Algorithmic Graph Theory: How hard is your combinatorial optimization problem?

Arie M.C.A. Koster

Lecture 7

Clemson, June 12, 2017

CLEMSON UNIVERSITY
Lehrstuhl II für Mathematik
RWTH AACHEN UNIVERSITY

Lower Bounds:

$$tw(G) \geq \delta C(G) \geq \delta D(G) \geq \omega(G) - 1 \geq \delta(G)$$

Lower Bounds:

$$tw(G) \geq \delta C(G) \geq \delta D(G) \geq \omega(G) - 1 \geq \delta(G)$$

Definition

Let $G = (V, E)$ be a graph and k integer. The $(k+1)$-neighbor improved graph $G' = (V, E')$ can be constructed as follows: take G and add edge uv as long as two non-adjacent vertices u, v exist, having $k+1$ joint neighbors.

Theorem

Let (X, T) be a tree decomposition for G wtih width at most k. Then, (X, T) is also a tree decomposition for the $(k+1)$-neighbor improved graph G' with width k and vice versa.

Theorem

Let (X, T) be a tree decomposition for G wtih width at most k. Then, (X, T) is also a tree decomposition for the $(k+1)$-neighbor improved graph G' with width k and vice versa.

Corollary

Let $G = (V, E)$ be a graph and ℓ a lower bound on $tw(G)$. Further, let G' be the $(\ell+1)$-neighbor improved graph and ℓ' be a further lower bound on $tw(G')$. If $\ell' > \ell$, then it holds that $tw(G) \geq \ell + 1$.

- Many NP-hard problems are easy on trees!

- Many NP-hard problems are easy on trees!
- Many NP-hard problems remain easy on series-parallel graphs!

- Many NP-hard problems are easy on trees!
- Many NP-hard problems remain easy on series-parallel graphs!
- Many NP-hard problems are still easy if the graph has bounded treewidth!

Max. weighted independent set

Given $G = (V, E)$ with vertex weights $c(v) \in \mathbf{Z}^+$, a **max. weighted independent set** is a subset of the vertices $S \subseteq V$ such that they are pairwise non-adjacent and the sum of the weights $c(S) = \sum_{v \in S} c(v)$ is maximised.

Max. weighted independent set

Given $G = (V, E)$ with vertex weights $c(v) \in \mathbf{Z}^+$, a **max. weighted independent set** is a subset of the vertices $S \subseteq V$ such that they are pairwise non-adjacent and the sum of the weights $c(S) = \sum_{v \in S} c(v)$ is maximised.

If G is a tree T,

- we root it at an arbitrary vertex $r \in V$ and
- let $T(v)$ denote the subtree with v as root.

Max. weighted independent set

Given $G = (V, E)$ with vertex weights $c(v) \in \mathbf{Z}^+$, a **max. weighted independent set** is a subset of the vertices $S \subseteq V$ such that they are pairwise non-adjacent and the sum of the weights $c(S) = \sum_{v \in S} c(v)$ is maximised.

If G is a tree T,

- we root it at an arbitrary vertex $r \in V$ and
- let $T(v)$ denote the subtree with v as root.

Define

$$
\begin{aligned}
A(v) &= \text{the max. weight of an independent set in } T(v) \\
B(v) &= \text{the max. weight of an independent set in } T(v) \text{ not containing } v
\end{aligned}
$$

Max. weighted independent set

Given $G = (V, E)$ with vertex weights $c(v) \in \mathbf{Z}^+$, a **max. weighted independent set** is a subset of the vertices $S \subseteq V$ such that they are pairwise non-adjacent and the sum of the weights $c(S) = \sum_{v \in S} c(v)$ is maximised.

If G is a tree T,

- we root it at an arbitrary vertex $r \in V$ and
- let $T(v)$ denote the subtree with v as root.

Define

$$A(v) = \text{the max. weight of an independent set in } T(v)$$
$$B(v) = \text{the max. weight of an independent set in } T(v) \text{ not containing } v$$

$A(r)$ provides the max. weight independent set

Compute $A(v)$ and $B(v)$ from the leafs to the root:

