Today: Fixed-parameter algorithms
- vertex cover
- fixed-parameter tractability
- kernelization
- connection to approximation

How fast can exact algorithms be? [Downey & Fellows 1997]

Idea: isolate exponential term to parameter
⇒ get fast solution for instances with small parameter value

Parameter = function: instance → \( \mathbb{N} \)
- often a “natural” parameter (\( k \) in input)
- not necessarily efficiently computable
- e.g. OPT

Parameterized problem = problem + parameter
“problem w.r.t. parameter”
(potentially many interesting parameterizations)
Example: k-Vertex Cover (NP-hard)
Given: graph $G = (V, E)$, $k \in \mathbb{N}$
Q: is there a set $S$ of $\leq k$ vertices
that "covers" all edges: $\forall e \in E \exists v \in S$ ne
Parameter: $k$

Note: can have $k \ll n$:

Trivial solution: (BAD)
- try all $\binom{n}{k} + \binom{n}{k-1} + \cdots + \binom{n}{0}$ sets of $\leq k$ vxs.
  can skip - bigger is better
- test coverage in $O(m)$ time
  $\Rightarrow O(n^k m)$ time
  - polynomial for fixed $k$
  - but not same polynomial - eg. not $O(n^{100})$
  - inefficient in most cases
  $\Rightarrow$ define $n^{f(k)}$ to be BAD
Bounded search-tree algorithm: \textit{(Good)}

- pick arbitrary edge \( e = (u, v) \)
- know that either \( u \in S \) or \( v \in S \) (or both)
- but don't know which
- guess: try both possibilities
  \begin{enumerate}
  \item add \( u \) to \( S \)
    - delete \( u \) & incident edges from \( G \)
    - recurse with \( k' = k - 1 \)
  \item ditto with \( v \) instead of \( u \)
  \end{enumerate}
- return OR of two outcomes
- like guessing in dynamic programming, but memoization doesn't help here

recursion tree:

- at leaf \((k = 0)\):
  - return \(|E| = 0\)
- \(O(V)\) time to delete \( u \) or \( v \)
- \(O(2^k \cdot V)\) time
  - \(O(V)\) for fixed \( k \)
  - degree of polynomial independent of \( k \)
  - also polynomial for \( k = O(\log V) \)
  - practical for e.g. \( k \leq 32 \)
- define \( f(k) \cdot n^{O(1)} \) to be \textit{Good}
FPT: A parameterized problem is fixed-parameter tractable (FPT) if there is an algorithm with running time \( f(k)n^{O(1)} \) where \( f: \mathbb{N} \to \mathbb{N} \) is a parameter (nonnegative) independent of \( k \) and \( n \).

**Question:** why \( f(k) \cdot n^{O(1)} \) not \( f(k) + n^{O(1)} \)?

**Theorem:** \( \exists f(k) \cdot n^c \) algorithm \( \iff \exists f'(k) + n^{c'} \) algorithm

**Proof:**

\( (\Leftarrow) \) trivial (assuming \( f'(k) \) & \( n^{c'} \geq 1 \))

\( (\Rightarrow) \)

if \( n \leq f(k) \) then \( f(k) \cdot n^c \leq f(k)^{c+1} \)

if \( f(k) \leq n \) then \( f(k) \cdot n^c \leq n^{c+1} \)

so \( f(k) \cdot n^c \leq \max \{ f(k)^{c+1}, n^{c+1} \} \)

\( \leq f(k)^{c+1} + n^{c+1} \)

\[ \frac{f'(k)}{c} \]

**Example:** \( O(2^k \cdot n) \Rightarrow O(4^k + n^2) \)
Kernelization: a simplifying self-reduction polynomial-time algorithm converting input \((x, k)\) into small equivalent input \((x', k')\)

\[|x'| \leq f(k) \iff \text{answer}(x) = \text{answer}(x')\]

Theorem: FPT \iff \exists kernelization

Proof: \((\Leftarrow)\) kernelize \(\Rightarrow n' \leq f(k)\)

run any finite \(g(n')\) algorithm

\[\Rightarrow n^{O(1)} + g(f(k))\] time

\((\Rightarrow)\) let \(A\) be an \(f(k) \cdot n^c\) algorithm

if \(n \leq f(k)\) then already kernelized

if \(f(k) \leq n:\)

- run \(A \Rightarrow f(k) \cdot n^c \leq n^{c+1}\) time \(\checkmark\)
- output \(O(1)\)-size YES/NO instance as appropriate (to kernelize)

assuming \(k\) is known

if \(k\) is unknown: run \(A\) for \(n^{c+1}\) time

& if not done, know already kernelized \(\square\)

So (exponential) kernel exists. Recent work aims to find polynomial (even linear) kernels when possible.
Polynomial kernel for k-vertex cover:
- make graph simple:
  - remove loops & multi-edges
- any vertex of degree \( \geq k \) must be in cover (else need \( \geq k \) vertices to cover inc. edges)
- remove such vertices (& incident edges) one at a time, decreasing \( k \) accordingly
  \( \Rightarrow \) remaining graph has max. degree \( \leq k \)
  \( \Rightarrow \) each remaining cover vertex covers \( \leq k \) edges
  \( \Rightarrow \) if \# remaining edges > \( k^2 \), answer is No:
    output canonical No instance
  - else \( |E'| \leq k^2 \)
- remove isolated vertices
  \( \Rightarrow |V'| \leq 2k^2 \)
  \( \Rightarrow \) reduced to instance \((V', E')\) of size \( O(k^3) \)
  in \( O(V+E) \) time \( \text{quadratic kernel} \)

- if we now apply:
  - trivial solution \( \Rightarrow O(V+E+(2k^2)^k k^2) = O(V+E+2^k k^{2k+2}) \) time
  - bounded solution \( \Rightarrow O(V+E+2^k k^2) \) time

Best algorithm to date: \( O(kV+1.274^k) \)
[Chen, Kanj, Xia - MFCS 2006]
Connection to approximation algorithms:
- take optimization problem, integral OPT
- consider associated decision problem: \( \text{OPT} \leq k? \)
- parameterize by \( k \)

**Theorem:** optimization problem has **EPTAS**

\[ \text{[L20]} \text{ efficient PTAS: } f(1/3) \cdot n^{O(1)} \]

\( \Rightarrow \) decision problem is FPT

**Proof:** (like PTAS \( \Leftrightarrow \) pseudo-poly. alg.)
- say maximization problem (& \( \leq k \) decision)
- run EPTAS with \( \varepsilon = \frac{1}{2k} \) in \( f(2k) \cdot n^{O(1)} \)
- relative error \( \leq \frac{1}{2k} < \frac{1}{k} \)
  \( \Rightarrow \) absolute error \( < 1 \) if \( \text{OPT} \leq k \)
- so if we find solution with value \( \leq k \)
  then \( \text{OPT} \leq (1 + \frac{1}{2k}) \cdot k \leq k + \frac{1}{2} \)
  integral \( \Rightarrow \text{OPT} \leq k \Rightarrow \text{YES.} \)
- else \( \text{OPT} > k \)

Also: \( = \), \( \leq \), \( \geq \) decision problems are equivalent w.r.t. FPT

\(--\) Can use this relation to prove EPTASs don't exist in some cases