Evaluating game trees in the priced information model

Ferdinando Cicalese Martin Milanič

AG Genominformatik
Technische Fakultät
Universität Bielefeld

Workshop on Discrete Mathematics, TU Wien
November 21, 2008

The function evaluation problem

$$f = (x \textit{ and } y) \textit{ or } (x \textit{ and } z)$$

- 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

Applications of the function evaluation problem:

- Reliability testing / diagnosis

  Telecommunications: testing connectivity of networks

  Manufacturing: testing machines before shipping

- Databases

  Query optimization

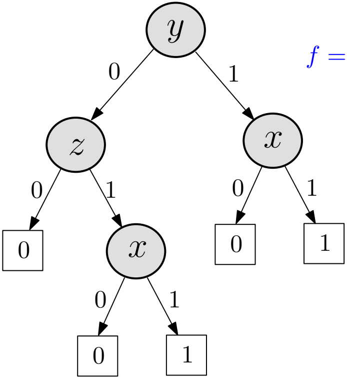
- Artificial Intelligence

  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



$f = (x \; and \; y) \; or \; (x \; and \; z)$

Evaluation measure

Evasive Functions

- For any possible algorithm, all the variables must be read in the worst case.
- $f = (x \textit{ and } y) \textit{ or } (x \textit{ and } z)$
- 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)

Evaluation measure

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)

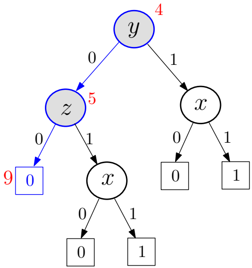
Cost of evaluation

$f = (x \textit{ and } y) \textit{ or } (x \textit{ and } z)$
$cost(x) = 3$, $cost(y) = 4$, $cost(z) = 5$

Asignment (x, y, z)	Value of f	Cheapest Proof	Cost
(0,0,0)	0	{x}	3
(1,1,0)	1	{x,y}	3+4=7

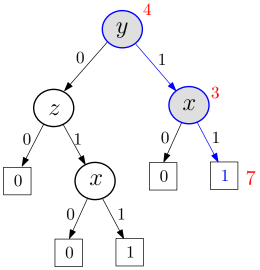
The competitive ratio

(x,y,z)	$f(x,y,z)$	Cost of Cheapest Proof	Algorithm Cost	Ratio
(0,0,0)	0	3	9	3



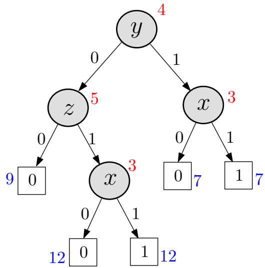
The competitive ratio

(x, y, z)	$f(x, y, z)$	Cost of Cheapest Proof	Algorithm Cost	Ratio
(0,0,0)	0	3	9	3
(1,1,0)	1	7	7	1



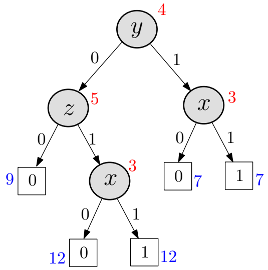
The competitive ratio

(x, y, z)	$f(x, y, z)$	Cost of Cheapest Proof	Algorithm Cost	Ratio
(0,0,0)	0	3	9	3
(1,0,0)	0	9	9	1
(0,1,0)	0	3	7	7/3
(0,0,1)	0	3	12	4
(1,1,0)	1	7	7	1
(1,0,1)	1	8	12	3/2
(0,1,1)	0	3	7	7/3
(1,1,1)	1	7	7	1



The competitive ratio

(x, y, z)	$f(x, y, z)$	Cost of Cheapest Proof	Algorithm Cost	Ratio
(0,0,0)	0	3	9	3
(1,0,0)	0	9	9	1
(0,1,0)	0	3	7	7/3
(0,0,1)	**0**	**3**	**12**	**4**
(1,1,0)	1	7	7	1
(1,0,1)	1	8	12	3/2
(0,1,1)	0	3	7	7/3
(1,1,1)	1	7	7	1



Measures of algorithm's performance

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*:

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

Measures of algorithm's performance

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*:

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

Measure of function's complexity

Extremal competitive ratio of f :

$$\min_{\substack{\textit{all deterministic algorithms } A \\ \textit{that evaluate } f}} \left\{ \textit{extremal competitive ratio of } A \textit{ for } f \right\}$$

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.

