Une approche formelle au problème de l'ancrage de symboles

Alexandre Blondin Massé

Département d'informatique
Université du Québec à Montréal

6 novembre 2014
Séminaire du doctorat en informatique cognitive
Département d'informatique
Université du Québec à Montréal

Table des matières

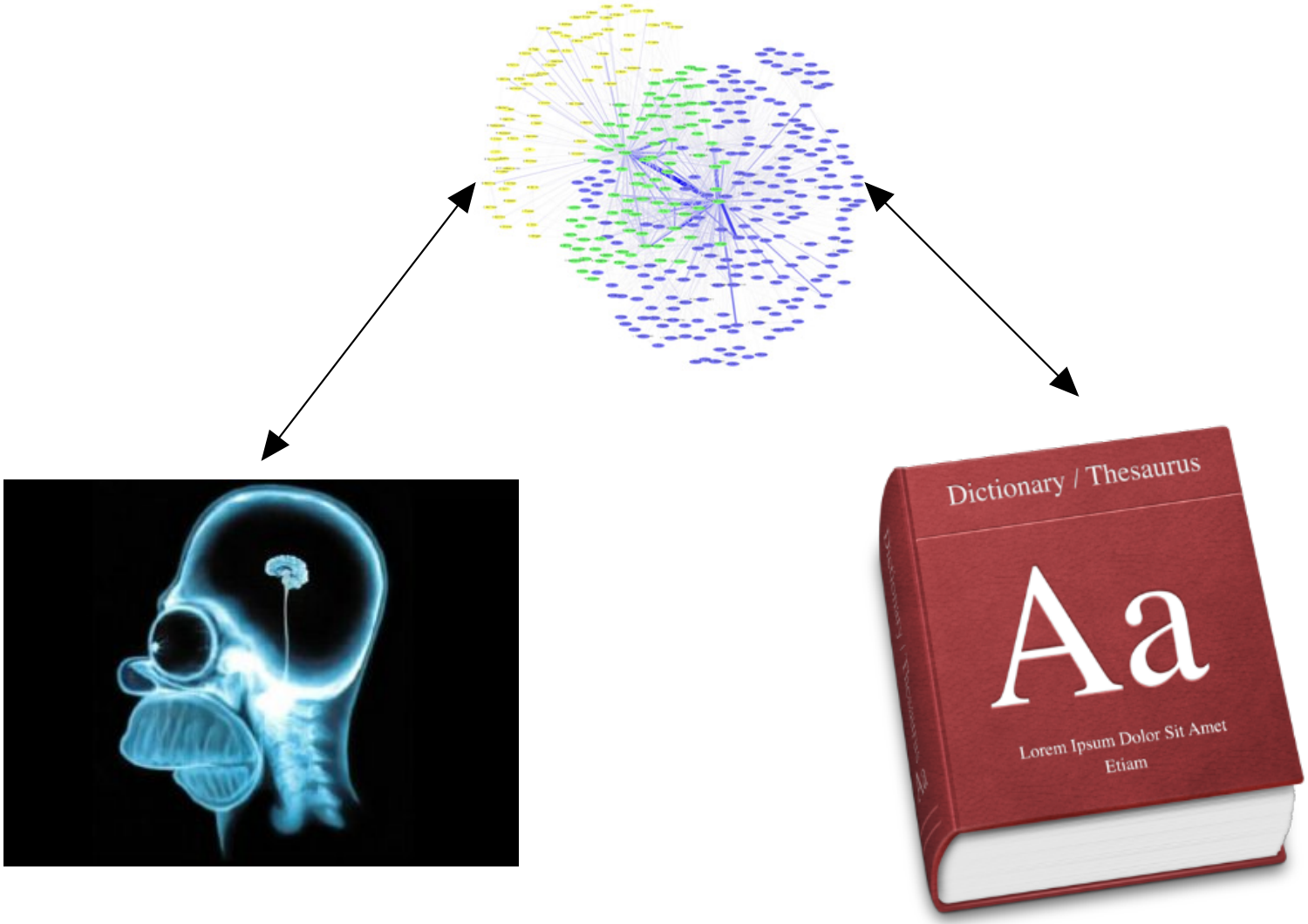
Table des matières

Apprentissage

Essentiellement **deux façons** d'apprendre des nouveaux mots :

- **Expérience sensorimotrice** : Par exemple, une **pomme** est
  - rouge;
  - sucrée;
  - ronde;
  - etc.
- **Instruction verbale** : Quelqu'un **décrit**, verbalement ou à l'écrit, ce qu'est une **pomme**.

Lexique mental



Dictionnaires

C'est une liste d'**associations** :

pomme	nom	fruit du pommier
pommier	nom	arbre dont le fruit est la pomme
arbre	nom	grand végétal dont la tige, appelée...
fruit_1	nom	ensemble des organes végétaux...
fruit_2	nom	fruit comestible
fruit_3	nom	résultat
organe_1	nom	partie d'un corps organisé...
organe_2	nom	mécanisme
organe_3	nom	publication périodique

Dictionnaires (suite)

Dans plusieurs situations, les mots sont **polysémiques** et/ou appartiennent à différentes **catégories grammaticales** :

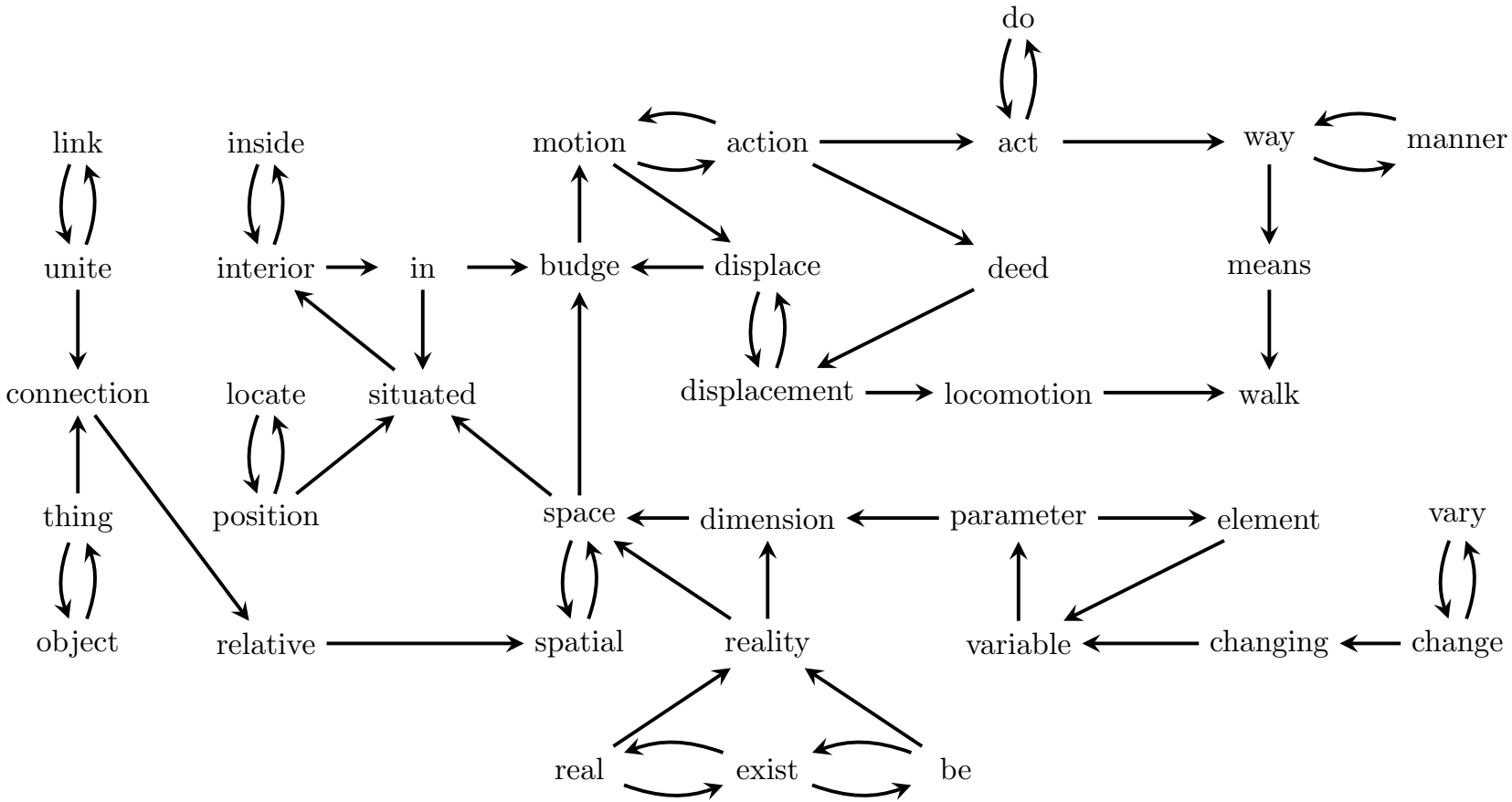
fait_1	nom	action
fait_2	nom	événement
fait_3	nom	réalité
fait_1	adjectif	fabriqué
fait_2	adjectif	qui a telle forme
fait_3	adjectif	arrivé à maturité
fait	participe	participe passé du verbe faire

D'un dictionnaire à un graphe (orienté)

Prétraitement :

- Suppression des mots **non définis** et des mots **fonctionnels**;
- Étiquetage **syntaxique** (*POS tagging*, POS = part-of-speech);
- **Lemmatisation** (par exemple eurent $\rightarrow$ avoir);
- **Désambiguïsation** sémantique (heuristique : première définition).

Graphes


do
link
inside
motion
action
act
way
manner
unite
interior
in
budge
displace
deed
means
connection
locate
situated
displacement
locomotion
walk
thing
position
space
dimension
parameter
element
vary
object
relative
spatial
reality
variable
changing
change
real
exist
be

Différents types de dictionnaires

- De **grands** dictionnaires :
  - **CIDE** Cambridge Int. Dict. of Eng. (48 000 mots);
  - **LDOCE** Longman Dict. of Cont. Eng. (70 000 mots);
  - **WN** WordNet (132 000 mots);
  - **MWC** Merriam-Webster's Coll. Dict. (250 000 mots).
- Dictionnaires obtenus dans un **jeu** :
  - Les **participants** doivent **construire** des dictionnaires;
  - Le nombre de mots varie entre 37 et 300.

Le jeu du dictionnaire

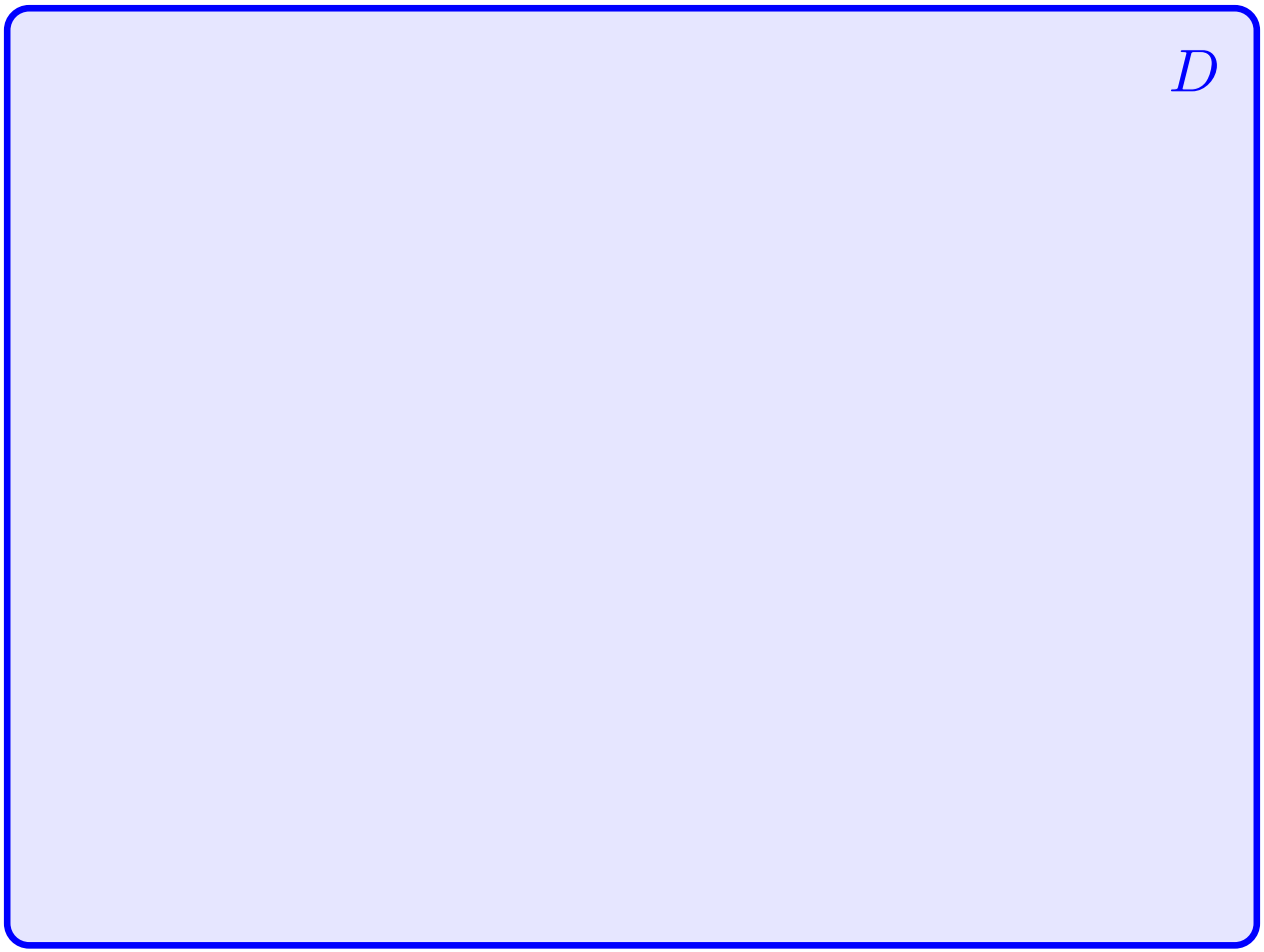
- Ancienne version :
  http://prodico.psycho.uqam.ca:8080/webDictGame3
- Nouvelle version (phase **alpha**) : disponible dans quelques semaines.

