# Fixed parameter and exponential algorithms

Journée COMRED, 01/03/2010

Frédéric Havet

# Context

Many (combinatorial) optimisation problems are NP-hard = no polynomial-time algorithms to solve them.

WE NEED TO SOLVE THEM.

Several classical approaches:

- approximation algorithms;
- randomised algorithms;
- heuristics.

Drawback: do not give the optimal solution.

WE NEED EXACT ALGORITHMS.

# Two (new) approaches

Exact exponential time algorithm: running time in $O(c^n)$ with $c$ as small as possible. $\Rightarrow$ if $c$ is small, one can solve large size instances.

Fixed parameter algorithm: running time in $f(k)P(n)$ with

- $k$ parameter (well chosen),
- $f$ function,
- $P$ polynomial.

$\Rightarrow$ if $k$ is small, one can solve.

# Optimisation problem and parameterisation

NP minimisation problem

Instance: $x \in \Sigma^*$ with $\Sigma$ a finite alphabet.

Goal: Find $\min\{cost(x,y) \mid y \in sol(x)\}$ with

- $sol(x)$ set of solutions of $x$;
- $cost : \{(x,y) \mid y \in sol(x)\}$.

Associated parameterized problem

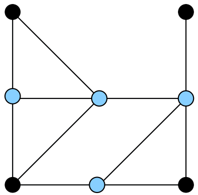
Instance: $x \in \Sigma^*$ and an integer $k$.

Parameter: k.

Question: $\min\{cost(x,y) \mid y \in sol(x)\} \leq k$?

# Example 1: Vertex cover

Vertex cover = set of vertices $C$ such that every edge has an endvertex in $C$.

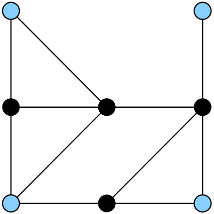


Minimum Vertex Cover Problem:
Input: Graph $G$.
Output: A vertex cover of $G$ of minimum size.

Parameterised Vertex Cover Problem:
Input: Graph $G$ and integer $k$.
Parameter: k.
Question: Does $G$ have a vertex cover of size (at most) $k$ ?

# Example 2: Maximum Independent Set

independent set = set of pairwise non-adjacent vertices.

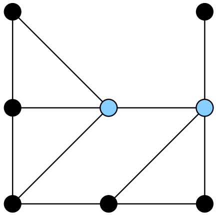


Maximum Independent Set Problem:
Input: Graph $G$.
Output: An independent set of maximum size in $G$.

Parameterised Independent Set Problem:
Input: Graph $G$ and integer $k$.
Parameter: k.
Question: Does $G$ have an independent of size (at least) $k$ ?

# Example 3: Minimum Dominating Set

dominating set = set $D$ of vertices such that $D \cup N(D) = V(G)$.

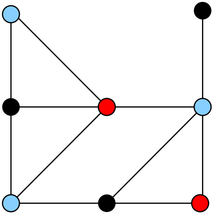


Minimum Dominating Set Problem:
Input: Graph $G$.
Output: A dominating set of minimum size in $G$.

Parameterised Dominating Set Problem:
Input: Graph $G$ and integer $k$.
Parameter: k.
Question: Does $G$ have a dominating set of size (at most) $k$ ?

# Example 4: Chromatic Number

colouring $= c\colon V(G) \to S$ s. t. $c(u) \neq c(v), \forall uv \in E(G)$.



Chromatic Number Problem:

Input: Graph $G$.

Output: A colouring with minimum number of colours $G$.

$k$-Colourability Problem:

Input: Graph $G$ and integer $k$.

Parameter: k.

Question: Is $G$ $k$-colourable? ($\chi(G) \leq k$?)

# FPT problems

A parameterized problem is Fixed Parameter Tractable if it is decidable in time $f(k)n^c$.

FPT implies polynomial-time solvable for any fixed $k$.

Vertex Cover, Independent Set, Dominating Set: OK
Trivial algorithm in $O(n^k)$.

Chromatic number is not FPT since 3-colourability is NP-complete.

# Parameterized complexity theory

$$P \subseteq FPT \subseteq \underbrace{W[1] \subseteq W[2] \subseteq \ldots \subseteq W[P] \subseteq XP}_{\text{presumably fixed-parameter intractable}}$$

Conjecture FPT $\neq$ W[1], and more generally W[i] $\neq W[i+1]$

$$P = NP \Rightarrow FPT = W[1]$$

but the converse seems not to hold.