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 σ of values to the variables in P such that f_σ 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* σ *of values to the variables in* P *such that* f_σ *is constantly equal to* r.

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

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

$$\textbf{LP}(f) \left\{ \begin{array}{llll} \textit{Minimize } \sum_{x \in V} s(x) & & & \\ \textit{s.t.} & & & \\ \sum_{x \in P} s(x) & \geq & 1 & \text{for every minimal proof } P \text{ of } f \\ \quad s(x) & \geq & 0 & \text{for every } x \in V \end{array} \right.$$

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

The feasible solutions to the **LP**(f) are precisely the fractional hitting sets of the set of minimal proofs of f.

The Linear Program by Cicalese-Laber

$$\mathbf{LP}(f) \begin{cases} \textit{Minimize } \sum_{x \in V} s(x) \\ \textit{s.t.} \\ \sum_{x \in P} s(x) \geq 1 \quad \text{for every minimal proof } P \text{ of } f \\ \quad s(x) \geq 0 \quad \text{for every } x \in V \end{cases}$$

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

The feasible solutions to the **LP**(f) are precisely the fractional hitting sets of the set of minimal proofs of f.

LP(f): example

$f(x_1, x_2, x_3) = (x_1 \textit{ and } x_2) \textit{ or } (x_1 \textit{ and } x_3)$

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

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

$$\textbf{LP}(f) \begin{cases} \textit{Minimize } s_1 + s_2 + s_3 \\ \textit{s.t.} \\ \quad s_1 + s_2 \geq 1 \\ \quad s_1 + s_3 \geq 1 \\ \quad s_1 \geq 1 \\ \quad s_2 + s_3 \geq 1 \\ s_1, s_2, s_3 \geq 0 \end{cases}$$

Optimal solution: $s = (1, 1/2, 1/2)$

The Linear Programming Approach

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)

$c(x) = c(x) - s(x)c(u)/s(u)$

$f \leftarrow$ restriction of f 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 $\mathbf{K}$ *be a positive number. If*

$$\mathit{ObjectiveFunctionValue}(s) \leq \mathbf{K}\,,$$

for every selected solution s

then

$$\mathit{ExtremalCompetitiveRatio}(f) \leq \mathbf{K}\,.$$

The fractional cover number

The **fractional cover number** of a function f is defined as

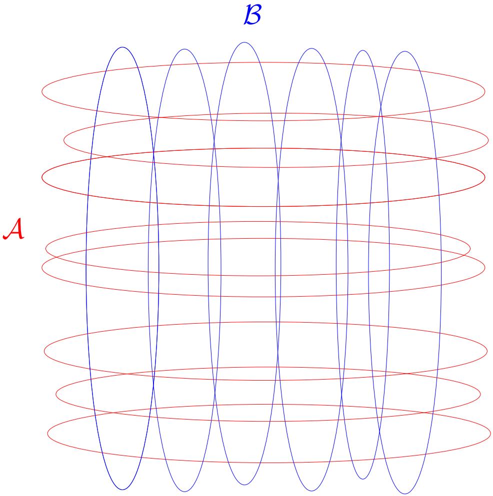
$$\Delta(f) = \max_{f' \text{ restriction of } f} \left\{ \textit{opt-value}(\mathbf{LP}(\mathbf{f'})) \right\}$$

Key Lemma (Cicalese-Laber 2006)

