

Resolució de problemes de reconeixement de patrons

2. Resolució de problemes de reconeixement de patrons.

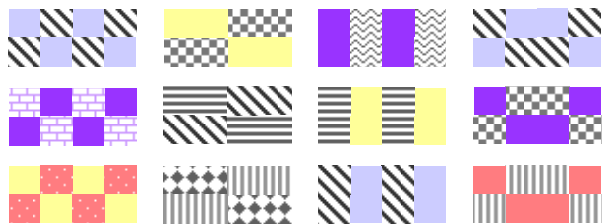
- Introducció.
- Reconeixement estadístic.
- Reconeixement estructural.

Bibliografia adicional:

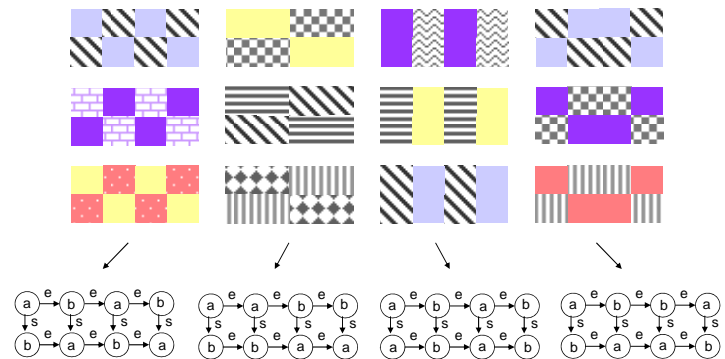
"Introduction to Pattern Recognition"
M. Friedman and A. Kandel
World Scientific, 1999

"Pattern Classification", 2nd Edition
R.O. Duda, P.E. Hart, D.G. Stork
Wiley, 2001

Exemple: Com es poden classificar els següents objectes en 4 classes ?



Exemple: Com es poden classificar els següents objectes en 4 classes ?



Xarxa semàntica o graf: es una representació de coneixement que definim com:

Lèxic: node, arc, etiqueta d'arc, correspondència, matriu d'adjacència.

Estructural:

- Qualsevol arc sempre va unit a dos nodes, node origen i node destí.
- Les etiquetes sempre van associades als arcs.
- La matriu d'adjacència d'una xarxa és $N \times N$, on N : núm. de nodes.

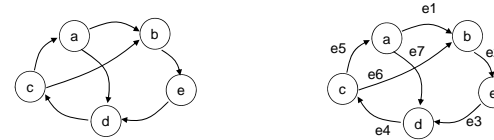
Semàntic:

- Depèn de l'aplicació, però en general:
 Node=Objecte Node=Estat d'un problema
 Arc= Relacions entre objectes Arc=Canvi d'estat

Procedimental:

- Proc. que construeixen un graf.
- Proc. que comparen o fan correspondència entre grafos.
- Proc. que retornen tots els nodes destí d'un node donat.

Representació amb matriu d'adjacència



1. Llista de nodes amb la seva llista de nodes destí.

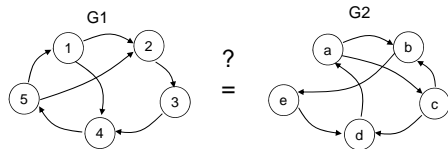
((a (b d)) (b (e)) (c (a b)) (d (c)) (e (d)))
 ((a ((e1 b) (e7 d))) (b ((e2 e))) (c ((e5 a) (e6 b)) (d ((e4 c))) (e ((e3 d))))

2. Matriu d'adjacència

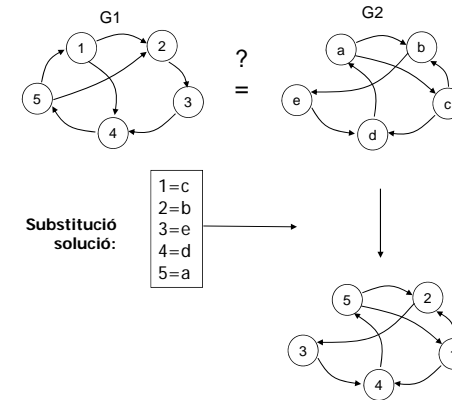
	a	b	c	d	e
a	0	1	0	1	0
b	0	0	0	0	1
c	1	1	0	0	0
d	0	0	1	0	0
e	0	0	0	1	0

