How Limited Interaction Hinders Real Communication
(and What It Means for Proof and Circuit Complexity)

Jakob Nordström

KTH Royal Institute of Technology
Stockholm, Sweden

CCC 2016 Satellite Kyoto Workshop
Kyoto University
June 2, 2016

Joint work with Susanna F. de Rezende and Marc Vinyals

The SAT Problem in Theory and Practice

Complexity theory

- Satisfiability of formulas in propositional logic foundational problem
- SAT proven NP-complete in [Coo71, Lev73]
- Hence most likely totally intractable
- Just remains to prove this — one of the million-dollar "Millennium Problems"

The SAT Problem in Theory and Practice

Complexity theory

- Satisfiability of formulas in propositional logic foundational problem
- SAT proven NP-complete in [Coo71, Lev73]
- Hence most likely totally intractable
- Just remains to prove this — one of the million-dollar "Millennium Problems"

Applied SAT solving

- Dramatic performance increase last 15–20 years
- State-of-the-art SAT solvers can deal with millions of variables
- But we also know tiny formulas that are totally beyond reach
- Why do SAT solvers work so well? And why do they sometimes miserably fail?

SAT solving and Proof Complexity

- State-of-the-art SAT solvers use methods such as
  - conflict-driven clause learning (CDCL)
  - Gröbner bases
  - pseudo-Boolean reasoning

SAT solving and Proof Complexity

- State-of-the-art SAT solvers use methods such as
  - conflict-driven clause learning (CDCL)
  - Gröbner bases
  - pseudo-Boolean reasoning
- Absolutely key to minimize
  - running time
  - memory usage

SAT solving and Proof Complexity

- State-of-the-art SAT solvers use methods such as
  - conflict-driven clause learning (CDCL) — resolution
  - Gröbner bases — polynomial calculus
  - pseudo-Boolean reasoning — cutting planes
- Absolutely key to minimize
  - running time — proof size
  - memory usage — proof space
- Only known rigorous analysis approach: use proof complexity [CR79] to study underlying methods of reasoning

SAT solving and Proof Complexity

- State-of-the-art SAT solvers use methods such as
  - conflict-driven clause learning (CDCL) — resolution
  - Gröbner bases — polynomial calculus
  - pseudo-Boolean reasoning — cutting planes
- Absolutely key to minimize
  - running time — proof size
  - memory usage — proof space
- Only known rigorous analysis approach: use proof complexity [CR79] to study underlying methods of reasoning
- Requires lower-bounding optimal, nondeterministic algorithms — yet possible to prove strong (and sometimes tight!) size-space trade-offs

SAT solving and Proof Complexity

- State-of-the-art SAT solvers use methods such as
  - conflict-driven clause learning (CDCL) — resolution
  - Gröbner bases — polynomial calculus
  - pseudo-Boolean reasoning — cutting planes
- Absolutely key to minimize
  - running time — proof size
  - memory usage — proof space
- Only known rigorous analysis approach: use proof complexity [CR79] to study underlying methods of reasoning
- Requires lower-bounding optimal, nondeterministic algorithms — yet possible to prove strong (and sometimes tight!) size-space trade-offs for resolution and polynomial calculus

SAT solving and Proof Complexity

- State-of-the-art SAT solvers use methods such as
  - conflict-driven clause learning (CDCL) — resolution
  - Gröbner bases — polynomial calculus
  - pseudo-Boolean reasoning — cutting planes
- Absolutely key to minimize
  - running time — proof size
  - memory usage — proof space
- Only known rigorous analysis approach: use proof complexity [CR79] to study underlying methods of reasoning
- Requires lower-bounding optimal, nondeterministic algorithms — yet possible to prove strong (and sometimes tight!) size-space trade-offs for resolution and polynomial calculus
- **This work:** First such strong trade-offs capturing also cutting planes

Informal Statement of Results

Theorem (Main)

First time-space trade-offs holding uniformly for resolution, polynomial calculus, and cutting planes for formulas such that:

- $\exists$ *proofs in small size*
- $\exists$ *proofs in small total space*
- $\forall$ *proofs few formulas in memory* $\Rightarrow$ *length exponential*

Informal Statement of Results

Theorem (Main)

First time-space trade-offs holding uniformly for resolution, polynomial calculus, and cutting planes for formulas such that:

- $\exists$ *proofs in small size*
- $\exists$ *proofs in small total space*
- $\forall$ *proofs few formulas in memory* $\Rightarrow$ *length exponential*

Theorem (By-product)

Exponential separation in monotone-AC^i *hierarchy (improving on [RM99])*

Conjunctive Normal Form

$$(x \vee y) \wedge (x \vee \overline{y} \vee z) \wedge (\overline{x} \vee z) \wedge (\overline{y} \vee \overline{z}) \wedge (\overline{x} \vee \overline{z})$$

- Literal a: variable x or its negation $\overline{x}$
- Clause $C = a_1 \vee \cdots \vee a_k$: disjunction of literals
  (Consider as sets, so no repetitions and order irrelevant)
- CNF formula $F = C_1 \wedge \cdots \wedge C_m$: conjunction of clauses
- Task: Refute given CNF formula (i.e., prove it is unsatisfiable)

The Theoretical Model

- Proof system operates with formulas of some syntactic form
- Proof/refutation is "presented on blackboard"
- Derivation steps:
  - Write down axiom clauses of CNF formula being refuted (as encoded by proof system)
  - Infer new lines by deductive rules of proof system
  - Erase lines not currently needed (to save space on blackboard)
- Refutation ends when (explicit) contradiction is derived

Cutting Planes (CP)

Clauses interpreted as linear inequalities

E.g., $x \vee y \vee \overline{z} \;\rightsquigarrow\; x + y + (1 - z) \geq 1 \;\rightsquigarrow\; x + y - z \geq 0$

Cutting Planes (CP)

Clauses interpreted as linear inequalities

E.g., $x \vee y \vee \overline{z} \rightsquigarrow x + y + (1 - z) \geq 1 \rightsquigarrow x + y - z \geq 0$

Variable axioms $\dfrac{}{0 \leq x \leq 1}$

Multiplication $\dfrac{\sum a_i x_i \geq A}{\sum c a_i x_i \geq cA}$

Addition $\dfrac{\sum a_i x_i \geq A \quad \sum b_i x_i \geq B}{\sum (a_i + b_i) x_i \geq A + B}$

Division $\dfrac{\sum c a_i x_i \geq A}{\sum a_i x_i \geq \lceil A/c \rceil}$

Goal: Derive $0 \geq 1 \Leftrightarrow$ formula unsatisfiable

Cutting Planes (CP)

Clauses interpreted as linear inequalities
E.g., $x \vee y \vee \overline{z} \rightsquigarrow x + y + (1 - z) \geq 1 \rightsquigarrow x + y - z \geq 0$

