On some algorithmic problems MPRI 2017–2018

Michel Habib
habib@irif.fr
http://www.irif.fr/~habib

Sophie Germain, octobre 2017

A hierarchy of graph models

1. Undirected graphs (graphes non orientés)
2. Tournaments (Tournois), sometimes 2-circuits are allowed.
3. Signed graphs (Graphes signés) each edge is labelled + or - (for example friend or enemy)
4. Oriented graphs (Graphes orientés), each edge is given a unique direction (no 2-circuits)
   An interesting subclass are the DAG Directed Acyclic Graphs (graphes sans circuit), for which the transitive closure is a partial order (ordre partiel)
   Duality comparability – cocomparability
   (graphes de comparabilité – graphes d'incomparabilité)
5. Directed graphs or digraphs (Graphes dirigés)

Other variants

1. 2-structures (kind of automaton)
2. Hypergraphs or set families

Problems have to be defined in each model and sometimes it could be hard.

- What is the right notion for a coloration in a directed graph ?
- No directed cycle unicolored, seems to be the good one.
- It took 20 years to find the right notion of oriented matroïd
- What is the right notion of treewidth for directed graphs ?
- Still an open question. It seems that all tentative definitions loose many properties of the undirected case treewidth.

Can we use partition refinement on all these types of graphs ?

Some imprecision (fuzziness) in the data

- Variant of the Sandwich 3 types of edges : safe, possible, forbidden.
  A realization associate to each possible edge exactly one of the two other modalities.
- Trigraphs (introduced by M. Chudnovsky) : the property must be true for all realizations.
  Sandwich Problems : there exists at least one realization for which the property is true.

For partial orders, comparability graphs or uncomparability graphs the independant set and maximum clique problems are polynomial.

Second Neighborhoods Conjecture

P.D. Seymour 1990

Every digraph without 2-circuits has a vertex with at least as many second neighbors as first neighbors.

Second neighbors, $SN(x)$ is the set of vertices at exact distance 2 of x.

Therefore we are looking for x such that $|SN(x)| \geq |N(x)|$.

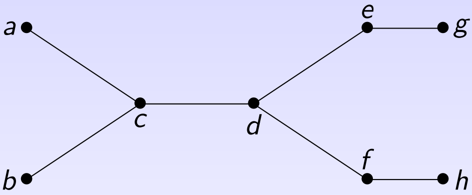
- If G has a sink then the results is true.
- So the conjecture is true for DAGs.
- The interesting case is for strongly connected graphs.

Degrees parts

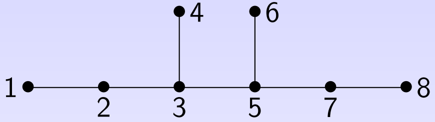
Classification of the vertices in parts having the same degree.
A variation of the folklore algorithm for twins.

Generalized degree partition

Classification of the vertices in parts having the same degree with respects to the other parts. To compute this partition we can use a variation of the partition refinement.
DegreeRefine(P, S) :
computes the partition of S in parts having same degree with P
The computation of this partition is the first step of the main isomorphism algorithms.



$$P(T) = \{\{c, d\}_3, \{e, f\}_2, \{a, b, g, h\}_1\}$$
$$P_{final}(T) = \{\{d\}_{3(3,2,2)}, \{c\}_{3(3,1,1)}, \{e, f\}_2, \{a, b\}_{1(3)}, \{g, h\}_{1(2)}\}$$



$$P(T') = \{\{3,5\}_3, \{2,7\}_2, \{1,4,6,8\}_1\}$$
$$P_{final}(T') = \{\{3,5\}_3, \{2,7\}_2, \{4,6\}_{1(3)}\{1,8\}_{1(2)}\}$$

- $P(T) = \{\{c, d\}_3, \{e, f\}_2, \{a, b, g, h\}_1\}$
  $P(T') = \{\{3, 5\}_3, \{2, 7\}_2, \{1, 4, 6, 8\}_1\}$
  These two degree partitions are isomorphic but T and T' are not isomorphic.
- $P_{final}(T) =$
  $\{\{d\}_{3(3,2,2)}, \{c\}_{3(3,1,1)}, \{e, f\}_2, \{a, b\}_{1(3)}, \{g, h\}_{1(2)}\}$
  $P_{final}(T') = \{\{3, 5\}_3, \{2, 7\}_2, \{4, 6\}_{1(3)}\{1, 8\}_{1(2)}\}$
  But their two generalized degree partitions are not isomorphic.

