Almost tight lower bounds on regular resolution refutations of Tseitin Formulas for all constant-degree graphs

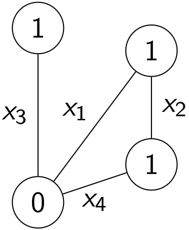
Dmitry Itsykson[1] Artur Riazanov[1] Danil Sagunov[1]
Petr Smirnov[2]

[1]Steklov institute of Mathematics at St. Petersburg
[2] St. Petersburg State University

Proof Complexity Workshop
Banff International Research Station
January 23, 2020

Tseitin formulas

- Let $G(V, E)$ be an undirected graph.
- $f : V \to \{0, 1\}$ is a charging function.
- Edge $e \in E \mapsto$ variable x_e.
- $T(G, f) = \bigwedge_{v \in v} Parity(v)$, where $Parity(v) = \left(\sum_{e \text{ is incident to } v} x_e = f(v) \bmod 2 \right)$.
- $T(G, f)$ is represented in CNF.

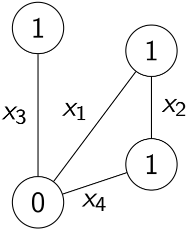


$$\begin{cases} x_1 + x_2 = 1 \\ x_1 + x_3 + x_4 = 0 \\ x_2 + x_4 = 1 \\ x_3 = 1 \end{cases}$$

- [Urquhart, 1987] $T(G, f)$ is satisfiable $\iff$ for every connected component $U \subseteq V$, $\sum_{v \in U} f(v) = 0$.

Tseitin formulas

- Let $G(V, E)$ be an undirected graph.
- $f : V \to \{0, 1\}$ is a charging function.
- Edge $e \in E \mapsto$ variable x_e.
- $T(G, f) = \bigwedge_{v \in v} Parity(v)$, where $Parity(v) = \left(\sum_{e \text{ is incident to } v} x_e = f(v) \bmod 2 \right)$.
- $T(G, f)$ is represented in CNF.



$$\begin{cases} x_1 + x_2 = 1 \\ x_1 + x_3 + x_4 = 0 \\ x_2 + x_4 = 1 \\ x_3 = 1 \end{cases}$$

- [Urquhart, 1987] $T(G, f)$ is satisfiable $\iff$ for every connected component $U \subseteq V$, $\sum_{v \in U} f(v) = 0$.

Resolution and its subsystems

- Resolution refutation of a CNF formula ϕ
  - **Resolution rule** $\frac{C \vee x, D \vee \neg x}{C \vee D}$,
  - A refutation of ϕ is a sequence of clauses $C_1, C_2, \ldots, C_s$ such that
    - for every i, C_i is either a clause of ϕ or is obtained by the resolution rule from previous.
    - C_s is an empty clause.
- **Regular resolution**: for any path in the proof-graph no variable is used twice in a resolution rule.
- **Tree-like resolution**: the proof-graph is a tree.

$$S(\phi) \leq S_{reg}(\phi) \leq S_T(\phi)$$

- **Resolution width** The width of a clause is the number of literals in it. The width of a refutation is the maximal width of a clause in it. $w(\phi)$ is the minimal posible width of resolution refutation of ϕ.

Tseitin formulas and resolution

- Lower bounds for particular graphs
  - $S_{reg}(T(\boxplus_n, f)) = n^{\omega(1)}$ where $\boxplus_n$ is $n \times n$ grid (Tseitin, 1968).
  - $S(T(\boxplus_n, f)) = 2^{\Omega(n)}$ (Dantchev, Riis, 2001)
  - $S(T(G, f)) = 2^{\Omega(n)}$ for an expander G with n vertices (Urquhart, 1987, Ben-Sasson, Wigderson, 2001).
- Upper bound (Alekhnovich, Razborov, 2011)
  - $S_{reg}(T(G, f) = 2^{O(w(T(G,f)))}\mathrm{poly}(|V|)$, where $w(\phi)$ is a resolution width of ϕ.
- **Urquhart's conjecture.** Regular resolution polynomially simulates general resolution on Tseitin formulas.
- **Stronger conjecture.** $S(T(G, f)) = 2^{\Omega(w(T(G,f)))}$

- It is false for star graph S_n, $S(T(S_n, f) = O(n)$, while $w(T(S_n, f)) = n$.
- Perhaps, the conjecture is true for constant-degree graphs.

Tseitin formulas and resolution

- Lower bounds for particular graphs
  - $S_{reg}(T(\boxplus_n, f)) = n^{\omega(1)}$ where $\boxplus_n$ is $n \times n$ grid (Tseitin, 1968).
  - $S(T(\boxplus_n, f)) = 2^{\Omega(n)}$ (Dantchev, Riis, 2001)
  - $S(T(G, f)) = 2^{\Omega(n)}$ for an expander G with n vertices (Urquhart, 1987, Ben-Sasson, Wigderson, 2001).
- Upper bound (Alekhnovich, Razborov, 2011)
  - $S_{reg}(T(G, f) = 2^{O(w(T(G,f)))}\mathrm{poly}(|V|)$, where $w(\phi)$ is a resolution width of ϕ.
- **Urquhart's conjecture.** Regular resolution polynomially simulates general resolution on Tseitin formulas.
- **Stronger conjecture.** $S(T(G, f)) = 2^{\Omega(w(T(G,f)))}$

- It is false for star graph S_n, $S(T(S_n, f) = O(n)$, while $w(T(S_n, f)) = n$.
- Perhaps, the conjecture is true for constant-degree graphs.

Tseitin formulas and resolution

- Lower bounds for particular graphs
  - $S_{reg}(T(\boxplus_n, f)) = n^{\omega(1)}$ where $\boxplus_n$ is $n \times n$ grid (Tseitin, 1968).
  - $S(T(\boxplus_n, f)) = 2^{\Omega(n)}$ (Dantchev, Riis, 2001)
  - $S(T(G, f)) = 2^{\Omega(n)}$ for an expander G with n vertices (Urquhart, 1987, Ben-Sasson, Wigderson, 2001).
- Upper bound (Alekhnovich, Razborov, 2011)
  - $S_{reg}(T(G, f) = 2^{O(w(T(G,f)))}\mathrm{poly}(|V|)$, where $w(\phi)$ is a resolution width of ϕ.
- **Urquhart's conjecture.** Regular resolution polynomially simulates general resolution on Tseitin formulas.
- **Stronger conjecture.** $S(T(G, f)) = 2^{\Omega(w(T(G,f)))}$

- It is false for star graph S_n, $S(T(S_n, f) = O(n)$, while $w(T(S_n, f)) = n$.
- Perhaps, the conjecture is true for constant-degree graphs.

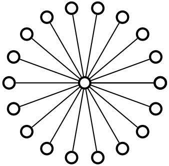
Tseitin formulas and resolution

- Lower bounds for particular graphs
  - $S_{reg}(T(\boxplus_n, f)) = n^{\omega(1)}$ where $\boxplus_n$ is $n \times n$ grid (Tseitin, 1968).
  - $S(T(\boxplus_n, f)) = 2^{\Omega(n)}$ (Dantchev, Riis, 2001)
  - $S(T(G, f)) = 2^{\Omega(n)}$ for an expander G with n vertices (Urquhart, 1987, Ben-Sasson, Wigderson, 2001).
- Upper bound (Alekhnovich, Razborov, 2011)
  - $S_{reg}(T(G, f) = 2^{O(w(T(G,f)))}\mathrm{poly}(|V|)$, where $w(\phi)$ is a resolution width of ϕ.
- **Urquhart's conjecture.** Regular resolution polynomially simulates general resolution on Tseitin formulas.
- **Stronger conjecture.** $S(T(G, f)) = 2^{\Omega(w(T(G,f)))}$

- It is false for star graph S_n, $S(T(S_n, f) = O(n)$, while $w(T(S_n, f)) = n$.
- Perhaps, the conjecture is true for constant-degree graphs.

