Bad News

on

Combinatorial Lower Bounds

for the

Extension Complexity

of the

Spanning Tree Polytope

Kaveh Khoshkhah

Dirk Oliver Theis

Aussois Combinatorial Optimization Workshop
Jan 9-13, 2017
Outline

Extended Formulations for Spanning Tree: Status Quo

Lower Bounds for Extension Complexity 101

What We Did

Glimpse At the Details
Extended Formulations

LP formulations for Minimum Spanning Tree (complete graph)

“Classical” formulation:

\[
\begin{align*}
\min_x & \quad \sum_e c_e x_e \\
\text{subject to} & \quad \sum_e x_e = n - 1 \\
& \quad \sum_{e \subseteq S} x_e \leq |S| - 1, \quad \forall S \subseteq [n], \ |S| > 1 \\
& \quad x_e \geq 0, \quad \forall e \in \binom{[n]}{2}.
\end{align*}
\]

Size: \# ieqs = \Theta(2^n)

Martin’s “extended” formulation:

\[
\begin{align*}
\min_{x,z} & \quad \sum_e c_e x_e \\
\text{subject to} & \quad \ldots
\end{align*}
\]

Size: \# ieqs = \Theta(n^3)

\[ [n] := \{1, \ldots, n\} \]
Extension Complexity

Definition

Let $P$ be a polytope.

"Extension" of $P$: Polytope $Q$ with projection mapping $\pi: Q \rightarrow P$ (onto).

Linear Programming over $P$ reduces to Linear Programming over $Q$.

"Size" of extension: number of facets.

"Extension Complexity" of $P$: Smallest size of an extension of $P$.

Theorem (Martin '91)

Extension complexity of Spanning Tree polytope on complete graph with $n$ nodes is $O(n^3)$.

Theorem (Trivial Lower Bound)

Extension complexity of $P$ is at least the dimension of $P$.

Corollary

Extension complexity of Spanning Tree polytope on complete graph with $n$ nodes is $\Omega(n^2)$. 
Extension Complexity

Definition

- Let $P$ be a polytope.
Extension Complexity

Definition

- Let $P$ be a polytope.
- “Extension” of $P$: Polytope $Q$ w/ proj. mapping $\pi: Q \rightarrow P$ (onto)
Extension Complexity

Definition

- Let $P$ be a polytope.
- “Extension” of $P$: Polytope $Q$ w/ proj. mapping $\pi: Q \rightarrow P$ (onto)
  - Linear Programming over $P$ reduces to Linear Programming over $Q$
Extension Complexity

Definition

- Let $P$ be a polytope.
- "Extension" of $P$: Polytope $Q$ w/ proj. mapping $\pi: Q \rightarrow P$ (onto)
  - $\nabla$ Linear Programming over $P$ reduces to Linear Programming over $Q$
- "Size" of extension: number of facets
Extension Complexity

Definition

- Let $P$ be a polytope.
- "Extension" of $P$: Polytope $Q$ w/ proj. mapping $\pi: Q \rightarrow P$ (onto)
  - Linear Programming over $P$ reduces to Linear Programming over $Q$
- "Size" of extension: number of facets
- "Extension Complexity" of $P$: Smallest size of an extension of $P$
Extension Complexity

Definition

- Let $P$ be a polytope.
- “Extension” of $P$: Polytope $Q$ w/ proj. mapping $\pi: Q \to P$ (onto)
  - Linear Programming over $P$ reduces to Linear Programming over $Q$
- “Size” of extension: number of facets
- “Extension Complexity” of $P$: Smallest size of an extension of $P$

Theorem (Martin ’91)

$Extension \ complexity \ of \ Spanning \ Tree \ polytope \ on \ complete \ graph \ w/ \ n \ nodes \quad O(n^3)$
Extension Complexity

Definition

- Let $P$ be a polytope.
- "Extension" of $P$: Polytope $Q$ w/ proj. mapping $\pi: Q \rightarrow P$ (onto)
  - Linear Programming over $P$ reduces to Linear Programming over $Q$
- "Size" of extension: number of facets
- "Extension Complexity" of $P$: Smallest size of an extension of $P$

Theorem (Martin '91)

Extension complexity of Spanning Tree polytope on complete graph w/ $n$ nodes $O(n^3)$

Theorem (Trivial Lower Bound)

Extension complexity of $P$ is at least dimension of $P$. 
Extension Complexity

Definition

- Let $P$ be a polytope.
- “Extension” of $P$: Polytope $Q$ w/ proj. mapping $\pi: Q \rightarrow P$ (onto)
  - Linear Programming over $P$ reduces to Linear Programming over $Q$
- “Size” of extension: number of facets
- “Extension Complexity” of $P$: Smallest size of an extension of $P$

Theorem (Martin ’91)

Extension complexity of Spanning Tree polytope on complete graph w/ $n$ nodes $O(n^3)$

Theorem (Trivial Lower Bound)

Extension complexity of $P$ is at least dimension of $P$.

Corollary

Extension complexity of Spanning Tree polytope on complete graph w/ $n$ nodes $\Omega(n^2)$
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3) \]

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3) \]

Progress on Upper Bounds
Only for special graphs (not complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity(spanning\_tree}_n) = O(n^3) \]

Progress on Upper Bounds
Only for special graphs (not complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity}(\text{spanning tree}_n) = O(n^3) \]

**Progress on Upper Bounds**

Only for special graphs (*not* complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity}(\text{spanning tree}_n) = O(n^3) \]

Progress on Upper Bounds
Only for special graphs (*not* complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity}(\text{spanning tree}_n) = O(n^3) \]

Progress on Upper Bounds
Only for special graphs (not complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3) \]

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3) \]

**Progress on Upper Bounds**

Only for special graphs (*not* complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

**Progress on Lower Bounds**
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3) \]

**Progress on Upper Bounds**

Only for special graphs (*not* complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