Vocabulaire de contrôle

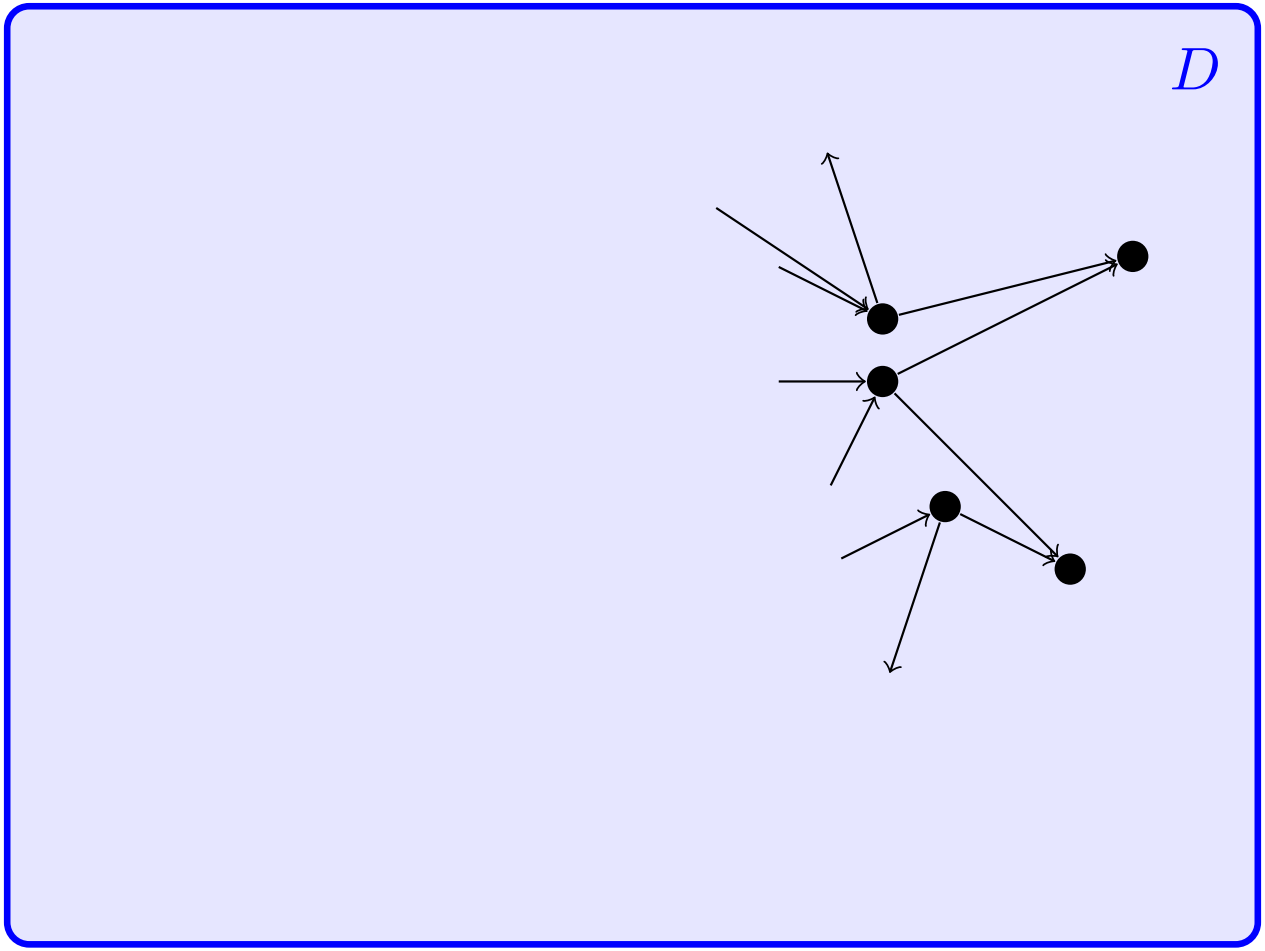
- Le **vocabulaire** d'un dictionnaire est l'ensemble des mots qui sont **utilisés** dans les définitions;
- Deux des dictionnaires, **LDOCE** et **CIDE**, ont été **construits** de façon à avoir un **vocabulaire restreint**;
- Cet ensemble de mots est appelé **vocabulaire de contrôle** était supposé avoir une taille d'environ **2 000 mots**;
- C'est cette **idée de départ** qui nous a amenés à étudier l'**anatomie** des dictionnaires numériques.

