6.890	Lecture 13	Oct. 16, 2014

Parameter k = function: instance $\to \mathbb{N}$
- usually one of the numbers in instance
- sometimes hard to compute e.g. OPT

[Downey & Fellows 1999]

Parameterized problem = decision problem + parameter
- e.g. (k-)Vertex Cover: is there a vertex cover of $\le k$?
  k is the natural parameter: comparing with OPT
- e.g. Vertex Cover with respect to OPT (Vertex Cover)
  - similar but k not given
  - for $k = 0, 1, 2, \ldots$: run k-Vertex Cover
- e.g. Vertex Cover w.r.t. crossing number

XP = {parameterized problems solvable in $n^{f(k)}$ time}

bad

Fixed-parameter tractable (FPT)

good

= {parameterized problems solvable in $f(k) \cdot n^{O(1)}$ time}
= {parameterized problems solvable in $f(k) + n^{O(1)}$ time}
- motivation: confine exponential to parameter k which may be $\ll$ problem size n

Example: (k-)Vertex Cover
- $\in$XP: guess k vertices, test coverage $|V|^k \cdot |E|$
- $\in$FPT: take edge, guess endpoint, delete, repeat

2^k "bounded search tree technique" depth $\le k$

EPTAS $\subseteq$ PTAS with running time $f(1/\varepsilon)\cdot n^{O(1)}$
- i.e. FPT w.r.t. $1/\varepsilon$ (cf. $n^{1/\varepsilon}$ etc.)
- $\Rightarrow$ FPT w.r.t. natural parameter k ($\Rightarrow$ w.r.t. OPT)
  - set $\varepsilon = 1 + 1/2k$
- $\notin$ FPT $\Rightarrow$ $\notin$ EPTAS

Parameterized reduction: $(A, k) \to (B, k')$

instance x of A $\xrightarrow{f}$ instance $x' = f(x)$ of B

- $f(k(x))\cdot|x|^{O(1)}$ time $\Rightarrow |x'| \le f(k(x))\cdot|x|^{O(1)}$
- answer preserving: x YES for $A \Leftrightarrow x'$ YES for B
  (just like NP/Karp reductions)
- parameter preserving: $k'(x') \le g(k(x))$
  for some $g: \mathbb{N} \to \mathbb{N}$

  $\forall x$

  parameter blowup
- $B \in$ FPT $\Rightarrow A \in$ FPT

Nonexample: independent set $\to$ vertex cover

$(G, k) \mapsto (G, n-k)$

- preserves answer but not parameter
- indeed, vertex cover $\in$ FPT
  but independent set is W[1]-hard
  $\Rightarrow \notin$ FPT unless FPT = W[1]

Example: independent set $\to$ clique (or vice versa)

$(G, k) \mapsto (\overline{G}, k)$

Canonical hard problem for W[1]: (analogy to NP)

k-step nondeterministic Turing machine

- given nondeterministic Turing machine

  code, state, finger to k-cell memory

  O(n) lines   O(n) options   O(n) states

  (guess can have n choices/branches)
- does some choice sequence finish in k steps?

Reduction to Independent Set:

- k^2 cliques. $k' = k^2 \Rightarrow 1$ node per clique
- clique (i,j) represents memory cell i at time j (n choices) + state of machine (e.g. PC = which of n instructions next)
- add edges to forbid certain transitions $j \to j'$; omit edges for allowed nondet. trans.

Reduction from Independent Set: $k' = \Theta(k^2)$

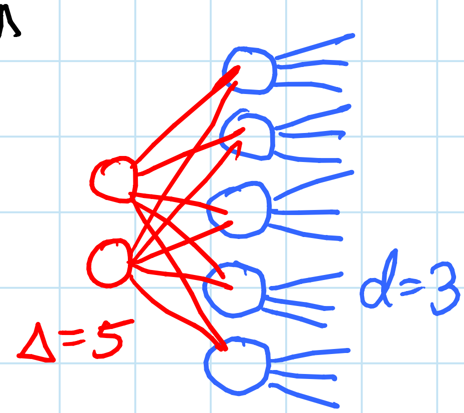
- guess k vertices — $\Theta(k)$
- for each pair of these vertices: — $\Theta(k^2)$

  check no edge (lookup table in code)

$\Rightarrow$ both W[1]-complete

Clique in regular graphs: reduction from Clique

- Δ = max. degree
- Δ copies of graph
- vertex v of degree $d \rightarrow v_1, v_2, \ldots, v_\Delta$ copies
  - add $\Delta - d$ vertices
  - biclique between ↑ & ↗
  - $\Rightarrow \Delta$-regular
- add no cliques (≥ 3): new vertices in no Δ



Independent set in regular graphs — just take complement

Partial vertex cover:

are there k vertices that cover ℓ edges?