**Progress on Lower Bounds**
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity(spanning tree}_n) = O(n^3) \]

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity}(\text{spanning tree}_n) = O(n^3) \]

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

$$\Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3)$$

Progress on Upper Bounds

Only for special graphs (*not* complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity} (\text{spanning\_tree}_n) = O(n^3) \]

**Progress on Upper Bounds**

Only for special graphs (*not* complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

**Progress on Lower Bounds**
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity}(\text{spanning tree}_n) = O(n^3) \]

Progress on Upper Bounds

Only for special graphs \textit{(not complete!)}, e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3) \]

**Progress on Upper Bounds**
Only for special graphs (*not* complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

**Progress on Lower Bounds**
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity(spanning tree}_n) = O(n^3) \]

**Progress on Upper Bounds**

Only for special graphs (not complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

**Progress on Lower Bounds**
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3) \]

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

$$\Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3)$$

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity}(\text{spanning tree}_n) = O(n^3) \]

Progress on Upper Bounds
Only for special graphs \textit{(not complete!)}, e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

$$\Omega(n^2) = \text{extension}_\text{complexity}(\text{spanning}_\text{tree}_n) = O(n^3)$$

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension\_complexity}(\text{spanning\_tree}_n) = O(n^3) \]

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity}(\text{spanning tree}_n) = O(n^3) \]

**Progress on Upper Bounds**

Only for special graphs (*not* complete!), e.g.,

- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

**Progress on Lower Bounds**
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension}\_\text{complexity}(\text{spanning}\_\text{tree}_n) = O(n^3) \]

Progress on Upper Bounds
Only for special graphs (not complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Extension Complexity of Spanning Tree Polytope

\[ \Omega(n^2) = \text{extension complexity}(\text{spanning tree}_n) = O(n^3) \]

Progress on Upper Bounds

Only for special graphs (not complete!), e.g.,
- Fiorini-Huynh-Pashkovich-Joret 2017
- Kolman-Koutecký-Tiwary 2016
- Williams 2002

Progress on Lower Bounds
Long-Term Goals:

- Improving upper bound to anything $n^3$ would be a breakthrough,
- e.g., extended formulation of size $n^3/\log \log(n)$;
- Improving lower bound to anything $n^2$ would be a breakthrough,
- e.g., $n^2 \cdot \log \log(n)$.

This Talk's Results:

- Negative results for the lower bounds
- The convenient "combinatorial" lower bounds give:
  - "Fooling set" bound: $O(n^2)$
  - "Nondeterministic communication complexity" bound: $O(n^2 \log n)$
- "Double negative" results: upper bounds on lower bounds.
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,

This Talk’s Results:

- Negative results for the lower bounds
- The convenient “combinatorial” lower bounds give:
  - “Fooling set” bound: $O(n^2)$
  - “Nondeterministic communication complexity” bound: $O(n^2 \log n)$

- “Double negative” results: upper bounds on lower bounds.
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,
  - e.g., extended formulation of size $n^3 / \log \log(n)$;
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,
  - e.g., extended formulation of size $n^3/\log\log(n)$;
- Improving lower bound to anything $\gg n^2$ would be a breakthrough,
Long-Term Goals:

- Improving upper bound to anything \( \ll n^3 \) would be a breakthrough,
  - e.g., extended formulation of size \( n^3 / \log \log(n) \);
- Improving lower bound to anything \( \gg n^2 \) would be a breakthrough,
  - e.g., \( n^2 \cdot \log \log(n) \).
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,
  - e.g., extended formulation of size $n^3 / \log \log(n)$;
- Improving lower bound to anything $\gg n^2$ would be a breakthrough,
  - e.g., $n^2 \cdot \log \log(n)$.

This Talk’s Results:
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,
  - e.g., extended formulation of size $n^3 / \log \log(n)$;
- Improving lower bound to anything $\gg n^2$ would be a breakthrough,
  - e.g., $n^2 \cdot \log \log(n)$.

This Talk’s Results:

- Negative results for the lower bounds
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,
  - e.g., extended formulation of size $n^3 / \log \log(n)$;
- Improving lower bound to anything $\gg n^2$ would be a breakthrough,
  - e.g., $n^2 \cdot \log \log(n)$.

This Talk’s Results:

- Negative results for the lower bounds
- The convenient “combinatorial” lower bounds give:
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,
  - e.g., extended formulation of size $n^3 / \log \log(n)$;
- Improving lower bound to anything $\gg n^2$ would be a breakthrough,
  - e.g., $n^2 \cdot \log \log(n)$.

This Talk’s Results:

- Negative results for the lower bounds
- The convenient “combinatorial” lower bounds give:
  - “Fooling set” bound: $O(n^2)$
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,
  - e.g., extended formulation of size $n^3 / \log \log(n)$;
- Improving lower bound to anything $\gg n^2$ would be a breakthrough,
  - e.g., $n^2 \cdot \log \log(n)$.

This Talk’s Results:

- Negative results for the lower bounds
- The convenient “combinatorial” lower bounds give:
  - “Fooling set” bound: $O(n^2)$
  - “Nondeterministic communication complexity” bound: $O(n^2 \log n)$
Long-Term Goals:

- Improving upper bound to anything $\ll n^3$ would be a breakthrough,
  - e.g., extended formulation of size $n^3 / \log \log(n)$;
- Improving lower bound to anything $\gg n^2$ would be a breakthrough,
  - e.g., $n^2 \cdot \log \log(n)$.

