Exact (Graph) Matching

Presentation for
TAMOP-4.2.2/B-10/1-2010-0012

Dr. Horst Bunke

Outline

- **Graph matching**
  - Definition
  - Manifold application areas
- **Exact matching**
  - Definition
  - Different forms of exact matching
  - Necessary concepts
- **Algorithms for exact matching**
  - Techniques based on tree search
  - Other techniques
- **Conclusion**

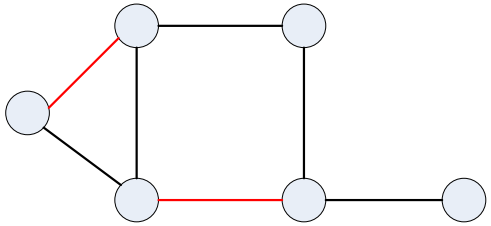
Graph matching

- **Definition:**
  - Graph matching is the process of finding a correspondence between the nodes and the edges of two graphs that satisfies some (more or less stringent) constraints ensuring that similar substructures in one graph are mapped to similar substructures in the other.

    [D. Conte, Pasquale Foggia, Carlo Sansone, and Mario Vento. Thirty Years of Graph Matching in Pattern Recognition.]

- **in comparison with matching (graph theory):**
  - A matching or independent edge set in a graph is a set of edges without common vertices. It may also be an entire graph consisting of edges without common vertices.

    [Wikipedia]



Graph matching

- **Manifold application areas**
  - 2D and 3D image analysis
  - Document processing
  - Biometric identification
  - Image databases
  - Video analysis
  - Biological and biomedical applications

- **Categories of matching**
  - Exact matching
  - Inexact matching
  - Other matching problems

Exact matching

- **Definition**
  - Exact graph matching is characterized by the fact that the mapping between the nodes of the two graphs must be *edge-preserving* in the sense that if two nodes in the first graph are linked by an edge, they are mapped to two nodes in the second graph that are linked by an edge as well.

- **Different forms of exact matching**
  - The most stringent form: graph isomorphism
  - A weaker form: subgraph isomorphism
  - A slightly weaker form: monomorphism
  - A still weaker form: homomorphism
  - Another interesting form: maximum common subgraph (MCS)

Necessary concepts

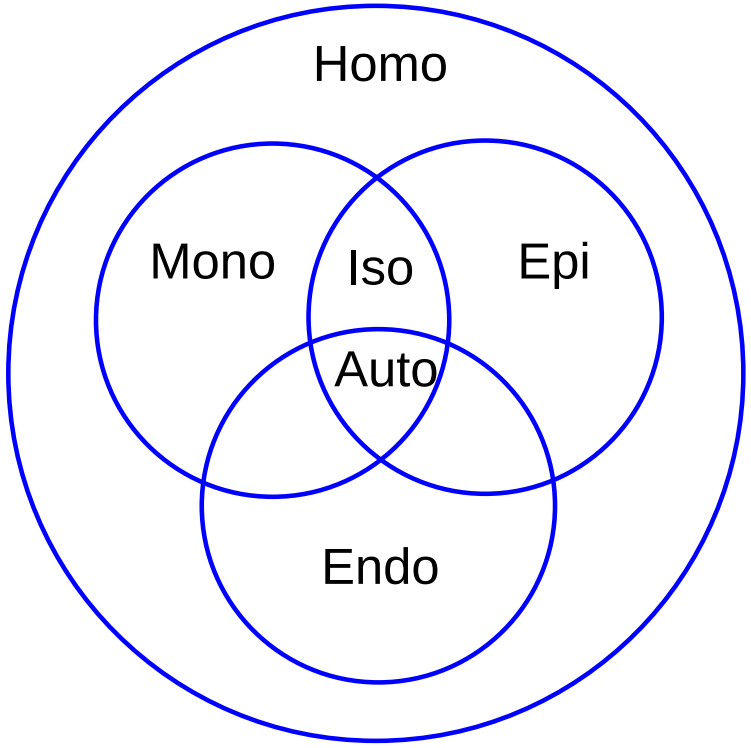
- **Morphism**
  - A morphism is an abstraction derived from structure-preserving mappings between two mathematical structures.
  - In comparison with homomorphism: a homomorphism is a structure-preserving map between two mathematical structures.
  - If a morphism f has domain X and codomain Y, we write $f : X \rightarrow Y$. Thus a morphism is represented by an arrow from its domain to its codomain. The collection of all morphisms from X to Y is denoted *Hom(X,Y)* or *Mor(X,Y)* .