- FPT w.r.t. ℓ
- W[1]-complete w.r.t. k

Reduction from Independent set in regular graphs:

Δ

- $k' = k$
- $\ell' = \Delta k$

(based on upcoming book by Cygan, Fomin, Kowalik, Lokshtanov, Marx, Pilipczuk, Pilipczuk, Saurabh 2015: Parameterized Algorithms)

Multicolored clique: — like (Numerical) 3DM

- given graph & vertex k-coloring
- find k vertices, one of each color, that form a k-clique
- W[1]-complete [Pietrzak – JCSS 2003]

[Fellows, Hermelin, Rosamond, Vialette – TCS 2009]

Reduction from Clique:

- vertex $v \to k$ copies $v_1, v_2, \ldots, v_k$

  colors: $1, 2, \ldots, k$

- edge $(v, w) \to$ edges (v_i, w_j) $\underbrace{\forall i \neq j}_{\Rightarrow \text{proper coloring}}$
- $k' = k$
- k-clique $\Leftrightarrow$ k-colored k-clique

Reduction to Clique:

- nothing: coloring $\Rightarrow$ all cliques are multicolored

Multicolored independent set — just take complement

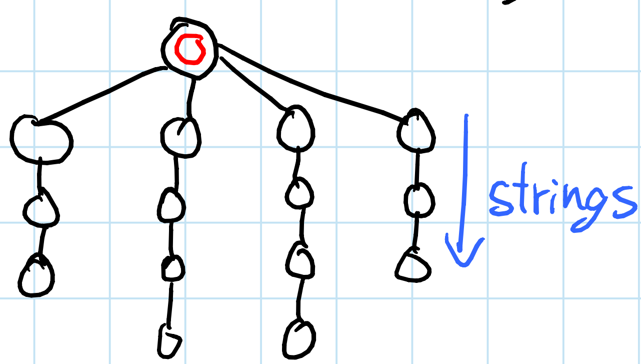
Shortest common supersequence:

- given k strings over alphabet Σ & number ℓ
- is there a common supersequence of length ℓ
- W[1]-hard w.r.t. k for $|\Sigma| = 2$ [Pietrzak – JCSS 2003]
  - reduction from Multicolored Clique

Reduces to restricted form where input strings never repeat character twice in a row parameterized by k & Σ

- add new symbol s_i after every character in string i ⇒ no repeats
- $k' = k$
- $|\Sigma'| = |\Sigma| + k$
- $\ell' = \ell +$ total length of input strings

Reduces to Flood-It on trees w.r.t. # colors ($|\Sigma|$) & # leaves (k)

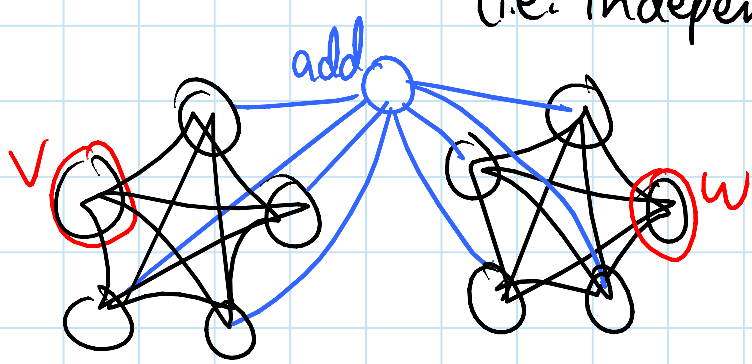


Dominating set:

(based on Cygan et al. book 2015)

Reduction from Multicolored independent set:

- vertex → vertex
- connect each color class in clique
  - also add 2 dummy vertices to each clique
- $k' = k$ ⇒ dominating set chooses one vertex from each clique, representing one vertex of each color in ind. set
- for each edge (v, w):
  - add vertex connected to all vertices in color classes of v & w, except v & w
  - ⇒ dominated ⟺ v & w not both chosen (i.e. independent set)



⇒ W[1]-hard
- W[2]-complete in fact

⇒ ∉ FPT unless FPT = W[2] (weaker assumption)

⇒ reverse reduction impossible unless W[1] = W[2]

Reduction to Set Cover: same as L11

- vertex v → set $N(v) \cup \{v\}$
- $k' = k$

Weighted Circuit SAT (Circuit k-Ones)

- given acyclic Boolean circuit & parameter k
- can we set k inputs to 1 to get output = 1?

W[P] = {parameterized problems reducible to Weighted Circuit SAT}

- **depth** = longest input → output path
- **weft** = max # big gates on input → output path
  ↳ not O(1) inputs: e.g. ≥ 3 inputs

W[t] = {parameterized problems reducible to O(1)-depth weft-t Weighted Circuit SAT}
= {parameterized problems reducible to depth-t output=AND Weighted Circuit SAT}
[Buss & Islam – TCS 2006]

W[*] = W[O(1)]

W[1]-complete:
- weighted O(1)-SAT (big AND of small ORs)

W[2]-complete:
- weighted CNF-SAT (big AND of big ORs)
- k-step 2-finger nondeterministic Turing machine
  = 2-tape

W[SAT] = reducible to SAT
- SAT → CNF-SAT reduction adds extra vars.
  so weighted problems not the same