Examples: Vertex Cover is FPT, Independent Set is W[1], Dominating Set is W[2]

# Difference between W[1] and W[2]

Independent Set: $\exists(x_1, \dots, x_k), \ \forall i \neq j, \ x_i x_j \notin E$

Dominating Set: $\exists(x_1, \dots, x_k), \ \forall v \in V, \ \exists i, \ x_i v \in E$ or $x_i = v$

One more level of quantifiers for Dominating Set.

# Algorithmic methods for fixed parameter algorithms

- Data reduction and problems kernels
- Depth-bounded search trees
- Color Coding
- Iterative compression
- Tree decomposition, minor theory

# Data reduction and problems kernels

Observation: If a vertex is incident to more than $k$ edges, it must be in every vertex cover of size at most $k$.

Buss' reduction for Vertex Cover: All vertices with degree $> k$ are added to the vertex cover.

In the resulting graph $G'$ each vertex has degree at most $k$. Then i$G'$ has a vertex cover of size $k' \leq k$, then it contains at most $k^2 + k$ vertices and at most $k^2$ edges.

Brute force: check the $\binom{k^2+k}{k'}$ possibilities $\Rightarrow$ algo in time $g(k) + n$.

# Kernelization

$f(k)$-kernelization: polynomial-time algorithm
instance $(G, k) \longrightarrow$ instance $(G', k')$ such that:

- $(G', k')$ is equivalent to $(G, k)$;
- $k' \leq k$ and $|G'| \leq f(k)$.

Kernelization + brute force $= O(g(k) + n^c)$ time algo.

Theorem: A parameterized problem is Fixed Parameter Tractable if and only if it has a kernelization.

# Kernel race

We want kernel but also as small as possible kernels.

Techniques for finding small kernels:

- Integer Linear Programming,
- Crown decomposition,
- ...

Techniques for non-existence of small kernel (under complexity assumptions):

- Distillation,
- Coloured strengthening,
- ...

# ILP-formulation of Vertex Cover

$$\begin{array}{rll} \text{Minimise} & \sum_{v \in V} x_v & \\ \text{Under :} & x_u + x_v \geq 1 & \forall uv \in E \\ & x_v \in \{0,1\} & \forall v \in V \quad \textit{relaxation } 0 \leq x_v \leq 1 \end{array}$$

The relaxation has an half-integral solution, i.e. $x_v \in \{0, 1/2, 1\}$.

For $t \in \{0, 1/2, 1\}$, set $V_t = \{v \in V \mid x_v = t\}$ and $G_t = G\langle V_t \rangle$.

Obs: $vc(G_{1/2}) \geq \frac{1}{2}|V_{1/2}|$.

[Nemhauser et Trotter '75] There is a minimum vertex cover $C$ of $G$ such that: $V_0 \cap C = \emptyset$ and $V_1 \subseteq C$.

# LP-based kernelization of Vertex Cover

1. Find an optimal solution $x$ of the fractional relaxation.
2. If the weight of $x$ is greater than $k$, return a “no”-instance.
3. Else return $(G_{1/2}, k - |V_1|)$.

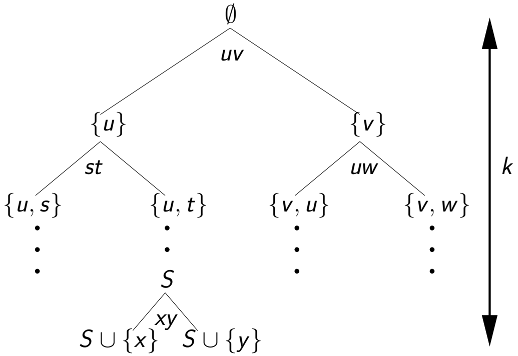
By the Observation, $|G_{1/2}| \leq 2(k - |V_1|) \leq 2k$.

So, we have a $2k$-kernel.

# Depth-bounded search trees

Idea: In polynomial time find a small subset s. t. at least one element of this subset is part of an optimal solution.

Vertex Cover: small subset = two endvertices of an edge.
$\Rightarrow$ binary tree of depth at most $k$. Time $2^k.n$.