- **Isomorphism**
  - An isomorphism is a *bijective* homomorphism.
  - An isomorphism is a morphism $f : X \rightarrow Y$ in a category for which there exists an "inverse" $f^{-1} : Y \rightarrow X$, with the property that both $f^{-1} \cdot f = id_X$ and $f \cdot f^{-1} = id_Y$.

Necessary concepts

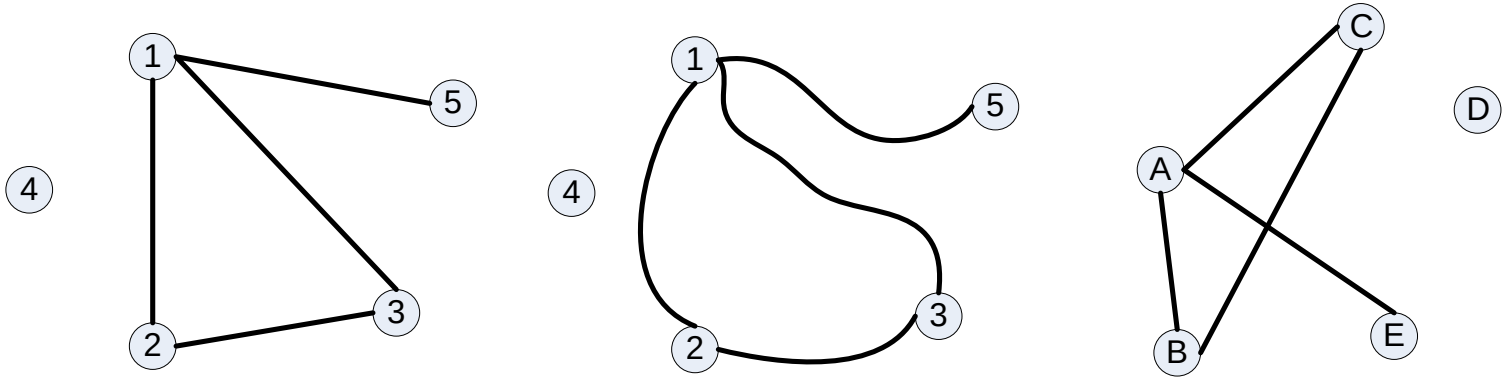
- **Epimorphism**
  - an *surjective* homomorphism
- **Monomorphism**
  - an *injective* homomorphism
- **Endomoprphism**
  - a homomorphism from an object to itself
- **Automoprphism**
  - an endomorphism which is also an isomorphism

    an isomorphism with itself



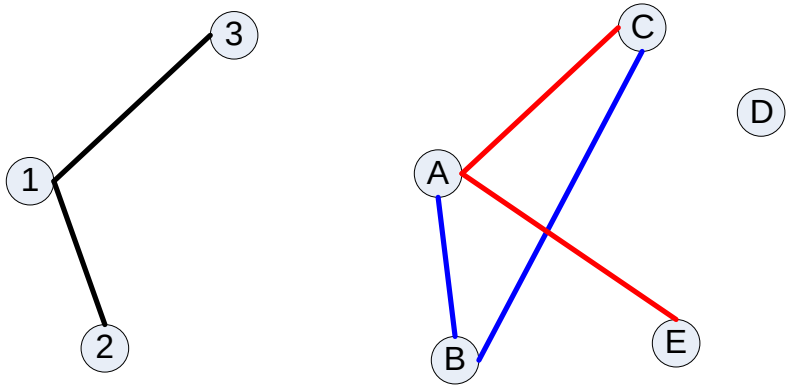
Different forms of exact matching

- **Graph isomorphism**
  - A one-to-one correspondence must be found between each node of the first graph and each node of the second graph.
  - Graphs $G(V_G,E_G)$ and $H(V_H,E_H)$ are isomorphic if there is an invertible F from V_G to V_H such that for all nodes u and v in V_G, $(u,v) \in E_G$ if and only if $(F(u),F(v)) \in E_H$.