For every function f,

$$\textit{ExtremalCompetitiveRatio}(f) \leq \Delta(f)\,.$$

Cross-intersecting families



- cross-intersecting: $A \in \mathcal{A}, B \in \mathcal{B} \Rightarrow A \cap B \neq \emptyset$

Minimal proofs and cross-intersecting families

$f : S_1 \times \cdots \times S_n \to S$, a function over $V = \{x_1, \dots, x_n\}$

R - range of f

For $r \in R$, let $\mathcal{P}(r)$ denote the set of **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

$$\textbf{LP}(f) \left\{ \begin{array}{lcll} \multicolumn{4}{l}{\textit{Minimize} \sum_{x\in V} s(x)} \\ \multicolumn{4}{l}{\textit{s.t.}} \\ \sum_{x\in P} s(x) & \geq & 1 & \text{for every } P \in \mathcal{P} \\ s(x) & \geq & 0 & \text{for every } x \in V \end{array} \right.$$

$$\mathcal{P} = \cup_{r\in R}\mathcal{P}(r)$$

union of pairwise cross-intersecting Sperner families

For every function $f : S_1 \times \cdots \times S_n \to S$, the $\textbf{LP}(f)$ 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 $\mathcal{A}$ and $\mathcal{B}$ be two non-empty **cross-intersecting** *families of subsets of V.*
Then, there is a fractional hitting set s of $\mathcal{A} \cup \mathcal{B}$ such that

$$\|s\|_1 = \sum_{x \in V} s(x) \leq \max\{|P| : P \in \mathcal{A} \cup \mathcal{B}\}\,.$$

- geometric proof
- generalizes to any number of pairwise cross-intersecting families

Applications of the cross-intersecting lemma

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

2. Monotone Boolean functions:
   $ExtremalCompetitiveRatio(f) = PROOF(f)$ [Cicalese-Laber 2008]

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

Applications of the cross-intersecting lemma

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

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

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

Applications of the cross-intersecting lemma

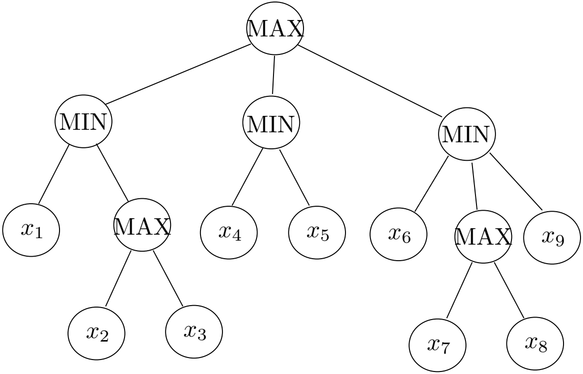
1. $f : S_1 \times \cdots \times S_n \rightarrow S$, nonconstant:
   $ExtremalCompetitiveRatio(f) \leq PROOF(f)$ [Cicalese-Laber 2008]

   ($PROOF(f) =$ size of the largest minimal proof of f)

2. Monotone Boolean functions:
   $ExtremalCompetitiveRatio(f) = PROOF(f)$ [Cicalese-Laber 2008]

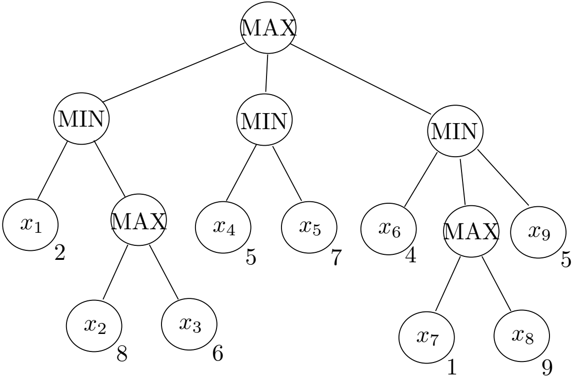
3. Game trees: $ExtremalCompetitiveRatio(f) \leq TBA$