Tseitin formulas and resolution

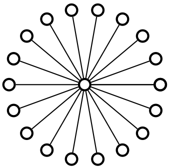
- Lower bounds for particular graphs
  - $S_{reg}(T(\boxplus_n, f)) = n^{\omega(1)}$ where $\boxplus_n$ is $n \times n$ grid (Tseitin, 1968).
  - $S(T(\boxplus_n, f)) = 2^{\Omega(n)}$ (Dantchev, Riis, 2001)
  - $S(T(G, f)) = 2^{\Omega(n)}$ for an expander G with n vertices (Urquhart, 1987, Ben-Sasson, Wigderson, 2001).
- Upper bound (Alekhnovich, Razborov, 2011)
  - $S_{reg}(T(G, f) = 2^{O(w(T(G,f)))}\mathrm{poly}(|V|)$, where $w(\phi)$ is a resolution width of ϕ.
- **Urquhart's conjecture.** Regular resolution polynomially simulates general resolution on Tseitin formulas.
- **Stronger conjecture.** $S(T(G, f)) = 2^{\Omega(w(T(G,f)))}$



- It is false for star graph S_n, $S(T(S_n, f) = O(n)$, while $w(T(S_n, f)) = n$.
- Perhaps, the conjecture is true for constant-degree graphs.

Tseitin formulas and resolution

- Lower bounds for particular graphs
  - $S_{reg}(T(\boxplus_n, f)) = n^{\omega(1)}$ where $\boxplus_n$ is $n \times n$ grid (Tseitin, 1968).
  - $S(T(\boxplus_n, f)) = 2^{\Omega(n)}$ (Dantchev, Riis, 2001)
  - $S(T(G, f)) = 2^{\Omega(n)}$ for an expander G with n vertices (Urquhart, 1987, Ben-Sasson, Wigderson, 2001).
- Upper bound (Alekhnovich, Razborov, 2011)
  - $S_{reg}(T(G, f) = 2^{O(w(T(G,f)))}\mathrm{poly}(|V|)$, where $w(\phi)$ is a resolution width of ϕ.
- **Urquhart's conjecture.** Regular resolution polynomially simulates general resolution on Tseitin formulas.
- **Stronger conjecture.** $S(T(G, f)) = 2^{\Omega(w(T(G,f)))}$



- It is false for star graph S_n, $S(T(S_n, f) = O(n)$, while $w(T(S_n, f)) = n$.
- Perhaps, the conjecture is true for constant-degree graphs.

Constant degree graphs

- (Galesi et al. 2018) $w(T(G,f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Håstad, 2017) Let S be the size of the shortest d-depth Frege proof of $T(\boxplus_n, f)$. Then $S \geq 2^{n^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$
  - For resolution this method gives $S(T(G,f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G,f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G, f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Håstad, 2017) Let S be the size of the shortest d-depth Frege proof of $T(\boxplus_n, f)$. Then $S \geq 2^{n^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$
  - For resolution this method gives $S(T(G, f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G, f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G,f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Håstad, 2017) Let S be the size of the shortest d-depth Frege proof of $T(\boxplus_n, f)$. Then $S \geq 2^{n^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$
  - For resolution this method gives $S(T(G,f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G,f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G, f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Håstad, 2017) Let S be the size of the shortest d-depth Frege proof of $T(\boxplus_n, f)$. Then $S \geq 2^{n^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$
  - For resolution this method gives $S(T(G, f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G, f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G,f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Håstad, 2017) Let S be the size of the shortest d-depth Frege proof of $T(\boxplus_n, f)$. Then $S \geq 2^{n^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$
  - For resolution this method gives $S(T(G,f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G,f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G,f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Håstad, 2017) Let S be the size of the shortest d-depth Frege proof of $T(\boxplus_n, f)$. Then $S \geq 2^{n^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$
  - For resolution this method gives $S(T(G,f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G,f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G,f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Håstad, 2017) Let S be the size of the shortest d-depth Frege proof of $T(\boxplus_n, f)$. Then $S \geq 2^{n^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$
  - For resolution this method gives $S(T(G,f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G,f)) \leq 2^{\Omega(\mathrm{tw}(G)\log|V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G,f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Galesi et. al., 2019) Let S be the size of the shortest d-depth Frege proof of $T(G,f)$. Then $S \geq 2^{\mathrm{tw}(G)^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$.
  - For resolution this method gives $S(T(G,f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G,f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G,f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Galesi et. al., 2019) Let S be the size of the shortest d-depth Frege proof of $T(G,f)$. Then $S \geq 2^{\mathrm{tw}(G)^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$.
  - For resolution this method gives $S(T(G,f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G,f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Constant degree graphs

- (Galesi et al. 2018) $w(T(G,f)) = \Theta(\mathrm{tw}(G))$ for $O(1)$-degree graphs.
- The inequality $S(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ is known for following $O(1)$-degree graphs:
  - (Size-width relation): graphs with large treewidth: $\mathrm{tw}(G) = \Omega(n)$
  - (Alekhnovich, Razborov, 2011): graphs with bounded cyclicity
  - (xorification): graphs with doubled edges
- **Grid Minor Theorem** (Robertson, Seymour 1986), (Chuzhoy 2015): Every graph G has a grid minor of size $t \times t$, where $t = \Omega\left(\mathrm{tw}(G)^{\delta}\right)$.
  - Known for $\delta = 1/10$. Necessary: $\delta \leq \frac{1}{2}$.
  - (Galesi et. al., 2019) Let S be the size of the shortest d-depth Frege proof of $T(G,f)$. Then $S \geq 2^{\mathrm{tw}(G)^{\Omega(1/d)}}$ for $d \leq \frac{C \log n}{\log \log n}$.
  - For resolution this method gives $S(T(G,f)) \geq 2^{\mathrm{tw}(G)^{\delta}}$.
- Tree-like resolution
  - $S_T(T(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$ (size-width relation)
  - $S_T(T(G,f)) \leq 2^{\Omega(\mathrm{tw}(G) \log |V|)}$ (Beame, Beck, Impagliazzo, 2013, I., Oparin, 2013)

Our results

Main theorem. $S_{reg}(T(G,f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size $S^{O(\log|V|)}$.
   - If $S_T(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size S.

2. $\text{1-BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Previouse result: (Glinskih, I. 2019)

     constant from Grid Minor Theorem

Example. There exist $O(1)$-degree graphs $G_n(V_n, E_n)$ such that $\text{1-BP}(\mathrm{T}(G_n, c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$ and $\mathrm{tw}(G_n) = n^{\Omega(1)}$.

- $S_T(\mathrm{T}(G_n,c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$, $S_{reg}(\mathrm{T}(G_n,c)) = 2^{\Theta(\mathrm{tw}(G_n))}$.

Our results

Main theorem. $S_{reg}(T(G, f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G, f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G, f')$ of size $S^{O(\log|V|)}$.
   - If $S_T(T(G, f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G, f')$ of size S.
     - Remark: it is not true for decision trees. Let P_n be a path with doubled edges. Then $S_T(T(P_n, f)) = O(n^2)$ but any decision tree computing satisfiable $T(P_n, f)$ has size at least 2^n.
2. $1\text{-BP}(T(G, f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Previouse result: (Glinskih, I., 2019) $2^{O(\mathrm{tw}(G)\log|V|)} \geq 1\text{-BP}(T(G, f')) \geq 2^{\Omega(\mathrm{tw}(G)^{\delta})}$, where δ is a constant from Grid Minor Theorem.

Example. There exist $O(1)$-degree graphs $G_n(V_n, E_n)$ such that $1\text{-BP}(\mathrm{T}(G_n, c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$ and $\mathrm{tw}(G_n) = n^{\Omega(1)}$.