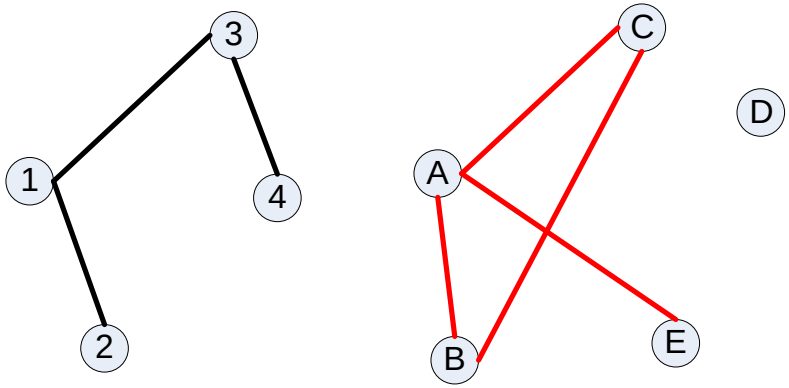
Different forms of exact matching

- **Subgraph isomorphism**
  - It requires that an isomorphism holds between one of the two graphs and a node-induced subgraph of the other.
- **Monomorphism**
  - It requires that each node of the first graph is mapped to a distinct node of the second one, and each edge of the first graph has a corresponding edge in the second one; the second graph, however, may have both extra nodes and extra edges.



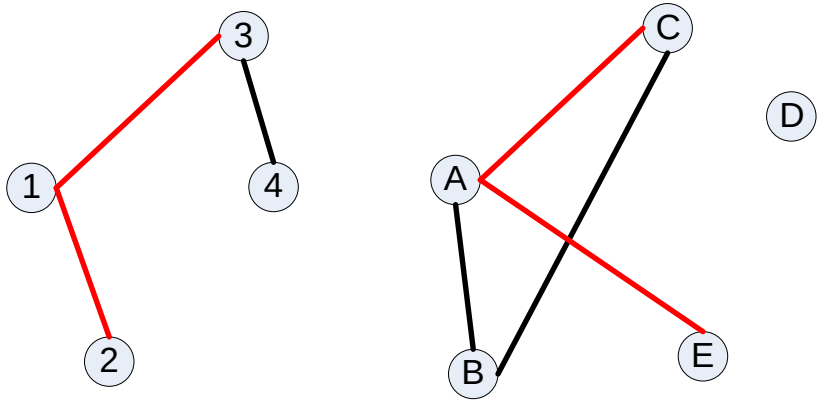
Different forms of exact matching

- **Homomorphism**
  - It drops the condition that nodes in the first graph are to be mapped to distinct nodes of the other; hence, the correspondence can be many-to-one.
  - A graph homomorphism F from Graph $G(V_G,E_G)$ and $H(V_H,E_H)$, is a mapping F from V_G to V_H such that $\{x,y\} \in E_G$ implies $\{F(x),F(y)\} \in E_H$.



Different forms of exact matching

- **Maximum common subgraph (MCS)**
  - A subgraph of the first graph is mapped to an isomorphic subgraph of the second one.
  - There are two possible definitions of the problem, depending on whether *node-induced subgraphs or plain subgraphs* are used.
  - The problem of finding the MCS of two graphs can be reduced to the problem of finding the *maximum clique* (i.e. a fully connected subgraph) in a suitably defined association graph.



Different forms of exact matching

- **Properties**
  - The matching problems are all NP-complete except for graph isomorphism, which has not yet been demonstrated whether in NP or not.
  - Exact graph matching has exponential time complexity in the worst case. However, in many PR applications the actual computation time can be still acceptable.
  - Exact isomorphism is very seldom used in PR. Subgraph isomorphism and monomorphism can be effectively used in many contexts.
  - The MCS problem is receiving much attention.

Algorithms for exact matching

- **Techniques based on tree search**
  - mostly based on some form of tree search with backtracking
    Ullmann's algorithm
    Ghahraman's algorithm
    VF and VF2 algorithm
    Bron and Kerbosh's algorithm
    Other algorithms for the MCS problem

- **Other techniques**
  - based on A* algorithm
    Demko's algorithm
  - based on group theory
    Nauty algorithm

