Tree-width i robber-and-cops

Matko Botinčan[1]

[1]PMF - Matematički odjel

Seminar za teorijsko računarstvo, 27. 03. 2006.

Sadržaj

Sadržaj

Sadržaj

Sadržaj

Pregled

1. Uvod

2. Tree-width
   - Osnovne definicije i svojstva
   - Igra robber-and-cops
   - Rezultati o složenosti

3. Problem verifikacije modela za strukture slične stablima
   - MSO
   - FO

4. Vezane teme

(Jedna od mogućih) motivacija

Mnogi NP-potpuni problemi definirani na klasi (konačnih) grafova postaju jednostavni na klasi (konačnih) stabala.

(Jedna od mogućih) motivacija

Mnogi NP-potpuni problemi definirani na klasi (konačnih) grafova postaju jednostavni na klasi (konačnih) stabala.

Primjeri

Neka je $\mathcal{T}$ konačno stablo.

(Jedna od mogućih) motivacija

Mnogi NP-potpuni problemi definirani na klasi (konačnih) grafova postaju jednostavni na klasi (konačnih) stabala.

Primjeri

Neka je $\mathcal{T}$ konačno stablo.

1. *Ima li $\mathcal{T}$ Hamiltonov ciklus?*

(Jedna od mogućih) motivacija

Mnogi NP-potpuni problemi definirani na klasi (konačnih) grafova postaju jednostavni na klasi (konačnih) stabala.

Primjeri

Neka je $\mathcal{T}$ konačno stablo.

1. *Ima li $\mathcal{T}$ Hamiltonov ciklus?*
   $\rightarrow \mathcal{O}(1)$

(Jedna od mogućih) motivacija

Mnogi NP-potpuni problemi definirani na klasi (konačnih) grafova postaju jednostavni na klasi (konačnih) stabala.

Primjeri

Neka je $\mathcal{T}$ konačno stablo.

1. *Ima li $\mathcal{T}$ Hamiltonov ciklus?*
   $\rightarrow \mathcal{O}(1)$
2. *Da li je $\mathcal{T}$ 3-obojivo?*

(Jedna od mogućih) motivacija

Mnogi NP-potpuni problemi definirani na klasi (konačnih) grafova postaju jednostavni na klasi (konačnih) stabala.

Primjeri

Neka je $\mathcal{T}$ konačno stablo.

1. *Ima li $\mathcal{T}$ Hamiltonov ciklus?*
   $\rightarrow \mathcal{O}(1)$
2. *Da li je $\mathcal{T}$ 3-obojivo?*
   $\rightarrow \mathcal{O}(1)$

(Jedna od mogućih) motivacija

Mnogi NP-potpuni problemi definirani na klasi (konačnih) grafova postaju jednostavni na klasi (konačnih) stabala.

Primjeri

Neka je $\mathcal{T}$ konačno stablo.

1. *Ima li $\mathcal{T}$ Hamiltonov ciklus?*
   $\rightarrow \mathcal{O}(1)$
2. *Da li je $\mathcal{T}$ 3-obojivo?*
   $\rightarrow \mathcal{O}(1)$
3. *Ima li $\mathcal{T}$ jezgru?*

(Jedna od mogućih) motivacija

Mnogi NP-potpuni problemi definirani na klasi (konačnih) grafova postaju jednostavni na klasi (konačnih) stabala.

Primjeri

Neka je $\mathcal{T}$ konačno stablo.

1. *Ima li $\mathcal{T}$ Hamiltonov ciklus?*
   $\rightarrow \mathcal{O}(1)$
2. *Da li je $\mathcal{T}$ 3-obojivo?*
   $\rightarrow \mathcal{O}(1)$
3. *Ima li $\mathcal{T}$ jezgru?*
   $\rightarrow \mathcal{O}(|\mathcal{T}|)$

Jezgra

Definicija

Za dani graf $\mathcal{G} = (V, E)$, skup vrhova $K \subseteq V$ je jezgra ako:

$$\forall x, y \in K \ \neg Exy \quad \wedge \quad \forall x \in (V \setminus K) \ \exists y \in K \ Exy.$$

Problem odlučivanja ima li dani graf jezgru je NP-potpun.

Na stablima taj problem je rješiv u linearnom vremenu.

Jezgra

Definicija

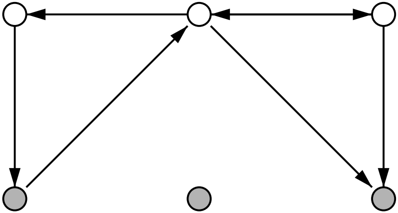
Za dani graf $\mathcal{G} = (V, E)$, skup vrhova $K \subseteq V$ je jezgra ako:

$$\forall x, y \in K \; \neg Exy \quad \wedge \quad \forall x \in (V \setminus K) \; \exists y \in K \; Exy.$$

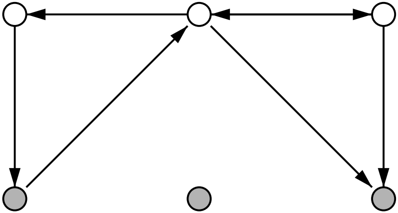
Problem odlučivanja ima li dani graf jezgru je NP-potpun.

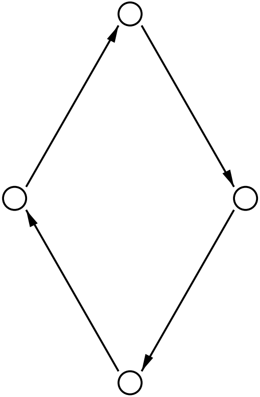
Na stablima taj problem je rješiv u linearnom vremenu.

Jezgra - primjer



Jezgra - primjer





Nalaženje jezgre u stablu u linearnom vremenu [1]

Neka je $\mathcal{T} = (T, E)$ stablo. Koristimo slijedeće oznake:

- $\mathcal{T}^u \equiv$ pripadno neusmjereno stablo;
- za $a \in T$, $\mathcal{T}_a = (T_a, E_a) \equiv$ podstablo od $\mathcal{T}$ s korijenom a;
- za $a \in T$ (a nije korijen), $\mathrm{parent}(a) \equiv$ roditelj od a u $\mathcal{T}^u$.

Osnovna ideja

Dinamičko programiranje: Za sve $a \in T$, počevši od listova prema korijenu provjeravamo:

- postoji li jezgra od $\mathcal{T}_a$ koja sadrži a;
- postoji li jezgra od $\mathcal{T}_a$ koja ne sadrži a.

Nalaženje jezgre u stablu u linearnom vremenu [1]

Neka je $\mathcal{T} = (T, E)$ stablo. Koristimo slijedeće oznake:

- $\mathcal{T}^u \equiv$ pripadno neusmjereno stablo;
- za $a \in T$, $\mathcal{T}_a = (T_a, E_a) \equiv$ podstablo od $\mathcal{T}$ s korijenom a;
- za $a \in T$ (a nije korijen), $\mathrm{parent}(a) \equiv$ roditelj od a u $\mathcal{T}^u$.

Osnovna ideja

Dinamičko programiranje: Za sve $a \in T$, počevši od listova prema korijenu provjeravamo:

- postoji li jezgra od $\mathcal{T}_a$ koja sadrži a;
- postoji li jezgra od $\mathcal{T}_a$ koja ne sadrži a.

Nalaženje jezgre u stablu u linearnom vremenu [1]

Neka je $\mathcal{T} = (T, E)$ stablo. Koristimo slijedeće oznake:

- $\mathcal{T}^u \equiv$ pripadno neusmjereno stablo;
- za $a \in T$, $\mathcal{T}_a = (T_a, E_a) \equiv$ podstablo od $\mathcal{T}$ s korijenom a;
- za $a \in T$ (a nije korijen), $\mathrm{parent}(a) \equiv$ roditelj od a u $\mathcal{T}^u$.

Osnovna ideja

Dinamičko programiranje: Za sve $a \in T$, počevši od listova prema korijenu provjeravamo:

- postoji li jezgra od $\mathcal{T}_a$ koja sadrži a;
- postoji li jezgra od $\mathcal{T}_a$ koja ne sadrži a.

Nalaženje jezgre u stablu u linearnom vremenu [1]

Neka je $\mathcal{T} = (T, E)$ stablo. Koristimo slijedeće oznake:

- $\mathcal{T}^u \equiv$ pripadno neusmjereno stablo;
- za $a \in T$, $\mathcal{T}_a = (T_a, E_a) \equiv$ podstablo od $\mathcal{T}$ s korijenom a;
- za $a \in T$ (a nije korijen), $\mathrm{parent}(a) \equiv$ roditelj od a u $\mathcal{T}^u$.

Osnovna ideja

Dinamičko programiranje: Za sve $a \in T$, počevši od listova prema korijenu provjeravamo:

- postoji li jezgra od $\mathcal{T}_a$ koja sadrži a;
- postoji li jezgra od $\mathcal{T}_a$ koja ne sadrži a.

