The function evaluation problem

\[ f = (x \text{ and } y) \text{ or } (x \text{ and } z) \]

- \( x, y, z \): binary variables
- for some inputs it is possible to evaluate \( f \) without reading all variables

Example:

\( (x, y, z) = (0, 1, 1) \)

It is enough to know the value of \( x \)
The function evaluation problem

\[ f = (x \text{ and } y) \text{ or } (x \text{ and } z) \]

- \( x, y, z \): binary variables
- for some inputs it is possible to evaluate \( f \) without reading all variables

Example:
- \((x, y, z) = (0, 1, 1)\)

It is enough to know the value of \( x \)
The function evaluation problem

Input:

- a function $f$ over the variables $x_1, \ldots, x_n$
- each variable has a positive cost of reading its value

Goal:

- Devise an algorithm for evaluating $f$ which incurs as little cost as possible reading its variables
Applications of the function evaluation problem:

- **Reliability testing / diagnosis**
  
  *Telecommunications: testing connectivity of networks*  
  
  *Manufacturing: testing machines before shipping*

- **Databases**
  
  *Query optimization*

- **Artificial Intelligence**
  
  *Finding optimal derivation strategies in knowledge-based systems*

- **Decision-making strategies** *(AND-OR trees)*

- **Computer-aided game playing** *(for two-player zero-sum games with perfect information, e.g. chess (game trees)*
Algorithms for evaluating $f$

- Dynamically select the next variable based on the values of the variables read so far
- Stop when the value of $f$ is determined

$$f = (x \text{ and } y) \text{ or } (x \text{ and } z)$$
Evasive Functions

- For any possible algorithm, all the variables must be read in the worst case.
- \( f = (x \text{ and } y) \text{ or } (x \text{ and } z) \)
- Some important functions are evasive (e.g. game trees, AND/OR trees and threshold trees).

Worst case analysis cannot distinguish among the performance of different algorithms.
Instead, we use competitive analysis (Charikar et al. 2002)
Evasive Functions

- For any possible algorithm, all the variables must be read in the worst case.
- \( f = (x \text{ and } y) \text{ or } (x \text{ and } z) \)
- Some important functions are evasive (e.g. game trees, AND/OR trees and threshold trees).

- Worst case analysis cannot distinguish among the performance of different algorithms.
- Instead, we use **competitive analysis** (Charikar et al. 2002)
Cost of evaluation

\[ f = (x \quad \text{and} \quad y) \quad \text{or} \quad (x \quad \text{and} \quad z) \]
\[ \text{cost}(x) = 3, \quad \text{cost}(y) = 4, \quad \text{cost}(z) = 5 \]

<table>
<thead>
<tr>
<th>Assignment ((x, y, z))</th>
<th>Value of (f)</th>
<th>Cheapest Proof</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>((0,0,0))</td>
<td>0</td>
<td>{x}</td>
<td>3</td>
</tr>
<tr>
<td>((1,1,0))</td>
<td>1</td>
<td>{x,y}</td>
<td>3+4=7</td>
</tr>
</tbody>
</table>
The competitive ratio

<table>
<thead>
<tr>
<th>((x, y, z))</th>
<th>(f(x, y, z))</th>
<th>Cost of Cheapest Proof</th>
<th>Algorithm Cost</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>((0,0,0))</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>

Cicalese–Milanić Evaluating game trees with priced variables
The competitive ratio

<table>
<thead>
<tr>
<th>(x, y, z)</th>
<th>f(x, y, z)</th>
<th>Cost of Cheapest Proof</th>
<th>Algorithm Cost</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,0,0)</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>(1,1,0)</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>1</td>
</tr>
</tbody>
</table>

![Game tree diagram]

Cicalese–Milanić Evaluating game trees with priced variables
The competitive ratio

<table>
<thead>
<tr>
<th>$(x, y, z)$</th>
<th>$f(x, y, z)$</th>
<th>Cost of Cheapest Proof</th>
<th>Algorithm Cost</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(0,0,0)$</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>$(1,0,0)$</td>
<td>0</td>
<td>9</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>$(0,1,0)$</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>7/3</td>
</tr>
<tr>
<td>$(0,0,1)$</td>
<td>0</td>
<td>3</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>$(1,1,0)$</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>$(1,0,1)$</td>
<td>1</td>
<td>8</td>
<td>12</td>
<td>3/2</td>
</tr>
<tr>
<td>$(0,1,1)$</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>7/3</td>
</tr>
<tr>
<td>$(1,1,1)$</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>1</td>
</tr>
</tbody>
</table>