	a	b	c	d	e
a	0	e1	0	e7	0
b	0	0	0	0	e2
c	e5	e7	0	0	0
d	0	0	e4	0	0
e	0	0	0	e3	0

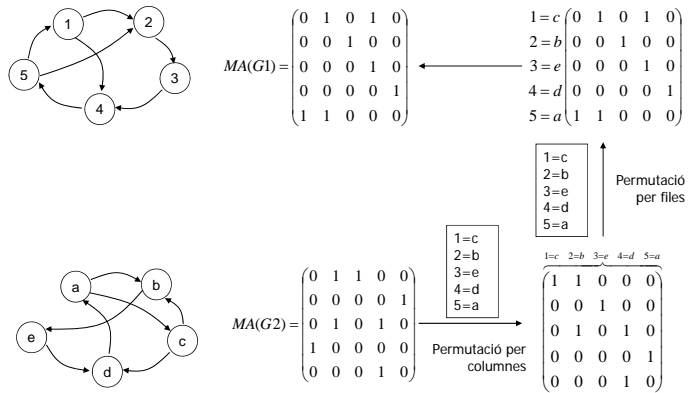
Correspondència entre dos grafos G1 i G2: Trobar una substitució dels nodes de G1 amb els de G2 que faci que els dos grafos siguin iguals.



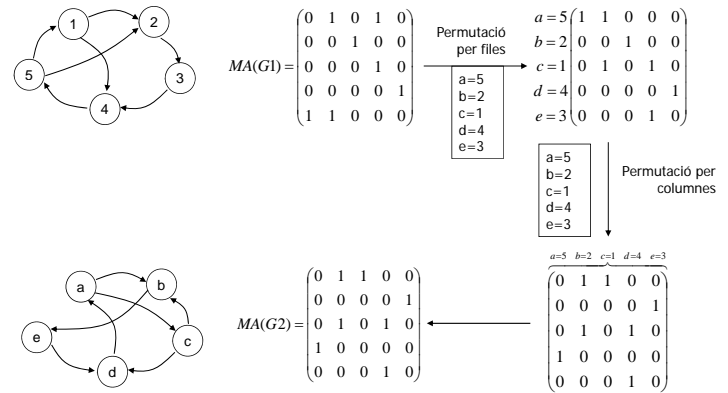
Correspondència entre dos grafos G1 i G2: Trobar una substitució dels nodes de G1 amb els de G2 que faci que els dos grafos siguin iguals.



Correspondència entre dos grafs G1 i G2: Trobar una substitució dels nodes de G1 amb els de G2 que faci que els dos grafs siguin iguals.



Correspondència entre dos grafs G1 i G2: Trobar una substitució dels nodes de G1 amb els de G2 que faci que els dos grafs siguin iguals.



Algorisme de correspondència de grafs

Funció Graph-matching(G1,G2)

1. L=llista de totes les permutacions dels nodes de G1.
2. Per a cada (element, k, de L) fer
 - A=Aplicar_Permutacio(k,MA(G1));
 - Si (A=MA(G2)) llavors Retornar(k és la solució)
 - Fsi
3. Fper
4. Retornar(No existeix solució exacta)

Ffuncio

MA(G): és la matriu d'adjacència del graf G.

Aplicar_Permutacio(k,M): retorna la matriu M després d'aplicar-li la permutació indicada a k.

Implementació RECURSIVA

IDEA: Les permutacions es poden construir a partir d'insertar cada un dels elements a totes les llistes que representen les permutacions de la resta d'elements.

Exemple:

Elements	Permutacions de la resta
a	permutacions((b c)) = ((b c) (c b)) ((a b c) (a c b))
b	permutacions((a c)) = ((a c) (c a)) ((b a c) (b c a))
c	permutacions((a b)) = ((a b) (b a)) ((c a b) (c b a))

```

Funció permute (Lin)
  Si (Lin=="buit") llavors
    Retornar(llista_buida)
  sinó
    Per a cada (element E de Lin) fer
      L=permute(Lin-{E})
      Per a cada (element Li de L) fer
        insertar(E,L);
      FPer
    Lout=Lout+L;
  Fsi
FFunció
    
```

Complexitat de l'algorisme:

$2 \cdot n! \rightarrow O(n!)$ n : núm de nodes d'un graf

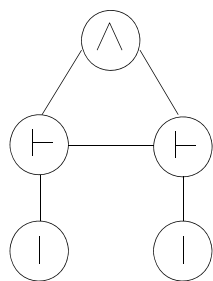
- Construcció de la llista de permutacions
- Exploració de la llista

Implementació basada en una CERCA en un arbre ?