- $S_T(\mathrm{T}(G_n, c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$, $S_{reg}(\mathrm{T}(G_n, c)) = 2^{\Theta(\mathrm{tw}(G_n))}$.

Our results

Main theorem. $S_{reg}(T(G,f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size $S^{O(\log|V|)}$.
   - If $S_T(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size S.
     - Remark: it is not true for decision trees. Let P_n be a path with doubled edges. Then $S_T(T(P_n,f)) = O(n^2)$ but any decision tree computing satisfiable $T(P_n,f)$ has size at least 2^n.
2. 1-$\mathrm{BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Previouse result: (Glinskih, I., 2019) $2^{O(\mathrm{tw}(G)\log|V|)} \geq$ 1-$\mathrm{BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G)^\delta)}$, where δ is a constant from Grid Minor Theorem.

Example. There exist $O(1)$-degree graphs $G_n(V_n,E_n)$ such that 1-$\mathrm{BP}(\mathrm{T}(G_n,c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$ and $\mathrm{tw}(G_n) = n^{\Omega(1)}$.

- $S_T(\mathrm{T}(G_n,c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$, $S_{reg}(\mathrm{T}(G_n,c)) = 2^{\Theta(\mathrm{tw}(G_n))}$.

Our results

Main theorem. $S_{reg}(T(G, f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G, f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G, f')$ of size $S^{O(\log|V|)}$.
   - If $S_T(T(G, f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G, f')$ of size S.
     - Remark: it is not true for decision trees. Let P_n be a path with doubled edges. Then $S_T(T(P_n, f)) = O(n^2)$ but any decision tree computing satisfiable $T(P_n, f)$ has size at least 2^n.
2. 1-$\mathrm{BP}(T(G, f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Previouse result: (Glinskih, I., 2019) $2^{O(\mathrm{tw}(G)\log|V|)} \geq$ 1-$\mathrm{BP}(T(G, f')) \geq 2^{\Omega(\mathrm{tw}(G)^{\delta})}$, where δ is a constant from Grid Minor Theorem.

Example. There exist $O(1)$-degree graphs $G_n(V_n, E_n)$ such that 1-$\mathrm{BP}(\mathrm{T}(G_n, c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$ and $\mathrm{tw}(G_n) = n^{\Omega(1)}$.

- $S_T(\mathrm{T}(G_n, c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$, $S_{reg}(\mathrm{T}(G_n, c)) = 2^{\Theta(\mathrm{tw}(G_n))}$.

Our results

Main theorem. $S_{reg}(T(G,f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size $S^{O(\log|V|)}$.
   - If $S_T(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size S.
     - Remark: it is not true for decision trees. Let P_n be a path with doubled edges. Then $S_T(T(P_n,f)) = O(n^2)$ but any decision tree computing satisfiable $T(P_n,f)$ has size at least 2^n.
2. 1-$\mathrm{BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Previouse result: (Glinskih, I., 2019) $2^{O(\mathrm{tw}(G)\log|V|)} \geq$ 1-$\mathrm{BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G)^{\delta})}$, where δ is a constant from Grid Minor Theorem.

Example. There exist $O(1)$-degree graphs $G_n(V_n,E_n)$ such that 1-$\mathrm{BP}(\mathrm{T}(G_n,c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$ and $\mathrm{tw}(G_n) = n^{\Omega(1)}$.

- $S_T(\mathrm{T}(G_n,c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$, $S_{reg}(\mathrm{T}(G_n,c)) = 2^{\Theta(\mathrm{tw}(G_n))}$.

Our results

Main theorem. $S_{reg}(T(G, f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G, f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G, f')$ of size $S^{O(\log|V|)}$.
   - If $S_T(T(G, f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G, f')$ of size S.
     - Remark: it is not true for decision trees. Let P_n be a path with doubled edges. Then $S_T(T(P_n, f)) = O(n^2)$ but any decision tree computing satisfiable $T(P_n, f)$ has size at least 2^n.
2. 1-$\mathrm{BP}(T(G, f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Previouse result: (Glinskih, I., 2019) $2^{O(\mathrm{tw}(G)\log|V|)} \geq$ 1-$\mathrm{BP}(T(G, f')) \geq 2^{\Omega(\mathrm{tw}(G)^{\delta})}$, where δ is a constant from Grid Minor Theorem.

Example. There exist $O(1)$-degree graphs $G_n(V_n, E_n)$ such that 1-$\mathrm{BP}(\mathrm{T}(G_n, c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$ and $\mathrm{tw}(G_n) = n^{\Omega(1)}$.

- $S_T(\mathrm{T}(G_n, c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$, $S_{reg}(\mathrm{T}(G_n, c)) = 2^{\Theta(\mathrm{tw}(G_n))}$.

Our results

Main theorem. $S_{reg}(T(G,f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size $S^{O(\log|V|)}$.
   - If $S_T(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size S.
     - Remark: it is not true for decision trees. Let P_n be a path with doubled edges. Then $S_T(T(P_n,f)) = O(n^2)$ but any decision tree computing satisfiable $T(P_n,f)$ has size at least 2^n.
2. 1-$\mathrm{BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Previouse result: (Glinskih, I., 2019) $2^{O(\mathrm{tw}(G)\log|V|)} \geq$ 1-$\mathrm{BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G)^{\delta})}$, where δ is a constant from Grid Minor Theorem.

Example. There exist $O(1)$-degree graphs $G_n(V_n, E_n)$ such that 1-$\mathrm{BP}(\mathrm{T}(G_n,c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$ and $\mathrm{tw}(G_n) = n^{\Omega(1)}$.

- $S_T(\mathrm{T}(G_n,c)) \geq 2^{\Omega(\mathrm{tw}(G_n)\log|V_n|)}$, $S_{reg}(\mathrm{T}(G_n,c)) = 2^{\Theta(\mathrm{tw}(G_n))}$.

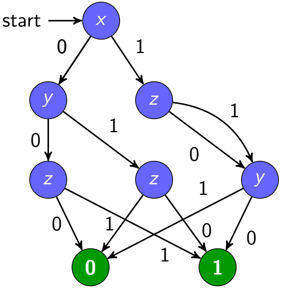
Our results

Main theorem. $S_{reg}(T(G,f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G,f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G,f')$ of size $S^{O(\log|V|)}$.
2. 1-$\mathrm{BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G))}$

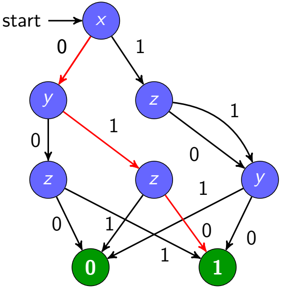
1-BP



- $f : \{0, 1\}^n \to X$ is represented by a DAG with the unique source.
- Sinks are labeled with distinct elements of X. Each non-sink node is labeled with a variable and has two outgoing edges: 0-edge and 1-edge.
- Given an assignment ξ a branching program returns the label of the sink at the end of the path corresponding to ξ.

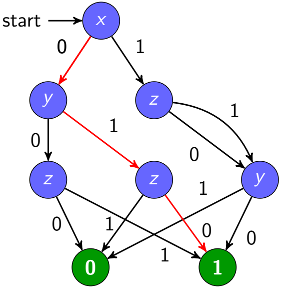
- Read-once branching program (1-BP): in every path every variable appears at most once.
- In 1-BP: $u \xrightarrow{a} v$, and u is labeled with x. If u computes f_u and v computes f_v, then $f_v = f_u|_{x=a}$.

1-BP



- $f : \{0,1\}^n \to X$ is represented by a DAG with the unique source.
- Sinks are labeled with distinct elements of X. Each non-sink node is labeled with a variable and has two outgoing edges: 0-edge and 1-edge.
- Given an assignment ξ a branching program returns the label of the sink at the end of the path corresponding to ξ.
- Read-once branching program (1-BP): in every path every variable appears at most once.
- In 1-BP: $u \xrightarrow{a} v$, and u is labeled with x. If u computes f_u and v computes f_v, then $f_v = f_u|_{x=a}$.

