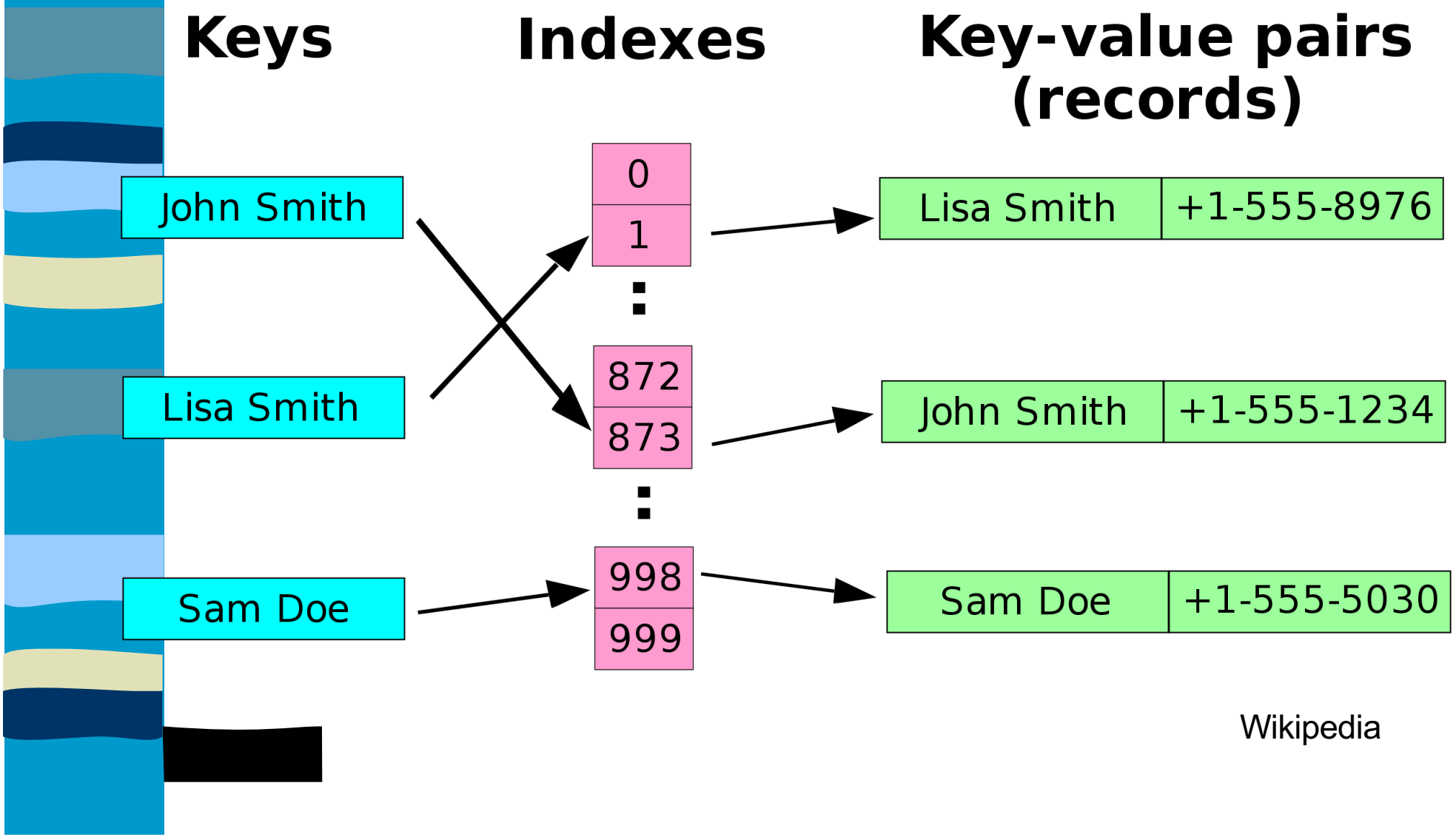
- Représentation
  - Listes et tableaux
  - Tables de hachage
  - Arbres
  - Automates d'états finis



Tables de hachage

- Structure de données qui permet une association clé-élément
- L'accès à un élément se fait en transformant la clé en une valeur de hachage
  - par l'intermédiaire d'une fonction de hachage
- Le hachage est un nombre qui permet la localisation des éléments dans le tableau
  - C’est l'index de l'élément dans le tableau
- La complexité dépend de la fonction de hachage mais est généralement faible (pour l’accès par valeur)

