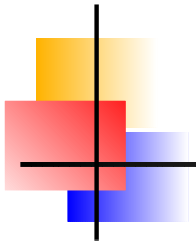
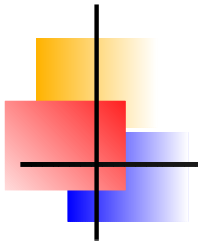


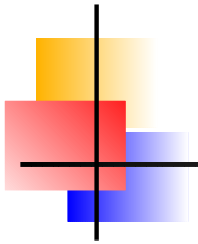
Mesurer la similarité de graphes étiquetés

Sébastien SORLIN, Pierre-Antoine CHAMPIN, Christine SOLNON



Motivations

- Besoin de mesurer la similarité d'objets pour :
  - Le raisonnement à partir de cas
  - La recherche d'informations sur le Web ou dans des BD semi-structurées
  - L'extraction d'images
  - L'identification de *patrons de conception* dans du code
  - ...
- Utilisation de graphes pour représenter ces objets

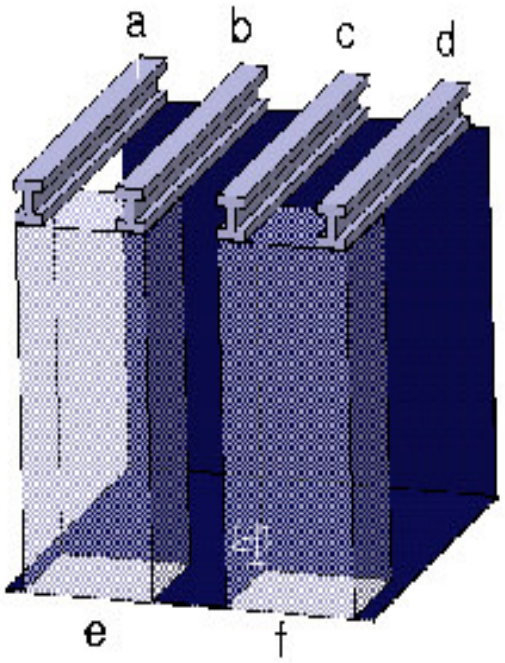


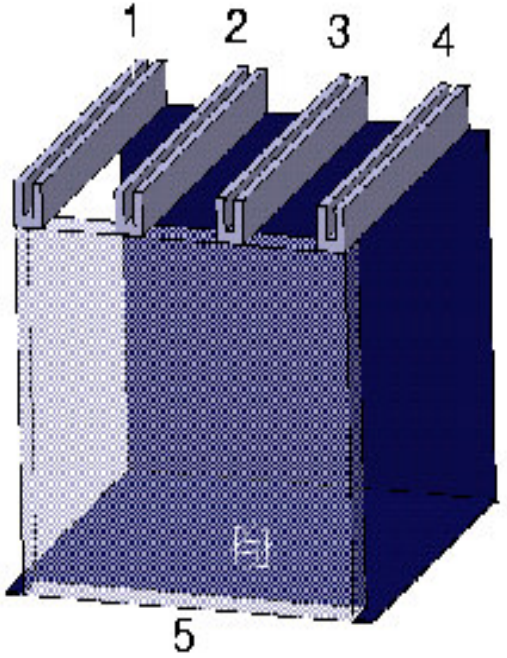
Travaux existants

- Isomorphisme de graphe $\Rightarrow$ égalité
- Isomorphisme de sous-graphe $\Rightarrow$ inclusion
  - $\rightarrow$ Formulation CSP [Régin 95]
  - $\rightarrow$ Restriction à des arbres [Ricci,Senter 98]
- Plus grand sous-graphe commun $\Rightarrow$ intersection

  Plus petit nombre d'opérations pour rendre les graphes isomorphes
  - $\rightarrow$ Approche type A^* [Nilsson 80]
  - $\rightarrow$ PPC avec explications [Guéhéneuc, Jussien 01]
  - $\rightarrow$ Recherche Taboue [Petrovic, Kendall, Yang 02]

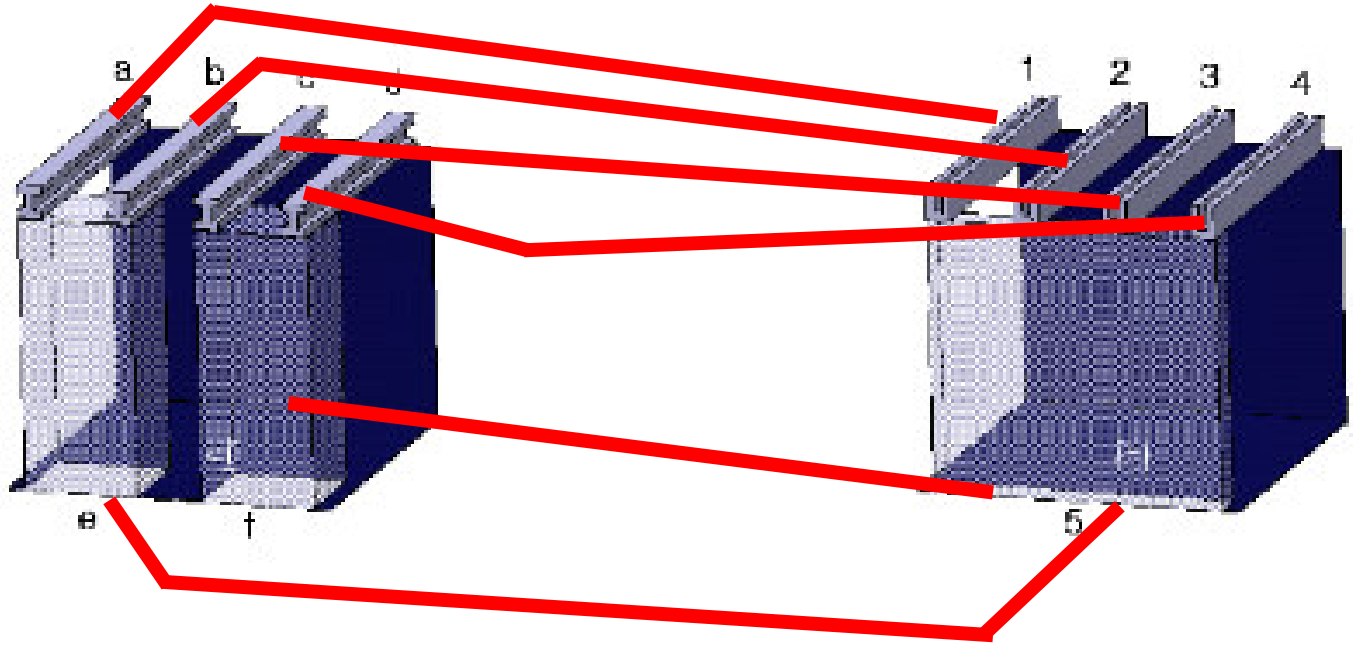
Exemple introductif (1/2)



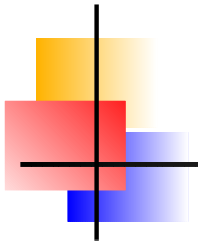


- Evaluer la similarité (ou trouver les différences !)
  - La forme des poutres
  - Le nombre de murs

Exemple introductif (2/2)

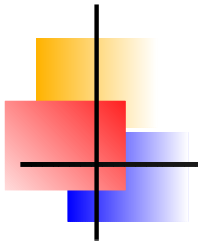


- Trouver les différences
  - $\Rightarrow$ mettre en correspondance les composants
    - $a/1, b/2, c/3, d/4$
    - e et $f/5$



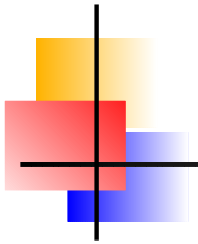
Discussion

- Deux composants différents peuvent partager des caractéristiques communes
  $\Rightarrow$ Multi-étiquetage des composants
- Toutes les caractéristiques ne sont pas forcément équivalentes
  $\Rightarrow$ Introduction de connaissances dans la mesure de similarité
- Une mise en correspondance 1 à 1 des composants ne suffit pas
  $\Rightarrow$ Pouvoir apparier un composant à plusieurs autres



Contribution, Plan

- Un modèle de représentation des objets structurés
  - Les graphes orientés multi-étiquetés
- Une mesure de similarité adaptée à ces graphes
  - Paramètrable en fonction de l'application
  - Autorisant l'appariement d'un sommet à plusieurs autres
- Des algorithmes de recherche de similarité



Graphes multi-étiquetés

- L_V = ens. d'étiquettes de sommets
- L_E = ens. d'étiquettes d'arcs

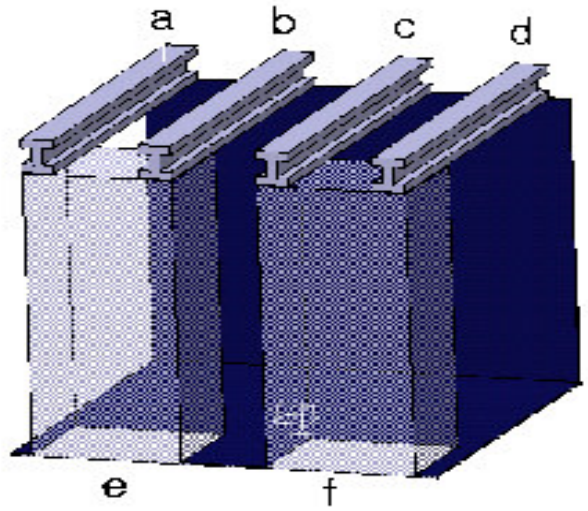
- Un graphe étiqueté $G = \langle V, r_V, r_E \rangle$ est défini par
  - V = ens. de sommets
  - $r_V \subseteq V \times L_V$ associe 1 ou plusieurs étiquettes à chaque sommet
  - $r_E \subseteq V \times V \times L_E$ associe 1 ou plusieurs étiquettes à chaque arc

