Separation Logic Modulo Theories

Juan A. Navarro Pérez

joint work with Andrey Rybalchenko

Analyse This

```
node* insert_at(node* p, int k, int v) {
   node* q = p;
   for (int i = 0; i < k - 1; i++)
      q = q->next;
   node* r = new node;
   r->next = r->next;
   r->data = v;
   q->next = r;
   return p;
}
```

Analyse This

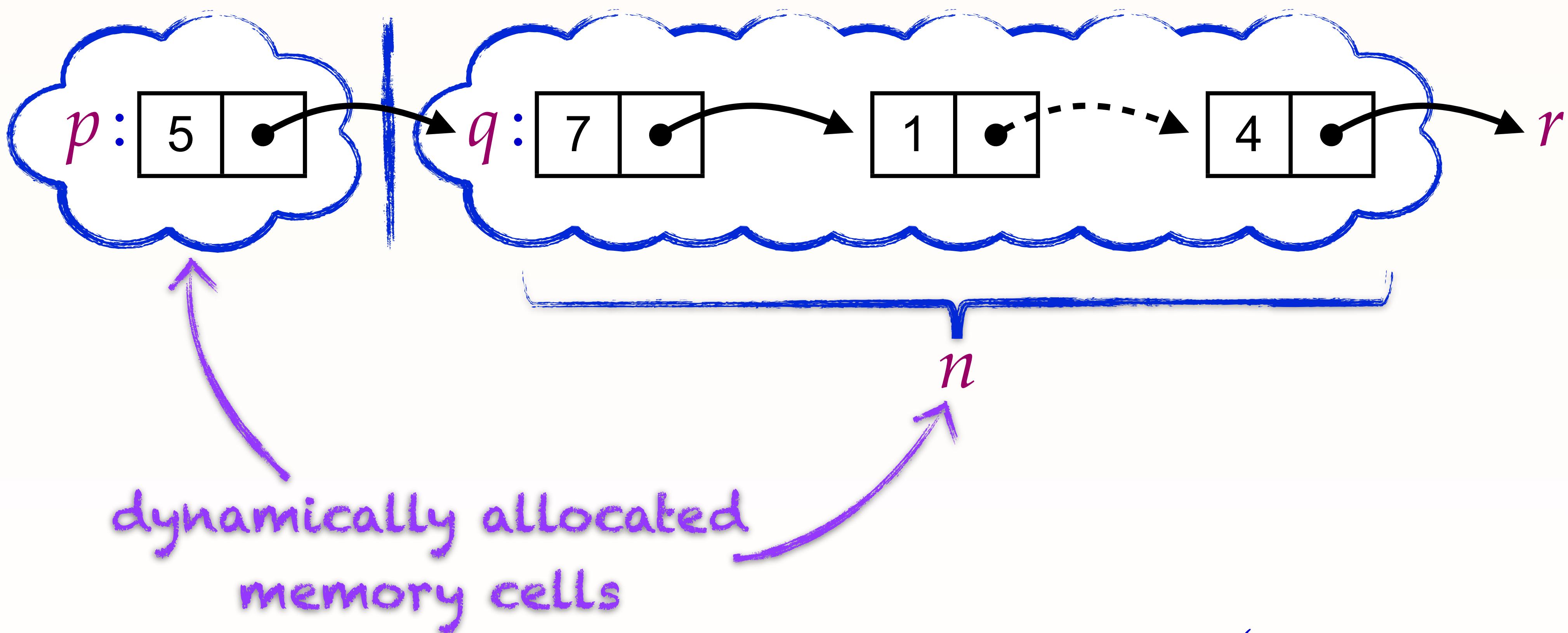
assume: $\mathsf{lseg}(p, \mathsf{nil}, n) \wedge 0 < k < n$

```
node* insert_at(node* p, int k, int v) {
    node* q = p;
    for (int i = 0; i < k - 1; i++)
        q = q->next;
    node* r = new node;
    r->next = r->next;
    r->data = v;
    q->next = r;
    return p;
}
```