Tables de hachage



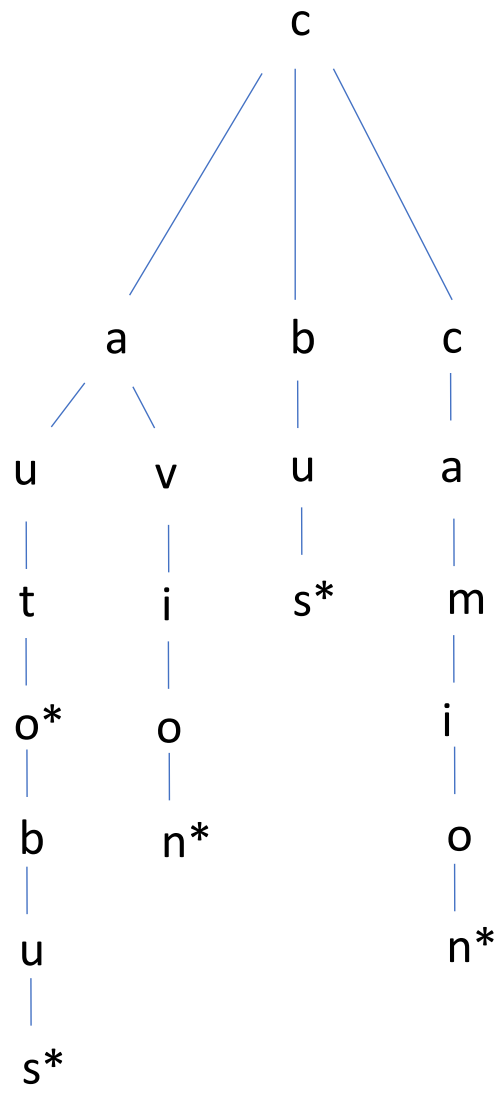
Wikipedia

Type abstrait « lexique »

- Représentation
  - Listes et tableaux
  - Tables de hachage
  - Arbres
  - Automates d'états finis

Arbres

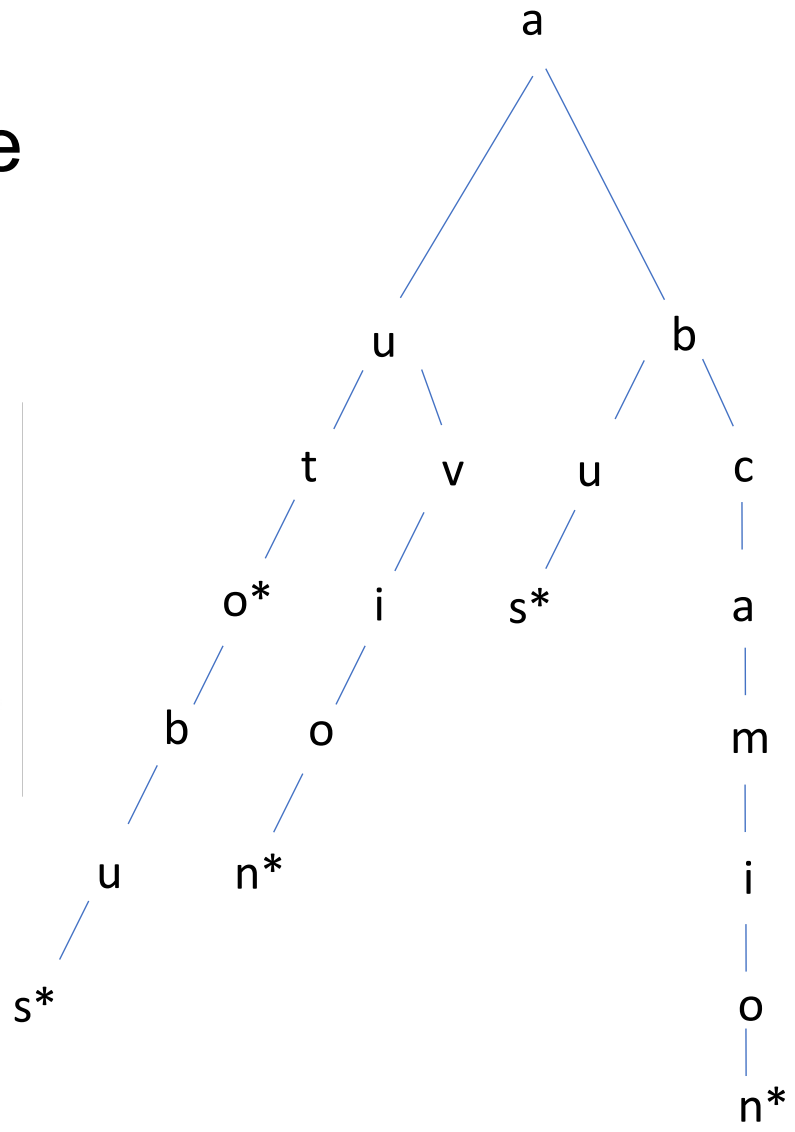
- Chaque chemin (de la racine à une feuille) représente un mot
- Optimisation en terme de taille
- Besoin de marquer la fin d'un mot