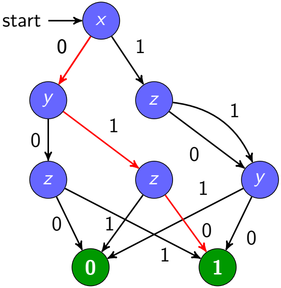
1-BP



- $f : \{0,1\}^n \to X$ is represented by a DAG with the unique source.
- Sinks are labeled with distinct elements of X. Each non-sink node is labeled with a variable and has two outgoing edges: 0-edge and 1-edge.
- Given an assignment ξ a branching program returns the label of the sink at the end of the path corresponding to ξ.

- Read-once branching program (1-BP): in every path every variable appears at most once.
- In 1-BP: $u \xrightarrow{a} v$, and u is labeled with x. If u computes f_u and v computes f_v, then $f_v = f_u|_{x=a}$.

1-BP



- $f : \{0,1\}^n \to X$ is represented by a DAG with the unique source.
- Sinks are labeled with distinct elements of X. Each non-sink node is labeled with a variable and has two outgoing edges: 0-edge and 1-edge.
- Given an assignment ξ a branching program returns the label of the sink at the end of the path corresponding to ξ.

- Read-once branching program (1-BP): in every path every variable appears at most once.
- In 1-BP: $u \xrightarrow{a} v$, and u is labeled with x. If u computes f_u and v computes f_v, then $f_v = f_u|_{x=a}$.

SearchVertex

- Search_ϕ: Let ϕ be an unsatisfiable CNF. Given an assignment σ, find a clause of ϕ falsified by σ.
- **Theorem (folklore).** ϕ has a regular resolution refutation of size S iff there exists a 1-BP of size S computing Search_ϕ.
- For an **unsatisfiable** formula $\mathrm{T}(G, f)$:
  - $\mathrm{Search}_{\mathrm{T}(G,f)}$: given an assignment, find a falsified clause of $\mathrm{T}(G, f)$.
  - $\mathrm{SearchVertex}(G, f)$: given an assignment, find a vertex of G with violated parity condition.
- **Simple observation**
  $\text{1-BP}(\mathrm{Search}_{\mathrm{T}(G,f)}) \geq \text{1-BP}(\mathrm{SearchVertex}(G, f))$.
- We are going to prove that
  $\text{1-BP}(\mathrm{T}(G, f')) \leq \text{1-BP}(\mathrm{SearchVertex}(G, f))^{\mathcal{O}(\log |V|)}$.

SearchVertex

- Search_ϕ: Let ϕ be an unsatisfiable CNF. Given an assignment σ, find a clause of ϕ falsified by σ.
- **Theorem (folklore).** ϕ has a regular resolution refutation of size S iff there exists a 1-BP of size S computing Search_ϕ.
- For an **unsatisfiable** formula $\mathrm{T}(G, f)$:
  - $\mathrm{Search}_{\mathrm{T}(G,f)}$: given an assignment, find a falsified clause of $\mathrm{T}(G, f)$.
  - $\mathrm{SearchVertex}(G, f)$: given an assignment, find a vertex of G with violated parity condition.
- **Simple observation**
  $\text{1-BP}(\mathrm{Search}_{\mathrm{T}(G,f)}) \geq \text{1-BP}(\mathrm{SearchVertex}(G, f))$.
- We are going to prove that
  $\text{1-BP}(\mathrm{T}(G, f')) \leq \text{1-BP}(\mathrm{SearchVertex}(G, f))^{\mathcal{O}(\log |V|)}$.

SearchVertex

- Search_ϕ: Let ϕ be an unsatisfiable CNF. Given an assignment σ, find a clause of ϕ falsified by σ.
- **Theorem (folklore).** ϕ has a regular resolution refutation of size S iff there exists a $1\text{-}\mathrm{BP}$ of size S computing Search_ϕ.
- For an **unsatisfiable** formula $\mathrm{T}(G, f)$:
  - $\mathrm{Search}_{\mathrm{T}(G,f)}$: given an assignment, find a falsified clause of $\mathrm{T}(G, f)$.
  - $\mathrm{SearchVertex}(G, f)$: given an assignment, find a vertex of G with violated parity condition.
- **Simple observation** $1\text{-}\mathrm{BP}(\mathrm{Search}_{\mathrm{T}(G,f)}) \geq 1\text{-}\mathrm{BP}(\mathrm{SearchVertex}(G, f))$.
- We are going to prove that $1\text{-}\mathrm{BP}(\mathrm{T}(G, f')) \leq 1\text{-}\mathrm{BP}(\mathrm{SearchVertex}(G, f))^{\mathcal{O}(\log |V|)}$.

SearchVertex

- Search_ϕ: Let ϕ be an unsatisfiable CNF. Given an assignment σ, find a clause of ϕ falsified by σ.
- **Theorem (folklore).** ϕ has a regular resolution refutation of size S iff there exists a 1-BP of size S computing Search_ϕ.
- For an **unsatisfiable** formula $\mathrm{T}(G, f)$:
  - $\mathrm{Search}_{\mathrm{T}(G,f)}$: given an assignment, find a falsified clause of $\mathrm{T}(G, f)$.
  - $\mathrm{SearchVertex}(G, f)$: given an assignment, find a vertex of G with violated parity condition.
- **Simple observation**
  $\text{1-BP}(\mathrm{Search}_{\mathrm{T}(G,f)}) \geq \text{1-BP}(\mathrm{SearchVertex}(G, f))$.
- We are going to prove that
  $\text{1-BP}(\mathrm{T}(G, f')) \leq \text{1-BP}(\mathrm{SearchVertex}(G, f))^{\mathcal{O}(\log |V|)}$.

SearchVertex

- Search_ϕ**:** Let ϕ be an unsatisfiable CNF. Given an assignment σ, find a clause of ϕ falsified by σ.
- **Theorem (folklore).** ϕ has a regular resolution refutation of size S iff there exists a 1-BP of size S computing Search_ϕ.
- For an **unsatisfiable** formula $\mathrm{T}(G, f)$:
  - $\mathrm{Search}_{\mathrm{T}(G,f)}$: given an assignment, find a falsified clause of $\mathrm{T}(G, f)$.
  - $\mathrm{SearchVertex}(G, f)$: given an assignment, find a vertex of G with violated parity condition.
- **Simple observation**
  $1\text{-}\mathrm{BP}(\mathrm{Search}_{\mathrm{T}(G,f)}) \geq 1\text{-}\mathrm{BP}(\mathrm{SearchVertex}(G, f))$.
- We are going to prove that
  $1\text{-}\mathrm{BP}(\mathrm{T}(G, f')) \leq 1\text{-}\mathrm{BP}(\mathrm{SearchVertex}(G, f))^{\mathcal{O}(\log |V|)}$.

$\mathrm{SearchVertex}(G, f)$ vs $\mathrm{Search}_{\mathrm{T}(G,f)}$

- $\mathrm{SearchVertex}(G, f)$ and $\mathrm{Search}_{\mathrm{T}(G,f)}$ are equivalent for decision trees.
- For 1-BP:

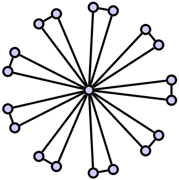
1. **Unrestricted degrees.** G_n: $\text{1-BP}(\mathrm{SearchVertex}(G_n, f)) = O(n)$, while $\mathrm{Search}_{\mathrm{T}(G_n,f)} = 2^{\Omega(n)}$.

