

How to correctly prune tropical trees

Jean-Vincent Loddo
Luca Saiu

<http://www-lipn.univ-paris13.fr/~loddo>
<http://www-lipn.univ-paris13.fr/~saiu>

LIPN, Université Paris 13

Paris, July 5, 2010

Motivation

Many problems can be solved with a **non-deterministic search** on a space where some metric is **optimized**.

Motivation

Many problems can be solved with a **non-deterministic search** on a space where some metric is **optimized**.

- **Tropical games** are a generalization of min-max games

Motivation

Many problems can be solved with a **non-deterministic search** on a space where some metric is **optimized**.

- **Tropical games** are a generalization of min-max games
- Tropical games support **tropical α -pruning**
 - tropical games whose dual is also tropical can be computed with α - β (not proved formally, but intuitive)

Motivation

Many problems can be solved with a **non-deterministic search** on a space where some metric is **optimized**.

- **Tropical games** are a generalization of min-max games
- Tropical games support **tropical α -pruning**
 - tropical games whose dual is also tropical can be computed with α - β (not proved formally, but intuitive)
- ***Approximated parsing*** is a tropical game
 - α -pruning seems to be effective
 - ...we suspect many more problems to be tropical games

Combinatorial games (1): min-max

Let a game arena $S = (\mathbb{P}, \lambda, \text{succ})$ and a payoff function $p : \mathbb{P}_T \rightarrow \mathbb{Z} \cup \{+\infty, -\infty\}$ be given.

The *game value* of a position $\pi \in \mathbb{P}$ is traditionally defined as:

$$v_p(\pi) = \begin{cases} p(\pi), & \pi \in \mathbb{P}_T \\ \min_{i=1}^n v_p(\pi_i), & \text{succ}(\pi) = \langle \pi_1 \dots \pi_n \rangle, \lambda(\pi) = \mathcal{P} \\ \max_{i=1}^n v_p(\pi_i), & \text{succ}(\pi) = \langle \pi_1 \dots \pi_n \rangle, \lambda(\pi) = \mathcal{O} \end{cases}$$

Combinatorial games (1): min-max

Let a game arena $S = (\mathbb{P}, \lambda, succ)$ and a payoff function $p : \mathbb{P}_T \rightarrow \mathbb{Z} \cup \{+\infty, -\infty\}$ be given.

The *game value* of a position $\pi \in \mathbb{P}$ is traditionally defined as:

$$v_p(\pi) = \begin{cases} p(\pi), & \pi \in \mathbb{P}_T \\ \min_{i=1}^n v_p(\pi_i), & succ(\pi) = \langle \pi_1 \dots \pi_n \rangle, \lambda(\pi) = \mathcal{P} \\ \max_{i=1}^n v_p(\pi_i), & succ(\pi) = \langle \pi_1 \dots \pi_n \rangle, \lambda(\pi) = \mathcal{O} \end{cases}$$

What if we generalize? **let's take a generic $\mathcal{A} = (\mathbb{U}, \oplus, \odot)$** instead of $(\mathbb{Z} \cup \{+\infty, -\infty\}, \min, \max)$

Combinatorial games (2): \oplus - \odot

Let a game arena $S = (\mathbb{P}, \lambda, succ)$ and a payoff function $\rho : \mathbb{P}_T \rightarrow \mathbb{U}$ be given.

The *game value* of a position $\pi \in \mathbb{P}$ is defined as:

$$v_\rho(\pi) = \begin{cases} \rho(\pi), & \pi \in \mathbb{P}_T \\ \bigoplus_{i=1}^n v_\rho(\pi_i), & succ(\pi) = \langle \pi_1 \dots \pi_n \rangle, \lambda(\pi) = \mathcal{P} \\ \bigodot_{i=1}^n v_\rho(\pi_i), & succ(\pi) = \langle \pi_1 \dots \pi_n \rangle, \lambda(\pi) = \mathcal{O} \end{cases}$$