Arbres

- Tout arbre n-aire peut être transformé en arbre binaire

```
typedef struct ARBRE {
        char lettre;
        bool_t mot;
        struct ARBRE* fils;
        struct ARBRE* frere;
}arbre_t;
```

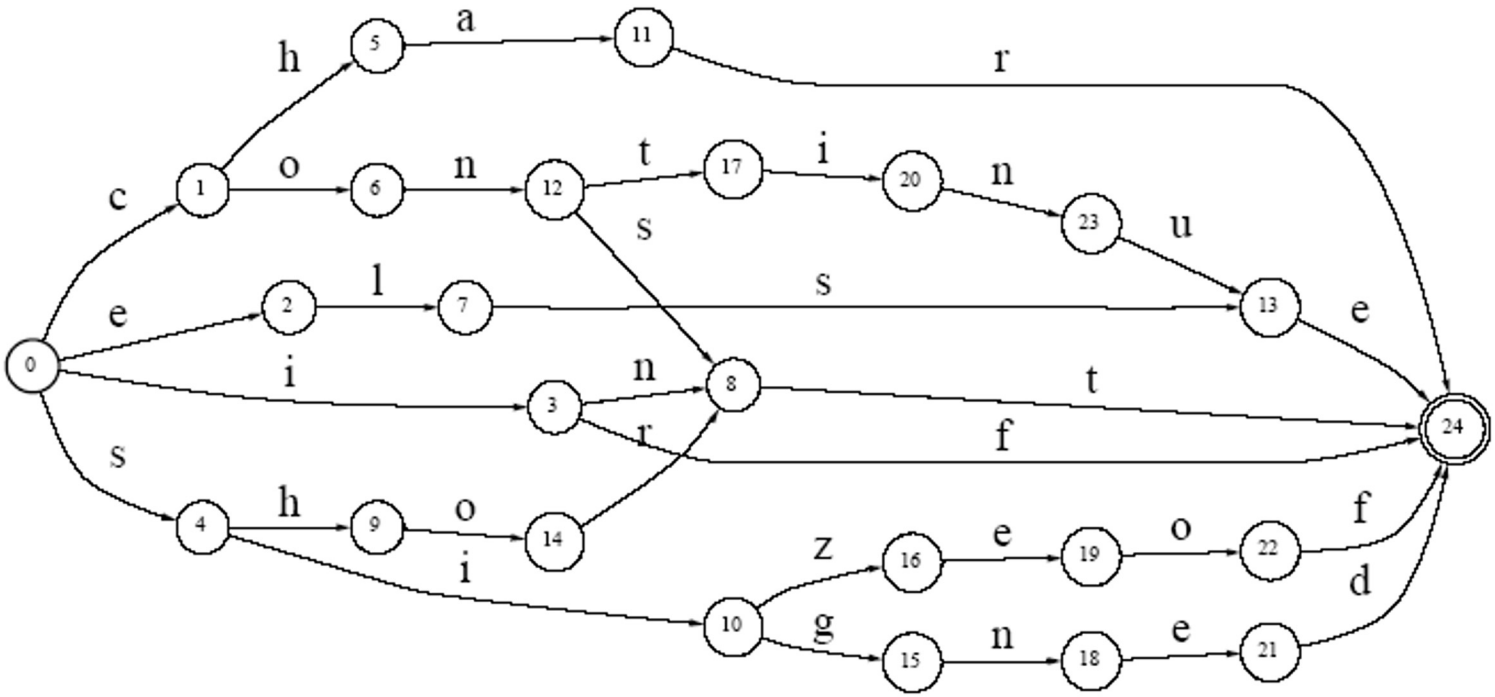


Type abstrait « lexique »

- Représentation
  - Listes et tableaux
  - Tables de hachage
  - Arbres
  - Automates d'états finis

Automates d'état finis

- Théoriquement la meilleure implémentation pour coder les formes de surface
  - Optimisation en terme de taille de stockage
- Cf expressions régulières en perl



Etiquetage de surface (syntaxique)

- *Part-of-speech (POS) tags*
- Etiquettes
  - Liste fermée
    - Pronoms, déterminants, prépositions, …
  - Liste ouverte
    - Noms, verbes, adjectifs, adverbes
  - Moins de 100 étiquettes possibles
- But : obtenir ces étiquettes automatiquement à partir d'une suite de mots
- Problème : ambiguïté
  - Ex: like
    - verbe: I **like** the class.
    - preposition: He is **like** me.