2. **Logarithmic degrees.** $K_{\log n}$: $\text{1-BP}(\mathrm{SearchVertex}(K_{\log n}, f)) = O(n)$, while $\text{1-BP}(\mathrm{Search}_{\mathrm{T}(K_{\log n},f)}) = 2^{\Omega(\log^2 n)}$ by size-width relation.
3. **Constant degrees.** We conjecture that for $O(1)$-degree graphs two problems are polynomially equivalent. But this conjecture implies stronger inequality $S_{reg}(\mathrm{T}(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$.
   - Xorification: $S(\phi^{\oplus}) \geq 2^{\Omega(w(\phi))}$.
   - For Tseitin formulas xorification = doubling of edges.
   - It improves bound on $\mathrm{Search}_{\mathrm{T}(G_n,f)^{\oplus}}$ but 1-BP for SearchVertex increases in at most a constant.

SearchVertex(G, f) vs $\mathrm{Search}_{\mathrm{T}(G,f)}$

- SearchVertex(G, f) and $\mathrm{Search}_{\mathrm{T}(G,f)}$ are equivalent for decision trees.
- For 1-BP:

1. **Unrestricted degrees.** G_n: 1-BP(SearchVertex$(G_n, f)) = O(n)$, while $\mathrm{Search}_{\mathrm{T}(G_n,f)} = 2^{\Omega(n)}$.

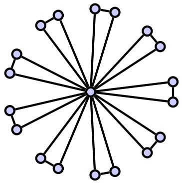


2. **Logarithmic degrees.** $K_{\log n}$: 1-BP(SearchVertex$(K_{\log n}, f)) = O(n)$, while 1-BP$(\mathrm{Search}_{\mathrm{T}(K_{\log n},f)}) = 2^{\Omega(\log^2 n)}$ by size-width relation.
3. **Constant degrees.** We conjecture that for $O(1)$-degree graphs two problems are polynomially equivalent. But this conjecture implies stronger inequality $S_{reg}(\mathrm{T}(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$.
   - Xorification: $S(\phi^{\oplus}) \geq 2^{\Omega(w(\phi))}$.
   - For Tseitin formulas xorification = doubling of edges.
   - It improves bound on $\mathrm{Search}_{\mathrm{T}(G_n,f)^{\oplus}}$ but 1-BP for SearchVertex increases in at most a constant.

$\mathrm{SearchVertex}(G, f)$ vs $\mathrm{Search}_{\mathrm{T}(G,f)}$

- $\mathrm{SearchVertex}(G, f)$ and $\mathrm{Search}_{\mathrm{T}(G,f)}$ are equivalent for decision trees.
- For 1-BP:

1. **Unrestricted degrees.** G_n: $\text{1-BP}(\mathrm{SearchVertex}(G_n, f)) = O(n)$, while $\mathrm{Search}_{\mathrm{T}(G_n,f)} = 2^{\Omega(n)}$.

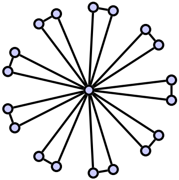


2. **Logarithmic degrees.** $K_{\log n}$: $\text{1-BP}(\mathrm{SearchVertex}(K_{\log n}, f)) = O(n)$, while $\text{1-BP}(\mathrm{Search}_{\mathrm{T}(K_{\log n},f)}) = 2^{\Omega(\log^2 n)}$ by size-width relation.
3. **Constant degrees.** We conjecture that for $O(1)$-degree graphs two problems are polynomially equivalent. But this conjecture implies stronger inequality $S_{reg}(\mathrm{T}(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$.
   - Xorification: $S(\phi^{\oplus}) \geq 2^{\Omega(w(\phi))}$.
   - For Tseitin formulas xorification = doubling of edges.
   - It improves bound on $\mathrm{Search}_{\mathrm{T}(G_n,f)^{\oplus}}$ but 1-BP for SearchVertex increases in at most a constant.

SearchVertex(G, f) vs $\mathrm{Search}_{\mathrm{T}(G,f)}$

- SearchVertex(G, f) and $\mathrm{Search}_{\mathrm{T}(G,f)}$ are equivalent for decision trees.
- For 1-BP:

1. **Unrestricted degrees.** G_n: 1-BP(SearchVertex$(G_n, f)) = O(n)$, while $\mathrm{Search}_{\mathrm{T}(G_n,f)} = 2^{\Omega(n)}$.

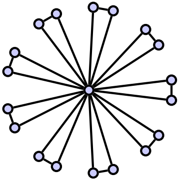


2. **Logarithmic degrees.** $K_{\log n}$: 1-BP(SearchVertex$(K_{\log n}, f)) = O(n)$, while 1-BP$(\mathrm{Search}_{\mathrm{T}(K_{\log n},f)}) = 2^{\Omega(\log^2 n)}$ by size-width relation.
3. **Constant degrees.** We conjecture that for $O(1)$-degree graphs two problems are polynomially equivalent. But this conjecture implies stronger inequality $S_{reg}(\mathrm{T}(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$.
   - Xorification: $S(\phi^{\oplus}) \geq 2^{\Omega(w(\phi))}$.
   - For Tseitin formulas xorification = doubling of edges.
   - It improves bound on $\mathrm{Search}_{\mathrm{T}(G_n,f)^{\oplus}}$ but 1-BP for SearchVertex increases in at most a constant.

$\mathrm{SearchVertex}(G, f)$ vs $\mathrm{Search}_{\mathrm{T}(G,f)}$

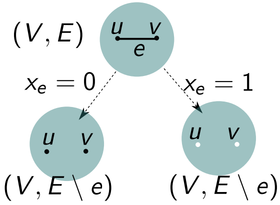
- $\mathrm{SearchVertex}(G, f)$ and $\mathrm{Search}_{\mathrm{T}(G,f)}$ are equivalent for decision trees.
- For 1-BP:

1. **Unrestricted degrees.** G_n: $\text{1-BP}(\mathrm{SearchVertex}(G_n, f)) = O(n)$, while $\mathrm{Search}_{\mathrm{T}(G_n,f)} = 2^{\Omega(n)}$.

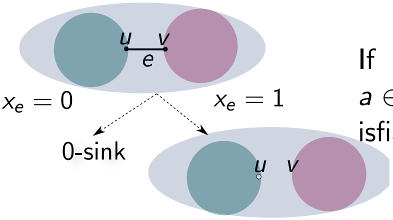


2. **Logarithmic degrees.** $K_{\log n}$: $\text{1-BP}(\mathrm{SearchVertex}(K_{\log n}, f)) = O(n)$, while $\text{1-BP}(\mathrm{Search}_{\mathrm{T}(K_{\log n},f)}) = 2^{\Omega(\log^2 n)}$ by size-width relation.
3. **Constant degrees.** We conjecture that for $O(1)$-degree graphs two problems are polynomially equivalent. But this conjecture implies stronger inequality $S_{reg}(\mathrm{T}(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$.
   - Xorification: $S(\phi^{\oplus}) \geq 2^{\Omega(w(\phi))}$.
   - For Tseitin formulas xorification = doubling of edges.
   - It improves bound on $\mathrm{Search}_{\mathrm{T}(G_n,f)^{\oplus}}$ but 1-BP for $\mathrm{SearchVertex}$ increases in at most a constant.

Structure of a 1-BP computing a satisfiable $\mathrm{T}(G, f)$

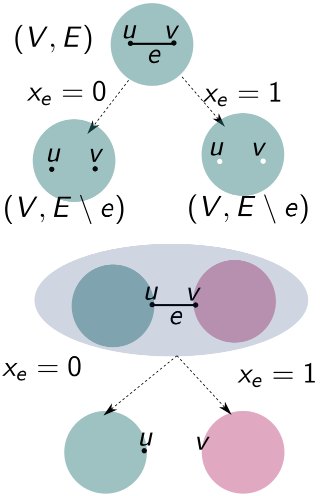


$\mathrm{T}(H, f)|_{x_e=a} = T(H - e, f + a(\mathbf{1}_u + \mathbf{1}_v))$, where $e = (u, v)$.