Small-step semantics I

$$[\text{Payoff}] \frac{\pi \in \mathbb{P}_T \quad \rho(\pi) = v}{\pi \rightarrow v}$$

$$[\mathcal{P}\text{-expand}] \frac{\text{succ}(\pi) = \vec{t} \quad \lambda(\pi) = \mathcal{P}}{\pi \rightarrow \sum \vec{t}} \quad \#\vec{t} \geq 1$$

$$[\mathcal{O}\text{-expand}] \frac{\text{succ}(\pi) = \vec{t} \quad \lambda(\pi) = \mathcal{O}}{\pi \rightarrow \prod \vec{t}} \quad \#\vec{t} \geq 1$$

$$[\mathcal{P}\text{-reduce}] \frac{}{\sum \vec{t} \langle v_1 \ v_2 \rangle \vec{z} \rightarrow \sum \vec{t} \langle v \rangle \vec{z}} \quad v_1 \oplus v_2 = v$$

Small-step semantics II

$$[\mathcal{O}\text{-reduce}] \frac{}{\prod \vec{t}\langle v_1 \ v_2 \rangle \vec{z} \rightarrow \prod \vec{t}\langle v \rangle \vec{z}} \quad v_1 \odot v_2 = v$$

$$[\text{Return}] \frac{}{\Lambda\langle v \rangle \rightarrow v}$$

$$[\text{Context}] \frac{t \rightarrow t'}{C[t] \rightarrow_c C[t']} \text{ for all contexts } C$$

The reflexive-transitive closure of \rightarrow_c defines rewrites.

Tropical algebras

Definition (Tropical algebra)

$\mathcal{A} = (\mathbb{U}, \oplus, \odot)$ is a *tropical algebra* if

- \oplus and \odot are associative
- \odot distributes over \oplus : for all $a, b, c \in \mathbb{U}$
 $a \odot (b \oplus c) = (a \odot b) \oplus (a \odot c)$ and
 $(a \oplus b) \odot c = (a \odot c) \oplus (b \odot c)$

Slightly simplified: see the paper.

Tropical algebras

Definition (Tropical algebra)

$\mathcal{A} = (\mathbb{U}, \oplus, \odot)$ is a *tropical algebra* if

- \oplus and \odot are associative
- \odot distributes over \oplus : for all $a, b, c \in \mathbb{U}$
 $a \odot (b \oplus c) = (a \odot b) \oplus (a \odot c)$ and
 $(a \oplus b) \odot c = (a \odot c) \oplus (b \odot c)$

Slightly simplified: see the paper.

The *min,+ algebra* $(\mathbb{Z} \cup \{+\infty\}, \min, +)$ is an important example of tropical algebra.

The *Rationality hypothesis*

For any α, β it's easy to see that:

$$\alpha \leq \beta + \alpha$$

The *Rationality hypothesis*

For any α, β it's easy to see that:

$$\alpha \leq \beta + \alpha$$

Or, using only the operations in $(\mathbb{Z} \cup \{+\infty\}, \min, +)$:

$$\alpha = \min\{\alpha, \beta + \alpha\}$$

The *Rationality hypothesis*

For any α, β it's easy to see that:

$$\alpha \leq \beta + \alpha$$

Or, using only the operations in $(\mathbb{Z} \cup \{+\infty\}, \min, +)$:

$$\alpha = \min\{\alpha, \beta + \alpha\}$$

Generalizing to $(\mathbb{U}, \oplus, \odot)$ we obtain:

Definition (Rationality)

$\mathcal{A} = (\mathbb{U}, \oplus, \odot)$ is *rational* iff for any $\alpha, \beta \in \mathbb{U}$ we have that
 $\alpha = \alpha \oplus (\beta \odot \alpha)$

Slightly simplified: see the paper.

The *Rationality hypothesis*

For any α, β it's easy to see that:

$$\alpha \leq \beta + \alpha$$

Or, using only the operations in $(\mathbb{Z} \cup \{+\infty\}, \min, +)$:

$$\alpha = \min\{\alpha, \beta + \alpha\}$$

Generalizing to $(\mathbb{U}, \oplus, \odot)$ we obtain:

Definition (Rationality)

$\mathcal{A} = (\mathbb{U}, \oplus, \odot)$ is *rational* iff for any $\alpha, \beta \in \mathbb{U}$ we have that
 $\alpha = \alpha \oplus (\beta \odot \alpha)$

Slightly simplified: see the paper.