Variable axioms $$\frac{}{0 \leq x \leq 1}$$

Multiplication $$\frac{\sum a_i x_i \geq A}{\sum c a_i x_i \geq cA}$$

Addition $$\frac{\sum a_i x_i \geq A \quad \sum b_i x_i \geq B}{\sum (a_i + b_i) x_i \geq A + B}$$

Division $$\frac{\sum c a_i x_i \geq A}{\sum a_i x_i \geq \lceil A/c \rceil}$$

Goal: Derive $0 \geq 1 \Leftrightarrow$ formula unsatisfiable

Exact derivation rules not too important for our work — just need to know that we operate with linear inequalities

Complexity Measures for Cutting Planes

Length = total # lines/inequalities in refutation

Size = sum also sizes of coefficients

Line space = max # lines in memory during refutation

Total space = sum of sizes of coefficients of lines in memory

Complexity Measures for Cutting Planes

Length = total # lines/inequalities in refutation

Size = sum also sizes of coefficients

Line space = max # lines in memory during refutation

Total space = sum of sizes of coefficients of lines in memory

Worst-case bounds size $\leq 2^{\mathcal{O}(n)}$ and total space $\leq \mathcal{O}(n^2)$ for CNF formula over n variables, so mindset should be

- large size $\approx \exp(n^\delta)$
- large space $\approx n^\delta$

Size and Space in Cutting Planes

- Short refutations of so-called (lifted) Tseitin formulas on expanders need large space [GP14] (but such refutations probably don't exist)

Size and Space in Cutting Planes

- Short refutations of so-called (lifted) Tseitin formulas on expanders need large space [GP14] (but such refutations probably don't exist)
- Short refutations of some so-called pebbling formulas need large space [HN12, GP14] (and such refutations do exist)

Size and Space in Cutting Planes

- Short refutations of so-called (lifted) Tseitin formulas on expanders need large space [GP14] (but such refutations probably don't exist)
- Short refutations of some so-called pebbling formulas need large space [HN12, GP14] (and such refutations do exist)
- Recent surprise: CP can refute any CNF in line space 5 (!) [GPT15] (But coefficients will be exponentially large)

Size and Space in Cutting Planes

- Short refutations of so-called (lifted) Tseitin formulas on expanders need large space [GP14] (but such refutations probably don't exist)
- Short refutations of some so-called pebbling formulas need large space [HN12, GP14] (and such refutations do exist)
- Recent surprise: CP can refute any CNF in line space 5 (!) [GPT15] (But coefficients will be exponentially large)
- Plug into [HN12, GP14] $\Rightarrow$ trade-off of sorts

Size and Space in Cutting Planes

- Short refutations of so-called (lifted) Tseitin formulas on expanders need large space [GP14] (but such refutations probably don't exist)
- Short refutations of some so-called pebbling formulas need large space [HN12, GP14] (and such refutations do exist)
- Recent surprise: CP can refute any CNF in line space 5 (!) [GPT15] (But coefficients will be exponentially large)
- Plug into [HN12, GP14] $\Rightarrow$ trade-off of sorts
- But "constant-space" proofs with exponential-size coefficients somehow doesn't feel quite right...

Size and Space in Cutting Planes

- Short refutations of so-called (lifted) Tseitin formulas on expanders need large space [GP14] (but such refutations probably don't exist)
- Short refutations of some so-called pebbling formulas need large space [HN12, GP14] (and such refutations do exist)
- Recent surprise: CP can refute any CNF in line space 5 (!) [GPT15] (But coefficients will be exponentially large)
- Plug into [HN12, GP14] $\Rightarrow$ trade-off of sorts
- But "constant-space" proofs with exponential-size coefficients somehow doesn't feel quite right...

What about "true" trade-offs?

Are there trade-offs where the space-efficient CP refutations have small coefficients? (Say, of polynomial or even constant size)

Our Main Result

Theorem (Informal sample)

There are families of 6*-CNF formulas* $\{F_N\}_{N=1}^{\infty}$ *of size* $\Theta(N)$ *such that:*

Our Main Result

Theorem (Informal sample)

There are families of 6*-CNF formulas* $\{F_N\}_{N=1}^{\infty}$ *of size* $\Theta(N)$ *such that:*

1. F_N *can be refuted by cutting planes with constant-size coefficients in size* $\mathcal{O}(N)$ *and total space* $\mathcal{O}(N^{2/5})$

Our Main Result

Theorem (Informal sample)

There are families of 6-CNF formulas $\{F_N\}_{N=1}^{\infty}$ of size $\Theta(N)$ such that:

1. F_N *can be refuted by cutting planes with constant-size coefficients in size* $\mathcal{O}(N)$ *and total space* $\mathcal{O}\big(N^{2/5}\big)$
2. F_N *can be refuted by cutting planes with constant-size coefficients in total space* $\mathcal{O}\big(N^{1/40}\big)$ *and size* $2^{\mathcal{O}(N^{1/40})}$

Our Main Result

Theorem (Informal sample)

There are families of 6-CNF formulas $\{F_N\}_{N=1}^{\infty}$ of size $\Theta(N)$ such that:

1. F_N *can be refuted by cutting planes with constant-size coefficients in size* $\mathcal{O}(N)$ *and total space* $\mathcal{O}\left(N^{2/5}\right)$
2. F_N *can be refuted by cutting planes with constant-size coefficients in total space* $\mathcal{O}\left(N^{1/40}\right)$ *and size* $2^{\mathcal{O}(N^{1/40})}$
3. *Any cutting planes refutation even with coefficients of unbounded size in line space* $\mathrm{o}\left(N^{1/20}\right)$ *requires length* $2^{\Omega(N^{1/40})}$

Our Main Result

Theorem (Informal sample)

There are families of 6-CNF formulas $\{F_N\}_{N=1}^{\infty}$ of size $\Theta(N)$ such that:

1. F_N *can be refuted by cutting planes with constant-size coefficients in size $\mathcal{O}(N)$ and total space $\mathcal{O}(N^{2/5})$*
2. F_N *can be refuted by cutting planes with constant-size coefficients in total space $\mathcal{O}(N^{1/40})$ and size $2^{\mathcal{O}(N^{1/40})}$*
3. *Any cutting planes refutation even with coefficients of unbounded size in line space $\mathrm{o}(N^{1/20})$ requires length $2^{\Omega(N^{1/40})}$*

Remarks:

- Upper bounds for # bits; lower bounds for # formulas/lines
- Analogous bounds also for resolution & polynomial calculus
- Even for semantic versions of proof systems where anything implied by blackboard can be inferred in just one step

Outline of Proof

Proof is by carefully constructed chain of delicate reductions

Outline of Proof

Proof is by carefully constructed chain of delicate reductions
(a.k.a. the kitchen sink)

Outline of Proof

Proof is by carefully constructed chain of delicate reductions
(a.k.a. the kitchen sink)

1. Short, space-efficient proof $\Rightarrow$ efficient communication protocol for falsified clause search problem [HN12].