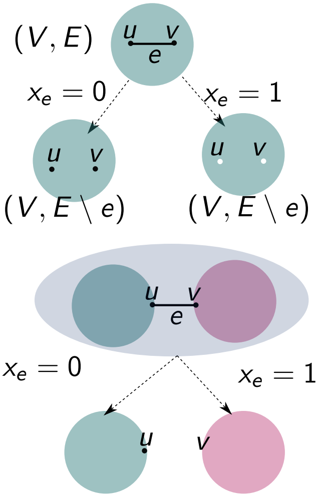
If e is a bridge, then for some $a \in \{0, 1\}$, $\mathrm{T}(H, f)|_{x_e=a}$ is unsatisfiable.

Structure of a 1-BP computing SearchVertex



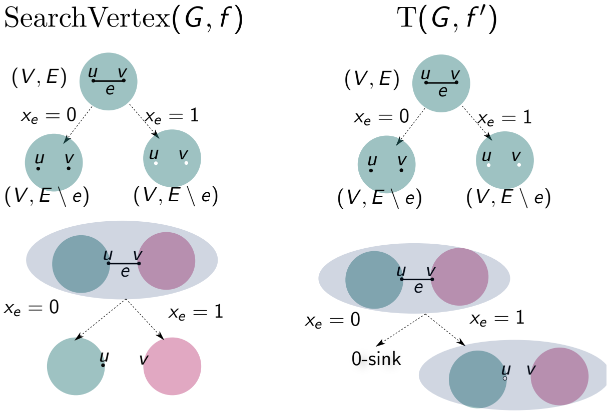
- Let D be a minimum-size 1-BP computing $\mathrm{SearchVertex}(G, f)$. Let s be a node of D computing $\mathrm{SearchVertex}(H, g)$ labeled by X_e. Then the children of s compute $\mathrm{SearchVertex}(H - e, g_0)$ and $\mathrm{SearchVertex}(H - e, g_1)$.
- **Structural lemma.** If e is a bridge of H and $H - e = C_1 \sqcup C_2$ for two connected components C_1 and C_2, then the children of s compute $\mathrm{SearchVertex}(C_1, g_0)$ and $\mathrm{SearchVertex}(C_2, g_1)$.

Structure of a 1-BP computing SearchVertex



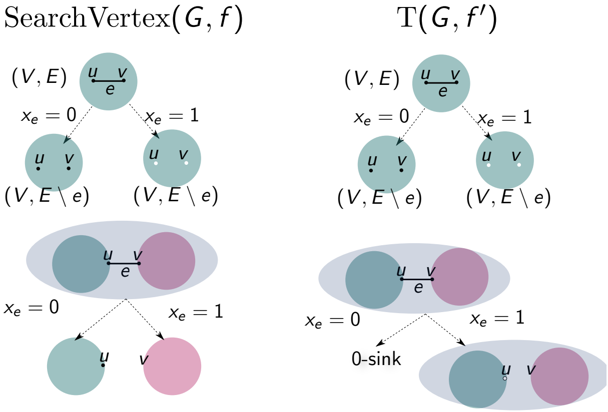
- Let D be a minimum-size 1-BP computing $\mathrm{SearchVertex}(G, f)$. Let s be a node of D computing $\mathrm{SearchVertex}(H, g)$ labeled by X_e. Then the children of s compute $\mathrm{SearchVertex}(H - e, g_0)$ and $\mathrm{SearchVertex}(H - e, g_1)$.
- **Structural lemma.** If e is a bridge of H and $H - e = C_1 \sqcup C_2$ for two connected components C_1 and C_2, then the children of s compute $\mathrm{SearchVertex}(C_1, g_0)$ and $\mathrm{SearchVertex}(C_2, g_1)$.

Transformation



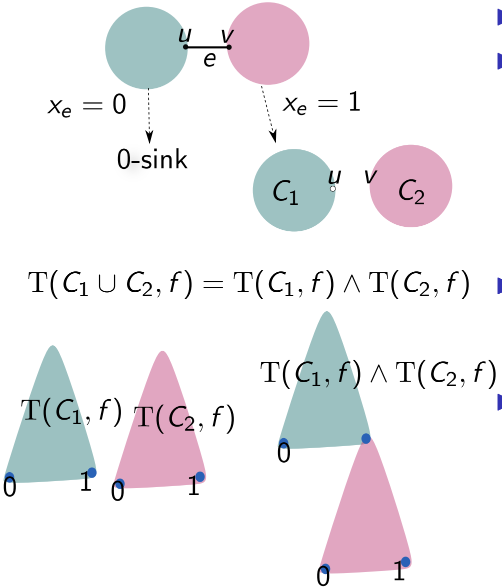
Let D be a 1-BP computing $\mathrm{SearchVertex}(G, f)$. By induction (from sinks) for every node $s \in D$ computing $\mathrm{SearchVertex}(H, c)$ and every $w \in V(H)$, we construct a node s computing $\mathrm{T}(H, c + \mathbf{1}_{\mathbf{w}})$.

Transformation



Let D be a 1-BP computing $\mathrm{SearchVertex}(G, f)$. By induction (from sinks) for every node $s \in D$ computing $\mathrm{SearchVertex}(H, c)$ and every $w \in V(H)$, we construct a node s computing $\mathrm{T}(H, c + \mathbf{1_w})$.

Transformation



- Nontrivial case: e is a bridge.
- By induction hypothesis we have node s_1 computing $\mathrm{T}(C_1, f)$ and s_2 computing $\mathrm{T}(C_1, f)$ but we need a node computing $\mathrm{T}(C_1 \cup C_2, f) = \mathrm{T}(C_1, f) \wedge \mathrm{T}(C_2, f)$.
- Make a copy of subprogram of s_1 where all edges to 1-sink redirected to s_2.
- The necessity to copy one of the subdiagrams results in a quasipolynomial ($S \mapsto S^{O(\log |V|)}$) blowup.

Our results

Main theorem. $S_{reg}(T(G,f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G,f)) = S$, then there exists a $1\text{-}\mathrm{BP}$ computing satisfiable Tseitin formula $T(G,f')$ of size $S^{O(\log|V|)}$.
2. $1\text{-}\mathrm{BP}(T(G,f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Minimal $1\text{-}\mathrm{BP}$ for $(\mathrm{T}(G,f))$ is OBDD (in every path variables appear in the same order).
   - $\mathrm{OBDD}(\mathrm{T}(G,f)) \geq 2^{\Omega(\mathrm{tw}(G))}$.

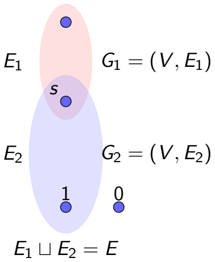
Our results

Main theorem. $S_{reg}(T(G, f) \geq 2^{\Omega(\mathrm{tw}(G)/\log|V|)}$.

Plan of the proof

1. If $S_{reg}(T(G, f)) = S$, then there exists a 1-BP computing satisfiable Tseitin formula $T(G, f')$ of size $S^{O(\log|V|)}$.
2. 1-$\mathrm{BP}(T(G, f')) \geq 2^{\Omega(\mathrm{tw}(G))}$
   - Minimal 1-BP for $(\mathrm{T}(G, f))$ is OBDD (in every path variables appear in the same order).
   - $\mathrm{OBDD}(\mathrm{T}(G, f)) \geq 2^{\Omega(\mathrm{tw}(G))}$.