“The player \mathcal{P} prefers α over α worsened by β ”

The *Rationality hypothesis*

For any α, β it's easy to see that:

$$\alpha \leq \beta + \alpha$$

Or, using only the operations in $(\mathbb{Z} \cup \{+\infty\}, \min, +)$:

$$\alpha = \min\{\alpha, \beta + \alpha\}$$

Generalizing to $(\mathbb{U}, \oplus, \odot)$ we obtain:

Definition (Rationality)

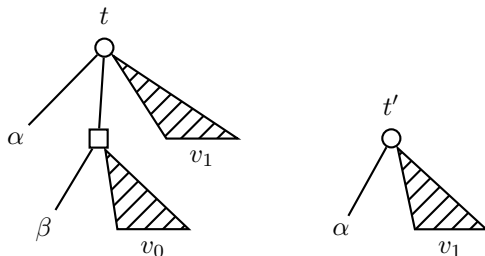
$\mathcal{A} = (\mathbb{U}, \oplus, \odot)$ is *rational* iff for any $\alpha, \beta \in \mathbb{U}$ we have that
 $\alpha = \alpha \oplus (\beta \odot \alpha)$

Slightly simplified: see the paper.

“The player \mathcal{P} prefers α over α worsened by β ”
 α -pruning depends on **tropicality** and **rationality**

\mathcal{P} -cut

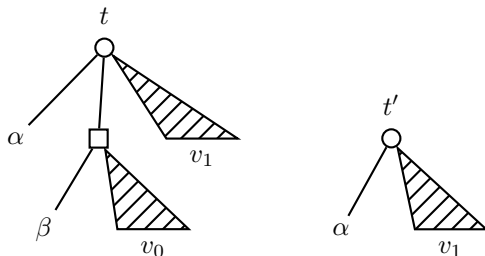
Intuition: think of \oplus as \min and \odot as $+$. For any $v_0, v_1, \vec{t}_0, \vec{t}_1, \alpha, \beta$ if $\alpha \oplus \beta = \alpha$ then $t \leq t'$ (" t' simulates t "):



"The player \mathcal{P} prefers α to β "

\mathcal{P} -cut

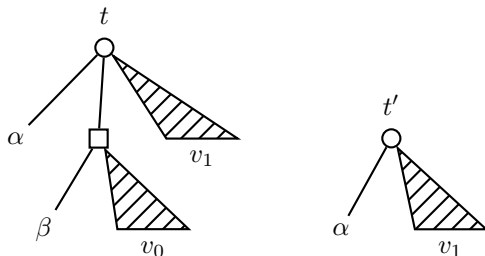
Intuition: think of \oplus as \min and \odot as $+$. For any $v_0, v_1, \vec{t}_0, \vec{t}_1, \alpha, \beta$ if $\alpha \oplus \beta = \alpha$ then $t \leq t'$ (" t' simulates t "):



"The player \mathcal{P} prefers α to β "
so computing v_0 is a waste of time

\mathcal{P} -cut

Intuition: think of \oplus as \min and \odot as $+$. For any $v_0, v_1, \vec{t}_0, \vec{t}_1, \alpha, \beta$
if $\alpha \oplus \beta = \alpha$ then $t \leq t'$ (" t' simulates t "):

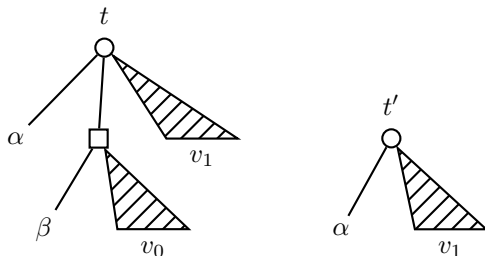


"The player \mathcal{P} prefers α to β "
so computing v_0 is a waste of time

if $\min\{\alpha, \beta\} = \alpha$ then $\min\{\alpha, (\beta + v_0), v_1\} = \min\{\alpha, v_1\}$

\mathcal{P} -cut

Intuition: think of \oplus as \min and \odot as $+$. For any $v_0, v_1, \vec{t}_0, \vec{t}_1, \alpha, \beta$
 if $\alpha \oplus \beta = \alpha$ then $t \leq t'$ (" t' simulates t "):