Exemples d'étiquettes

Penn POS Tag Set *(from University of Pennsylvania)*

Adjective:	JJ
Adverb:	RB
Cardinal Number:	CD
Determiner:	DT
Preposition:	IN
Coordinating Conjunction	CC
Subordinating Conjunction:	IN
Singular Noun:	NN
Plural Noun:	NNS
Personal Pronoun:	PP
Proper Noun:	NP
Verb base form:	VB
Modal verb:	MD
Verb (3sg Pres):	VBZ
Wh-determiner:	WDT
Wh-pronoun:	WP

Ambiguïté

- Ex: « *Time flies like an arrow* »
- Interpretation-1
  - "One cannot find enough time"
  - "flies" est le verbe principal et "Time" est le sujet
  - "Time flies like an arrow" N V P Art N
- Interpretation-2
  - "There is a race amongst flies, and one is asked to take their timing"
  - "Time" est le verbe principal
  - "Time flies like an arrow" V N P Art N
- Interpretation-3
  - "There is a special kind of flies called Time Flies, and they like an arrow"
  - "like" est le verbe principal
  - "Time flies like an arrow" N N V Art N

Étiquetage « POS tagging »

- Connaissances utiles
  - Les mots eux-mêmes
    - Certains mots ne peuvent être que des noms
      - Ex : *arrow*
    - D'autres sont ambigüs
      - Ex : *like, flies*
    - Les probabilités peuvent être utiles (certaines étiquettes plus probables que d'autres)
  - Contexte
    - On trouve très rarement deux verbes l'un à la suite de l'autre
    - Un déterminant est presque toujours suivi par un adjectif ou un nom

Règle de Bayes

- On veut trouver la meilleure séquence d'étiquettes T pour une phrase S

$$\text{argmax}_T \; p(T|S)$$

- La règle de Bayes nous donne

$$p(T|S) = \frac{p(S|T)\; p(T)}{p(S)}$$

- Et finalement

$$\text{argmax}_T \; p(T|S) = \text{argmax}_T \; p(S|T)\; p(T)$$

Décomposition du modèle

- p(S/T) peut être décomposé en

$$p(S|T) = \prod_i p(w_i|t_i)$$

- p(T) est le modèle de langage d'étiquettes ; nous pouvons utiliser un modèle de langage n-gramme

$$p(T) = p(t_1)\ p(t_2|t_1)\ p(t_3|t_1,t_2)...p(t_n|t_{n-2},t_{n-1})$$

- Nous pouvons estimer p(S/T) et p(T) par maximum de vraisemblance

Apprendre les probabilités

- From [Allen, 95]

Category	Count at i	Pair	Count at i,i+1	Bigram	Estimate
Φ	300	Φ,ART	213	P(ART|Φ)	.71
Φ	300	Φ,N	87	P(N|Φ)	.29
ART	558	ART,N	558	P(N|ART)	1
N	833	N,V	358	P(V|N)	.43
N	833	N,N	108	P(N|N)	.13
N	833	N,P	366	P(P|N)	.44
V	300	V,N	75	P(N|V)	.35
V	300	V,ART	194	P(ART|V)	.65
P	307	P,ART	226	P(ART|P)	.74
P	307	P,N	81	P(N|P)	.26

Apprendre les probabilités

- From [Allen, 95]

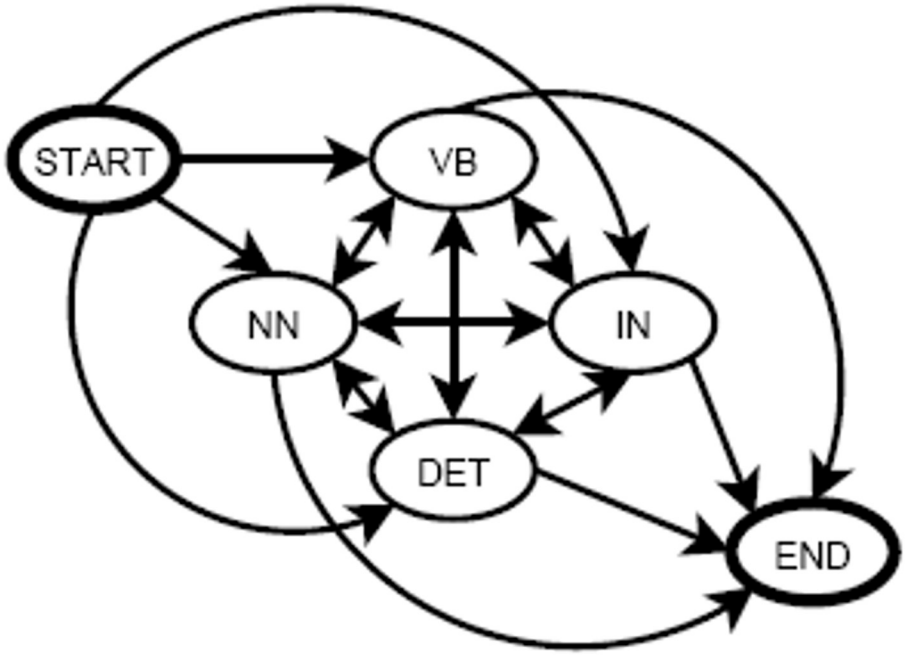
P(the|ART)	.54	P(a|ART)	.360
P(flies|N)	.025	P(a|N)	.001
P(flies|V)	.076	P(flower|N)	.063
P(like|V)	.1	P(flower|V)	.05
P(like|P)	.068	P(birds|N)	.076
P(like|N)	.012		