Outline of Proof

Proof is by carefully constructed chain of delicate reductions (a.k.a. the kitchen sink)

1. Short, space-efficient proof $\Rightarrow$ efficient communication protocol for falsified clause search problem [HN12]. Crucial twists:
   - Study real communication model [Kra98, BEGJ00]
   - Consider round efficiency of protocols

Outline of Proof

Proof is by carefully constructed chain of delicate reductions (a.k.a. the kitchen sink)

1. Short, space-efficient proof ⇒ efficient communication protocol for falsified clause search problem [HN12]. Crucial twists:
   - Study real communication model [Kra98, BEGJ00]
   - Consider round efficiency of protocols
2. Protocol for composed search problem ⇒ parallel decision tree via simulation theorem à la [RM99, GPW15]

Outline of Proof

Proof is by carefully constructed chain of delicate reductions (a.k.a. the kitchen sink)

1. Short, space-efficient proof ⇒ efficient communication protocol for falsified clause search problem [HN12]. Crucial twists:
   - Study real communication model [Kra98, BEGJ00]
   - Consider round efficiency of protocols
2. Protocol for composed search problem ⇒ parallel decision tree via simulation theorem à la [RM99, GPW15]
3. Parallel decision tree for pebbling formulas Peb_G ⇒ pebbling strategy for Dymond–Tompa game on G [DT85]

Outline of Proof

Proof is by carefully constructed chain of delicate reductions (a.k.a. the kitchen sink)

1. Short, space-efficient proof ⇒ efficient communication protocol for falsified clause search problem [HN12]. Crucial twists:
   - Study real communication model [Kra98, BEGJ00]
   - Consider round efficiency of protocols
2. Protocol for composed search problem ⇒ parallel decision tree via simulation theorem à la [RM99, GPW15]
3. Parallel decision tree for pebbling formulas Peb_G ⇒ pebbling strategy for Dymond–Tompa game on G [DT85]
4. Construct graphs G with strong round-cost trade-offs for Dymond–Tompa pebbling

Real Communication

- Main players:
  - Alice with private input x
  - Bob with private input y
  - Both deterministic but have unbounded computational powers

Real Communication

- Main players:
  - Alice with private input x
  - Bob with private input y
  - Both deterministic but have unbounded computational powers
- Task: compute $f(x, y)$ by sending messages to referee

Real Communication

- Main players:
  - Alice with private input x
  - Bob with private input y
  - Both deterministic but have unbounded computational powers
- Task: compute $f(x, y)$ by sending messages to referee
- Method: In each round v
  - Alice sends $a_{v,1}(x), \ldots, a_{v,c_v}(x) \in \mathbb{R}^{c_v}$
  - Bob sends $b_{v,1}(y), \ldots, b_{v,c_v}(y) \in \mathbb{R}^{c_v}$
  - Referee announces results of comparisons $a_{v,i}(x) \leq b_{v,i}(y)$ for $i \in [c_v]$

Real Communication

- Main players:
  - Alice with private input x
  - Bob with private input y
  - Both deterministic but have unbounded computational powers
- Task: compute $f(x, y)$ by sending messages to referee
- Method: In each round v
  - Alice sends $a_{v,1}(x), \ldots, a_{v,c_v}(x) \in \mathbb{R}^{c_v}$
  - Bob sends $b_{v,1}(y), \ldots, b_{v,c_v}(y) \in \mathbb{R}^{c_v}$
  - Referee announces results of comparisons $a_{v,i}(x) \leq b_{v,i}(y)$ for $i \in [c_v]$
- Function f solved by r-round real communication in cost c if $\exists$ protocol such that
  - $\#$ rounds $\leq r$
  - total $\#$ comparisons made by referee $\leq c$

Real Communication

- Main players:
  - Alice with private input x
  - Bob with private input y
  - Both deterministic but have unbounded computational powers
- Task: compute $f(x, y)$ by sending messages to referee
- Method: In each round v
  - Alice sends $a_{v,1}(x), \ldots, a_{v,c_v}(x) \in \mathbb{R}^{c_v}$
  - Bob sends $b_{v,1}(y), \ldots, b_{v,c_v}(y) \in \mathbb{R}^{c_v}$
  - Referee announces results of comparisons $a_{v,i}(x) \leq b_{v,i}(y)$ for $i \in [c_v]$
- Function f solved by r-round real communication in cost c if $\exists$ protocol such that
  - $\#$ rounds $\leq r$
  - total $\#$ comparisons made by referee $\leq c$
- Strictly stronger than standard deterministic communication

Falsified Clause Search Problem

Fix:

- unsatisfiable CNF formula F
- (devious) partition of $Vars(F)$ between Alice and Bob

Falsified clause search problem $Search(F)$

Input: Assignment α to $Vars(F)$ split between Alice and Bob

Output: Clause $C \in F$ such that α falsifies C

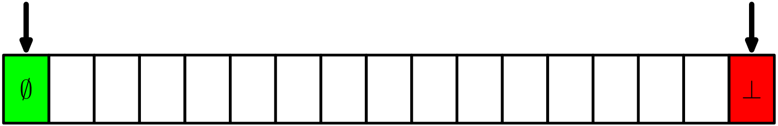
Actually, computing not function but relation — will mostly ignore this for simplicity

Succinct Refutations Yield Efficient Protocols

Evaluate blackboard configurations of a refutation of F under α

Succinct Refutations Yield Efficient Protocols

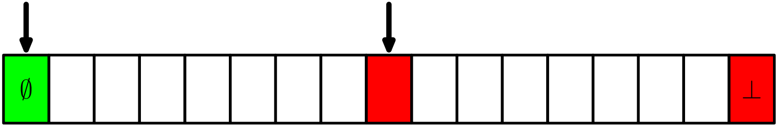
Evaluate blackboard configurations of a refutation of F under α



Use binary search to find transition from true to false blackboard

Succinct Refutations Yield Efficient Protocols

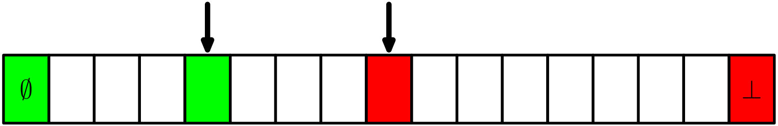
Evaluate blackboard configurations of a refutation of F under α



Use binary search to find transition from true to false blackboard

Succinct Refutations Yield Efficient Protocols

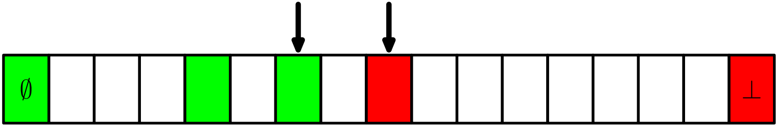
Evaluate blackboard configurations of a refutation of F under α