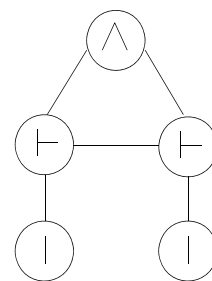
Exemple: Implementació d'un OCR basat en les propietats estructurals de les lletres



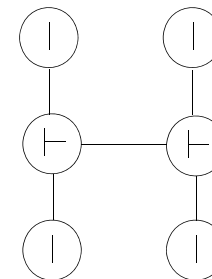
Exemple: Implementació d'un OCR basat en les propietats estructurals de les lletres



Exemple: Implementació d'un OCR basat en les propietats estructurals de les lletres



Problemes:



Inexistència de solució: Quan no existeixi una solució que tingui correspondència perfecta, aleshores es pot buscar una correspondència parcial amb una mesura de similitud, per exemple:

$$\text{Similitud}(G1, G2) = \alpha \times \#(S_{G1} \cap S_{G2}) + \beta \times \#(S_{G1} - S_{G2}) + \gamma \times \#(S_{G2} - S_{G1}) \quad \alpha, \beta, \gamma \in \mathbb{R}$$

Possible millora:

Algorisme AC4 (basat en Relaxació Discreta)

Consisteix en fer un pas previ de relaxació d'una sèrie de restriccions sobre les etiquetes dels enllaços o dels nodes, això permet dividir els n nodes d'un graf en varis subconjunts:

$$n = n_1 + n_2 + \dots + n_p$$

on n_1 : nodes de $G1$ i $G2$ que comparteixen propietat 1

n_2 : nodes de $G1$ i $G2$ que comparteixen propietat 2

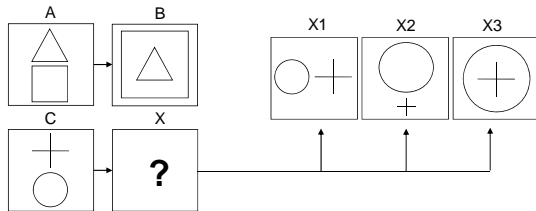
⋮

n_p : nodes de $G1$ i $G2$ que comparteixen propietat p

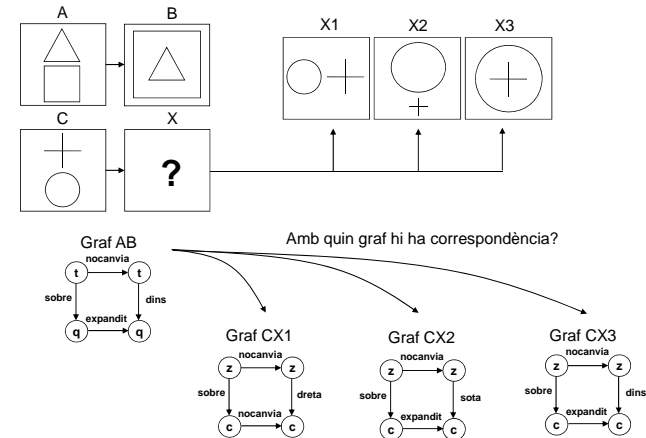
Per tant, la complexitat de la correspondència dels subgrafs serà:

$$O(n_1! n_2! \dots n_p!) \ll O(n!)$$

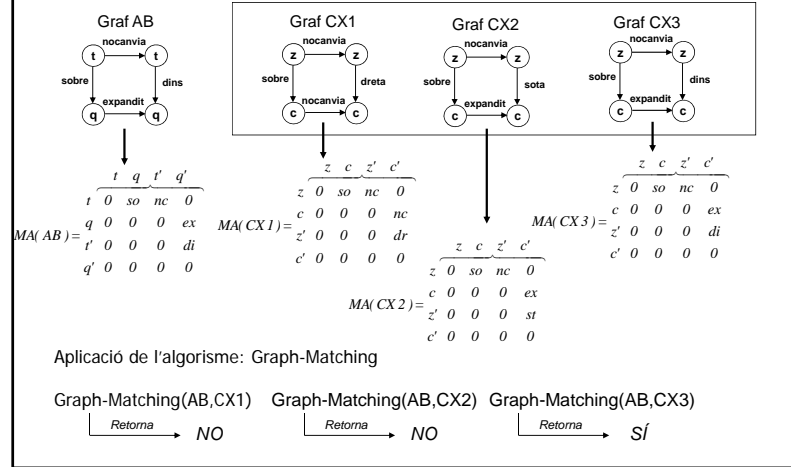
Resolució del problema de l'analogia geomètrica



Resolució del problema de l'analogia geomètrica

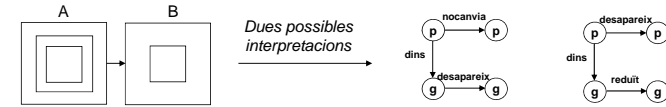


Representació amb Matrius d'adjacència:



Algunes consideracions:

- Existència d'ambigüitats:



Per tractar les ambigüitats hi ha diverses possibilitats:

1. Considerar totes les possibles interpretacions de tots els grafs, fet que augmenta la complexitat.
2. Considerar només la interpretació més habitual (això implica obtenir una taula de valoracions habituals) i només considerar les altres si no hi ha solució.