Table des matières

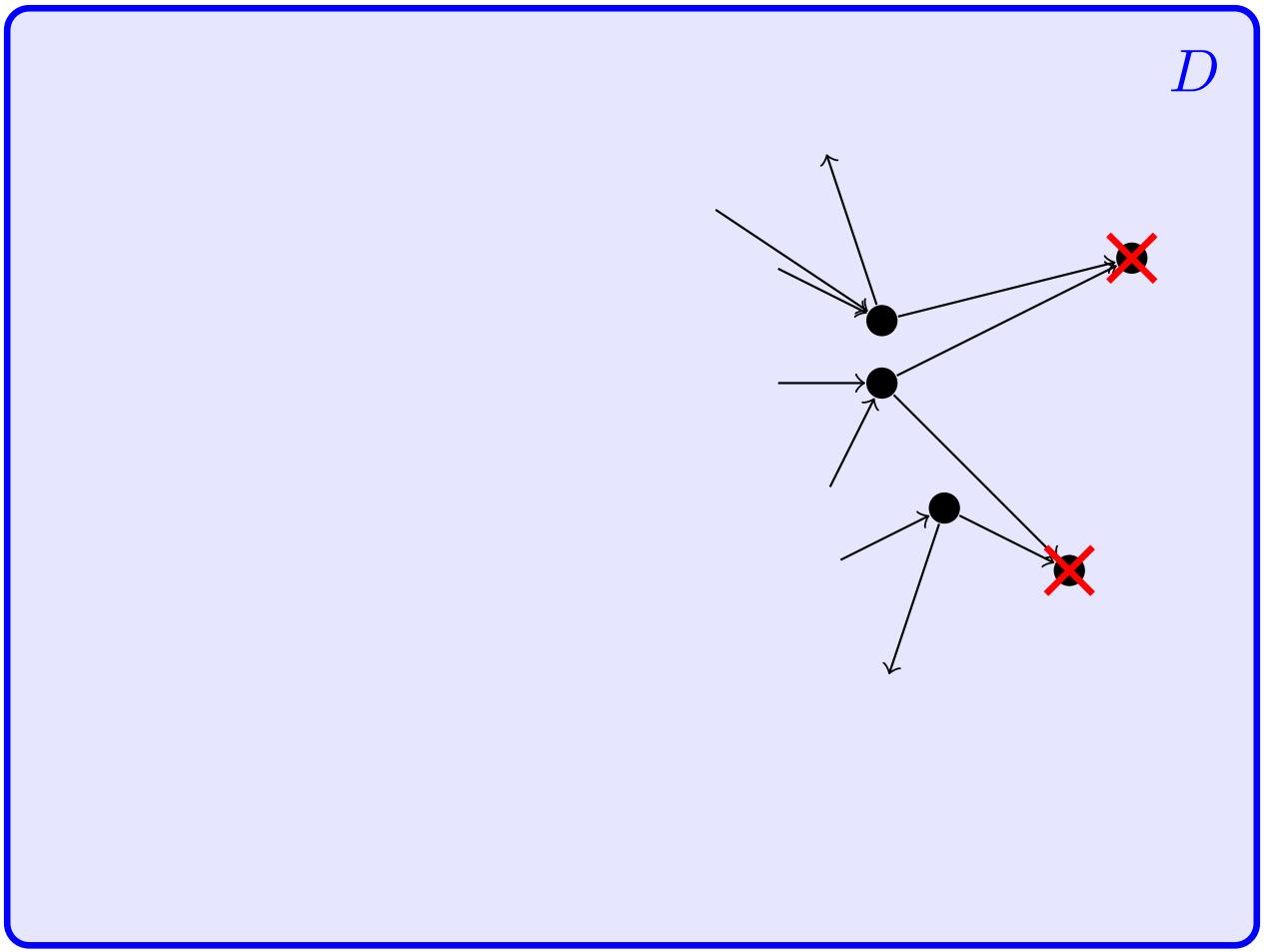
Anatomie d'un dictionnaire



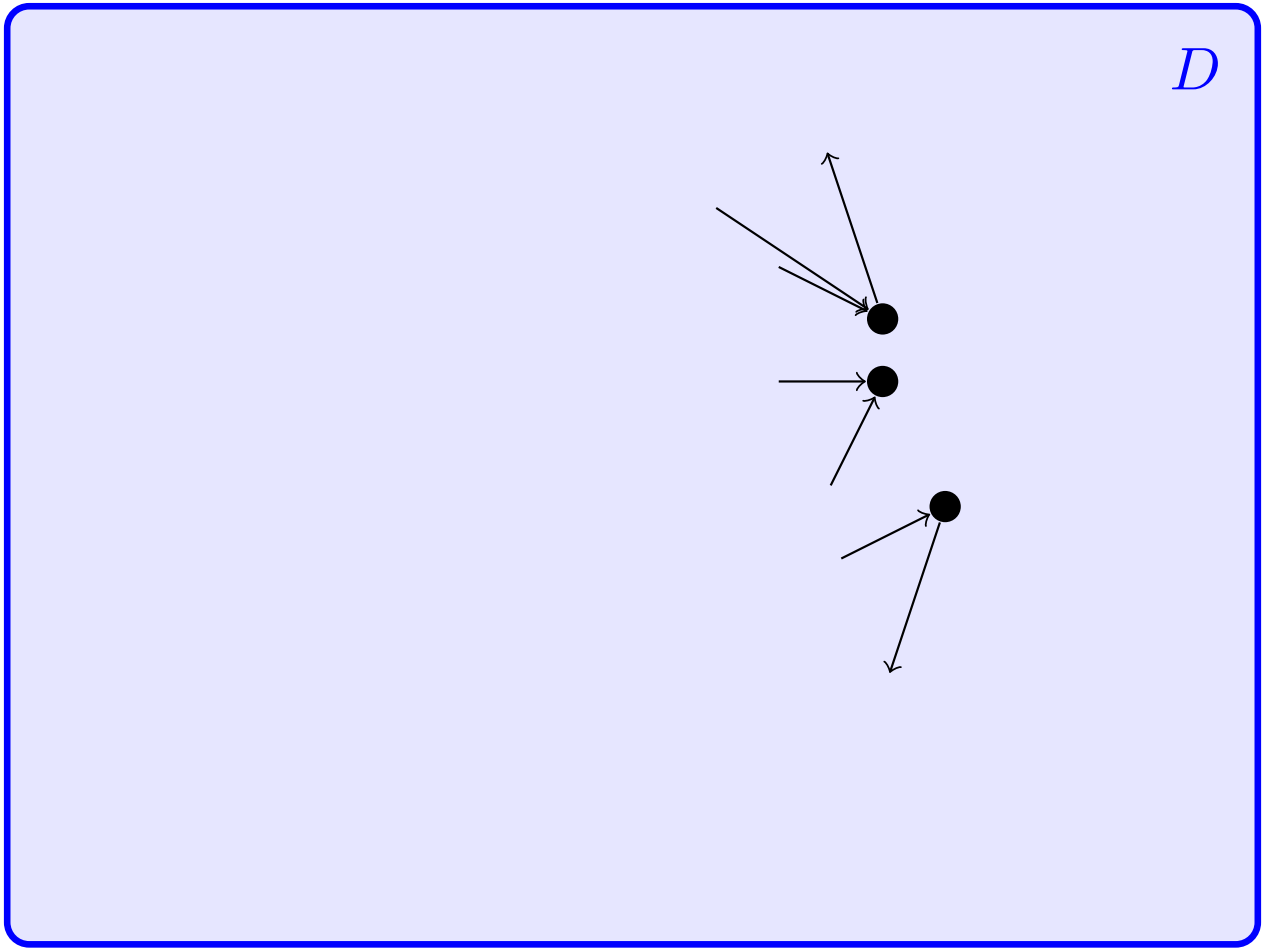
Anatomie d'un dictionnaire



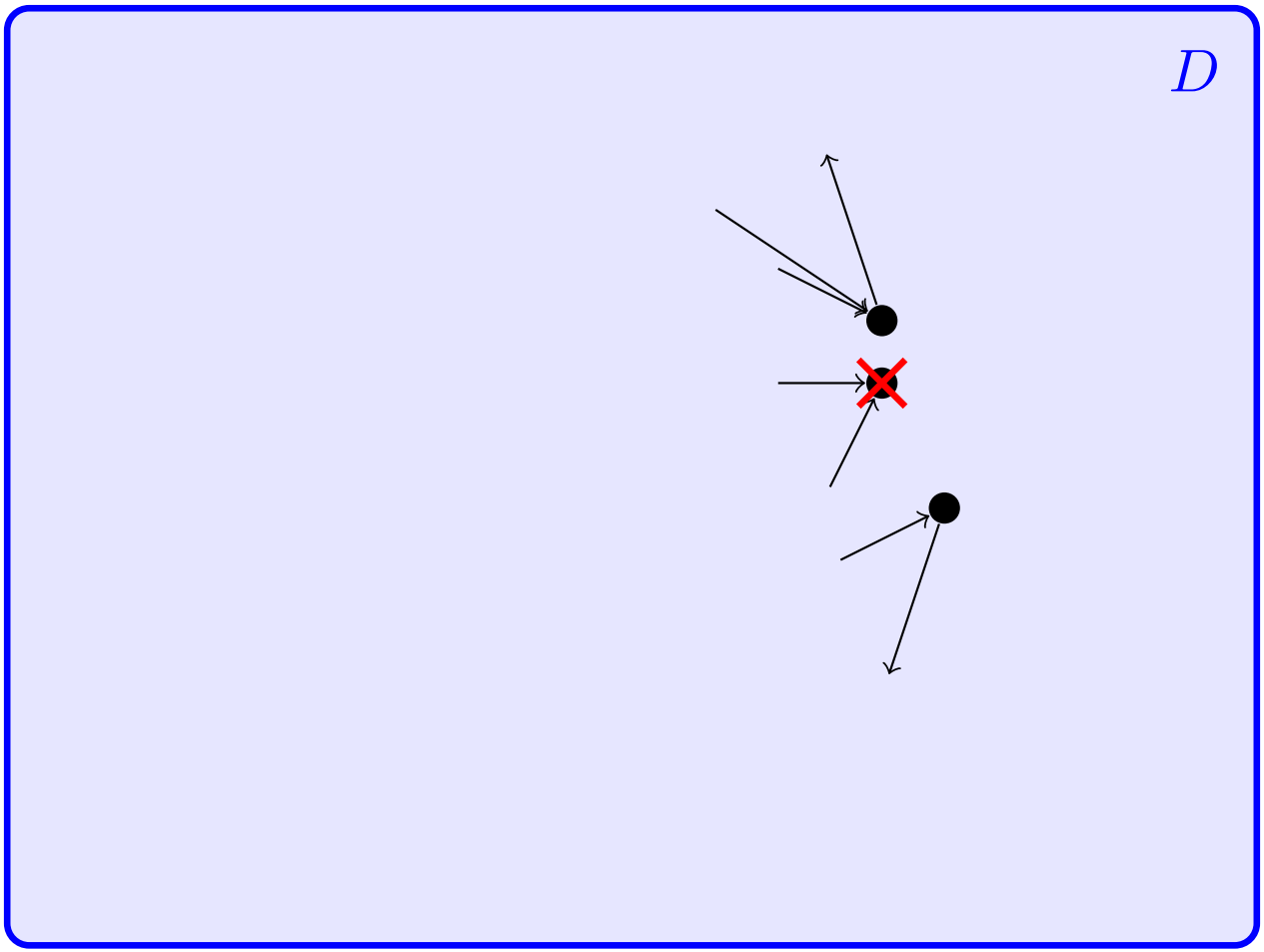
Anatomie d'un dictionnaire



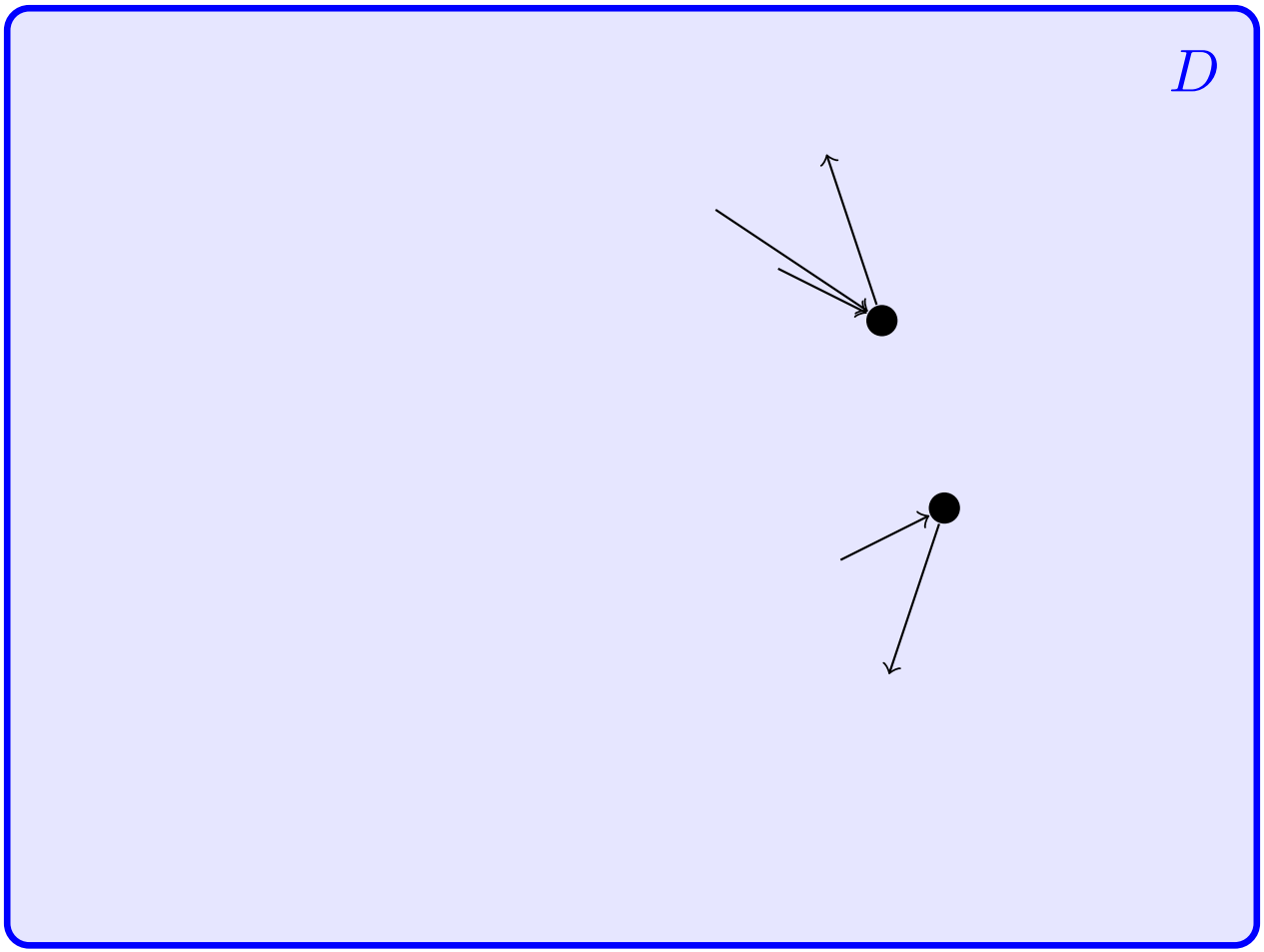
Anatomie d'un dictionnaire



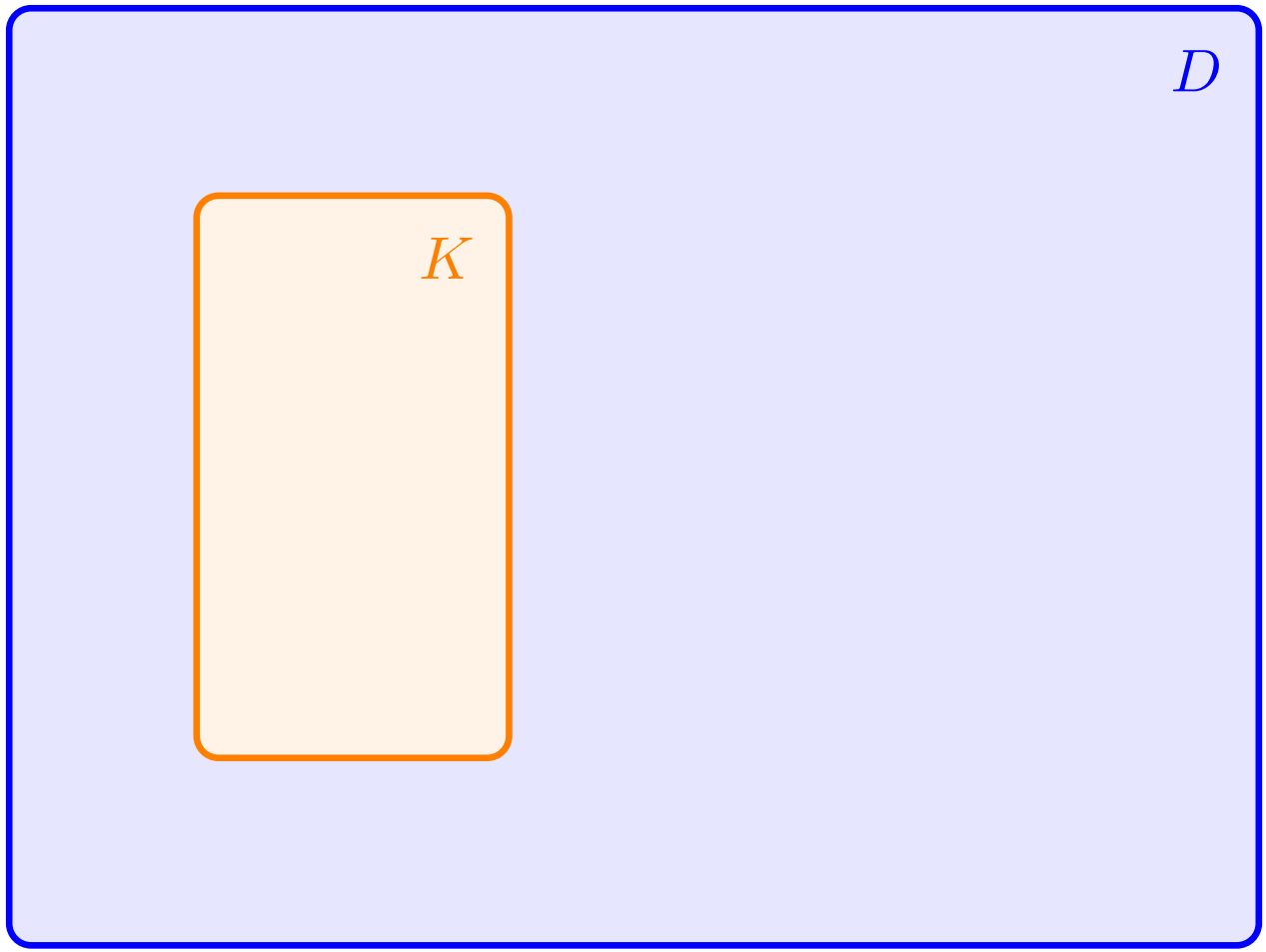
Anatomie d'un dictionnaire



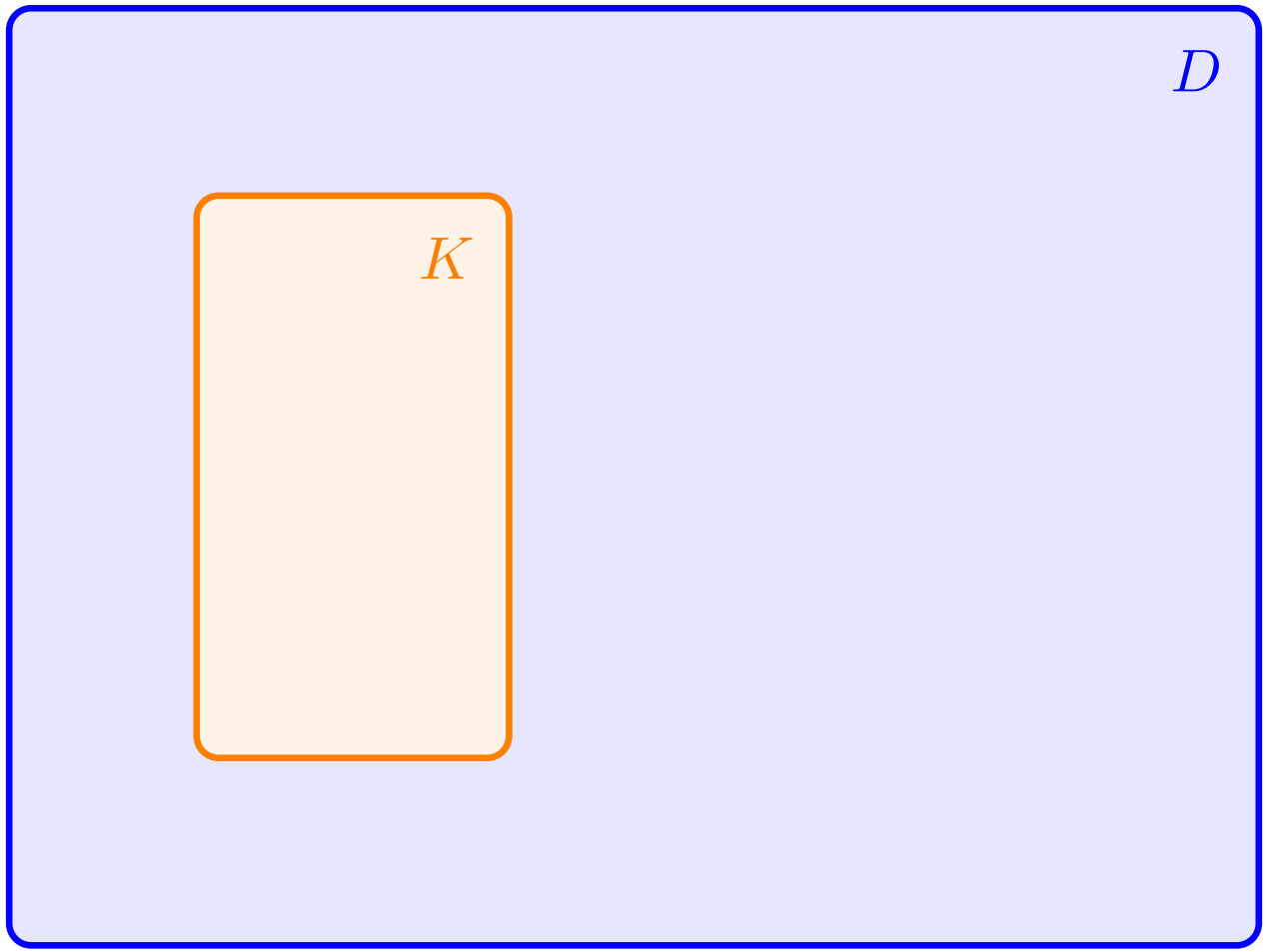
Anatomie d'un dictionnaire



Anatomie d'un dictionnaire



Anatomie d'un dictionnaire

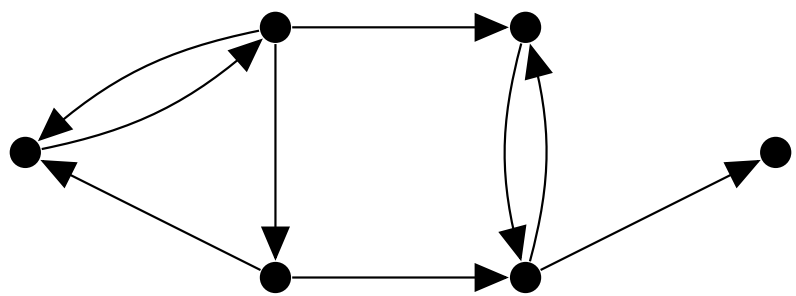


Pour les **grands dictionnaires** la taille est réduite de plus de 90%.

Composantes fortement connexes

Soit $G = (V, E)$ un **graphe orienté**.

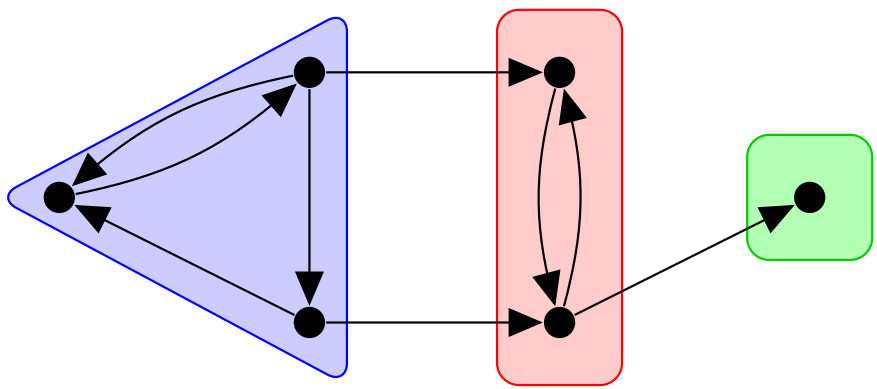
- On écrit $u \to v$ si et seulement s'il existe un **chemin** de u vers v;
- De la même façon $u \leftrightarrow v$ si et seulement si $u \to v$ et $v \to u$;
- La relation $\leftrightarrow$ est une **relation d'équivalence**;
- Les **classes d'équivalence** de la relation $\leftrightarrow$ sont appelés **composantes fortement connexes**.



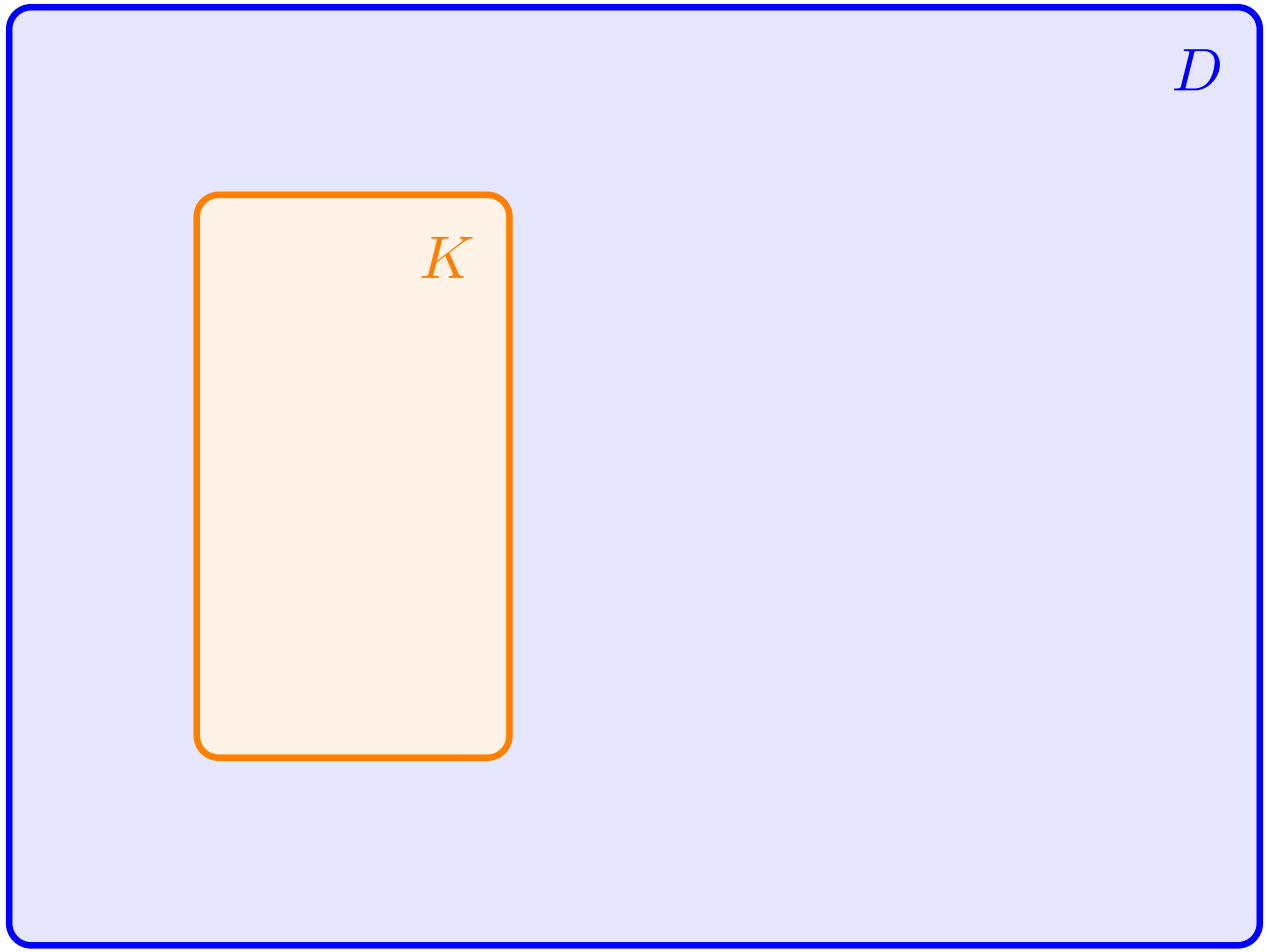
Composantes fortement connexes

Soit $G = (V, E)$ un **graphe orienté**.

