La représentation des images par les graphes

Aline Deruyver

Laboratoire Icube, équipe BFO

Plan du cours

- Pourquoi les graphes ?
- Rappels et définitions
- De l’image à l ’objet graphe: les graphes d’adjacence de régions
- Simplification de graphes
  - division et fusion de régions
  - fusion pyramidale
- Interprétation d ’images: Mise en correspondance de graphes
  - Isomorphisme de graphes et sous graphes
  - Isomorphisme de sous-graphe avec tolérance d ’erreur
  - Mise en correspondance inexacte: un problème de satisfaction de contraintes

Pourquoi des graphes ?

- Intérêt : représentation compacte, structurée, complète, facile à manipuler
- Applications :
  - Traitement d'images : segmentation, détection de contours
  - Reconnaissance des formes : caractères, objets (bâtiments 2D ou 3D,
  - structures cérébrales, reconnaissance de visages (avec ou sans modèle)
  - Recalage d'images
  - Indexation
  - Interprétation de scènes structurées …

Définition et rappels

- Graphe : G = (X,E)
  - X ensemble des sommets (|X| ordre du graphe)
  - E ensemble des arêtes (|E| taille du graphe)
  - graphe complet (taille $n(n-1)/2$)
  - graphe partiel G = (X,E ') avec E ' partie de E
  - sous-graphe F = (Y,E '), $Y \subseteq X$ et $E' \subseteq E$
  - degré d'un sommet x : d(x) = nombre d'arêtes
  - graphe connexe : pour toute paire de sommets, il existe une chaîne les reliant
  - arbre : graphe connexe sans cycles
  - clique : sous-graphe complet
  - graphe dual (face $\rightarrow$ noeud)
  - graphe aux arêtes (arête $\rightarrow$ noeud)
  - hypergraphe (relations n-aires)
  - graphes pondérés : coûts ou poids sur les arcs

Exemples de graphes

- Graphe relationnel attribué : $G = (X, E, \mu, \nu)$
  - $\mu : X \rightarrow LX$ interpréteur de sommets (LX = attributs des noeuds)

    $\nu : E \rightarrow LE$ interpréteur d'arcs (LE = attributs des arcs)
  - Exemples :
    - graphe des pixels
    - graphe d'adjacence de régions
    - régions de Voronoï / triangulation de Delaunay
    - graphe de primitives avec des relations plus complexes
- Graphe aléatoire : sommets et arcs = variables aléatoires
- Graphe flou : $G = (X, E = X \times X, \mu f, \nu f)$
  - $\mu f : X \rightarrow [0, 1]$

    $\nu f : E \rightarrow [0, 1]$
  - avec $\forall (u, v) \in X \times X \quad \nu f(u, v) \leq \mu f(u) \mu f(v)$ ou $\nu f(u, v) \leq \min[\mu f(u) \mu f(v)]$

Exemples de graphes (suite)

- Graphe d'attributs flous : graphe relationnel attribué avec valeur floue pour chaque attribut
- Graphe hiérarchique :
  - graphes à plusieurs niveaux et graphe biparti entre deux niveaux
  - (approches multi-échelles, regroupements d'objets, ...)
  - Exemples :
    - quadtrees, octrees
    - pyramide de graphes d 'adjacence de régions
- Graphes de raisonnement
  - (graphe de mise en correspondance, arbre de décision,...)

Plan du cours

- Pourquoi les graphes ?
- Rappels et définitions
- De l’image à l ’objet graphe: les graphes d’adjacence de régions
- Simplification de graphes
  - division et fusion de régions
  - fusion pyramidale
- Interprétation d ’images: Mise en correspondance de graphes
  - Isomorphisme de graphes et sous graphes
  - Isomorphisme de sous-graphe avec tolérance d ’erreur
  - Mise en correspondance inexacte: un problème de satisfaction de contraintes

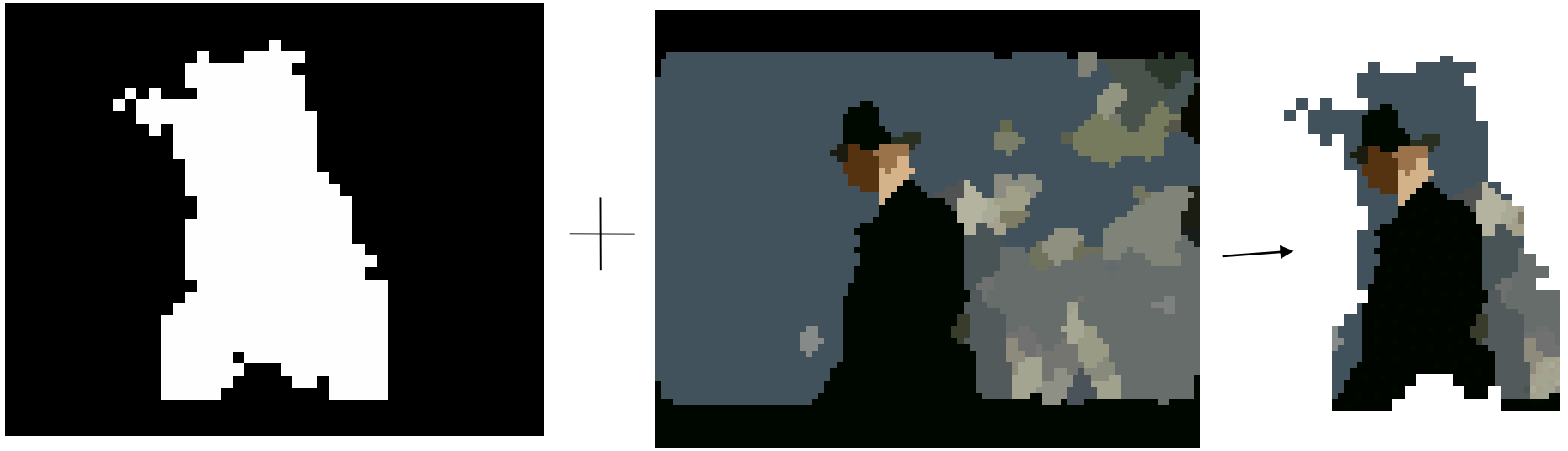
De l'image à l'objet-graphe

- **Segmentation**

Manerbba, Benois-Pineau, Leonardi : *Extraction of foreground objects from a mpeg2 video stream in rough indexing framework*, SPIE '04, San José, California

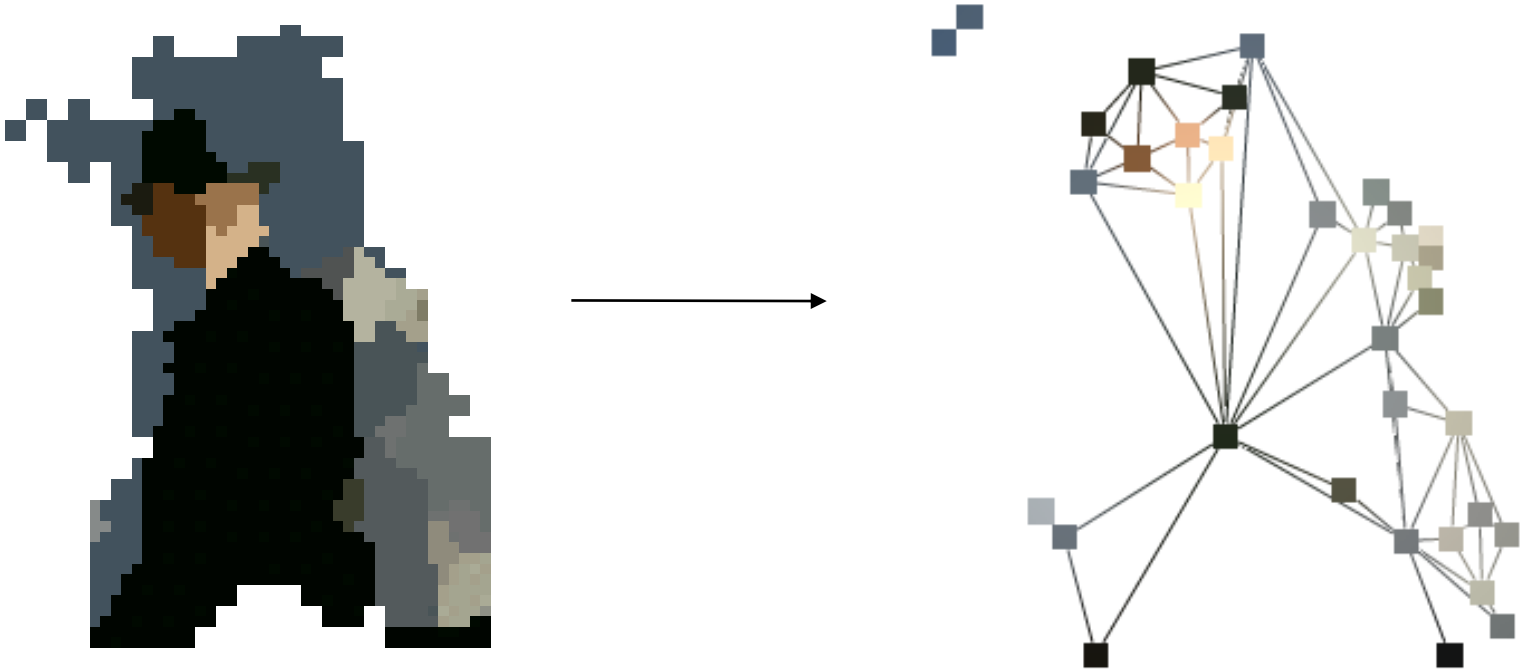
De l’image à l’objet-graphe

- Segmentation
- **Récupération et application du masque de l'objet**



De l’image à l’objet-graphe

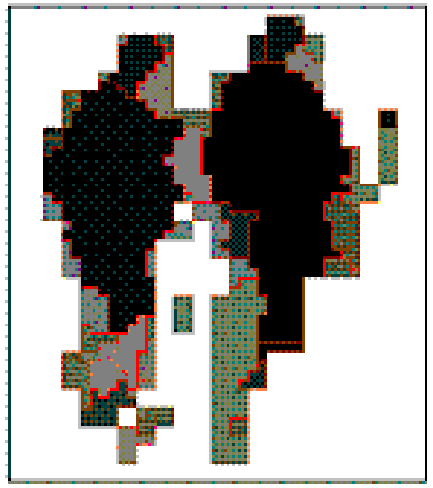
- Segmentation
- Récupération et application du masque de l'objet
- **Construction du graphe d'adjacence**

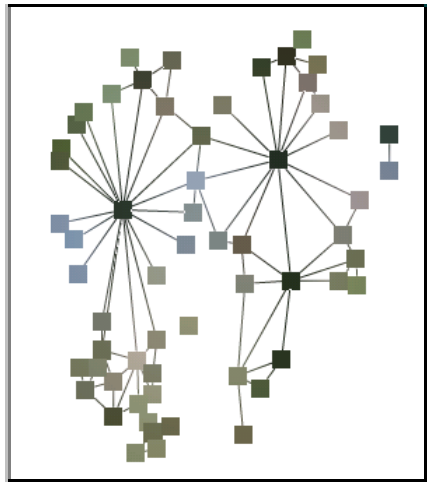


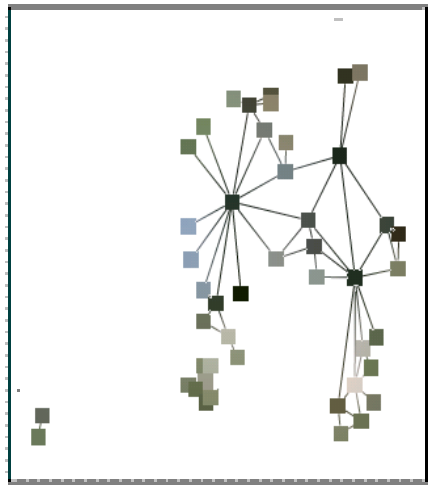
De l’image à l’objet-graphe

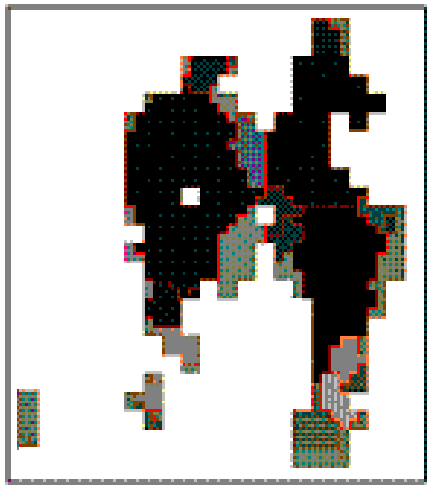
60 régions

46 régions









De l'image à l'objet-graphe

- Défauts

  – Sursegmentation

    ⇨ beaucoup de petites régions

  – Problèmes liés à l'extraction du masque

    ⇨ pixels du fond

    ⇨ régions coupées

Plan du cours

- Pourquoi les graphes ?
- Rappels et définitions
- De l'image à l 'objet graphe: les graphes d'adjacence de régions
- Simplification de graphes
  - division et fusion de régions
  - fusion pyramidale
- Interprétation d 'images: Mise en correspondance de graphes
  - Isomorphisme de graphes et sous graphes
  - Isomorphisme de sous-graphe avec tolérance d 'erreur
  - Mise en correspondance inexacte: un problème de satisfaction de contraintes