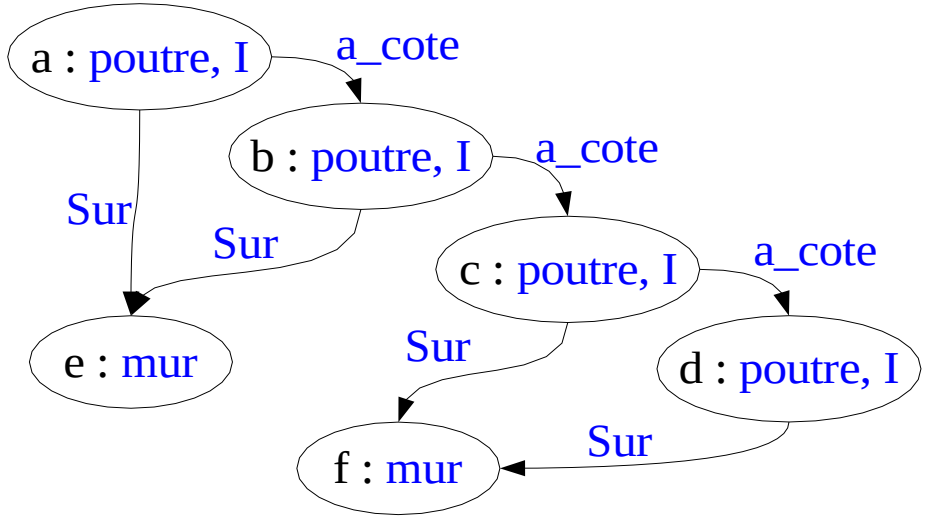
$$\rightsquigarrow \text{Arcs } E = \{(u, v)/\exists (u, v, l) \in r_E\}$$

- Rm : G est "décrit" par ses relations r_V et r_E :

$$descr(G) = r_V \cup r_E$$

Graphes multi-étiquetés : Exemple



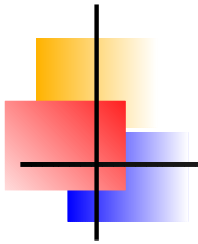


$$G_1 = \langle\, V_1 = \{\, a, b, c, d, e, f \,\}$$

$$r_{V_1} = \{\, (a, poutre), (b, poutre), (c, poutre), (d, poutre),$$
$$(a, I), (b, I), (c, I), (d, I),$$
$$(e, mur), (f, mur) \,\}$$

$$r_{E_1} = \{\, (a, b, a\_cote), (b, c, a\_cote), (c, d, a\_cote),$$
$$(a, e, sur), (b, e, sur), (c, f, sur), (d, f, sur) \,\} \,\rangle$$

$$descr(G_1) = r_{V1} \cup r_{E1}$$



Similarité

- [Tversky,77] : La similarité de 2 objets a et b est fonction de leurs caractéristiques communes par rapport à l'ensemble de leurs caractéristiques
  - Si a est décrit par A et b par B

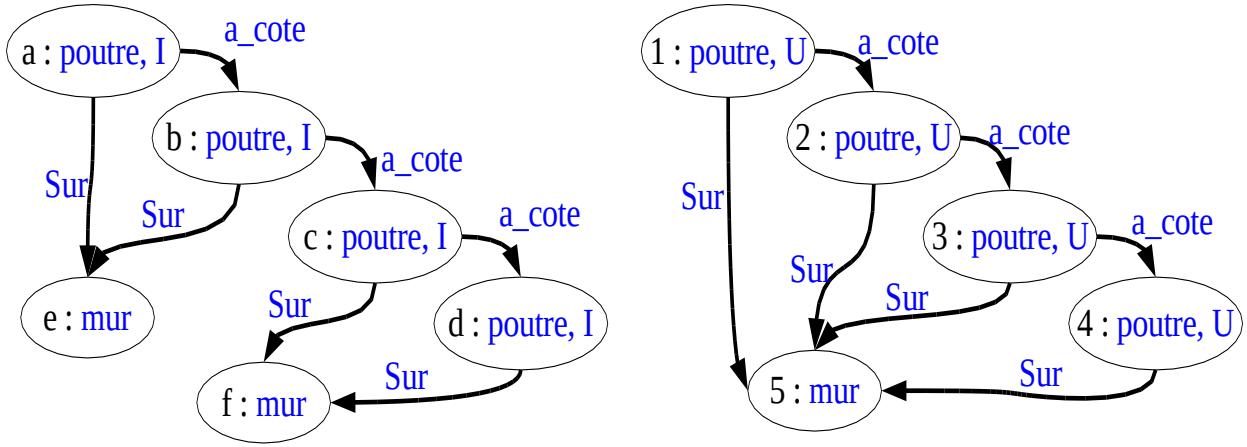
$$sim_{Tversky}(a,b) = \frac{f(A \cap B)}{f(A \cup B)}$$

- Similarité de 2 graphes G_1 et G_2 fonction des caractéristiques communes à $descr(G_1)$ et $descr(G_2)$
  - Problème :

$$V_1 \cap V_2 = \emptyset \Rightarrow descr(G_1) \cap descr(G_2) = \emptyset$$

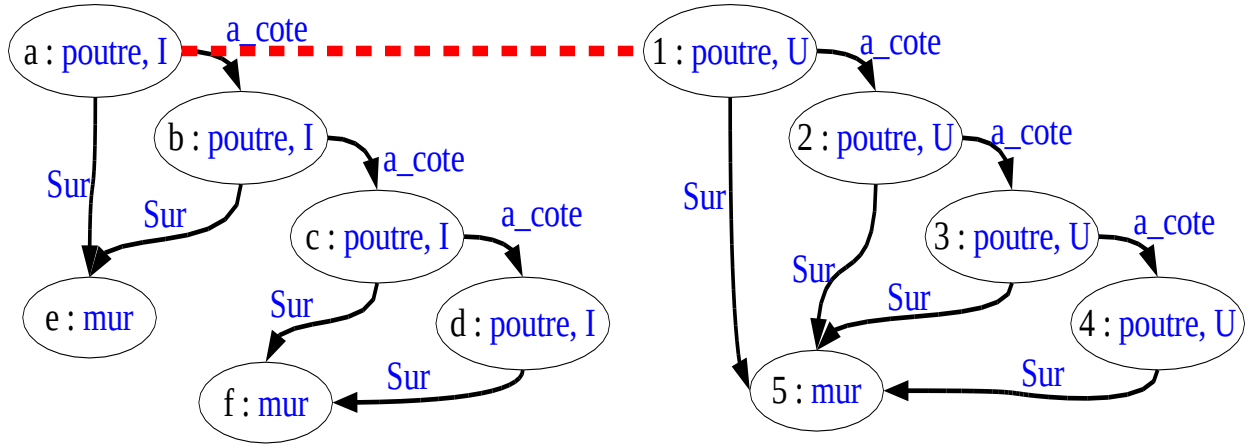
Appariement (1/2)

- Besoin de mettre les éléments en correspondance pour comparer les graphes :



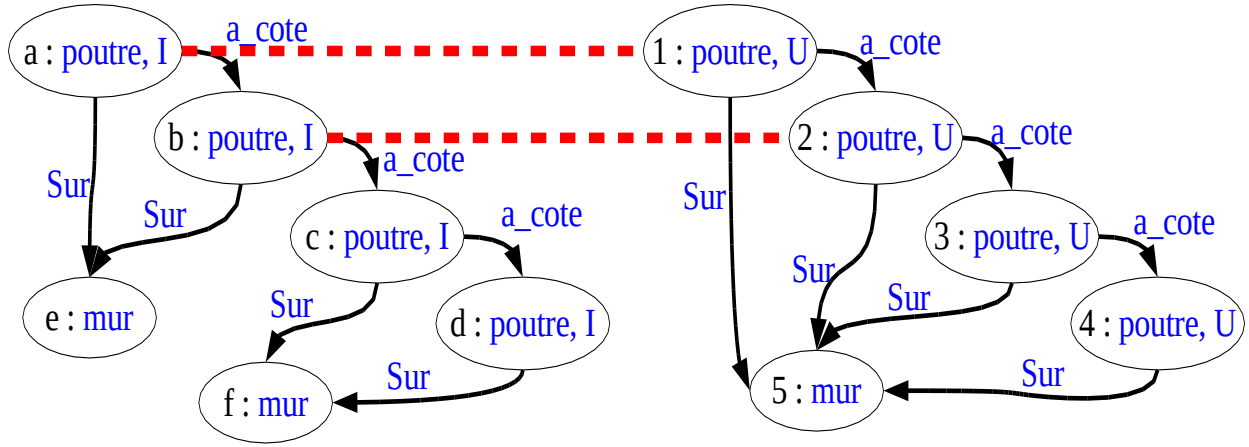
Appariement (1/2)

- Besoin de mettre les éléments en correspondance pour comparer les graphes :



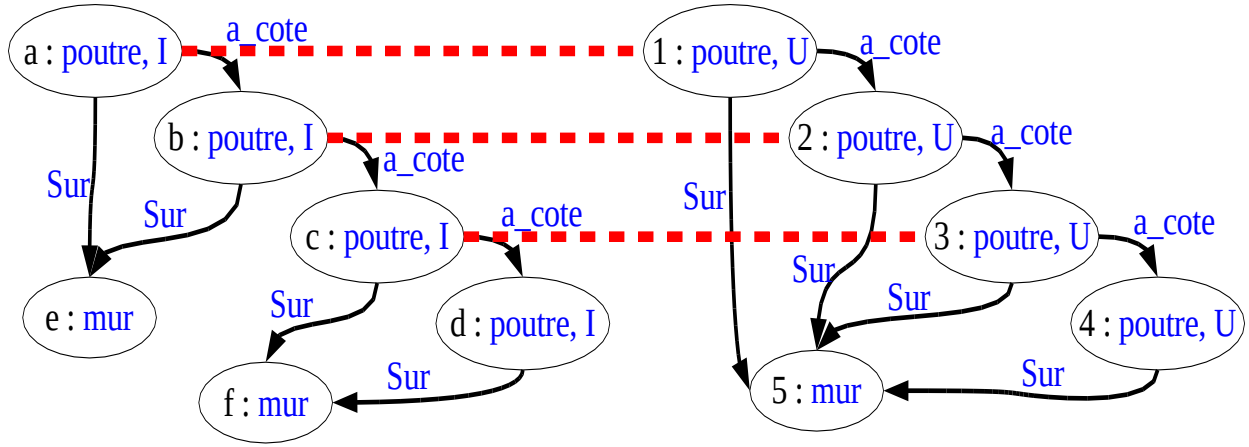
Appariement (1/2)

- Besoin de mettre les éléments en correspondance pour comparer les graphes :