Techniques based on tree search

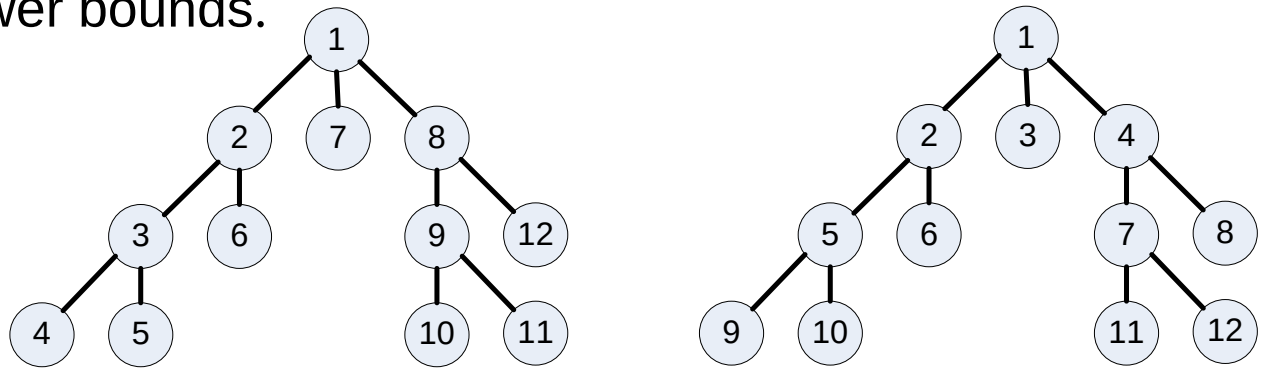
- **The basic idea**
  - A partial match (initially empty) is iteratively expanded by adding to it new pairs of matched nodes.
  - The pair is chosen using some necessary conditions, usually also some heuristic condition to prune unfruitful search paths.
  - Eventually, either the algorithm finds a complete matching, or no further vertex pairs may be added (*backtracking*)
  - For PR the algorithm may consider the attributes of nodes and edges in constraining the desired matching.

Techniques based on tree search

- **The backtracking algorithm**
  - depth-first search(DFS):

    it progresses by expanding the first child node of the search tree that appears and thus going deeper and deeper until a goal node is found, or until it hits a node that has no children.
  - Branch and bound(B&B):

    it is a BFS(breadth-first search)-like search for the optimal solution. Branch is that a set of solution candidates is splitted into two or more smaller sets; bound is that a procedure upper and lower bounds.



Techniques based on tree search

- **Ullmann's algorithm**
  - Probably the most popular graph matching algorithm
  - Application for graph isomorphism, subgraph isomorphism and monomorphism, also for MCS problem
  - A refinement procedure based on matrix of possible future matched node pairs to prune unfruitful matches
  - The simple enumeration algorithm for the isomorphisms between a graph G and a subgraph of another graph H with the adjacency matrices A_G and A_H

    An M' matrix with $|V_G|$ rows and $|V_H|$ columns can be used to permute the rows and columns of A_H to produce a further matrix P. If $(a_{G\,i,j} = 1) \Rightarrow (p_{i,j} = 1)$, then M' specifies an isomorphism between G and the subgraph of H.

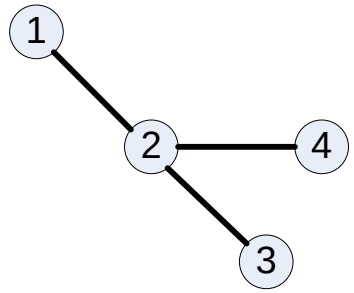
    $$P = M'(M' A_H)^T$$

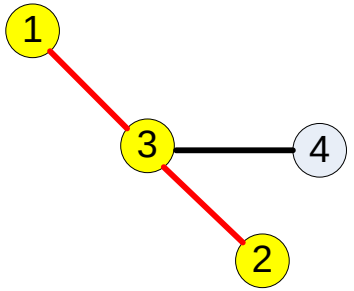
[J.R.Ullmann.An algorithm for subgraph isomorphism.]

Techniques based on tree search

- **Ullmann's algorithm**
  - Example for permutation matrix
  - The elements of M' are 1's and 0's, such that each row contains 1 and each column contains 0 or 1

H





$$P = M'(M'A_H)^T$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}^T \right)$$

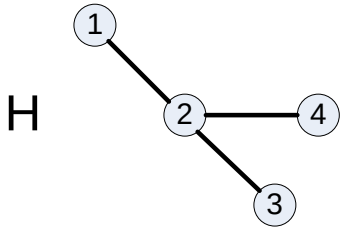
$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Techniques based on tree search

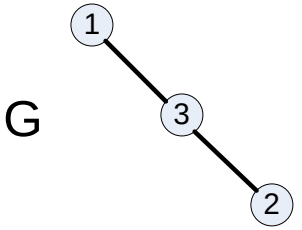
- **Ullmann's algorithm**
  - Construction of another matrix $M^{(0)}$ with the same size of M'

$$m_{i,j}^{(0)} = \begin{cases} 1 & \text{if } \deg(V_{Hi}) \geq \deg(V_{Gi}) \\ 0 & \text{otherweise} \end{cases}, m_{i,j} \in \{0,1\}$$