Nalaženje jezgre u stablu u linearnom vremenu [1]

Neka je $\mathcal{T} = (T, E)$ stablo. Koristimo slijedeće oznake:

- $\mathcal{T}^u \equiv$ pripadno neusmjereno stablo;
- za $a \in T$, $\mathcal{T}_a = (T_a, E_a) \equiv$ podstablo od $\mathcal{T}$ s korijenom a;
- za $a \in T$ (a nije korijen), $\mathrm{parent}(a) \equiv$ roditelj od a u $\mathcal{T}^u$.

Osnovna ideja

Dinamičko programiranje: Za sve $a \in T$, počevši od listova prema korijenu provjeravamo:

- postoji li jezgra od $\mathcal{T}_a$ koja sadrži a;
- postoji li jezgra od $\mathcal{T}_a$ koja ne sadrži a.

Nalaženje jezgre u stablu u linearnom vremenu [1]

Neka je $\mathcal{T} = (T, E)$ stablo. Koristimo slijedeće oznake:

- $\mathcal{T}^u \equiv$ pripadno neusmjereno stablo;
- za $a \in T$, $\mathcal{T}_a = (T_a, E_a) \equiv$ podstablo od $\mathcal{T}$ s korijenom a;
- za $a \in T$ (a nije korijen), $\mathrm{parent}(a) \equiv$ roditelj od a u $\mathcal{T}^u$.

Osnovna ideja

Dinamičko programiranje: Za sve $a \in T$, počevši od listova prema korijenu provjeravamo:

- postoji li jezgra od $\mathcal{T}_a$ koja sadrži a;
- postoji li jezgra od $\mathcal{T}_a$ koja ne sadrži a.

Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo
$(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

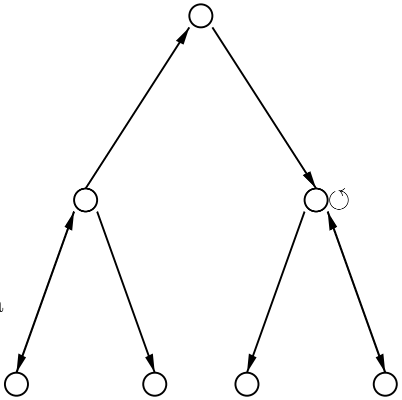
$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \not\exists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo
$(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \nexists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

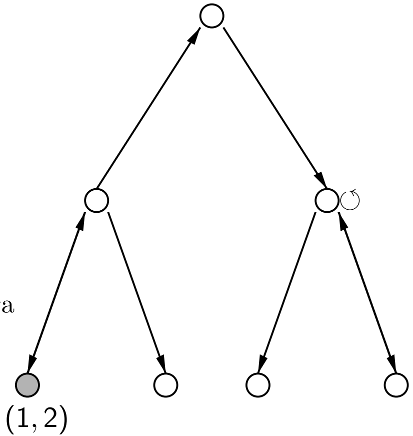


Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo
$(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \not\exists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

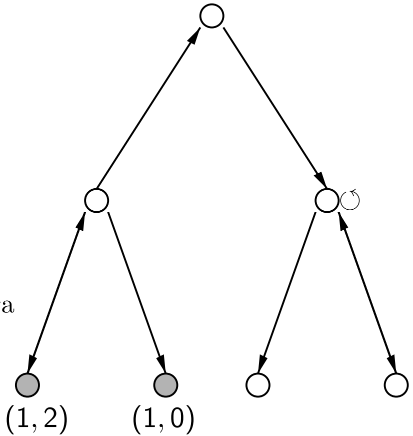


Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo
$(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \nexists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

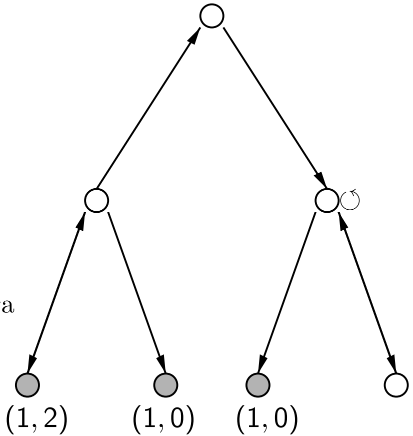


Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo
$(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \nexists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

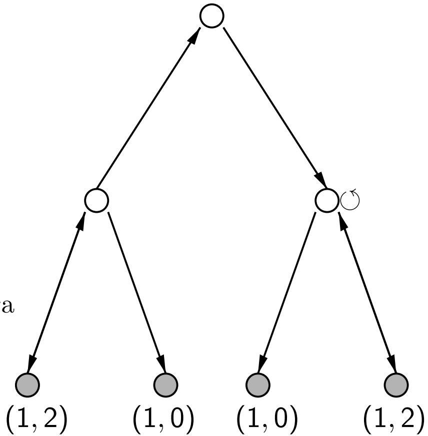


Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo $(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \nexists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

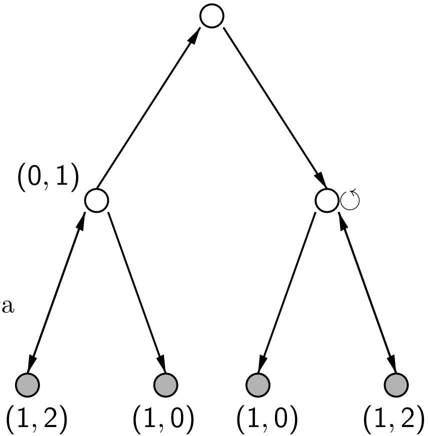


Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo
$(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \nexists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

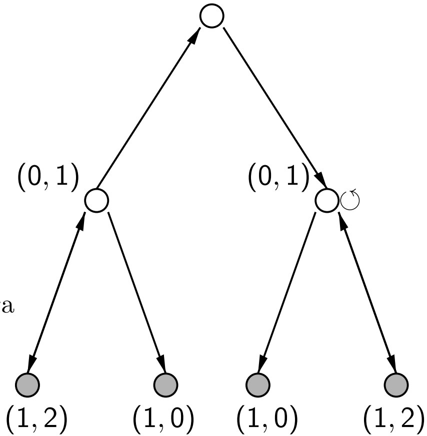


Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo
$(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \nexists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

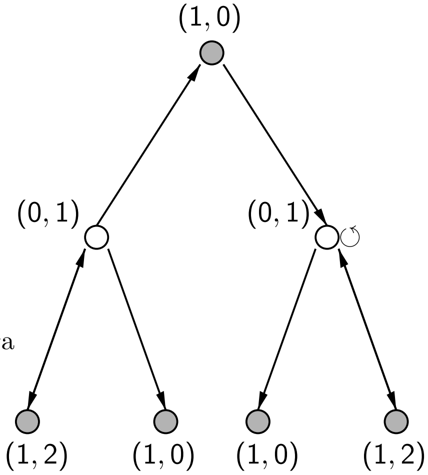


Nalaženje jezgre u stablu u linearnom vremenu [2]

Svakom $a \in T$ pridužujemo $(k_a, n_a) \in \{0,1\} \times \{0,1,2\}$:

$$k_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \ni a; \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 1, & \text{ako } \exists \text{ jezgra od } \mathcal{T}_a \not\ni a; \\ 2, & \text{ako } \nexists \text{ jezgra od } \mathcal{T}_a \not\ni a, \\ & \text{ali } E(a, \text{parent}(a)) \text{ i } \exists \text{ jezgra} \\ & \text{od } (T_a \cup \{\text{parent}(a)\}, \\ & E_a \cup \{(a, \text{parent}(a))\}) \not\ni a; \\ 0, & \text{inače.} \end{cases}$$

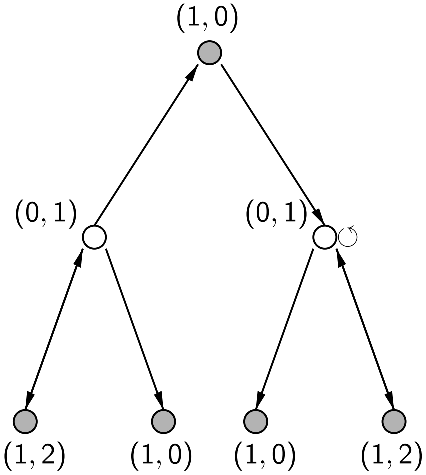


Nalaženje jezgre u stablu u linearnom vremenu [3]

Ako je a list:

$$k_a = \begin{cases} 1, & \text{ako } \neg E(a,a); \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 2, & \text{ako } E(a, \text{parent}(a)); \\ 0, & \text{inače.} \end{cases}$$

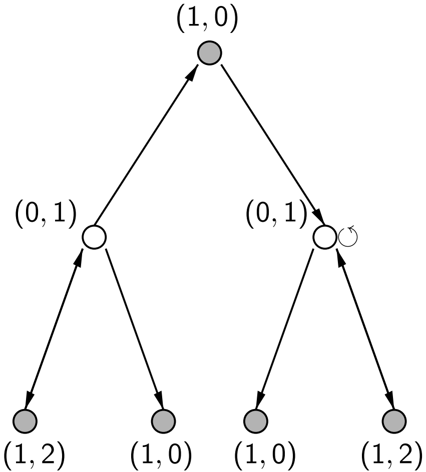


Nalaženje jezgre u stablu u linearnom vremenu [3]