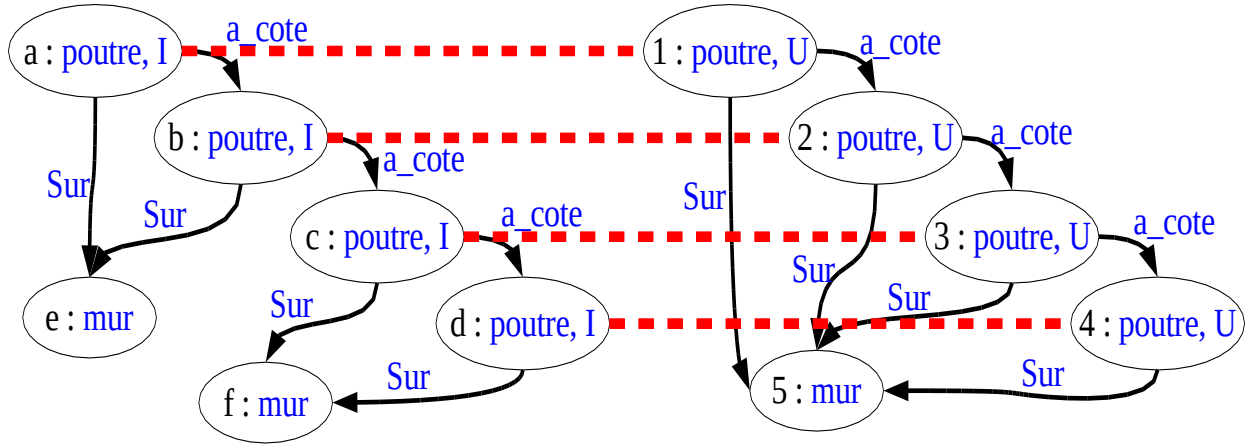
Appariement (1/2)

- Besoin de mettre les éléments en correspondance pour comparer les graphes :



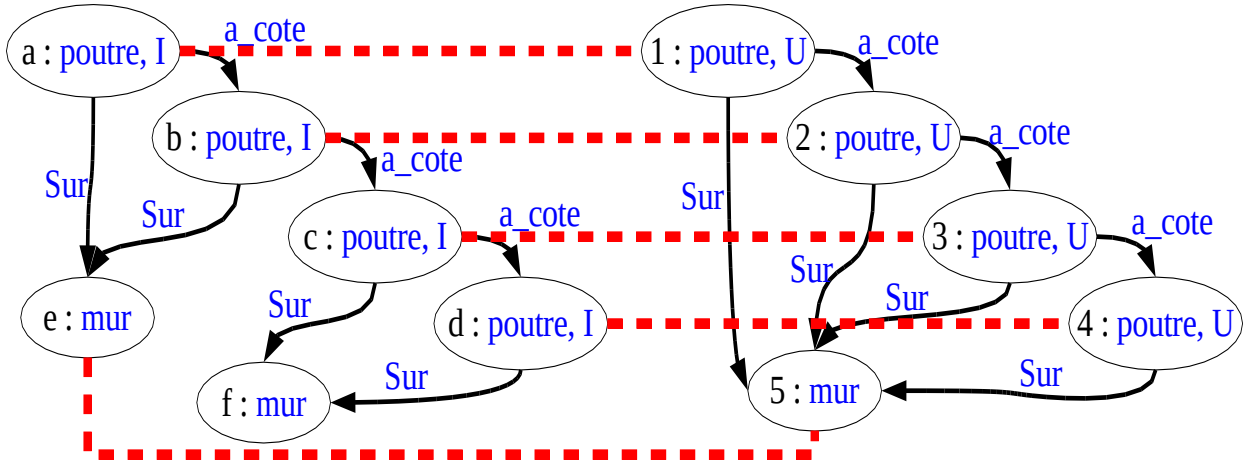
Appariement (1/2)

- Besoin de mettre les éléments en correspondance pour comparer les graphes :



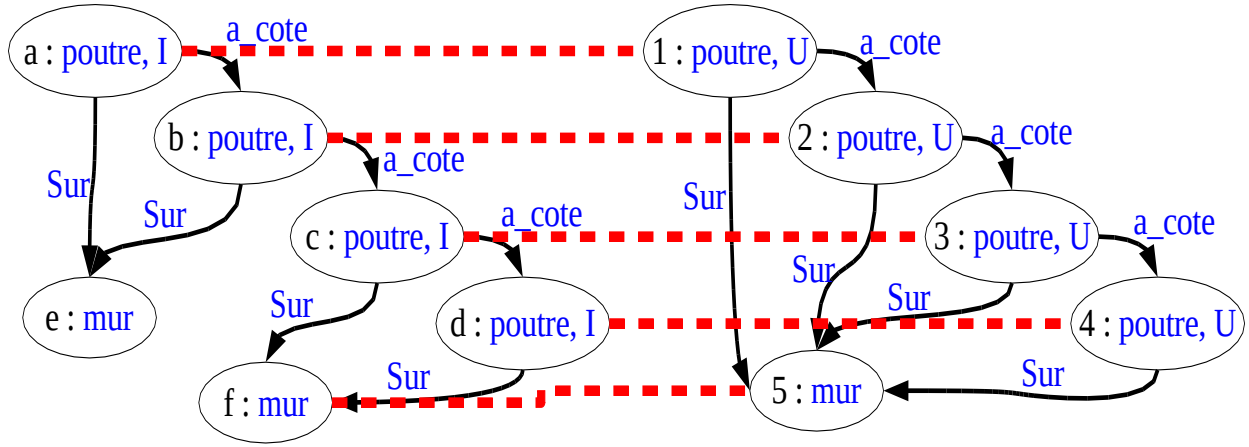
Appariement (1/2)

- Besoin de mettre les éléments en correspondance pour comparer les graphes :



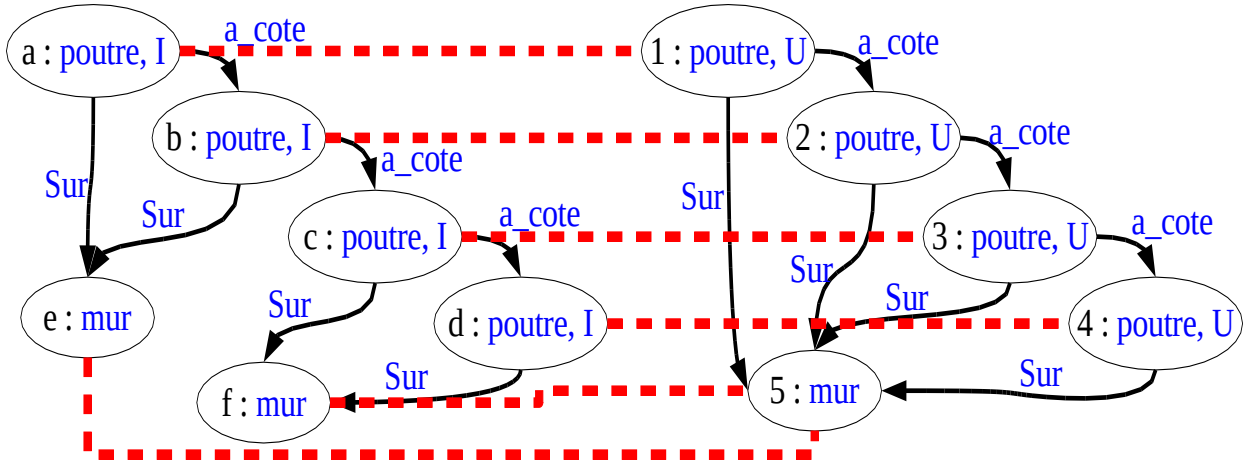
Appariement (1/2)

- Besoin de mettre les éléments en correspondance pour comparer les graphes :



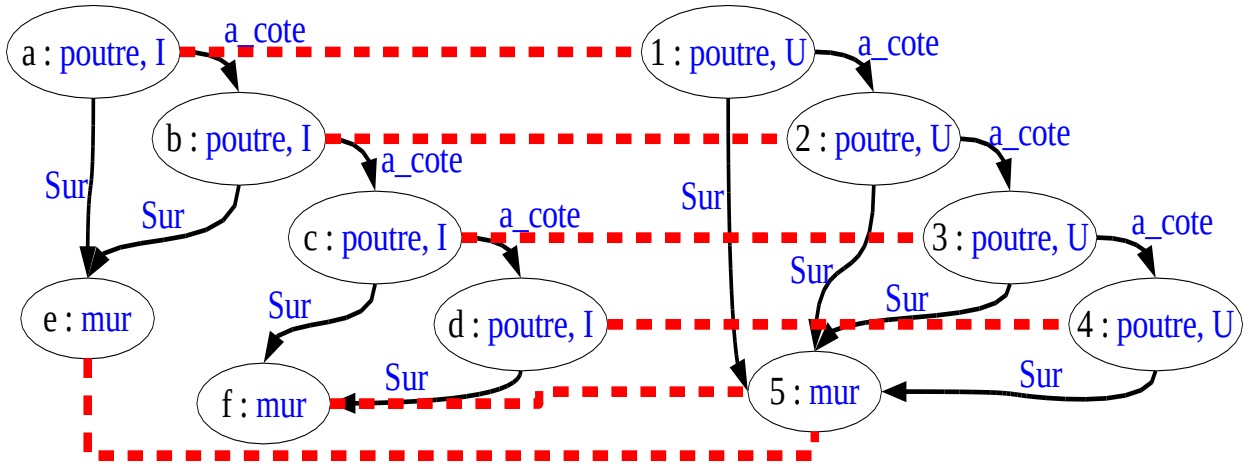
Appariement (1/2)

- Besoin de mettre les éléments en correspondance pour comparer les graphes :



Appariement (2/2)

- A chaque sommet d'un graphe sont associés $0, 1$ ou plusieurs sommets
- Un appariement est une relation $m \subseteq V_1 \times V_2$
- Exemple :