Game trees



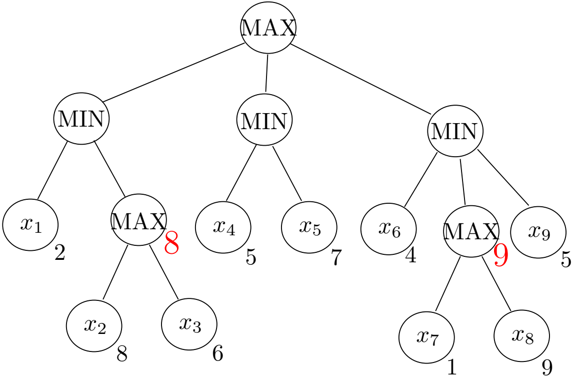
game tree

Game trees



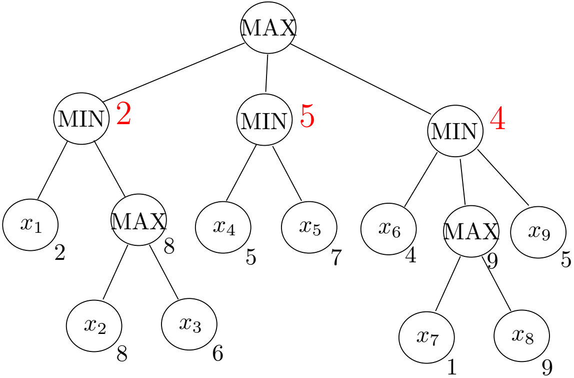
game tree

Game trees



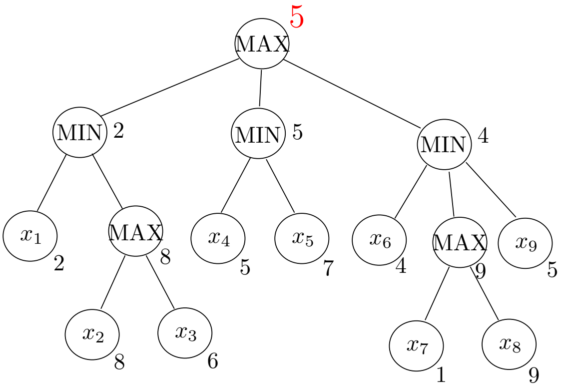
game tree

Game trees



game tree

Game trees



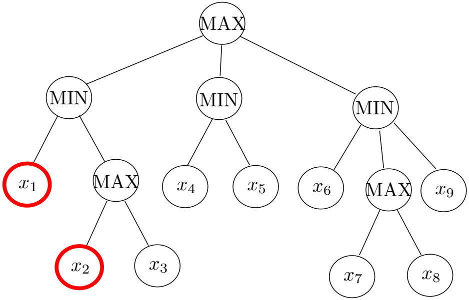
game tree

Minterms of game trees

Minterm: minimal set $A \subseteq V$ of variables such that

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

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

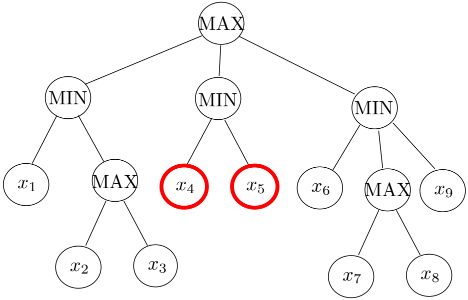


Minterms of game trees

Minterm: minimal set $A \subseteq V$ of variables such that

value of f $\geq$ *value of A* $:=$ *min value of variables in A*

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

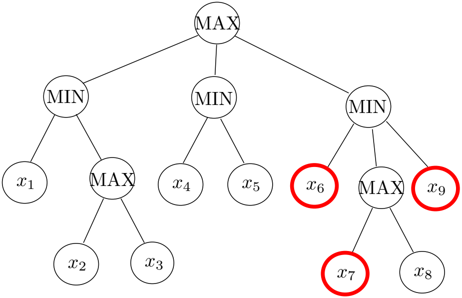


Minterms of game trees

Minterm: minimal set $A \subseteq V$ of variables such that

value of f $\geq$ *value of A* $:=$ *min value of variables in A*

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

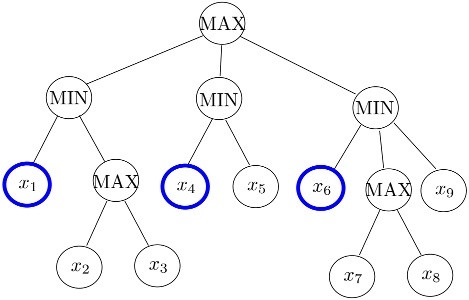


Maxterms of game trees

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

value of f $\leq$ *value of B* $:=$ *max value of variables in B*

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

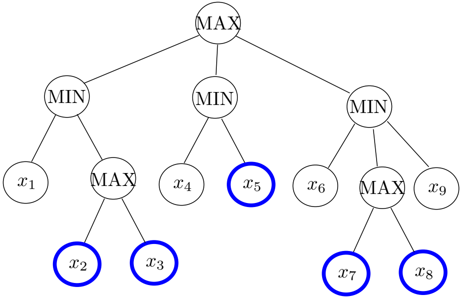


Maxterms of game trees

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

value of f $\leq$ *value of B* $:=$ *max value of variables in B*

