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 $\geq k$?

- Ind. set of size 3
- 

Brute force $O\left(\binom{n^k}{k}(n+m)\right)$

# k subsets of n vertices

\underline{NOT} Fixed parameter tractable.

\underline{FACT:} No one has found a FPT alg. for Ind. Set (for this parameter).

(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.

\underline{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?
  - can't just take levels
- with weights on vertices, w(v)

Dynamic Programming

Subproblems

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

\[ IS^0(v) = \] that DOES NOT include \( v \)

Alg.

Initialize: for every leaf \( v \)

\[ IS^0(v) = 0 \]
\[ IS(v) = w(v) \]

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

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

\[ IS^0(v) = \sum_{i=1}^{t} IS(u_i) \]
\[ IS(v) = \max\left( w(v) + \sum_{i=1}^{t} IS^0(u_i) \right) \]

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{align*}
S_1 \rightarrow \bigcirc \rightarrow t_1 \\
S_2 \rightarrow \bigcirc \rightarrow t_2 \\
\end{align*}
```

**Parallel**

```
\begin{align*}
\text{S1} \rightarrow 1 \rightarrow 1 \\
\text{S2} \rightarrow 2 \rightarrow 2 \\
\end{align*}
```

**Series**

```
\begin{align*}
1 \rightarrow 4 \rightarrow 1 \\
4 \rightarrow 4 \rightarrow 4 \\
\end{align*}
```

**FACTS**
- Series-parallel graphs can be recognized in linear time
- Equivalently, they are graphs without $K_4$ (as a minor)
  - [Compare: planar graphs are those without $K_5, K_{3,3}$ (as minors)]

**Ind. Set in series-parallel graph**
- We can do dynamic programming based on
  - Max. ind. set including $s$ and $t$
    - $\ldots$
      - $s$, not $t$
    - $\ldots$
      - $\not s$, $t$
    - $\not s$, $\not t$
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 r of G corresponds to a subtree of T.
Generalization: **tree-width** - Robertson & Seymour represent graph $G$ as tree $T$

**conditions:**
- 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 $\leq n-1$ (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? / NP-hard

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

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

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

Courcelle's Thm gives some general conditions - any problem expressible in monadic second order logic is FPT

Hardness wrt FPT alg's:
- 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 equivalent problems.

Compare w/ approx. where PCT proofs Tam gave:
PTAS for problem $X$ $\Rightarrow$ $P = NP$.

No such strong results for FPT.