Compute $A(v)$ and $B(v)$ from the leafs to the root:

v is a leaf

$$\begin{aligned} A(v) &= c(v) \\ B(v) &= 0 \end{aligned}$$

Compute $A(v)$ and $B(v)$ from the leafs to the root:

v is a leaf

$$
\begin{aligned}
A(v) &= c(v) \\
B(v) &= 0
\end{aligned}
$$

v is non-leaf with children $x_1, \ldots, x_r$

$$
\begin{aligned}
A(v) &= \\
B(v) &= A(x_1) + \ldots + A(x_r)
\end{aligned}
$$

Compute $A(v)$ and $B(v)$ from the leafs to the root:

v is a leaf

$$\begin{aligned} A(v) &= c(v) \\ B(v) &= 0 \end{aligned}$$

v is non-leaf with children $x_1, \ldots, x_r$

$$\begin{aligned} A(v) &= \max\{c(v) + B(x_1) + \ldots + B(x_r), A(x_1) + \ldots + A(x_r)\} \\ B(v) &= A(x_1) + \ldots + A(x_r) \end{aligned}$$

Compute $A(v)$ and $B(v)$ from the leafs to the root:

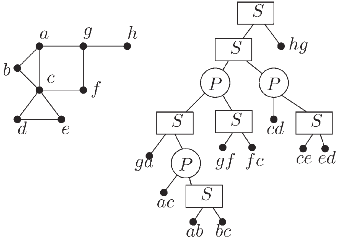
v is a leaf

$$\begin{aligned} A(v) &= c(v) \\ B(v) &= 0 \end{aligned}$$

v is non-leaf with children $x_1, \ldots, x_r$

$$\begin{aligned} A(v) &= \max\{c(v) + B(x_1) + \ldots + B(x_r), A(x_1) + \ldots + A(x_r)\} \\ B(v) &= A(x_1) + \ldots + A(x_r) \end{aligned}$$

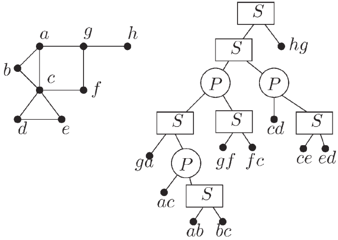
Running time: $O(n)$



If G is a series-parallel graph with SP-tree $T(G)$,

- the leafs of the SP-tree $T(G)$ correspond to the edges $e \in E$
- internal nodes labelled S or P for series and parallel composition

SP-trees



If G is a series-parallel graph with SP-tree $T(G)$,

- the leafs of the SP-tree $T(G)$ correspond to the edges $e \in E$
- internal nodes labelled S or P for series and parallel composition

$AA(i)$:	maximum weight of independent set containing both s and t
$AB(i)$:	maximum weight of independent set containing s but not t
$BA(i)$:	maximum weight of independent set containing t but not s
$BB(i)$:	maximum weight of independent set containing neither s nor t

Max. Weighted Ind. Set

v is a leaf

$AA(i) := -\infty$, $AB(i) := c(s)$, $BA(i) := c(t)$, and $BB(i) := 0$

Max. Weighted Ind. Set

v is a leaf

$AA(i) := -\infty$, $AB(i) := c(s)$, $BA(i) := c(t)$, and $BB(i) := 0$

Internal node i with children i_1 and i_2

If i is an S node (with s' the terminal between i_1 and i_2):

$$\begin{aligned}
AA(i) &:= \max\{AA(i_1) + AA(i_2) - c(s'), AB(i_1) + BA(i_2)\}\,, \\
AB(i) &:= \max\{AA(i_1) + AB(i_2) - c(s'), AB(i_1) + BB(i_2)\}\,, \\
BA(i) &:= \max\{BA(i_1) + AA(i_2) - c(s'), BB(i_1) + BA(i_2)\}\,, \text{ and} \\
BB(i) &:= \max\{BA(i_1) + AB(i_2) - c(s'), BB(i_1) + BB(i_2)\}\,.
\end{aligned}$$

Max. Weighted Ind. Set

Internal node i with children i_1 and i_2

If i is an P node:

$$
\begin{aligned}
AA(i) &:= AA(i_1) + AA(i_2) - c(s) - c(t)\,, \\
AB(i) &:= AB(i_1) + AB(i_2) - c(s)\,, \\
BA(i) &:= BA(i_1) + BA(i_2) - c(t)\,, \text{ and} \\
BB(i) &:= BB(i_1) + BB(i_2)\,.
\end{aligned}
$$

Internal node i with children i_1 and i_2

If i is an P node:

$$\begin{aligned} AA(i) &:= AA(i_1) + AA(i_2) - c(s) - c(t)\,, \\ AB(i) &:= AB(i_1) + AB(i_2) - c(s)\,, \\ BA(i) &:= BA(i_1) + BA(i_2) - c(t)\,, \text{ and} \\ BB(i) &:= BB(i_1) + BB(i_2)\,. \end{aligned}$$

Running time: $O(m)$

Nice Tree Decompositions

In a nice tree decomposition is rooted, and each node $i \in I$ is of one of the four following types:

- **Leaf**: Node i is a leaf of T, and $|X_i| = 1$.
- **Join**: Node i has exactly two children, say j_1 and j_2 and $X_i = X_{j_1} = X_{j_2}$.
- **Introduce**: Node i has exactly one child, say j, and there is a vertex $v \in V$ with $X_i = X_j \cup \{v\}$.
- **Forget**: Node i has exactly one child, say j, and there is a vertex $v \in V$ with $X_j = X_i \cup \{v\}$.