Modèle de Markov Caché

- Le modèle décrit précédemment est un Modèle de Markov Caché (*HMM ou Hidden Markov Model*)
- Eléments d'un HMM
  - Un ensemble d'états (ici : les étiquettes)
  - Un alphabet de sortie (ici : les mots)
  - Un état initial (ici : le début de la phrase à analyser)
  - Des probabilités de transition entre états (ici : $p(tn|tn-2, tn-1)$)
  - Des probabilités d'émission (ici : $p(wi|ti)$)

Représentation graphique

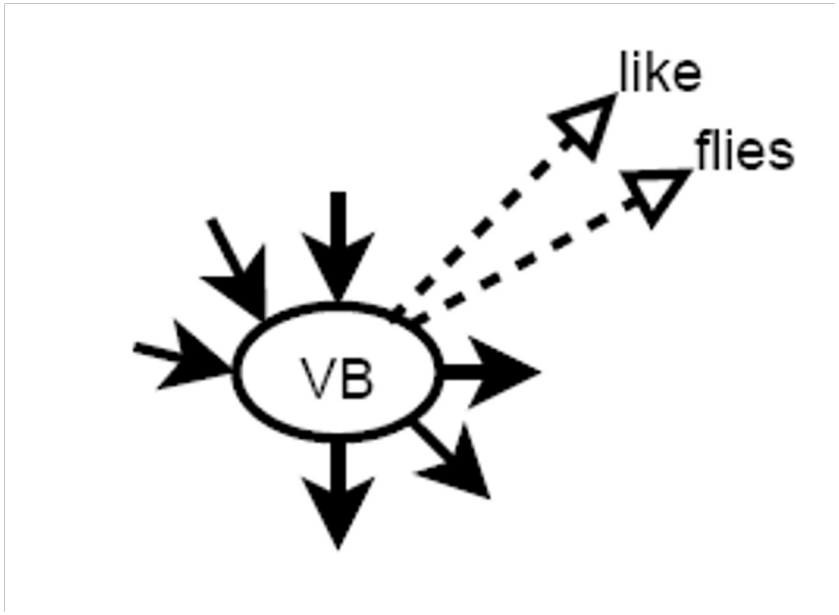
- En étiquetant une phrase, on parcours le graphe suivant



- Probabilités de transition entre états (p(tn|tn−1)

Représentation graphique

- Chaque état émet un mot



- probabilités d'émission (p(wi|ti)

Décodage

- Trouver la meilleure séquence d'étiquettes T* étant donnée une phrase S=w1,w2,…,wN
- Algorithme naïf : calculer, pour toutes les séquences T possibles

$$p(S|T)\ p(T) = \prod_i p(w_i|t_i)\ p(t_i|t_{i-2}, t_{i-1})$$

- Et choisir la séquence T la plus probable (argmax)
- Si on a en moyenne *c* étiquettes possible pour chacun des *n* mots de la phrase à étiqueter, on aura au total *c^n* séquences possibles
  - Problématique pour des longues phrases (*n* grand)