Ako je a list:

$$k_a = \begin{cases} 1, & \text{ako } \neg E(a,a); \\ 0, & \text{inače.} \end{cases}$$

$$n_a = \begin{cases} 2, & \text{ako } E(a, \mathrm{parent}(a)); \\ 0, & \text{inače.} \end{cases}$$



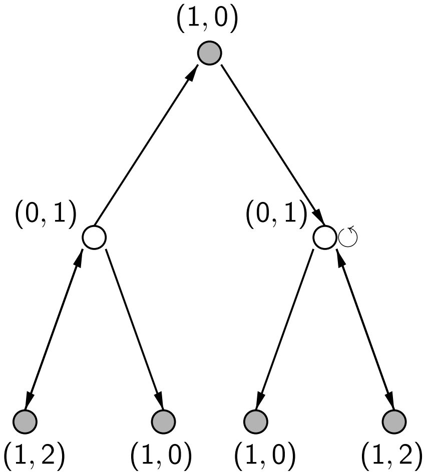
Nalaženje jezgre u stablu u linearnom vremenu [4]

Ako a nije list:

$k_a = 1$ akko $\neg E(a, a)$ i za svu djecu b od a (u $\mathcal{T}^u$) je $n_b \neq 0$;

$n_a = 1$ akko za svu djecu b od a:
$k_b + n_b > 0$ i
$(n_b = 2 \Rightarrow k_b = 1)$, te
$\exists$ b dijete od a t.d. $E(a, b)$ i
$k_b = 1$;

$n_a = 2$ akko za svu djecu b od a:
$k_b + n_b > 0$ i
$(n_b = 2 \Rightarrow k_b = 1)$, te
$\nexists$ b dijete od a t.d. $E(a, b)$ i
$k_b = 1$ i $E(a, \mathrm{parent}(a))$.



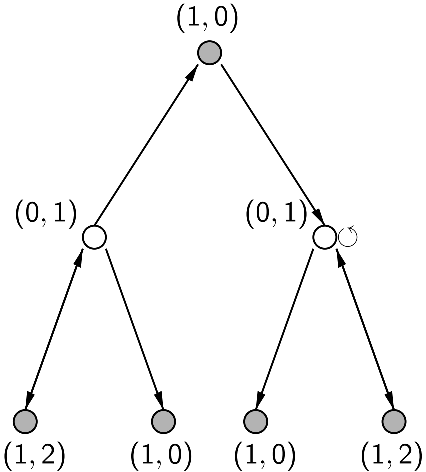
Nalaženje jezgre u stablu u linearnom vremenu [4]

Ako a nije list:

$k_a = 1$ akko $\neg E(a, a)$ i za svu djecu b od a (u $\mathcal{T}^u$) je $n_b \neq 0$;

$n_a = 1$ akko za svu djecu b od a:
$k_b + n_b > 0$ i
$(n_b = 2 \Rightarrow k_b = 1)$, te
$\exists$ b dijete od a t.d. $E(a, b)$ i
$k_b = 1$;

$n_a = 2$ akko za svu djecu b od a:
$k_b + n_b > 0$ i
$(n_b = 2 \Rightarrow k_b = 1)$, te
$\nexists$ b dijete od a t.d. $E(a, b)$ i
$k_b = 1$ i $E(a, \mathrm{parent}(a))$.



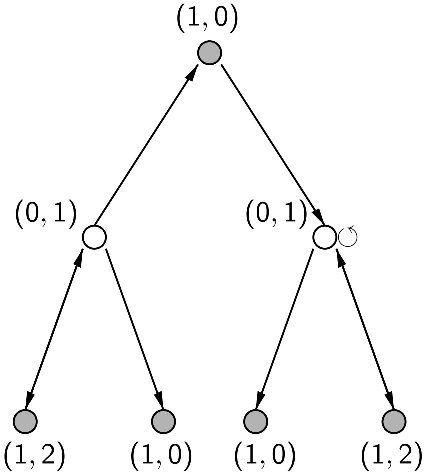
Nalaženje jezgre u stablu u linearnom vremenu [4]

Ako a nije list:

$k_a = 1$ akko $\neg E(a,a)$ i za svu djecu b od a (u $\mathcal{T}^u$) je $n_b \neq 0$;

$n_a = 1$ akko za svu djecu b od a:
$k_b + n_b > 0$ i
$(n_b = 2 \Rightarrow k_b = 1)$, te
$\exists$ b dijete od a t.d. $E(a,b)$ i
$k_b = 1$;

$n_a = 2$ akko za svu djecu b od a:
$k_b + n_b > 0$ i
$(n_b = 2 \Rightarrow k_b = 1)$, te
$\nexists$ b dijete od a t.d. $E(a,b)$ i
$k_b = 1$ i $E(a, \mathrm{parent}(a))$.

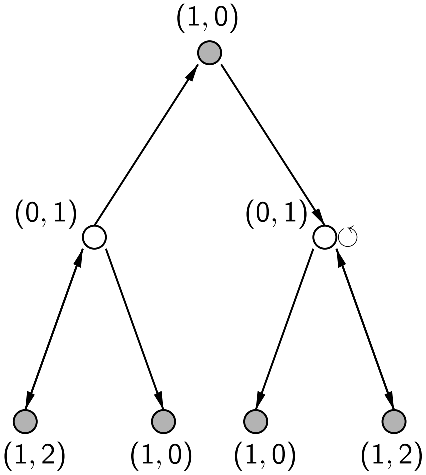


Ako a nije list:

$k_a = 1$ akko $\neg E(a,a)$ i za svu djecu b od a (u $\mathcal{T}^u$) je $n_b \neq 0$;

$n_a = 1$ akko za svu djecu b od a:
$k_b + n_b > 0$ i
$(n_b = 2 \Rightarrow k_b = 1)$, te
$\exists$ b dijete od a t.d. $E(a,b)$ i
$k_b = 1$;

$n_a = 2$ akko za svu djecu b od a:
$k_b + n_b > 0$ i
$(n_b = 2 \Rightarrow k_b = 1)$, te
$\nexists$ b dijete od a t.d. $E(a,b)$ i
$k_b = 1$ i $E(a, \mathrm{parent}(a))$.



Nije sve tako lijepo...

Ne postoji za svaki NP-potpuni problem definiran na klasi grafova efikasan algoritam za njegovu restrikciju na klasi stabala.

Band-width minimization

Problem minimizacije širine pojasa:

> *Za dani neusmjereni graf $\mathcal{G} = (V, E)$ i $k \in \mathbb{N}$, treba odrediti postoji li permutacija $(v_1, \ldots, v_n)$ od V takva da $(v_i, v_j) \in E$ povlači $|i - j| < k$.*

je NP-potpun čak i u slučaju kada se $\mathcal{G}$ uzima iz klase stabala.

C. H. Papadimitriou: Computational Complexity, p. 215, Problem 9.5.30.

Nije sve tako lijepo...

Ne postoji za svaki NP-potpuni problem definiran na klasi grafova efikasan algoritam za njegovu restrikciju na klasi stabala.

Band-width minimization

Problem minimizacije širine pojasa:

> *Za dani neusmjereni graf $\mathcal{G} = (V, E)$ i $k \in \mathbb{N}$, treba odrediti postoji li permutacija $(v_1, \ldots, v_n)$ od V takva da $(v_i, v_j) \in E$ povlači $|i - j| < k$.*

je NP-potpun čak i u slučaju kada se $\mathcal{G}$ uzima iz klase stabala.

C. H. Papadimitriou: Computational Complexity, p. 215, Problem 9.5.30.

Nije sve tako lijepo...

Ne postoji za svaki NP-potpuni problem definiran na klasi grafova efikasan algoritam za njegovu restrikciju na klasi stabala.

Band-width minimization

Problem minimizacije širine pojasa:

> *Za dani neusmjereni graf* $\mathcal{G} = (V, E)$ *i* $k \in \mathbb{N}$, *treba odrediti postoji li permutacija* $(v_1, \ldots, v_n)$ *od* V *takva da* $(v_i, v_j) \in E$ *povlači* $|i - j| < k$.

je NP-potpun čak i u slučaju kada se $\mathcal{G}$ uzima iz klase stabala.

C. H. Papadimitriou: Computational Complexity, p. 215, Problem 9.5.30.

All together...

Problem	Složenost na stablima
Hamiltonov ciklus	DTIME(1)
3-obojivost	DTIME(1)
Jezgra	DTIME(n)
Band-width	NPTIME

All together...

Problem	Složenost na stablima
Hamiltonov ciklus	$\text{DTIME}(1)$
3-obojivost	$\text{DTIME}(1)$
Jezgra	$\text{DTIME}(n)$
Band-width	NPTIME

Postoji li ipak neko objašnjenje?

