

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

E-matching for Fun and Profit

Michał Moskal, Jakub Łopuszański, Joseph R. Kiniry

University of Wrocław, Poland
University College, Dublin, Ireland

SMT Workshop, Berlin
July 1st, A.D. 0x7D7

Instantiation

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

Let's take the formula:

$$P(f(42)) \wedge \forall x. (P(f(x)) \Rightarrow x < 0)$$

Instantiation

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

Let's take the formula:

$$P(f(42)) \wedge \forall x. (P(f(x)) \Rightarrow x < 0)$$

Once the solver figures out the implication:

$$(\forall x. (P(f(x)) \Rightarrow x < 0)) \Rightarrow (P(f(42)) \Rightarrow 42 < 0)$$

Instantiation

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

Let's take the formula:

$$P(f(42)) \wedge \forall x. (P(f(x)) \Rightarrow x < 0)$$

Once the solver figures out the implication:

$$(\forall x. (P(f(x)) \Rightarrow x < 0)) \Rightarrow (P(f(42)) \Rightarrow 42 < 0)$$

it can deduce the query to be unsatisfiable using only ground reasoning.

Instantiation

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

Let's take the formula:

$$P(f(42)) \wedge \forall x. (P(f(x)) \Rightarrow x < 0)$$

Once the solver figures out the implication:

$$(\forall x. (P(f(x)) \Rightarrow x < 0)) \Rightarrow (P(f(42)) \Rightarrow 42 < 0)$$

it can deduce the query to be unsatisfiable using only ground reasoning.

How do we figure the correct instantiation out?

Triggers (an idea borrowed from Simplify)

- triggers are subterms present in the quantified formula, which can be automatically generated, or come with the input
- we only consider instances, for which the trigger after substitution is present somewhere in the model returned by the SAT solver

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

Triggers (an idea borrowed from Simplify)

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- triggers are subterms present in the quantified formula, which can be automatically generated, or come with the input
- we only consider instances, for which the trigger after substitution is present somewhere in the model returned by the SAT solver
- for example if we have a formula

$$\psi \equiv \forall x, y. F(x, y) \Rightarrow g(x) = h(y)$$

for which $F(x, y)$ is a trigger and the current model returned by the SAT solver is:

$$\psi \wedge F(1, c) \wedge g(7) = h(c)$$

we only consider instance where $x \rightarrow 1, y \rightarrow c$, because $F(1, c)$ is present in the monome.

E-matching

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- this picture with the triggers is slightly more involved, because we are interested in the trigger being present in the monome *up to equivalence*
- for example trigger $F(x, c)$ does not syntactically match in $F(1, d) \wedge c = d$, but we would like it to

E-matching: definition

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem
E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees
The matcher

Flat matcher

Implementation

Wrapping up

Input

- a finite set \mathcal{A} of *active* ground terms,
- a relation $\cong_g \subseteq \mathcal{A} \times \mathcal{A}$,
- a finite set of non-variable, non-constant *triggers* p_1, \dots, p_n .

Definitions

- let $\cong \subseteq \mathcal{T} \times \mathcal{T}$ be the smallest congruence relation containing \cong_g ,
- let $\text{root}(t)$ denote a canonical representative of equivalence class containing t .

The solution to the E-matching problem is the set:

$$\mathcal{T} = \left\{ \sigma \mid \begin{array}{l} \exists t_1, \dots, t_n \in \mathcal{A}. \sigma(p_1) \cong t_1 \wedge \dots \wedge \sigma(p_n) \cong t_n, \\ \forall x \in \mathcal{V}. \sigma(x) = \text{root}(\sigma(x)) \end{array} \right\}$$

E-matching: complexity

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- the problem of checking if $T \neq \emptyset$ for a fixed \mathcal{A} and \cong_g is NP-hard
- there can be exponential number of instances of a trigger