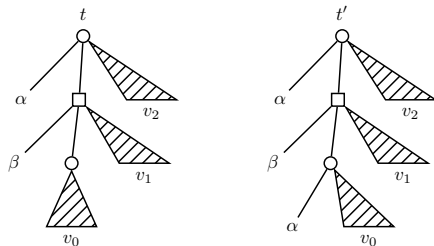


"The player \mathcal{P} prefers α to β "
 so computing v_0 is a waste of time

if $\min\{\alpha, \beta\} = \alpha$ then $\min\{\alpha, (\beta + v_0), v_1\} = \min\{\alpha, v_1\}$

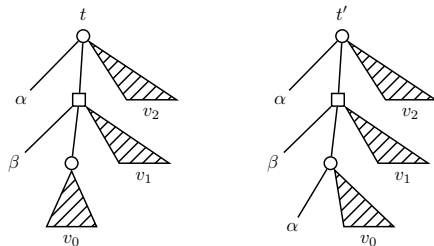
if $\alpha \oplus \beta = \alpha$ then $\alpha \oplus (\beta \odot v_0) \oplus v_1 = \alpha \oplus v_1$

\mathcal{P} -will: for any $\vec{t}_0, \vec{t}_1, \vec{t}_2, \alpha, \beta$ we have $t \leq t'$



“The player \mathcal{P} can bequeath α to her grandchild”
(which allows to prune more)

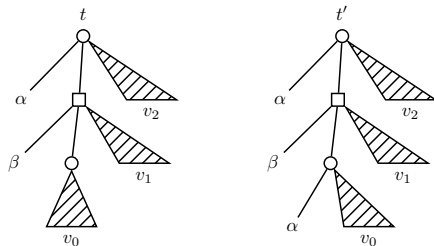
\mathcal{P} -will: for any $\vec{t}_0, \vec{t}_1, \vec{t}_2, \alpha, \beta$ we have $t \leq t'$



“The player \mathcal{P} can bequeath α to her grandchild”
 (which allows to prune more)

$$\min\{\alpha, (\beta + v_0 + v_1), v_2\} = \min\{\alpha, (\beta + \min\{\alpha, v_0\} + v_1), v_2\}$$

\mathcal{P} -will: for any $\vec{t}_0, \vec{t}_1, \vec{t}_2, \alpha, \beta$ we have $t \leq t'$

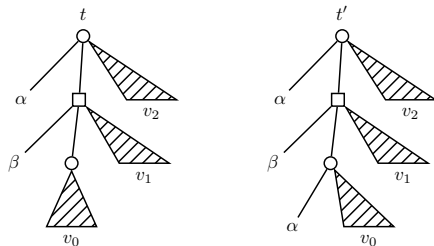


“The player \mathcal{P} can bequeath α to her grandchild”
 (which allows to prune more)

$$\min\{\alpha, (\beta + v_0 + v_1), v_2\} = \min\{\alpha, (\beta + \min\{\alpha, v_0\} + v_1), v_2\}$$

$$\alpha \oplus (\beta \odot v_0 \odot v_1) \oplus v_2 = \alpha \oplus (\beta \odot (\alpha \oplus v_0) \odot v_1) \oplus v_2$$

\mathcal{P} -will: for any $\vec{t}_0, \vec{t}_1, \vec{t}_2, \alpha, \beta$ we have $t \leq t'$



“The player \mathcal{P} can bequeath α to her grandchild”
 (which allows to prune more)

$$\min\{\alpha, (\beta + v_0 + v_1), v_2\} = \min\{\alpha, (\beta + \min\{\alpha, v_0\} + v_1), v_2\}$$

$$\alpha \oplus (\beta \odot v_0 \odot v_1) \oplus v_2 = \alpha \oplus (\beta \odot (\alpha \oplus v_0) \odot v_1) \oplus v_2$$

{This is due to rationality and to properties of tropical algebras}

Pruning rules (for bi-tropical games)

$$[\mathcal{P}\text{-will}] \frac{}{\sum \langle \alpha \mid \Pi \langle \beta \mid (\sum \vec{t}_1) \rangle \vec{t}_2 \rangle \vec{t}_3 \rightarrow \sum \langle \alpha \mid \Pi \langle \beta \mid (\sum \langle \alpha \rangle \vec{t}_1) \rangle \vec{t}_2 \rangle \vec{t}_3}$$

$$[\mathcal{O}\text{-will}] \frac{}{\Pi \langle \beta \mid [\sum \langle \alpha \mid (\Pi \vec{t}_1) \rangle \vec{t}_2] \rangle \vec{t}_3 \rightarrow \Pi \langle \beta \mid [\sum \langle \alpha \mid (\Pi \langle \beta \rangle \vec{t}_1) \rangle \vec{t}_2] \rangle \vec{t}_3}$$