Nice Tree Decompositions

In a nice tree decomposition is rooted, and each node $i \in I$ is of one of the four following types:

- **Leaf**: Node i is a leaf of T, and $|X_i| = 1$.
- **Join**: Node i has exactly two children, say j_1 and j_2 and $X_i = X_{j_1} = X_{j_2}$.
- **Introduce**: Node i has exactly one child, say j, and there is a vertex $v \in V$ with $X_i = X_j \cup \{v\}$.
- **Forget**: Node i has exactly one child, say j, and there is a vertex $v \in V$ with $X_j = X_i \cup \{v\}$.

Lemma

If G has treewidth at most k, then G also has a nice tree decomposition of width at most k which has $O(n)$ tree nodes.

Max. Weighted Ind. Set

Nice tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$
For $i \in I$, let $G_i = (V_i, E_i)$ with

- V_i is the union of all bags X_j, with $j = i$ or j a descendant of i in T, and
- $E_i = E \cap (V_i \times V_i)$ is the set of all edges in E which have both endpoints in V_i

Max. Weighted Ind. Set

Nice tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$
For $i \in I$, let $G_i = (V_i, E_i)$ with

- V_i is the union of all bags X_j, with $j = i$ or j a descendant of i in T, and
- $E_i = E \cap (V_i \times V_i)$ is the set of all edges in E which have both endpoints in V_i

For each node $i \in I$, we compute a table C_i:

- $C_i(S)$, for $S \subseteq X_i$, equals the maximum weight of an independent set $W \subseteq V_i$ in G_i such that $X_i \cap W = S$

Max. Weighted Ind. Set

Nice tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$
For $i \in I$, let $G_i = (V_i, E_i)$ with

- V_i is the union of all bags X_j, with $j = i$ or j a descendant of i in T, and
- $E_i = E \cap (V_i \times V_i)$ is the set of all edges in E which have both endpoints in V_i

For each node $i \in I$, we compute a table C_i:

- $C_i(S)$, for $S \subseteq X_i$, equals the maximum weight of an independent set $W \subseteq V_i$ in G_i such that $X_i \cap W = S$

Number of entries to compute for node $i \in I$: $2^{|X_i|}$

Leaf node $i \in I$:

Say $X_i = \{v\}$.

$$\begin{aligned} C_i(\emptyset) &= 0 \\ C_i(\{v\}) &= c(v) \end{aligned}$$

Leaf node $i \in I$:

Say $X_i = \{v\}$.

$$\begin{aligned} C_i(\emptyset) &= 0 \\ C_i(\{v\}) &= c(v) \end{aligned}$$

Introduce node i with child j

Suppose $X_i = X_j \cup \{v\}$. Let $S \subseteq X_j$.

1. $C_i(S) = C_j(S)$.
2. If there is a vertex $w \in S$ with $\{v, w\} \in E$, then $C_i(S \cup \{v\}) = -\infty$.
3. If for all $w \in S$, $\{v, w\} \notin E$, then $C_i(S \cup \{v\}) = C_j(S) + c(v)$.

Forget node i with child j

Suppose $v \in X_j \setminus X_i$ (unique). Let $S \subseteq X_i$.

- $C_i(S) = \max\{C_j(S), C_j(S \cup \{v\})\}$.

Max. Weighted Ind. Set

Forget node i with child j

Suppose $v \in X_j \setminus X_i$ (unique). Let $S \subseteq X_i$.

- $C_i(S) = \max\{C_j(S), C_j(S \cup \{v\})\}$.

Join node i with children j_1 and j_2

G_i can be seen as a kind of union of G_{j_1} and G_{j_2}.

Forget node i with child j

Suppose $v \in X_j \setminus X_i$ (unique). Let $S \subseteq X_i$.

- $C_i(S) = \max\{C_j(S), C_j(S \cup \{v\})\}$.