Illustration graphique

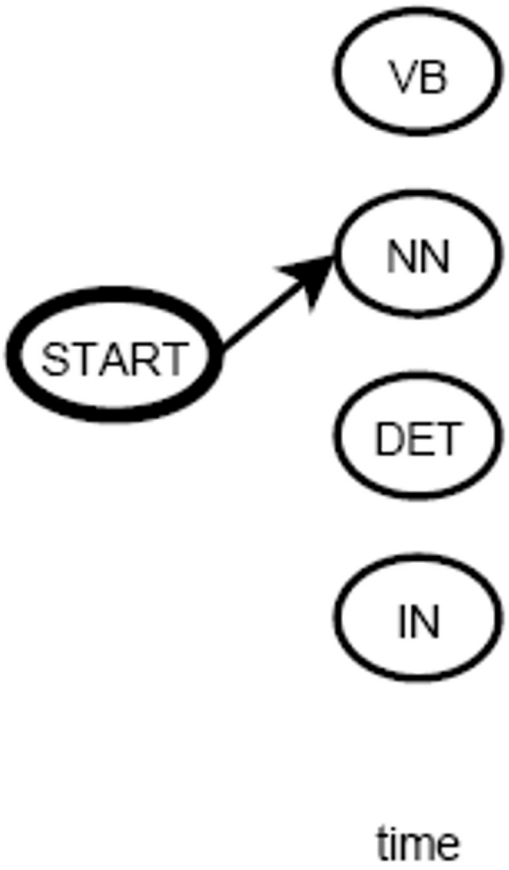


Illustration graphique

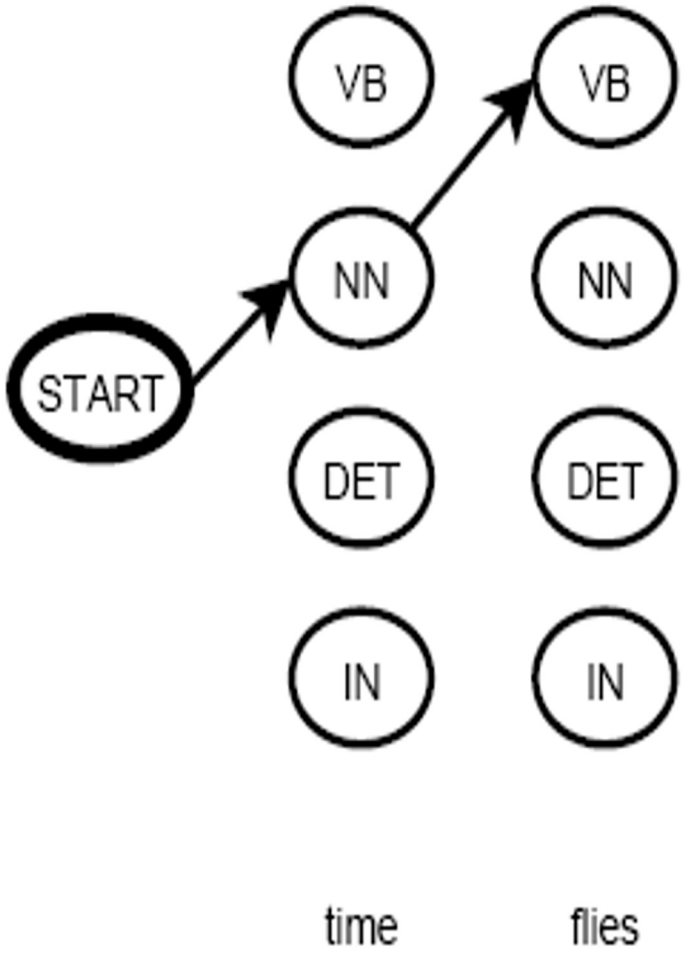
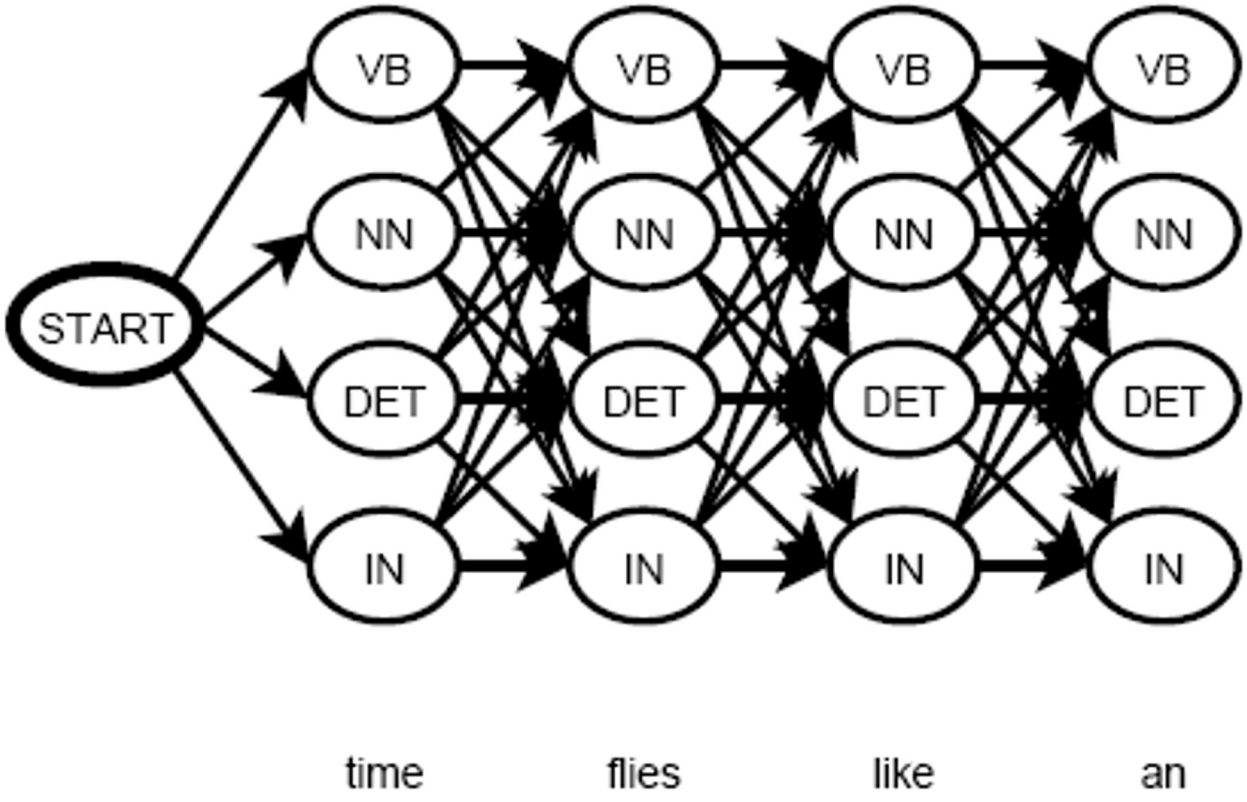


Illustration graphique

Beaucoup de chemins possibles !!!