This Talk’s Results:

- Negative results for the lower bounds
- The convenient “combinatorial” lower bounds give:
  - “Fooling set” bound: $O(n^2)$
  - “Nondeterministic communication complexity” bound: $O(n^2 \log n)$
- “Double negative” results: upper bounds on lower bounds.
Outline

Extended Formulations for Spanning Tree: Status Quo

Lower Bounds for Extension Complexity 101

What We Did

Glimpse At the Details
Combinatorial Lower Bounds

Some Combinatorial Bounds

fooling set bound

\[ \leq \text{fractional rectangle covering bound} \]

\[ \leq 2^{\text{nondeterministic communication complexity}} \]

\[ \leq \text{extension complexity} \]

Successful for

TSP

Max-Cut

Bipartite Matching

...
Combinatorial Lower Bounds

Some Combinatorial Bounds

fooling set bound
\[ \leq \text{fractional rectangle covering bound} \leq 2^{\text{nondeterministic communication complexity}} \leq \text{extension complexity} \]

▶ “Combinatorial” means: consider vertex-facet incidences only.

Successful for

- TSP
- Max-Cut
- Bipartite Matching

...
Combinatorial Lower Bounds

Some Combinatorial Bounds

- Fooling set bound
  \[ \leq \text{fractional rectangle covering bound} \]
  \[ \leq 2^{\text{nondeterministic communication complexity}} \]
  \[ \leq \text{extension complexity} \]

- "Combinatorial" means: consider vertex-facet incidences only.
  - As opposed to: consider distance of vertex from facet hyperplane (non-combinatorial).

Successful for

- TSP
- Max-Cut
- Bipartite Matching
  . . .
Combinatorial Lower Bounds

Some Combinatorial Bounds

- fooling set bound
  \[ \leq \text{fractional rectangle covering bound} \]
  \[ \leq 2^{\text{nondeterministic communication complexity}} \]
  \[ \leq \text{extension complexity} \]

- “Combinatorial” means: consider vertex-facet incidences only.
  - As opposed to: consider distance of vertex from facet hyperplane (non-combinatorial).
- Nondeterministic communication complexity bound is the strongest combinatorial bound.

Successful for

- TSP
- Max-Cut
- Bipartite Matching
Combinatorial Lower Bounds
Some Combinatorial Bounds

fooling set bound
\[ \leq \text{fractional rectangle covering bound} \]
\[ \leq 2^{\text{nondeterministic communication complexity}} \]
\[ \leq \text{extension complexity} \]

▶ “Combinatorial” means: consider vertex-facet incidences only.
▶ As opposed to: consider distance of vertex from facet hyperplane (non-combinatorial).
▶ Nondeterministic communication complexity bound is the strongest combinatorial bound.
▶ “Nondet. Comm. Complexity bound” = \(2^{\text{nondet. comm. complexity}}\)

Successful for

- TSP
- Max-Cut
- Bipartite Matching
...
Combinatorial Lower Bounds

Some Combinatorial Bounds

fooling set bound

\[ \leq \text{fractional rectangle covering bound} \]

\[ \leq 2^{\text{nondeterministic communication complexity}} \]

\[ \leq \text{extension complexity} \]

“Combinatorial” means: consider vertex-facet incidences only.

- As opposed to: consider distance of vertex from facet hyperplane (non-combinatorial).

Nondeterministic communication complexity bound is the strongest combinatorial bound.

- “Nondet. Comm. Complexity bound” = \(2^{\text{nondet. comm. complexity}}\)

Successful for

- TSP
Combinatorial Lower Bounds

Some Combinatorial Bounds

- fooling set bound
  \[ \leq \text{fractional rectangle covering bound} \leq 2^{\text{nondeterministic communication complexity}} \leq \text{extension complexity} \]

- “Combinatorial” means: consider vertex-facet incidences only.
  - As opposed to: consider distance of vertex from facet hyperplane (non-combinatorial).
- Nondeterministic communication complexity bound is the strongest combinatorial bound.
  - “Nondet. Comm. Complexity bound” = \(2^{\text{nondet. comm. complexity}}\)

Successful for
- TSP
- Max-Cut
Combinatorial Lower Bounds

Some Combinatorial Bounds

fooling set bound

≤ fractional rectangle covering bound

≤ $2^{\text{nondeterministic communication complexity}}$

≤ extension complexity

“Combinatorial” means: consider vertex-facet incidences only.

As opposed to: consider distance of vertex from facet hyperplane (non-combinatorial).

Nondeterministic communication complexity bound is the strongest combinatorial bound.

“Nondet. Comm. Complexity bound” = $2^{\text{nondet. comm. complexity}}$

Successful for

- TSP
- Max-Cut
- Bipartite Matching
Combinatorial Lower Bounds

Some Combinatorial Bounds

fooling set bound

\[ \leq \text{fractional rectangle covering bound} \]

\[ \leq 2^{\text{nondeterministic communication complexity}} \]

\[ \leq \text{extension complexity} \]

▶ “Combinatorial” means: consider vertex-facet incidences only.
  ▶ As opposed to: consider distance of vertex from facet hyperplane (non-combinatorial).

▶ Nondeterministic communication complexity bound is the strongest combinatorial bound.

▶ “Nondet. Comm. Complexity bound” = \(2^{\text{nondet. comm. complexity}}\)

Successful for

▶ TSP
▶ Max-Cut
▶ Bipartite Matching
▶ …
Nondeterministic Communication Protocols

Diagram for nondeterministic communication protocols computing Boolean function $f(.,.,.)$

Cost of the protocol: 
# bits communicated (worst-case)

“Nondeterministic”: No false positive outputs;
If $f(S, T) = \text{true}$, $\exists$ certificate s.t. output is true
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol

**Link to Extensions of Polytopes**

Fix polytope $P$
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol

Link to Extensions of Polytopes
Fix polytope $P$
  ▶ Alice gets a facet $S$
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol

**Link to Extensions of Polytopes**

Fix polytope $P$

- Alice gets a facet $S$
- Bob gets a vertex $T$
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol

**Link to Extensions of Polytopes**

Fix polytope $P$

- Alice gets a facet $S$
- Bob gets a vertex $T$

\[ f(S, T) = \begin{cases} 
  \text{true,} & \text{if vertex is off facet;} \\
  \text{false,} & \text{if vertex on facet.} 
\end{cases} \]
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol

Link to Extensions of Polytopes

Fix polytope $P$

- Alice gets a facet $S$
- Bob gets a vertex $T$

\[
f(S, T) = \begin{cases} 
    \text{true}, & \text{if vertex is off facet;} \\
    \text{false}, & \text{if vertex on facet.}
\end{cases}
\]

\[
2^{\text{ndCC}(f)} \leq \text{extension complexity of } P
\]
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol

**Link to Extensions of Polytopes**

Fix polytope $P$

- Alice gets a facet $S$
- Bob gets a vertex $T$

$$f(S, T) = \begin{cases} 
\text{true,} & \text{if vertex is off facet;} \\
\text{false,} & \text{if vertex on facet.}
\end{cases}$$

$$2^{\text{ndCC}(f)} \leq \text{extension complexity of } P$$

**Careful!**
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol

**Link to Extensions of Polytopes**

Fix polytope $P$

- Alice gets a facet $S$
- Bob gets a vertex $T$

$f(S, T) = \begin{cases} 
\text{true,} & \text{if vertex is off facet;} \\
\text{false,} & \text{if vertex on facet.} 
\end{cases}$

$2^{\text{ndCC}(f)} \leq \text{extension complexity of } P$

**Careful!**

- “Nondeterministic Communication Complexity”: $\# \text{ bits (in best protocol)}$
“Nondeterministic Communication Complexity” of $f$: Cost of best protocol

Link to Extensions of Polytopes

Fix polytope $P$

- Alice gets a facet $S$
- Bob gets a vertex $T$

$$f(S, T) = \begin{cases} 
  \text{true}, & \text{if vertex is off facet;} \\
  \text{false}, & \text{if vertex on facet.}
\end{cases}$$

$$2^{\text{ndCC}(f)} \leq \text{extension complexity of } P$$

Careful!

“Nondeterministic Communication Complexity”: \# bits (in best protocol)

“Nondeterministic Communication Complexity bound” = \[2^{\text{nondet. comm. complexity}}\]
Back to Spanning Tree:

Polytope:

\[ P := \text{spanning tree polytope} \]

Vertices = trees

Facets:

\[ \sum_{e \subset S} x_e \leq |S| - 1 \quad \forall S \subset [n], |S| > 1 \]

Ignore \( x \geq 0 \) inequalities.

Communication complexity part

Alice gets set \( S \) (facet)

Bob gets a tree \( T \) (vertex)

\[ f(S, T) = \begin{cases} 
\text{true}, & \text{if } S \text{ disconnected in } T \\
\text{false}, & \text{if } S \text{ connected in } T 
\end{cases} \]
Back to Spanning Tree:

Polytope:

\[ P \] := spanning tree polytope

Vertices \( \hat{=} \) trees

Facets:

\[
\sum_{e \subset S} x_e \leq |S| - 1 \quad \forall S \subset \{1, \ldots, n\}, |S| > 1
\]

Ignore \( x \geq 0 \) inequalities.

Communication complexity part

Alice gets set \( S \) (facet)

Bob gets a tree \( T \) (vertex)

\[ f(S, T) = \begin{cases} 
\text{true}, & \text{if } S \text{ disconnected in } T; \\
\text{false}, & \text{if } S \text{ connected in } T.
\end{cases} \]
Back to Spanning Tree:

Polytope:

- $P :=$ spanning tree polytope
Back to Spanning Tree:

Polytope:
- $P := \text{spanning tree polytope}$
- Vertices $\hat{=} \text{trees}$
Back to Spanning Tree:

Polytope:
- $P :=$ spanning tree polytope
- Vertices $\hat{=} \text{trees}$
- Facets:

$$\sum_{e \subset S} x_e \leq |S| - 1$$

$\forall S \subset \{n\}, |S| > 1$
Back to Spanning Tree:

Polytope:
- \( P := \) spanning tree polytope
- Vertices \( \sim \) trees
- Facets:
  \[
  \sum_{e \subset S} x_e \leq |S| - 1
  \forall S \subset [n], \ |S| > 1
  \]
- Ignore \( x \geq 0 \) inequalities.
Back to Spanning Tree:

Polytope:

- $P :=$ spanning tree polytope
- Vertices $\hat{=} \text{trees}$
- Facets:

$$\sum_{e \subseteq S} x_e \leq |S| - 1$$

$\forall S \subset [n], |S| > 1$

- Ignore $x \geq 0$ inequalities.

Communication complexity part

- Alice gets set $S$ (facet)
- Bob gets a tree $T$ (vertex)
- $f(S, T) = \begin{cases} 
  \text{true} , & \text{if } S \text{ disconnected in } T \\
  \text{false} , & \text{if } S \text{ connected in } T 
\end{cases}$
Back to Spanning Tree:

Polytope:
- $P :=$ spanning tree polytope
- Vertices $\cong$ trees
- Facets:
  \[
  \sum_{e \subset S} x_e \leq |S| - 1
  \]
  \[
  \forall S \subset [n], \ |S| > 1
  \]
- Ignore $x \geq 0$ inequalities.

Communication complexity part
- Alice gets set $S$ (facet)
- Bob gets a tree $T$ (vertex)
  \[
  f(S, T) = \begin{cases} 
  \text{true} & \text{if } S \text{ disconnected in } T \\
  \text{false} & \text{if } S \text{ connected in } T
  \end{cases}
  \]
Back to Spanning Tree:

Polytope:
- $P :=$ spanning tree polytope
- Vertices $\hat{=}$ trees
- Facets:

$$\sum_{e \subset S} x_e \leq |S| - 1$$

for all $S \subset [n]$, $|S| > 1$

- Ignore $x \geq 0$ inequalities.

Communication complexity part
- Alice gets set $S$ (facet)
- Bob gets a tree $T$ (vertex)
Back to Spanning Tree:

Polytope:
- $P :=$ spanning tree polytope
- Vertices $\hat{=} \text{trees}$
- Facets:
  \[
  \sum_{e \subset S} x_e \leq |S| - 1 \\
  \forall S \subset [n], \ |S| > 1
  \]
- Ignore $x \geq 0$ inequalities.