Use binary search to find transition from true to false blackboard

Succinct Refutations Yield Efficient Protocols

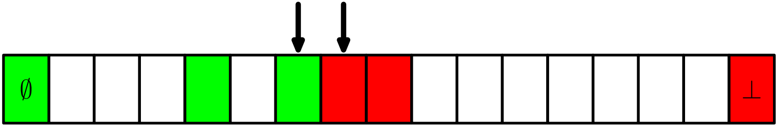
Evaluate blackboard configurations of a refutation of F under α



Use binary search to find transition from true to false blackboard

Succinct Refutations Yield Efficient Protocols

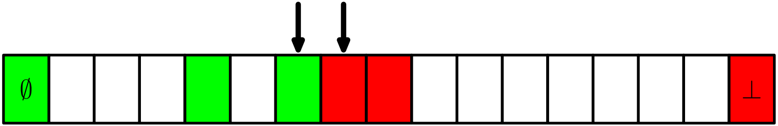
Evaluate blackboard configurations of a refutation of F under α



Use binary search to find transition from true to false blackboard

Succinct Refutations Yield Efficient Protocols

Evaluate blackboard configurations of a refutation of F under α

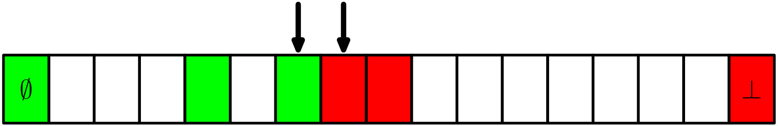


Use binary search to find transition from true to false blackboard

Must happen when $C \in F$ written down — answer to $Search(F)$

Succinct Refutations Yield Efficient Protocols

Evaluate blackboard configurations of a refutation of F under α



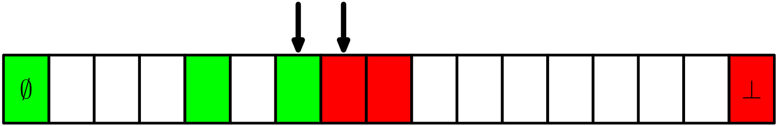
Use binary search to find transition from true to false blackboard

Must happen when $C \in F$ written down — answer to $Search(F)$

Length $L \Rightarrow$ evaluate $\log L$ blackboards

Succinct Refutations Yield Efficient Protocols

Evaluate blackboard configurations of a refutation of F under α



Use binary search to find transition from true to false blackboard

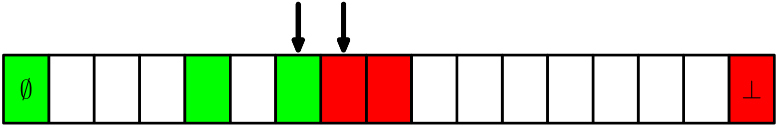
Must happen when $C \in F$ written down — answer to $Search(F)$

Length $L \Rightarrow$ evaluate $\log L$ blackboards

Line space $s \Rightarrow$ max s bits of communication per blackboard

Succinct Refutations Yield Efficient Protocols

Evaluate blackboard configurations of a refutation of F under α



Use binary search to find transition from true to false blackboard

Must happen when $C \in F$ written down — answer to $Search(F)$

Length $L \Rightarrow$ evaluate $\log L$ blackboards

Line space $s \Rightarrow$ max s bits of communication per blackboard

Only one round per blackboard evaluation

(Alice and Bob simply evaluate their parts of each inequality and ask referee to compare)

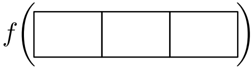
Lifting of Functions

Construct hard communication problems by “hardness amplification” using lifting or composition

Lifting of Functions

Construct hard communication problems by “hardness amplification” using lifting or composition

Start with function $f : \{0,1\}^m \to \{0,1\}$

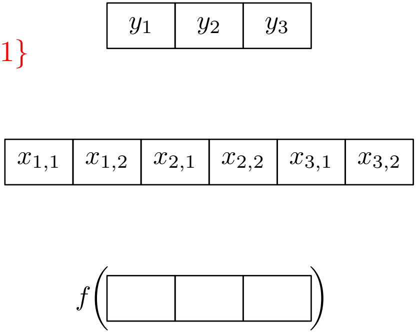


Lifting of Functions

Construct hard communication problems by "hardness amplification" using lifting or composition

Start with function $f : \{0,1\}^m \to \{0,1\}$

Construct new function on inputs $x \in \{0,1\}^{\ell m}$ and $y \in [\ell]^m$



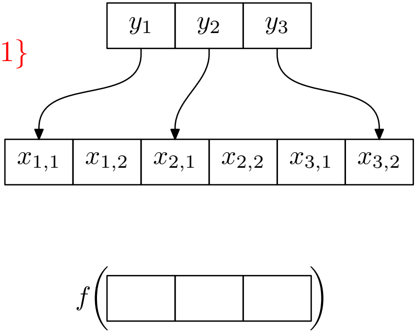
Lifting of Functions

Construct hard communication problems by "hardness amplification" using lifting or composition

Start with function $f : \{0,1\}^m \to \{0,1\}$

Construct new function on inputs $x \in \{0,1\}^{\ell m}$ and $y \in [\ell]^m$

Bob's y-variables determine...



Lifting of Functions

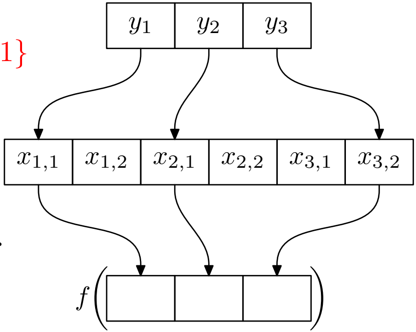
Construct hard communication problems by "hardness amplification" using lifting or composition

Start with function $f : \{0,1\}^m \to \{0,1\}$

Construct new function on inputs $x \in \{0,1\}^{\ell m}$ and $y \in [\ell]^m$

Bob's y-variables determine...

...which of Alice's x-bits to feed to f



Lifting of Functions

Construct hard communication problems by “hardness amplification” using lifting or composition

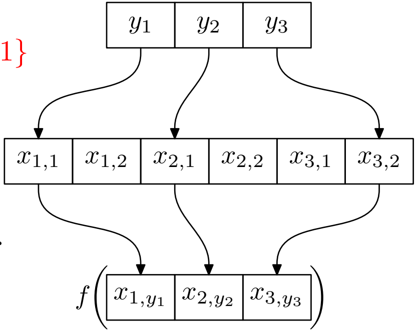
Start with function $f : \{0,1\}^m \rightarrow \{0,1\}$

Construct new function on inputs $x \in \{0,1\}^{\ell m}$ and $y \in [\ell]^m$

Bob's y-variables determine. . .