Cicalese–Milanič
Evaluating game trees with priced variables
The competitive ratio

### Evaluating game trees with priced variables

<table>
<thead>
<tr>
<th>((x, y, z))</th>
<th>(f(x, y, z))</th>
<th>Cost of Cheapest Proof</th>
<th>Algorithm Cost</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,0,0)</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>(1,0,0)</td>
<td>0</td>
<td>9</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>(0,1,0)</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>(7/3)</td>
</tr>
<tr>
<td>(0,0,1)</td>
<td>0</td>
<td>3</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>(1,1,0)</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>(1,0,1)</td>
<td>1</td>
<td>8</td>
<td>12</td>
<td>(3/2)</td>
</tr>
<tr>
<td>(0,1,1)</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>(7/3)</td>
</tr>
<tr>
<td>(1,1,1)</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>1</td>
</tr>
</tbody>
</table>
Measures of algorithm’s performance

**Competitive ratio of algorithm A for \((f, c)\):**

\[
\max_{\text{all assignments } \sigma} \frac{\text{cost of A to evaluate } f \text{ on } \sigma}{\text{cost of cheapest proof of } f \text{ on } \sigma}
\]

**In this talk:**

**Extremal competitive ratio of A for f:**

\[
\max_{\text{all assignments } \sigma, \text{ all cost vectors } c} \frac{\text{cost of A to evaluate } f \text{ on } (\sigma, c)}{\text{cost of cheapest proof of } f \text{ on } (\sigma, c)}
\]
Competitive ratio of algorithm $A$ for $(f, c)$:

$$\max_{\text{all assignments } \sigma} \frac{\text{cost of } A \text{ to evaluate } f \text{ on } \sigma}{\text{cost of cheapest proof of } f \text{ on } \sigma}$$

In this talk:

**Extremal competitive ratio of $A$ for $f$:**

$$\max_{\text{all assignments } \sigma, \text{ all cost vectors } c} \frac{\text{cost of } A \text{ to evaluate } f \text{ on } (\sigma, c)}{\text{cost of cheapest proof of } f \text{ on } (\sigma, c)}$$
Measure of function’s complexity

Extremal competitive ratio of $f$:

\[
\min \left\{ \text{extremal competitive ratio of } A \text{ for } f \right\}
\]

\[
\min \text{ all deterministic algorithms } A \text{ that evaluate } f
\]
The function evaluation problem

**Given:**
a function $f$ over the variables $x_1, x_2, \ldots, x_n$

**Combinatorial Goal:**
- Determine the extremal competitive ratio of $f$

**Algorithmic Goal:**
- Devise an algorithm for evaluating $f$ that:
  1. achieves the optimal (or close to optimal) extremal competitive ratio
  2. is efficient (*runs in time polynomial in the size of $f$*)

The algorithm knows the reading costs.
**The function evaluation problem**

**Given:**
a function $f$ over the variables $x_1, x_2, \ldots, x_n$

**Combinatorial Goal:**
- Determine the extremal competitive ratio of $f$

**Algorithmic Goal:**
- Devise an algorithm for evaluating $f$ that:
  1. achieves the optimal (or close to optimal) extremal competitive ratio
  2. is efficient (*runs in time polynomial in the size of $f$*)

The algorithm knows the reading costs.

Cicalese–Milanič

Evaluating game trees with priced variables
Related work - other models/measures

- **Non-uniform costs & competitive analysis**
  Charikar et al. [STOC 2000, JCSS 2002]

- **Unknown costs**
  Cicalese and Laber [SODA 2006]

- **Restricted costs (selection and sorting)**
  Gupta and Kumar [FOCS 2001], Kannan and Khanna [SODA 2003]

- **Randomized algorithms**
  Snir [TCS 1985], Saks and Wigderson [FOCS 1986], Laber [STACS 2004]

- **Stochastic models**
  Random input, uniform probabilities
  Tarsi [JACM 1983], Boros and Ünlüyurt [AMAI 1999]
  Charikar et al. [STOC 2000, JCSS 2002], Greiner et al. [AI 2005]

  Random input, arbitrary probabilities
  Kaplan et al. [STOC 2005]

  Random costs
  Angelov et al. [LATIN 2008]
How to evaluate general functions?

Good algorithms are expected to test...