Communication complexity part
- Alice gets set $S$ (facet)
- Bob gets a tree $T$ (vertex)
- $f(S, T) =
  \begin{cases} 
  \text{true,} & \text{if } S \text{ disconnected in } T; \\
  \text{false,} & \text{if } S \text{ connected in } T.
  \end{cases}
Outline

Extended Formulations for Spanning Tree: Status Quo

Lower Bounds for Extension Complexity 101

What We Did

Glimpse At the Details
Our Results

Theorem

\textit{Fooling set bound} = O(n^2)
Our Results

Theorem

Fooling set bound $= \mathcal{O}(n^2)$

arXiv:1701.00350
Our Results

**Theorem**

*Fooling set bound* \(= O(n^2)\)


**Theorem**

*Nondet. communication complexity bound* \(= O(n^2 \log n)\)
Our Results

Theorem

Fooling set bound $= O(n^2)$

\[ \text{arXiv:1701.00350} \]

Theorem

Nondet. communication complexity bound $= O(n^2 \log n)$

Previously best bounds:
Our Results

Theorem
Fooling set bound = $O(n^2)$

arXiv:1701.00350

Theorem
Nondet. communication complexity bound = $O(n^2 \log n)$

Previously best bounds:

- $O(n^{8/3} \log n)$ for fractional rectangle covering bound (Weltge ’15)
Our Results

Theorem

\textit{Fooling set bound} = \(O(n^2)\)

\(\text{arXiv:1701.00350}\)

Theorem

\textit{Nondet. communication complexity bound} = \(O(n^2 \log n)\)

Previously best bounds:

\begin{itemize}
  \item \(O(n^{8/3} \log n)\) for \textit{fractional} rectangle covering bound (Weltge '15)
  \item \(O(n^3)\) for nondet. communication complexity bound (trivial)
\end{itemize}
Diagram for nondeterministic communication protocols computing Boolean function $f(\ldots)$

Cost of the protocol: # bits communicated (worst-case)

How to prove an upper bound for Nondet. CC
How to prove an upper bound for Nondet. CC

- Alice has set $S$

Diagram for nondeterministic communication protocols computing Boolean function $f(\ldots)$

Cost of the protocol: # bits communicated (worst-case)
Diagram for nondeterministic communication protocols computing Boolean function \( f(.,.) \)

Cost of the protocol: # bits communicated (worst-case)

How to prove an upper bound for Nondet. CC

- Alice has set \( S \)
- Bob has tree \( T \)
How to prove an upper bound for Nondet. CC

- Alice has set $S$
- Bob has tree $T$
- $f(S, T) = \text{whether } S \text{ is disconnected in } T$
Diagram for nondeterministic communication protocols computing Boolean function $f(\ldots)$

Cost of the protocol: 
\# bits communicated (worst-case)

How to prove an upper bound for Nondet. CC

- Alice has set $S$
- Bob has tree $T$
- $f(S, T) =$ whether $S$ is disconnected in $T$
- Need: Protocol with short certificates
Alice has a set $S$, Bob has a tree $T$, They want to decide if $S$ is disconnected in $T$. The "obvious" certificate Protocol:

Prover sends $(a, b, x) \in \mathbb{Z}_n^3$

Alice: Check "$a, b \in S$, $x \not\in S$"? Output answer

Bob: Check "$a \dashv x \dashv b$ in $T$" (i.e., $x$ on path in $T$ between $a$ and $b$) Output answer

That works! Communication: $\log_2(n^3)$ bits $\Rightarrow O(n^3)$ upper bound
- Alice has a set $S$,
- Bob has a tree $T$, 

The "obvious" certificate Protocol:
- Prover sends $(a, b, x) \in [n]^3$ 
- Alice: Check "$a, b \in S, x \not\in S$" 
- Output answer 
- Bob: Check "$a \dashv x \dashv b$ in $T$" (i.e., $x$ on path in $T$ between $a$ and $b$) 
- Output answer 

That works! 

Communication: $\log_2(3^n)$ bits $\Rightarrow O(n^3)$ upper bound
- Alice has a set $S$,
- Bob has a tree $T$,
- They want to decide if $S$ is disconnected in $T$. 

**The "obvious" certificate Protocol:**
- Prover sends $(a, b, x) \in \mathbb{N}^3$.
- Alice: Check "$a, b \in S, x /\in S$"?
- Output answer.
- Bob: Check "$a \triangleright x \triangleright b$ in $T$" (i.e., $x$ on path in $T$ between $a$ and $b$).
- Output answer.

"That works!"

Communication: $\log_2(n^3) \Rightarrow O(n^3)$ upper bound.
Alice has a set $S$, Bob has a tree $T$, They want to decide if $S$ is disconnected in $T$.

The “obvious” certificate

Protocol:

- Prover sends $(a, b, x) \in [n]^3$
- Alice: Check "$a, b \in S, x \not\in S$"?
  Output answer
- Bob: Check "$a \dashv x \dashv b$ in $T$" (i.e., $x$ on path in $T$ between $a$ and $b$)
  Output answer
- That works!
- Communication: $\log_2 (n^3)$ bits $\Rightarrow O(n^3)$ upper bound
- Alice has a set $S$,
- Bob has a tree $T$,
- They want to decide if $S$ is disconnected in $T$.

**The “obvious” certificate**

**Protocol:**
- Prover sends $(a, b, x) \in [n]^3$
Alice has a set $S$, Bob has a tree $T$, they want to decide if $S$ is disconnected in $T$.

The “obvious” certificate

Protocol:
- Prover sends $(a, b, x) \in [n]^3$
- Alice:
  - Check "$a, b \in S, x \not\in S$"?
  - Output answer
- Bob:
  - Check "$a \triangleright x \triangleright b$ in $T$" (i.e., $x$ on path in $T$ between $a$ and $b$)
  - Output answer
- That works!
- Communication: $\log_2(n^3)$ bits $\Rightarrow O(n^3)$ upper bound
Alice has a set $S$, 
Bob has a tree $T$, 
They want to decide if $S$ is disconnected in $T$.