E-matching: complexity

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

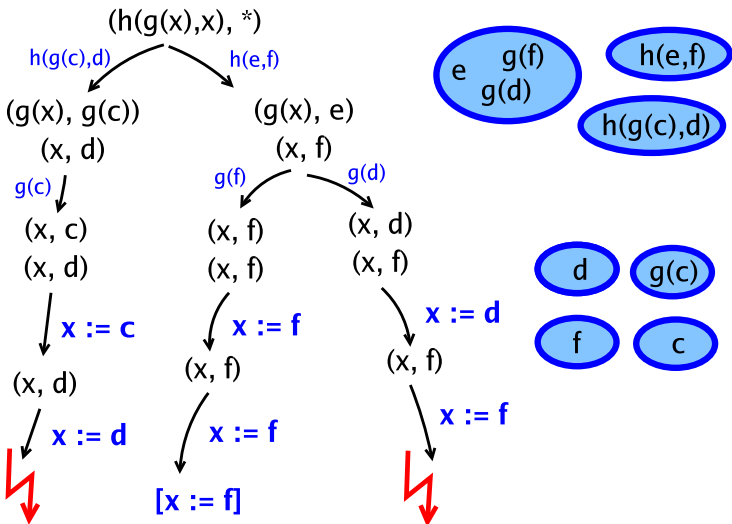
Flat matcher

Implementation

Wrapping up

- the problem of checking if $T \neq \emptyset$ for a fixed \mathcal{A} and \cong_g is NP-hard
- there can be exponential number of instances of a trigger
- the practical problem is, however, that there are often millions of matching problems to solve during solving of a single SMT query

Simplify's matcher example



E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

Simplify's algorithm

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem
E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees
The matcher

Flat matcher

Implementation

Wrapping up

```
fun simplify_match([ $p_1, \dots, p_n$ ])  
   $R := \emptyset$   
  proc match( $\sigma, j$ )  
    if  $j = \text{nil}$  then  $R := R \cup \{\sigma\}$   
    else case  $hd(j)$  of  
      ( $c, t$ )  $\Rightarrow$  /* 1 */  
        if  $c \cong t$  then match( $\sigma, tl(j)$ )  
        else skip  
      ( $x, t$ )  $\Rightarrow$  /* 2 */  
        if  $\sigma(x) = x$  then match( $\sigma[x := \text{root}(t)], tl(j)$ )  
        else if  $\sigma(x) = \text{root}(t)$  then match( $\sigma, tl(j)$ )  
        else skip  
      ( $f(p_1, \dots, p_n), t$ )  $\Rightarrow$  /* 3 */  
        foreach  $f(t_1, \dots, t_n)$  in  $\mathcal{A}$  do  
          if  $t = * \vee \text{root}(f(t_1, \dots, t_n)) = t$  then  
            match( $\sigma, (p_1, \text{root}(t_1)) :: \dots :: (p_n, \text{root}(t_n)) :: tl(j)$ )  
    match([], ( $p_1, *$ ) ::  $\dots$  :: ( $p_n, *$ ) :: nil) /* 4 */  
  return  $R$ 
```

S-tree sum

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

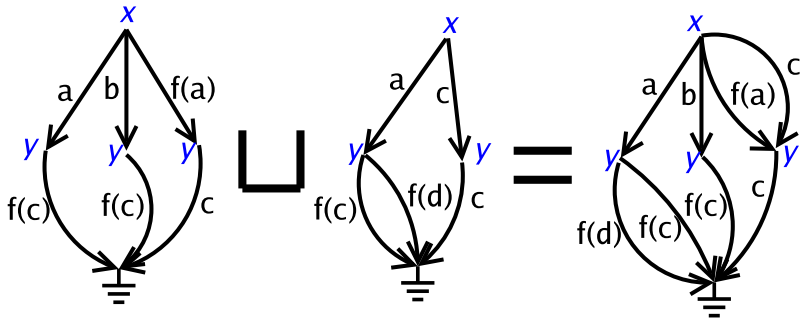
S-trees

The matcher

Flat matcher

Implementation

Wrapping up



S-tree merge

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

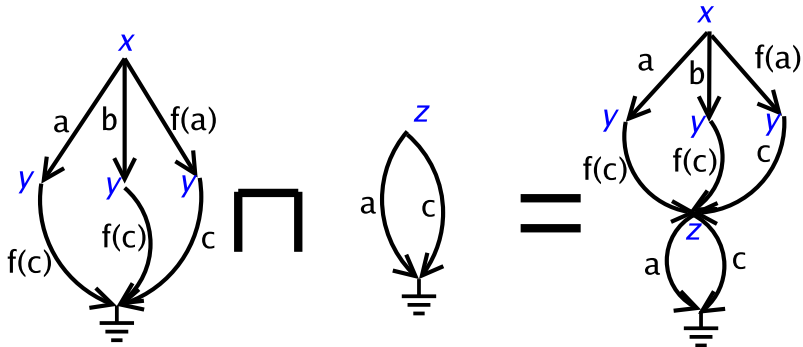
S-trees

The matcher

Flat matcher

Implementation

Wrapping up



Subtrigger matcher example, part I

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

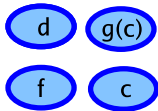
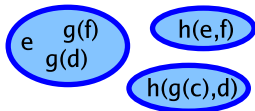
The matcher