$$[\mathcal{P}\text{-cut}] \frac{\alpha \oplus \beta = \alpha}{\sum \langle \alpha \mid (\Pi \langle \beta \rangle \vec{t}_1) \rangle \vec{t}_2 \rightarrow \sum \langle \alpha \rangle \vec{t}_2}$$

$$[\mathcal{O}\text{-cut}] \frac{\beta \odot \alpha = \beta}{\Pi \langle \beta \mid (\sum \langle \alpha \rangle \vec{t}_1) \rangle \vec{t}_2 \rightarrow \Pi \langle \beta \rangle \vec{t}_2}$$

α - β -pruning (1) [α best till now for \mathcal{P} /min; β best till now for \mathcal{O} /max]

```
1  function alpha_beta( $\pi : \mathbb{P}$ ;  $\alpha, \beta : \mathbb{Z}$ ): $\mathbb{Z}$ 
2    if  $\pi \in \mathbb{P}_T$  then
3      return  $\rho(\pi)$ 
4     $\pi_1 \dots \pi_n := \text{succ}(\pi)$  #  $n \geq 1$ 
5    if  $\lambda(\pi) = \mathcal{P}$  then
6       $v := \alpha$  # Best for  $\mathcal{P}$  till now
7      for  $i$  from 1 to  $n$ 
8        and while  $\beta <_{\mathbb{Z}} v$  do
9           $v := \min\{v, \text{alpha\_beta}(\pi_i, v, \beta)\}$ 
10   else #  $\lambda(\pi) = \mathcal{O}$ 
11      $v := \beta$ 
12     for  $i$  from 1 to  $n$ 
13       and while  $v <_{\mathbb{Z}} \alpha$  do
14          $v := \max\{v, \text{alpha\_beta}(\pi_i, \alpha, v)\}$ 
15   return  $v$ 
```

α - β -pruning (2) [α best till now for \mathcal{P} /min; β best till now for \mathcal{O} /max]

```
1  function alpha_beta( $\pi : \mathbb{P}$ ;  $\alpha, \beta : \mathbb{Z}$ ): $\mathbb{Z}$ 
2    if  $\pi \in \mathbb{P}_T$  then
3      return  $\rho(\pi)$ 
4     $\pi_1 \dots \pi_n := \text{succ}(\pi)$  #  $n \geq 1$ 
5    if  $\lambda(\pi) = \mathcal{P}$  then
6       $v := \alpha$  # Best for  $\mathcal{P}$  till now
7      for  $i$  from 1 to  $n$ 
8        and while  $\beta <_{\mathbb{Z}} v$  do # There's no  $\beta$ 
9           $v := \min\{v, \text{alpha\_beta}(\pi_i, v, \beta)\}$  # There's no  $\beta$ 
10   else #  $\lambda(\pi) = \mathcal{O}$ 
11      $v := \beta$  # There's no  $\beta$ 
12     for  $i$  from 1 to  $n$ 
13       and while  $v <_{\mathbb{Z}} \alpha$  do
14          $v := \max\{v, \text{alpha\_beta}(\pi_i, \alpha, v)\}$ 
15   return  $v$ 
```

Tropical α -pruning [α current best for \mathcal{P}/\oplus]

```
1 function tropical( $\pi : \mathbb{P}$ ;  $\alpha : \mathbb{U}$ ):  $\mathbb{U}$ 
2   if  $\pi \in \mathbb{P}_T$  then
3     return  $\rho(\pi)$ 
4    $\pi_1 \dots \pi_n := \text{succ}(\pi)$  #  $n \geq 1$ 
5   if  $\lambda(\pi) = \mathcal{P}$  then
6      $v := \alpha$  # Best for  $\mathcal{P}$  till now
7     for  $i$  from 1 to  $n$  do
8       # do not prune at  $\mathcal{P}$ 's level
9        $v := v \oplus \text{tropical}(\pi_i, v)$ 
10  else #  $\lambda(\pi) = \mathcal{O}$ 
11     $v := \text{tropical}(\pi_1, \alpha)$  # No  $1_{\mathbb{U}}$ 
12    for  $i$  from 2 to  $n$ 
13      and while  $\alpha \oplus v \neq \alpha$  do
14         $v := v \odot \text{tropical}(\pi_i, \alpha)$ 
15  return  $v$ 
```

Divide and Conquer

Traditional D&C algorithms consist in:

- Divide a problem into sub-problems
- Combine solutions

For example, *merge-sort*.