The “obvious” certificate

Protocol:
- Prover sends $(a, b, \chi) \in [n]^3$
- Alice:
  - Check “$a, b \in S$, $\chi \notin S$”?
Alice has a set $S$,
Bob has a tree $T$,
They want to decide if $S$ is disconnected in $T$.

The “obvious” certificate

Protocol:
- Prover sends $(a, b, x) \in [n]^3$
- Alice:
  - Check “$a, b \in S, x \notin S$”?
  - Output answer
- That works!

Communication: $\log_2 (n^3)$ bits $\Rightarrow O(n^3)$ upper bound
Alice has a set $S$,
Bob has a tree $T$,
They want to decide if $S$ is disconnected in $T$.

The “obvious” certificate

Protocol:

- Prover sends $(a, b, x) \in [n]^3$
- Alice:
  - Check “$a, b \in S$, $x \notin S$”?  
  - Output answer
- Bob:

\[ \xrightarrow{\text{communication}} \log_2(n^3) \text{ bits} \Rightarrow O(n^3) \text{ upper bound} \]
- Alice has a set $S$,
- Bob has a tree $T$,
- They want to decide if $S$ is disconnected in $T$.

### The “obvious” certificate

**Protocol:**

- **Prover** sends $(a, b, x) \in [n]^3$
- **Alice:**
  - Check “$a, b \in S, x \notin S$”?
  - Output answer
- **Bob:**
  - Check “$a \leadsto x \leadsto b$ in $T$?”
    (i.e., $x$ on path in $T$ between $a$ and $b$)

SSL communication: $\log_2(n^3) \Rightarrow O(n^3)$ upper bound
Alice has a set $S$,
Bob has a tree $T$,
They want to decide if $S$ is disconnected in $T$.

The “obvious” certificate

Protocol:
- Prover sends $(a, b, x) \in [n]^3$
- Alice:
  - Check “$a, b \in S, x \notin S$”? 
  - Output answer
- Bob:
  - Check “$a \leadsto x \leadsto b$ in $T$?”
    (i.e., $x$ on path in $T$ between $a$ and $b$)
  - Output answer

That works!
Communication: $\log_2(n^3)$ bits $\Rightarrow O(n^3)$ upper bound
Alice has a set $S$,
Bob has a tree $T$,
They want to decide if $S$ is disconnected in $T$.

The “obvious” certificate

Protocol:

- Prover sends $(a, b, x) \in [n]^3$
- Alice:
  - Check “$a, b \in S, x \notin S$”?  
  - Output answer
- Bob:
  - Check “$a \leadsto x \leadsto b$ in $T$?”  
    (i.e., $x$ on path in $T$ between $a$ and $b$)  
  - Output answer

That works!
Alice has a set \( S \),
Bob has a tree \( T \),
They want to decide if \( S \) is disconnected in \( T \).

The “obvious” certificate

Protocol:
- Prover sends \((a, b, x) \in [n]^3\)
- Alice:
  - Check “\(a, b \in S, x \notin S\)”?
  - Output answer
- Bob:
  - Check “\(a \rightsquigarrow x \rightsquigarrow b \) in \( T \)”
    (i.e., \( x \) on path in \( T \) between \( a \) and \( b \))
  - Output answer

That works!
Communication: \( \log_2(n^3) \) bits \( \sim O(n^3) \) upper bound
Our Approach

Protocol/Proof Ingredients

Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?

There are many possible choices for \((a, b, x)\)

Maybe, if chosen smartly, . . .

What works is this: Prover sends

\[a \quad b \quad h(a, b, x)\]

\(h: \{n\}^3 \rightarrow \{\ell\} \) with \(\ell = \log n + O(1)\).
Our Approach

Protocol/Proof Ingredients

- Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?
Our Approach

Protocol/Proof Ingredients

- Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?
- There are many possible choices for \((a, b, x)\)
Our Approach

Protocol/Proof Ingredients

- Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?
- There are many possible choices for \((a, b, x)\)
- Maybe, if chosen smartly, …
Our Approach

Protocol/Proof Ingredients

- Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?
- There are many possible choices for \((a, b, x)\)
- Maybe, if chosen smartly, . . .
- What works is this: Prover sends

\[ h(a, b, x) : n^3 \rightarrow \ell \text{ with } \ell = \log n + O(1). \]
Our Approach

Protocol/Proof Ingredients

- Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?
- There are many possible choices for \((a, b, x)\)
- Maybe, if chosen smartly, . . .
- What works is this: Prover sends
  - \(a\)
Our Approach

Protocol/Proof Ingredients

- Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?
- There are many possible choices for \((a, b, x)\)
- Maybe, if chosen smartly, . . .
- What works is this: Prover sends
  - \(a\)
  - \(b\)
Our Approach

Protocol/Proof Ingredients

- Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?
- There are many possible choices for \((a, b, x)\)
- Maybe, if chosen smartly, . . .
- What works is this: Prover sends
  - \(a\)
  - \(b\)
  - \(h(a, b, x)\)
Our Approach

Protocol/Proof Ingredients

- Basic idea: “compress” \((a, b, x)\). Maybe send some kind of a hash key?
- There are many possible choices for \((a, b, x)\)
- Maybe, if chosen smartly, . . .
- What works is this: Prover sends
  - \(a\)
  - \(b\)
  - \(h(a, b, x)\)
- \(h: [n]^3 \rightarrow [\ell] \) with \(\ell = \log n + O(1)\).
Outline

Extended Formulations for Spanning Tree: Status Quo

Lower Bounds for Extension Complexity 101

What We Did

Glimpse At the Details
Let’s say \((a, b, x)\) w/ \(a, b \in S, x \notin S, a \bowtie x \bowtie b\) in \(T\) is a “witness” for “\(f(S, T) = \text{true}\)”. 