Proposition

Two trees T and T' are isomorphic iff their generalized degree partitions are isomorphic

Proof

Clearly if two graphs are isomorphic their generalized degree partitions must be isomorphic.
Let us consider the converse in the case of trees.
By induction on $|T| = |T'|$ and deleting one leaf in each tree denoted by x, x' (these leaves must belong to isomorphic parts of the generalized degree partitions).
The generalized degree partitions of $T \setminus x$ and $T' \setminus x'$ are still isomorphic but parts could be different due to some merging of parts from T and T'.

How to compute this generalized degree partition for a given graph G ?

- The first degree partition can be computed in $O(|V(G)| + |E(G)|)$
- Generalized degree partition can be computed using some partition refinement techniques.

Unfortunately

If G is regular then $P(G) = P_final(G)$

Vertex partitioning

Les exemples difficiles pour les algorithmes : les graphes fortement réguliers définis à l'aide de 3 paramètres entiers :
(r, λ, μ)
La classe $G(r, \lambda, \mu)$ contient les graphes réguliers de degré r tels que :
$\forall x, y \in V(G)$, $xy \in E(G)$ alors $|N(x) \cap N(y)| = \lambda$
$\forall x, y \in V(G)$, $xy \notin E(G)$ alors $|N(x) \cap N(y)| = \mu$

Exercises

X a finite set
$\mathcal{F}$ a family of subsets $S_1, \ldots, S_k$ de X.

A-1 Write an algorithm to check if there is no i, j such that : $S_i \subseteq S_j$

A-2 **Disjoint Set Problem :**
Find i, j such that $S_i \cap S_j = \emptyset$

A-3 Check if : $\forall i, j,\ i \neq j\ S_i \cap S_j = \emptyset$

A-4 Check if there exists i such that : S_i intersects all other subsets of $\mathcal{F}$

A-5 Compute the set of maximal (resp. minimal) for inclusion of $\mathcal{F}$

A-6 Check if $\mathcal{F}$ is laminar.

A-7 Check if $\mathcal{F}$ satisfies the Helly property

Solutions

A-1 Not linear

A-2 Not linear via a reduction to SETH

A-3 Linear using a simple scan of the sets, halting when some element is marked twice.

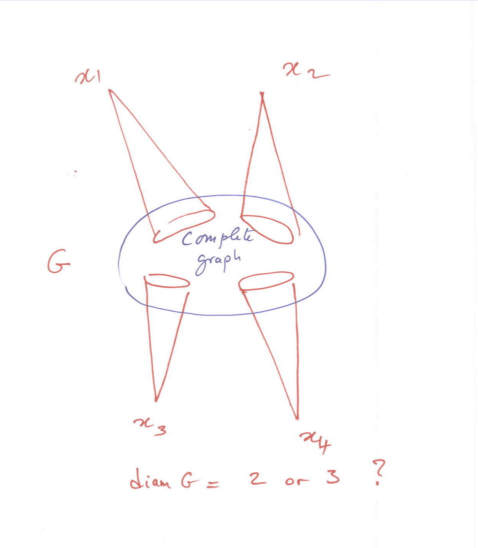
A-4 Linear using partition refinement.

A-5 Harder than A-1

A-6 Linear using partition refinement.

A-7 Hard, because we have to test for all subset of indexes

Chordal graphs and split graphs



Disjoint sets problem

Disjoint sets problem

A finite set X, $\mathcal{F}$ a collection $\{S_1, \ldots, S_k\}$ of subsets of X.
$\exists i, j \in [1, k]$ such that $S_i \cap S_j = \emptyset$?

Linearity

Can this problem be solved in linear time ?
Size of the problem : $|X| + k + \sum_{i=1}^{i=k} |S_i|$
size of the incidence bipartite graph

SETH : Strong Exponential Time Hypothesis

SETH

There is no algorithm for solving the k-SAT problem with n variables in $O((2-\epsilon)^n)$ where ϵ does not depend on k.

Let us consider an instance I of $k - SAT$ with 2n boolean variables $x_1, \ldots, x_{2n}$, and a set $\mathcal{C}$ of m clauses $C_1, \ldots C_m$, we build an instance of Disjoint-set problem as follows :

- The gound set X is the set of clauses + 2 extras vertices a, b.
- We consider now A, B the sets of all truth assignments of $x_1, \ldots, x_n$, and $x_{n+1}, \ldots x_{2n}$, respectively.
- For each truth t assigment in A (resp. in B) we define $S_t = \{C \in \mathcal{C}$ such that t does not satisfy $C\} \cup \{a\}$ (resp. $\cup\{b\}$).