Flat matcher

Implementation

Wrapping up

$h(g(x),x) \rightarrow$



Subtrigger matcher example, part II

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

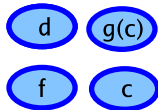
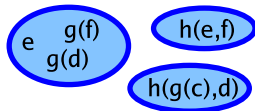
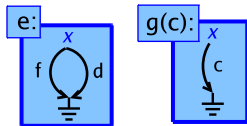
Flat matcher

Implementation

Wrapping up

$h(g(x),x) \rightarrow$

$g(x) \rightarrow$



Subtrigger matcher example, part III

E-matching for Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem
E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

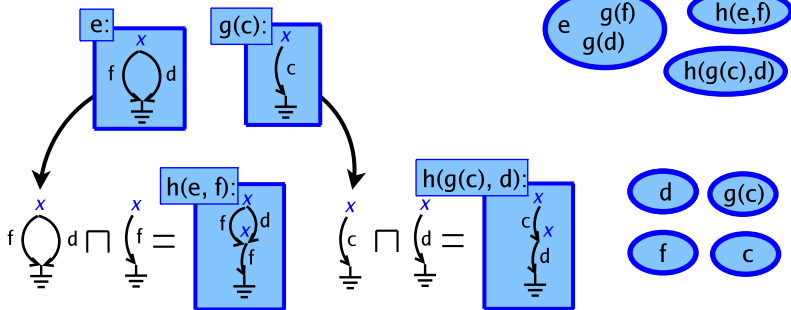
Flat matcher

Implementation

Wrapping up

$h(g(x),x) \rightarrow$

$g(x) \rightarrow$



Subtrigger matcher example, part IV

E-matching for Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

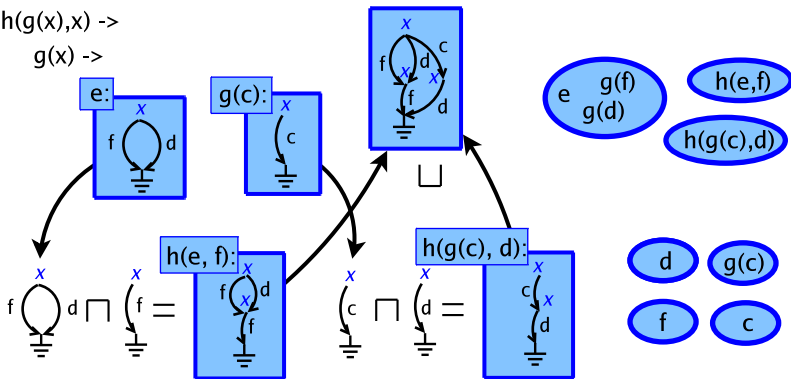
Flat matcher

Implementation

Wrapping up

$h(g(x), x) \rightarrow$

$g(x) \rightarrow$



Subtrigger matcher

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem
E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees
The matcher

Flat matcher

Implementation

Wrapping up

```
fun fetch( $S, t, p$ )  
  if  $S = \top$  then return  $\{[p := \text{root}(t)]\}$   
  else if  $S = \times \wedge t \cong p$  then return  $\{\}\}$   
  else if  $S = \times$  then return  $\emptyset$   
  else return  $S(\text{root}(t))$   
fun match( $p$ )  
  case  $p$  of  
     $x \Rightarrow$  return  $\top$   
     $c \Rightarrow$  return  $\times$   
     $f(p_1, \dots, p_n) \Rightarrow$   
      foreach  $i$  in  $1 \dots n$  do  $S_i = \text{match}(p_i)$            /* 1 */  
      if  $\exists i. S_i = \perp$  then return  $\perp$                        /* 2 */  
      if  $\forall i. S_i = \times$  then return  $\times$                        /* 3 */  
       $S := \{t \mapsto \emptyset \mid t \in \mathcal{A}\}$   
      foreach  $f(t_1, \dots, t_n)$  in  $\mathcal{A}$  do                   /* 4 */  
         $t := \text{root}(f(t_1, \dots, t_n))$   
         $S := S[t \mapsto S(t) \sqcup (\text{fetch}(S_1, t_1, p_1) \sqcap \dots \sqcap \text{fetch}(S_n, t_n, p_n))]$   
      if  $\forall t. S(t) = \perp$  then return  $\perp$   
      else return  $S$ 
```

Subtrigger matcher, cont.