Triangle Lemma \((\text{arXiv:1701.00350})\)

Let \(c \in S\). If \((a, b, x)\) is a witness, then so is at least one of \((a, c, x), (c, b, x)\).
A Glimpse at the Details (1)

- Let's say \((a, b, x)\) w/ \(a, b \in S, x \notin S, a \sim x \sim b\) in \(T\) is a "witness" for "\(f(S, T) = true\)".

**Triangle Lemma (arXiv:1701.00350)**

Let \(c \in S\). If \((a, b, x)\) is a witness, then so is at least one of \((a, c, x)\), \((c, b, x)\).
Protocol:
If \( f(S, T) = true \), prover sends

- \( \left( a, b \right) \)
- \( \text{distance of } x \text{ from } \{a, b\} \)
- \( 1 \text{ bit: clockwise or counter-clockwise?} \)
- \( 1 \text{ bit: distance from } a \text{ or } b \) (which one is closer)?
- \( \text{(couple of bits for special cases)} \)
Protocol:
If \( f(S, T) = \text{true} \), prover sends

- \((a, b)\)

- 1 bit: clockwise or counter-clockwise?
- 1 bit: distance from \(a\) or \(b\) (which one is closer)?
- (couple of bits for special cases)
A Glimpse at the Details (2)

Protocol:
If \( f(S, T) = \text{true} \), prover sends
- \((a, b)\)
- distance of \(x\) from \(\{a, b\}\)
A Glimpse at the Details (2)

Protocol:
If \( f(S, T) = \text{true} \), prover sends

- \((a, b)\)
- distance of \(x\) from \(\{a, b\}\)
- 1 bit: clockwise or counter-clockwise?
A Glimpse at the Details (2)

Protocol:
If \( f(S, T) = \text{true} \), prover sends

- \((a, b)\)
- distance of \(x\) from \(\{a, b\}\)
- 1 bit: clockwise or counter-clockwise?
- 1 bit: distance from \(a\) or \(b\) (which one is closer)?
A Glimpse at the Details (2)

Protocol:
If \( f(S, T) = \text{true} \), prover sends

- \((a, b)\)
- distance of \( x \) from \( \{a, b\}\)
- 1 bit: clockwise or counter-clockwise?
- 1 bit: distance from \( a \) or \( b \) (which one is closer)?
- (couple of bits for special cases)
A Glimpse at the Details (2)

Protocol:
If $f(S, T) = \text{true}$, prover sends:

- $(a, b)$
- $\lceil \log_2(\text{distance of } x \text{ from } \{a, b\}) \rceil$
- 1 bit: clockwise or counter-clockwise?
- 1 bit: distance from $a$ or $b$ (which one is closer)?
- (couple of bits for special cases)
A Glimpse at the Details (3)

Prover sends \((a, b, j = \log(\text{dist}), \ldots)\).

Alice

\[
\text{Can identify region } R = \{a - 2^j, \ldots, a - 2^j + 1\} \text{ on circle.
}
\]

Knows that \(x \in R\) — but doesn't know \(x\).

Accepts if \(\forall x \in R: x \not\in S\).

Bob

\[
\text{Can identify region } R = \{2^j, \ldots, 2^j + 1\} \text{ on circle.
}
\]

Knows that \(x \in R\) — but doesn't know \(x\).

Accepts if \(\exists x \in R: a \text{⌢} x \text{⌢} b\) in \(T\).
A Glimpse at the Details (3)

Prover sends \((a, b, j = \log(\text{dist}), \ldots)\).

**Alice**

- Can identify region \(R := \{a - 2^j, \ldots, a - 2^{j+1} - 1\}\) on circle

**Bob**

- Can identify region \(R = \{2^j, \ldots, 2^{j+1} - 1\}\) on circle

- Knows that \(x \in R\) — but doesn’t know \(x\)

- Accepts if \(\forall x \in R: x \notin T\)
A Glimpse at the Details (3)

Prover sends \((a, b, j = \log(\text{dist}), \ldots)\).

Alice

- Can identify region \(R \:= \{a - 2^j, \ldots, a - 2^{j+1} - 1\}\) on circle
- Knows that \(x \in R\) — but doesn’t know \(x\)
A Glimpse at the Details (3)

Prover sends \((a, b, j = \log(\text{dist}), \ldots)\).

Alice

- Can identify region \(R := \{a - 2^j, \ldots, a - 2^{j+1} - 1\}\) on circle
- Knows that \(x \in R\) — but doesn’t know \(x\)
- Accepts if \(\forall x \in R: x \notin S\)
A Glimpse at the Details (3)

Prover sends \((a, b, j = \log(\text{dist}), \ldots)\).

Alice

- Can identify region \(R := \{a - 2^j, \ldots, a - 2^{j+1} - 1\}\) on circle
- Knows that \(x \in R\) — but doesn’t know \(x\)
- Accepts if \(\forall x \in R: x \notin S\)

Bob
A Glimpse at the Details (3)

Prover sends ($a, b, j = \log(\text{dist}), \ldots$).

Alice

► Can identify region $R := \{a - 2^j, \ldots, a - 2^{j+1} - 1\}$ on circle
► Knows that $x \in R$ — but doesn’t know $x$
► Accepts if $\forall x \in R: x \notin S$

Bob

► Can identify region $R = \{2^j, \ldots, 2^{j+1} - 1\}$ on circle
A Glimpse at the Details (3)

Prover sends \((a, b, j = \log(\text{dist}), \ldots)\).

Alice

- Can identify region \(R := \{a - 2^j, \ldots, a - 2^{j+1} - 1\}\) on circle
- Knows that \(x \in R\) — but doesn’t know \(x\)
- Accepts if \(\forall x \in R: x \notin S\)

Bob

- Can identify region \(R = \{2^j, \ldots, 2^{j+1} - 1\}\) on circle
- Knows that \(x \in R\) — but doesn’t know \(x\)
A Glimpse at the Details (3)

Prover sends \((a, b, j = \log(\text{dist}), \ldots)\).