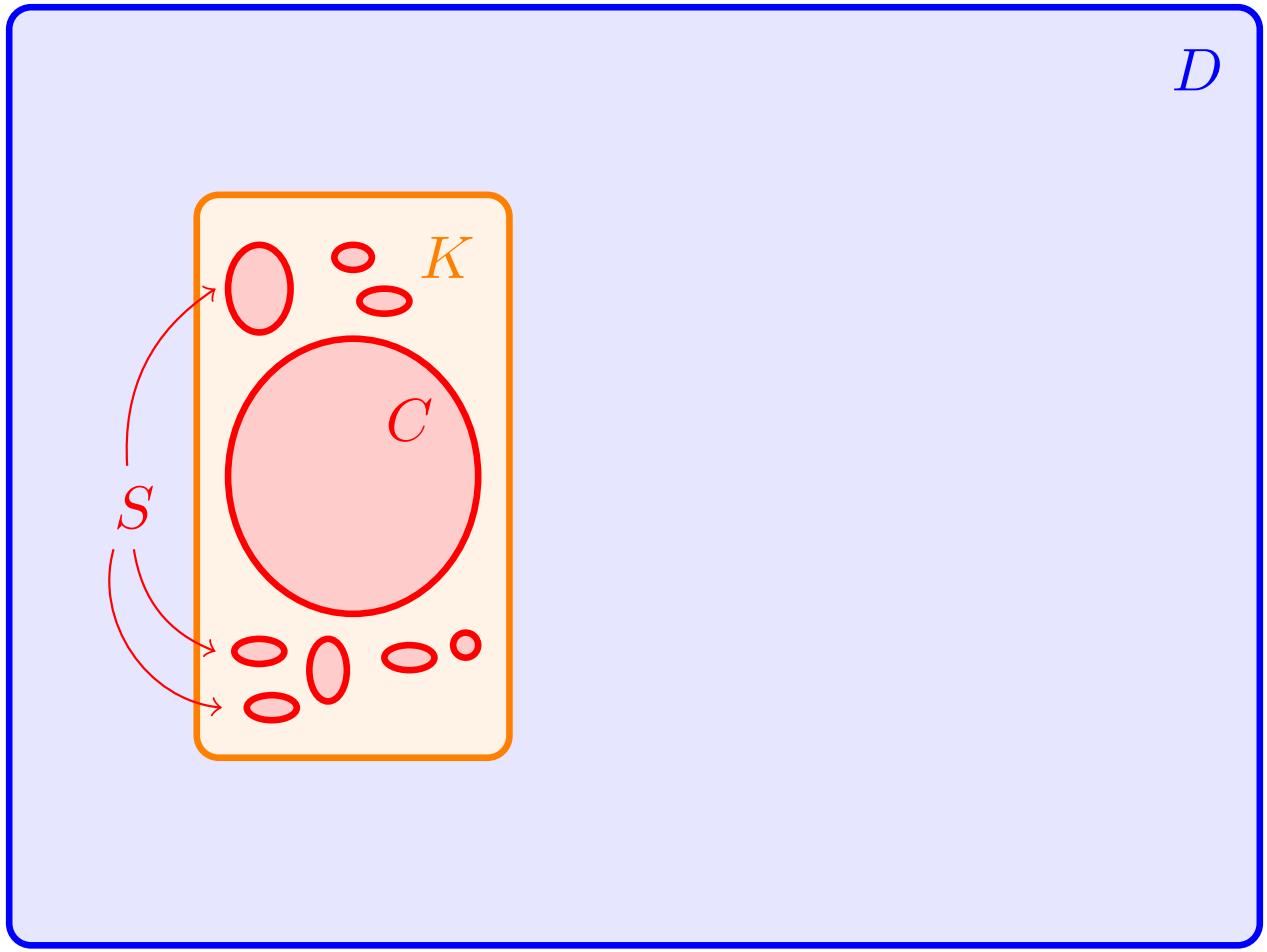
- On écrit $u \to v$ si et seulement s'il existe un **chemin** de u vers v;
- De la même façon $u \leftrightarrow v$ si et seulement si $u \to v$ et $v \to u$;
- La relation $\leftrightarrow$ est une **relation d'équivalence**;
- Les **classes d'équivalence** de la relation $\leftrightarrow$ sont appelés **composantes fortement connexes**.



Anatomie d'un dictionnaire (suite)

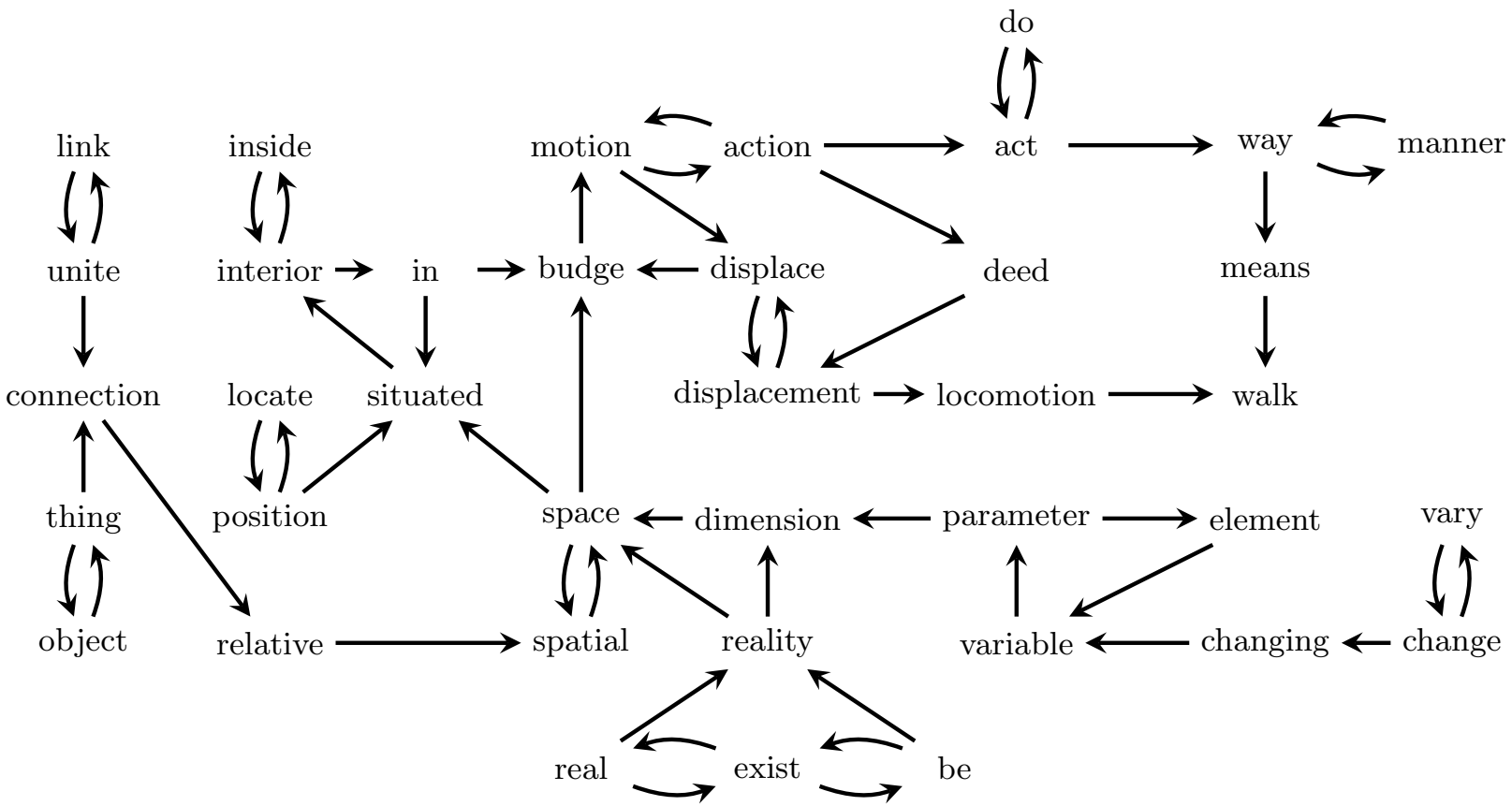


Anatomie d'un dictionnaire (suite)



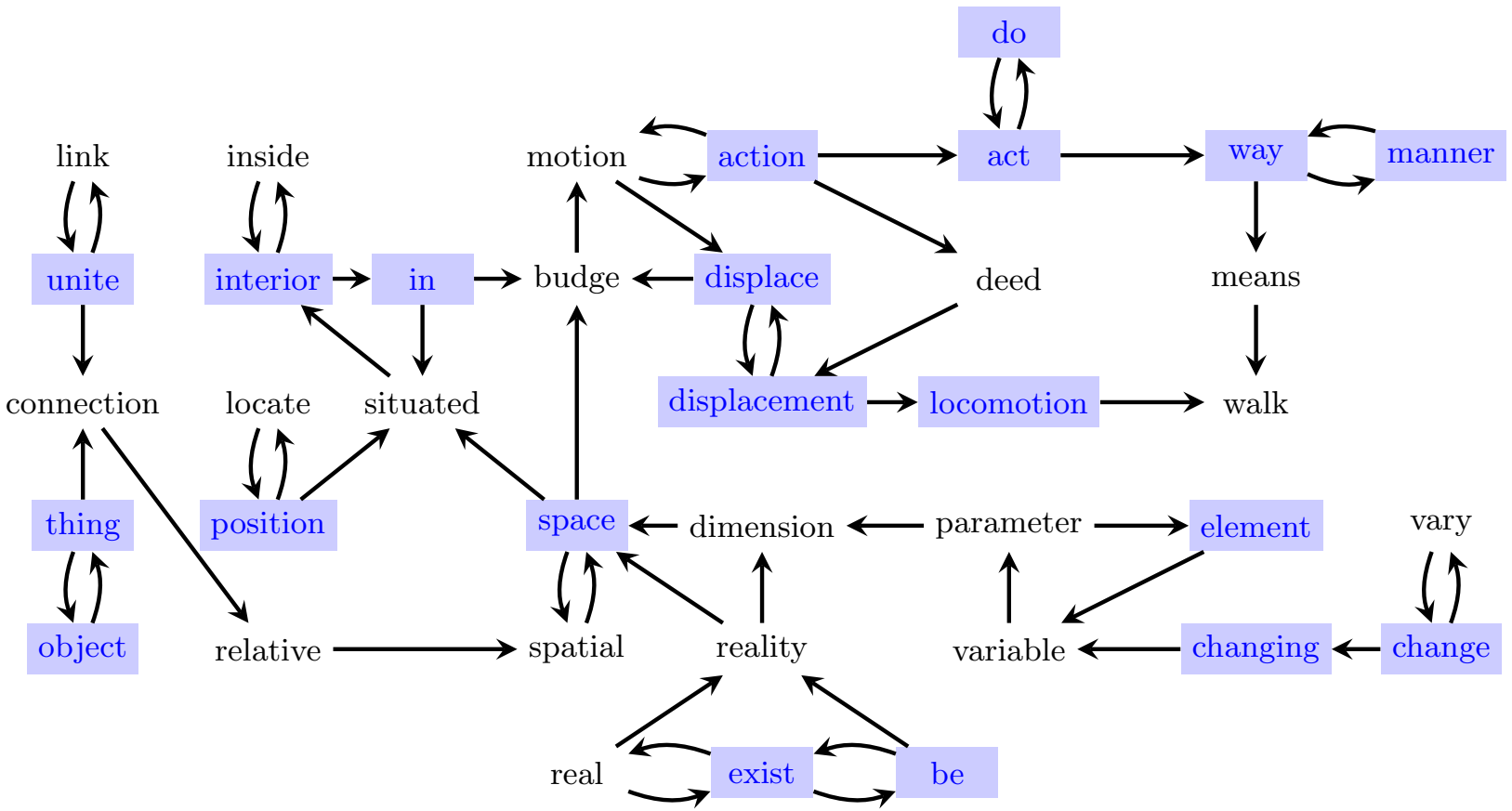
Ensembles d'ancrage

- Un ensemble de sommets U est appelé **ensemble d'ancrage** s'il nous permet d'apprendre **tous les mots** du graphe:



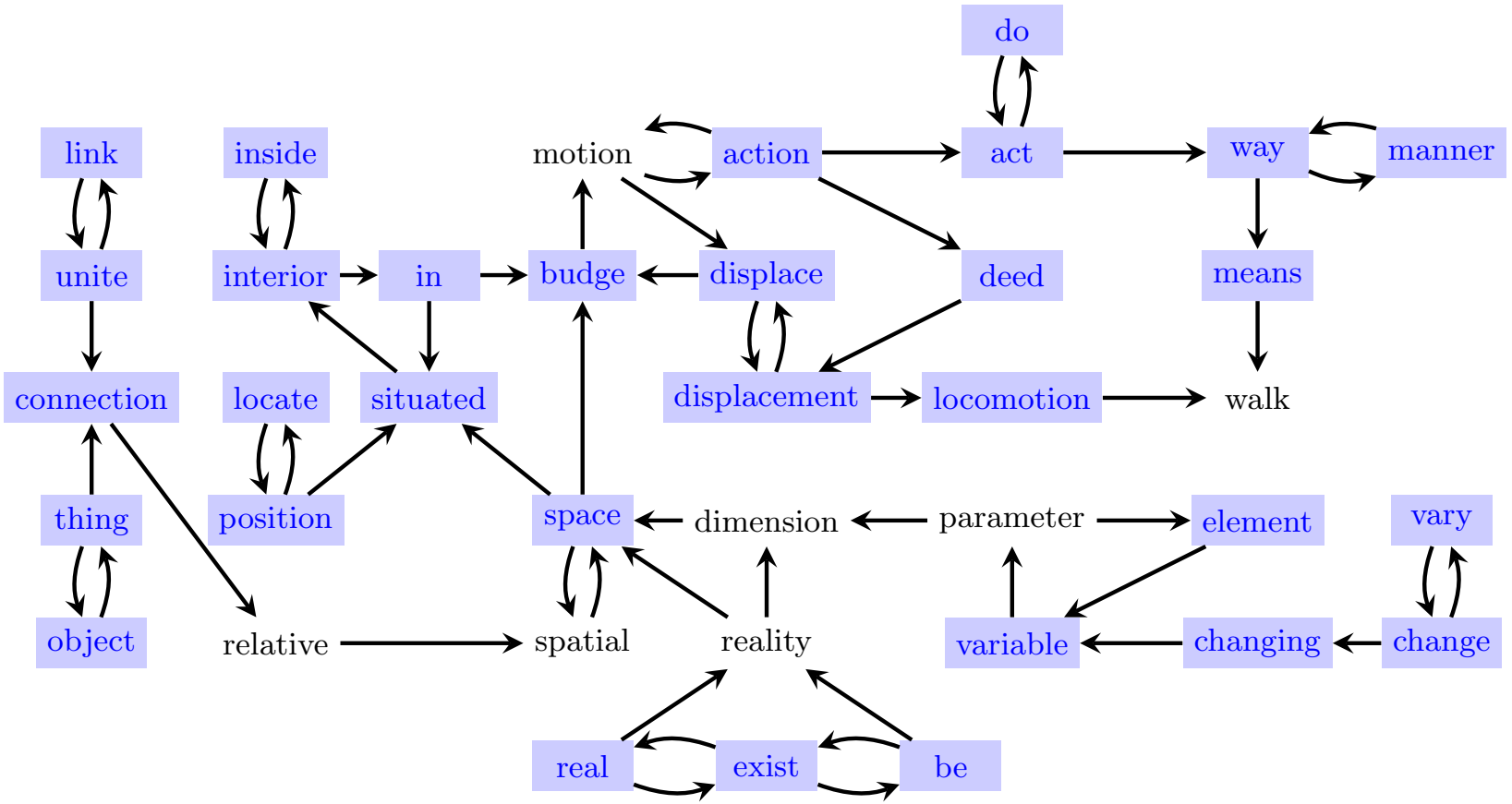
Ensembles d'ancrage

- Un ensemble de sommets U est appelé **ensemble d'ancrage** s'il nous permet d'apprendre **tous les mots** du graphe:



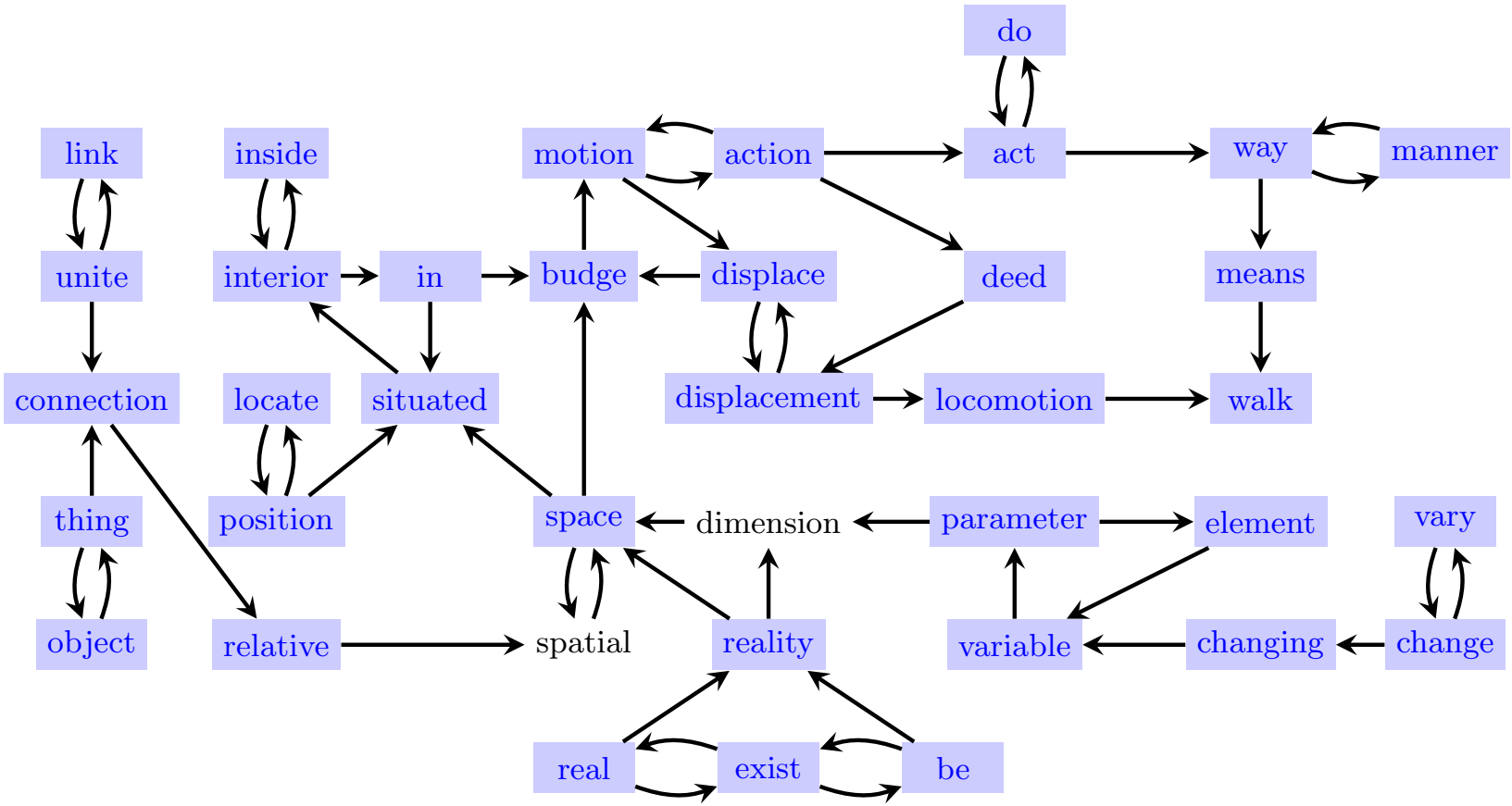
Ensembles d'ancrage

- Un ensemble de sommets U est appelé **ensemble d'ancrage** s'il nous permet d'apprendre **tous les mots** du graphe:



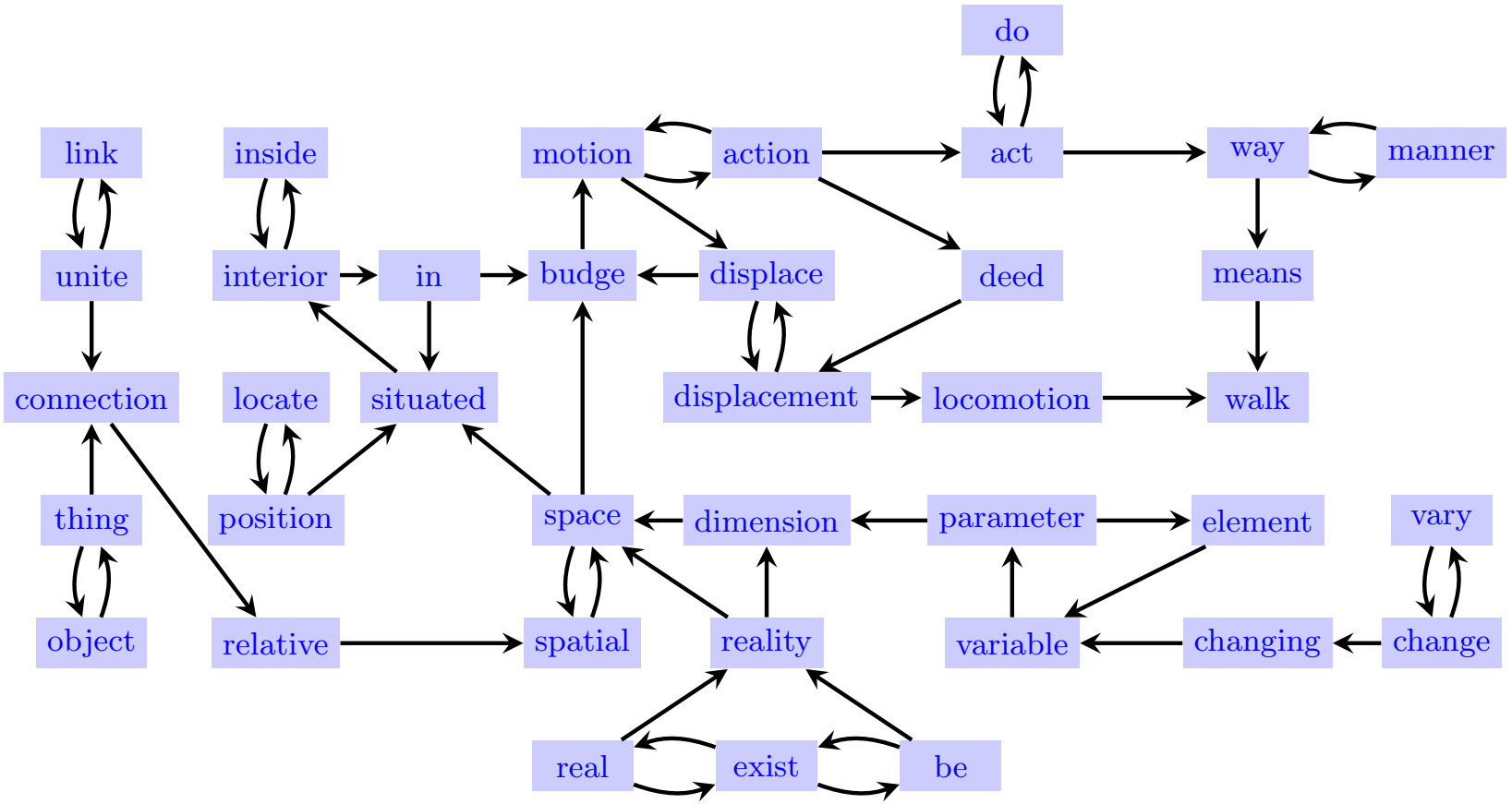
Ensembles d'ancrage

- Un ensemble de sommets U est appelé **ensemble d'ancrage** s'il nous permet d'apprendre **tous les mots** du graphe:

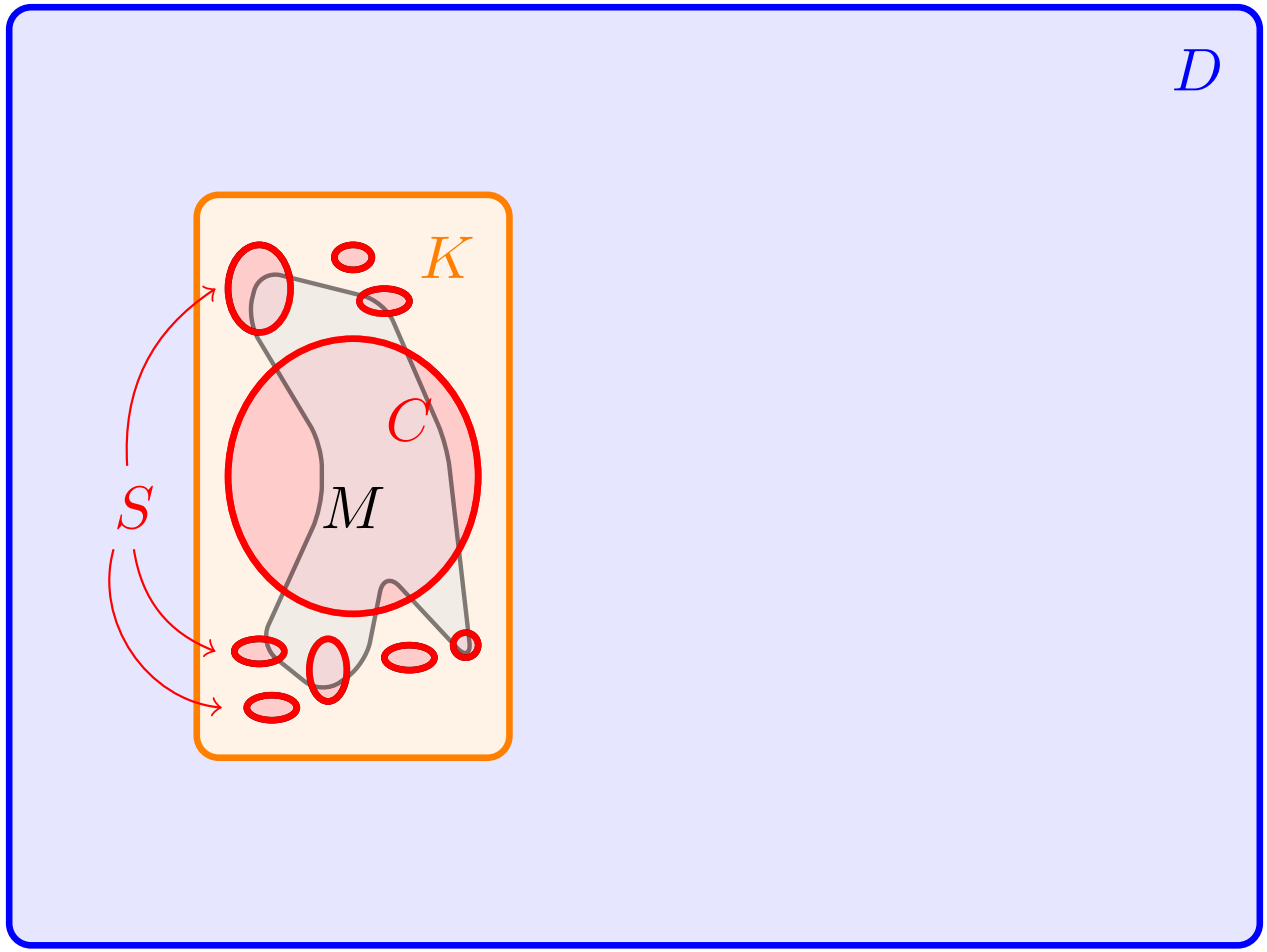


Ensembles d'ancrage

- Un ensemble de sommets U est appelé **ensemble d'ancrage** s'il nous permet d'apprendre **tous les mots** du graphe:



Anatomie d'un dictionnaire (suite)



Statistiques des dictionnaires

	Cambridge	Longman	Webster	WordNet	Game dictionaries (average)
Total words meanings	47147	69223	248466	132477	182
First sense meanings	25132	31026	91388	85195	-
Rest	22891 (91%)	28700 (93%)	80433 (88%)	75393 (88%)	10.14(7%)
Kernel	2241 (9%)	2326 (8%)	10955 (12%)	9802 (12%)	171.68 (93%)
Satellites	232 (1%)	540 (2%)	2978 (3%)	3410 (4%)	54.47 (29%)
Core	2009 (8%)	1786 (6%)	7977 (9%)	6392 (8%)	117.21 (64%)
MinSets	373 (1%)	452 (1%)	1396 (2%)	1094 (1%)	32.81 (18%)
Satellites-MinSets	59 (16%)	167 (37%)	596 (43%)	532 (49%)	20.59 (63%)
Core-MinSets	314 (84%)	285 (63%)	800 (57%)	562 (51%)	12.22 (37%)

Distances par rapport au noyau

- Soit G un **graphe-dictionnaire** et K son **noyau**.
- On définit une **fonction** dist_K sur les **sommets** par

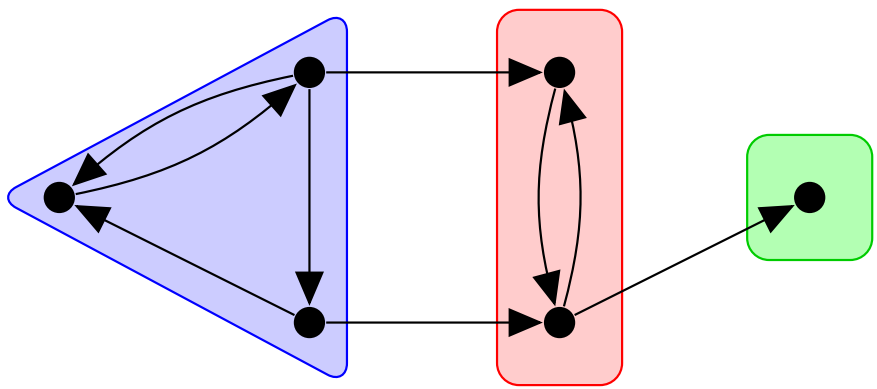
$$\text{dist}_K(u) = \begin{cases} 0, & \text{si } u \in K; \\ 1 + \max\{\text{dist}_K(v) \mid v \to u\}, & \text{sinon.} \end{cases}$$

- Meilleure granularité pour les mots **hors noyau** (*Rest of dictionary*).

Distance par rapport aux CFC

De façon similaire, soit G' le graphe obtenu de G en **fusionnant** les composantes fortement connexes. On définit dist_C récursivement comme suit :

1. $\text{dist}_C(u) = 0$ si si u est dans une source de G';
2. $\text{dist}_C(u) = 1 + \max\{\text{dist}_K(v) \mid v$ est dans une composante de G' qui est prédécesseur de celle de $u\}$.



Meilleure granularité pour les mots **intra noyau**.