- cheap variables - well defined
- important variables ???

Cicalese and Laber created a linear program that captures the impact of the variables
The minimal proofs

$f$ - a function over $V = \{x_1, \ldots, x_n\}$

$R$ - range of $f$

**Definition**

Let $r \in R$. A **minimal proof** for $f(x) = r$ is a minimal set of variables $P \subseteq V$ such that *there is an assignment* $\sigma$ *of values* *to the variables* *in* $P$ *such that* $f_\sigma$ *is constantly equal to* $r$.

**Example:** $f(x_1, x_2, x_3) = (x_1 \text{ and } x_2) \text{ or } (x_1 \text{ and } x_3)$, $R = \{0, 1\}$

minimal proofs for $f(x) = 1$: $\{\{x_1, x_2\}, \{x_1, x_3\}\}$

minimal proofs for $f(x) = 0$: $\{\{x_1\}, \{x_2, x_3\}\}$
The minimal proofs

\( f \) - a function over \( V = \{ x_1, \ldots, x_n \} \)

\( R \) - range of \( f \)

**Definition**

Let \( r \in R \). A **minimal proof** for \( f(x) = r \) is a minimal set of variables \( P \subseteq V \) such that there is an assignment \( \sigma \) of values to the variables in \( P \) such that \( f_\sigma \) is constantly equal to \( r \).

Example: \( f(x_1, x_2, x_3) = (x_1 \text{ and } x_2) \text{ or } (x_1 \text{ and } x_3) \), \( R = \{0, 1\} \)

minimal proofs for \( f(x) = 1 \): \( \{ \{x_1, x_2\}, \{x_1, x_3\} \} \)

minimal proofs for \( f(x) = 0 \): \( \{ \{x_1\}, \{x_2, x_3\} \} \)
The Linear Program by Cicalese-Laber

\[ \text{LP}(f) \begin{align*}
\text{Minimize} & \quad \sum_{x \in V} s(x) \\
\text{s.t.} & \quad \sum_{x \in P} s(x) \geq 1 \quad \text{for every minimal proof } P \text{ of } f \\
& \quad s(x) \geq 0 \quad \text{for every } x \in V
\end{align*} \]

Intuitively, \( s(x) \) measures the \textbf{impact} of variable \( x \).

The feasible solutions to the \( \text{LP}(f) \) are precisely the fractional hitting sets of the set of minimal proofs of \( f \).
Minimize $\sum_{x \in V} s(x)$

s.t.

$\sum_{x \in P} s(x) \geq 1$ for every minimal proof $P$ of $f$

$s(x) \geq 0$ for every $x \in V$

Intuitively, $s(x)$ measures the impact of variable $x$.

The feasible solutions to the $\text{LP}(f)$ are precisely the fractional hitting sets of the set of minimal proofs of $f$. 
LP($f$): example

\[ f(x_1, x_2, x_3) = (x_1 \text{ and } x_2) \text{ or } (x_1 \text{ and } x_3) \]

minimal proofs for $f = 1$: \{\{x_1, x_2\}, \{x_1, x_3\}\}

minimal proofs for $f = 0$: \{\{x_1\}, \{x_2, x_3\}\}

\[
\text{LP}(f) \begin{cases} 
\text{Minimize } s_1 + s_2 + s_3 \\
\text{s.t.} \\
\quad s_1 + s_2 \geq 1 \\
\quad s_1 + s_3 \geq 1 \\
\quad s_1 \geq 1 \\
\quad s_2 + s_3 \geq 1 \\
\quad s_1, s_2, s_3 \geq 0 
\end{cases}
\]

Optimal solution: $s = (1, 1/2, 1/2)$
**LPA**(*f*: function)

While the value of *f* is unknown

Select a feasible solution *s*() of **LP**(*f*)

Read the variable *u* which minimizes \( c(x)/s(x) \) (cost/impact)

\[
c(x) = c(x) - s(x)c(u)/s(u)
\]

\[f \leftarrow \text{restriction of } f \text{ after reading } u\]

End While
The LPA bounds the extremal competitive ratio

The selection of solution $s$ determines both the **computational efficiency** and the **performance** (extremal competitive ratio) of the algorithm.