- The sets $S's$ defined with A (resp. B) always intersect because of a (resp. b).
- If there exists S_u, S_v that do not intersect. Necessarily u is a truth assignment in A and v in B (or the converse, but they cannot be on the same set of variables because of the dummy vertices a, b).
  This means that for each clause C_i of I, if $C_i \notin S_u$, then the truth v assignment satisfies C_i.
  Similarly if $C_i \notin S_v$, then the truth u assignment satisfies C_i.
  But $S_u \cap S_v = \emptyset$ means that for every clause C_i either :
  $C_i \notin S_u$ or $C_i \notin S_v$.
- Therefore :
  I is satisfiable iff there exist 2 disjoint sets S_u, S_v.

Complexity issues

- Size of the $k - SAT$ instance is bounded by :
  $K = 2n + m + km$
- Size of the Disjoint set instance :
  $N = 2^{n+1} + m + 2$ vertices
  and at most $M = m2^{n+1}$ edges.
- To compute this instance we need to evaluate the m, k-clauses for each half-truth assigment.
  Can be done in $O(K)$, so in the whole : $O(2^{n+1}K)$.
- If there exists an algorithm for the Disjoint set problem in less than $O(NM^{1-\epsilon})$
  it would imply an algorithm for $k - SAT$ in less than $O((2-\epsilon)^{2n})$ contradiction the SETH.

Consequences

Practically there is no hope to design a linear time algorithm for :

1. Disjoint set problem
2. Diameter computations for chordal graphs and split graphs
3. And many other related problems ... such as betweenness centrality
4. but not all $O(mn)$ problems as for example transitive closure, existence of a triangle ...

Research Problem

- Since sparse graphs are not available for the above reduction.
- Can we compute in linear time the diameter of planar graphs ?
- This class contains all grids !
- Very hot subject. Every year a paper at Soda or Focs or Stocs on this problem.

Notes

- They are some variations on this proof.
- Disjoint set problem is also called the orthogonal vector problem.
- We have the same type of results for the radius computation via the hitting set problem, generalisation of problem A_4

Similar problems

1. Recognition of permutation graphs
2. Consecutive one property
3. Transitive orientation of a comparability graph
4. Planarity testing
5. Decomposition of boolean matrices
6. Robinsonian matrices
7. Other problems on symmetric positive matrices.

Recognition of Permutation graphs

PERMUTATION(G) :

Input: A connected graph $G = (V, E)$, cocomp ordering σ of G and cocomp ordering τ of $\overline{G}$

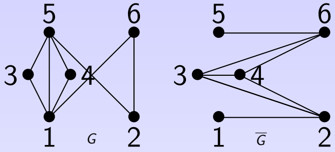
Output: The message that at least one of σ, τ is not a cocomp ordering of its graph $(G, \overline{G})$ or total orderings π_1, π_2^{dual} of $1, 2, \cdots |V|$ that certify that G is a permutation graph

$P_{G(\tau)} \leftarrow$ an acyclic orientation of G using τ ;

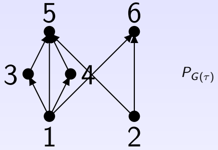
$\pi_1 \leftarrow \mathit{dfgreedy}^+(P_{G(\tau)}, \sigma)$;

$\pi_2 \leftarrow \mathit{dfgreedy}^+(P_{G(\tau)}, \sigma^{dual})$;

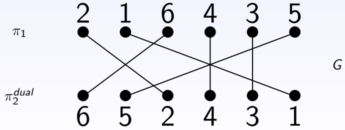
Check if π_1, π_2^{dual} represent G as a permutation graph ;



$\sigma = 1, 3, 5, 4, 2, 6$ is a cocomp ordering of G, but not a cocomp ordering of $\overline{G}$ (see umbrella (1, 5, 2))
$\tau = 1, 2, 3, 4, 5, 6$ is a cocomp ordering of $\overline{G}$, but not a cocomp ordering of G (see umbrella (2, 3, 6))



$\pi_1 = 2, 1, 6, 4, 3, 5 = dfgreedy(P_{G(\tau)}, \sigma)$
$\pi_2 = 1, 3, 4, 2, 5, 6 = dfgreedy(P_{G(\tau)}, \sigma^{dual})$



π_1 and π_2 are cocomp orderings of both G and $\overline{G}$

- dfgreedy is a graph search applied on posets and if σ is a cocomp then $dfgreedy^{+}(P_{G(\tau)}, \sigma)$ is also a cocomp.
- If G is a permutation graph, $dfgreedy^{+}(P_{G(\tau)}, \sigma)$ and $dfgreedy^{+}(P_{G(\tau)}, \sigma^{dual})$ provide a representation of G as a permutation graph.