Division et fusion de régions dans un graphe

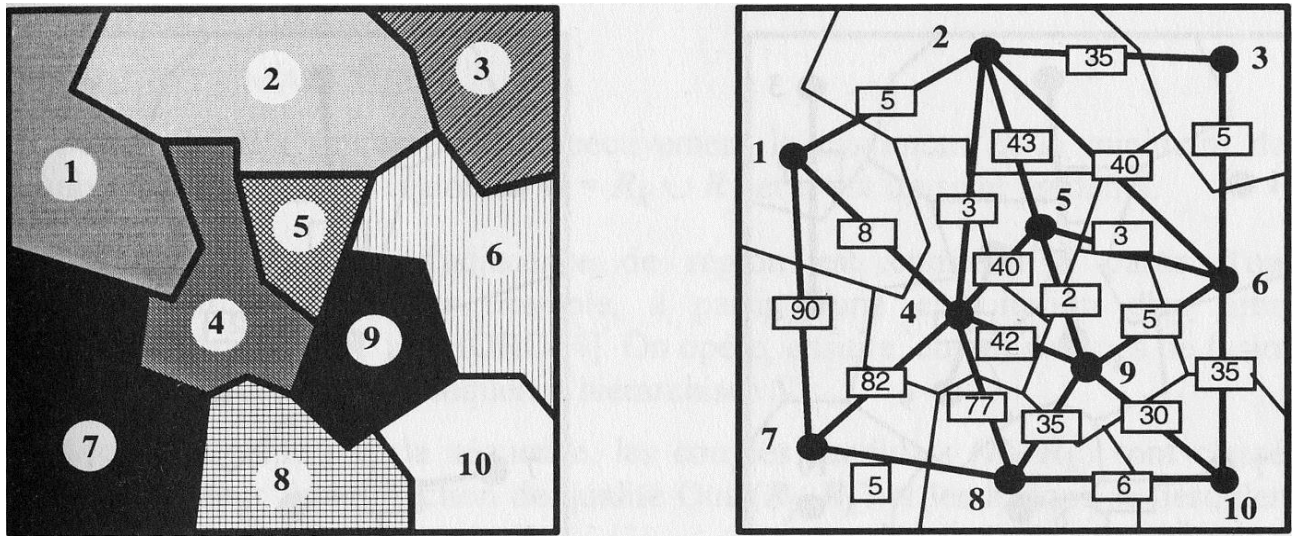
- Procède par éclatement et regroupement de composantes de l'image décrite à l'aide d'un graphe d'adjacence.

- Définition: Un graphe d'adjacence de régions est défini par G=(V,E)
  - V: ensemble des sommets
  - E: ensemble des arêtes

  Chaque sommet vi est associé à une région Ri. Une arête est un couple (vj,vk) de sommets entre 2 régions adjacentes. G est non orienté.

Division et fusion de régions

- Soit une segmentation initiale est représentée par un tel graphe



Division et fusion de régions: algorithme général (Pavlidis 77)

- Module division
  - T := VRAI
  - Tant que t répéter
    - t:= faux
    - pour chaque sommet Rk faire
      - si non(Pred1(Rk)) alors
        - » t:=VRAI
        - » partager Rk en Nk régions suivant un critère donné
        - » mettre à jour le graphe
      - Fsi
    - Fpour
  - ftq

Division et fusion de régions: algorithme général (Pavlidis 77)

- Module de fusion
  - T:= VRAI
  - Tant que t répéter
    - T:= FAUX
    - Pour chaque sommet Rk adjacent à un sommet Rj faire
      - Si Pred2(Rk $\cup$ Rj)=VRAI alors
        - » T:=vrai
        - » Fusionner Rj et Rk
        - » Mettre à jour le graphe
      - Fsi
    - Fpour
  - ftq

Division et fusion de régions: convergence

- La procédure de division converge: le graphe limite est celui correspondant au maillage
- La procédure de fusion converge: le graphe limite possède un seul sommet correspondant à toute l'image
- Pas de cycle possible: toutes les fusions ont lieu avant les divisions

Division et fusion de régions: charge de calcul

- Elle est liée:
  - Au calcul des attributs associés aux sommets et aux arêtes du graphe
  - À la mise à jour du graphe après chaque fusion ou division. Cette mise à jour est généralement moins coûteuse après une fusion.
  - Exemple: si pour un attribut A(Ri) il existe une loi de composition telle que

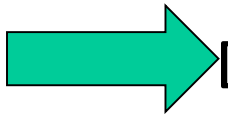
$$A(Ri \cup Rj) = A(Ri) \bigcirc A(Rj)$$

Division et fusion de régions: avantage de l'approche

- L'information obtenue à partir des régions est statistiquement plus fiable que celle calculée à partir des pixels (robustesse vis-à-vis du bruit).

Division et fusion de régions: problème de l'ordre des fusions

- Des ordres de fusion différents conduisent à des segmentations différentes.
- Solution: établir une hiérarchie de critères
- Exemple: associer à chaque arête de G[V,E]un coût de fusion.



Détecter sur le graphe l'arête de moindre coût à chaque itération. La segmentation est alors unique

Division et fusion de régions:sélection et fusion des arêtes candidates

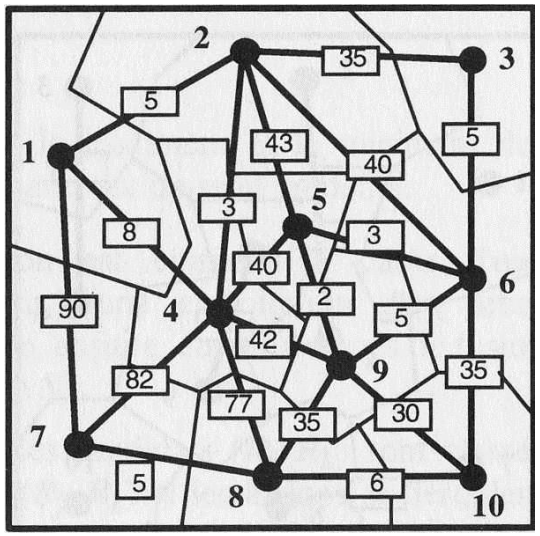
- Répéter
  - Sélectionner un sous-ensemble E’ de E tel que chaque sommet de V n’apparaisse qu’une fois dans E’
  - Fusionner 2 à 2 les régions associées aux éléments de E’
  - Mettre à jour V et E
- Jusqu’à plus de fusion possible

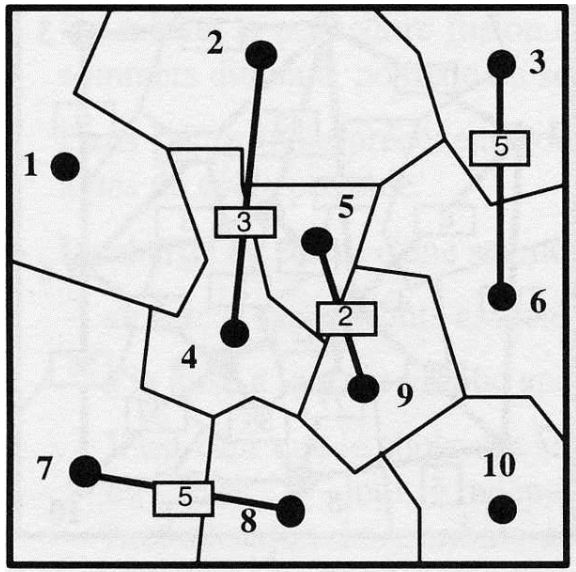
Division et fusion de régions: choix de E’

- But: garantir une bonne segmentation
- Exemples de choix:
  - Soit lc le coût minimale de fusion des arêtes et t un seuil donné. E’ contient les arêtes de coût de fusion < lc+t
  - Test de reconnaissance mutuelle entre les régions: chaque sommet vj sélectionne l’arc vers le sommet adjacent vk de meilleur coût de fusion. Si vj est aussi le meilleur coût pour vk alors (vj,vk) $\in$ E’
  - Extraction itérative: Un seuil t sur le coût de fusion est choisi. Parmi les arêtes de coût <t on sélectionne l’arête (vj,vk) de moindre coût. On supprime les arêtes aboutissant aux sommets vj et vk. On sélectionne parmi les arêtes restantes celle de coût minimal et on recommence jusqu'à avoir traité toutes les arêtes.

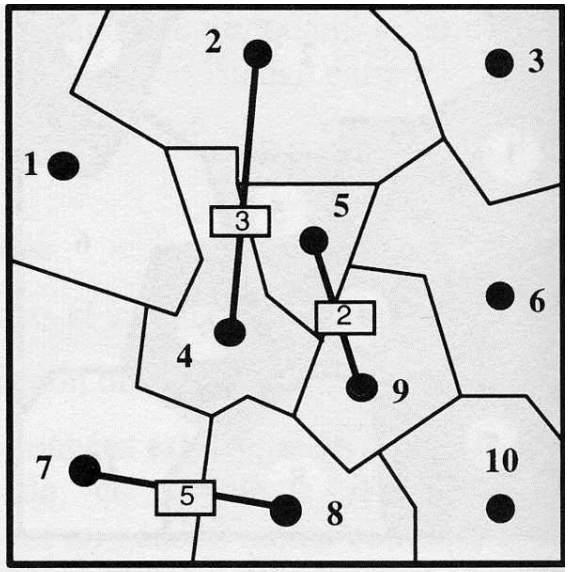
Division et fusion de régions

Graphe d'origine





a) *minimums locaux stricts du coût de fusion*



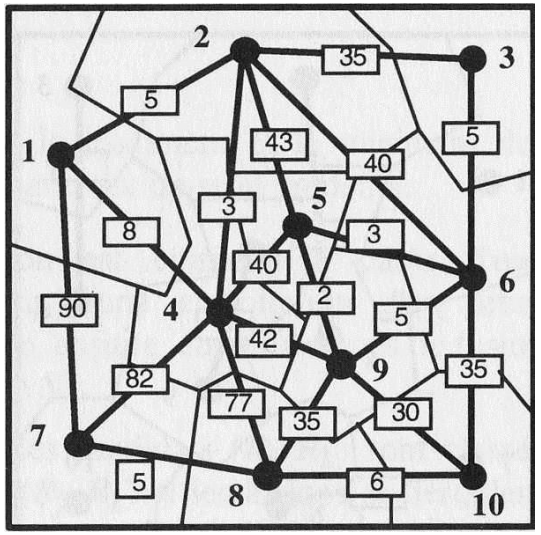
b) *accord mutuel entre sommets*

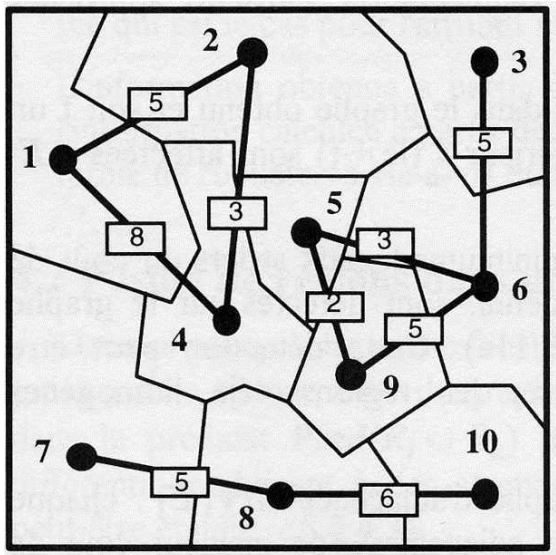
Figure 12.11 : Sélection de E', à partir de la figure 12.10 (seuil fusion = 5)