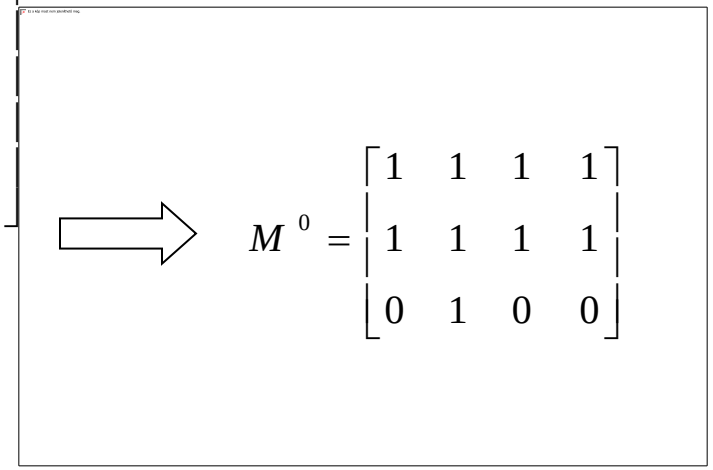
  - Generation of all M' by setting all but one of each row of $M^{(0)}$
  - A subgraph isomorphism has been found if $(a_{G\,i,j} = 1) \Rightarrow (p_{i,j} = 1)$



$$A_H = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$



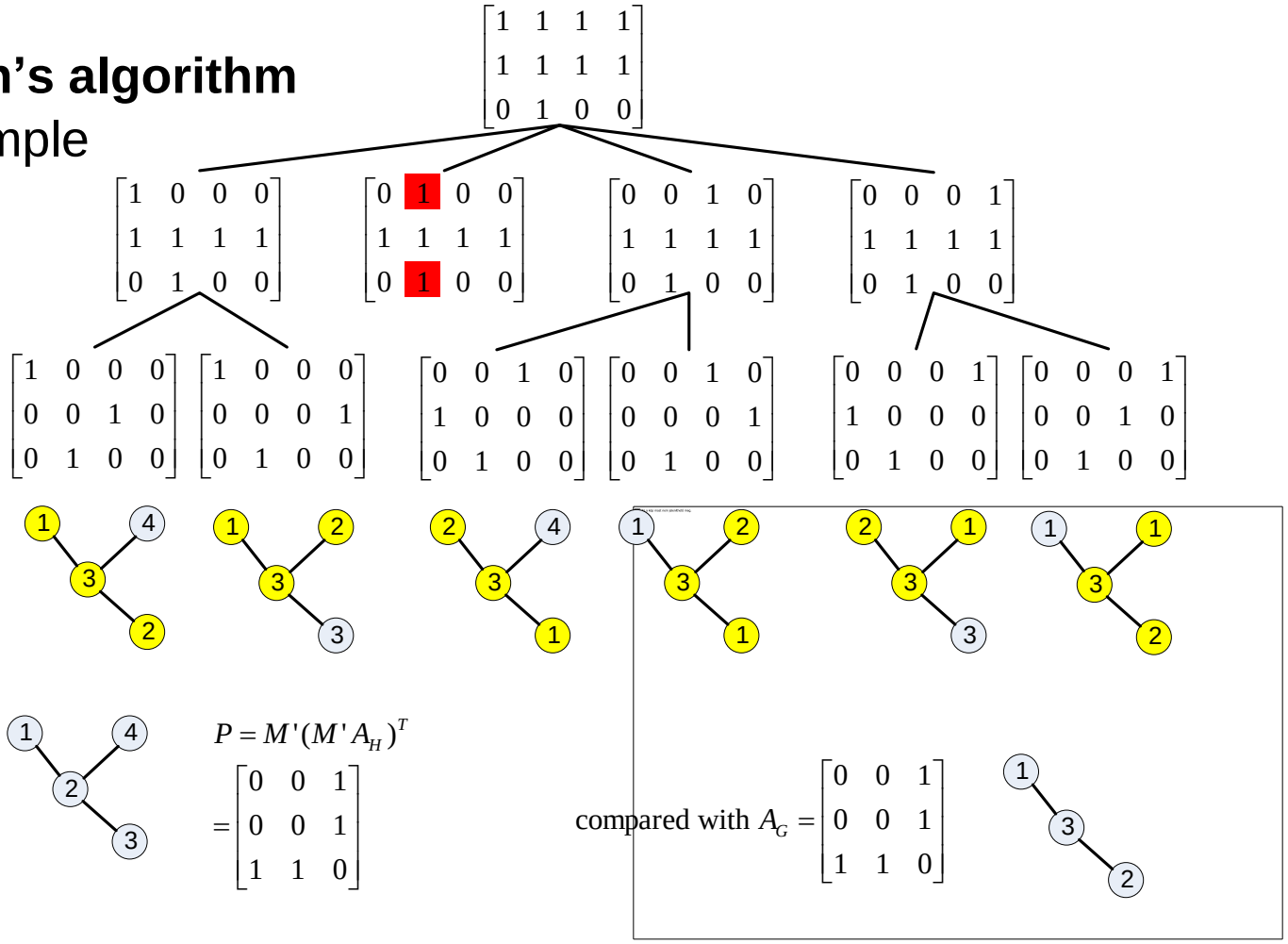
$$A_G = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$