### Key Lemma (Cicalese-Laber 2006)

Let $K$ be a positive number. If

\[
\text{ObjectiveFunctionValue}(s) \leq K,
\]

for every selected solution $s$

then

\[
\text{ExtremalCompetitiveRatio}(f) \leq K.
\]
The fractional cover number of a function $f$ is defined as

$$\Delta(f) = \max_{f' \text{ restriction of } f} \left\{ \text{opt-value}(\text{LP}(f')) \right\}$$

**Key Lemma (Cicalese-Laber 2006)**

For every function $f$,

$$\text{ExtremalCompetitiveRatio}(f) \leq \Delta(f).$$
Cross-intersecting families

- cross-intersecting: $A \in \mathcal{A}, B \in \mathcal{B} \Rightarrow A \cap B \neq \emptyset$
$f : S_1 \times \cdots \times S_n \to S$, a function over $V = \{x_1, \ldots, x_n\}$

$R$ - range of $f$

For $r \in R$, let $\mathcal{P}(r)$ denote the set of \textbf{minimal proofs for $f(x) = r$}.

Then:

- each $\mathcal{P}(r)$ is a non-empty Sperner family
- for every $r \neq r'$, the families $\mathcal{P}(r)$ and $\mathcal{P}(r')$ are cross-intersecting
The Linear Program by Cicalese-Laber

\[ \text{LP}(f) \begin{cases} 
\text{Minimize} & \sum_{x \in V} s(x) \\
\text{s.t.} & \sum_{x \in P} s(x) \geq 1 \quad \text{for every } P \in \mathcal{P} \\
& s(x) \geq 0 \quad \text{for every } x \in V
\end{cases} \]

\[ \mathcal{P} = \bigcup_{r \in \mathcal{R}} \mathcal{P}(r) \]

union of pairwise cross-intersecting Sperner families

For every function \( f : S_1 \times \cdots \times S_n \to S \), the \( \text{LP}(f) \) seeks a minimal fractional hitting set of a union of pairwise cross-intersecting Sperner families.
Cross-intersecting lemma

Cross-Intersecting Lemma (Cicalese-Laber 2008)

Let $A$ and $B$ be two non-empty cross-intersecting families of subsets of $V$.

Then, there is a fractional hitting set $s$ of $A \cup B$ such that

$$\|s\|_1 = \sum_{x \in V} s(x) \leq \max\{|P| : P \in A \cup B\}.$$
Applications of the cross-intersecting lemma

1. \( f : S_1 \times \cdots \times S_n \rightarrow S \), nonconstant:
   \[ \text{ExtremalCompetitiveRatio}(f) \leq \text{PROOF}(f) \] [Cicalese-Laber 2008]
   \( (\text{PROOF}(f) = \text{size of the largest minimal proof of } f) \)

2. Monotone Boolean functions:
   \[ \text{ExtremalCompetitiveRatio}(f) = \text{PROOF}(f) \] [Cicalese-Laber 2008]

3. Game trees: \( \text{ExtremalCompetitiveRatio}(f) \leq TBA \)
1. $f : S_1 \times \cdots \times S_n \rightarrow S$, nonconstant:

$\text{ExtremalCompetitiveRatio}(f) \leq \text{PROOF}(f)$ [Cicalese-Laber 2008]

($\text{PROOF}(f)$ = size of the largest minimal proof of $f$)

2. Monotone Boolean functions:

$\text{ExtremalCompetitiveRatio}(f) = \text{PROOF}(f)$ [Cicalese-Laber 2008]

3. Game trees: $\text{ExtremalCompetitiveRatio}(f) \leq \text{TBA}$
1. $f : S_1 \times \cdots \times S_n \rightarrow S$, nonconstant:

$\text{ExtremalCompetitiveRatio}(f) \leq PROOF(f)$ [Cicalese-Laber 2008]

$(PROOF(f) = \text{size of the largest minimal proof of } f)$

2. Monotone Boolean functions:

$\text{ExtremalCompetitiveRatio}(f) = PROOF(f)$ [Cicalese-Laber 2008]

3. Game trees: $\text{ExtremalCompetitiveRatio}(f) \leq TBA$
Game trees

game tree
Game trees
Game trees

Cicalese–Milanič
Evaluating game trees with priced variables
Game trees

game tree
Game trees

\[
\begin{array}{c}
\text{MIN} \quad 2 \\
\text{MAX} \quad 8 \\
\text{MIN} \quad 6 \\
\text{MAX} \quad 8 \\
\text{MIN} \quad 5 \\
\text{MAX} \quad 4 \\
\text{MIN} \quad 5 \\
\text{MAX} \quad 9 \\
\text{MIN} \quad 5
\end{array}
\]
**Minterm**: minimal set $A \subseteq V$ of variables such that

\[
\text{value of } f \geq \text{value of } A := \min \text{ value of variables in } A
\]