Veza s MSO-definabilnošću [1]

Problem	Složenost na stablima	MSO-definabilnost
Hamiltonov ciklus	$\textsc{Dtime}(1)$	
3-obojivost	$\textsc{Dtime}(1)$	
Jezgra	$\textsc{Dtime}(n)$	
Band-width	$\textsc{NPtime}$	

Veza s MSO-definabilnošću [1]

Problem	Složenost na stablima	MSO-definabilnost
Hamiltonov ciklus	$\textsc{Dtime}(1)$	
3-obojivost	$\textsc{Dtime}(1)$	✓
Jezgra	$\textsc{Dtime}(n)$	
Band-width	$\textsc{NPtime}$	

3-obojivost

$$\exists R \exists G \exists B\ (\forall u (Ru \vee Gu \vee Bu) \wedge$$
$$\forall u \forall v\ Euv \rightarrow ((Ru \wedge \neg Rv) \vee (Gu \wedge \neg Gv) \vee (Bu \wedge \neg Bv)))$$

Veza s MSO-definabilnošću [1]

Problem	Složenost na stablima	MSO-definabilnost
Hamiltonov ciklus	$\textsc{Dtime}(1)$	
3-obojivost	$\textsc{Dtime}(1)$	✓
Jezgra	$\textsc{Dtime}(n)$	✓
Band-width	$\textsc{NPtime}$	

Jezgra

$$\exists K\ (\forall u \forall v((Ku \land Kv) \leftrightarrow \neg Euv) \land$$
$$\forall u(\neg Ku \rightarrow \exists v(Kv \land Euv)))$$

Veza s MSO-definabilnošću [1]

Problem	Složenost na stablima	MSO-definabilnost
Hamiltonov ciklus	$\textsc{Dtime}(1)$	$\times$
3-obojivost	$\textsc{Dtime}(1)$	$\checkmark$
Jezgra	$\textsc{Dtime}(n)$	$\checkmark$
Band-width	$\textsc{NPtime}$	$\times$

Veza s MSO-definabilnošću [2]

Teorem (Thatcher, Wright (1968.))

Neka je φ MSO-rečenica i $\mathcal{T}$ (konačno) stablo. Tada je $\mathcal{T} \models \varphi$ moguće odlučiti u vremenu $\mathcal{O}(|\mathcal{T}|)$.

Glavni korak dokaza.

Za danu MSO-rečenicu φ konstruirati automat na stablima $\mathcal{A}(\varphi)$ i pokazati da $\mathcal{A}(\varphi)$ prihvaća točno stabla iz skupa $\{\mathcal{T} \mid \mathcal{T} \models \varphi\}$. $\square$

Cilj

Generalizacija teorema na strukture koje su "slične" stablima.

Veza s MSO-definabilnošću [2]

Teorem (Thatcher, Wright (1968.))

Neka je φ MSO-rečenica i $\mathcal{T}$ (konačno) stablo.
Tada je $\mathcal{T} \models \varphi$ moguće odlučiti u vremenu $\mathcal{O}(|\mathcal{T}|)$.

Glavni korak dokaza.

Za danu MSO-rečenicu φ konstruirati automat na stablima $\mathcal{A}(\varphi)$ i pokazati da $\mathcal{A}(\varphi)$ prihvaća točno stabla iz skupa $\{\mathcal{T} \mid \mathcal{T} \models \varphi\}$. $\square$

Cilj

Generalizacija teorema na strukture koje su "slične" stablima.

Veza s MSO-definabilnošću [2]

Teorem (Thatcher, Wright (1968.))

Neka je φ MSO-rečenica i $\mathcal{T}$ (konačno) stablo.
Tada je $\mathcal{T} \models \varphi$ moguće odlučiti u vremenu $\mathcal{O}(|\mathcal{T}|)$.

Glavni korak dokaza.

Za danu MSO-rečenicu φ konstruirati automat na stablima $\mathcal{A}(\varphi)$ i pokazati da $\mathcal{A}(\varphi)$ prihvaća točno stabla iz skupa $\{\mathcal{T} \mid \mathcal{T} \models \varphi\}$. $\square$

Cilj

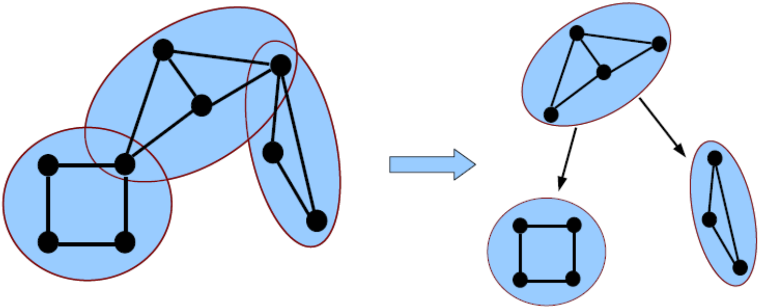
Generalizacija teorema na strukture koje su "slične" stablima.

Pregled

Pregled

Intuicija

Tree-width grafa određuje u kojoj mjeri je graf "sličan" stablu.



Graf ima tree-width $\leq k$ ukoliko se može pokriti podgrafovima veličine $\leq (k+1)$ na "stablasti" način.

tree-dekompozicija

Definicija (tree-dekompozicija)

$(\mathcal{T}, (X_t)_{t \in T})$ *je* *tree-dekompozicija* *grafa* $\mathcal{G} = (V, E)$ *ako:*

- $\mathcal{T}$ *je stablo;*
- $\bigcup_{t \in T} X_t = V$;
- *za sve* $(v, w) \in E$ *postoji* $t \in T$ *t.d.* $v \in X_t$ *i* $w \in X_t$;
- *za sve* $t, t', t'' \in T$, *ako* t *leži na putu između* t' *i* t'', *onda* $X_{t'} \cap X_{t''} \subseteq X_t$
  (tj. za svaki $v \in V$ *skup* $\{t \in T \mid v \in X_t\}$ *je povezan u* $\mathcal{T}$*).*

N. Robertson, P. D. Seymour: Graph Minors II. Algorithmic Aspects of Tree-Width. Journal of Algorithms 7, 309–322, 1986.

Definicija (tree-dekompozicija)

$(\mathcal{T}, (X_t)_{t \in T})$ *je* *tree-dekompozicija* *grafa* $\mathcal{G} = (V, E)$ *ako:*

- $\mathcal{T}$ *je stablo;*
- $\bigcup_{t \in T} X_t = V$;
- *za sve* $(v, w) \in E$ *postoji* $t \in T$ *t.d.* $v \in X_t$ *i* $w \in X_t$;
- *za sve* $t, t', t'' \in T$, *ako* t *leži na putu između* t' *i* t'', *onda* $X_{t'} \cap X_{t''} \subseteq X_t$ *(tj. za svaki* $v \in V$ *skup* $\{t \in T \mid v \in X_t\}$ *je povezan u* $\mathcal{T}$*).*

N. Robertson, P. D. Seymour: Graph Minors II. Algorithmic Aspects of Tree-Width. Journal of Algorithms 7, 309–322, 1986.

tree-dekompozicija

Definicija (tree-dekompozicija)

$(\mathcal{T}, (X_t)_{t \in T})$ *je* *tree-dekompozicija* *grafa* $\mathcal{G} = (V, E)$ *ako:*

- $\mathcal{T}$ *je stablo;*
- $\bigcup_{t \in T} X_t = V$;
- *za sve* $(v, w) \in E$ *postoji* $t \in T$ *t.d.* $v \in X_t$ *i* $w \in X_t$;
- *za sve* $t, t', t'' \in T$, *ako* t *leži na putu između* t' *i* t'', *onda* $X_{t'} \cap X_{t''} \subseteq X_t$ *(tj. za svaki* $v \in V$ *skup* $\{t \in T \mid v \in X_t\}$ *je povezan u* $\mathcal{T}$*).*

N. Robertson, P. D. Seymour: Graph Minors II. Algorithmic Aspects of Tree-Width. Journal of Algorithms 7, 309–322, 1986.

Definicija (tree-dekompozicija)

$(\mathcal{T}, (X_t)_{t \in T})$ *je* ***tree-dekompozicija*** *grafa* $\mathcal{G} = (V, E)$ *ako:*

- $\mathcal{T}$ *je stablo;*
- $\bigcup_{t \in T} X_t = V$;
- *za sve* $(v, w) \in E$ *postoji* $t \in T$ *t.d.* $v \in X_t$ *i* $w \in X_t$;
- *za sve* $t, t', t'' \in T$, *ako* t *leži na putu između* t' *i* t'', *onda* $X_{t'} \cap X_{t''} \subseteq X_t$
  (tj. za svaki $v \in V$ *skup* $\{t \in T \mid v \in X_t\}$ *je povezan u* $\mathcal{T}$*).*

N. Robertson, P. D. Seymour: Graph Minors II. Algorithmic Aspects of Tree-Width. Journal of Algorithms 7, 309–322, 1986.

tree-dekompozicija

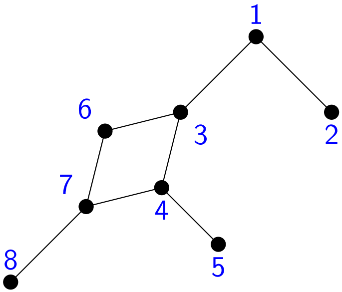
Definicija (tree-dekompozicija)

$(\mathcal{T}, (X_t)_{t\in T})$ *je* *tree-dekompozicija* *grafa* $\mathcal{G} = (V, E)$ *ako:*