$$M^0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Techniques based on tree search

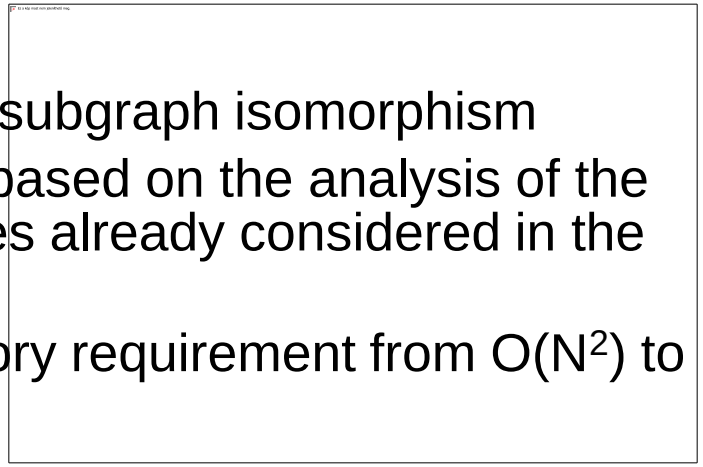
- **Ullmann's algorithm**
  - Example



Techniques based on tree search

- **Ghahraman's algorithm**
  - Application for monomorphism
  - Use of the *netgraph* obtained from the Cartesian product of the nodes of two graph, monomorphisms correspond to particular subgraphs of the netgraph
  - A strong necessary condition and a weak one, then two versions of the algorithm to detect unfruitful partial solutions

- **VF and VF2 algorithm**
  - Application for isomorphism and subgraph isomorphism
  - VF algorithm defines a heuristic based on the analysis of the sets of nodes adjacent to the ones already considered in the partial mapping.
  - VF2 algorithm reduces the memory requirement from $O(N^2)$ to $O(N)$.



Techniques based on tree search

- **Bron and Kerbosh's algorithm**
  - Application for the clique detection and the MCS problem
  - Based on the use of a heuristic for pruning the search tree
  - Simplicity and an acceptable performance in most cases

- **Other algorithms for the MCS problem**
  - Balas and Yu's alogrithm also defines a heristic, but based on graph colouring techniques.
  - McGregor's alogrithm is not applicated for a maximum clique problem.
  - Koch's alogrithm is applicated for a slightly simplified version of the MCS problem and suggests the use of the Bron and Kerbosh's algorithm.

Other techniques

- **Based on the A* algorithm**
  - It uses a distance-plus-cost heuristic function to determine the order in which the search visits nodes in the tree. The heuristic is a sum of two functions:
    the path-cost function, i.e. the cost from the starting node to the current node, and an admissible “heuristic estimate” of the distance to the goal.
  - Demko’s algorithm investigates a generalization of MCS to hypergraphs.

- **Based on group theory**
  - McKay’s Nauty(No automorphisms yes?) algorithm deals only with the isomorphism problem.

    It constructs the automorphism group of each of the input graphs and derives a canonical labeling. The isomorphism can be checked by verifying the equality of the adjacency matrices.

Conclusion

- **The exact graph matching problem is of interest in a variety of different pattern recognition contexts.**
- **Of the exact graph matching problems, exact isomorphism is very seldom used in PR while subgraph isomorphism, monomorphism and the MCS problem are popularly proposed.**
- **Ullmann' algorithm, VF2 algorithm and Nauty algorithm are mostly used algorithms, which based on the search methods, and may outperform others. Most modified algorithms adopt some conditions to prune the unfruitful partial matching.**

Thank you!