to prove: $\mathsf{lseg}(p, \mathsf{nil}, n + 1)$

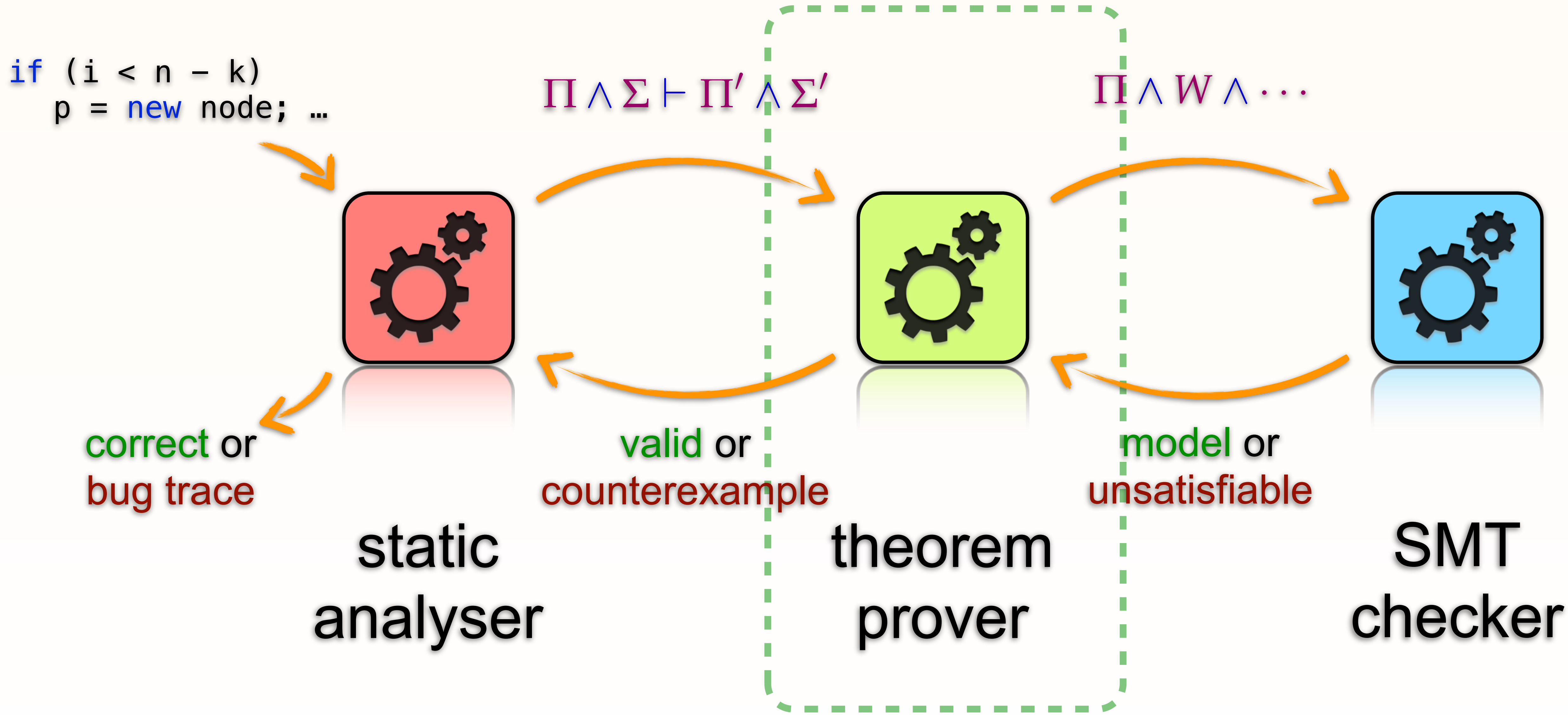
Separation Logic

$\mathsf{next}(p, q) * \mathsf{lseg}(q, r, n)$



$n \geq 0 \wedge (q = r \leftrightarrow n = 0)$

This work in context



Program Analysis

assume: $\mathsf{lseg}(p, \mathsf{nil}, n) \wedge 0 < k < n$

```
node* insert_at(node* p, int k, int v) {
   node* q = p;
   for (int i = 0; i < k - 1; i++)
      q = q->next;
   node* r = new node;
   r->next = q->next;
   r->data = v;
   q->next = r;
   return p;
}
```