Divide and Conquer

Traditional D&C algorithms consist in:

- Divide a problem into sub-problems [deterministically]
- Combine solutions

For example, *merge-sort*.

Let's generalize this to *nondeterministic* divisions

Choose-How-To-Divide and Conquer

- non-deterministic choices in a solution space
- some way of “combining” sub-solutions.

Choose-How-To-Divide and Conquer

- non-deterministic choices in a solution space
choose a division (\mathcal{P}, \oplus) : for example *min*
- some way of “combining” sub-solutions.

Choose-How-To-Divide and Conquer

- non-deterministic choices in a solution space
choose a division (\mathcal{P}, \oplus) : for example *min*
- some way of “combining” sub-solutions.
“accumulating values” (\mathcal{O}, \odot) : for example $+$

Choose-How-To-Divide and Conquer

- non-deterministic choices in a solution space
choose a division (\mathcal{P}, \oplus) : for example *min*
- some way of “combining” sub-solutions.
“accumulating values” (\mathcal{O}, \odot) : for example $+$

If $\mathcal{A} = (\mathbb{U}, \oplus, \odot)$ is tropical we can use α -pruning

Choose-How-To-Divide and Conquer

- non-deterministic choices in a solution space
choose a division (\mathcal{P}, \oplus) : for example *min*
- some way of “combining” sub-solutions.
“accumulating values” (\mathcal{O}, \odot) : for example $+$

If $\mathcal{A} = (\mathbb{U}, \oplus, \odot)$ is tropical we can use α -pruning

\oplus represents choice according to a quality criterion
 \odot “accumulates”

Parsing as a tropical game

Let a context-free grammar be given.

Find the “least-wrong” parse, where each parse may contain unmatched terminals

Parsing as a tropical game

Let a context-free grammar be given.

Find the “least-wrong” parse, where each parse may contain unmatched terminals

- Minimum number of errors
- Minimum number of unmatched characters

Parsing as a tropical game

Let a context-free grammar be given.

Find the “least-wrong” parse, where each parse may contain unmatched terminals

- Minimum number of errors
- Minimum number of unmatched characters

Hypotheses to make exposition simpler (liftable):

- No ϵ -productions
- No two adjacent non-terminals
- No non-terminal alone on a right side

Parsing as a tropical game: nondeterministic parsing

We have a substring to parse with a nonterminal, for example

"1 + 2 + 3" with E .

Parsing as a tropical game: nondeterministic parsing

We have a substring to parse with a nonterminal, for example

"1 + 2 + 3" with E .

- \mathcal{P} : Choose a production and a division (pivot characters)

Parsing as a tropical game: nondeterministic parsing

We have a substring to parse with a nonterminal, for example

"1 + 2 + 3" with E .

- \mathcal{P} : Choose a production and a division (pivot characters)
- \mathcal{O} : Parse nonterminals "in between" pivots

Parsing as a tropical game: nondeterministic parsing

We have a substring to parse with a nonterminal, for example

"1 + 2 + 3" with E .

- \mathcal{P} : Choose a production and a division (pivot characters)
- \mathcal{O} : Parse nonterminals "in between" pivots

$$\pi_{\mathcal{P}} = ("1 + 2 + 3", E)$$

Parsing as a tropical game: nondeterministic parsing

We have a substring to parse with a nonterminal, for example

"1 + 2 + 3" with E .

- \mathcal{P} : Choose a production and a division (pivot characters)
- \mathcal{O} : Parse nonterminals "in between" pivots

$$\pi_{\mathcal{P}} = ("1 + 2 + 3", E)$$

$$\pi_{\mathcal{O}} = ("1", E), ("2 + 3", E)$$

Parsing as a tropical game: nondeterministic parsing

We have a substring to parse with a nonterminal, for example

"1 + 2 + 3" with E .