Minterms can prove a lower bound for the value of $f$. 

```
x_3
x_1
x_2
```

```
MIN
  MIN
    MIN
      MIN
        MAX
          MIN
            MAX
              MAX
```

Cicalese–Milanič Evaluating game trees with priced variables
**Minterm**: minimal set $A \subseteq V$ of variables such that

\[
\text{value of } f \geq \text{value of } A := \min \text{ value of variables in } A
\]

Minterms can prove a lower bound for the value of $f$. 
**Minterm**: minimal set $A \subseteq V$ of variables such that

$$\text{value of } f \geq \text{value of } A := \text{min value of variables in } A$$

Minterms can prove a lower bound for the value of $f$. 

[Diagram of a game tree with variables and operators.]
**Maxterm**: minimal set $B \subseteq V$ of variables such that

$$\text{value of } f \leq \text{value of } B := \max \text{ value of variables in } B$$

Maxterms can prove an upper bound for the value of $f$. 

![Game tree diagram](diagram.png)
Maxterms of game trees

**Maxterm**: minimal set $B \subseteq V$ of variables such that

$$\text{value of } f \leq \text{value of } B := \text{max value of variables in } B$$

Maxterms can prove an upper bound for the value of $f$. 

![Diagram of a game tree with nodes labeled by variables and operators MIN and MAX.](image)
Lower bound for the extremal competitive ratio

- \( k(f) = \max\{|A| : A \text{ minterm of } f \} \)
- \( l(f) = \max\{|B| : B \text{ maxterm of } f \} \)

**Theorem (Cicalese-Laber 2005)**

Let \( f \) be a game tree with no minterms or maxterms of size 1. Then,

\[
\text{ExtremalCompetitiveRatio}(f) \geq \max\{k(f), l(f)\}.
\]
To prove that the value of $f$ is $b$, we need:

- a minterm of value $b$ \([\text{proves } f \geq b]\)
- a maxterm of value $b$ \([\text{proves } f \leq b]\)

Every minimal proof = union of a minterm and a maxterm
A first upper bound

It follows that

\[ \text{PROOF}(f) = \text{size of the largest minimal proof of } f \]
\[ = k(f) + l(f) - 1. \]

**Theorem (Cicalese-Laber 2008)**

\[ f : S_1 \times \cdots \times S_n \to S, \text{ nonconstant:} \]
\[ \text{ExtremalCompetitiveRatio}(f) \leq \text{PROOF}(f) \]

For a game tree \( f \),
\[ \text{ExtremalCompetitiveRatio}(f) \leq k(f) + l(f) - 1. \]

Lower bound: \( \max\{k(f), l(f)\} \)
A first upper bound

It follows that

\[ \text{PROOF}(f) = \text{size of the largest minimal proof of } f \]
\[ = k(f) + l(f) - 1. \]

Theorem (Cicalese-Laber 2008)

\[ f : S_1 \times \cdots \times S_n \to S, \text{ nonconstant:} \]
\[ \text{ExtremalCompetitiveRatio}(f) \leq \text{PROOF}(f) \]

For a game tree \( f \),
\[ \text{ExtremalCompetitiveRatio}(f) \leq k(f) + l(f) - 1. \]

Lower bound: \( \max\{k(f), l(f)\} \)
A first upper bound

It follows that

\[ PROOF(f) = \text{size of the largest minimal proof of } f \]
\[ = k(f) + l(f) - 1. \]

Theorem (Cicalese-Laber 2008)

\[ f : S_1 \times \cdots \times S_n \to S, \text{ nonconstant:} \]
\[ ExtremalCompetitiveRatio(f) \leq PROOF(f) \]

For a game tree \( f \),
\[ ExtremalCompetitiveRatio(f) \leq k(f) + l(f) - 1. \]

Lower bound: \[ \max\{k(f), l(f)\} \]
Yes:

**Claim**

For every restriction $f'$ of $f$, there is a fractional hitting set $s$ of the set of minimal proofs of $f'$ such that

$$\|s\|_1 \leq \max\{k(f), l(f)\}.$$

$$\Delta(f) = \max_{f' \text{ restriction of } f} \left\{ \text{opt-value}(\text{LP}(f')) \right\} \leq \max\{k(f), l(f)\}$$

By the Key Lemma,

$$\text{ExtremalCompetitiveRatio}(f) \leq \max\{k(f), l(f)\}$$
Can we close the gap?

