Advanced Algorithms

Winter term 2019/20

Lecture 11. Alternative Parameterization: Tree Decomposition

Source: PA §7.2, 7.3.1

(slides by Thomas van Dijk & Alexander Wolff)

Steven Chaplick

Lehrstuhl für Informatik I
** Independent Set **

**INDEPENDENT SET**

Given: graph $G$, weight function $\omega : V \to \mathbb{N}$

Question: What is the maximum weight of a set $S \subseteq V$ where no pair in $S$ forms an edge in $G$?

**Thm:** Independent Set is NP-complete.

**Thm:** Independent Set can be solved in linear time on trees.
Independent Sets in Trees

Choose an arbitrary root \( w \).

Let \( T(v) := \) subtree rooted at \( v \)

Let \( A(v) := \) maximum weight of an independent set \( S \) in \( T(v) \)

Let \( B(v) := \) maximum weight of an independent set \( S \) in \( T(v) \) where \( v \not\in S \)

When \( v \) is a leaf: \( A(v) = \omega(v) \) and \( B(v) = 0 \)

When \( v \) has children \( x_1, \ldots, x_r \):

\[
A(v) = \max \{ \sum_{i=1}^{r} A(x_i), \ \omega(v) + \sum_{i=1}^{r} B(x_i) \}
\]

\[
B(v) = \sum_{i=1}^{r} A(x_i)
\]

Algo: Compute \( A(\cdot) \) and \( B(\cdot) \) bottom-up

\( A(w) = \) solution
**s, t-series parallel graphs**

**Def.:** A graph $G = (V, E)$ is *2-terminal* when it contains two special vertices $s$ and $t$.

**Def.:** A 2-terminal graph $G$ is *series parallel* when:

- $G$ is a single edge $(s, t)$
- $G$ is a *series composition* of two series parallel graphs
- $G$ is a *parallel composition* of two series parallel graphs

**recursive definition:**

series parallel graphs have a natural tree-structure
Let $i$ be a node in an SP-tree. $G(i) :=$ graph represented by the subtree rooted at $i$.
Independent Set on SP-trees

Dynamic program on SP-tree indexed by $G(i)$

$AA(i) :=$ maximum weight independent set $S$ in $G(i)$ where $s_i \in S$ and $t_i \in S$

$BA(i) :=$ maximum weight independent set $S$ in $G(i)$ where $s_i \notin S$ and $t_i \in S$

$AB(i)$ and $BB(i)$ def. similarly

other cases omitted... (easy exercise)

$O(1)$ time per SP-node

**Thm:** **Independent Set** on series parallel graphs with a given SP-tree can be solved in $O(n)$ time.
Generalization?

Many ways to generalize the concept of having a “tree structure”

Ex.: $k$-terminal graph $G = (V, E, T)$, $|T| = k$

Example Operation: “gluing”
Example: Tree Decomposition

Graph \( G = (V, E) \):

Tree Decomposition:
Def. A tree decomposition of a graph $G = (V, E)$ is:

- a tuple $D = (X, T)$
- $T = (P, F)$ is a tree
- $X = \{X_p \mid p \in P\}$ is a set family of subsets of $V$ (one for each node in $P$)
- $\bigcup_{p \in P} X_p = V$
- $\forall \{u, v\} \in E \exists p \in P$ where $u, v \in X_p$
- $\forall v \in V : \{p \in P \mid v \in X_p\}$ is connected in $T$
Treewidth (formal)

- a tuple $D = (X, T)$
- $T = (P, F)$ is a tree

**Def.** Width (tree decomposition): $\max_{p \in P} |X_p| - 1$, i.e., cardinality of the largest bag $-1$

**Def.** Treewidth $\text{tw}(G)$ is the minimum width of a tree decomposition of $G$

**Obs.** $\text{tw}(G) < n$

**Question:** Which graphs have treewidth 0? $E = \emptyset$

**Exercise:** Trees have treewidth 1

**Exercise:** Series parallel graphs have treewidth 2

**Thm:** There is a tree decomposition of width $\text{tw}(G)$ where $|P|$ is polynomial in $n$, i.e., the tree has polynomial size in $n$
Parameterized Problems

Given: Instance of size $n$ and parameter $k$

**Def.** Problem is FPT when solvable in $O(f(k) \cdot \text{poly}(n))$ time.

$O(f(tw(G)) \cdot \text{poly}(n))$ time.

**Ex.:**

- **$k$-Vertex Cover**  
  FPT

- **$k$-Independent Set**  

- **$k$-Dominating Set**  

- **$k$-Coloring**  
  NP-comp. $k \geq 3$

- **Independent Set (treewidth)**  
  FPT

- **List Coloring (treewidth)**  

- **Channel Assignment (treewidth)**  
  NP-comp. $k \geq 3$

See PA §13.3
Computing Treewidth

**Treewidth**

**Given:** Graph $G = (V, E)$, number $k$

**Question:** $\text{tw}(G) \leq k$?

**Thm:** *Treewidth* is NP-complete

$k$-**Treewidth**

**Given:** graph $G = (V, E)$

**Parameter:** number $k$

**Question:** $\text{tw}(G) \leq k$?

**Thm:** $k$-*Treewidth* is FPT

See PA §7.6.

How can we make “fixed-treewidth-tractable” algorithms?
item #1: nice tree decompositions

In a *nice* tree decomp., one bag is marked as the root and there are only 4 types of bags:

- **Leaf:** the bag is a leaf and contains only one vertex.

- **Introduce:** The bag has exactly one child and contains the child’s vertices and exactly one new vertex.

- **Forget:** The bag has exactly one child and contains one vertex fewer than the child.

- **Join:** The bag has exactly two children and these three nodes have exactly the same vertices.
item #1: nice tree decompositions

Introduce b

Forget f

Join

Introduce f

Introduce a

Forget d

Forget e

Introduce e

Introduce d
item #2: DP on nice Tree Decomp.

**Thm:** $k$-TreeWidth is FPT

**Thm:** Tree decompositions $\rightarrow$ nice in polynomial time.

**Cor:** For FPT-Algorithms it suffices to use nice tree decomp.

**Strategy:** Build a recurrence for each type of bag, and use dynamic programming.
Indep.Set on Nice Tree Decomp.

Let $G(i) :=$ Graph induced by the vertices in the subtree at $i$

For bag $i$ and $S \subseteq X_i$, let:
$R(i, S) :=$ maximum weight of an indep. set $I$ in $G(i)$ with $I \cap X_i = S$

Algo.: Compute $R(i, S)$ for all $i$ and corresponding $S$

Runtime: ?

Thm: The independent set problem is FPT parameterized by treewidth.