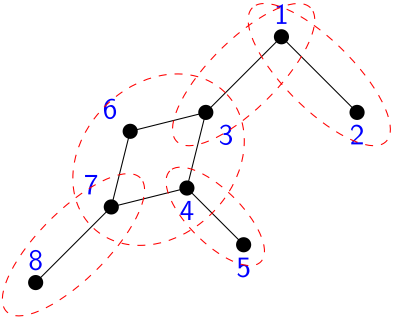
- $\mathcal{T}$ *je stablo;*
- $\bigcup_{t\in T} X_t = V$*;*
- *za sve* $(v, w) \in E$ *postoji* $t \in T$ *t.d.* $v \in X_t$ *i* $w \in X_t$*;*
- *za sve* $t, t', t'' \in T$*, ako* t *leži na putu između* t' *i* t''*, onda* $X_{t'} \cap X_{t''} \subseteq X_t$
  (tj. za svaki $v \in V$ *skup* $\{t \in T \mid v \in X_t\}$ *je povezan u* $\mathcal{T}$*).*

N. Robertson, P. D. Seymour: Graph Minors II. Algorithmic Aspects of Tree-Width. Journal of Algorithms 7, 309–322, 1986.

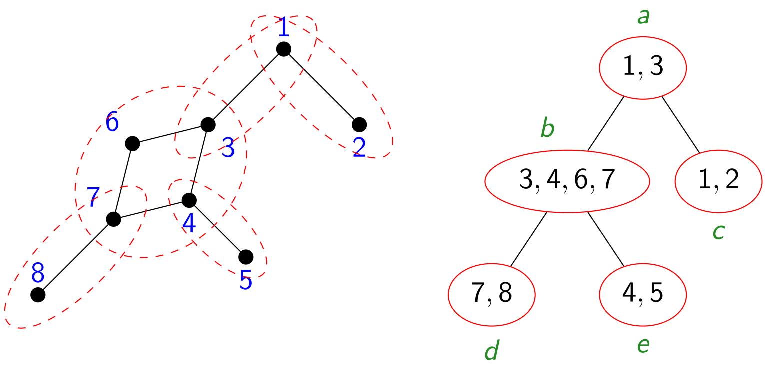
Primjer tree-dekompozicije



Primjer tree-dekompozicije



Primjer tree-dekompozicije



Definicija

Širina tree-dekompozicije $(\mathcal{T}, (X_t)_{t \in T})$:

$$\max\{|X_t| - 1 \mid t \in T\}.$$

Definicija (tree-width)

Tree-width grafa $\mathcal{G}$:

$$\mathrm{tw}(\mathcal{G}) := \min\{w \mid \mathcal{G} \textit{ ima tree-dekompoziciju širine } w\}.$$

Tree-dekompozicija i tree-width proizvoljne relacijske strukture $\mathfrak{A}$ definira se kao tree-dekompozicija i tree-width pripadajućeg Gaifmanovog grafa $\mathcal{G}(\mathfrak{A})$.

Definicija

Širina tree-dekompozicije $(\mathcal{T}, (X_t)_{t \in T})$*:*

$$\max \{|X_t| - 1 \mid t \in T\}.$$

Definicija (tree-width)

Tree-width grafa $\mathcal{G}$*:*

$$\mathrm{tw}(\mathcal{G}) := \min \{w \mid \mathcal{G} \textit{ ima tree-dekompoziciju širine } w\}.$$

Tree-dekompozicija i tree-width proizvoljne relacijske strukture $\mathfrak{A}$ definira se kao tree-dekompozicija i tree-width pripadajućeg Gaifmanovog grafa $\mathcal{G}(\mathfrak{A})$.

Definicija

Širina tree-dekompozicije $(\mathcal{T}, (X_t)_{t \in T})$*:*

$$\max\{|X_t| - 1 \mid t \in T\}.$$

Definicija (tree-width)

Tree-width grafa $\mathcal{G}$*:*

$$\mathrm{tw}(\mathcal{G}) := \min\{w \mid \mathcal{G} \textit{ ima tree-dekompoziciju širine } w\}.$$

Tree-dekompozicija i tree-width proizvoljne relacijske strukture $\mathfrak{A}$ definira se kao tree-dekompozicija i tree-width pripadajućeg Gaifmanovog grafa $\mathcal{G}(\mathfrak{A})$.

Još primjera

Primjeri (Tree-width nekih grafova)

Još primjera

Primjeri (Tree-width nekih grafova)

1. *tree-width stabala i šuma:*

Još primjera

Primjeri (Tree-width nekih grafova)

1. *tree-width stabala i šuma:* ≤ 1

Još primjera

Primjeri (Tree-width nekih grafova)

1. *tree-width stabala i šuma:* ≤ 1
2. *tree-width ciklusa:*

Još primjera

Primjeri (Tree-width nekih grafova)

1. *tree-width stabala i šuma:* ≤ 1
2. *tree-width ciklusa:* 2

Još primjera

Primjeri (Tree-width nekih grafova)

1. *tree-width stabala i šuma:* ≤ 1
2. *tree-width ciklusa:* 2
3. *tree-width klike s k vrhova:*

Još primjera

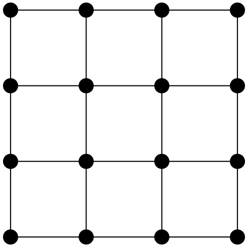
Primjeri (Tree-width nekih grafova)

1. *tree-width stabala i šuma:* ≤ 1
2. *tree-width ciklusa:* 2
3. *tree-width klike s k vrhova:* $k - 1$

Još primjera

Primjeri (Tree-width nekih grafova)

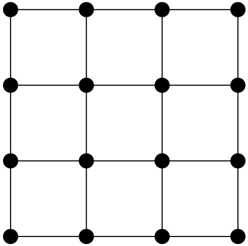
1. *tree-width stabala i šuma:* ≤ 1
2. *tree-width ciklusa:* 2
3. *tree-width klike s k vrhova:* $k-1$
4. *tree-width* $n \times n$ *mreže:*



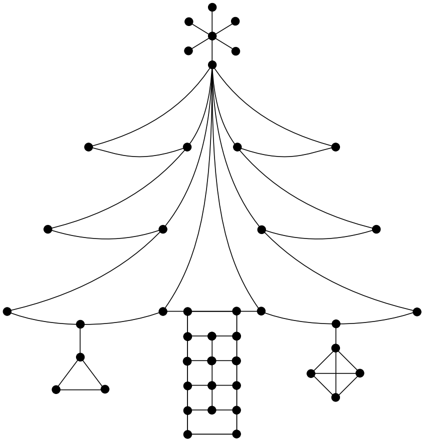
Još primjera

Primjeri (Tree-width nekih grafova)

1. *tree-width stabala i šuma:* ≤ 1
2. *tree-width ciklusa:* 2
3. *tree-width klike s k vrhova:* $k-1$
4. *tree-width $n \times n$ mreže:* n

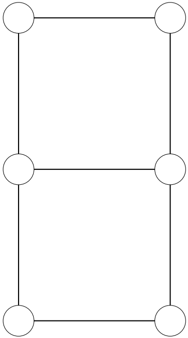


5.) :-)



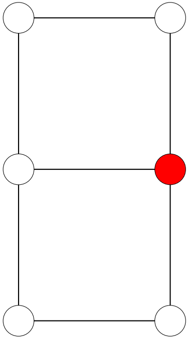
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



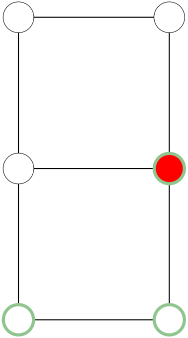
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



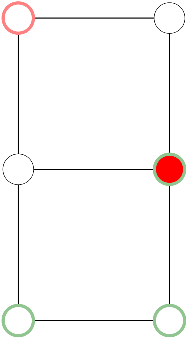
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



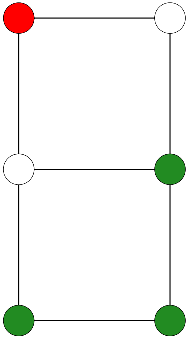
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



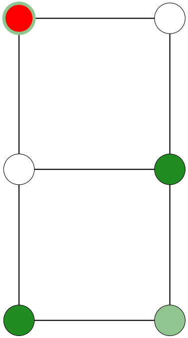
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



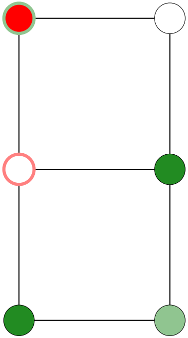
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



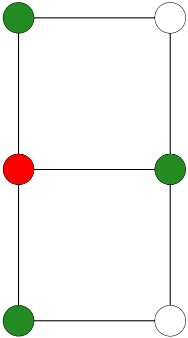
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \rightarrow (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



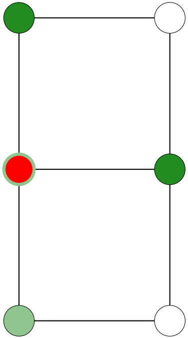
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



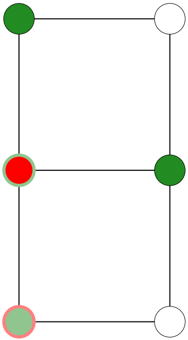
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