Résultat obtenu avec la technique de reconnaissance mutuelle. Seuil t=5

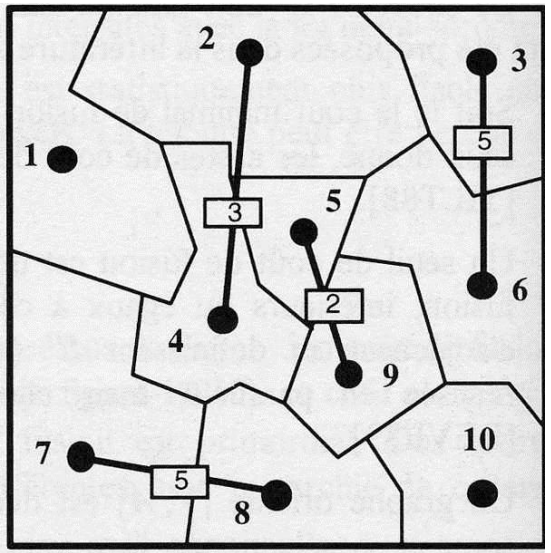
Division et fusion de régions

Graphe d'origine





a) *restriction de* E *aux arêtes de coût de fusion inférieur au seuil (=15)*



b) *résultat de l'extraction itérative des arêtes*

Résultat obtenu par extraction itérative des arêtes.
Seuil=15

Fusion multicritère

- Dans la phase de fusion un couple de région Rk,Rj devra:
  - Vérifier un prédicat d'uniformité ($Pred_i$)
  - Minimiser une fonction de qualité ($Qual_i$)
- Monga et Gagalowicz ont développé un algorithme qui utilise une séquence hiérarchisée de critères de fusion:
  - S= {($Pred_1$, $Qual_1$), …, ($Pred_n$,$Qual_n$)}

- On opère autant d'étapes de fusion qu'il y a de critères dans la séquence.

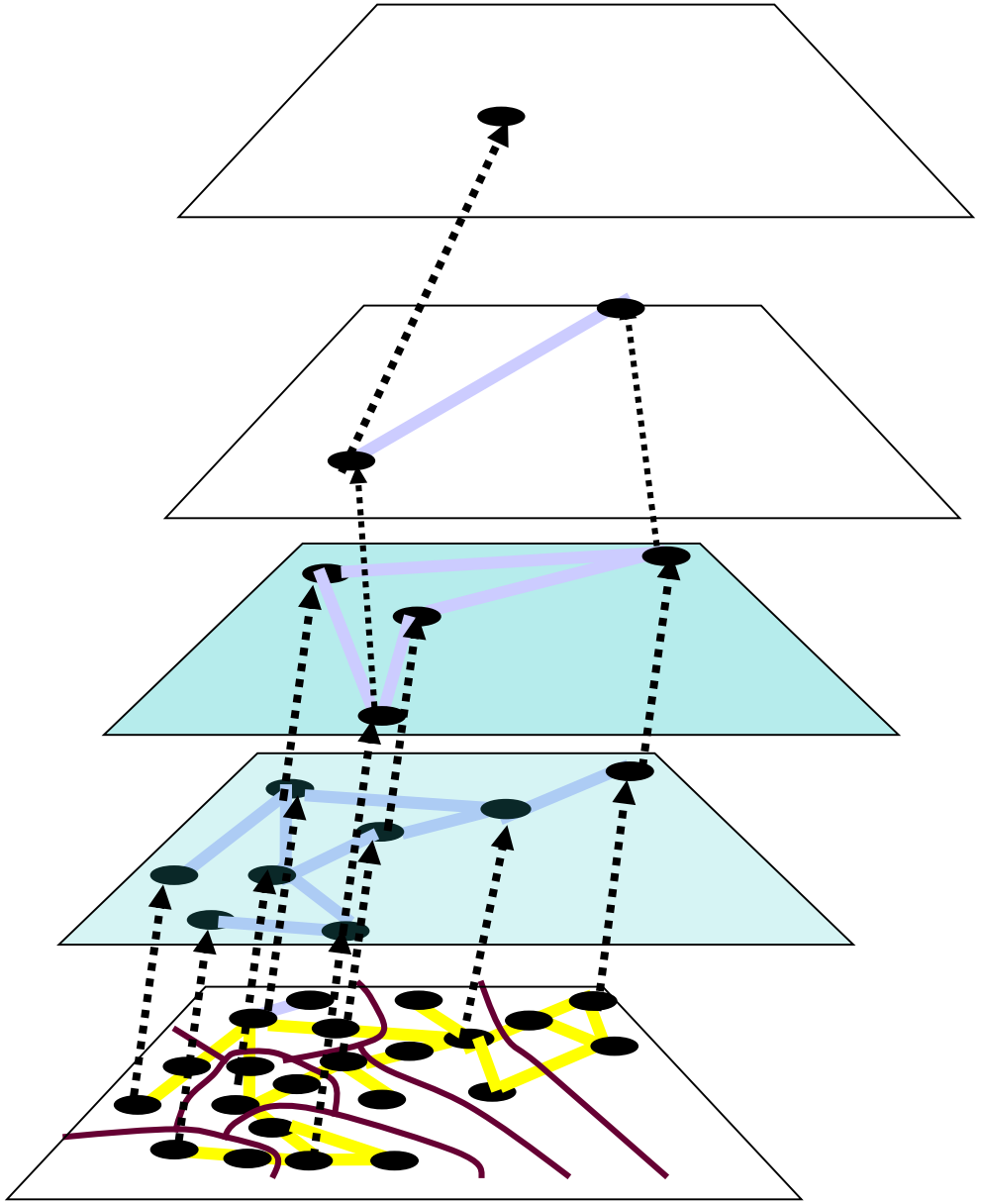
- A chaque étape de la séquence, les couples (Rk,Rj) sont classés suivant la valeur de la fonction de qualité et les fusions se déroulent suivant l'ordre établi.

Pyramides adaptatives

- Principe: partitionnement de l'image par un procédé de type fusion ascendante de régions.

- Initialisation:
  - Un pixel est un nœud du graphe
  - Les arêtes sont les liens de voisinage entre régions

- Processus itératif: à chaque itération on tente de supprimer le maximum de régions par agglomération (réduction de graphe)



Fusion pyramidale

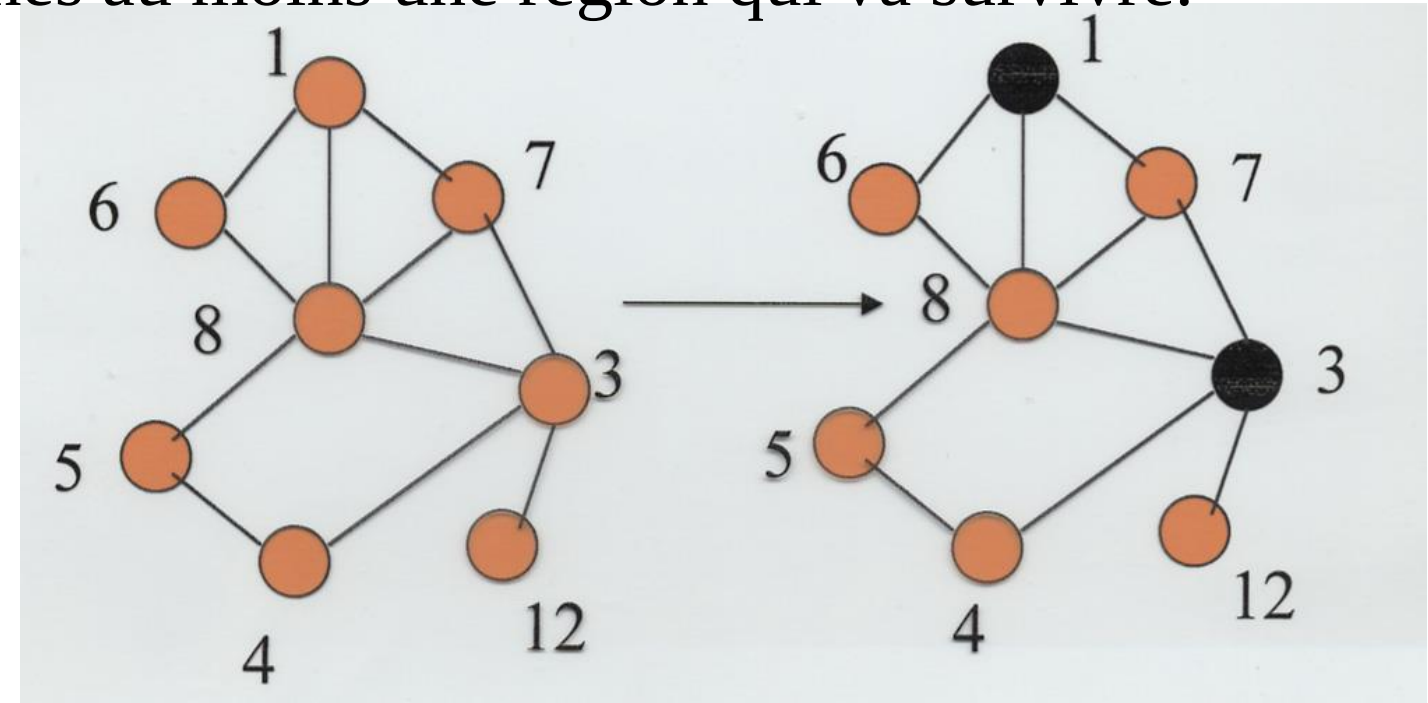
Pyramides adaptatives

- Règles de fusion:
  - Sélectionner un sous-ensemble du graphe: régions survivantes
  - Affectation de toutes les régions non survivantes à une région survivante en conservant la cohérence du graphe
- Les valeurs associées aux nœuds pourront être une mesure de variation (variance). Les régions survivantes seront des minimums locaux (régions homogènes)
- Les régions non survivantes fusionneront avec la région survivante voisine qui leurs ressemble le plus

Pyramides adaptatives

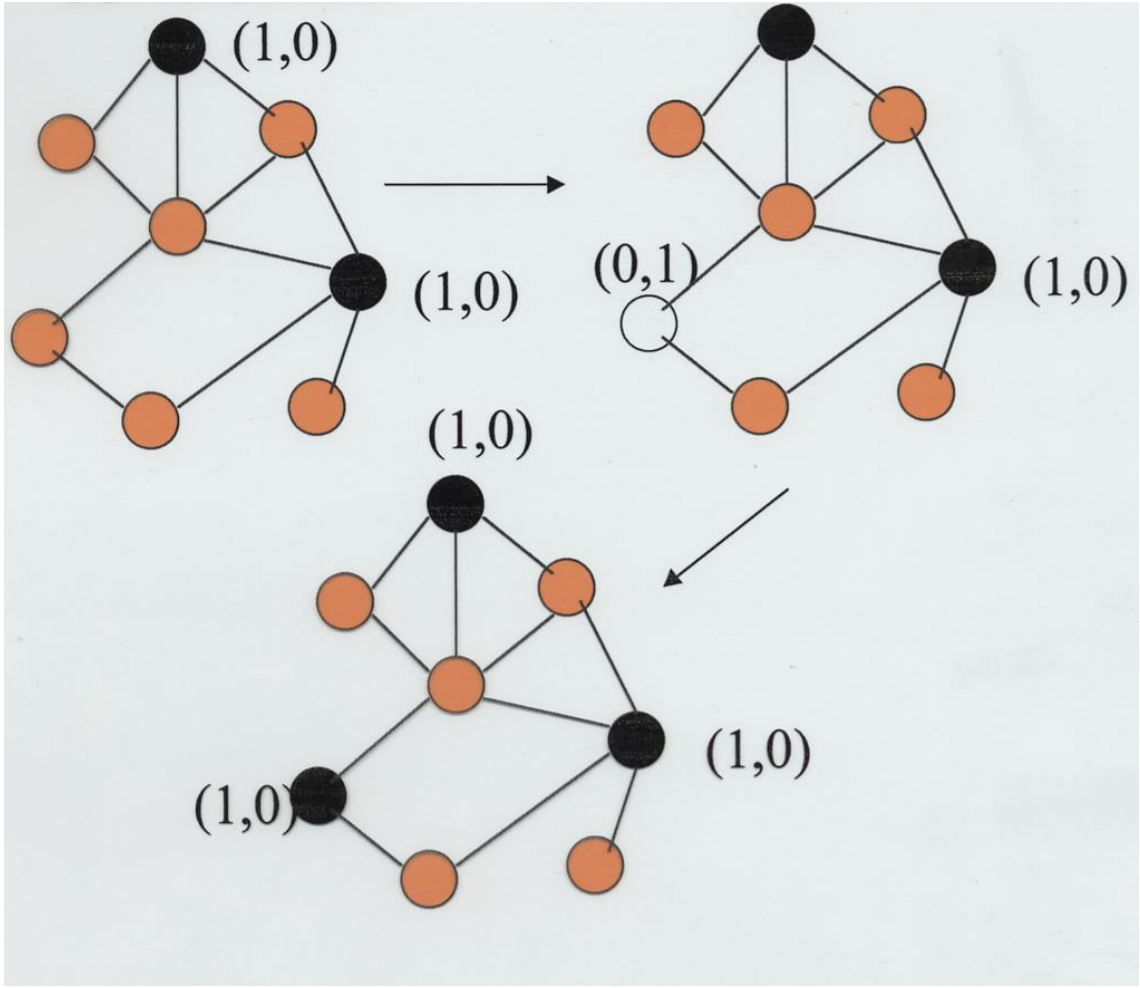
Contraintes:

- deux régions voisines ne peuvent survivre toutes les deux.
- toute région non survivante possède, parmi ses régions voisines au moins une région qui va survivre.