Analyses statistiques

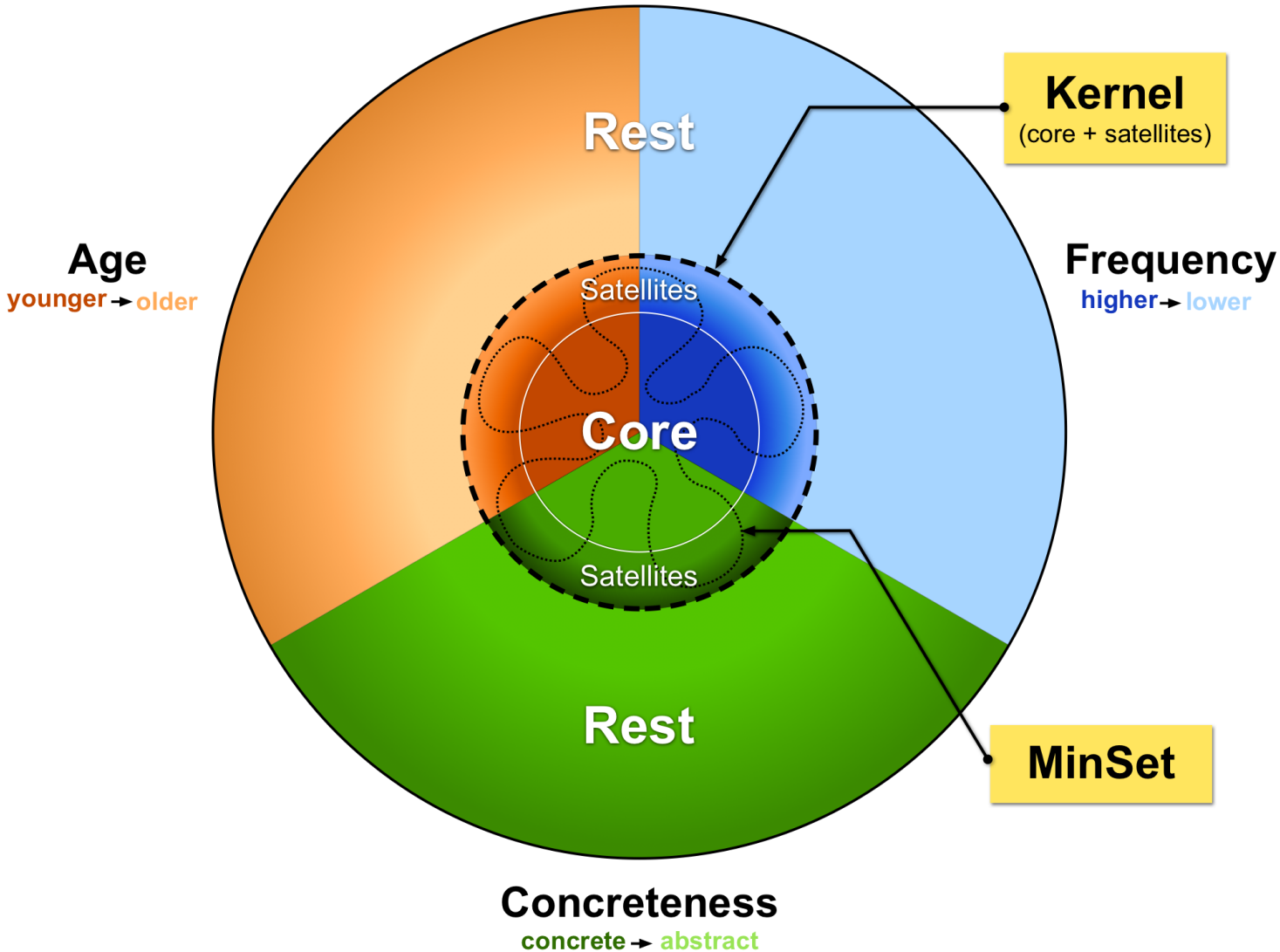


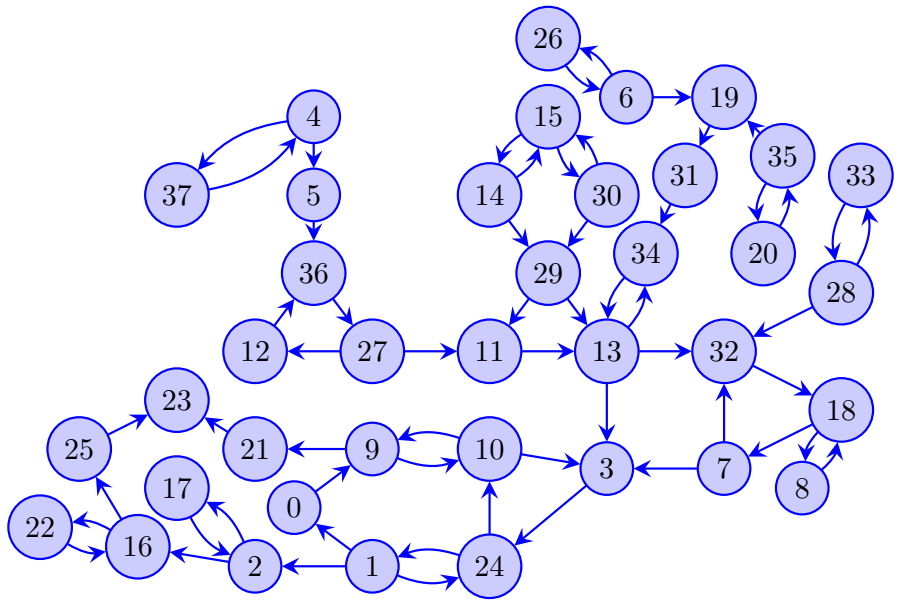
Table des matières

Transversaux de circuits

Définition

Soit $G = (V, E)$ un **graphe orienté**. Un **transversal de circuits** (en anglais, **feedback vertex set**) est un ensemble de sommets $U \subseteq V$ qui **couvre** tous les circuits de G.

- Lorsque S est de **taille minimale**, l'ensemble S est appelé **minimum feedback vertex set (MFVS)**.
- Le sous-graphe **induit** par $V - S$ est **acyclique**.

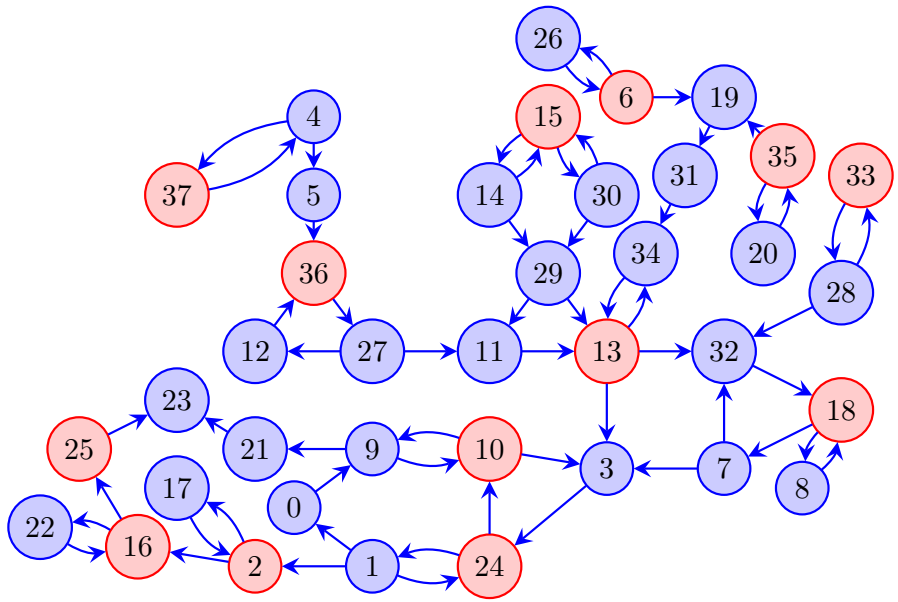


Transversaux de circuits

Définition

Soit $G = (V, E)$ un **graphe orienté**. Un **transversal de circuits** (en anglais, **feedback vertex set**) est un ensemble de sommets $U \subseteq V$ qui **couvre** tous les circuits de G.

- Lorsque S est de **taille minimale**, l'ensemble S est appelé **minimum feedback vertex set (MFVS)**.
- Le sous-graphe **induit** par $V - S$ est **acyclique**.

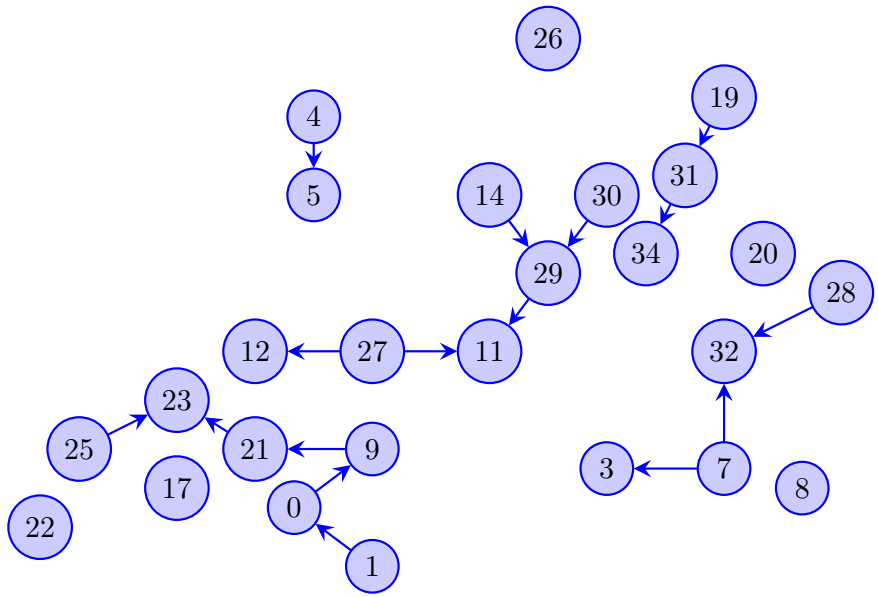


Transversaux de circuits

Définition

Soit $G = (V, E)$ un **graphe orienté**. Un **transversal de circuits** (en anglais, **feedback vertex set**) est un ensemble de sommets $U \subseteq V$ qui **couvre** tous les circuits de G.

- Lorsque S est de **taille minimale**, l'ensemble S est appelé **minimum feedback vertex set (MFVS)**.
- Le sous-graphe **induit** par $V - S$ est **acyclique**.



Applications du MFVSP

Le problème de trouver un **transversal de circuits** de taille minimale apparaît dans des domaines aussi **variés** que :

- **Électronique**;
- Conception **assistée** par ordinateur;
- Détection de **livelock**;
- **Sécurité**;
- Réseaux **bayésiens**.

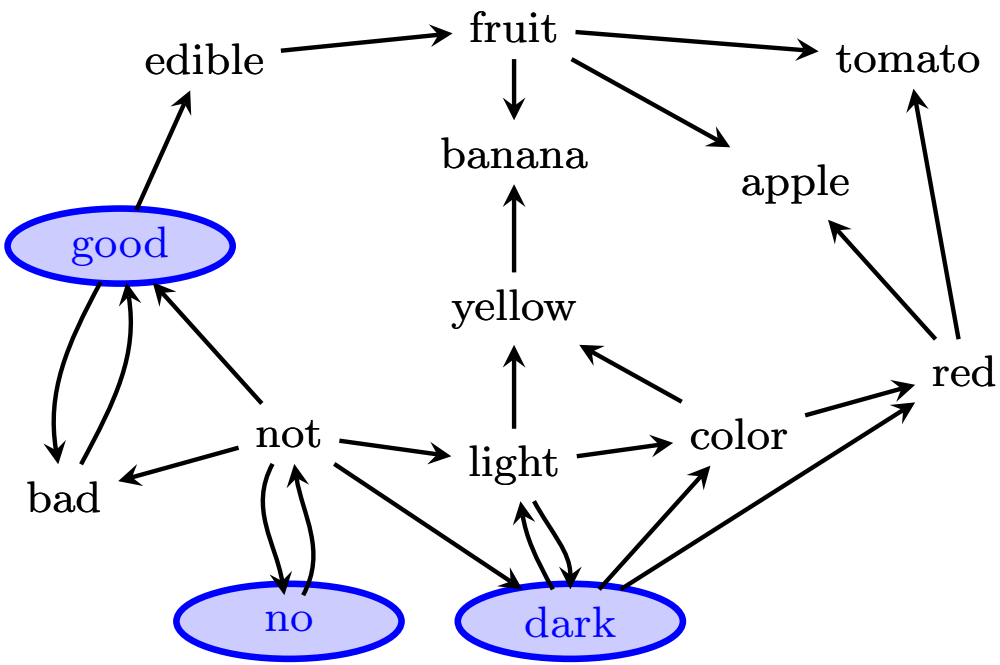
Transversaux de circuits

Théorème (B. M. et al., 2008)

Soit $G = (V, E)$ un graphe **orienté** et $U \subseteq V$. Alors U est un **ensemble d'ancrage si et seulement si** U est un **transversal de circuits**.

Corollaire (B. M. et al., 2008)

Le problème de trouver un ensemble d'ancrage est **NP-difficile**.



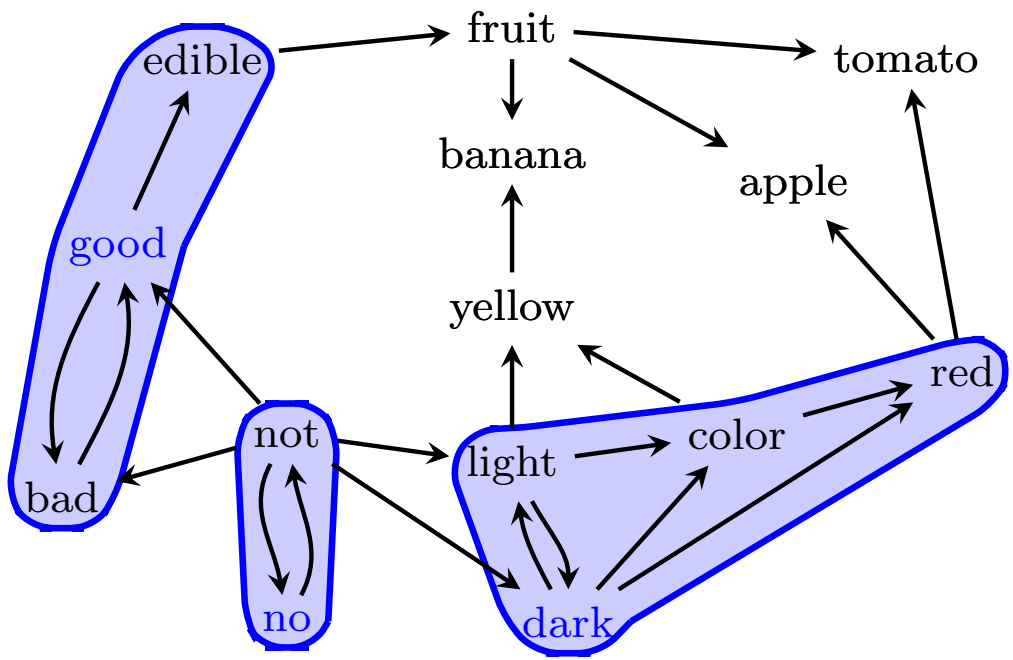
Transversaux de circuits

Théorème (B. M. et al., 2008)

Soit $G = (V, E)$ un graphe **orienté** et $U \subseteq V$. Alors U est un **ensemble d'ancrage si et seulement si** U est un **transversal de circuits**.

Corollaire (B. M. et al., 2008)

Le problème de trouver un ensemble d'ancrage est **NP-difficile**.



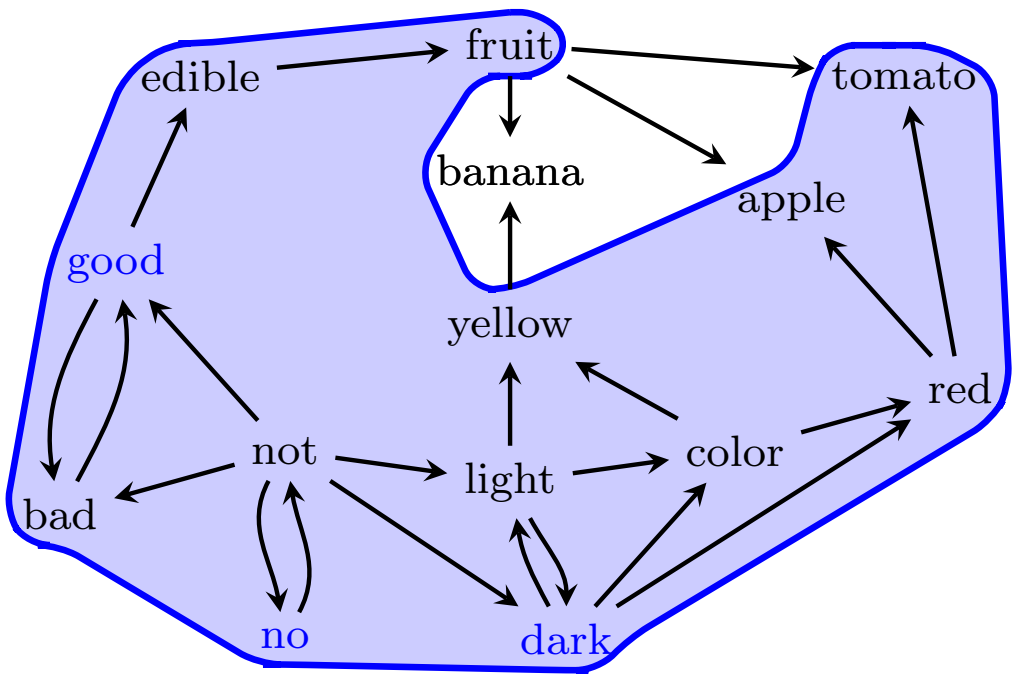
Transversaux de circuits

Théorème (B. M. et al., 2008)

Soit $G = (V, E)$ un graphe **orienté** et $U \subseteq V$. Alors U est un **ensemble d'ancrage si et seulement si** U est un **transversal de circuits**.

Corollaire (B. M. et al., 2008)

Le problème de trouver un ensemble d'ancrage est **NP-difficile**.