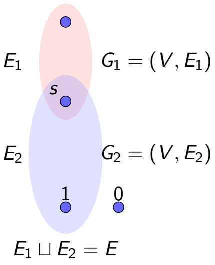
Number of acc. paths passing a node of 1-BP



- Let s computes $\mathrm{T}(G_2, c_2)$, where $G_2 = (V, E_2)$. Hence, there are exactly $\sharp\mathrm{T}(G_2, c_2)$ paths from s to 1-sink.
- Every path from the source to s is a sat. assignment of $\mathrm{T}(G_1, c_1)$, where $G_1 = (V, E_1)$. Hence, there are at most $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.
- In minimal OBDD all paths starts with E_1, hence all sat. assignments of $\mathrm{T}(G_1, c_1)$ can be realized. Hence there are exactly $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.

- In 1-BP: at most $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting passing s.
- In minimal OBDD: exactly $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting paths passing s.

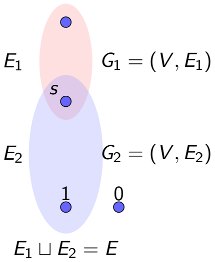
Number of acc. paths passing a node of 1-BP



- Let s computes $\mathrm{T}(G_2, c_2)$, where $G_2 = (V, E_2)$. Hence, there are exactly $\sharp\mathrm{T}(G_2, c_2)$ paths from s to 1-sink.
- Every path from the source to s is a sat. assignment of $\mathrm{T}(G_1, c_1)$, where $G_1 = (V, E_1)$. Hence, there are at most $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.
- In minimal OBDD all paths starts with E_1, hence all sat. assignments of $\mathrm{T}(G_1, c_1)$ can be realized. Hence there are exactly $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.

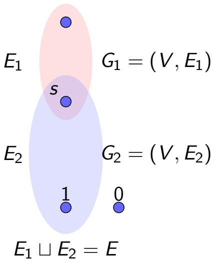
- In 1-BP: at most $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting passing s.
- In minimal OBDD: exactly $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting paths passing s.

Number of acc. paths passing a node of 1-BP



- Let s computes $\mathrm{T}(G_2, c_2)$, where $G_2 = (V, E_2)$. Hence, there are exactly $\sharp\mathrm{T}(G_2, c_2)$ paths from s to 1-sink.
- Every path from the source to s is a sat. assignment of $\mathrm{T}(G_1, c_1)$, where $G_1 = (V, E_1)$. Hence, there are at most $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.
- In minimal OBDD all paths starts with E_1, hence all sat. assignments of $\mathrm{T}(G_1, c_1)$ can be realized. Hence there are exactly $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.
- In 1-BP: at most $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting passing s.
- In minimal OBDD: exactly $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting paths passing s.

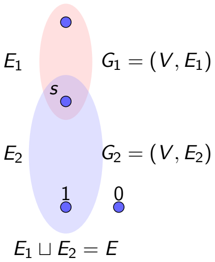
Number of acc. paths passing a node of 1-BP



- Let s computes $\mathrm{T}(G_2, c_2)$, where $G_2 = (V, E_2)$. Hence, there are exactly $\sharp\mathrm{T}(G_2, c_2)$ paths from s to 1-sink.
- Every path from the source to s is a sat. assignment of $\mathrm{T}(G_1, c_1)$, where $G_1 = (V, E_1)$. Hence, there are at most $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.
- In minimal OBDD all paths starts with E_1, hence all sat. assignments of $\mathrm{T}(G_1, c_1)$ can be realized. Hence there are exactly $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.

- In 1-BP: at most $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting passing s.
- In minimal OBDD: exactly $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting paths passing s.

Number of acc. paths passing a node of 1-BP



- Let s computes $\mathrm{T}(G_2, c_2)$, where $G_2 = (V, E_2)$. Hence, there are exactly $\sharp\mathrm{T}(G_2, c_2)$ paths from s to 1-sink.
- Every path from the source to s is a sat. assignment of $\mathrm{T}(G_1, c_1)$, where $G_1 = (V, E_1)$. Hence, there are at most $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.
- In minimal OBDD all paths starts with E_1, hence all sat. assignments of $\mathrm{T}(G_1, c_1)$ can be realized. Hence there are exactly $\sharp\mathrm{T}(G_1, c_1)$ paths from the source to s.

- In 1-BP: at most $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting passing s.
- In minimal OBDD: exactly $\sharp\mathrm{T}(G_1, c_1) \times \sharp\mathrm{T}(G_2, c_2)$ accepting paths passing s.

Optimal 1-BP computing a Tseitin formula is OBDD

Theorem. $1\text{-BP}(\mathrm{T}(G,c)) \geq \mathrm{OBDD}(\mathrm{T}(G,c))$.

- Let D be a minimal 1-BP computing $\mathrm{T}(G,c)$.
- Let a_s be the number of accepting paths passing s.
- For an accepting path p we denote by $\gamma(p) = \sum_{s \in p} \frac{1}{a_s}$.
- Let $\mathcal{P}$ be the set of accepting paths in D; $|\mathcal{P}| = \sharp\mathrm{T}(G,c)$.
- $|D| - 1 = \sum_{p \in \mathcal{P}} \gamma(p) \geq |\mathcal{P}| \min_{p \in \mathcal{P}} \gamma(p) = |\mathcal{P}|\gamma(p^*)$.
- Let D' be a minimal OBDD for $\mathrm{T}(G,c)$ in order corresponding p^*.
- For D' we define a'_s and $\gamma'(p)$. a'_s depends only on the distance from the source. Hence, $\gamma'(p)$ does not depend on accepting path. We know that $\gamma(p^*) \geq \gamma'(p^*)$.
- $|D| - 1 \geq |\mathcal{P}|\gamma(p^*) = \sharp\mathrm{T}(G,c)\gamma(p^*) \geq \sharp\mathrm{T}(G,c)\gamma'(p^*) = |D'| - 1$.

Optimal 1-BP computing a Tseitin formula is OBDD

Theorem. $1\text{-BP}(\mathrm{T}(G,c)) \geq \mathrm{OBDD}(\mathrm{T}(G,c))$.

- Let D be a minimal 1-BP computing $\mathrm{T}(G,c)$.
- Let a_s be the number of accepting paths passing s.
- For an accepting path p we denote by $\gamma(p) = \sum_{s\in p} \frac{1}{a_s}$.
- Let $\mathcal{P}$ be the set of accepting paths in D; $|\mathcal{P}| = \sharp\mathrm{T}(G,c)$.
- $|D| - 1 = \sum_{p\in\mathcal{P}} \gamma(p) \geq |\mathcal{P}| \min_{p\in\mathcal{P}} \gamma(p) = |\mathcal{P}|\gamma(p^*)$.
- Let D' be a minimal OBDD for $\mathrm{T}(G,c)$ in order corresponding p^*.
- For D' we define a'_s and $\gamma'(p)$. a'_s depends only on the distance from the source. Hence, $\gamma'(p)$ does not depend on accepting path. We know that $\gamma(p^*) \geq \gamma'(p^*)$.
- $|D| - 1 \geq |\mathcal{P}|\gamma(p^*) = \sharp\mathrm{T}(G,c)\gamma(p^*) \geq \sharp\mathrm{T}(G,c)\gamma'(p^*) = |D'| - 1$.

Optimal 1-BP computing a Tseitin formula is OBDD

Theorem. $1\text{-}\mathrm{BP}(\mathrm{T}(G,c)) \geq \mathrm{OBDD}(\mathrm{T}(G,c))$.