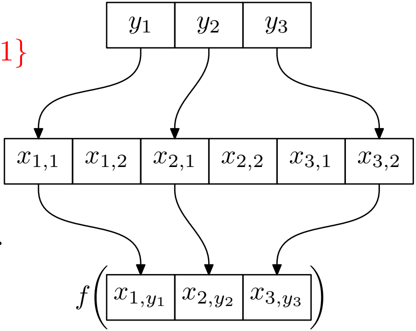
. . . which of Alice's x-bits to feed to f

Length-ℓ lifting of f defined as $Lift_\ell(f)(x,y) := f(x_{1,y_1}, \ldots, x_{m,y_m})$



Lifting of Functions

Construct hard communication problems by “hardness amplification” using lifting or composition



Start with function $f : \{0,1\}^m \to \{0,1\}$

Construct new function on inputs $x \in \{0,1\}^{\ell m}$ and $y \in [\ell]^m$

Bob's y-variables determine…

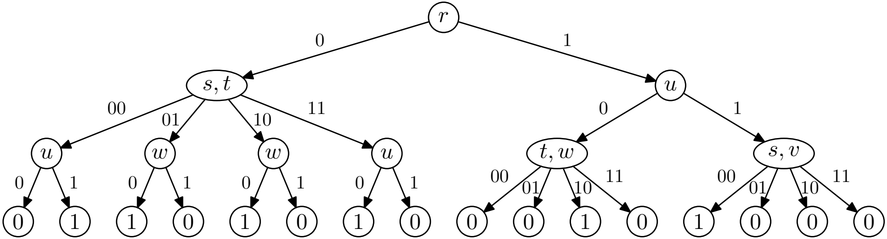
…which of Alice's x-bits to feed to f

Length-ℓ lifting of f defined as
$Lift_\ell(f)(x,y) := f(x_{1,y_1}, \ldots, x_{m,y_m})$

Building on ideas from e.g. [She08, BHP10]
Can encode lifted search problem for F as new CNF formula $Lift(F)$

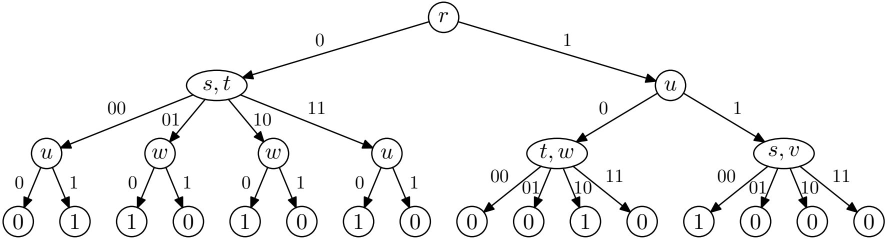
Parallel Decision Trees

Relate lifted problem to parallel decision tree [Val75] for original problem



Parallel Decision Trees

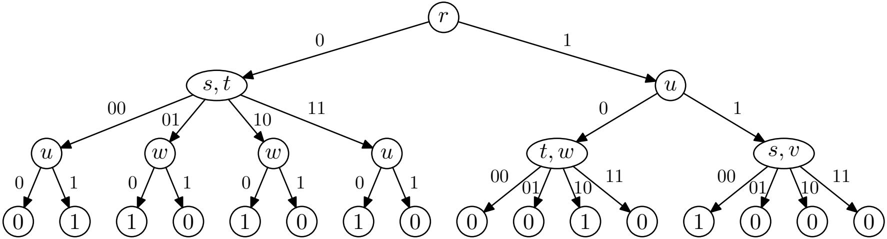
Relate lifted problem to parallel decision tree [Val75] for original problem



- Each node t labelled by variables V_t; exactly $2^{|V_t|}$ outgoing edges

Parallel Decision Trees

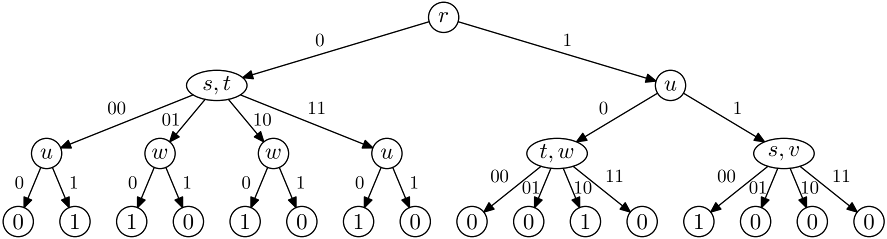
Relate lifted problem to parallel decision tree [Val75] for original problem



- Each node t labelled by variables V_t; exactly $2^{|V_t|}$ outgoing edges
- \# queries = max $\sum |V_t|$ along any path (4 in this example)

Parallel Decision Trees

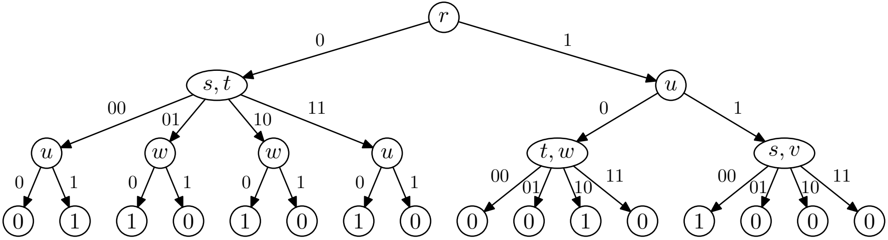
Relate lifted problem to parallel decision tree [Val75] for original problem



- Each node t labelled by variables V_t; exactly $2^{|V_t|}$ outgoing edges
- # queries = max $\sum |V_t|$ along any path (4 in this example)
- depth = length of longest path (3 in this example)

Parallel Decision Trees

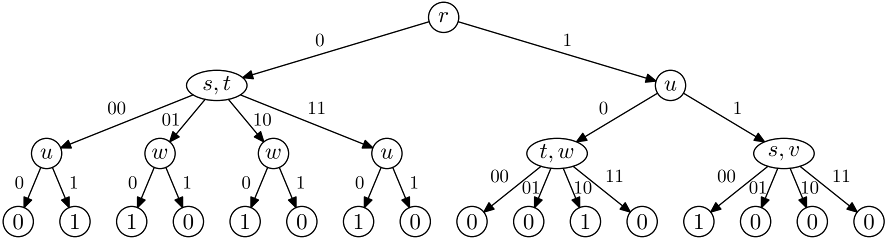
Relate lifted problem to parallel decision tree [Val75] for original problem



- Each node t labelled by variables V_t; exactly $2^{|V_t|}$ outgoing edges
- # queries = max $\sum|V_t|$ along any path (4 in this example)
- depth = length of longest path (3 in this example)
- solves search problem $S \subseteq \{0,1\}^m \times Q$ if $\forall\ \alpha \in \{0,1\}^m$ path defined by α ends in leaf with q s.t. $(\alpha, q) \in S$