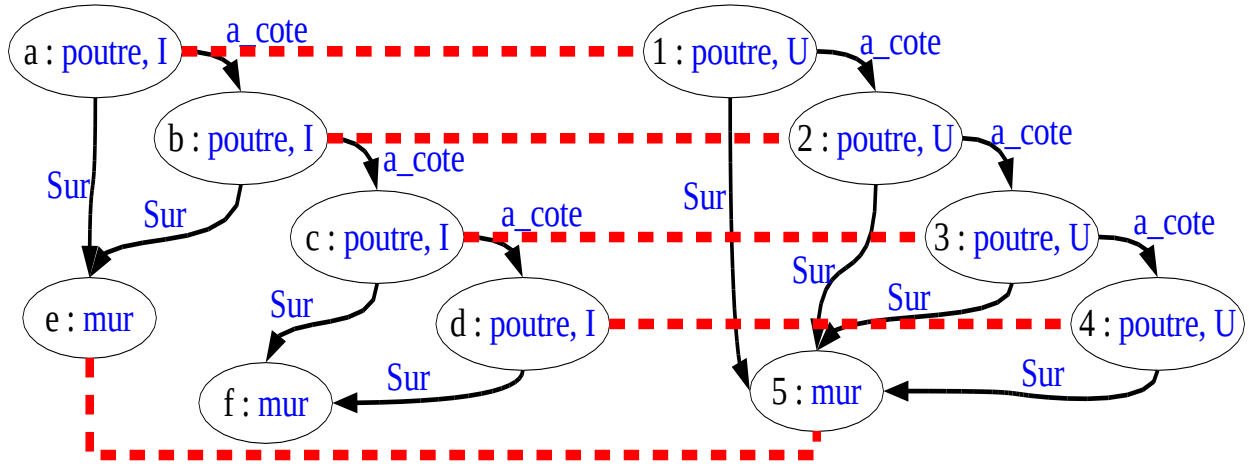


$$m_A = \{(a, 1), (b, 2), (c, 3), (d, 4), (e, 5), (f, 5)\}$$

$\Rightarrow$ Eclatement (*split*) du sommet 5 vers les sommets e et f

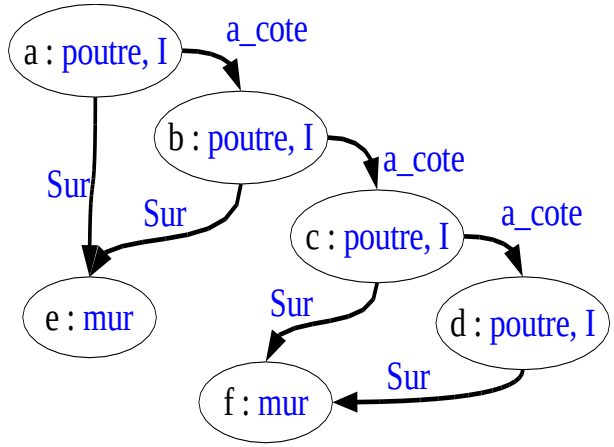
Caractéristiques communes

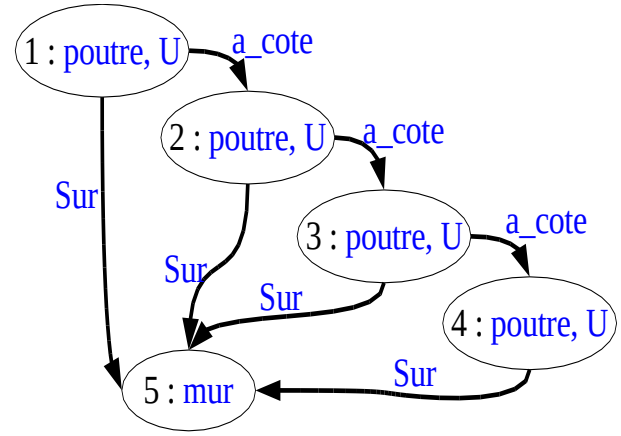
- Caractéristiques communes à G_1 et G_2 par rapport à un appariement m :



$$
\begin{aligned}
descr(G_1) \sqcap_m descr(G_2) \quad \doteq \quad & \{(v,l) \in r_{V_1} \mid \exists v' \in m(v),\ (v',l) \in r_{V_2}\} \\
\cup \quad & \{(v,l) \in r_{V_2} \mid \exists v' \in m(v),\ (v',l) \in r_{V_1}\} \\
\cup \quad & \{(v_i,v_j,l) \in r_{E_1} \mid \exists v'_i \in m(v_i), \exists v'_j \in m(v_j)\ (v'_i,v'_j,l) \in r_{E_2}\} \\
\cup \quad & \{(v_i,v_j,l) \in r_{E_2} \mid \exists v'_i \in m(v_i), \exists v'_j \in m(v_j)\ (v'_i,v'_j,l) \in r_{E_1}\}
\end{aligned}
$$

Caractéristiques communes : exemple



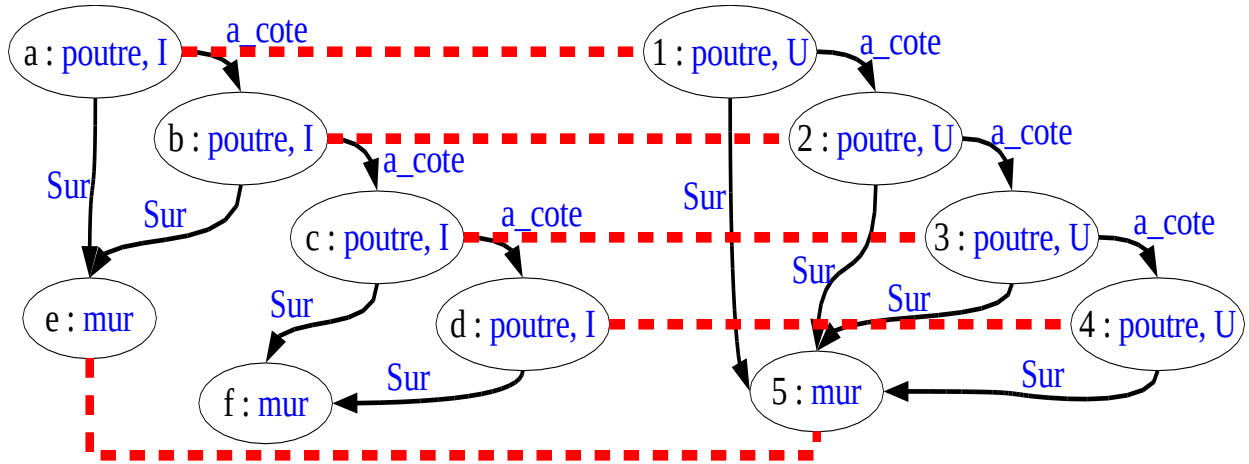


$$m_A = \emptyset$$

$\{(a, poutre), (a, I), (b, poutre), (b, I),$
$(c, poutre), (c, I)(d, poutre), (d, I),$
$(e, mur), (f, mur), (a, b, a\_cote),$
$(b, c, a\_cote), (c, d, a\_cote),$
$(a, e, Sur), (b, e, Sur),$
$(c, f, Sur), (d, f, Sur)\}$

$\{(1, poutre), (1, I), (2, poutre), (2, I),$
$(3, poutre), (3, I), (4, poutre), (4, I)$
$(5, mur), (1, 2, a\_cote),$
$(2, 3, a\_cote), (3, 4, a\_cote),$
$(1, 5, Sur), (2, 5, Sur),$
$(3, 5, Sur), (4, 5, Sur)\}$

Caractéristiques communes : exemple

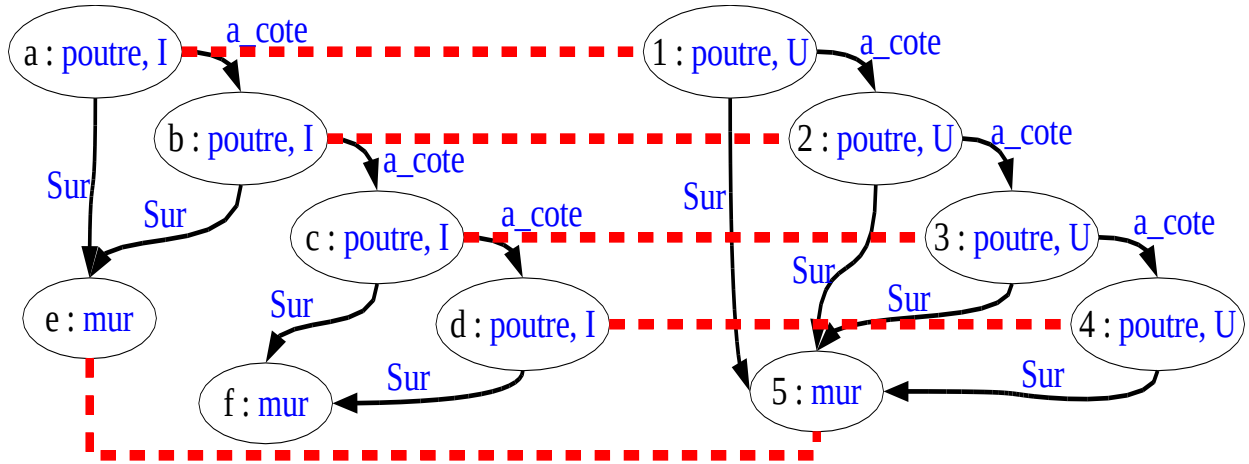


$$m_A = \{(a,1),(b,2),(c,3),(d,4),(e,5)\}$$

$\{(a, poutre),(a, I),(b, poutre),(b, I),$
$(c, poutre),(c, I)(d, poutre),(d, I),$
$(e, mur),(f, mur),(a, b, a\_cote),$
$(b, c, a\_cote),(c, d, a\_cote),$
$(a, e, Sur),(b, e, Sur),$
$(c, f, Sur),(d, f, Sur)\}$

$\{(1, poutre),(1, I),(2, poutre),(2, I),$
$(3, poutre),(3, I),(4, poutre),(4, I)$
$(5, mur),(1, 2, a\_cote),$
$(2, 3, a\_cote),(3, 4, a\_cote),$
$(1, 5, Sur),(2, 5, Sur),$
$(3, 5, Sur),(4, 5, Sur)\}$