Yes:

**Claim**

For every restriction $f'$ of $f$, there is a fractional hitting set $s$ of the set of minimal proofs of $f'$ such that

$$\|s\|_1 \leq \max\{k(f), l(f)\}.$$  

$$\Delta(f) = \max_{f' \text{ restriction of } f} \left\{ \text{opt-value}(\text{LP}(f')) \right\} \leq \max\{k(f), l(f)\}$$

By the **Key Lemma**,  

$$\text{ExtremalCompetitiveRatio}(f) \leq \max\{k(f), l(f)\}$$
Restrictions of game trees

Restriction of $f$

$$f'(x_3, x_5, x_7, x_8, x_9)$$
Restrictions of game trees

\[ \tilde{f}(x_3, x_5, x_7, x_8, x_9) \geq 6 \]
Restrictions of game trees

\[ \tilde{f}(x_3, x_5, x_7, x_8, x_9) \leq 15 \]
Restrictions of game trees

\[ \tilde{f}(x_3, x_5, x_7, x_8, x_9) \in [6, 15] = [LB, UB] \]
Minimal proofs of a restriction of a game tree

maxterm proofs

\[ b = LB \]

\[ b = UB \]

minterm proofs

combined proofs

\[ LB \leq b \leq UB \]
Case 1: No maxterm has been fully evaluated yet

maxterm proofs

minterm proofs

combined proofs
Let $s$ be (the characteristic vector of) a minimal hitting set of the shaded sets.

$$\|s\|_1 \leq k(f) \leq \max\{k(f), l(f)\}$$
Case 2: There is a fully evaluated maxterm

maxterm proofs

minterm proofs

combined proofs
Case 2: There is a fully evaluated maxterm

\[ x_7 \land x_9 \land x_8 \land \overline{x_3} \land \overline{x_7} \land x_9 \land \overline{x_7} \land x_8 \land \overline{x_7} \land x_8 \]

\[ \overline{x_7} \land \overline{x_9} \land x_3 \land x_7 \land x_9 \land \overline{x_7} \land x_8 \land \overline{x_7} \land x_8 \]

**R**: the family of the minterm proofs

**B**: the family of the maxterm parts of the non-minterm proofs
Case 2: There is a fully evaluated maxterm

- $R$ and $B$ are non-empty sets
- $R$ and $B$ are cross-intersecting
- every minimal proof contains a member of $R \cup B$

By the Cross-intersecting lemma, there exists a feasible solution $s$ to the LP($f'$) such that

$$\|s\|_1 \leq \max\{|P| : P \in R \cup B\} \leq \max\{k(f), l(f)\}.$$

Putting together the two cases, it follows that

$$\Delta(f) \leq \max\{k(f), l(f)\}.$$
Case 2: There is a fully evaluated maxterm

- $R$ and $B$ are non-empty sets
- $R$ and $B$ are cross-intersecting
- every minimal proof contains a member of $R \cup B$

By the **Cross-intersecting lemma**, there exists a feasible solution $s$ to the $\text{LP}(f')$ such that

$$\|s\|_1 \leq \max\{|P| : P \in R \cup B\} \leq \max\{k(f), l(f)\}.$$  

Putting together the two cases, it follows that

$$\Delta(f) \leq \max\{k(f), l(f)\}.$$
Case 2: There is a fully evaluated maxterm

- \( R \) and \( B \) are non-empty sets
- \( R \) and \( B \) are cross-intersecting
- every minimal proof contains a member of \( R \cup B \)

By the **Cross-intersecting lemma**, there exists a feasible solution \( s \) to the \( \text{LP}(f') \) such that

\[
\|s\|_1 \leq \max\{|P| : P \in R \cup B\} \leq \max\{k(f), l(f)\}.
\]

Putting together the two cases, it follows that

\[
\Delta(f) \leq \max\{k(f), l(f)\}
\]
Let \( f \) be a game tree with no minterms or maxterms of size 1. Then,

\[
\text{ExtremalCompetitiveRatio}(f) = \max\{k(f), l(f)\}.
\]