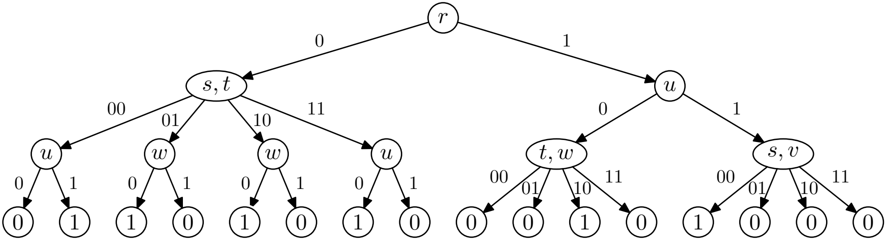
Parallel Decision Trees

Relate lifted problem to parallel decision tree [Val75] for original problem

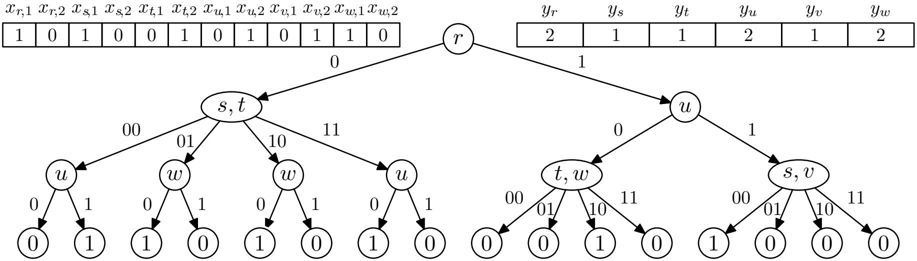


- Each node t labelled by variables V_t; exactly $2^{|V_t|}$ outgoing edges
- # queries = max $\sum |V_t|$ along any path (4 in this example)
- depth = length of longest path (3 in this example)
- solves search problem $S \subseteq \{0,1\}^m \times Q$ if $\forall\, \alpha \in \{0,1\}^m$ path defined by α ends in leaf with q s.t. $(\alpha, q) \in S$
- Easy for Alice & Bob to simulate decision tree to solve lifted problem

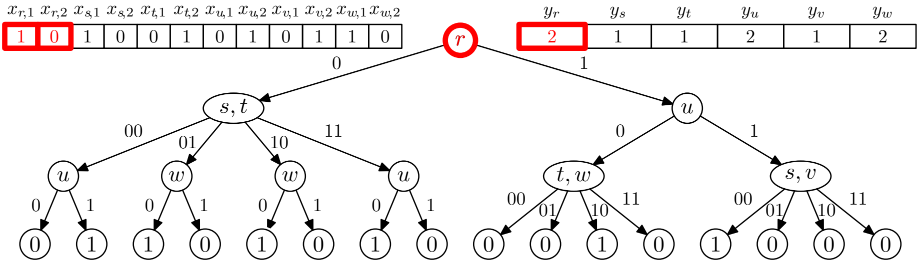
Simulation of Decision Trees by Protocols



Simulation of Decision Trees by Protocols

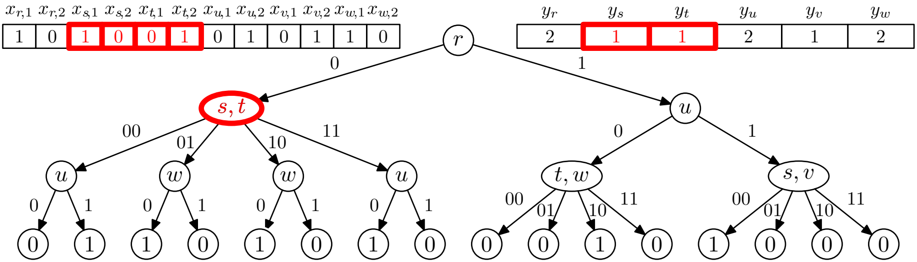


Simulation of Decision Trees by Protocols



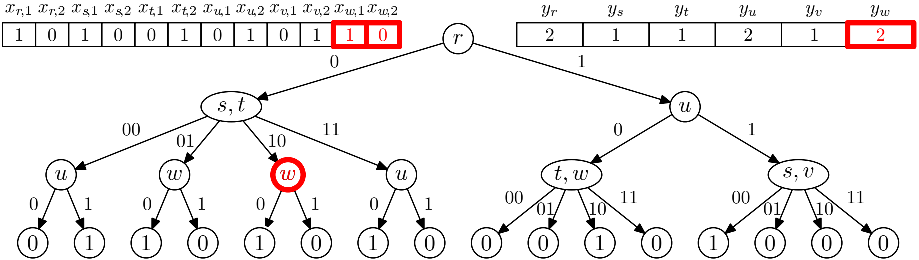
- Bob sends $y_r = 2$, Alice sends $x_{r,2} = 0$, go left;

Simulation of Decision Trees by Protocols



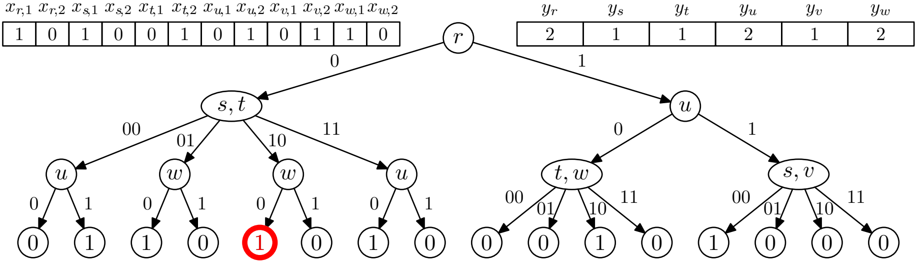
- Bob sends $y_r = 2$, Alice sends $x_{r,2} = 0$, go left;
- Bob sends $(y_s, y_t) = (1, 1)$, Alice sends $(x_{s,1}, x_{t_1}) = (1, 0)$, go 2nd right;

Simulation of Decision Trees by Protocols



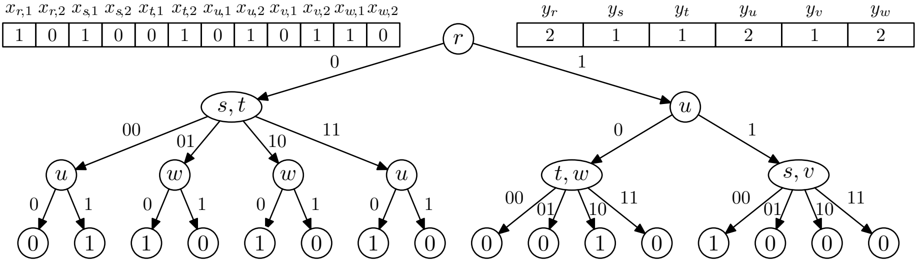
- Bob sends $y_r = 2$, Alice sends $x_{r,2} = 0$, go left;
- Bob sends $(y_s, y_t) = (1, 1)$, Alice sends $(x_{s,1}, x_{t_1}) = (1, 0)$, go 2nd right;
- Bob sends $y_w = 2$, Alice sends $x_{w,2} = 0$, go left

