NP-complete problems seem to only have exponential-time algs.
but how exponential? e.g. we saw pseudo-pol. time for knapsack-
poly. time if profits have log n bits.
more generally, consider other parameters. Small parameter $\Rightarrow$ poly. time.

Vertex Cover: Given graph $G = (V,E)$ and $k \in \mathbb{N}$, is there a vertex cover of size $\leq k$?

what if we use $k$ itself as a parameter?
$k = 1, 2, 3$ Try all possible $k$-sets $(\binom{n}{k})$ which is $O(n^k)$
checking a subset takes $O(n \cdot k)$ so total time is $O(k \cdot n^{k+1})$
this is poly. time for constant $k$.

Same idea works for clique or ind. set
but not for graph colouring - given a graph, is it 3-colourable?
is NP-complete.

$O(n^k)$ is still pretty bad.
We want $O(f(k) \cdot n^c)$ for some fin. $f(k)$ ind. of $n$
and some constant $< \text{ind. of } k$.
or even $O(f(k) + n^c)$

We can achieve this for Vertex Cover (but not for Ind. Set unless $P = NP$).

For each edge $(u,v)$ we need $u$ or $v$ in the Vertex Cover.
We can do exhaustive search, branching on this choice.

At each tree node, we pick an uncovered edge.
$|C| \leq k \Rightarrow$ stop at depth $k$. 

Alg. VC2 \((G, k)\) - return True or False

if \(E = \emptyset\) return True
if \(k = 0\) return False
pick \(e = (u, v) \in E\)
return VC2 \((G - u, k - 1)\) or VC \((G - v, k - 1)\)

remove \(u\) and incident edges

Time \(O(2^k \cdot n)\) \(\approx\) work for each node

EX: find the vertex cover for \(G\).

We can improve to \(O(f(k) + m+n)\)

technique called "kernelization".

Claim If \(G\) has a vertex \(v\) with \(\deg(v) > k\) then \(v\) must be in \(C\).

\((C\) vertex cover of size \(k)\)

Pf \(v \in \not C > k\) if \(v \not C\), we need all \(\geq k\) neighbours.

Alg. VC3 \((G, k)\) - return True/False

\(C' \leftarrow \) all vertices of \(\deg > k\)

\(k' \leftarrow k - |C'|\)

\(G' \leftarrow G \setminus (C' \text{ and all incident edges})\)

and remove isolated vertices

if \(G'\) has \(> 2k^2\) vertices
return False
else return VC2 \((G', k')\)

\(\text{Note:}\)

\(G'\) is not very big:
- \(\max\) degree in \(G'\) is \(\leq k\)
- if \(G'\) has VC of size \(\leq k\)
  then \(G'\) has \(\leq k^2\) edges
- \(G'\) has \(\leq 2k^2\) vertices.
the actual vertex cover is \( C' \cup \text{vertex cover of } G', k' \)

idea due to Prof. J. Buss '93

Analysis: call to VC2 takes \( O(2^k \cdot (\text{# vertices of } G')) \)

finding \( C' \) takes \( O(m+n) \)

total run time \( O(2^k \cdot k^2 + m + n) \)

This 3rd alg. is even practical sometimes (much better than other two)

Comparison

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Time</th>
<th>( n = 10^4 )</th>
<th>( k = 10 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>brute force</td>
<td>( k \cdot n^{k+1} )</td>
<td>( 10^{95} )</td>
<td>( \times )</td>
</tr>
<tr>
<td>VC2</td>
<td>( 2^k n )</td>
<td>( 10^9 )</td>
<td>( ? ) possible</td>
</tr>
<tr>
<td>VC3</td>
<td>( 2^k k^2 + n + m )</td>
<td>( 2 \cdot 10^5 + m )</td>
<td></td>
</tr>
</tbody>
</table>

**Defn** A problem is **fixed parameter tractable** in parameter \( k \)
if it has an alg. with run time \( O(f(k) \cdot n^c) \) \( n \) = input size
where \( f(k) \) is a fn of \( k \) (ind. of \( n \))
and \( c \) is a constant (ind. of \( k \))

Examples of parameters
- value of opt., e.g. Ind. set of size \( k \)
- max degree of graph
- dimension for geometric problems
- genus of graph

\( \text{genus 0 = planar} \)
\( 1 = \text{on torus} \)
Some examples of FPT alg's:
- Vertex cover of size \( k \)
- Simple path of length \( k \)
- Finding \( k \) disjoint triangles in graph
- Drawing graph in plane with \( k \) edge crossings
- Finding disjoint paths connecting \( k \) pairs of nodes

Hardness - some parameterized problems have no FPT alg., unless \( P = NP \), e.g. Ind. Set.

The kernelization method generalizes

Thm. If a problem with parameter \( k \) is FPT i.e. there's an alg. with run time \( O(f(k) \cdot n^c) \)
then there exists an alg. with run time \( O(f'(k) + n^{c'}) \)

But proof is not constructive. (It's not hard either)
$k$-path: Given graph $G$, $k \in \mathbb{N}$, vertices $s,t$, find a simple $s-t$ path with exactly $k$ internal vertices. 

A shortest $s-t$ path will be simple.

But if $k$ is larger, this problem is NP-hard — it generalizes Hamiltonian Path.

**Ex.**

A randomized FPT algorithm

Colour vertices $V \setminus \{s,t\}$ randomly with $k$ colours

Test for colour-ful $s-t$ path — each colour appears exactly once.

We’ll see how to test later.

Error analysis:

no simple $s-t$ path $\Rightarrow$ no colour-ful $s-t$ path $\Rightarrow$ alg. returns NO (for any colouring)

3 simple $s-t$ path $P$

\[
\text{Prob } \exists P \text{ is colour-ful} = \frac{k!}{k^k} \geq \frac{(\frac{k}{e})^k}{k^k} = \frac{1}{e^k}
\]

(order colours on path, Stirling $\left(\frac{k}{e}\right)^k \sqrt{2\pi k}$, colour all vertices on path)

So Alg. outputs YES (i.e. is correct) with prob. $\geq \frac{1}{e^k}$

If prob. success $\geq p$ then prob. failure after $\frac{1}{p}$ repetitions is

\[
(1-p)^{\frac{1}{p}} < (e^{-p})^{\frac{1}{p}} = \frac{1}{e}
\]

\[
1-p + \frac{p^2}{2} \ldots \text{ Taylor expansion}
\]

So in our case, repeating $e^k$ times gives prob. error $\leq \frac{1}{e}$.
Finding a colourful $s$-$t$ path.

The colours on such a path are a permutation of $1 \cdots k$.

Try all $k!$ permutations.

The problem is easy for fixed permutation.

Run time $O(k! \cdot m)$, $m = \# edges$

Fact: can be improved to $O(2^k \cdot m)$

So we have a FPT alg. with small prob. of error, $\leq \frac{1}{2}$

and with run time $O(e^k 2^k \cdot m)$

Fact: can be de-randomized (using $k$-perfect hash functions)