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



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,

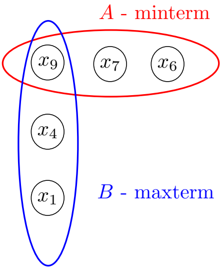
$$ExtremalCompetitiveRatio(f) \geq \max\{k(f), l(f)\} .$$

Minimal proofs of game trees

To prove that the value of *f* is **b**, we need:

- a minterm of value **b** [*proves* $f \geq$ **b**]
- a maxterm of value **b** [*proves* $f \leq$ **b**]

Every minimal proof = union of a minterm and a maxterm



A first upper bound

It follows that

$PROOF(f)$ = size of the largest minimal proof of f
$= k(f) + l(f) - 1.$

Theorem (Cicalese-Laber 2008)

$f : S_1 \times \cdots \times S_n \to S$, nonconstant:
$ExtremalCompetitiveRatio(f) \leq PROOF(f)$

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

Lower bound: $\max\{k(f), l(f)\}$

A first upper bound

It follows that

$PROOF(f)$ = size of the largest minimal proof of f
$= k(f) + l(f) - 1.$

Theorem (Cicalese-Laber 2008)

$f : S_1 \times \cdots \times S_n \to S$, nonconstant:
$ExtremalCompetitiveRatio(f) \leq PROOF(f)$

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

Lower bound: $\max\{k(f), l(f)\}$

A first upper bound

It follows that

$PROOF(f) =$ size of the largest minimal proof of f
$= k(f) + l(f) - 1.$

Theorem (Cicalese-Laber 2008)

$f : S_1 \times \cdots \times S_n \to S$, nonconstant:
$ExtremalCompetitiveRatio(f) \leq PROOF(f)$