Caractéristiques communes : exemple

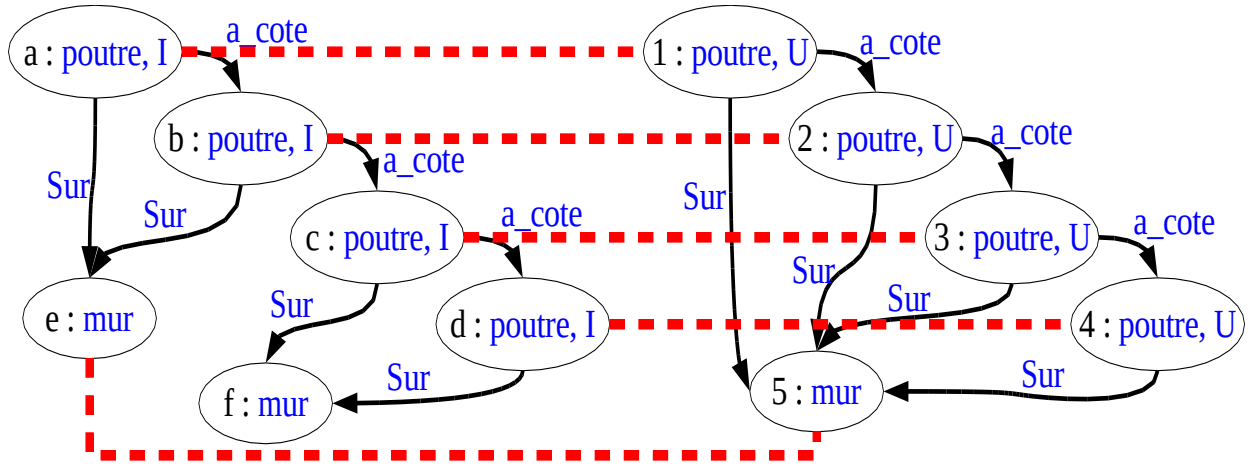


$$m_A = \{(a,1),(b,2),(c,3),(d,4),(e,5)\}$$

$\{(a, poutre),(a, I),(b, poutre),(b, I),$
$(c, poutre),(c, I)(d, poutre),(d, I),$
$(e, mur),(f, mur),(a, b, a\_cote),$
$(b, c, a\_cote),(c, d, a\_cote),$
$(a, e, Sur),(b, e, Sur),$
$(c, f, Sur),(d, f, Sur)\}$

$\{(1, poutre),(1, I),(2, poutre),(2, I),$
$(3, poutre),(3, I),(4, poutre),(4, I)$
$(5, mur),(1, 2, a\_cote),$
$(2, 3, a\_cote),(3, 4, a\_cote),$
$(1, 5, Sur),(2, 5, Sur),$
$(3, 5, Sur),(4, 5, Sur)\}$

Caractéristiques communes : exemple

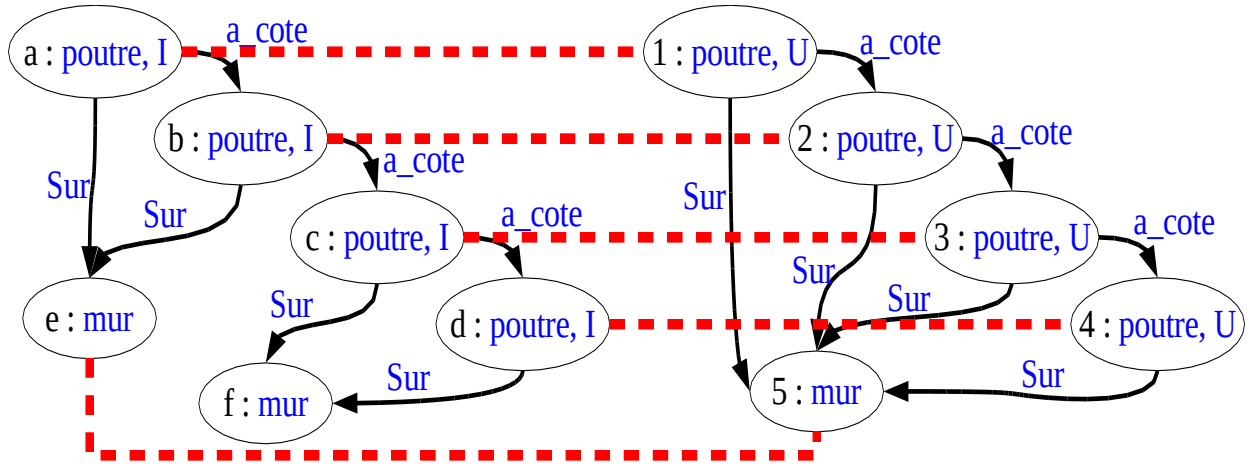


$$m_A = \{(a,1),(b,2),(c,3),(d,4),(e,5)\}$$

$\{(a, poutre), (a, I), (b, poutre), (b, I),$
$(c, poutre), (c, I)(d, poutre), (d, I),$
$(e, mur), (f, mur), (a, b, a\_cote),$
$(b, c, a\_cote), (c, d, a\_cote),$
$(a, e, Sur), (b, e, Sur),$
$(c, f, Sur), (d, f, Sur)\}$

$\{(1, poutre), (1, I), (2, poutre), (2, I),$
$(3, poutre), (3, I), (4, poutre), (4, I)$
$(5, mur), (1, 2, a\_cote),$
$(2, 3, a\_cote), (3, 4, a\_cote),$
$(1, 5, Sur), (2, 5, Sur),$
$(3, 5, Sur), (4, 5, Sur)\}$

Caractéristiques communes : exemple

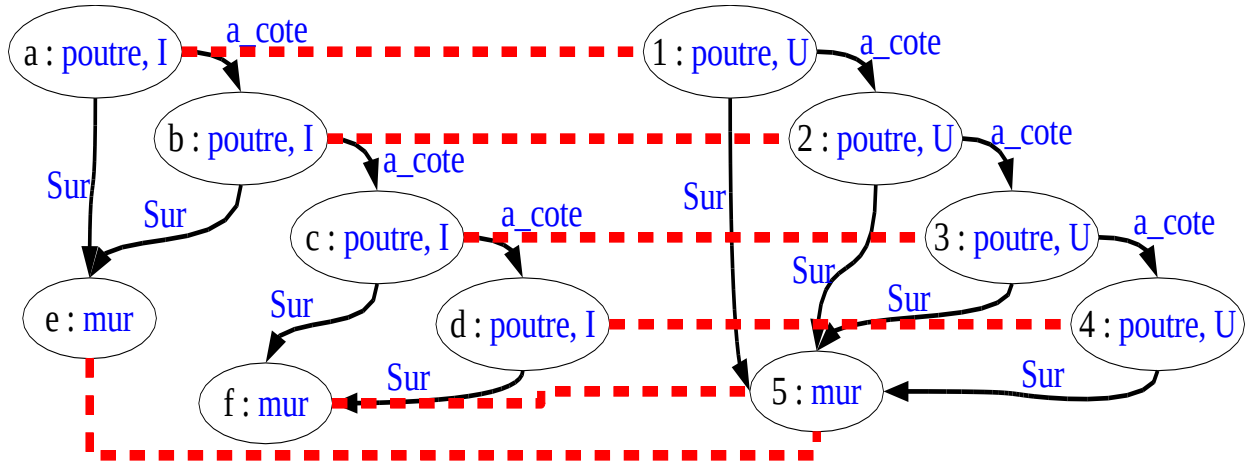


$$m_A = \{(a,1),(b,2),(c,3),(d,4),(e,5)\}$$

$$descr(G_1) \sqcap_{mA} descr(G_2) =$$

$$\{(a, poutre),(b, poutre),(c, poutre),(d, poutre),(e, mur),(a, b, a\_cote),(b, c, a\_cote),(c, d, a\_cote),(a, e, Sur),(b, e, Sur)\} \cup \{(1, poutre),(2, poutre),(3, poutre),(4, poutre),(5, mur),(1, 2, a\_cote),(2, 3, a\_cote),(3, 4, a\_cote),(1, 5, Sur),(2, 5, Sur)\}$$

Caractéristiques communes : exemple



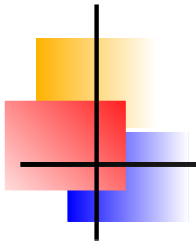
$$m_{A'} = \{(a,1),(b,2),(c,3),(d,4),(e,5),(f,5)\}$$

$$descr(G_1) \sqcap_{mA'} descr(G_2) =$$

$\{(a, poutre), (b, poutre),$
$(c, poutre), (d, poutre),$
$(e, mur), (a, b, a\_cote),$
$(b, c, a\_cote), (c, d, a\_cote),$
$(a, e, Sur), (b, e, Sur),$
$(f, mur), (c, f, Sur)$
$(d, f, Sur)\}$

$\bigcup$

$\{(1, poutre), (2, poutre),$
$(3, poutre), (4, poutre),$
$(5, mur), (1, 2, a\_cote),$
$(2, 3, a\_cote), (3, 4, a\_cote),$
$(1, 5, Sur), (2, 5, Sur),$
$(3, 5, Sur), (4, 5, Sur)\}$



Similarité (2)

- Similarité de deux graphes

$$sim_1(G_1, G_2) = \frac{f(descr(G_1) \sqcap_m descr(G_2))}{f(descr(G_1) \cup descr(G_2))}$$

- Problème
  - Ne prends pas en compte les éclatements

$$splits(m) \doteq \{(v, s_v) |\ v \in V_1 \cup V_2,\ s_v = m(v),\ |m(v)| \geq 2\}$$

- Similarité de deux graphes G_1 et G_2 :

$$sim_m(G_1, G_2) = \frac{f(descr(G_1) \sqcap_m descr(G_2)) - g(splits(m))}{f(descr(G_1) \cup descr(G_2))}$$

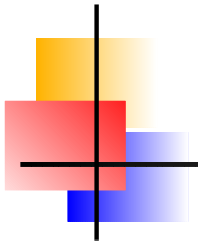
Connaissances de similarité

- f et g permettent d'introduire des connaissances de similarité propres au domaine considéré
- Exemples :
  - Rendre le concept *"être une poutre"* plus important que celui d' *"avoir une forme en U"*
  - Eclater le sommet 1 est plus gênant qu'éclater le sommet 5

- On suppose dans la suite

$$f(E) = \sum_{e \in E} poids(e)$$

$$g(E) = \sum_{(s,E_s) \in E} (\mid E_s \mid -1)$$



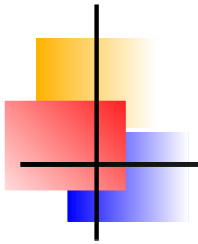
Similarité maximum

- $sim_m(G_1, G_2)$ est définie par rapport à un appariement
  - La similarité maximum entre deux graphes sera la similarité générée par le meilleur appariement :

$$sim(G_1, G_2) = \max_{m \subseteq V_1 \times V_2} \frac{f(descr(G_1) \sqcap_m descr(G_2)) - g(splits(m))}{f(descr(G_1) \cup descr(G_2))}$$

- On cherche l'appariement qui maximise $score$

$$score(m) = f(descr(G_1) \sqcap_m descr(G_2)) - g(splits(m))$$



Calcul de la similarité

- Complexité du problème
  - Plus général que l'isomorphisme de graphe, de sous-graphe, que le k-partitionnement, que max-clique...

$$\Rightarrow \text{NP-Complet}$$

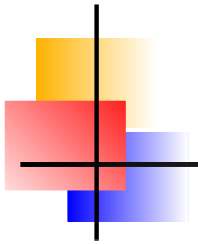
- $2^{|V_1 \times V_2|}$ appariements possibles

Approches envisagées

- CSP valué
  - Variables : V_1
  - Domaine des variables : $\wp(V_2)$
  - Contraintes :
    - Retrouver chaque caractéristiques de $descr(G_1) \cup descr(G_2)$
    - Les sommets ne doivent pas être éclatés
    - Poids définis par f et g

      Domaine des variables : $2^{|V_2|}$ éléments !