- \mathcal{P} : Choose a production and a division (pivot characters)
- \mathcal{O} : Parse nonterminals "in between" pivots

$$\pi_{\mathcal{P}} = ("1 + 2 + 3", E)$$

$$\pi_{\mathcal{O}} = ("1", E), ("2 + 3", E)$$

$$\mathcal{A} = (\mathbb{Z} \cup \{+\infty\}, \min, +)$$

Parsing as a tropical game: example

Parsing "1 + 2 + 3":

$E ::= n$

$E ::= v$

$E ::= (E)$

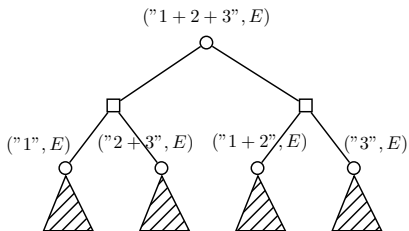
$E ::= \text{let } v = E \text{ in } E$

$E ::= \text{if } E \text{ then } E \text{ else } E$

$E ::= E = E$

$E ::= E + E$

$E ::= E * E$



Parsing as a tropical game: example

Parsing "1 + 2 + 3":

$E ::= n$

$E ::= v$

$E ::= (E)$

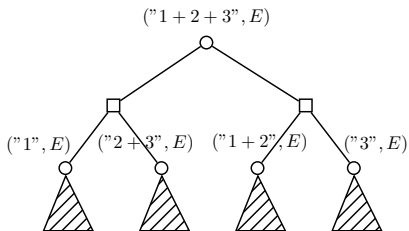
$E ::= \text{let } v = E \text{ in } E$

$E ::= \text{if } E \text{ then } E \text{ else } E$

$E ::= E = E$

$E ::= E + E$

$E ::= E * E$



What if we had let instead of 3? An error...

Memoization

Memoization: avoiding to re-examined repeated positions

Preliminary practical test

Is α -pruning useful?

Is memoization useful?

[little demo]

Other α -pruning examples

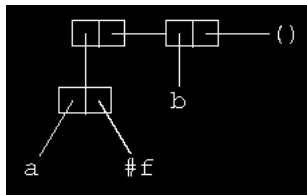
- Searching the most general solution in logic programming
[Loddo-DiCosmo, LPAR 2000] [Loddo PhD 2002]

Other α -pruning examples

- Searching the most general solution in logic programming [Loddo-DiCosmo, LPAR 2000] [Loddo PhD 2002]
 - optimization criterion: “less-instantiated terms”. *inf*, *sup* bi-tropical

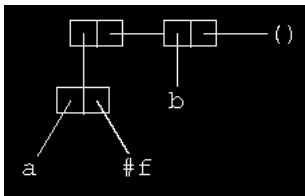
Other α -pruning examples

- Searching the most general solution in logic programming [Loddo-DiCosmo, LPAR 2000] [Loddo PhD 2002]
 - optimization criterion: “less-instantiated terms”. *inf*, *sup* bi-tropical
- Draw a planar graph...



Other α -pruning examples

- Searching the most general solution in logic programming [Loddo-DiCosmo, LPAR 2000] [Loddo PhD 2002]
 - optimization criterion: “less-instantiated terms”. *inf*, *sup* bi-tropical
- Draw a planar graph...

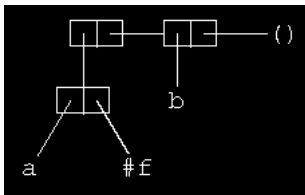


...minimizing:

- the occupied area

Other α -pruning examples

- Searching the most general solution in logic programming [Loddo-DiCosmo, LPAR 2000] [Loddo PhD 2002]
 - optimization criterion: “less-instantiated terms”. *inf*, *sup* bi-tropical
- Draw a planar graph...



...minimizing:

- the occupied area
- the number of crossings

Conclusions

We...

- ...defined and proved correct *tropical α -pruning*
- ...formalized *min-max* games as particular (bi-)tropical games
- ...modeled *approximated parsing* as a tropical game
 - α -pruning seems to be effective

We would like...

- ...to find more applications
- ...more realistic implementations

Conclusions

We...

- ...defined and proved correct *tropical α -pruning*
- ...formalized *min-max* games as particular (bi-)tropical games
- ...modeled *approximated parsing* as a tropical game
 - α -pruning seems to be effective

We would like...

- ...to find more applications
- ...more realistic implementations

Thanks!