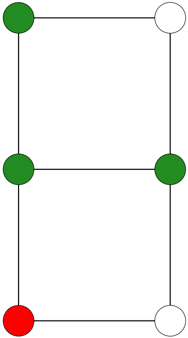
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



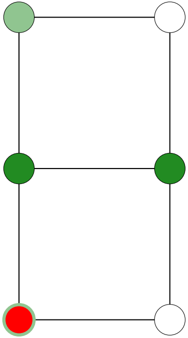
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



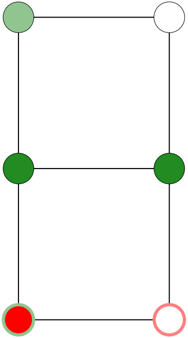
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



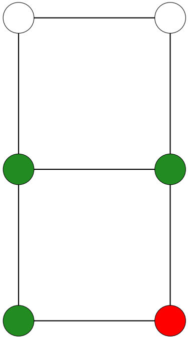
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \rightarrow (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



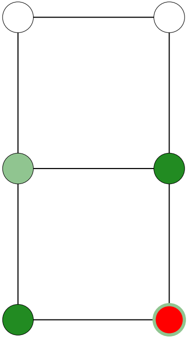
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



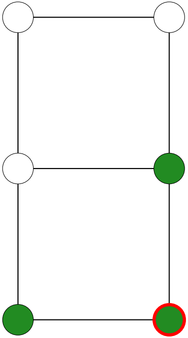
Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \rightarrow (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



Igra robber-and-cops [1]

- Igrači: Policija (cops) i lopov (robber);
- Pozicije: (C, r);
- Inicijalna pozicija: $(\emptyset, r_0)$, gdje $r_0 \in V$ bira robber;
- Potezi: $(C, r) \to (C', r')$
  Cops prvo biraju C', a zatim robber bira r' takav da postoji put od r do r' u $\mathcal{G} \setminus (C \cap C')$;
- Pobjednički uvjet: Cops pobjeđuju ako dostignu poziciju (C, r) za koju je $r \in C$; ako takva pozicija ne postoji pobjeđuje robber.



Igra robber-and-cops [2]

Teorem (Seymour, Thomas (1993.))

Graf $\mathcal{G}$ ima tree-width $\leq k$
akko
$(k+1)$ *policajaca ima pobjedničku strategiju u robber-and-cops igri na $\mathcal{G}$.*

P. D. Seymour, R. Thomas: Graph Searching and a Min-Max Theorem for Tree-Width. Journal of Combinatorial Theory, Series B 58, 22–33, 1993.

Pregled

Rezultati o složenosti [1]

Teorem (Arnborg, Corneil, Proskurowski (1987.))

Za dani graf $\mathcal{G}$ i $k \in \mathbb{N}$ problem odlučivanja da li je $\mathrm{tw}(\mathcal{G}) \leq k$ je NP-potpun.

S. Arnborg, D. G. Corneil, A. Proskurowski. Complexity of finding embeddings in a *k*-tree. SIAM J. Alg. Dis. Meth. 8: 277–284, 1987.

Otvoreni problem

Složenost nalaženja tree-widtha na klasi planarnih grafova.

Rezultati o složenosti [1]

Teorem (Arnborg, Corneil, Proskurowski (1987.))

Za dani graf $\mathcal{G}$ i $k \in \mathbb{N}$ problem odlučivanja da li je $\mathrm{tw}(\mathcal{G}) \leq k$ je NP-potpun.

S. Arnborg, D. G. Corneil, A. Proskurowski. Complexity of finding embeddings in a *k*-tree. SIAM J. Alg. Dis. Meth. 8: 277–284, 1987.

Otvoreni problem

Složenost nalaženja tree-widtha na klasi planarnih grafova.

Rezultati o složenosti [2]

Za fiksno k, problem odlučivanja da li je $\mathrm{tw}(\mathcal{G}) \leq k$ prestaje biti NP-potpun.

- Arnborg, Corneil, Proskurowski (1987.): $\mathcal{O}(n^{k+2})$;
- Robertson, Seymour (1995.): $\mathcal{O}(n^2)$
  (dokazana “samo” egzistencija algoritma)
- Bodlaender (1996.): $\mathcal{O}(n)$

H. L. Bodlaender: A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM Journal on Computing, 25: 1305–1317, 1996.

Rezultati o složenosti [2]

Za fiksno k, problem odlučivanja da li je $\mathrm{tw}(\mathcal{G}) \leq k$ prestaje biti NP-potpun.

- Arnborg, Corneil, Proskurowski (1987.): $\mathcal{O}(n^{k+2})$;
- Robertson, Seymour (1995.): $\mathcal{O}(n^2)$
  (dokazana "samo" egzistencija algoritma)
- Bodlaender (1996.): $\mathcal{O}(n)$

H. L. Bodlaender: A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM Journal on Computing, 25: 1305–1317, 1996.

Rezultati o složenosti [2]

Za fiksno k, problem odlučivanja da li je $\mathrm{tw}(\mathcal{G}) \leq k$ prestaje biti NP-potpun.

- Arnborg, Corneil, Proskurowski (1987.): $\mathcal{O}(n^{k+2})$;
- Robertson, Seymour (1995.): $\mathcal{O}(n^2)$ (dokazana "samo" egzistencija algoritma)
- Bodlaender (1996.): $\mathcal{O}(n)$

H. L. Bodlaender: A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM Journal on Computing, 25: 1305–1317, 1996.

Rezultati o složenosti [2]

Za fiksno k, problem odlučivanja da li je $\mathrm{tw}(\mathcal{G}) \leq k$ prestaje biti NP-potpun.

- Arnborg, Corneil, Proskurowski (1987.): $\mathcal{O}(n^{k+2})$;
- Robertson, Seymour (1995.): $\mathcal{O}(n^2)$ (dokazana "samo" egzistencija algoritma)
- Bodlaender (1996.): $\mathcal{O}(n)$

H. L. Bodlaender: A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM Journal on Computing, 25: 1305–1317, 1996.

Pregled

Pregled

1. Uvod

2. Tree-width
   - Osnovne definicije i svojstva
   - Igra robber-and-cops
   - Rezultati o složenosti

3. Problem verifikacije modela za strukture slične stablima
   - MSO
   - FO

4. Vezane teme

Problem verifikacije modela

Definicija (Problem verifikacije modela)

Neka je $\mathcal{C}$ klasa (konačnih) struktura i $\mathcal{L}$ logika.
Za rečenicu $\psi \in \mathcal{L}$ i strukturu $\mathfrak{A} \in \mathcal{C}$ potrebno je odrediti vrijedi li $\mathfrak{A} \models \psi$.

Složenost problema verifikacije modela

- Kombinirana složenost:
  složenost problema $\mathrm{MC}(\mathcal{C}, \mathcal{L}) := \{(\psi, \mathfrak{A}) \mid \psi \in \mathcal{L}, \mathfrak{A} \in \mathcal{C} \text{ i } \mathfrak{A} \models \psi\}$;
- Semantička složenost (za rečenicu $\psi \in \mathcal{L}$):
  složenost problema $\text{D-MC}(\mathcal{C}, \mathcal{L}) := \{\mathfrak{A} \mid \mathfrak{A} \in \mathcal{C} \text{ i } \mathfrak{A} \models \psi\}$.

Problem verifikacije modela

Definicija (Problem verifikacije modela)

Neka je $\mathcal{C}$ klasa (konačnih) struktura i $\mathcal{L}$ logika.
Za rečenicu $\psi \in \mathcal{L}$ i strukturu $\mathfrak{A} \in \mathcal{C}$ potrebno je odrediti vrijedi li $\mathfrak{A} \models \psi$.

Složenost problema verifikacije modela

- Kombinirana složenost:
  složenost problema $\mathrm{MC}(\mathcal{C}, \mathcal{L}) := \{(\psi, \mathfrak{A}) \mid \psi \in \mathcal{L}, \mathfrak{A} \in \mathcal{C} \text{ i } \mathfrak{A} \models \psi\}$;
- Semantička složenost (za rečenicu $\psi \in \mathcal{L}$):
  složenost problema $\text{D-MC}(\mathcal{C}, \mathcal{L}) := \{\mathfrak{A} \mid \mathfrak{A} \in \mathcal{C} \text{ i } \mathfrak{A} \models \psi\}$.

Problem verifikacije modela

Definicija (Problem verifikacije modela)

Neka je $\mathcal{C}$ klasa (konačnih) struktura i $\mathcal{L}$ logika.
Za rečenicu $\psi \in \mathcal{L}$ i strukturu $\mathfrak{A} \in \mathcal{C}$ potrebno je odrediti vrijedi li $\mathfrak{A} \models \psi$.

Složenost problema verifikacije modela

- Kombinirana složenost:
  složenost problema $\mathrm{MC}(\mathcal{C}, \mathcal{L}) := \{(\psi, \mathfrak{A}) \mid \psi \in \mathcal{L}, \mathfrak{A} \in \mathcal{C} \text{ i } \mathfrak{A} \models \psi\}$;
- Semantička složenost (za rečenicu $\psi \in \mathcal{L}$):
  složenost problema $\mathrm{D\text{-}MC}(\mathcal{C}, \mathcal{L}) := \{\mathfrak{A} \mid \mathfrak{A} \in \mathcal{C} \text{ i } \mathfrak{A} \models \psi\}$.