- Approche complète "séparation et évaluation"
- Glouton
- Recherche taboue réactive

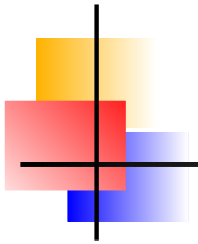


Recherche "séparation et évaluation"

- Structuration de l'espace de recherche en treillis
- Fonction d'évaluation f_{max} :
  - Evalue la borne maximale du $score$ d'un ens. d'appariement incluant m

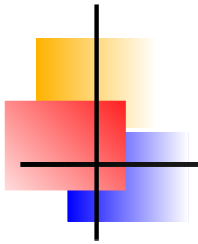
$$\forall m' \supseteq m, f_{max} \geq score(m')$$

- Problème
  - La fonction $score$ n'est pas monotone
  - Cette approche ne permet l'appariement de graphes de seulement quelques sommets



Algorithme glouton (1/2)

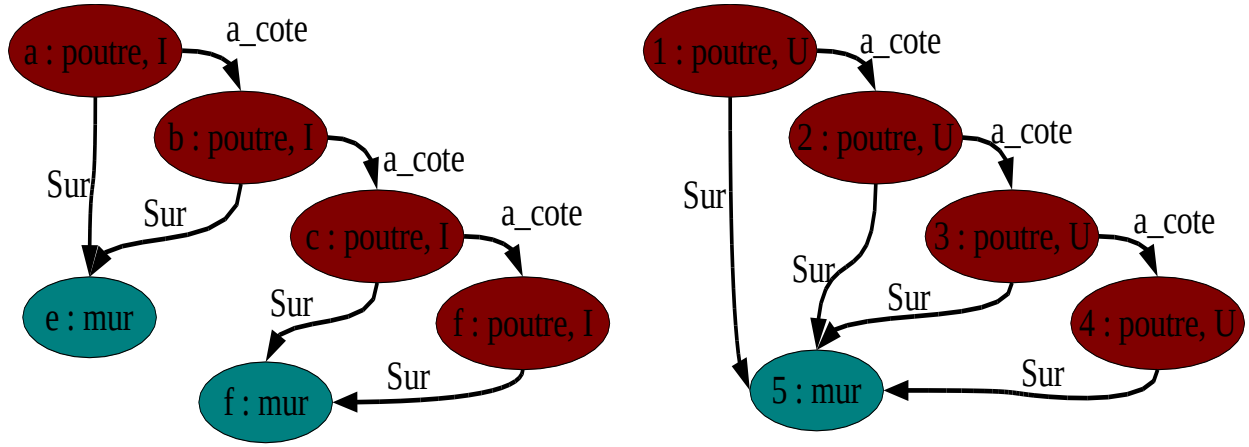
- Construire rapidement un appariement m de *bonne* qualité
  - Pas de garantie d'optimalité
  - Complexité en $\mathcal{O}((|V_1| \times |V_2|)^2)$
  - Non déterministe $\Rightarrow$ peut être exécuté plusieurs fois
    - Conserver le meilleur appariement trouvé
    - Proposer à l'utilisateur un ensemble d'appariements de *bonne* qualité



Algorithme glouton (2/2)

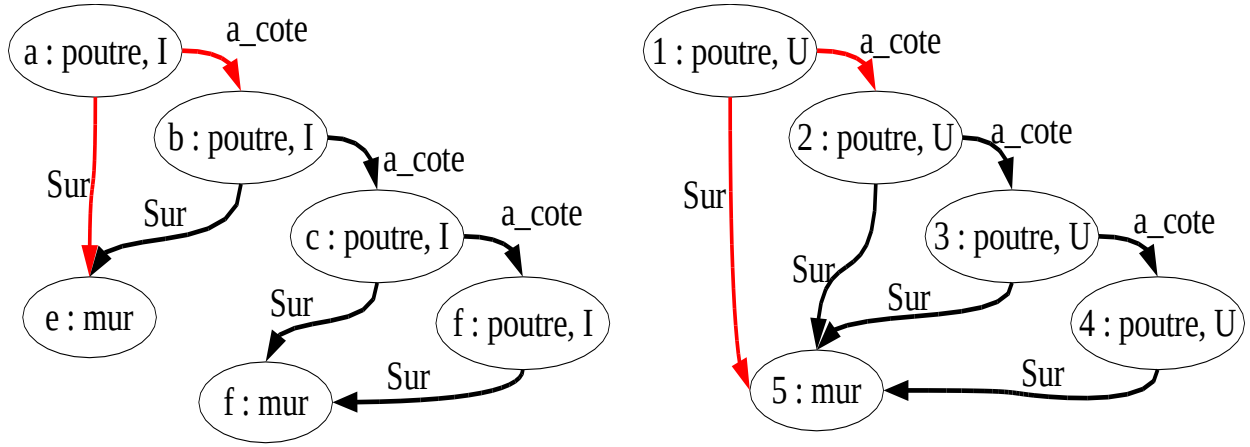
- Déroulement de l'algorithme
  1. Démarrer avec un appariement vide $m = \emptyset$
  2. Construire l'ens. $cand$ des couples (u, v) qui maximisent $score(m \cup \{(u, v)\})$
  3. Construire l'ens. $cand' \subseteq cand$ en départageant les ex-aequo par anticipation
     - Caractéristiques d'arcs pouvant par la suite être retrouvées
  4. Choisir aléatoirement $c \in cand'$ et l'ajouter à m
  5. Continuer en (2) tant que l'appariement est amélioré

Algorithme glouton : exemple



1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$

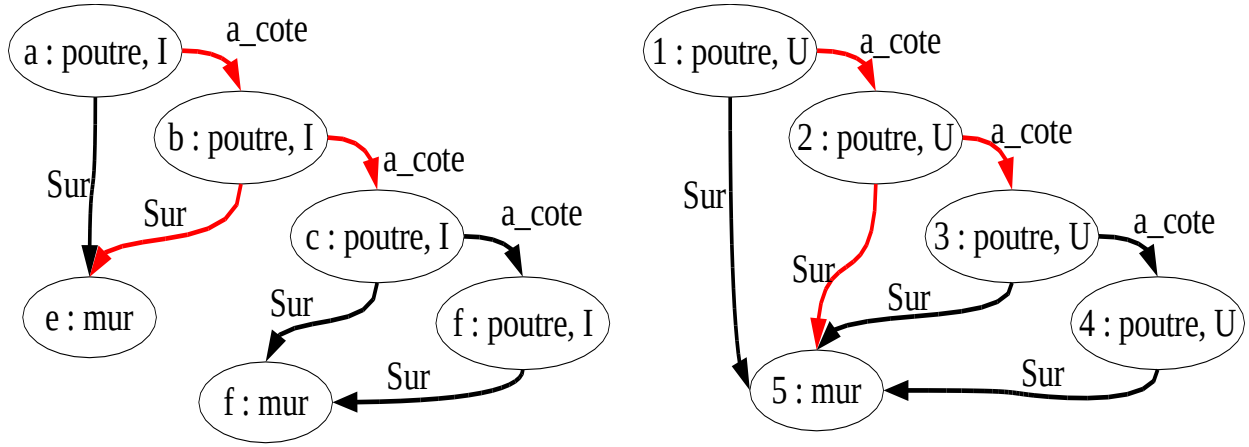
Algorithme glouton : exemple



1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$

- Anticipation $(a, 1), (d, 4) = 2 + 2$ caractéristiques d'arcs

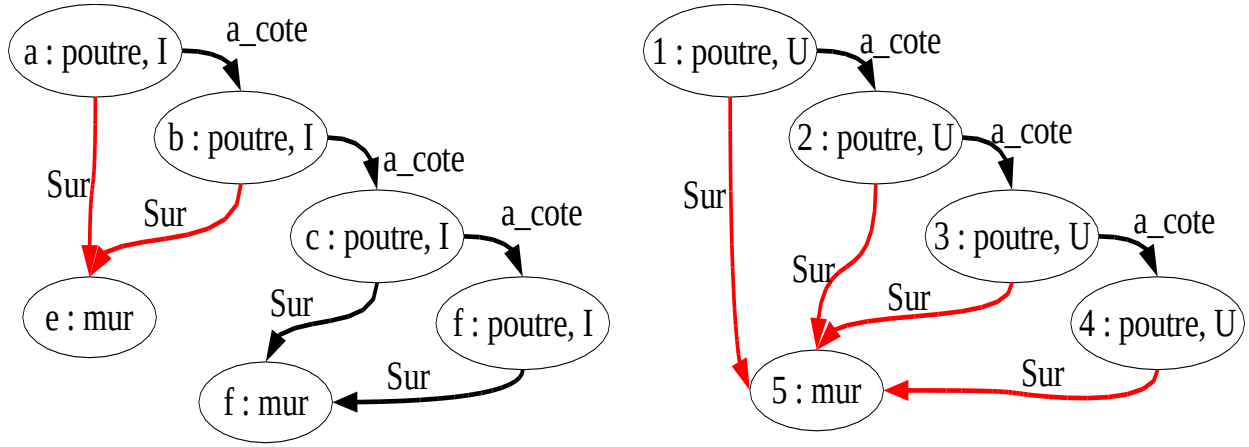
Algorithme glouton : exemple



1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$

- Anticipation $(a, 1), (d, 4) = 2 + 2$ caractéristiques d'arcs
- Anticipation $(b, 2), (b, 3), (c, 2), (c, 3) = 3 + 3$ caractéristiques d'arcs