Simulation of Decision Trees by Protocols



- Bob sends $y_r = 2$, Alice sends $x_{r,2} = 0$, go left;
- Bob sends $(y_s, y_t) = (1, 1)$, Alice sends $(x_{s,1}, x_{t_1}) = (1, 0)$, go 2nd right;
- Bob sends $y_w = 2$, Alice sends $x_{w,2} = 0$, go left

Simulation of Decision Trees by Protocols (and Vice Versa)



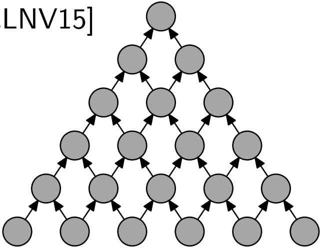
- Bob sends $y_r = 2$, Alice sends $x_{r,2} = 0$, go left;
- Bob sends $(y_s, y_t) = (1, 1)$, Alice sends $(x_{s,1}, x_{t_1}) = (1, 0)$, go 2nd right;
- Bob sends $y_w = 2$, Alice sends $x_{w,2} = 0$, go left

Simulation theorem of protocol by decision tree (hard direction)

Let S search problem with domain $\{0,1\}^m$ and let $\ell = m^{3+\epsilon}$, $\epsilon > 0$. Then:
$\exists$ r-round real communication protocol in cost c solving $Lift_\ell(S)$
$\Rightarrow$ $\exists$ depth-r parallel decision tree solving S width $\mathcal{O}(c/\log \ell)$ queries

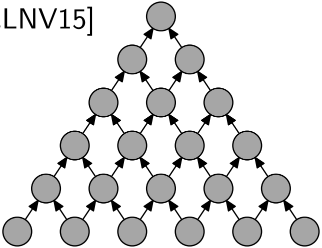
From Parallel Decision Trees to Dymond–Tompa Games

- From [DT85]; recently studied in [Cha13, CLNV15]



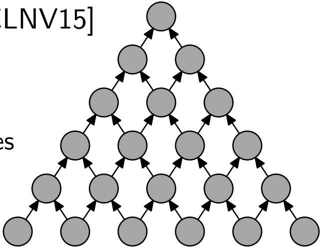
From Parallel Decision Trees to Dymond–Tompa Games

- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger



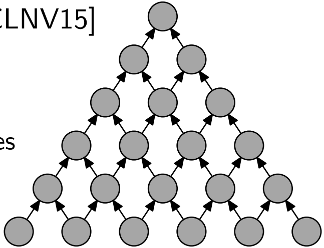
From Parallel Decision Trees to Dymond–Tompa Games

- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays

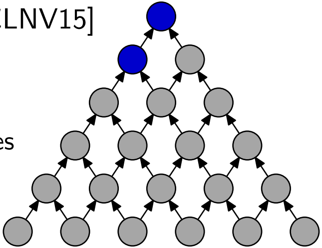


From Parallel Decision Trees to Dymond–Tompa Games

- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays
- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)



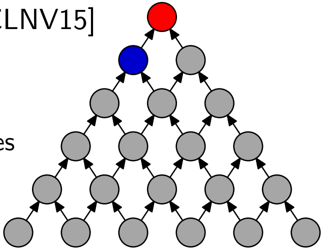
From Parallel Decision Trees to Dymond–Tompa Games



- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays
- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)

From Parallel Decision Trees to Dymond–Tompa Games

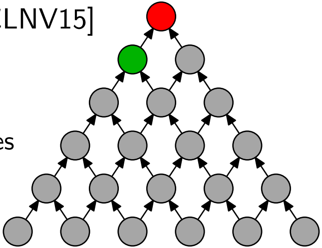
- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays



- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)

From Parallel Decision Trees to Dymond–Tompa Games

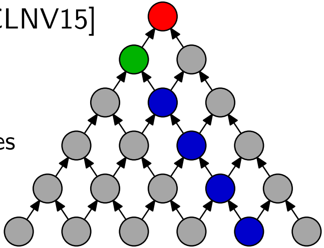
- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays



- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)

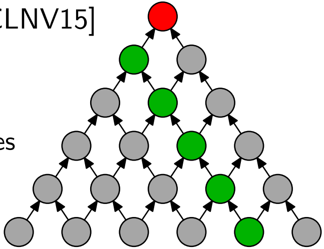
From Parallel Decision Trees to Dymond–Tompa Games

- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays
- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)



From Parallel Decision Trees to Dymond–Tompa Games

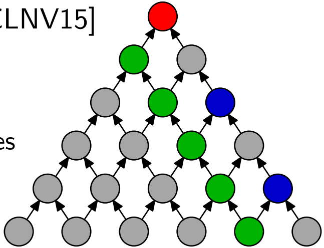
- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays



- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)

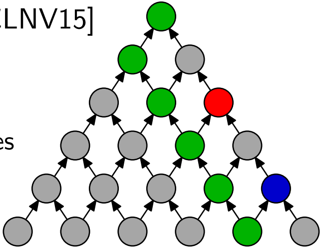
From Parallel Decision Trees to Dymond–Tompa Games

- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays
- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)



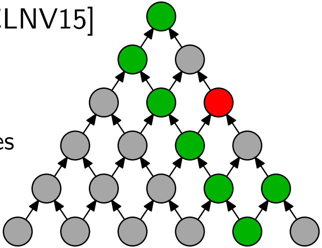
From Parallel Decision Trees to Dymond–Tompa Games

- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
    - Pebbler places pebbles on subset of vertices (including sink in 1st round)
    - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays
- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)



From Parallel Decision Trees to Dymond–Tompa Games

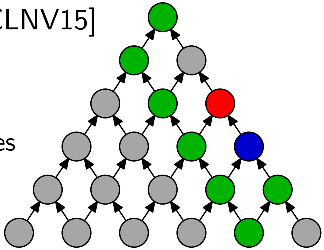
- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays



- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)

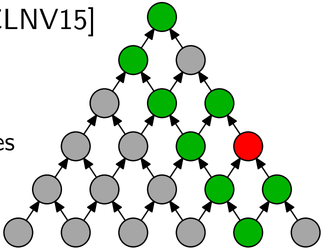
From Parallel Decision Trees to Dymond–Tompa Games

- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays



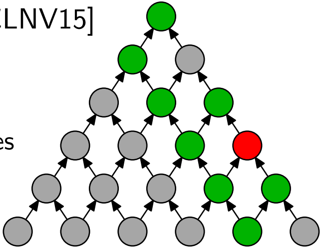
- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)

From Parallel Decision Trees to Dymond–Tompa Games



- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays
- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)