Problem verifikacije modela

Definicija (Problem verifikacije modela)

Neka je $\mathcal{C}$ klasa (konačnih) struktura i $\mathcal{L}$ logika.
Za rečenicu $\psi \in \mathcal{L}$ i strukturu $\mathfrak{A} \in \mathcal{C}$ potrebno je odrediti vrijedi li $\mathfrak{A} \models \psi$.

Složenost problema verifikacije modela

- Kombinirana složenost:
  složenost problema $\mathrm{MC}(\mathcal{C}, \mathcal{L}) := \{(\psi, \mathfrak{A}) \mid \psi \in \mathcal{L},\ \mathfrak{A} \in \mathcal{C} \text{ i } \mathfrak{A} \models \psi\}$;
- Semantička složenost (za rečenicu $\psi \in \mathcal{L}$):
  složenost problema $\mathrm{D\text{-}MC}(\mathcal{C}, \mathcal{L}) := \{\mathfrak{A} \mid \mathfrak{A} \in \mathcal{C} \text{ i } \mathfrak{A} \models \psi\}$.

Verifikacija modela za MSO [1]

Teorem (Courcelle (1990.))

Neka je $\mathcal{C}$ klasa struktura omeđenog tree-widtha.
Tada je $\mathrm{D\text{-}MC}(\mathcal{C}, \mathrm{MSO})$ rješiv u linearnom vremenu.

B. Courcelle: Graph rewritting: An algebraic and logic approach. Handbook of Theoretical Computer Science, Vol. 2, 194–242, Elsevier, 1990.

Reminder: Složenost nalaženja tree-dekompozicije

Teorem (Bodlaender (1996.))

Postoji algoritam koji za danu strukturu $\mathfrak{A}$ *i* $w \geq 1$ *nalazi tree-dekompoziciju od* $\mathfrak{A}$ *širine najviše* w *ako je* $\mathrm{tw}(\mathfrak{A}) \leq w$*, odnosno odbacuje ako je* $\mathrm{tw}(\mathfrak{A}) > w$*.*
Vremenska složenost algoritma iznosi $\mathcal{O}(2^{c \cdot w^3} \cdot |\mathfrak{A}|)$*.*

Verifikacija modela za MSO [2]

Skica dokaza

- Neka je $\varphi \in MSO$, $q := \mathrm{qr}(\varphi)$, $w := \mathrm{tw}(\mathcal{C})$.
  Neka je $\mathfrak{A} \in \mathcal{C}$ i $(\mathcal{T}, (X_t)_{t \in T})$ tree-dekompozicija od $\mathfrak{A}$.
- Za $t \in T$, označimo $Y_t := \bigcup_{s \succcurlyeq t} X_s$.
  Definiramo niz $\bar{a}^t := a_1 \ldots a_{w+1}$ t.d. $X_t = \{a_1, \ldots, a_{w+1}\}$.
  q-tip od $\bar{a}^t$: $\mathbf{T}^q(\bar{a}^t) := \left\{\psi \in \mathrm{MSO} \mid \mathrm{qr}(\psi) \leq q \wedge \langle Y_t \rangle^{\mathcal{A}} \models \psi \right\}$.

Usporedba s algoritmom za nalaženje jezgre

Odnos uloga za $t \in T$:

- Podstablo ukorijenjeno u t $\quad\leftrightarrow\quad$ Y_t;
- t $\quad\leftrightarrow\quad$ $\bar{a}^t$;
- (k_t, n_t) $\quad\leftrightarrow\quad$ $\mathbf{T}^q(\bar{a}^t)$.

Verifikacija modela za MSO [2]

Skica dokaza

- Neka je $\varphi \in MSO$, $q := \mathrm{qr}(\varphi)$, $w := \mathrm{tw}(\mathcal{C})$.
  Neka je $\mathfrak{A} \in \mathcal{C}$ i $(\mathcal{T}, (X_t)_{t \in T})$ tree-dekompozicija od $\mathfrak{A}$.
- Za $t \in T$, označimo $Y_t := \bigcup_{s \succcurlyeq t} X_s$.
  Definiramo niz $\bar{a}^t := a_1 \ldots a_{w+1}$ t.d. $X_t = \{a_1, \ldots, a_{w+1}\}$.
  q-tip od $\bar{a}^t$: $\mathbf{T}^q(\bar{a}^t) := \left\{ \psi \in \mathrm{MSO} \mid \mathrm{qr}(\psi) \leq q \wedge \langle Y_t \rangle^{\mathcal{A}} \models \psi \right\}$.

Usporedba s algoritmom za nalaženje jezgre

Odnos uloga za $t \in T$:

- Podstablo ukorjenjeno u t $\leftrightarrow$ Y_t;
- t $\leftrightarrow$ $\bar{a}^t$;
- (k_t, n_t) $\leftrightarrow$ $\mathbf{T}^q(\bar{a}^t)$.

Verifikacija modela za MSO [2]

Skica dokaza

- Neka je $\varphi \in MSO$, $q := \mathrm{qr}(\varphi)$, $w := \mathrm{tw}(\mathcal{C})$.
  Neka je $\mathfrak{A} \in \mathcal{C}$ i $(\mathcal{T}, (X_t)_{t \in T})$ tree-dekompozicija od $\mathfrak{A}$.
- Za $t \in T$, označimo $Y_t := \bigcup_{s \succcurlyeq t} X_s$.
  Definiramo niz $\bar{a}^t := a_1 \ldots a_{w+1}$ t.d. $X_t = \{a_1, \ldots, a_{w+1}\}$.
  q-tip od $\bar{a}^t$: $\mathbf{T}^q(\bar{a}^t) := \left\{\psi \in \mathrm{MSO} \mid \mathrm{qr}(\psi) \le q \wedge \langle Y_t \rangle^{\mathcal{A}} \models \psi \right\}$.

Usporedba s algoritmom za nalaženje jezgre

Odnos uloga za $t \in T$:

- Podstablo ukorjenjeno u t $\quad\leftrightarrow\quad$ Y_t;
- t $\quad\leftrightarrow\quad$ $\bar{a}^t$;
- (k_t, n_t) $\quad\leftrightarrow\quad$ $\mathbf{T}^q(\bar{a}^t)$.

Verifikacija modela za MSO [2]

Skica dokaza

- Neka je $\varphi \in MSO$, $q := \mathrm{qr}(\varphi)$, $w := \mathrm{tw}(\mathcal{C})$.
  Neka je $\mathfrak{A} \in \mathcal{C}$ i $(\mathcal{T}, (X_t)_{t \in T})$ tree-dekompozicija od $\mathfrak{A}$.
- Za $t \in T$, označimo $Y_t := \bigcup_{s \succcurlyeq t} X_s$.
  Definiramo niz $\bar{a}^t := a_1 \dots a_{w+1}$ t.d. $X_t = \{a_1, \dots, a_{w+1}\}$.
  q-tip od $\bar{a}^t$: $\mathbf{T}^q(\bar{a}^t) := \left\{\psi \in \mathrm{MSO} \mid \mathrm{qr}(\psi) \leq q \wedge \langle Y_t \rangle^{\mathcal{A}} \models \psi \right\}$.

Usporedba s algoritmom za nalaženje jezgre

Odnos uloga za $t \in T$:

- Podstablo ukorjenjeno u t $\quad\leftrightarrow\quad$ Y_t;
- t $\quad\leftrightarrow\quad$ $\bar{a}^t$;
- (k_t, n_t) $\quad\leftrightarrow\quad$ $\mathbf{T}^q(\bar{a}^t)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\rightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\longrightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\rightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\rightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\rightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\rightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\longrightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\rightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [3]

Skica dokaza

- Koristeći E-F igre za MSO pokaže se da za $t \in T$ s djecom u i v, $\mathbf{T}^q(\bar{a}^t)$ ovisi o:
  - tipu izomorfizma podstrukture inducirane s $\bar{a}^t$;
  - $\mathbf{T}^q(\bar{a}^u)$ i $\mathbf{T}^q(\bar{a}^v)$;
  - "presjeku" $\bar{a}^t$ i $\bar{a}^u$, te $\bar{a}^t$ i $\bar{a}^v$.

  Za fiksno q, funkcija koja opisuje ovu ovisnost može se odrediti unaprijed.
- Za $t \in T$, računaju se $\mathbf{T}^q(\bar{a}^t)$ počevši od listova prema korijenu.
- Iz q-tipa korijena određuje se vrijedi li $\mathfrak{A} \models \varphi$.

Algoritam prolazi jednom kroz svaki čvor stabla, a u svakom čvoru vrši ograničen broj koraka $\rightarrow$ ukupan broj koraka iznosi $\mathcal{O}(|\mathfrak{A}|)$.

Verifikacija modela za MSO [4]

Teorem (Flum, Frick, Grohe (2002.))