Algorithme de Viterbi

- Complexité : nc^2 au lieu de c^n

Algorithme de Viterbi

- La transition d'un état à un autre ne dépend que de l'état courant
  - On peut conserver, pour chaque état j, à l'étape s, le chemin optimal menant à cet état et le coût associé $\psi_j(s)$ $\delta_j(s)$
- Pour chaque nouvelle étape, on calcule

$$\delta_j(s+1) = \max_{1 \leq i \leq N} \delta_i(s)\, p(t_i|t_j)\, p(w_s|t_j)$$

$$\psi_j(s+1) = \operatorname{argmax}_{1 \leq i \leq N} \delta_i(s)\, p(t_i|t_j)\, p(w_s|t_j)$$

- Le meilleur état final est alors $\operatorname{argmax}_{1 \leq i \leq N} \delta_i(S+1)$
- Le chemin optimal est obtenu ensuite par retour en arrièr à partir de cet état final

Remplacer p(ti/tj) par p(tj/ti)

Pseudo-code

Input: word sequence $w_1, w_2, \ldots, w_T$

Output: Tag sequence $C = C_1, C_2, \ldots, C_T$

Given:

1. Lexical categories, $L = L_1, L_2, \ldots, L_N$
2. Lexical probabilities, $P(w_i|L_j)$
3. Bigram probabilities, $P(L_i|L_j)$

Data Structures:

1. Seqscore: an NXT array
2. Backptr: another NXT array

Pseudo-code

Initialization

For $i=1$ to N do

$Seqscore[i,1] = P(w_1|L_i) * P[L_i|\Phi]$

$Backptr[I,1]=0$

Iteration

For $t= 2$ to T

For $i=1$ to N

$Seqscore[i,t]=MAX_{j=1,N}(Seqscore[j,t-1] * P(L_i|L_j) * P(w_t|L_i)$

$Backptr[i,t]=$index of j that gave the max score

Sequence identification

$C[T]=i$ that maximizes $Seqscore[i,T]$

For $i=T-1$ to 1 do

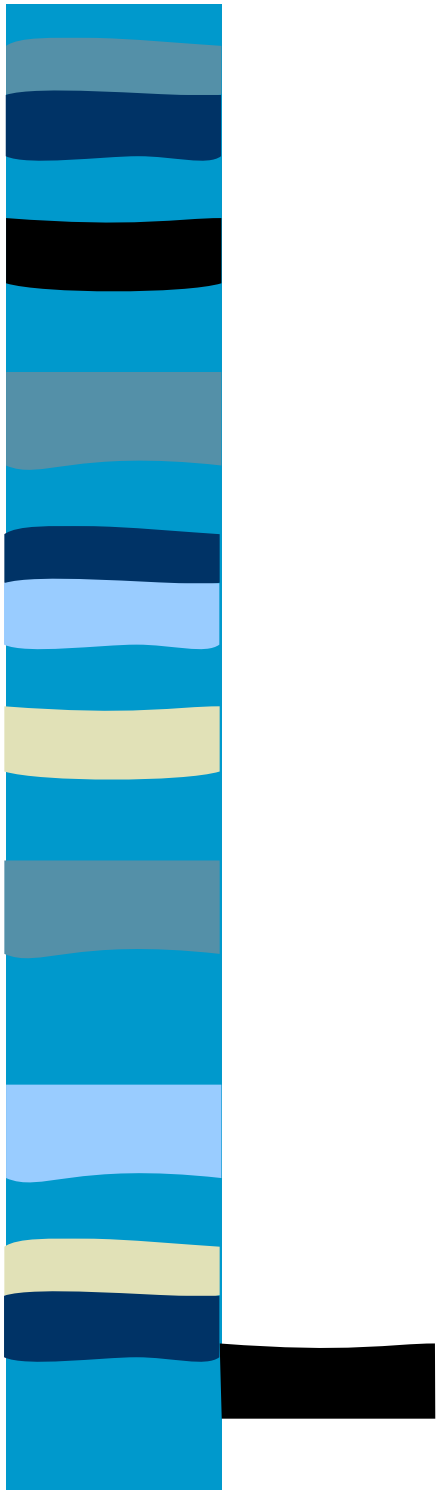
$C[i]=backptr(C[i+1],i+1)$

Autres problèmes d'étiquetage

- Peuvent être résolus avec des algorithmes similaires
- Détection d'entités nommées
  - Noms de personnes, d'organisations (ex: Tony Blair, World Trade Center)
- Restauration d'accents
- Restauration de la casse
  - Cf : http://www-clips.imag.fr/geod/User/laurent.besacier/NEW-TPs/TP-CL/tp4.html

Autres algorithmes

- Beam-search
- A*
  - On en reparlera dans le cours sur la « reconnaissance automatique de la parole »



MERCI !

