Recall: Fixed Parameter Tractable Algorithm (FPT) has run time \( O(f(k)n^c) \)
- \( n \) - input size
- \( k \) - a parameter of the input
- \( f(k) \) - function of \( k \), independent of \( n \)
- \( c \) - constant, independent of \( k \).

**Ex. Ind. Set** - Given a graph \( G \), does it have an ind. set \( \geq k \)?

\( \mathcal{O} \) Ind. set of size 3

Brute force \( O\left(\binom{n^k}{k} \cdot (n+m)\right) \)
- \# \( k \) subsets of \( n \) vertices

\underline{NOT} Fixed parameter tractable.

**FACT:** No one has found a FPT alg. for Ind. Set.
(Note: would such an FPT alg. imply \( P=NP \)? No one knows.)

Ex. Vertex Cover parameter = size of min v.c.
FPT alg. \( O(2^k \cdot n) \) - using branching alg.
\( O(2^k \cdot k^2 + m+n) \) - kernelization.

Today: Ind. Set and FPT algorithm with a parameter that measures how "tree-like" is our graph.
Ind. Set on a tree

? start at leaves?

- with weights on vertices, \( w(v) \)

Dynamic Programming

Subproblems

\[ IS(v) = \max \text{ weight of ind. set in subtree rooted at } v \]

\[ IS^0(v) = \max \text{ weight of ind. set in subtree rooted at } v \text{ that DOES NOT include } v \]

Alg.

Initialize: for every leaf \( v \)

\[ IS^0(v) = \begin{cases} 0 \\ w(v) \end{cases} \]

\[ IS(v) = \begin{cases} 0 \\ w(v) \end{cases} \]

For all nodes \( v \) in leaf-root order

\( v \) has children \( u_1 \ldots u_t \)

\[ IS^0(v) = \sum_{i=1}^{k} IS(u_i) \]

\[ IS(v) = \max \left( \sum_{i=1}^{k} IS(u_i), IS^0(v) \right) \]

Return \( IS(\text{root}) \)
Can use same idea for graphs that are "close to" trees.

Series-parallel graphs (SP graphs)
defined recursively - graph + two "terminal" nodes $s, t$.

- $s \rightarrow t$ is SP

```
\begin{tikzpicture}[node distance=2cm, thick, main node/.style={circle,draw}]
  \node[main node] (s) {$s$};
  \node[main node] (t) [right of=s] {$t$};
  \draw (s) node [right] {\text{parallel}} node [left] {\text{series}}
   (s) -- (t);
\end{tikzpicture}
```

Series

Parallel

Ind. Set in series-parallel graph.

We can do dp programming based on
- max. ind. set including $s$ and $t$
- $s$, not $t$
- not $s$, $t$
- not $s$, not $t$. 
We can model as a tree

For each "series" step, include middle terminal in parent node of tree.

Properties

1. If \( e = (u, v) \) is an edge of \( G \) then \( u \) and \( v \) appear together in a tree node.
2. Every vertex \( v \) of \( G \) corresponds to a subtree of \( T \).
Generalization: **tree-width**

represent graph $G$ as tree $T$

- for every vertex $v$ of $G$, the bags containing it form a subtree
- for every edge $e=(u,v)$ of $G$, there is a bag containing $u$ and $v$.

width of decomposition = size of largest bag - 1

Tree-width of a graph = min. width of any tree decomposition ≤ $n-1$ (single bag).

Graphs of tree-width 1 = forests (subgraph of tree)

Graphs of tree-width 2 = subgraphs of $SP$ graphs.
Thm. Max weight Ind. Set in graph of tree-width $k$ can be found in time $O(2^k \cdot n)$

Idea: use dynamic programming. For each bag $B$ (size $\leq k+1$) we find for each subset $A \subseteq B$ (there are $O(2^k)$ of them) the max weight ind. set in subtree rooted at $B$ that includes $A$ and excludes $B-A$.

How do we find tree-width of a graph? / FPT

Thm. There is an alg. w/ running time $O(2^{O(k^3)} \cdot n)$.

There are more efficient algns if we're ok with approx. tree-width.

Many problems are FPT in tree-width
- 3 colouring
- min. colouring
- Hamiltonian cycle

Hardness wrt FPT algns:
- must use reductions that preserve fixed parameter tractability.

Can prove results like:

FPT alg. for problem $X \Rightarrow$ FPT alg. for problem $Y$.

and W[1]-class of equivalently hard problems.

Compare w/ approx. where PCT proofs Tam gave:

PTAS for problem $X \Rightarrow P = NP$.

No such strong results for FPT.