Neka je $\mathcal{C}$ klasa struktura omeđenog tree-widtha i $\varphi(\bar{x}, \overline{Y})$ MSO-formula. Postoji algoritam koji za danu strukturu $\mathfrak{A} \in \mathcal{C}$ odlučuje postoje li $a_1, \ldots, a_m \in A$ i $B_1, \ldots, B_n \subseteq A$ takvi da $\mathfrak{A} \models \varphi[a_1, \ldots, a_m, B_1, \ldots, B_n]$ i u potvrdnom slučaju ih efektivno određuje.
Vremenska složenost algoritma iznosi $\mathcal{O}(|\mathfrak{A}|)$.

J. Flum, M. Frick, M. Grohe: Query evaluation via tree-decompositions. Journal of the ACM 49:716-752, 2002.

Pregled

Verifikacija modela za FO [1]

Možemo li u slučaju logike prvog reda kazati nešto više?

Verifikacija modela za FO [1]

Možemo li u slučaju logike prvog reda kazati nešto više?

Teorem (Gaifman)

Svaka FO*-rečenica ekvivalentna je "lokalnoj" rečenici.*

Verifikacija modela za FO [1]

Možemo li u slučaju logike prvog reda kazati nešto više?

Teorem (Gaifman)

Svaka FO*-rečenica ekvivalentna je "lokalnoj" rečenici.*

Svaka FO-rečenica tvrdi egzistenciju u parovima disjunktnih "kugala" čiji "centri" zadovoljavaju određena svojstva u podstrukturama koje su generirane tim "kuglama".

Verifikacija modela za FO [2]

Definicija (Lokalni tree-width)

Lokalni tree-width strukture $\mathfrak{A}$ *je funkcija* $\mathrm{ltw}^{\mathfrak{A}}\colon \mathbb{N} \to \mathbb{N}$ *dana s:*

$$\mathrm{ltw}^{\mathfrak{A}}(r) := \max\left\{\mathrm{tw}(\langle B_r(a)\rangle^{\mathfrak{A}}) \mid a \in A\right\},$$

gdje $B_r(a)$ *označava kuglu radijusa* r *oko* a *u Gaifmanovom grafu* $\mathcal{G}(\mathfrak{A})$.

Teorem (Flum, Frick (2001.))

Neka je $\mathcal{C}$ *klasa struktura omeđenog lokalnog tree-widtha. Tada je* $\mathrm{D\text{-}MC}(\mathcal{C}, \mathrm{FO})$ *rješiv u linearnom vremenu.*

J. Flum, M. Frick: Deciding first-order properties of locally tree-decomposable structures. Journal of the ACM 48:1184-1206, 2001.

Verifikacija modela za FO [2]

Definicija (Lokalni tree-width)

Lokalni tree-width strukture $\mathfrak{A}$ *je funkcija* $\mathrm{ltw}^{\mathfrak{A}} \colon \mathbb{N} \to \mathbb{N}$ *dana s:*

$$\mathrm{ltw}^{\mathfrak{A}}(r) := \max \left\{ \mathrm{tw}(\langle B_r(a) \rangle^{\mathfrak{A}}) \mid a \in A \right\},$$

gdje $B_r(a)$ *označava kuglu radijusa* r *oko* a *u Gaifmanovom grafu* $\mathcal{G}(\mathfrak{A})$.

Teorem (Flum, Frick (2001.))

Neka je $\mathcal{C}$ *klasa struktura omeđenog lokalnog tree-widtha. Tada je* $\mathrm{D\text{-}MC}(\mathcal{C}, \mathrm{FO})$ *rješiv u linearnom vremenu.*

J. Flum, M. Frick: Deciding first-order properties of locally tree-decomposable structures. Journal of the ACM 48:1184-1206, 2001.

Pregled

Još o odnosu tree-widtha i MSO-logike

Teorem (Courcelle, 1990.)

Ako je $\mathcal{C}$ klasa grafova omeđenog tree-widtha, tada je MSO-teorija od $\mathcal{C}$ odlučiva.

B. Courcelle. The monadic second-order logic of graphs I. Recognizable sets of finite graphs. Information and Computation, 85: 12–75, 1990.

Teorem (Seese, 1991.)

Ako je MSO-teorija klase $\mathcal{C}$ planarnih grafova odlučiva, tada je $\mathcal{C}$ omeđenog tree-widtha.

D. Seese. The structure of the models of decidable monadic theories of graphs. Ann. Pure Appl. Logic, 53: 169–195, 1991.

Seese-ova slutnja

Ako je $\mathcal{C}$ proizvoljna klasa grafova čija MSO-teorija je odlučiva, tada je $\mathcal{C}$ omeđenog tree-widtha.

Još o odnosu tree-widtha i MSO-logike

Teorem (Courcelle, 1990.)

Ako je $\mathcal{C}$ klasa grafova omeđenog tree-widtha, tada je MSO-teorija od $\mathcal{C}$ odlučiva.

B. Courcelle. The monadic second-order logic of graphs I. Recognizable sets of finite graphs. Information and Computation, 85: 12–75, 1990.

Teorem (Seese, 1991.)

Ako je MSO-teorija klase $\mathcal{C}$ planarnih grafova odlučiva, tada je $\mathcal{C}$ omeđenog tree-widtha.

D. Seese. The structure of the models of decidable monadic theories of graphs. Ann. Pure Appl. Logic, 53: 169–195, 1991.

Seese-ova slutnja

Ako je $\mathcal{C}$ proizvoljna klasa grafova čija MSO-teorija je odlučiva, tada je $\mathcal{C}$ omeđenog tree-widtha.

Još o odnosu tree-widtha i MSO-logike

Teorem (Courcelle, 1990.)

Ako je $\mathcal{C}$ klasa grafova omeđenog tree-widtha, tada je MSO-teorija od $\mathcal{C}$ odlučiva.

B. Courcelle. The monadic second-order logic of graphs I. Recognizable sets of finite graphs. Information and Computation, 85: 12–75, 1990.

Teorem (Seese, 1991.)

Ako je MSO-teorija klase $\mathcal{C}$ planarnih grafova odlučiva, tada je $\mathcal{C}$ omeđenog tree-widtha.

D. Seese. The structure of the models of decidable monadic theories of graphs. Ann. Pure Appl. Logic, 53: 169–195, 1991.

Seese-ova slutnja

Ako je $\mathcal{C}$ proizvoljna klasa grafova čija MSO-teorija je odlučiva, tada je $\mathcal{C}$ omeđenog tree-widtha.

Još o odnosu tree-widtha i MSO-logike

Teorem (Courcelle, 1990.)

Ako je $\mathcal{C}$ klasa grafova omeđenog tree-widtha, tada je MSO-teorija od $\mathcal{C}$ odlučiva.

B. Courcelle. The monadic second-order logic of graphs I. Recognizable sets of finite graphs. Information and Computation, 85: 12–75, 1990.

Teorem (Seese, 1991.)

Ako je MSO-teorija klase $\mathcal{C}$ planarnih grafova odlučiva, tada je $\mathcal{C}$ omeđenog tree-widtha.

D. Seese. The structure of the models of decidable monadic theories of graphs. Ann. Pure Appl. Logic, 53: 169–195, 1991.

Seese-ova slutnja

Ako je $\mathcal{C}$ proizvoljna klasa grafova čija MSO-teorija je odlučiva, tada je $\mathcal{C}$ omeđenog tree-widtha.

Koncepti bliski tree-dekompoziciji

- Parcijalna *k*-stabla
  (Arnborg, Proskurovski)
- Nostrumi
  (Ajtai, Gurevich)
- Path-width, Branch-width
  (Robertson, Seymour)
- Clique-width
  (Courcelle, Engelfriet, Rozenberg)

Dekompozicija usmjerenih grafova

- Directed tree-width
  Johnson, Robertson, Seymour, Thomas: Directed Tree-Width. Journal of Combinatorial Theory, Series B 82, 138–155, 2001.
- Entanglement
  Berwanger, Grädel. LPAR 2004.
- DAG-width
  Berwanger, Dawar, Hunter, Kreutzer. STACS 2006.
  Obdržálek. SODA 2006.

- Directed tree-width

  Johnson, Robertson, Seymour, Thomas: Directed Tree-Width. Journal of Combinatorial Theory, Series B 82, 138–155, 2001.

- Entanglement

  Berwanger, Grädel. LPAR 2004.

- DAG-width

  Berwanger, Dawar, Hunter, Kreutzer. STACS 2006.

  Obdržálek. SODA 2006.

Dekompozicija usmjerenih grafova

- Directed tree-width

  Johnson, Robertson, Seymour, Thomas: Directed Tree-Width. Journal of Combinatorial Theory, Series B 82, 138–155, 2001.

- Entanglement

  Berwanger, Grädel. LPAR 2004.

- DAG-width

  Berwanger, Dawar, Hunter, Kreutzer. STACS 2006.

  Obdržálek. SODA 2006.

Dekompozicija usmjerenih grafova

- Directed tree-width

  Johnson, Robertson, Seymour, Thomas: Directed Tree-Width. Journal of Combinatorial Theory, Series B 82, 138–155, 2001.

- Entanglement

  Berwanger, Grädel. LPAR 2004.

- DAG-width

  Berwanger, Dawar, Hunter, Kreutzer. STACS 2006.

  Obdržálek. SODA 2006.