Pyramides adaptatives

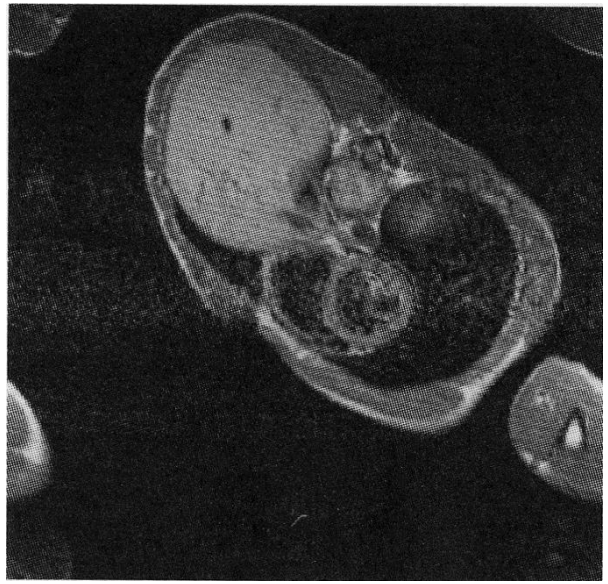
Etape de décimation: satisfaction de la deuxième contrainte:

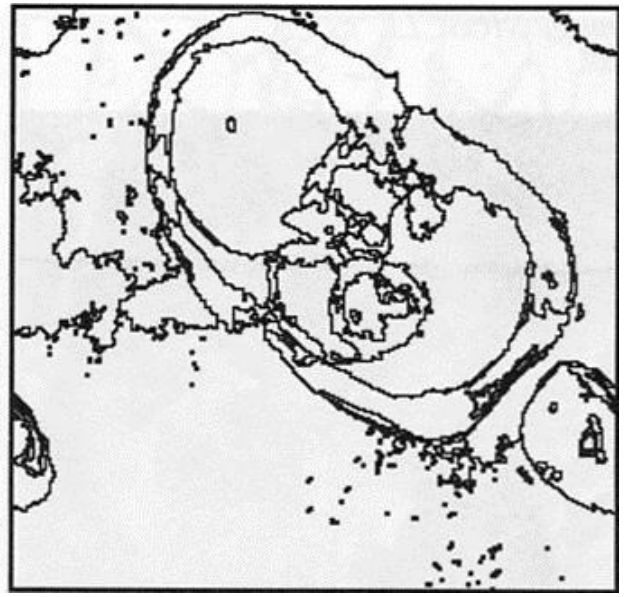


Pyramides adaptatives: neutralisation de certaines régions

- Si P est non survivante et si P’ est la plus similaire des ses régions voisines survivantes alors
  - Si contraste(P,P’) > seuil alors
    - P est neutralisée
  - Sinon P fusionne avec P’

Pyramide adaptatives





Plan du cours

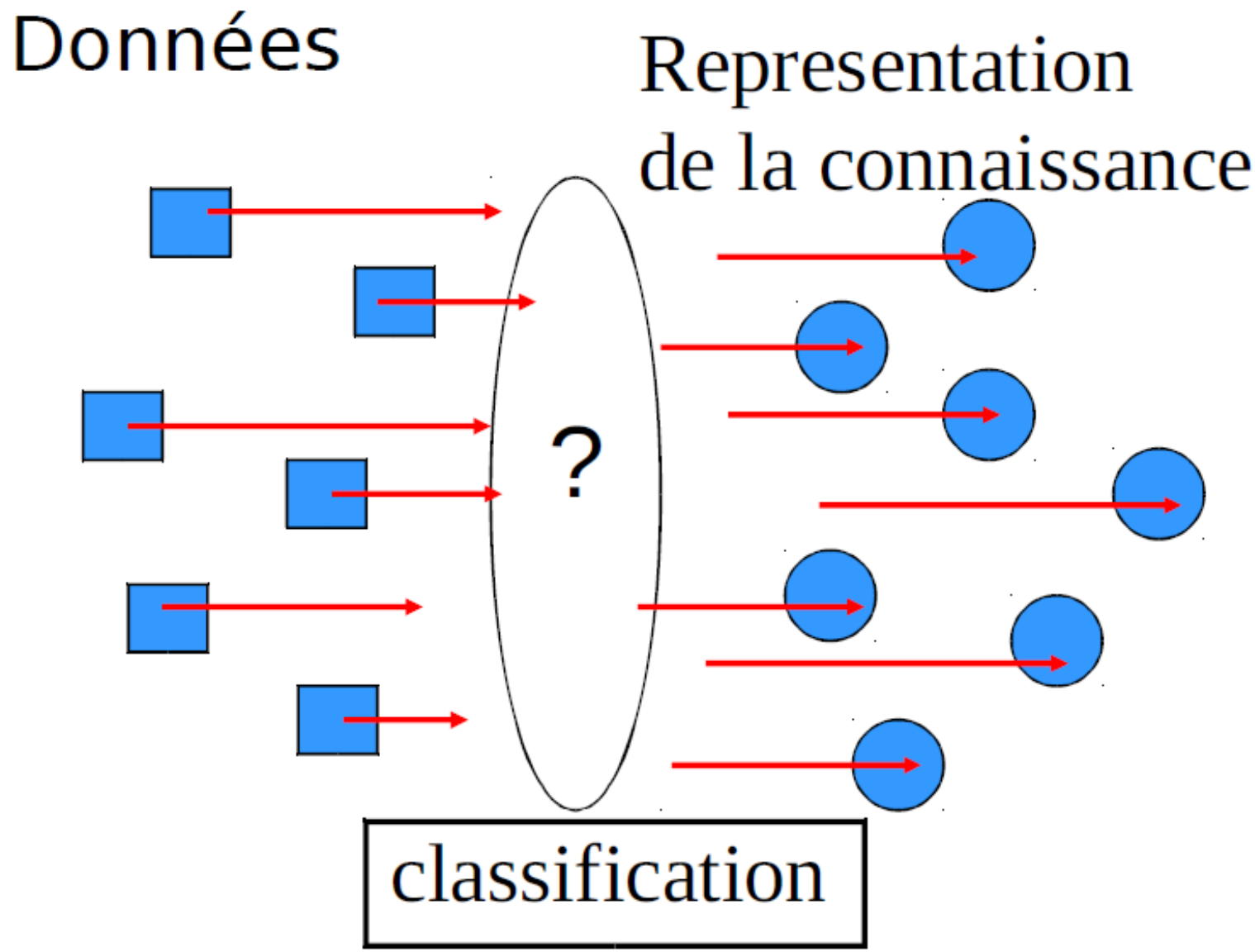
- Pourquoi les graphes ?
- Rappels et définitions
- De l’image à l ’objet graphe: les graphes d’adjacence de régions
- Simplification de graphes
  - division et fusion de régions
  - fusion pyramidale
- Interprétation d ’images: Mise en correspondance de graphes
  - Isomorphisme de graphes et sous graphes
  - Isomorphisme de sous-graphe avec tolérance d ’erreur
  - Mise en correspondance inexacte: un problème de satisfaction de contraintes

Reconnaissance des formes

- Objet : défini par un ensemble de primitives (noeuds du graphe)
- Relations binaires de compatibilité entre primitives (arcs du graphe)
- Clique : sous-ensemble de primitives compatibles 2 à 2 = configuration possible de l'objet
- Reconnaissance par détection de la clique maximale
- Recherche des cliques :
  - Problème NP-complet
  - Pas d'algorithme qui soit dans tous les cas non exponentiel
  - Construction d'un arbre de décision : un noeud de l'arbre = 1 clique du graphe
  - Elagage de l'arbre pour ne pas réengendrer les mêmes cliques

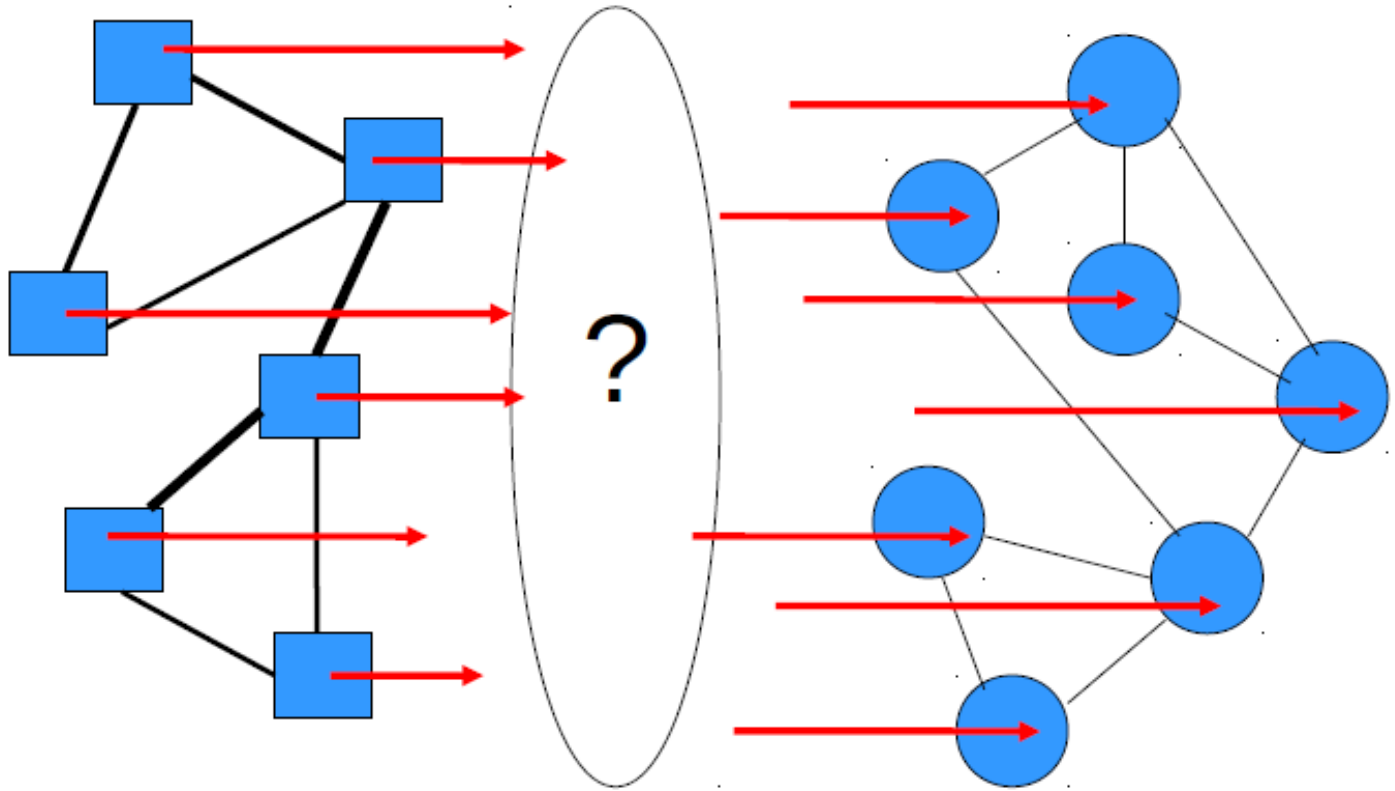
Interprétation d'images

- Tenenbaum et Barrow (1977)
  - Segmentation en régions
  - Construction du graphe d'adjacence des régions
  - Interprétation contrainte par un ensemble de règles sur :
    - 1. les objets (taille, couleur, texture,...)
    - 2. les relations entre objets (au-dessus de, à l'intérieur de, à côté de ...)