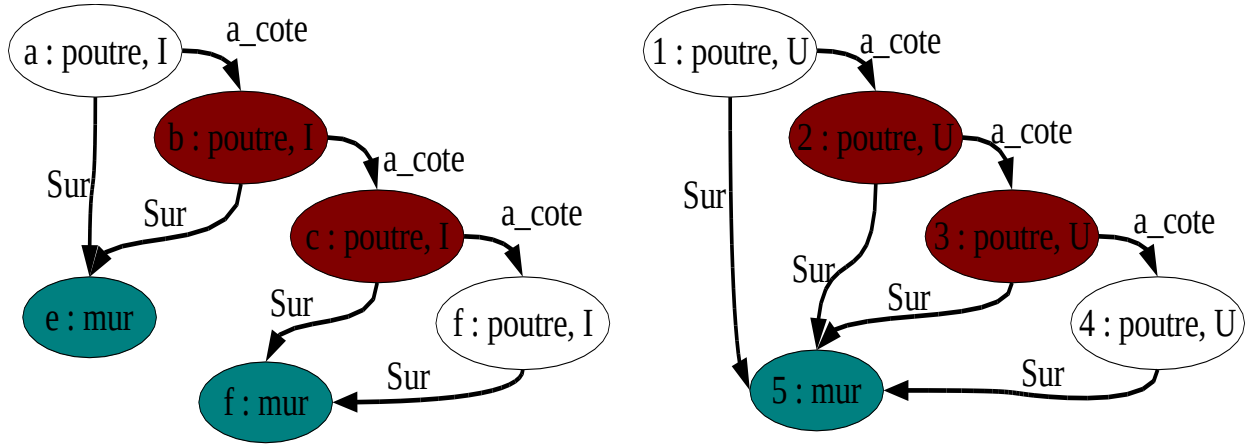
Algorithme glouton : exemple



1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$

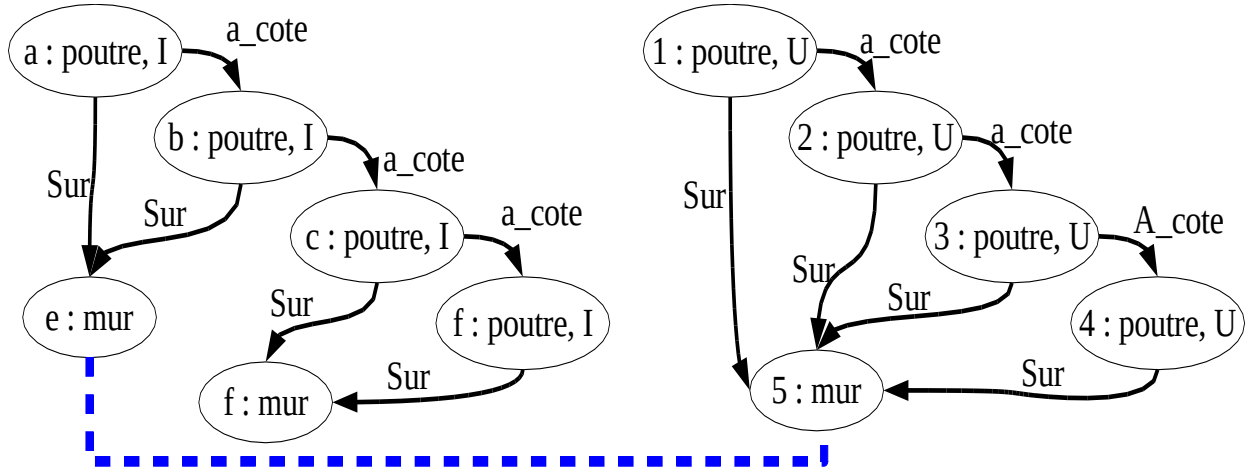
- Anticipation $(a, 1), (d, 4) = 2 + 2$ caractéristiques d'arcs
- Anticipation $(b, 2), (b, 3), (c, 2), (c, 3) = 3 + 3$ caractéristiques d'arcs
- Anticipation $(e, 5), (f, 5) = 2 + 4$ caractéristiques d'arcs

Algorithme glouton : exemple



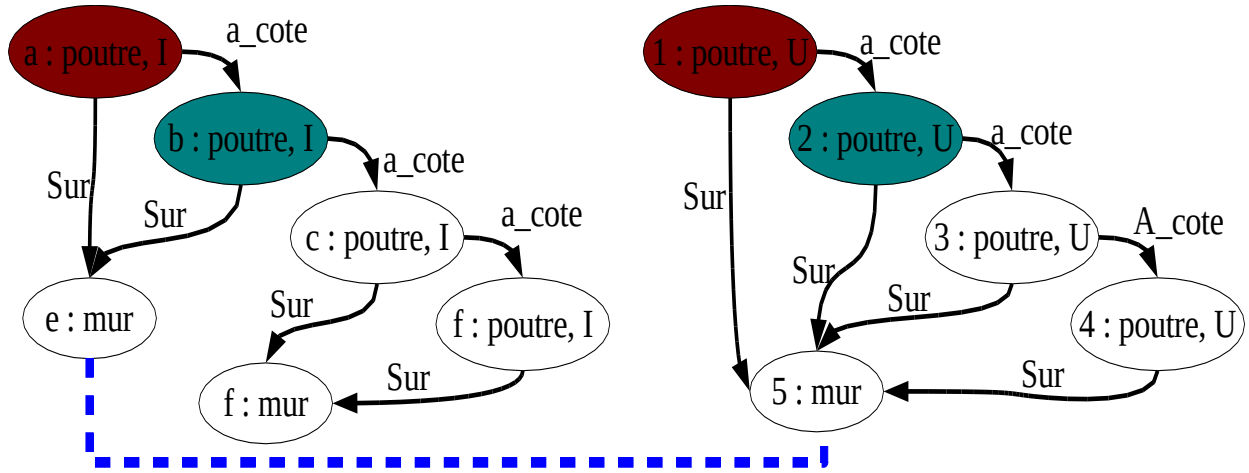
1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$
2. $cand' = \{(b, 2), (b, 3), (c, 2), (c, 3), (e, 5), (f, 5)\}$

Algorithme glouton : exemple



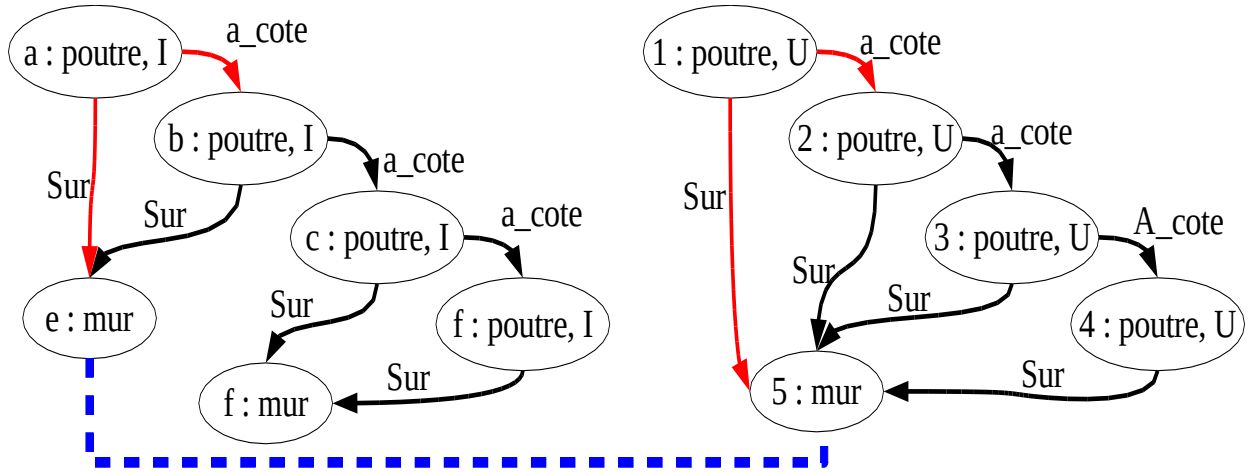
1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$
2. $cand' = \{(b, 2), (c, 3), (e, 5), (f, 5)\}$
3. Choix de $(e, 5) \Rightarrow m = \{(e, 5)\}$

Algorithme glouton : exemple



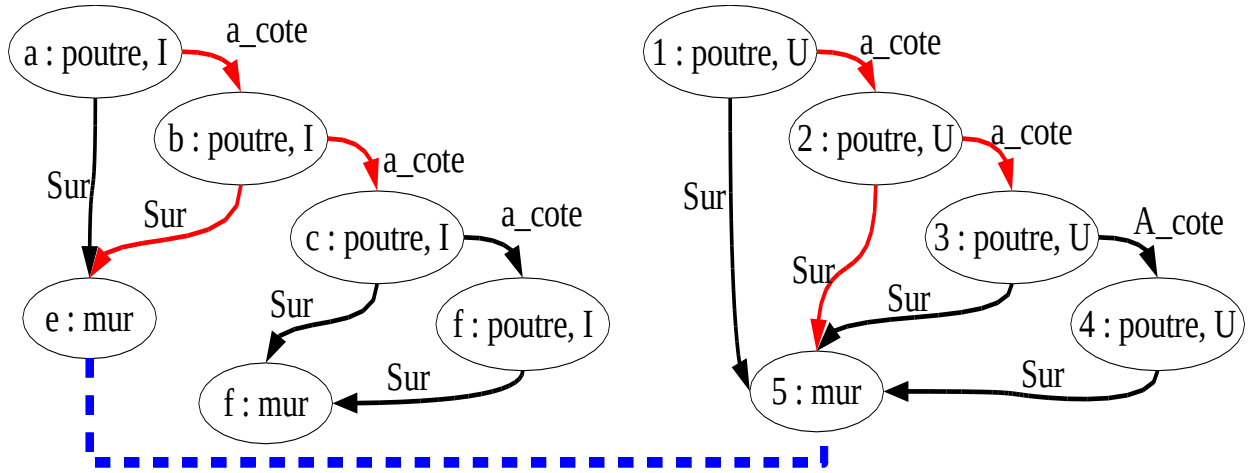
1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$
2. $cand' = \{(b, 2), (c, 3), (e, 5), (f, 5)\}$
3. Choix de $(e, 5) \Rightarrow m = \{(e, 5)\}$
4. $cand = \{(a, 1), (b, 2)\}$