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

```
fun topmatch(p)           /* 5 */  
  S := match(p)  
  return  $\bigsqcup_{t \in \mathcal{A}} S(t)$   
fun subtrigger_match([p1, ..., pn])  
  return topmatch(p1)  $\sqcap$  ...  $\sqcap$  topmatch(pn)
```

Even simpler triggers

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- even this algorithm was taking up a lot of time, mainly because the loop over all terms with given head is performed for each trigger

Even simpler triggers

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- even this algorithm was taking up a lot of time, mainly because the loop over all terms with given head is performed for each trigger
- but it seems that a lot (most?) of the triggers are even simpler – they have variables only at depth one: $f(X, Y)$, $f(c, X)$, $f(X, c, Y, g(g(d)), Z)$
- this means one can put all such *flat* triggers with head f in an indexing tree and match them all at once during one loop over terms with head f

Flat trigger index

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

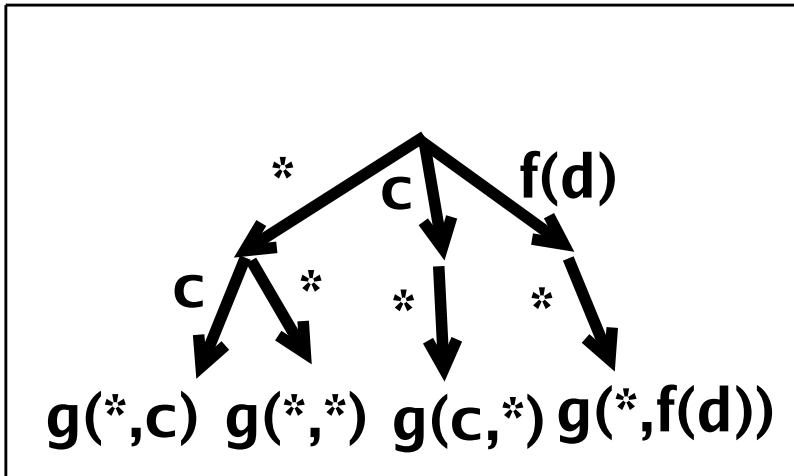
S-trees

The matcher

Flat matcher

Implementation

Wrapping up



Let's match something!