Données
Representation
de la connaissance
?
classification

Données

Representation de
La connaissance



Mise en correspondance

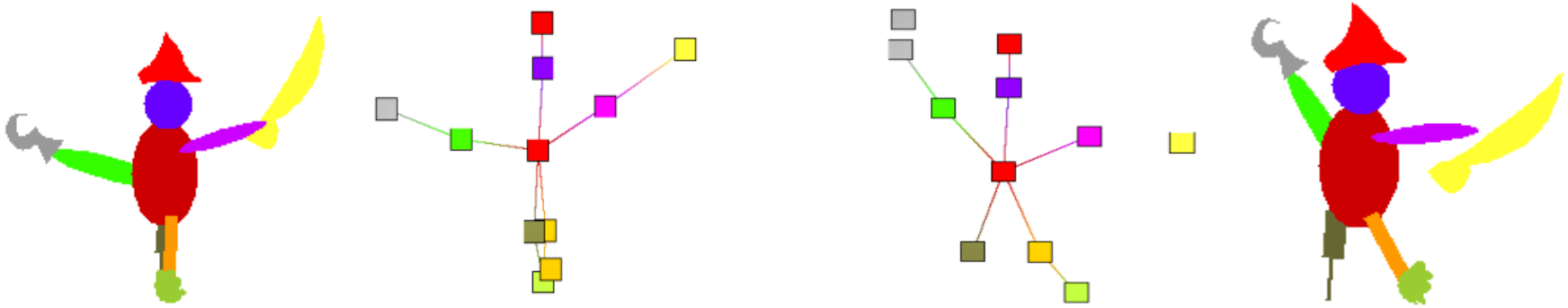
Mise en correspondance

- Soient deux graphes *G(V,E)* et *G'(V',E')*

mise en correspondance de P avec V $\mathbf{P'}$ $\mathbf{V'}$

$$\| |\nu| - |\nu'| \| \leq \varepsilon$$

ε un entier positif



Plan du cours

- Pourquoi les graphes ?
- Rappels et définitions
- De l’image à l ’objet graphe: les graphes d’adjacence de régions
- Simplification de graphes
  - division et fusion de régions
  - fusion pyramidale
- Interprétation d ’images: Mise en correspondance de graphes
  - Isomorphisme de graphes et sous graphes
  - Isomorphisme de sous-graphe avec tolérance d ’erreur
  - Mise en correspondance inexacte: un problème de satisfaction de contraintes

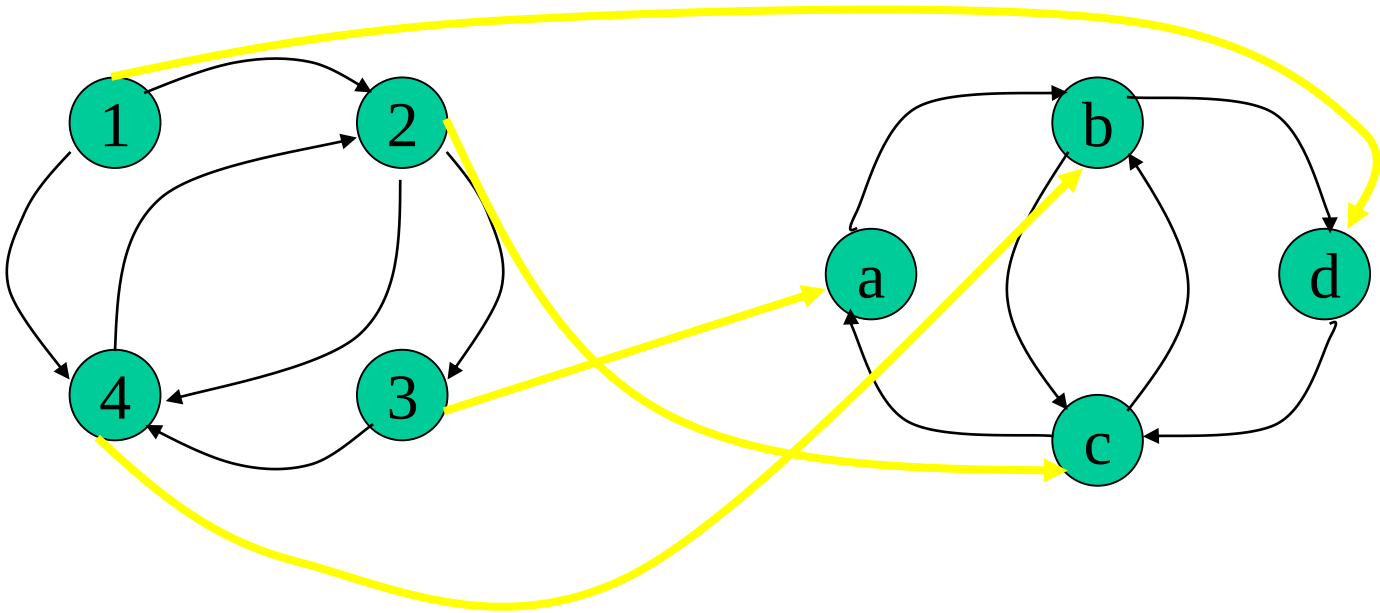
Mise en correspondance (matching) de graphes

- Problème :
  - Graphe(s) modèle(s) (atlas, carte, modèle(s) d'objet(s))
  - Graphe image construit à partir des données
  - Mise en correspondance des deux graphes
  - $G = (X,E, \mu,\nu) \rightarrow ?\ G' = (X',E', \mu', \nu')$
- Isomorphismes de graphes : fonction bijective f : $X \rightarrow X'$
  - $\mu(x) = \mu'(f(x))$

    $\forall e = (x1, x2), \exists e' = (f(x1), f(x2))$ / $\nu(e) = \nu'(e')$ et réciproquement

Isomorphisme de graphes

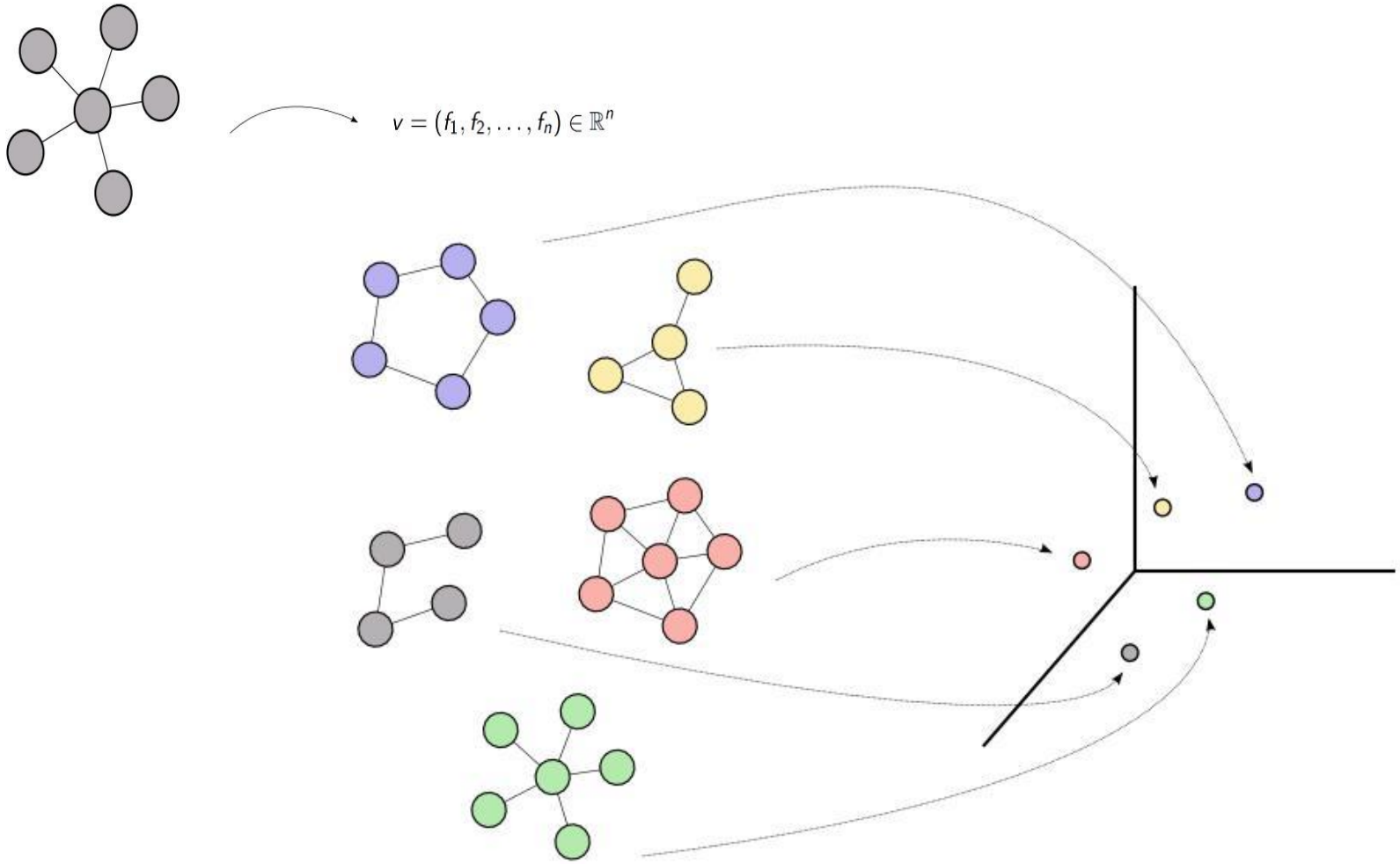
- Appariement de graphe, Exemple:



Appariement avec la fonction f

Souvent trop strict ⇒ isomorphismes de sous-graphes

Classification en terme de graphes



Classification en termes de graphes

- Soient X et Y les objets (resp. scènes), on cherche à savoir si

$$X \cong Y \text{ ou } X \subseteq Y.$$

- L'isomorphisme de graphe $G = (V, E), G' = (V', E')$ $(X \cong Y)$
  - $|V| = |V'|$ et
  - il existe $\phi : V \to V'$ bijective tel que :

$$(v_1, v_2) \in E \Leftrightarrow (\phi(v_1), \phi(v_2)) \in E'$$

- L'isomorphisme partiel de sous graphes $(X \subseteq Y)$
  - $|V| \leq |V'|$ et
  - il existe $\phi : V \to V'$ injective tel que :

$$(v_1, v_2) \in E \Rightarrow (\phi(v_1), \phi(v_2)) \in E'$$

Classification en termes de graphes

- L'isomorphisme de sous graphe
  - idem que isomorphisme partiel de sous graphe avec en plus :

$$\forall (v_1, v_2) \in V^2 \ (v_1, v_2) \notin E \Rightarrow (\phi(v_1), \phi(v_2)) \notin E'$$

on a donc :

$$(\phi(v_1), \phi(v_2)) \in E' \Rightarrow (v_1, v_2) \in E$$

Classification en termes de graphes

- Sous graphe partiel commun de taille maximum (mcps).

  graphe de taille maximum (en nombre de noeuds), sous graphe partiel de G et G'.

- Sous graphe commun de taille maximum (mcs)

  idem que mcps mais isomorphisme de sous graphe au lieu d’isomorphisme partiel de sous graphes.

Classification en termes de graphes

- Sous graphe (partiel) commun **maximum** ou **maximal** ?
- Étant donné un sous graphe (partiel) commun de G et G', celui-ci est dit **maximal**, si on ne peut plus y ajouter de sommet sans briser les isomorphismes.
- Un sous graphe est dit **maximum** si tout sous graphe (partiel) commun de G et G' à un cardinal (nombre de noeuds) inférieur.

Classification en termes de graphes

- L'appariement non bijectif :
  - Trouver $\phi : V' \rightarrow \mathcal{P}(V)$ qui associe à chaque sommet du graphe modèle $v' \in V'$ un ensemble de sommets du graphe scène et tel que :

1. chaque sommet de V est associé à exactement 1 sommet de V'.
2. $v \in \phi(v')$ ssi cet appariement est vraisemblable (en terme de similarité)
3. chaque sous graphe de G induit par $\phi(v'), v' \in V'$ est connexe.
4. pour tout $v' \in V', |\phi(v')| \geq 1$.