**Alice**

- Can identify region \(R := \{a - 2^j, \ldots, a - 2^{j+1} - 1\}\) on circle
- Knows that \(x \in R\) — but doesn’t know \(x\)
- Accepts if \(\forall x \in R: x \notin S\)

**Bob**

- Can identify region \(R = \{2^j, \ldots, 2^{j+1} - 1\}\) on circle
- Knows that \(x \in R\) — but doesn’t know \(x\)
- Accepts if \(\exists x \in R: a \circlearrowleft x \circlearrowleft b\) in \(T\)
Lemma 1
If $f(S, T) = \text{true}$, Alice and Bob accept.
Lemma 1
If \( f(S, T) = \text{true} \), Alice and Bob accept.

Proof.
Lemma 1
If \( f(S, T) = \text{true} \), Alice and Bob accept.

Proof.

- Bob: √
Lemma 1
If \( f(S, T) = \text{true} \), Alice and Bob accept.

Proof.

- Bob: √
- Alice: remember: accepts if \( \forall x \in R: x \notin S \)
Lemma 1
If $f(S, T) = \text{true}$, Alice and Bob accept.

Proof.

- Bob: √
- Alice: remember: accepts if $\forall x \in R: x \notin S$
- Why $\forall$? Why not $\exists$?
Lemma 1
If $f(S, T) = \text{true}$, Alice and Bob accept.

Proof.

- Bob: √
- Alice: remember: accepts if $\forall x \in R: x \notin S$
- Why $\forall$? Why not $\exists$?
- $\Rightarrow$ Triangle Lemma: If $c \in R \cap S$, then one of $(a, c, x), (c, b, x)$ is witness for $f(S, T) = \text{true}$.
Lemma 1
If $f(S, T) = \text{true}$, Alice and Bob accept.

Proof.

- Bob: $\sqrt{}$
- Alice: remember: accepts if $\forall x \in R: x \notin S$
- Why $\forall$? Why not $\exists$?
- $\Rightarrow$ Triangle Lemma: If $c \in R \cap S$, then one of $(a, c, x), (c, b, x)$ is witness for $f(S, T) = \text{true}$.
- Smart choice of witness that prover sends:

  **Prover sends** $(a, b, x)$ **w/** $d(a, x) + d(x, b)$ **minimal.**
Lemma 1
If \( f(S, T) = \text{true} \), Alice and Bob accept.

Proof.

- Bob: \( \sqrt{\text{✓}} \)
- Alice: remember: accepts if \( \forall x \in R: x \notin S \)
- Why \( \forall \)? Why not \( \exists \)?
- \( \Rightarrow \) Triangle Lemma: If \( c \in R \cap S \), then one of \((a, c, x), (c, b, x)\) is witness for \( f(S, T) = \text{true} \).
- Smart choice of witness that prover sends:
  \[
  \text{Prover sends } (a, b, x) \text{ w/ } d(a, x) + d(x, b) \text{ minimal.}
  \]
- Such a \( c \) would contradict minimality.
Lemma 2
If \( f(S, T) = \text{false} \), Alice rejects or Bob rejects.
Lemma 2
If $f(S, T) = \text{false}$, Alice rejects or Bob rejects.

Proof.
Counterpositive:
Lemma 2
If \( f(S, T) = \text{false} \), Alice rejects or Bob rejects.

Proof.
Counterpositive:
- Suppose both Alice and Bob accept.

\[ \forall x \in R : x \not\in S \]

\[ \exists x \in R : a \triangleright x \triangleright b \text{ in } T \]

\[ \Rightarrow S \text{ disconnected in } T \]

\[ f(S, T) = \text{true} \]
Lemma 2
If $f(S, T) = false$, Alice rejects or Bob rejects.

Proof.
Counterpositive:
- Suppose both Alice and Bob accept.
- Alice: $\forall x \in R: x \notin S$. 
Lemma 2
If \( f(S, T) = \text{false} \), Alice rejects or Bob rejects.

Proof.
Counterpositive:
- Suppose both Alice and Bob accept.
- Alice: \( \forall x \in R: x \notin S \).
- Bob: \( \exists x \in R: a \leadsto x \leadsto b \) in \( T \).
Lemma 2
If $f(S, T) = \text{false}$, Alice rejects or Bob rejects.

Proof.
Counterpositive:
- Suppose both Alice and Bob accept.
- Alice: $\forall x \in R: x \notin S$.
- Bob: $\exists x \in R: a \sim x \sim b$ in $T$.
- $\Rightarrow S$ disconnected in $T$. 

\checkmark
Lemma 2
If $f(S, T) = \text{false}$, Alice rejects or Bob rejects.

Proof.
Counterpositive:
- Suppose both Alice and Bob accept.
- Alice: $\forall x \in R: x \not\in S$.
- Bob: $\exists x \in R: a \sim x \sim b$ in $T$.
- $\Rightarrow S$ disconnected in $T$.
- $f(S, T) = \text{true}$.
Conclusion

- Nondet. CC lower bound for Spanning Tree extension complexity is (almost) useless
Conclusion

- Nondet. CC lower bound for Spanning Tree extension complexity is (almost) useless
  - $\Omega(n^2) \sim \Omega(n^2 \log n)$ improvement is still a possibility
Conclusion

- Nondet. CC lower bound for Spanning Tree extension complexity is (almost) useless
  - $\Omega(n^2) \Rightarrow \Omega(n^2 \log n)$ improvement is still a possibility
- Other than that, “non-combinatorial” techniques are needed.
Nondet. CC lower bound for Spanning Tree extension complexity is (almost) useless
- $\Omega(n^2) \sim \Omega(n^2 \log n)$ improvement is still a possibility
- Other than that, “non-combinatorial” techniques are needed.
- ...or a $o(n^3)$ extended formulation.
Need More Brainpower
Need More Brainpower