Table de Hachage

- Résolution des collisions
  - par chainage
  - par adressage ouvert

Table de Hachage

- Résolution des collisions
  - par chainage

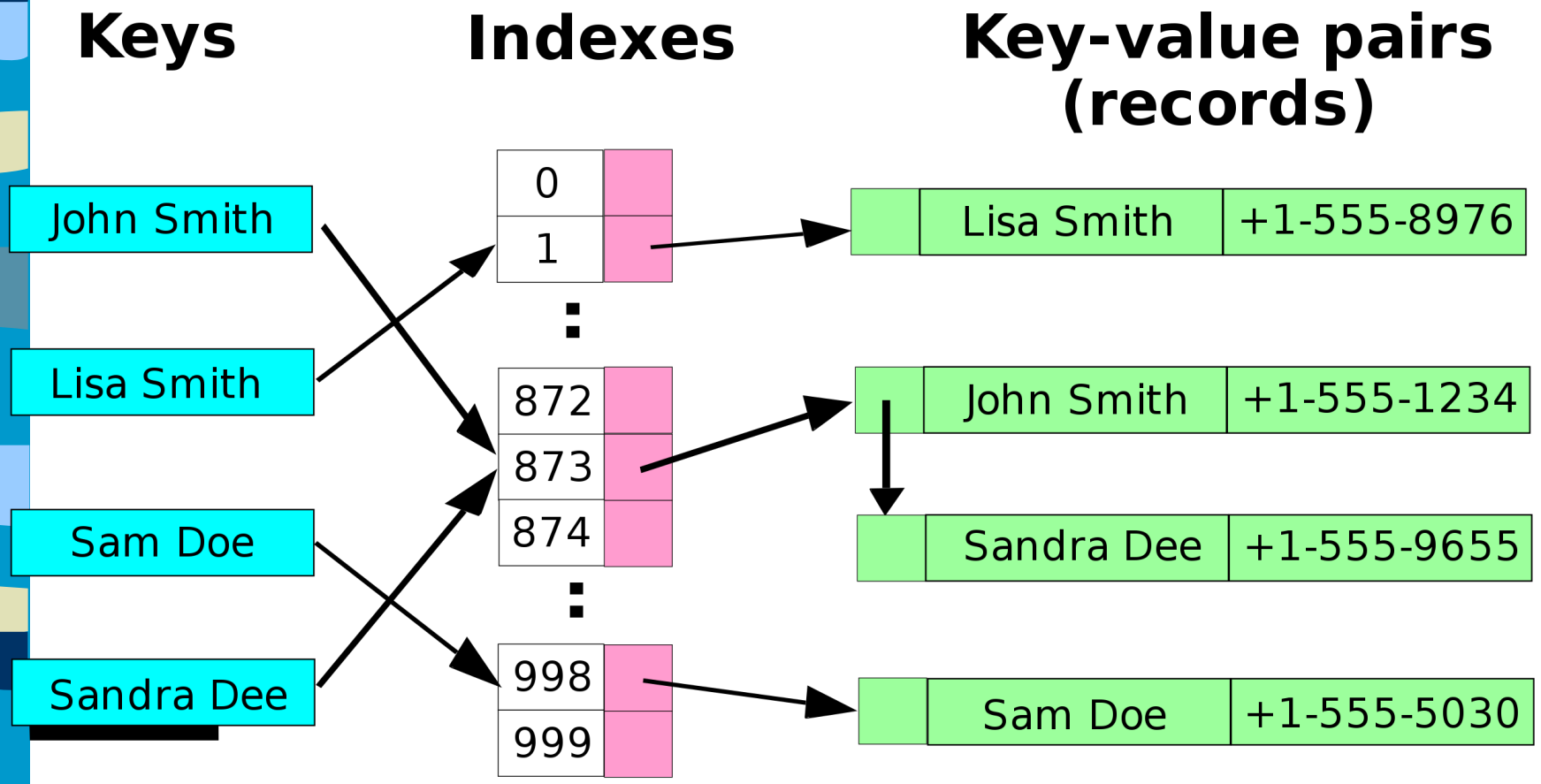


Table de Hachage

- Résolution des collisions
  - par adressage ouvert