From Parallel Decision Trees to Dymond–Tompa Games



- From [DT85]; recently studied in [Cha13, CLNV15]
- Two players Pebbler and Challenger
- In each round
  - Pebbler places pebbles on subset of vertices (including sink in 1st round)
  - Challenger either jumps to newly pebbled vertex (always in 1st round) or stays
- Pebbler wins at end of round when Challenger on vertex with all predecessors pebbled (or on source vertex)

Lemma

$\exists$ *depth-r parallel decision tree for pebbling formula Peb_G with $\leq c$ queries*
$\Rightarrow$ *Pebbler wins r-round Dymond–Tompa game on G in cost $\leq c+1$*

Putting the Pieces Together (Including the Ones Skipped)

Prove round-cost trade-offs for Dymond–Tompa games on graphs G
(hacking graph constructions from [CS82, LT82, Nor12])

Putting the Pieces Together (Including the Ones Skipped)

Prove round-cost trade-offs for Dymond–Tompa games on graphs G
(hacking graph constructions from [CS82, LT82, Nor12])

$\Downarrow$

Depth-query trade-offs for decision trees for pebbling formulas Peb_G

Putting the Pieces Together (Including the Ones Skipped)

Prove round-cost trade-offs for Dymond–Tompa games on graphs G
(hacking graph constructions from [CS82, LT82, Nor12])

$\Downarrow$

Depth-query trade-offs for decision trees for pebbling formulas Peb_G

$\Downarrow$

Communication round-cost trade-offs for lifted search problem for Peb_G

Putting the Pieces Together (Including the Ones Skipped)

Prove round-cost trade-offs for Dymond–Tompa games on graphs G
(hacking graph constructions from [CS82, LT82, Nor12])

$\Downarrow$

Depth-query trade-offs for decision trees for pebbling formulas Peb_G

$\Downarrow$

Communication round-cost trade-offs for lifted search problem for Peb_G

$\Downarrow$

Cutting planes length-space trade-offs for lifted CNF formulas $Lift(Peb_G)$

Some Remaining Open Questions

Communication complexity

- Smaller lifting gadget? ($\Rightarrow$ stronger trade-offs)
- Simulation theorems for stronger communication models (randomized, multi-party)?

Some Remaining Open Questions

Communication complexity

- Smaller lifting gadget? ($\Rightarrow$ stronger trade-offs)
- Simulation theorems for stronger communication models (randomized, multi-party)?

Proof complexity

- Better Dymond–Tompa trade-offs?
- Size-space trade-offs for Tseitin formulas à la [BBI12, BNT13]?
- Line space lower bounds for CP with bounded coefficients (strengthening [GPT15])

Take-Home Message

Summary of results

- Modern SAT solvers enormously successful in practice — key issue is to minimize time and memory consumption
- Modelled by proof size and space in proof complexity
- We show uniform trade-offs indicating that simultaneous optimization impossible for (essentially all) state-of-the-art techniques

Take-Home Message

Summary of results

- Modern SAT solvers enormously successful in practice — key issue is to minimize time and memory consumption
- Modelled by proof size and space in proof complexity
- We show uniform trade-offs indicating that simultaneous optimization impossible for (essentially all) state-of-the-art techniques

Future directions

- Proof complexity: Understand size and space in cutting planes better
- Communication complexity: Tighter reductions and/or lower bounds in stronger models

Take-Home Message

Summary of results

- Modern SAT solvers enormously successful in practice — key issue is to minimize time and memory consumption
- Modelled by proof size and space in proof complexity
- We show uniform trade-offs indicating that simultaneous optimization impossible for (essentially all) state-of-the-art techniques

Future directions

- Proof complexity: Understand size and space in cutting planes better
- Communication complexity: Tighter reductions and/or lower bounds in stronger models

Thank you for your attention!

References I

[BBI12] Paul Beame, Chris Beck, and Russell Impagliazzo. Time-space tradeoffs in resolution: Superpolynomial lower bounds for superlinear space. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 213–232, May 2012.

[BEGJ00] María Luisa Bonet, Juan Luis Esteban, Nicola Galesi, and Jan Johannsen. On the relative complexity of resolution refinements and cutting planes proof systems. *SIAM Journal on Computing*, 30(5):1462–1484, 2000. Preliminary version in *FOCS '98*.

[BHP10] Paul Beame, Trinh Huynh, and Toniann Pitassi. Hardness amplification in proof complexity. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC '10)*, pages 87–96, June 2010.

[BNT13] Chris Beck, Jakob Nordström, and Bangsheng Tang. Some trade-off results for polynomial calculus. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13)*, pages 813–822, May 2013.

[Cha13] Siu Man Chan. Just a pebble game. In *Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC '13)*, pages 133–143, June 2013.

References II

[CLNV15] Siu Man Chan, Massimo Lauria, Jakob Nordström, and Marc Vinyals. Hardness of approximation in PSPACE and separation results for pebble games (Extended abstract). In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15)*, pages 466–485, October 2015.

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (STOC '71)*, pages 151–158, 1971.

[CR79] Stephen A. Cook and Robert Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, March 1979.

[CS82] David A. Carlson and John E. Savage. Extreme time-space tradeoffs for graphs with small space requirements. *Information Processing Letters*, 14(5):223–227, 1982.

[DT85] Patrick W. Dymond and Martin Tompa. Speedups of deterministic machines by synchronous parallel machines. *Journal of Computer and System Sciences*, 30(2):149–161, April 1985. Preliminary version in *STOC '83*.

[GP14] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC '14)*, pages 847–856, May 2014.

References III

[GPT15] Nicola Galesi, Pavel Pudlák, and Neil Thapen. The space complexity of cutting planes refutations. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC '15)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 433–447, June 2015.

[GPW15] Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15)*, pages 1077–1088, October 2015.

[HN12] Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: Amplifying communication complexity hardness to time-space trade-offs in proof complexity (Extended abstract). In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 233–248, May 2012.

[Kra98] Jan Krajíček. Interpolation by a game. *Mathematical Logic Quarterly*, 44:450–458, 1998.

[Lev73] Leonid A. Levin. Universal'nye perebornye zadachi. *Problemy Peredachi Informatsii*, 9(3):115–116, 1973. In Russian.

References IV

[LT82] Thomas Lengauer and Robert Endre Tarjan. Asymptotically tight bounds on time-space trade-offs in a pebble game. *Journal of the ACM*, 29(4):1087–1130, October 1982. Preliminary version in *STOC '79*.

[Nor12] Jakob Nordström. On the relative strength of pebbling and resolution. *ACM Transactions on Computational Logic*, 13(2):16:1–16:43, April 2012. Preliminary version in *CCC '10*.

[RM99] Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, March 1999. Preliminary version in *FOCS '97*.

[She08] Alexander A. Sherstov. The pattern matrix method for lower bounds on quantum communication. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC '08)*, pages 85–94, May 2008.

[Val75] Leslie G. Valiant. Parallelism in comparison problems. *SIAM Journal on Computing*, 4(3):348–355, March 1975.