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

Lower bound: $\max\{k(f), l(f)\}$

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

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

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

By the **Key Lemma**,

$$\textit{ExtremalCompetitiveRatio}(f) \leq \max\{k(f), l(f)\}$$

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

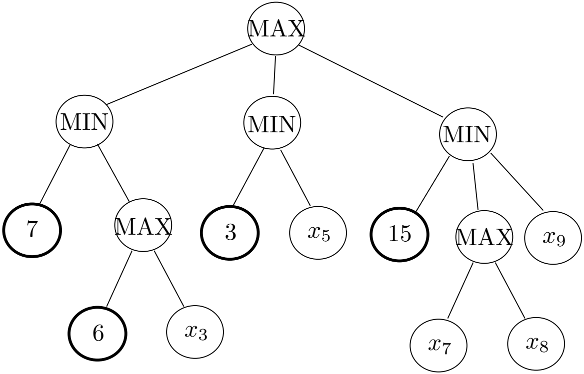
$$\|s\|_1 \leq \max\{k(f), l(f)\}\,.$$

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

By the **Key Lemma**,

$$\textit{ExtremalCompetitiveRatio}(f) \leq \max\{k(f), l(f)\}$$

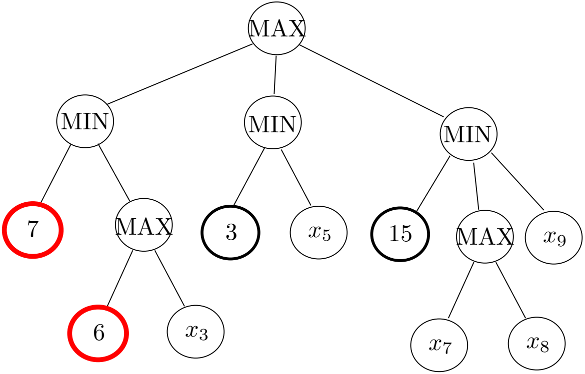
Restrictions of game trees



restriction of f

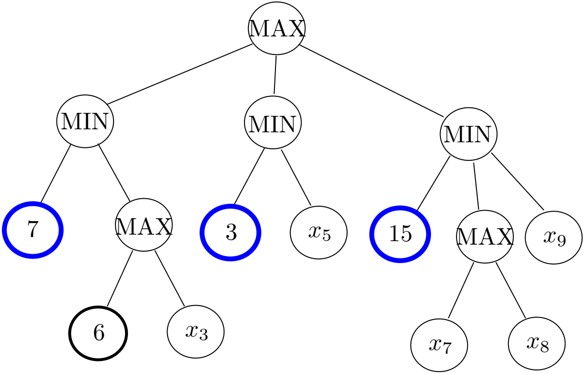
$$f'(x_3, x_5, x_7, x_8, x_9)$$

Restrictions of game trees



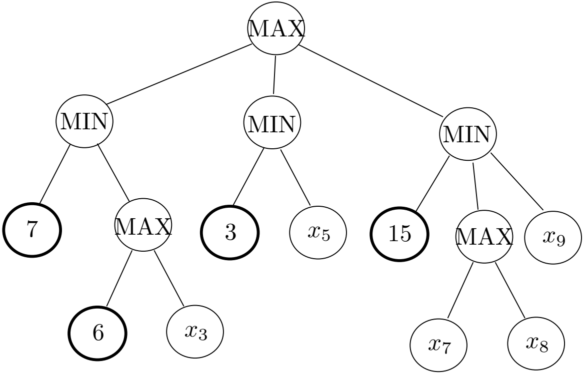
$$\tilde{f}(x_3, x_5, x_7, x_8, x_9) \geq 6$$

Restrictions of game trees



$$\tilde{f}(x_3, x_5, x_7, x_8, x_9) \leq 15$$

Restrictions of game trees

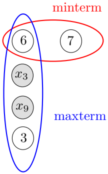


$$\tilde{f}(x_3, x_5, x_7, x_8, x_9) \in [6, 15] = [LB, UB]$$

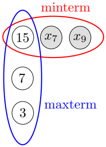
Minimal proofs of a restriction of a game tree