Applications aux images sur segmentées.

⚠ fondamental en terme d'applications.

Isomorphisme de graphes ou de sous graphes :

- Isomorphisme de graphes ou de sous graphes : Problème NP complet
- Heuristiques pour obtenir des solutions (éventuellement) sous optimales.
- Deux approches :
- approche symbolique ou algorithmique :
    - Définir un algorithme qui éffectue une recherche dans l'espace des solutions en rejetant à priori certaines solutions.
    - Bon contrôle sur la solution.
- approche numérique :
    - Définir le problème en terme de minimisation/maximisation d'une fonction/énergie.
    - Utilisation de tout l'arsenal des méthodes de minimisation

Principaux Acteurs

- Approches Algorithmiques
  - Horst Bunke (Suisse),
  - Marcello Pellilo(Italie),
  - Mario Vento (Italie).

- Approches numériques
  - Edwin Hancock (UK),
  - Kittler (UK),
  - Sven Dickinson (Canada),

Isomorphisme

- - Matrice d’adjacence et permutation (Ullman Bunke)
- Appariement par représentation de l’espace des états
- Algorithme Hongrois (algorithme de Kühn), minimisation d’affectation.

Approches algorithmiques (Ullman/Bunke)

- Définitions :
  - Un **graphe labélisé** :$G = (V, E, \mu, \nu, L_v, L_e)$

$$\begin{cases} \mu : V \rightarrow L_v & \text{fonction de label des sommets} \\ \nu : E \rightarrow L_e & \text{fonction de label des arêtes} \end{cases}$$

  - Un **sous graphe labélisé** $G_s = (V_s, E_s, \mu_s, \nu_s, L_v, L_e)$ de $G = (V, E, \mu, \nu, L_v, L_e)$.
    - μ_s et ν_s restriction de μ et ν à $V_s \subset V$ et $E_e \subset E$ (avec $E_s = E \cap V_s \times V_s$).

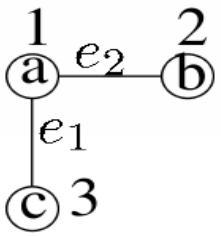
Matrice d'adjacence, permutation

- La **matrice d'adjacence** $M = (m_{i,j})$ d'un graphe $G = (V, E, \mu, \nu, L_v, L_e$ définie par :
  1. $\forall i \in \{1, \ldots, n\}\ m_{i,i} = \mu(v_i)$
  2. $\forall (i, j) \in \{1, \ldots, n\}^2, i \neq j$

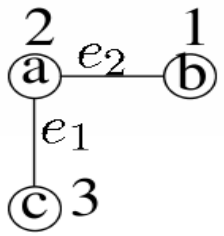
$$m_{i,j} = \begin{cases} \nu((v_i, v_j)) & \text{si } (v_i, v_j) \in E \\ 0 & \text{sinon} \end{cases}$$

- Une **matrice de permutation** $n \times n$, $P = (p_{i,j})$ vérifie :
  1. $\forall (i, j) \in \{1, \ldots, n\}^2\ p_{i,j} \in \{0, 1\}$,
  2. $\forall j \in \{1, \ldots, n\}\ \sum_{i=0}^{n} p_{i,j} = 1$,
  3. $\forall i \in \{1, \ldots, n\}\ \sum_{j=0}^{n} p_{i,j} = 1$.

Matrice d'adjacence, Permutation



ou



$$M = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} a & e_2 & e_1 \\ e_2 & b & 0 \\ e_1 & 0 & c \end{pmatrix} \end{array} , \quad P = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{array}$$

$$M' = PMP^t = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} b & e_2 & 0 \\ e_2 & a & \text{e1} \\ 0 & \text{e1} & c \end{pmatrix} \end{array}$$

Bunke/Ullman : Définitions

- Deux graphes G_1 et G_2 de matrices M_1 et M_2 sont dis **isomorphes** ssi il existe P matrice de permutation telle que :

$$M_2 = PM_1P^t$$

- Il existe un isomorphisme de sous graphe entre G_1 et G_2 ssi il existe $S \subset G_2$ tel que G_1 et S sont isomorphe $(S = (S, E \cap S \times S, \mu_{|S}, \nu_{|S}, L_v, L_e)$.
- Soit $M = (m_{i,j})$ une $n \times n$ matrice d'adjacence.

$$\forall (k,m) \in \{1,\ldots,n\}^2 \; S_{k,m}(M) = (m_{i,j})_{i \in \{1,\ldots,k\}, j \in \{1,\ldots,m\}}$$

  - $S_{k,k}(M)$ matrice d'adjacence du sous graphe restreint aux k premiers sommets.

Bunke/Ullman : isomorphisme de sous-graphe

- Soient G_1 et G_2 de matrices d'adjacences M_1 et M_2
  - $M_1 : m \times m$,
  - $M_2 : n \times n$ avec $m \leq n$.

  Il existe un isomorphisme de sous graphe entre G_1 et G_2 ssi il existe une matrice de permutation $n \times n$ P telle que :

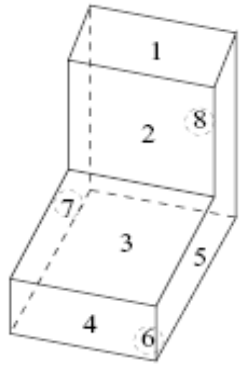
$$M_1 = S_{m,m}(PM_2P^t)$$

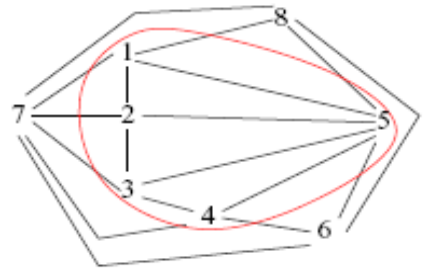
- Remarque 1 :

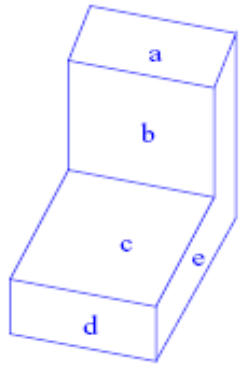
$$M_1 = S_{m,m}(PM_2P^t) = S_{m,n}(P)M_2\left(S_{m,n}(P)\right)^t$$

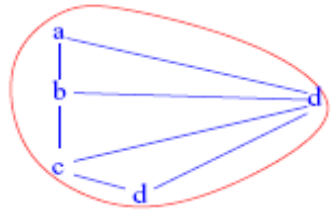
Isomorphismes de sous-graphes

- Il existe un sous-graphe S' de G' tel que f soit un isomorphisme de G dans S'









- Il existe un sous-graphe S de G et un sous-graphe S' de G' tel que f soit un isomorphisme de S dans S'

Bunke/Ullman : isomorphisme de sous-graphe

- Remarque 2 : si pour $i \leq m \leq n$

$$S_{i,i}(M_1) = S_{i,i}(PM_2P^t) = S_{i,n}(P)M_2\,(S_{i,n}(P))^t$$

$S_{i,n}(P)$ représente un appariement partiel entre les i premiers sommets de G_1 et les sommets de G_2.

- L'algorithme

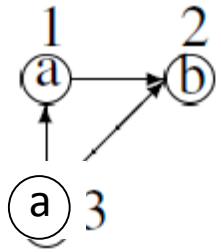
procédure Ullman (G_1,G_2)
  Déclaration
  $P = (p_{i,j})$ matrice $n \times n$ de permutation,
  $m = |V_1|$,$n = |V_2|$,
  M_1, M_2 matrices d'adjacences
début
  retourner backtrack($M_1, M_2, P, 1$)
fin

Bunke/Ullman : isomorphisme de sous-graphe

procédure backtrack (M_1, M_2, P, k)
début
  si $k > m$ **alors**
    Remarque : *P est un isomorphisme de sous graphe.*
    retourner P
  finsi
  pour i $\leftarrow$1 **à** n **faire**
    $p_{k,i} \leftarrow 1$
    $\forall j \neq i \; p_{k,j} \leftarrow 0$
  finpour
  si $S_{k,k}(M_1) = S_{k,n}(P) M_2 S_{k,n}(P)^t$ **alors**
    backtrack($M_1, M_2, P, k+1$)
  finsi
fin

Bunke: Exemple

- Étant donné une matrice M_i de G_i calculer l'ensemble $A(G_i)$ des graphes isomorphes à G_i :



1	2	3
b	1	1
0	a	1
0	0	a

A

1	3	2
b	1	1
0	a	0
0	1	a

B

2	1	3
a	0	1
1	b	1
0	1	a

C

2	3	1
a	1	0
0	a	0
1	1	a

D

3	1	2
a	0	0
1	b	1
1	0	a

E

3	2	1
a	0	0
1	a	0
1	1	b

F

Bunke : Exemple

- Décomposer une matrice en vecteurs :

1	2	3
b	1	1
0	a	1
0	0	a

=

a_1

b

a_2

	1
0	a

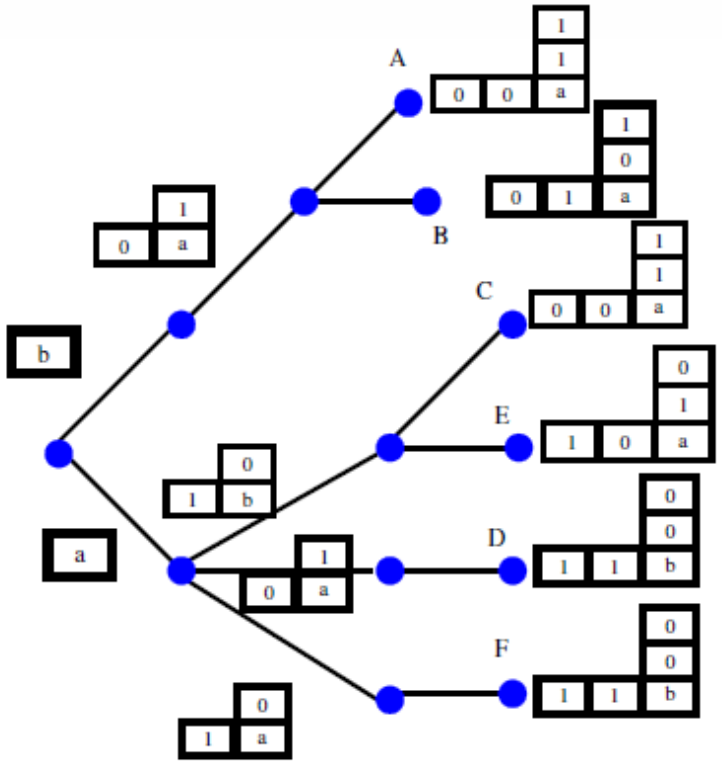
a_3

		1
		1
0	0	a

$$\begin{cases} a_1 = (b) \\ a_2 = (1, a, 0) \\ a_3 = (1, 1, a, 0, 0) \end{cases}$$