# Hardness results for Maximum Independent Set

- NP-hard [Karp '72]
- Not approximable within $O(n^{1-\epsilon})$ unless P $=$ NP [Zucherman '06]
- No exact $O(c^{o(n)})$ algorithm unless SNP $\subseteq$ SUBEXP [Impagliazzo,Paturi,Zane '01]
- W[1]-hard [Downey & Fellows '92]

$\Rightarrow$ The best we can hope for is a $O(c^n)$ exact algorithm for some small constant $c \in ]1,2]$.

# Race for Maximum Independent Set

- $O(1.261^n)$ poly-space [Tarjan &Trojanowski '77]
- $O(1.235^n)$ poly-space [Jian '86]
- $O(1.228^n)$ poly-space, $O(1.221^n)$ exp-space [Robson '86]
- better results for sparse graphs [Beigel99, Chen,Kanj & Xia '03]
- simpler $O(1.221^n)$ exp-space [Fomin, Grandoni and Kratsch '06]

# New techniques for exact exponential algorithms

Design techniques: Branch and Recharge, Inclusion-Exclusion, ...

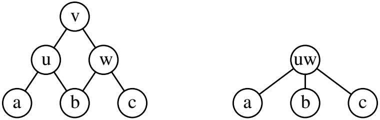
Running time analysis techniques: Measure and Conquer, ...

# Reduction rules for Maximum Independent Set

components: $C$ connected component $\alpha(G) = \alpha(C) + \alpha(G - C)$.

dominance: if $N[w] \subset N[v]$ $\alpha(G) = \alpha(G - v)$.

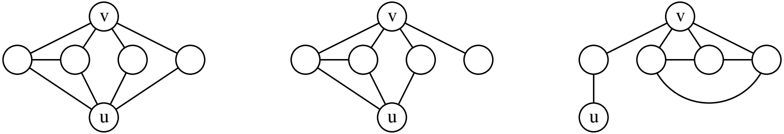
folding: if $v$ has 2 neighbours $v$ and $w$ $\alpha(G) = 1 + \alpha(\tilde{G})$.




mirroring: $M$ set of mirrors of $v$

$$\alpha(G) = \max\{\alpha(G - v - M), 1 + \alpha(G - N[v])\}$$

$u$ is a mirror of $v$ if $d(u, v) = 2$ and $N(v) \setminus N(u)$ is a clique.



# Algorithm for Maximum Independent Set

$\texttt{mis}(G)$

1. **if** $|V(G)| \leq 1$ **return** $|V(G)|$;
2. **if** $\exists$ component $C$ of $G$ **return** $\texttt{mis}(C) + \texttt{mis}(G - C)$;
3. **if** $\exists$ $v$ and $w$ s.t. $N[w] \subset N[v]$ **return** $\texttt{mis}(G - v)$;
4. **if** $\exists$ $v$ s.t. $d(v) = 2$ **return** $1 + \texttt{mis}(\tilde{G})$;
5. pick a vertex $v$ of max. degree;
6. **return** $\max\{\texttt{mis}(G - v - M), 1 + \texttt{mis}(G - N[v])\}$

# Standard analysis

l.5 & 6 Branching Step

$d(v) = 3$: when discarding $v$, we also discard a mirror of $v$, or at next step we fold a neighbour of $v$, so we remove at least 2 vertices; when selecting we remove at least 4 vertices. $\Rightarrow$ $B(n) \leq B(n-2) + B(n-4)$.

$d(v) \geq 4$: we remove at least one or five vertices $\Rightarrow$ $B(n) \leq B(n-1) + B(n-5)$.

$B(n) = O^*(\lambda^n)$ with $\lambda = 1.3247$ the largest root of $x^4 - x^2 - 1$ and $x^5 - x^4 - 1$.

# Measure and conquer

In the standard analysis, measure of a graph: $\mu(G)$ = number of vertices.

Idea: take into account the fact that reducing the degree has a positive impact on the long term. (degree 2 vertices are removed without branching)

New measure: $n_i$: number of vertices of degree $i$.
$\alpha_i$: weight (in $[0,1]$) of every vertex of degree $i$.

$$\mu'(G) = \sum_{i \geq 0} \alpha_i n_i$$

We obtain new recurrence relations and by choosing adequate value of $\alpha_i$, we get $B(n) = O^*(1.221^n)$.