Algorithme glouton : exemple



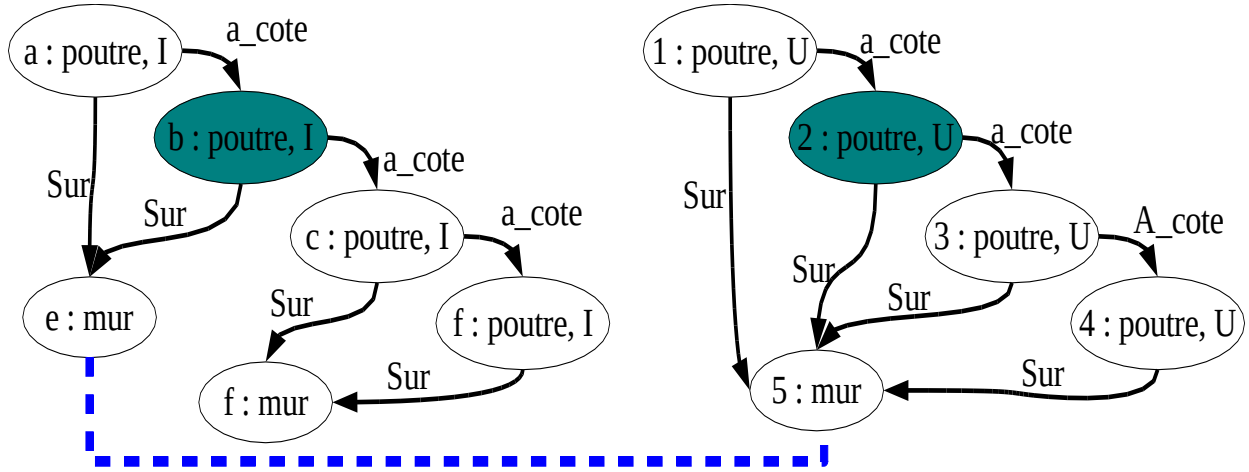
1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$
2. $cand' = \{(b, 2), (c, 3), (e, 5), (f, 5)\}$
3. Choix de $(e, 5) \Rightarrow m = \{(e, 5)\}$
4. $cand = \{(a, 1), (b, 2)\}$

Algorithme glouton : exemple



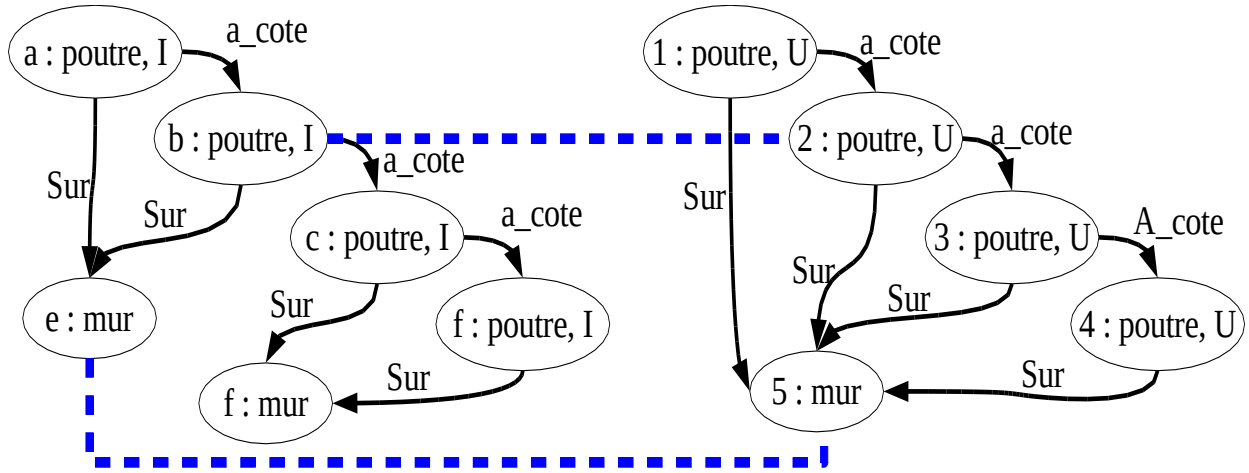
1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$
2. $cand' = \{(b, 2), (c, 3), (e, 5), (f, 5)\}$
3. Choix de $(e, 5) \Rightarrow m = \{(e, 5)\}$
4. $cand = \{(a, 1), (b, 2)\}$

Algorithme glouton : exemple



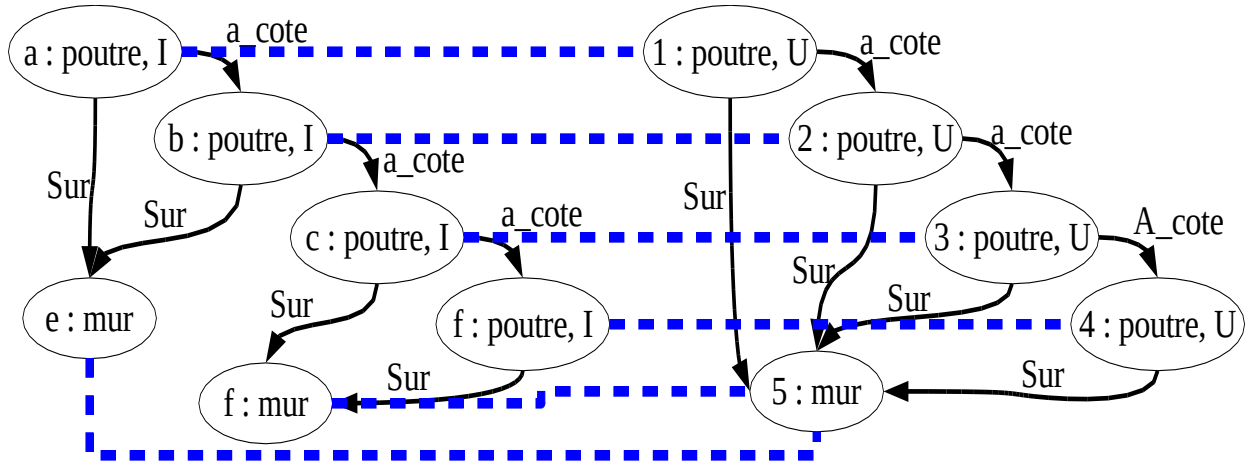
1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$
2. $cand' = \{(b, 2), (c, 3), (e, 5), (f, 5)\}$
3. Choix de $(e, 5) \Rightarrow m = \{(e, 5)\}$
4. $cand = \{(a, 1), (b, 2)\}$
5. $cand' = \{(b, 2)\}$

Algorithme glouton : exemple



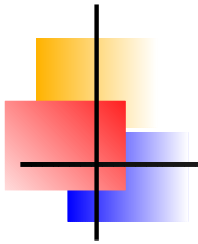
1. $cand = \{a, b, c, d\} \times \{1, 2, 3, 4\} \cup \{e, f\} \times \{5\}$
2. $cand' = \{(b, 2), (c, 3), (e, 5), (f, 5)\}$
3. Choix de $(e, 5) \Rightarrow m = \{(e, 5)\}$
4. $cand = \{(a, 1), (b, 2)\}$
5. $cand' = \{(b, 2)\}$
6. Seul $(b, 2)$ est dans $cand' \Rightarrow m = \{(b, 2), (e, 5)\}$

Algorithme glouton : exemple



- Fin de l'exécution

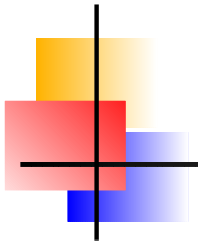
$$m = \{(a, 1), (b, 2), (c, 3), (d, 4), (e, 5), (f, 5)\}$$



Algorithme glouton : résultats

- Tests réalisés sur des graphes de différents types générés aléatoirement

- La difficulté du problème varie en fonction
  - Du poids des $splits$ par rapport aux poids des caractéristiques
  - Du nombre moyen d'étiquettes par sommets et arcs
  - De la taille des ensembles d'étiquettes
  - De la connectivité
  - De la répartition des arcs sur les sommets



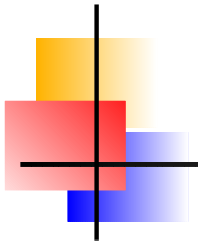
Recherche taboue : principe

- Algorithme $glouton \Rightarrow$ Maximum local

- Explorer le voisinage par recherche locale

  - Mouvements : ajout/suppression d'un couple de sommet

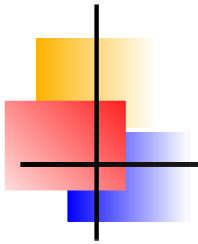
  - Liste taboue : derniers couples ajoutés/supprimés
    $\Rightarrow$ Diversifier la recherche



Recherche taboue : résultats

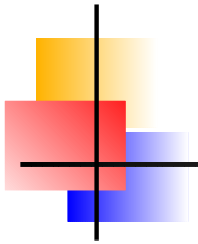
- Les appariements de l'algorithme $glouton$ sont améliorés

- Mais problème de la longueur de la liste taboue
  - Trop courte, la diversification n'est pas suffisante
  - Trop longue, le voisinage des maxima locaux ne sont pas suffisamment explorés
  - La taille optimale varie d'une instance à l'autre



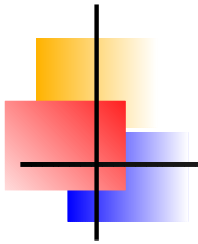
Recherche taboue réactive

- [Battiti & Protasi 01], rendre la longueur de la liste taboue dynamique
  - Lorsque la recherche doit être diversifiée
    $\Rightarrow$ la longueur est augmentée
  - Lorsque la recherche est suffisamment diversifiée
    $\Rightarrow$ la longueur est réduite
  - Détecter les redondances
    - Utilisation d'une table de hashage
    - Collision $\Rightarrow$ Diversifier
    - Pas de collision depuis n itérations $\Rightarrow$ Intensifier



Recherche taboue réactive : résultats

- Par rapport à l'algorithme Tabou *classique*
  - La diversification est meilleure
  - Tabou réactif est équivalent ou meilleur à 4 Tabous classiques avec des tailles différentes $(10, 15, 20, 25)$
  - Les paramètres optimaux (fréquence de diversification, taux d'accroissement...) sont les mêmes pour toutes les instances
  - Le temps de calcul supplémentaire est faible



Conclusions

- Un modèle de représentation de données
  - Les graphes orientés *multi-étiquetés*
- Une mesure de similarité sur ces graphes
  - *Générique* (les fonctions f et g)
  - Autorisant les *fusions/éclatements* de sommets
- Des algorithmes de recherche de similarité
  - Un algorithme de recherche complète par *"séparation et évaluation"*
    - Optimal mais limité à des petits problèmes
  - Un algorithe *Tabou réactif*
    - Qui améliore les résultats d'un algorithme glouton