Bunke: Exemple

- Construire un arbre de recherche à partir de la décomposition



1	2	3
b	1	1
0	a	1
0	0	a

A

1	3	2
b	1	1
0	a	0
0	1	a

B

2	1	3
a	0	1
1	b	1
0	1	a

C

2	3	1
a	1	0
0	a	0
1	1	a

D

3	1	2
a	0	0
1	b	1
1	0	a

E

3	2	1
a	0	0
1	a	0
1	1	b

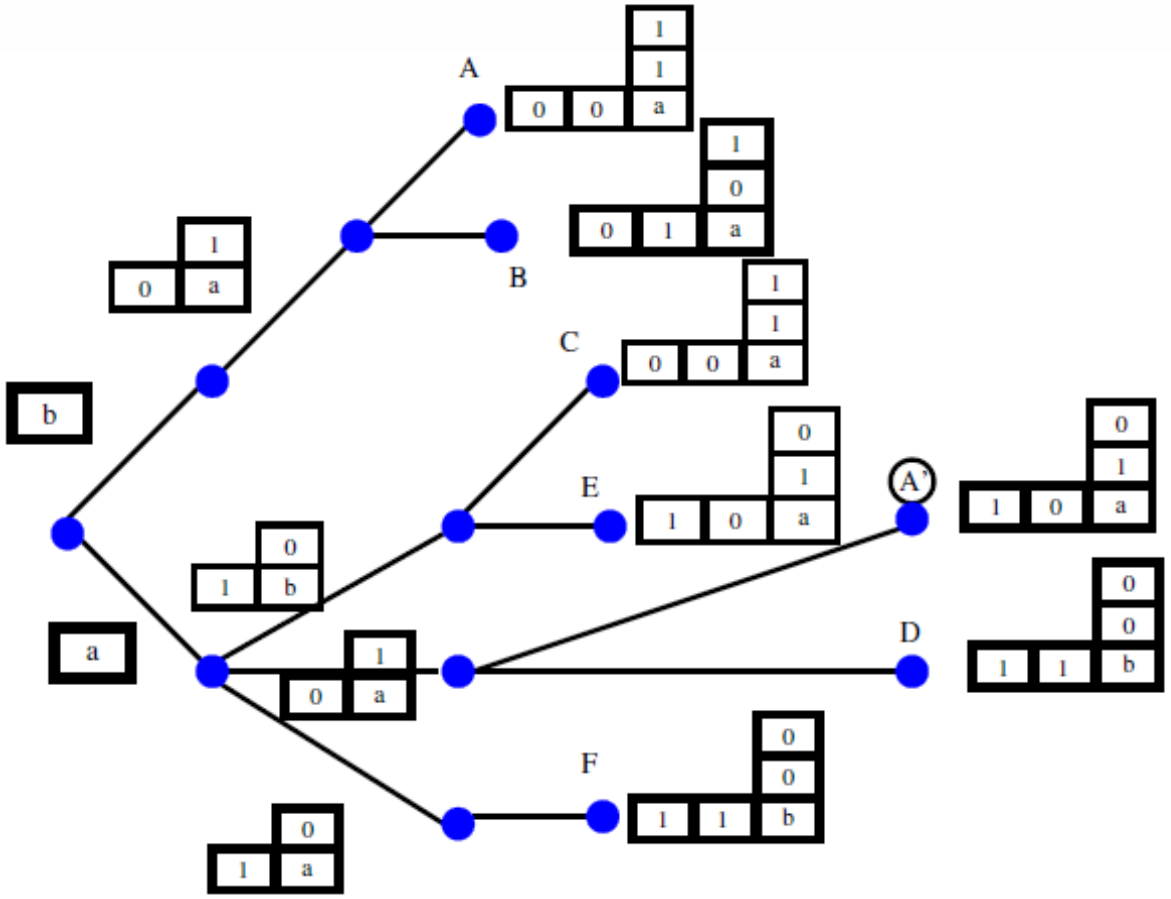
F

Bunke: Exemple

- Créer un arbre contenant tous les modèles.

1	2	3
a	1	0
0	a	1
1	0	a

A’



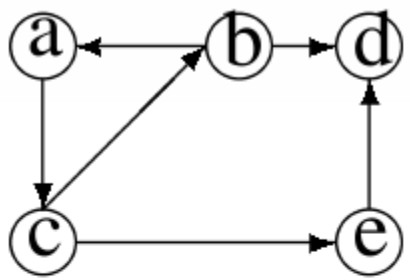
Appariement par représentation de l'espace des états

- State Space Representation (SSR).
- Un état : Un appariement partiel.
- Méthode : explorer successivement les différents états.
- Différentiation des méthodes :
  - Passage d'un état à un autre,
  - Méthode pour ne pas boucler (considérer 2 fois le même état),
  - Heuristique pour restreindre l'espace de recherche.

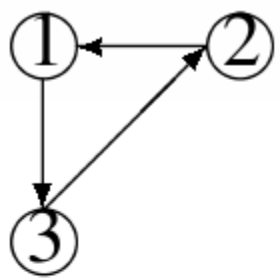
SSR : Algorithme de Cordella 2001-VF2

- Notations :
  - $G_1 = (V_2, E_2)$, $G_1 = (V_2, E_2)$ les deux graphes *orientés*,
  - On cherche soit :
    - un isomorphisme entre G_1 et G_2,
    - Soit un isomorphisme de sous graphe entre G_2 et G_1 ($|V_2| \leq |V_1|$)
  - état s,
  - $M(s)$ appariement partiel associé à s,
  - $M_1(s)$ sommets de $M(s)$ dans V_1,
  - $M_2(s)$ sommets de $M(s)$ dans V_2
  - $P(s)$ ensemble des couples (dans $V_1 \times V_2$) candidats à un ajout dans s,
  - $F(s, n, m)$ l'ajout de (n, m) à s définit il un isomorphisme partiel ?
  - $T_1^{in}(s)(T_1^{out}(s))$ ensemble des sommets de G_1 prédécesseurs (successeurs) d'un sommet de $M_1(s)$.
  - $T_2^{in}(s)(T_2^{out}(s))$ ensemble des sommets de G_2 prédécesseurs (successeurs) d'un sommet de $M_2(s)$.

Cordella : exemple de notations



G_1



G_2

- $M(s) = (a, 1)$
- $M_1(s) = \{a\}, M_2(s) = \{1\}$
- $T_1^{in}(s) = \{b\}; T_1^{out}(s) = \{c\};$
- $T_2^{in}(s) = \{2\}; T_2^{out}(s) = \{3\};$

Cordella : l'algorithme

procédure appariement (E s)
début
Remarque : s_0 *entrée initiale telle que* $M(s_0) = \emptyset$
si $M(s)$ contient tous les sommets de G_2 **alors**
retourner $M(s)$
sinon
Calculer $P(s)$
Pour chaque (n, m) **dans** $P(s)$ **faire**
si $F(s, n, m)$ **alors**
Calculer s' après ajout de (n, m) à $M(s)$
appariement(s')
finsi
finpour
Restauration des structures de données
finsi
fin

Calcul de *P(s)*

- Si $T_1^{out}(s)$ et $T_2^{out}(s)$ non vides

$$P(s) = T_1^{out}(s) \times \{minT_2^{out}(s)\}$$

  min relation d'ordre quelconque.
- Sinon Si $T_1^{in}(s)$ et $T_2^{in}(s)$ non vides

$$P(s) = T_1^{in}(s) \times \{minT_2^{in}(s)\}$$

- Sinon si $T_1^{in}(s) = T_2^{in}(s) = T_1^{out}(s) = T_2^{out}(s) = \emptyset$

$$P(s) = (V_1 - M_1(s)) \times \{min(V_2 - M_2(s))\}$$

- Note : Si un des $T^{in}(s)$ (resp. $T^{out}(s)$) est vide et pas l'autre $M(s)$ ne peut conduire à un appariement.

Calcul de *F(s,n,m)*

$$F(s,n,m) = R_{pred}(s,n,m) \wedge R_{succ}(s,n,m) \wedge R_{in}(s,n,m) \wedge R_{out}(s,n,m) \wedge R_{new}(s,n,m)$$

- R_{pred}, R_{succ} : M(s') définit il un appariement ?
- R_{in}, R_{out} : pourra t on construire un appariement au coup d'après ?
- R_{new} : Pourrai je arriver à un apariement (à terme) ?
  - $R_{pred}(s,m,n)$: les prédécesseurs correspondent :

$$(\forall n' \in M_1(s) \cap Pred(G_1,n) \exists m' \in Pred(G_2,m) | (n',m') \in M(s)) \wedge$$
$$(\forall m' \in M_2(s) \cap Pred(G_2,m) \exists n' \in Pred(G_1,n) | (n',m') \in M(s))$$

  - $R_{succ}(s,m,n)$: les successeurs correspondent :

$$(\forall n' \in M_1(s) \cap Succ(G_1,n) \exists m' \in Succ(G_2,m) | (n',m') \in M(s)) \wedge$$
$$(\forall m' \in M_2(s) \cap Succ(G_2,m) \exists n' \in Succ(G_1,n) | (n',m') \in M(s))$$

Prédicats *Rin* et *Rout*

- Les successeurs (prédécesseurs) de n et m doivent être en correspondance localement.
  - $R_{in}(s, n, m)$

$$\left(\left|T_1^{in}(s) \cap Succ(G_1, n)\right| \geq \left|T_2^{in}(s) \cap Succ(G_2, m)\right|\right) \wedge$$
$$\left(\left|T_1^{in}(s) \cap Pred(G_1, n)\right| \geq \left|T_2^{in}(s) \cap Pred(G_2, m)\right|\right)$$

  - $R_{out}(s, n, m)$

$$\left(\left|T_1^{out}(s) \cap Succ(G_1, n)\right| \geq \left|T_2^{out}(s) \cap Succ(G_2, m)\right|\right) \wedge$$
$$\left(\left|T_1^{out}(s) \cap Pred(G_1, n)\right| \geq \left|T_2^{out}(s) \cap Pred(G_2, m)\right|\right)$$

- Remplacer $\geq$ par $=$ pour l'isomorphisme de graphe.

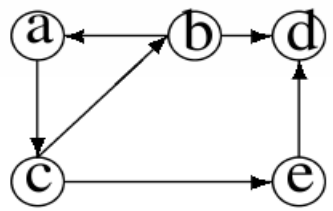
Prédicat *Rnew*

- Les successeurs et prédécesseurs doivent se correspondre en dehors des $M_i(s), T_i^{in}(s)$ et $T_i^{out}(s), i = 1, 2$.
  - $R_{new}(s, n, m)$

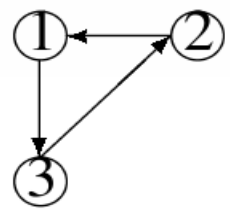
$$(|N_1(s) \cap Succ(G_1, n)| \geq |N_2(s) \cap Succ(G_2, m)|) \wedge$$
$$(|N_1(s) \cap Pred(G_1, n)| \geq |N_2(s) \cap Pred(G_2, m)|)$$

  - avec :
    - $N_1(s) = V_1 - M_1(s) - T_1^{in}(s) \cup T_1^{out}(s)$: tout ce qui reste a voir dans V_1
    - $N_2(s) = V_2 - M_2(s) - T_2^{in}(s) \cup T_2^{out}(s)$: tout ce qui reste a voir dans V_2

Exemple



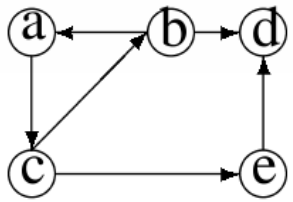
G_1



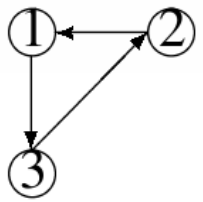
G_2

- $M(s) = (a, 1)$
- $M_1(s) = \{a\}, M_2(s) = \{1\}$
- $T_1^{in}(s) = \{b\}; T_1^{out}(s) = \{c\};$
- $T_2^{in}(s) = \{2\}; T_2^{out}(s) = \{3\};$
- $P(s) = (c, 3)$
- $Pred(G_1, c) = \{a\}; Succ(G_1, c) = \{b, e\}$
- $Pred(G_2, 3) = \{1\}; Succ(G_2, 3) = \{2\}$
- $F(s, c, 3) = vrai.$

Exemple



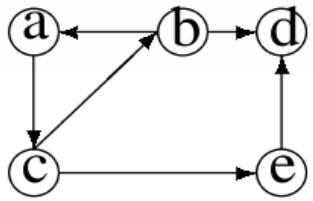
G_1



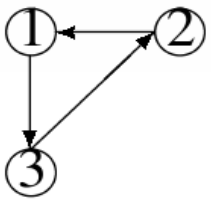
G_2

- $M(s') = \{(a,1),(c,3)\}$
- $M_1(s') = \{a,c\}, M_2(s) = \{1,3\}$
- $T_1^{in}(s') = \{b\}; T_1^{out}(s') = \{b,e\};$
- $T_2^{in}(s') = \{2\}; T_2^{out}(s') = \{2\};$
- $P(s') = (b,2),(e,2)$
- $Pred(G_1,e) = \{c\}; Succ(G_1,e) = \{d\}$
- $Pred(G_2,2) = \{3\}; Succ(G_2,2) = \{1\}$
- $(e,2)$ viole le prédicat $R_{succ}(s',e,2)$. De fait :
  - $1 \in M_2(s') \cap Succ(G_2,2)$ or $Succ(G_1,e) \cap M_1(s') = \emptyset$

Exemple



G_1



G_2

- On arrive donc à l'appriement : $M(s'') = \{(a, 1), (c, 3), (b, 2)\}$ et l'on a terminé puisque l'on couvre tous les sommets de G_2.