to prove: $\mathsf{lseg}(p, \mathsf{nil}, n+1)$

Program Analysis

assume: $\mathsf{lseg}(p, \mathsf{nil}, n) \land 0 < k < n$

```
node* insert_at(node* p, int k, int v) {
    node* q = p;
    for (int i = 0; i < k - 1; i++)
        q = q->next;
    ...
    r->next = q->next;
    r->data = v;
    q->next = r;
    return p;
}
```

automatically and efficiently check:

$$i = i' + 1 \land \mathsf{lseg}(p, q', i') * \mathsf{next}(q', q) * \mathsf{lseg}(q, \mathsf{nil}, n - i' - 1)$$
$$\vdash \mathsf{lseg}(p, q, i) * \mathsf{lseg}(q, \mathsf{nil}, n - i)$$

to prove: $\mathsf{lseg}(p, \mathsf{nil}, n+1)$

Wellformedness

$$i = i' + 1 \land \mathsf{lseg}(p, q', i') * \mathsf{next}(q', q) * \mathsf{lseg}(q, \mathsf{nil}, n - i' - 1)$$
$$\vdash \mathsf{lseg}(p, q, i) * \mathsf{lseg}(q, \mathsf{nil}, n - i)$$

Prover

hypothesis

side conditions

$$i = i' + 1 \land i' \geq 0 \land (p = q' \leftrightarrow i' = 0) \land$$
$$n - i' - 1 \geq 0 \land (q = \mathsf{nil} \leftrightarrow n - i' - 1 = 0) \land$$
$$q' \neq \mathsf{nil} \land (p = q \rightarrow p = q' \lor q = \mathsf{nil}) \land (q' = q \rightarrow q = \mathsf{nil})$$

separation conditions

Wellformedness

$$i = i' + 1 \land \mathsf{lseg}(p, q', i') * \mathsf{next}(q', q) * \mathsf{lseg}(q, \mathsf{nil}, n - i' - 1)$$
$$\vdash \mathsf{lseg}(p, q, i) * \mathsf{lseg}(q, \mathsf{nil}, n - i)$$

Prover

hypothesis

side conditions

$$i = i' + 1 \land i' \geq 0 \land (p = q' \leftrightarrow i' = 0) \land$$

equisatisfiable with the premise of the entailment, pure formula

$$q' \neq \mathsf{nil} \land (p = q \rightarrow p = q' \lor q = \mathsf{nil}) \land (q' = q \rightarrow q = \mathsf{nil})$$

separation conditions

SMT

Matching

$$i = i' + 1 \wedge \mathsf{lseg}(p, q', i') * \mathsf{next}(q', q) * \mathsf{lseg}(q, \mathsf{nil}, n - i' - 1)$$
$$\vdash \mathsf{lseg}(p, q, i) * \mathsf{lseg}(q, \mathsf{nil}, n - i)$$

Prover

$$\mathsf{lseg}(p, q', 1) * \mathsf{next}(q', \mathsf{nil}) * \mathsf{lseg}(\mathsf{nil}, \mathsf{nil}, 0)$$
$$\vdash \mathsf{lseg}(p, \mathsf{nil}, 2) * \mathsf{lseg}(\mathsf{nil}, \mathsf{nil}, 0)$$

$p' = 42$	$p = 42$
$q' = 41$	$q = 0$
$i' = 1$	$i = 2$
$\mathsf{nil} = 0$	$n = 2$

match: $q = \mathsf{nil}$

$\cdots \wedge q \neq \mathsf{nil}$

SMT

SMT

Rinse and repeat

$$i = i' + 1 \land \mathsf{lseg}(p, q', i') * \mathsf{next}(q', q) * \mathsf{lseg}(q, \mathsf{nil}, n - i' - 1)$$
$$\vdash \mathsf{lseg}(p, q, i) * \mathsf{lseg}(q, \mathsf{nil}, n - i)$$

Prover

$$\mathsf{lseg}(p, p, 0) * \mathsf{next}(p, q) * \mathsf{lseg}(q, \mathsf{nil}, 1)$$
$$\vdash \mathsf{lseg}(p, q, 1) * \mathsf{lseg}(q, \mathsf{nil}, 1)$$

$$p' = 42 \quad p = 42$$
$$q' = 42 \quad q = 41$$
$$i' = 0 \quad i = 1$$
$$\mathsf{nil} = 0 \quad n = 2$$