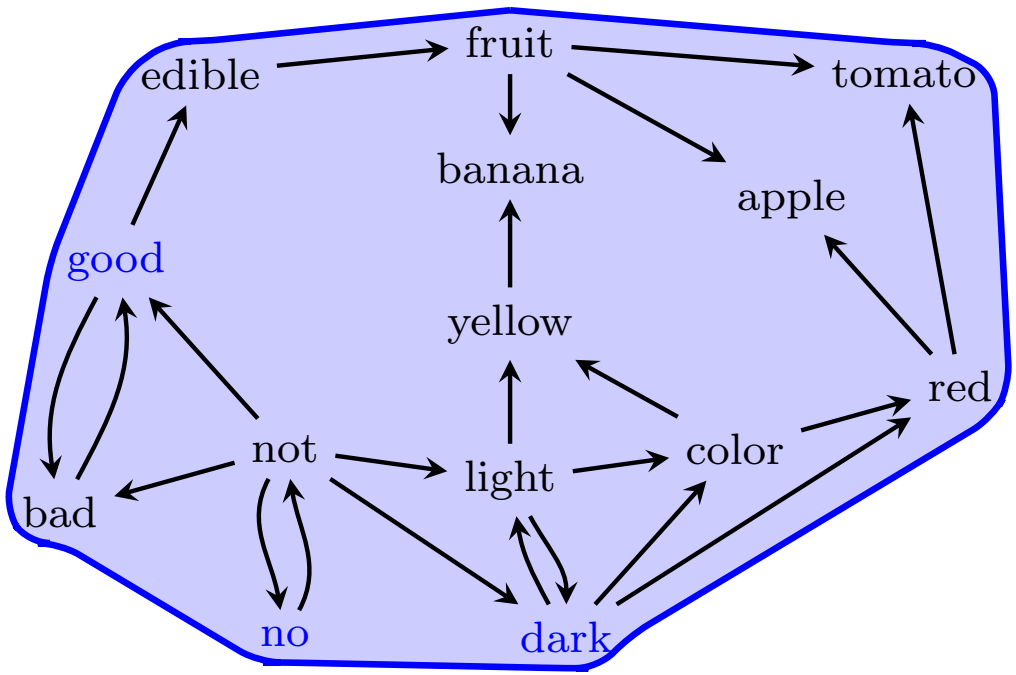
Transversaux de circuits

Théorème (B. M. et al., 2008)

Soit $G = (V, E)$ un graphe **orienté** et $U \subseteq V$. Alors U est un **ensemble d'ancrage si et seulement si** U est un **transversal de circuits**.

Corollaire (B. M. et al., 2008)

Le problème de trouver un ensemble d'ancrage est **NP-difficile**.

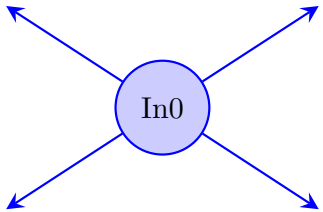


Difficulté du problème

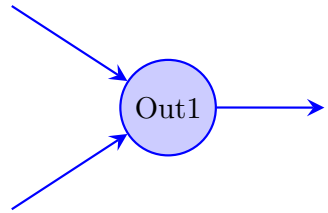
- Le problème de **trouver un MFVS** dans un graphe orienté est **NP-difficile**;
- **Trois** stratégies possibles :
  - Concevoir un algorithme **exact**, mais **exponentiel**;
  - Se **restreindre** à une **classe** de graphes pour laquelle un algorithme **polynomial** existe;
  - Utiliser des **approximations** ou des **méta-heuristiques**.
- En **2000**, Lin et Jou ont proposé 8 **contractions** de graphes orientés qui **préservent** l'existence d'un **traversal**.

Contractions

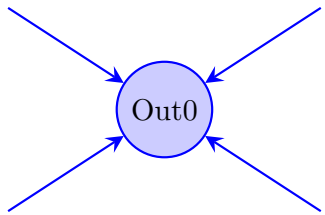
In0



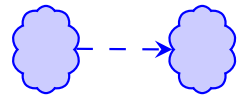
Out1



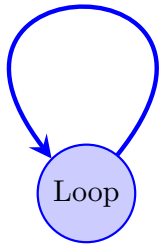
Out0



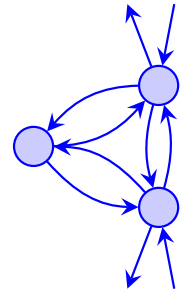
Pie



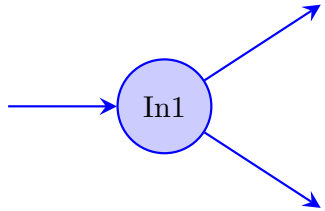
Loop



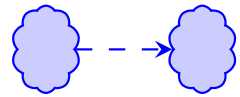
Core



In1



Dome

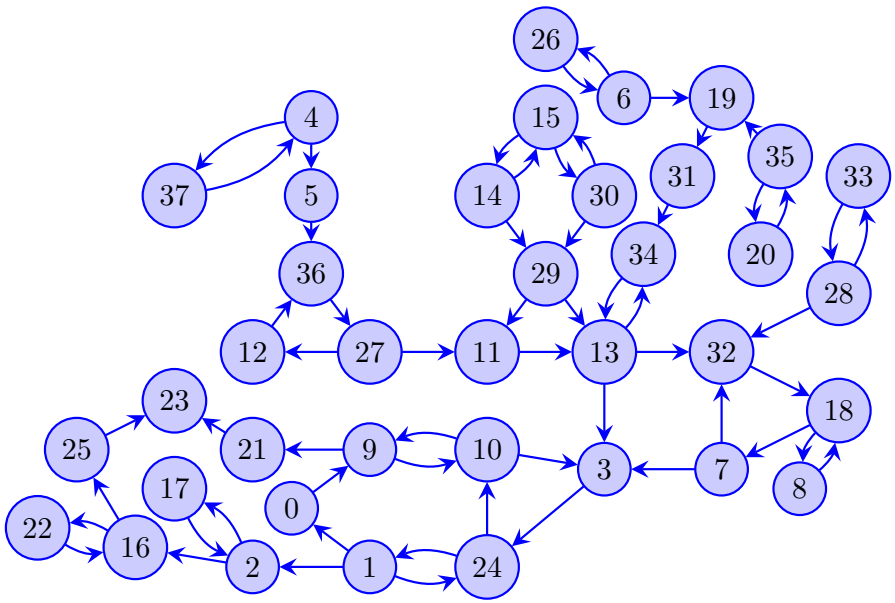


Graphes contractibles

Définition

Un **graphe contractible** est un graphe orienté pour lequel l'application **successive** des **huit opérateurs** résulte en un **graphe vide**.

- Par conséquent, les sommets **ajoutés** par les opérateurs Loop et Core forment un **MFVS** du graphe de départ.



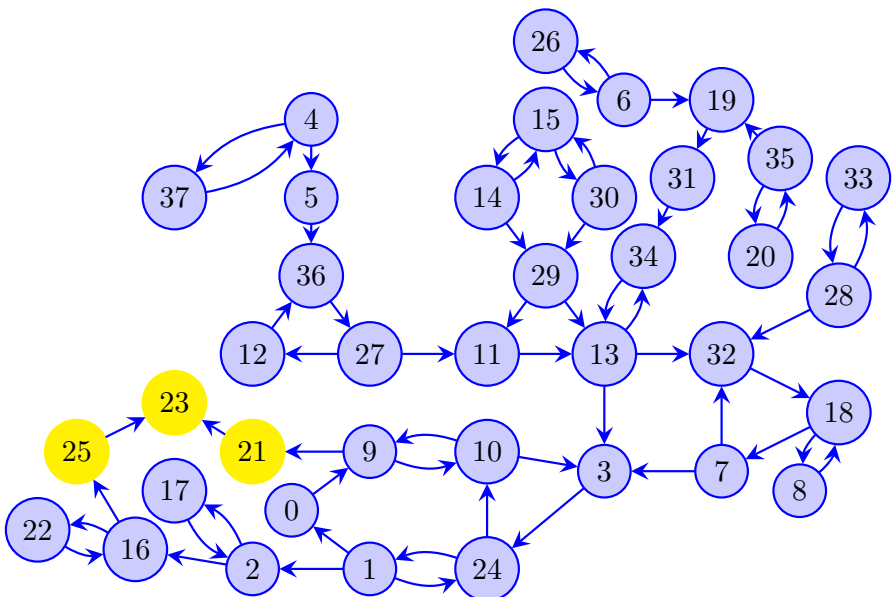
Un graphe contractible

Graphes contractibles

Définition

Un **graphe contractible** est un graphe orienté pour lequel l'application **successive** des **huit opérateurs** résulte en un **graphe vide**.

- Par conséquent, les sommets **ajoutés** par les opérateurs Loop et Core forment un **MFVS** du graphe de départ.



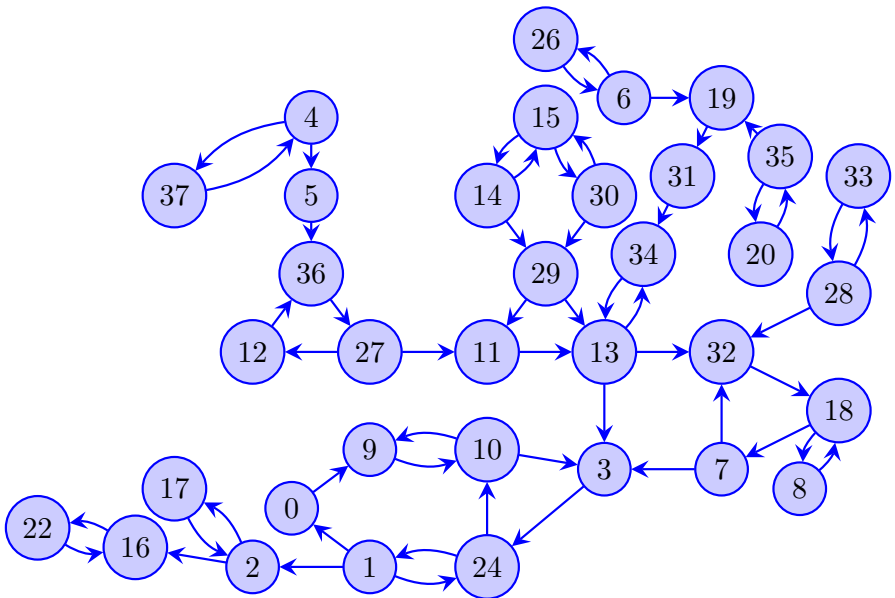
On applique In0 et Out0

Graphes contractibles

Définition

Un **graphe contractible** est un graphe orienté pour lequel l'application **successive** des **huit opérateurs** résulte en un **graphe vide**.

- Par conséquent, les sommets **ajoutés** par les opérateurs Loop et Core forment un **MFVS** du graphe de départ.



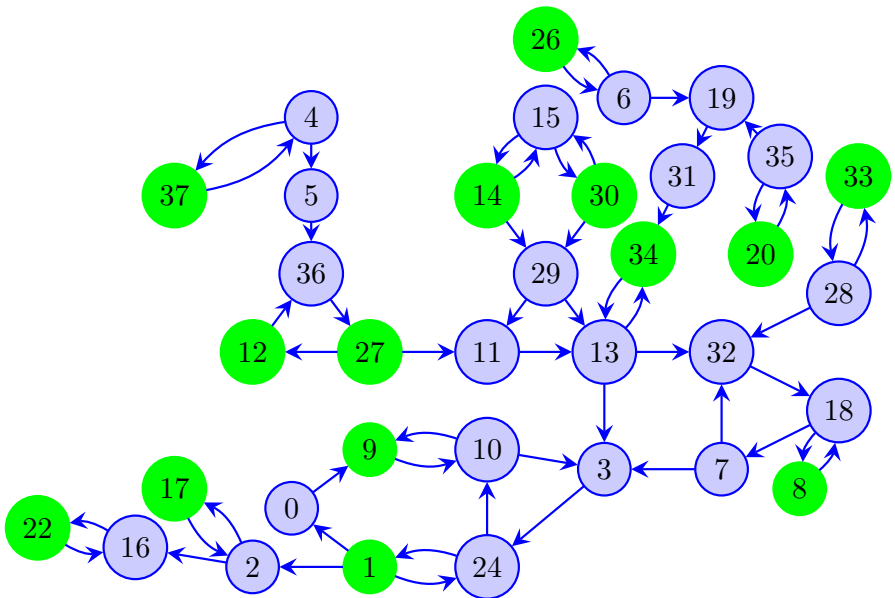
On applique Loop

Graphes contractibles

Définition

Un **graphe contractible** est un graphe orienté pour lequel l'application **successive** des **huit opérateurs** résulte en un **graphe vide**.

- Par conséquent, les sommets **ajoutés** par les opérateurs Loop et Core forment un **MFVS** du graphe de départ.



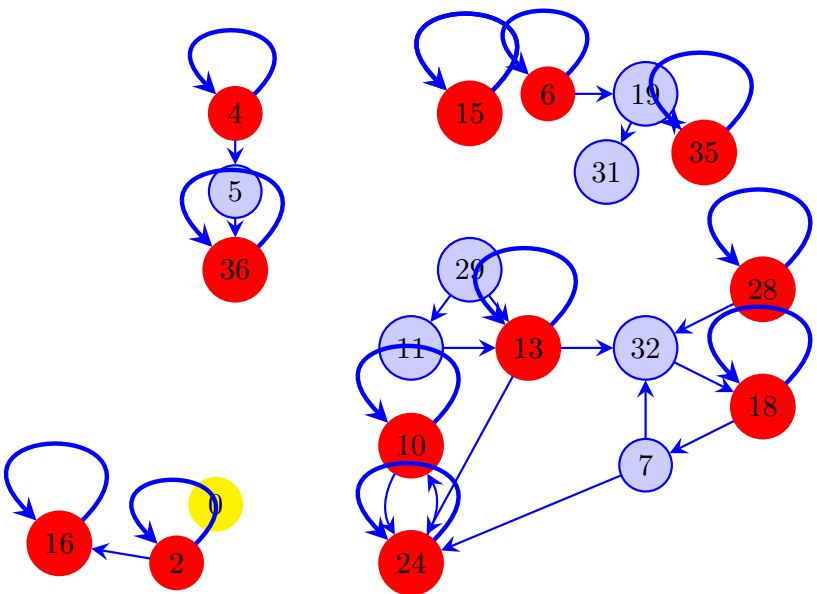
On applique In1 et Out1

Graphes contractibles

Définition

Un **graphe contractible** est un graphe orienté pour lequel l'application **successive** des **huit opérateurs** résulte en un **graphe vide**.

- Par conséquent, les sommets **ajoutés** par les opérateurs Loop et Core forment un **MFVS** du graphe de départ.



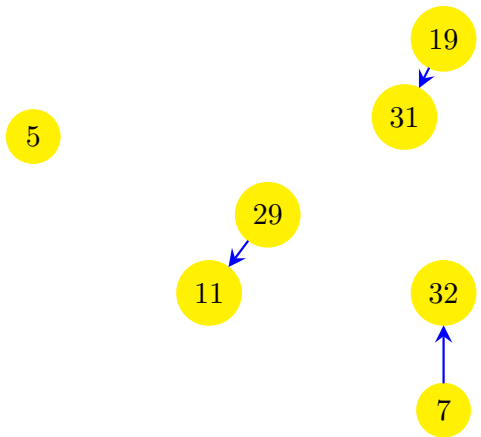
On applique à nouveau In0, Out0 et Loop

Graphes contractibles

Définition

Un **graphe contractible** est un graphe orienté pour lequel l'application **successive** des **huit opérateurs** résulte en un **graphe vide**.

- Par conséquent, les sommets **ajoutés** par les opérateurs Loop et Core forment un **MFVS** du graphe de départ.



Puis encore In0

Algorithme exact

fonction MFVS(G : graphe, S, B : sommets, p, L: entiers) : sommets
  $S \leftarrow S \cup$ Reduce(G) ▷ **On réduit**
  $k \leftarrow$ NumberConnectedComponents(G)
  si $k > 1$ **alors** ▷ **On partitionne selon les composantes connexes**
    pour $C \in$ ConnectedComponents(G) **faire**
      $S \leftarrow S \cup$ MFVS($C, 0, V(C), 0, 0$)
    fin pour
    retourner S
  sinon
    $\ell =$ MFVSLowerBound(G)
    si $L = 0$ **alors**
      $p \leftarrow \ell$
      $B \leftarrow S \cup$ MFVSApproximateSolution(G)
    fin si
    si $|S| + \ell > |B|$ **alors** ▷ **On élague**
      retourner B
    sinon si $G = \emptyset$ **alors**
      retourner S
    fin si
    ▷ **On effectue un branchement en incluant/excluant un sommet**
    $v \leftarrow$ MaxDegreeVertex(G)
    $S' \leftarrow$ MFVS(G.Delete(v), $S \cup \{v\}, B, p, L + 1$)
    Soit B l'ensemble le plus petit entre S' et B
    Si $p = |B|$, alors retourner B
    $S'' \leftarrow$ MFVS(G.Vanish(v), $S, B, p, L + 1$)
    retourner l'ensemble le plus petit entre S'' et B
  fin si
fin fonction

Réduction des dictionnaires

- Tous les dictionnaires issus du **jeu** sont **entièrement réductibles** ou presque;
- Pour les **grands** dictionnaires :

CIDE	:	$\approx$ 800 mots
LDOCE	:	$\approx$ 650 mots
WN	:	$\approx$ 2600 mots
MWC	:	$\approx$ 3000 mots

Algorithmes exacts et approximations

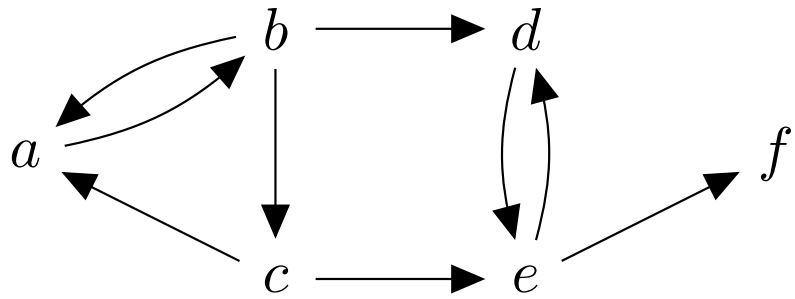
- **Programmation linéaire en nombres entiers** :
  - Chaque **circuit** correspond à une **contrainte**;
  - Le nombre de circuits est **exponentiel**;
  - Il faut se restreindre aux circuits **courts** et utiliser des **plans coupants**.
- CIDE et LDOCE : **solutions exactes**;
- WN et MWC : **solutions approximatives**;
- **Méta-heuristiques** (en cours).