Conclusion

- Complexité :($N = |V_1| + |V_2|$)

	VF2		Ullman	
Complexité	au mieux	au pire	au mieux	au pire
Temps	$\mathcal{O}(N^2)$	$\mathcal{O}(N!N)$	$\mathcal{O}(N^3)$	$\mathcal{O}(N!N^2)$
Espace	$\mathcal{O}(N)$	$\mathcal{O}(N)$	$\mathcal{O}(N^3)$	$\mathcal{O}(N^3)$

- Utilisable pour de gros graphes (jusqu'a 1000 sommets).

Algorithme Hongrois

- Permet de résoudre des problèmes d'affectation.
- On représente dans une matrice le coût de l'affectation entre deux éléments (ce coût peut être dans notre cas un critère de ressemblance)
- On calcule alors le coût minimal

Exemple

Réduction du tableau initial: on soustrait à chaque ligne du tableau initial le plus petit élément de la ligne. On fait de même avec les colonnes.

17	**15**	**9**	**5**	**12**
16	16	10	5	10
12	15	14	11	5
4	8	14	17	13
13	9	8	12	17

12	**10**	**4**	**0**	**7**
11	11	5	0	5
7	10	9	6	0
0	4	10	13	9
5	1	0	4	9

12	**9**	**4**	**0**	**7**
11	10	5	0	5
7	9	9	6	0
0	3	10	13	9
5	0	0	4	9

Exemple

- Encadrer et barrer des zéros
- On cherche la ligne comportant le moins de zéros non barrés (en cas d'égalité, choisir la ligne la plus haute). On encadre un des zéros de cette ligne (arbitrairement le plus à gauche)
- On barre tous les zéros se trouvant sur la même ligne ou sur la même colonne que le zéro encadré.

12	**9**	**4**	**0**	**7**
11	10	5	~~0~~	5
7	9	9	6	0
0	3	10	13	9
5	0	0	4	9

12	**9**	**4**	**0**	**7**
11	10	5	~~0~~	5
7	9	9	6	0
0	3	10	13	9
5	0	0	4	9

Exemple

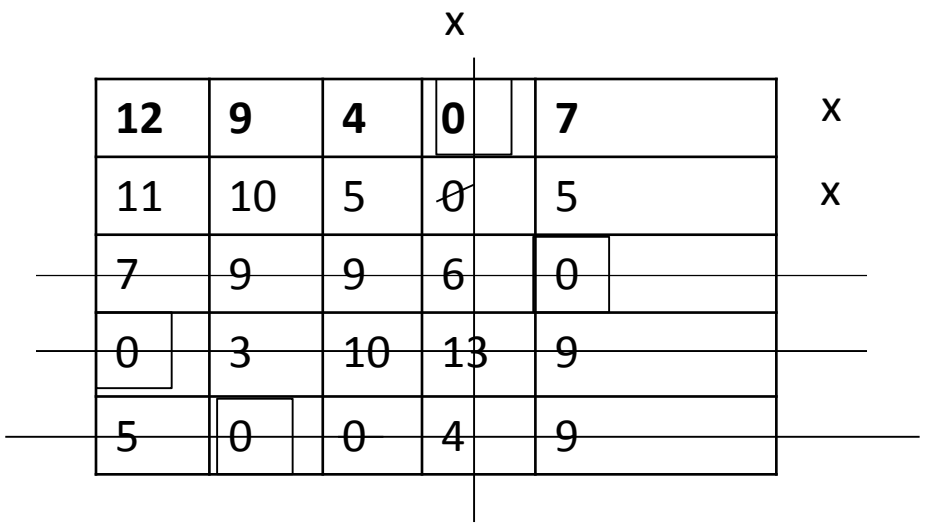
12	**9**	**4**	**0**	**7**
11	10	5	~~0~~	5
7	9	9	6	0
0	3	10	13	9
5	0	~~0~~	4	9

x

12	**9**	**4**	**0**	**7**	x
11	10	5	~~0~~	5	x
7	9	9	6	0	
0	3	10	13	9	
5	0	~~0~~	4	9	

1- Marquer d'une croix toutes les lignes ne contenant aucun zéro encadré
2- Marquer toute colonne ayant un zéro barré sur une ligne marquée
3- Marquer toute ligne ayant un zéro encadré dans une colonne marquée
On répète 2 et 3 jusqu'à ne plus pouvoir marquer de ligne ou de colonne

Exemple



12	**9**	**4**	**0**	**7**
11	10	5	~~0~~	5
~~7~~	~~9~~	~~9~~	~~6~~	0
~~0~~	~~3~~	~~10~~	~~13~~	~~9~~
~~5~~	0	~~0~~	~~4~~	~~9~~

12	**9**	**4**	**0**	**7**
11	10	5	~~0~~	5
7	9	9	6	0
0	3	10	13	9
5	0	~~0~~	4	9

- On trace un trait sur toute ligne non marquée et sur toute colonne marquée
- On retranche à toutes les cases du tableau partiel le plus petit élément de celui-ci
- On ajoute ce même élément à toutes les case du tableau initial barrées deux fois

Exemple

12	**9**	**4**	**0**	**7**
11	10	5	~~0~~	5
7	9	9	6	0
0	3	10	13	9
5	0	~~0~~	4	9

8	**5**	**0**	**0**	**3**
7	6	1	0	1
7	9	9	10	0
0	3	10	17	9
5	0	~~0~~	8	9

On obtient un nouveau tableau sur lequel on peut répéter les trois étapes
En revenant sur le tableau initial on obtient la valeur de l'affectation minimale
9+5+5+4+9=32

Algorithme hongrois

- Ce principe peut être utilisé pour effectuer l'appariement des nœuds d'un graphe.

- Appariement d'ensembles avec édition, Application à la distance d'édition bipartie entre graphes,
- Sébastien Bougleux, Luc Brun, Benoit Gaüzère, RFIA 2016, Clermont ferrand.

Problèmes de sous graphes communs

- Liens entre les isomorphismes de sous graphes et les sous graphes communs
  - Étant donné deux graphes G_1 et G_2, G est un sous graphe commun de G_1 et G_2 ssi il existe :
    - $G \xrightarrow{\varphi} G_1$
    - $G \xrightarrow{\psi} G_2$

    φ, ψ isomorphismes de sous graphes.
- G est maximum (maximal) si on ne peut trouver de sous graphe de cardinal supérieur (incluant G).
- Première idée : utiliser un algorithme SSR (VF2,McGregor).

Utilisation d'algorithmes SSR

```
procédure appariement (E s)
début
    Remarque : s_0 entrée initiale telle que M(s_0) = MCS = ∅
    si M(s) contient tous les sommets de G_2 alors
        retourner M(s)
    sinon
        Calculer P(s)
        Pour chaque (n, m) dans P(s) faire
            si F(s, n, m) alors
                Calculer s' après ajout de (n, m) à M(s)
                si taille(M(s'))> tailleMax alors
                    tailleMax ← taille(M(s'))
                    MCS ← M(s')
                finsi
                appariement(s')
            finsi
```

Graphe d'association

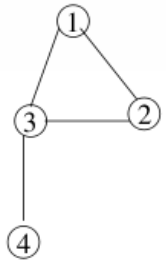
- Soient $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$, le graphe d'association $G = (V, E)$ de G_1 et G_2 est défini par :
  - Sommets :

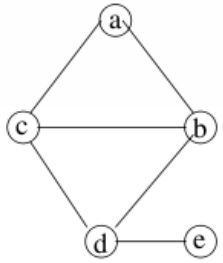
$$V = V_1 \times V_2$$

  - Arêtes :

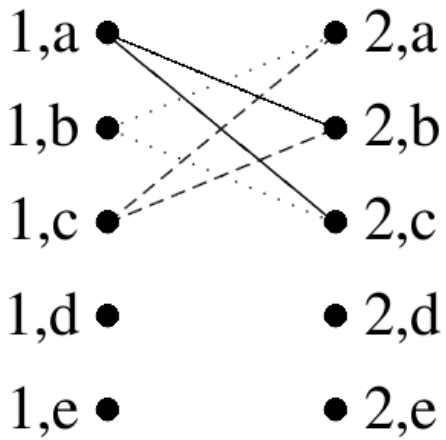
$$E = \{((i,h),(j,k)) \in V \times V | i \neq j, h \neq k \text{ et } (i,j) \in E_1 \Leftrightarrow (h,k) \in E_2 \}$$

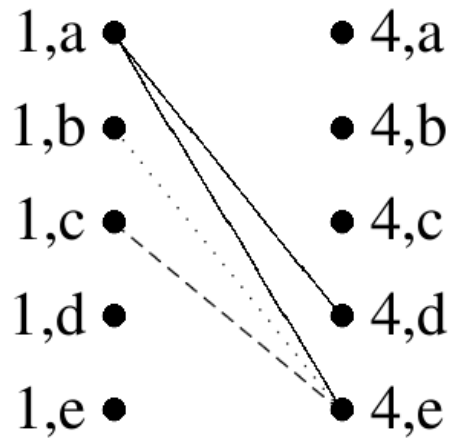
Graphe d'association : exemple



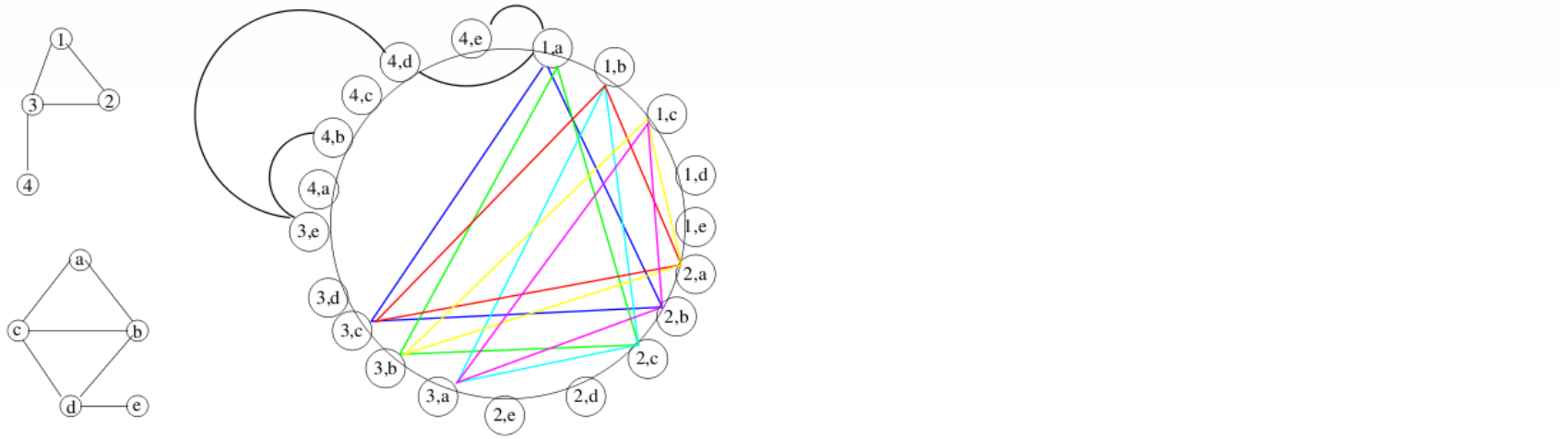


mais également





Graphe d'association : exemple



- Le sous graphe de G restreint à $\{(1,a),(2,b),(3,c)\}$ est complet (lignes bleu)
  - Tous les appariements sont compatibles.

MCS et clique

- Un sous graphe complet d'un graphe G est appellé un **clique** de G.
- On parle de clique maximal (resp. maximum).
- Le clique-number de G, w(G), est la taille (en nombre de sommets) du clique maximum.
- Théorème :

Soient G1 et G2 deux graphes et G leur graphe d'association. Il existe une relation bijective entre

- les cliques maximal (maximum) de G et

- les sous graphes communs maximal (maximum) de G1 et G2.

- Donc : Calculer les cliques du graphe d'association G est **équivalent** à calculer les sous graphes communs de G1 et G2.

DurandParisi (1999)

```
procédure durandParisi (E s)
début
    tant que nextNode(s,n) faire
        si isLegalNode(s,n) et non pruningCondition(s) alors
            s' ← addNode(s,n)
            si taille(s')> tailleMax alors
                MCS ← M(s')
                tailleMax ← taille
            finsi
            si non leafOfSearchTree(s') alors
                durandParisi(s')
            finsi
            backtrack(s')
        finsi
    fintantque
fin
```

DurandParisi : exemple