Let \( f \) be a game tree. Let \( p \) (\( q \)) denote the number of minterms (maxterms) of \( f \) of size 1. Then

\[
\text{ExtremalCompetitiveRatio}(f) = \begin{cases} 
\max\{k(f), l(f)\}, & \text{if } p = q = 0 \text{ or } p = q = 1; \\
\max\{k(f), l(f) - p\}, & \text{if } p \geq 1 \text{ and } q = 0; \\
\max\{k(f) - q, l(f)\}, & \text{if } p = 0 \text{ and } q \geq 1.
\end{cases}
\]

There is a polynomial-time algorithm for evaluating game trees with optimal extremal competitiveness.
The extremal competitive ratio for game trees

Theorem

Let $f$ be a game tree with no minterms or maxterms of size 1. Then,

$$\text{ExtremalCompetitiveRatio}(f) = \max\{k(f), l(f)\}.$$ 

Theorem

Let $f$ be a game tree. Let $p$ ($q$) denote the number of minterms (maxterms) of $f$ of size 1. Then

$$\text{ExtremalCompetitiveRatio}(f) = \begin{cases} 
\max\{k(f), l(f)\}, & \text{if } p = q = 0 \text{ or } p = q = 1; \\
\max\{k(f), l(f) - p\}, & \text{if } p \geq 1 \text{ and } q = 0; \\
\max\{k(f) - q, l(f)\}, & \text{if } p = 0 \text{ and } q \geq 1.
\end{cases}$$

There is a polynomial-time algorithm for evaluating game trees with optimal extremal competitiveness.
Suppose that the cost of reading a variable can depend on the variable’s value:

\[ c(x) = \begin{cases} 
50, & \text{if } x = 0; \\
1000, & \text{if } x = 1.
\end{cases} \]

**Theorem**

Let \( f \) be a monotone Boolean function or a game tree. Then,

\[
\text{ExtremalCompetitiveRatio}(f, r) = r \cdot \text{ECR}(f) - r + 1,
\]

where

\[
r = \max_{x \in V} \frac{c_{\max}(x)}{c_{\min}(x)}.
\]
Suppose that the cost of reading a variable can depend on the variable’s value:

\[ c(x) = \begin{cases} 
50, & \text{if } x = 0; \\
1000, & \text{if } x = 1. 
\end{cases} \]

**Theorem**

Let \( f \) be a monotone Boolean function or a game tree. Then,

\[ \text{ExtremalCompetitiveRatio}(f, r) = r \cdot \text{ECR}(f) - r + 1, \]

where

\[ r = \max_{x \in V} \frac{c_{\text{max}}(x)}{c_{\text{min}}(x)}. \]
LPA has a very broad applicability

LPA does not depend on the structure of $f$

It can be used to derive upper bounds on the extremal competitive ratios of very different functions:

- $f =$ minimum of a list: $ExtremalCompetitiveRatio(f) \leq n - 1$ [Cicalese-Laber 2005]

- $f =$ the sorting function: $ExtremalCompetitiveRatio(f) \leq n - 1$ [Cicalese-Laber 2008]

- $f : S_1 \times \cdots \times S_n \rightarrow S$, nonconstant: $ExtremalCompetitiveRatio(f) \leq PROOF(f)$ [Cicalese-Laber 2008]

- $f =$ monotone Boolean function: $ExtremalCompetitiveRatio(f) = PROOF(f)$ [Cicalese-Laber 2008]

- $f =$ game tree: $ExtremalCompetitiveRatio(f) \leq \max \{k(f), l(f)\}$
We have seen:

- the **Linear Programming Approach** for the development of competitive algorithms for the function evaluation problem,
- the related combinatorial notion of the **fractional cover number**,
- the **extremal competitive ratio** for **game trees**, and
- the more general model of **value dependent costs**.
Some Open Questions

- Is the extremal competitive ratio always integer?
- Find the extremal comp. ratio of general Boolean functions.
- Is there a polynomial algorithm with optimal extremal comp. ratio for evaluating monotone Boolean functions (given by an oracle/by the list of minterms)?
Some Open Questions

- Is the extremal competitive ratio always integer?
- Find the extremal comp. ratio of general Boolean functions.
- Is there a polynomial algorithm with optimal extremal comp. ratio for evaluating monotone Boolean functions (given by an oracle/by the list of minterms)?
Some Open Questions

- Is the extremal competitive ratio always integer?

- Find the extremal comp. ratio of general Boolean functions.

- Is there a polynomial algorithm with optimal extremal comp. ratio for evaluating monotone Boolean functions (given by an oracle/by the list of minterms)?
THANK YOU