Table des matières

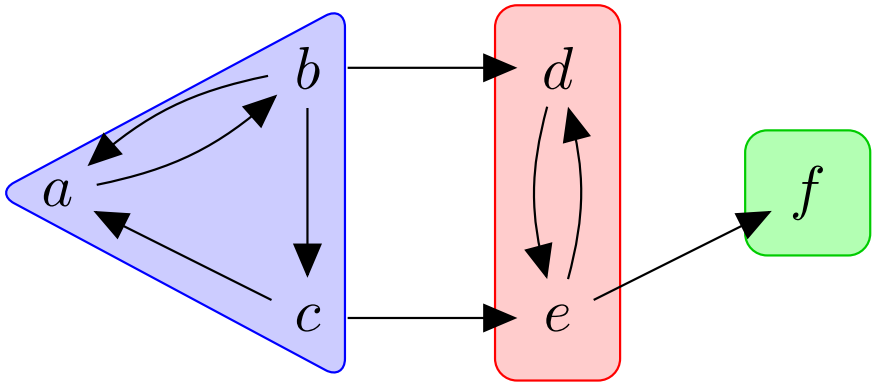
Énumération de transversaux

- L'algorithme de Lin et Jou permet de calculer **un transversal** de cardinalité minimale;
- **Problème :** Est-ce que ce transversal est **représentatif** de **tous les autres** transversaux ?
- Pour répondre à cette question, une première idée consiste à énumérer **tous les transversaux**;
- Une approche naïve ne **suffit pas**, car il peut y en avoir un **très grand nombre**;
- Il faut trouver une façon de **stocker implicitement** toutes les solutions.

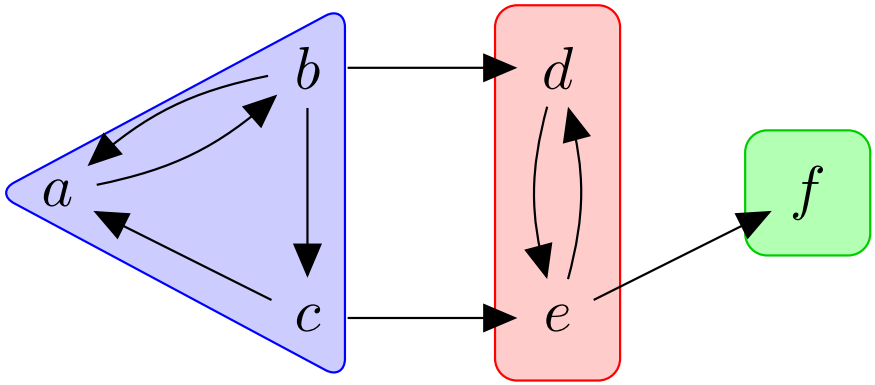
Exemple



Exemple

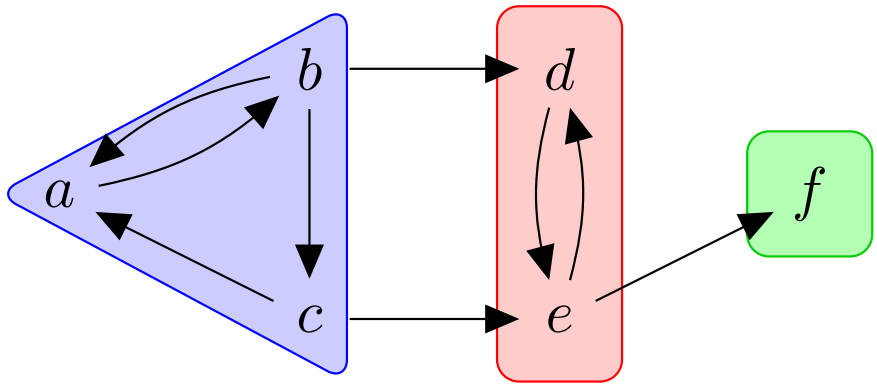


Exemple



- c et f n'appartiennent à **aucune solution**;
- a et b sont **interchangeables**;
- d et e sont **interchangeables**;

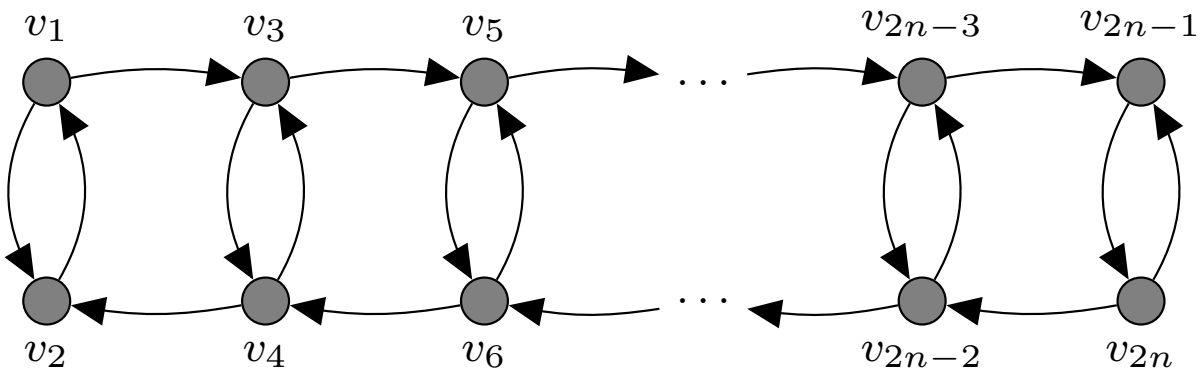
Exemple



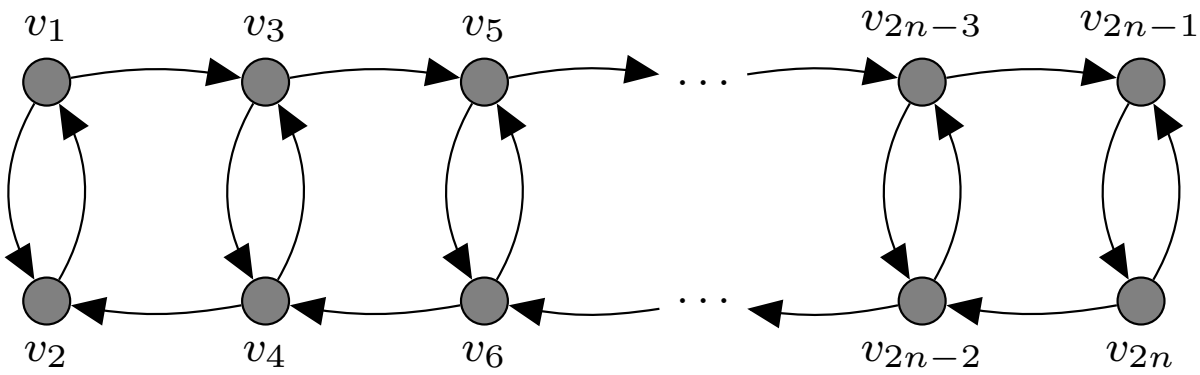
- c et f n'appartiennent à **aucune solution**;
- a et b sont **interchangeables**;
- d et e sont **interchangeables**;
- Les solutions sont donc

$$\{a, d\}, \{a, e\}, \{b, d\}, \{b, e\}.$$

Arbres et/ou

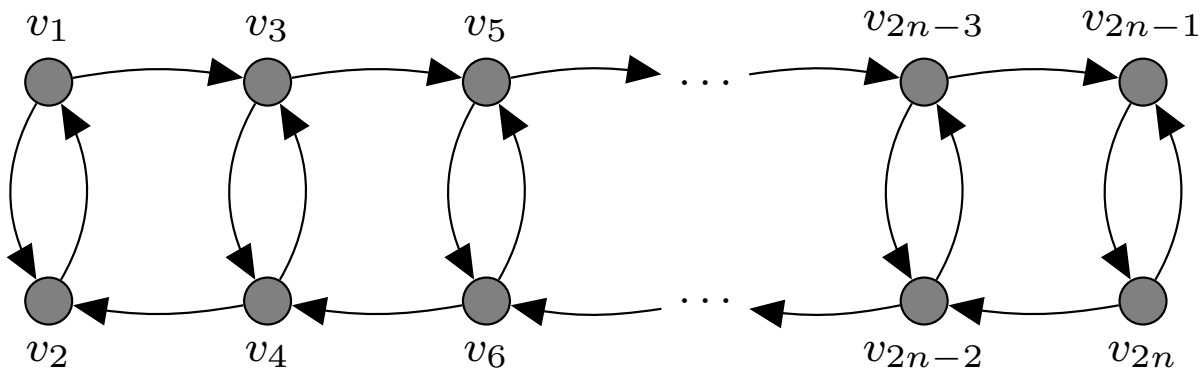


Arbres et/ou

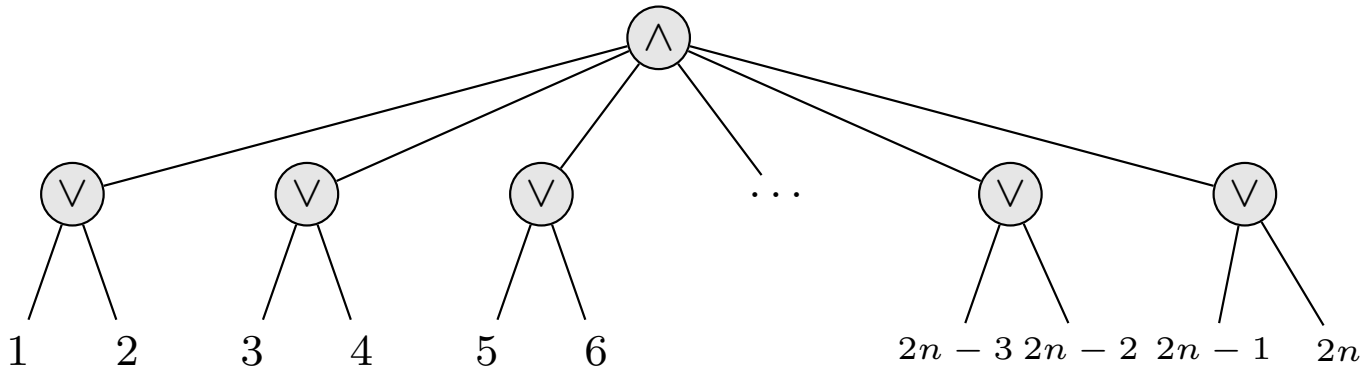


- Tout MFVS du graphe ci-bas est obtenu en choisissant un sommet parmi $\{v_i, v_{i+1}\}$, pour $i = 1, 3, 5, \ldots, 2n-1$:

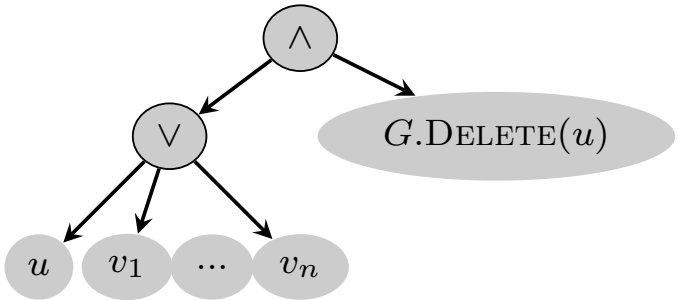
Arbres et/ou



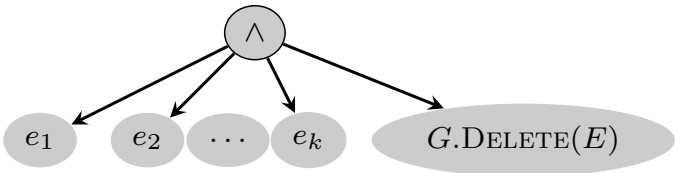
- Tout MFVS du graphe ci-bas est obtenu en choisissant un sommet parmi $\{v_i, v_{i+1}\}$, pour $i = 1, 3, 5, \ldots, 2n-1$:
- On peut donc représenter l'**ensemble de toutes les solutions** par :



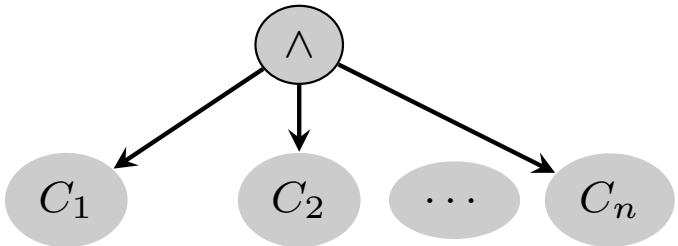
Règles de branchement



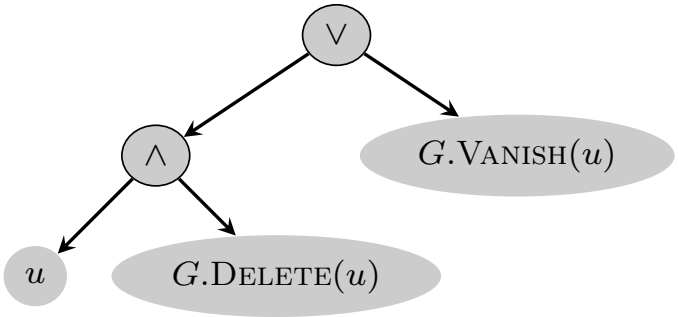
Sommets **équivalents**



Sommets **essentiels**



Composantes **connexes**



En dernier **recours**

Algorithme d'énumération

```
fonction AllMFVS(G : graphe orienté) : arbre et/ou
    Faire disparaître tous les sommets inutiles
    Soit E l'ensemble des sommets essentiels
    si |E| > 0 alors
                                          ▷ Suppression des sommets essentiels
        x ← NewNode(∧)
        pour u ∈ E faire
            Ajouter NewLeaf(u) à x.children
        fin pour
        G' ← G.Delete(E)
        Ajouter AllMFVS(G') à x.children
    sinon
        Appliquer Pie et Dome
        C ← ConnectedComponents(G)
        si |C| > 1 alors                  ▷ Branchement selon les composantes connexes
            x ← NewNode(∧)
            pour C ∈ C faire
                Ajouter AllMFVS(C) à x.children
            fin pour
        sinon                 ▷ Branchement en incluant/excluant un sommet quelconque
            Soit u un sommet quelconque
            x ← NewNode(∨)
            y ← NewNode(∧)
            y.children ← {NewLeaf(u)}
            G' ← G.Delete(u)
            Ajouter AllMFVS(G') à y.children
            Ajouter y à x.children
            G' ← G.Vanish(u)
            Ajouter AllMFVS(G') à x.children
        fin si
    fin si
    retourner x
fin fonction
```

Table des matières

Conclusion

- Étudier plus en profondeur les propriétés des **arbres et/ou** en tant que **structure de données**;
- Étendre les analyses statistiques à d'**autres langues** :
  - **Espagnol**, **français** et **portugais** en cours (Wiktionnaires);
  - Problème **majeur** : **désambiguïsation** et **segmentation** des définitions.
- **Stratégies** utilisées dans le **jeu du dictionnaire**.
- etc.