maxterm proofs

$b = LB$

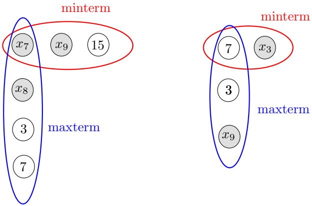


minterm proofs



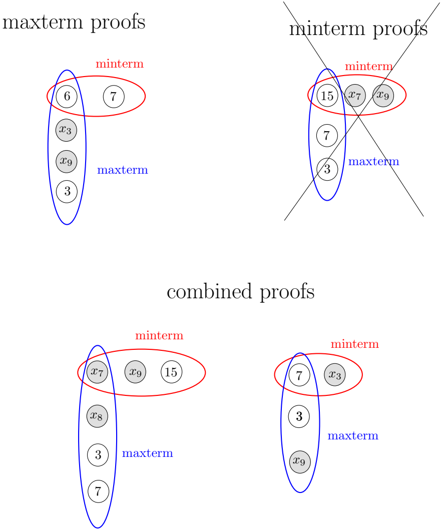
$b = UB$

combined proofs

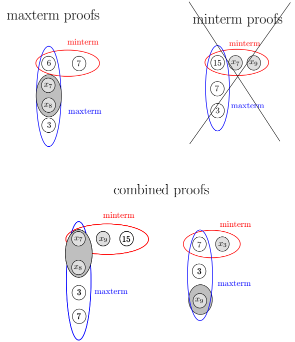


$$LB \leq b \leq UB$$

Case 1: No maxterm has been fully evaluated yet



Case 1: No maxterm has been fully evaluated yet

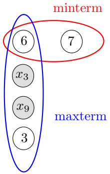


Let s be (the characteristic vector of) **a minimal hitting set of the shaded sets**.

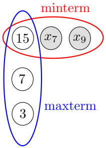
$$\|s\|_1 \leq k(f) \leq \max\{k(f), l(f)\}$$

Case 2: There is a fully evaluated maxterm

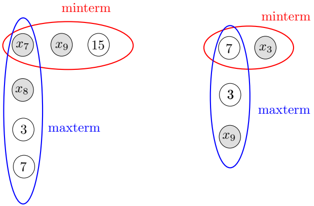
maxterm proofs



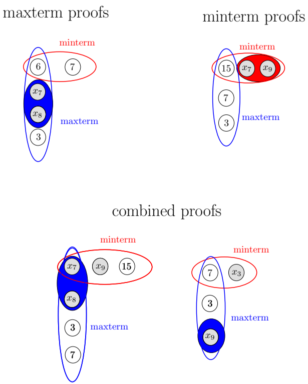
minterm proofs



combined proofs



Case 2: There is a fully evaluated maxterm



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 $\mathbf{R} \cup \mathbf{B}$

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

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

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 $\mathbf{R} \cup \mathbf{B}$

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

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

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 $\mathbf{R} \cup \mathbf{B}$

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

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

Putting together the two cases, it follows that

$$\Delta(f) \leq \max\{k(f), l(f)\}$$

The extremal competitive ratio for game trees

Theorem

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

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

Theorem

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

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

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,

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

Theorem

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

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

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

Value dependent costs

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,

$$\mathit{ExtremalCompetitiveRatio}(f, r) = r \cdot \mathit{ECR}(f) - r + 1\,,$$

where

$$r = \max_{x \in V} \frac{c_{max}(x)}{c_{min}(x)}\,.$$

Value dependent costs

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,

$$\mathit{ExtremalCompetitiveRatio}(f, r) = r \cdot \mathit{ECR}(f) - r + 1\,,$$

where

$$r = \max_{x \in V} \frac{c_{max}(x)}{c_{min}(x)}\,.$$

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]
- $f : S_1 \times \cdots \times S_n \to S$, nonconstant:
  $ExtremalCompetitiveRatio(f) \leq PROOF(f)$ [Cicalese-Laber 2008]
- f = monotone Boolean function:
  $ExtremalCompetitiveRatio(f) = PROOF(f)$ [Cicalese-Laber 2008]
- f = game tree: $ExtremalCompetitiveRatio(f) \leq max\{k(f), l(f)\}$

Summary

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**,
- 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)?

The end

THANK YOU