Join node i with children j_1 and j_2

G_i can be seen as a kind of union of G_{j_1} and G_{j_2}.

- If $v \in V_{j_1}$, $w \in V_{j_2}$, and $v, w \notin X_i$, then $\{v, w\} \notin E$.

Forget node i with child j

Suppose $v \in X_j \setminus X_i$ (unique). Let $S \subseteq X_i$.

- $C_i(S) = \max\{C_j(S), C_j(S \cup \{v\})\}$.

Join node i with children j_1 and j_2

G_i can be seen as a kind of union of G_{j_1} and G_{j_2}.

- If $v \in V_{j_1}$, $w \in V_{j_2}$, and $v, w \notin X_i$, then $\{v, w\} \notin E$.
- Let $S \subseteq X_i$. $C_i(S) = C_{j_1}(S) + C_{j_2}(S) - c(S)$.

Forget node i with child j

Suppose $v \in X_j \setminus X_i$ (unique). Let $S \subseteq X_i$.

- $C_i(S) = \max\{C_j(S), C_j(S \cup \{v\})\}$.

Join node i with children j_1 and j_2

G_i can be seen as a kind of union of G_{j_1} and G_{j_2}.

- If $v \in V_{j_1}$, $w \in V_{j_2}$, and $v, w \notin X_i$, then $\{v, w\} \notin E$.
- Let $S \subseteq X_i$. $C_i(S) = C_{j_1}(S) + C_{j_2}(S) - c(S)$.

Optimal solution: $\max_{S \subseteq X_r} C_r(S)$

Max. Weighted Ind. Set

Forget node i with child j

Suppose $v \in X_j \setminus X_i$ (unique). Let $S \subseteq X_i$.

- $C_i(S) = \max\{C_j(S), C_j(S \cup \{v\})\}$.

Join node i with children j_1 and j_2

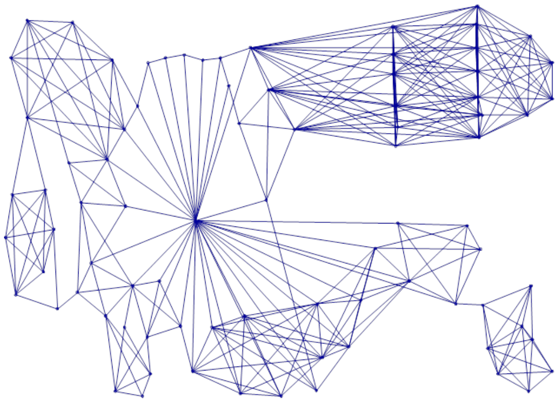
G_i can be seen as a kind of union of G_{j_1} and G_{j_2}.

- If $v \in V_{j_1}$, $w \in V_{j_2}$, and $v, w \notin X_i$, then $\{v, w\} \notin E$.
- Let $S \subseteq X_i$. $C_i(S) = C_{j_1}(S) + C_{j_2}(S) - c(S)$.

Optimal solution: $\max_{S \subseteq X_r} C_r(S)$

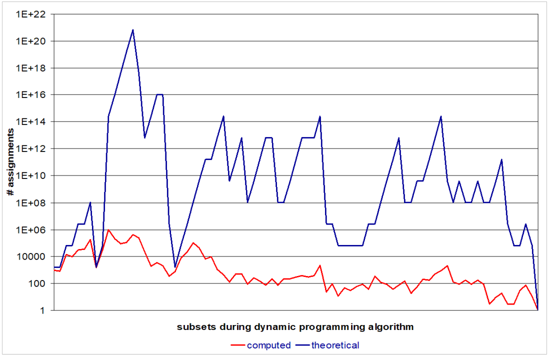
If $tw(G) = k$, the running time is $O(2^k \cdot n)$

Frequency Assignment



The (Minimum Interfernce) Frequency Assignment Problem asks for a coloring of the vertices of a graph $G = (V, E)$ such that

- each vertex $v \in V$ is colored with a color $f(v)$ from its domain $F(v)$,
- the sum of assignment cost $\sum_{v \in V} c_v(f(v))$ and interference cost $\sum_{vw \in E} c_{vw}(f(v), f(w))$ is minimized.



Theoretical number of assignments vs. actual number

Actual number of assignments achieved by

- graph reduction
- upper bounding techniques, dominance techniques

Algorithmic Graph Theory: How hard is your combinatorial optimization problem?

Arie M.C.A. Koster

Lecture 7

Clemson, June 12, 2017

Lehrstuhl II für Mathematik