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

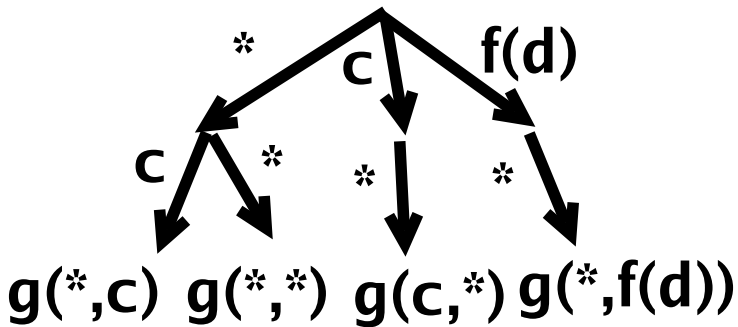
The matcher

Flat matcher

Implementation

Wrapping up

$g(e,c)$, where $e=c$



At the root

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

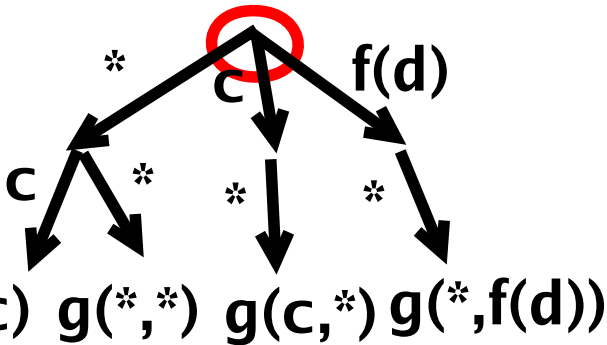
The matcher

Flat matcher

Implementation

Wrapping up

$g(e,c)$, where $e=c$



Child #1

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

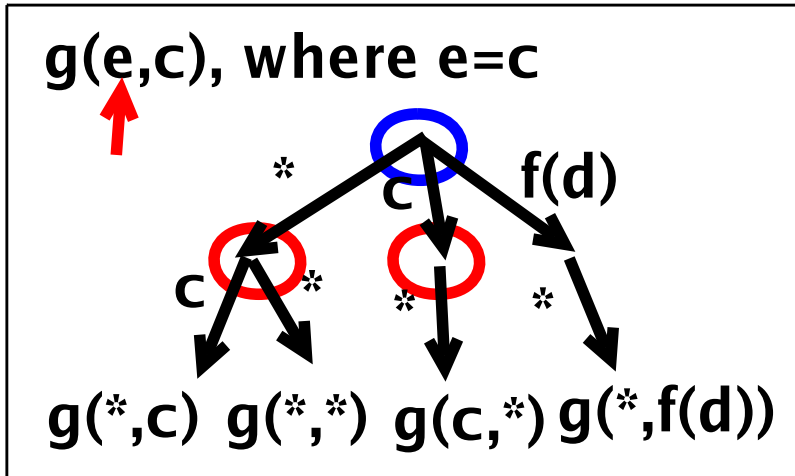
S-trees

The matcher

Flat matcher

Implementation

Wrapping up



Child #2

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

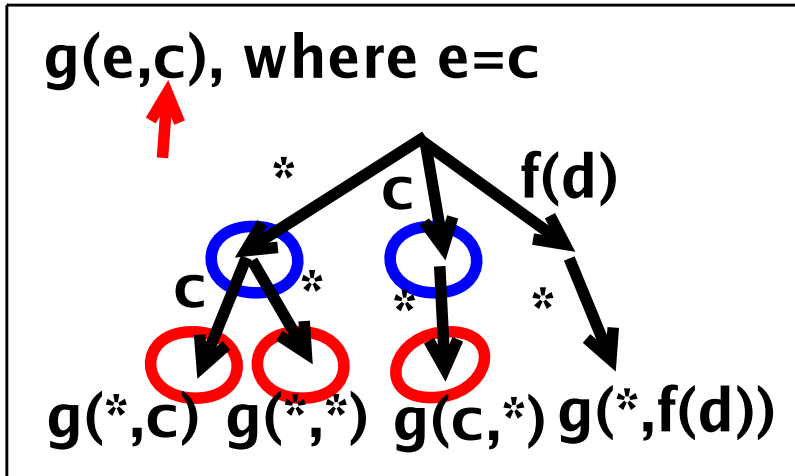
S-trees

The matcher

Flat matcher

Implementation

Wrapping up



Memoization

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- maximal sharing in terms and s-trees
- \sqcup, \sqcap memoize results
- s-tree subtraction to remove previously returned results
- mapping of all the variables to $*$, to maximize sharing in subtrigger matcher
- mixed effects with *mod-time*

Explanation

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- consider trigger: $f(g_1(x_1), \dots, g_n(x_n))$
- $g_1(x_1)$ through $g_{n-1}(x_{n-1})$ return two matches each
- $g_n(x_n)$ does not match anything

Explanation

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- consider trigger: $f(g_1(x_1), \dots, g_n(x_n))$
- $g_1(x_1)$ through $g_{n-1}(x_{n-1})$ return two matches each
- $g_n(x_n)$ does not match anything
- Simplify's matcher: $O(2^n)$ steps, subtrigger matcher: $O(n)$ steps

Explanation

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

- consider trigger: $f(g_1(x_1), \dots, g_n(x_n))$
- $g_1(x_1)$ through $g_{n-1}(x_{n-1})$ return two matches each
- $g_n(x_n)$ does not match anything
- Simplify's matcher: $O(2^n)$ steps, subtrigger matcher: $O(n)$ steps
- even if $g_n(x_n)$ matches something, subtrigger will still do $O(n)$ steps to match, and only $O(2^n)$ much cheaper steps to walk s-tree

Thank you!

E-matching for
Fun and Profit

Michał Moskal,
Jakub
Łopuszański,
Joseph R. Kiniry

Problem

E-matching

Simplify's
matcher

Subtrigger
matcher

S-trees

The matcher

Flat matcher

Implementation

Wrapping up

Questions?