- Let D be a minimal 1-BP computing $\mathrm{T}(G,c)$.
- Let a_s be the number of accepting paths passing s.
- For an accepting path p we denote by $\gamma(p) = \sum_{s\in p} \frac{1}{a_s}$.
- Let $\mathcal{P}$ be the set of accepting paths in D; $|\mathcal{P}| = \sharp\mathrm{T}(G,c)$.
- $|D| - 1 = \sum_{p\in\mathcal{P}} \gamma(p) \geq |\mathcal{P}| \min_{p\in\mathcal{P}} \gamma(p) = |\mathcal{P}|\gamma(p^*)$.
- Let D' be a minimal OBDD for $\mathrm{T}(G,c)$ in order corresponding p^*.
- For D' we define a'_s and $\gamma'(p)$. a'_s depends only on the distance from the source. Hence, $\gamma'(p)$ does not depend on accepting path. We know that $\gamma(p^*) \geq \gamma'(p^*)$.
- $|D| - 1 \geq |\mathcal{P}|\gamma(p^*) = \sharp\mathrm{T}(G,c)\gamma(p^*) \geq \sharp\mathrm{T}(G,c)\gamma'(p^*) = |D'| - 1$.

Optimal 1-BP computing a Tseitin formula is OBDD

Theorem. $1\text{-}\mathrm{BP}(\mathrm{T}(G, c)) \geq \mathrm{OBDD}(\mathrm{T}(G, c))$.

- Let D be a minimal 1-BP computing $\mathrm{T}(G, c)$.
- Let a_s be the number of accepting paths passing s.
- For an accepting path p we denote by $\gamma(p) = \sum_{s \in p} \frac{1}{a_s}$.
- Let $\mathcal{P}$ be the set of accepting paths in D; $|\mathcal{P}| = \sharp\mathrm{T}(G, c)$.
- $|D| - 1 = \sum_{p \in \mathcal{P}} \gamma(p) \geq |\mathcal{P}| \min_{p \in \mathcal{P}} \gamma(p) = |\mathcal{P}|\gamma(p^*)$.
- Let D' be a minimal OBDD for $\mathrm{T}(G, c)$ in order corresponding p^*.
- For D' we define a'_s and $\gamma'(p)$. a'_s depends only on the distance from the source. Hence, $\gamma'(p)$ does not depend on accepting path. We know that $\gamma(p^*) \geq \gamma'(p^*)$.
- $|D| - 1 \geq |\mathcal{P}|\gamma(p^*) = \sharp\mathrm{T}(G, c)\gamma(p^*) \geq \sharp\mathrm{T}(G, c)\gamma'(p^*) = |D'| - 1$.

Optimal 1-BP computing a Tseitin formula is OBDD

Theorem. $1\text{-BP}(\mathrm{T}(G,c)) \geq \mathrm{OBDD}(\mathrm{T}(G,c))$.

- Let D be a minimal 1-BP computing $\mathrm{T}(G,c)$.
- Let a_s be the number of accepting paths passing s.
- For an accepting path p we denote by $\gamma(p) = \sum_{s\in p} \frac{1}{a_s}$.
- Let $\mathcal{P}$ be the set of accepting paths in D; $|\mathcal{P}| = \sharp\mathrm{T}(G,c)$.
- $|D| - 1 = \sum_{p\in\mathcal{P}} \gamma(p) \geq |\mathcal{P}| \min_{p\in\mathcal{P}} \gamma(p) = |\mathcal{P}|\gamma(p^*)$.
- Let D' be a minimal OBDD for $\mathrm{T}(G,c)$ in order corresponding p^*.
- For D' we define a'_s and $\gamma'(p)$. a'_s depends only on the distance from the source. Hence, $\gamma'(p)$ does not depend on accepting path. We know that $\gamma(p^*) \geq \gamma'(p^*)$.
- $|D| - 1 \geq |\mathcal{P}|\gamma(p^*) = \sharp\mathrm{T}(G,c)\gamma(p^*) \geq \sharp\mathrm{T}(G,c)\gamma'(p^*) = |D'| - 1$.

OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
  - Fix a spanning forest, take arbitrary values to all edges out of it. The value of edges from the spanning forest will be uniquely determined.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

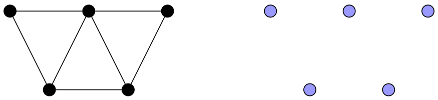
OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp \mathrm{T}(G,f)}{\sharp \mathrm{T}(G_1,f_1) \sharp \mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

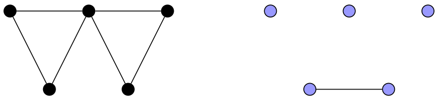
$\alpha_0 = 6 \quad \alpha_{min} = 6$



OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

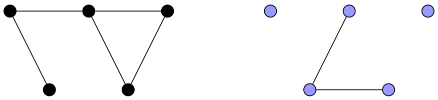
$\alpha_0 = 6 \quad \alpha_{min} = 5$



OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

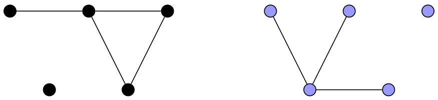
$\alpha_0 = 6 \quad \alpha_{min} = 4$



OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

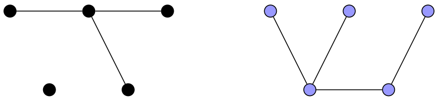
$\alpha_0 = 6 \quad \alpha_{min} = 4$



OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G $(\mathrm{compw}(G))$ is the minimum possible Bob's payout.

$\alpha_0 = 6 \quad \alpha_{min} = 3$



OBDD and component width

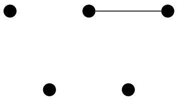
- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

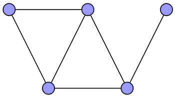
$\alpha_0 = 6 \quad \alpha_{min} = 3$

OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

$\alpha_0 = 6 \quad \alpha_{min} = 3$

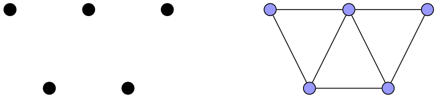




OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp\mathrm{T}(G,f)}{\sharp\mathrm{T}(G_1,f_1)\sharp\mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

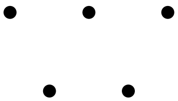
$\alpha_0 = 6 \quad \alpha_{min} = 3 \qquad \mathrm{payout} = 3$



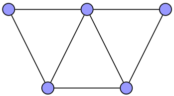
OBDD and component width

- The number of satisfying assignments of a satisfiable $\mathrm{T}(G, f)$ is $2^{|E|-|V|+\mathrm{cc}(G)}$.
- Consider a node s of a minimal OBDD D computing $\mathrm{T}(G, f)$. The number of nodes on level ℓ equals $\frac{\sharp \mathrm{T}(G,f)}{\sharp \mathrm{T}(G_1,f_1)\sharp \mathrm{T}(G_2,f_2)} = 2^{|V|+\mathrm{cc}(G)-\mathrm{cc}(G_1)-\mathrm{cc}(G_2)}$.
- Bob plays the following game: $G_1 = G$, G_2 is the empty graph on V. Every his move, Bob remove one edge from G_1 and add it to G_2. Bob calculates a value $\alpha = \mathrm{cc}(G_1) + \mathrm{cc}(G_2)$. Initially $\alpha_0 = |V| + \mathrm{cc}(G)$. Bob pays the maximal value of $\alpha_0 - \alpha$. The component width of G ($\mathrm{compw}(G)$) is the minimum possible Bob's payout.

$\alpha_0 = 6$ $\alpha_{min} = 3$



payout $= 3$



- **Proposition.** $|E|2^{\mathrm{compw}(G)} \geq \mathrm{OBDD}(\mathrm{T}(G, f)) \geq 2^{\mathrm{compw}(G)}$.
- **Theorem.** $\mathrm{pw}(G) + 1 \geq \mathrm{compw}(G) \geq \frac{1}{2}(\mathrm{tw}(G) - 1)$.

Open problems

- Is it possible to prove that $S_R(\mathrm{T}(G, c)) \geq 2^{\Omega(\mathrm{tw}(G))}$?
- Is it possible to prove a similar lower bound for unrestricted resolution?
- Is it possible to separate $\mathrm{Search}_{\mathrm{T}(G,c)}$ and $\mathrm{SearchVertex}_{G,c}$ for constant degree graphs?