SMT

match: $p = q'$

$$\cdots \land p \neq q'$$

SMT

... and repeat

$$i = i' + 1 \wedge \mathsf{lseg}(p, q', i') * \mathsf{next}(q', q) * \mathsf{lseg}(q, \mathsf{nil}, n - i' - 1) \vdash \mathsf{lseg}(p, q, i) * \mathsf{lseg}(q, \mathsf{nil}, n - i)$$

Prover

$$\mathsf{lseg}(p, q', 1) * \mathsf{next}(q', q) * \mathsf{lseg}(q, \mathsf{nil}, 1) \vdash \mathsf{lseg}(p, q, 2) * \mathsf{lseg}(q, \mathsf{nil}, 1)$$

$p' = 42$	$p = 42$
$q' = 43$	$q = 41$
$i' = 1$	$i = 2$
$\mathsf{nil} = 0$	$n = 3$

SMT

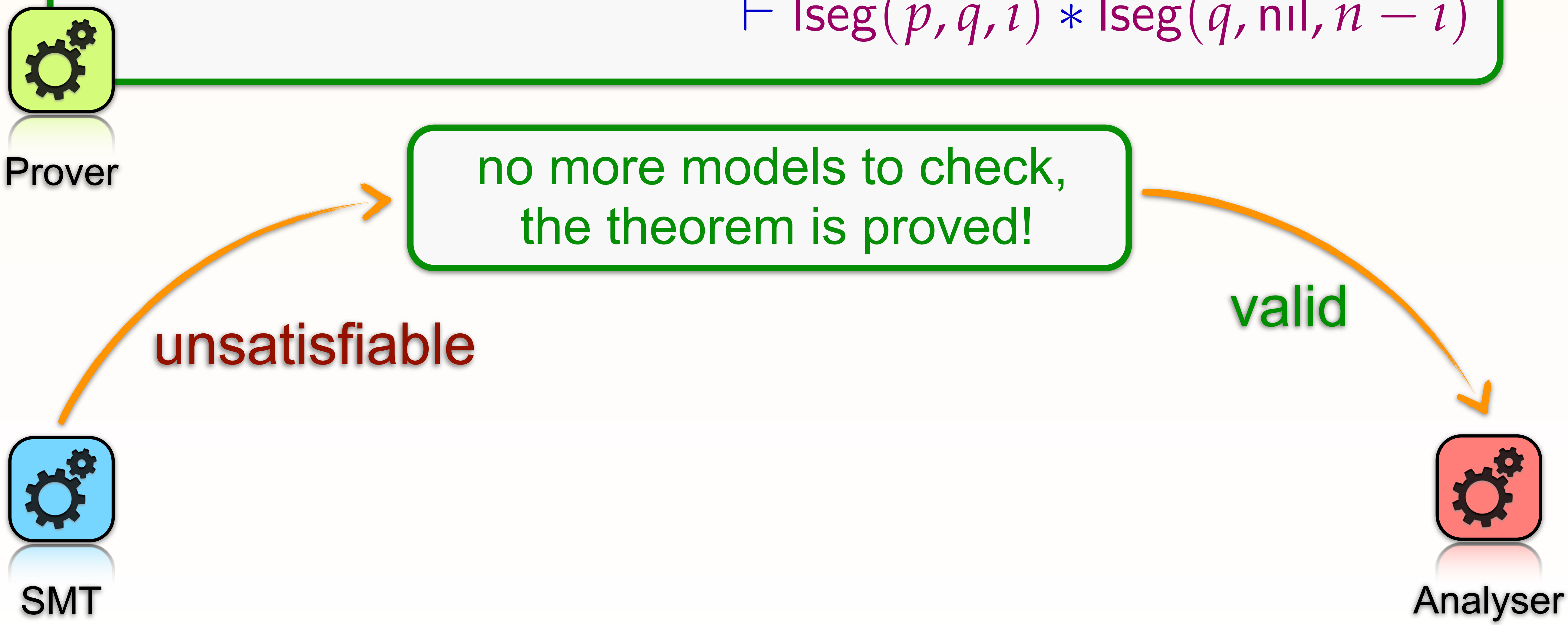
match: $q \neq \mathsf{nil}$

$\cdots \wedge q = \mathsf{nil}$

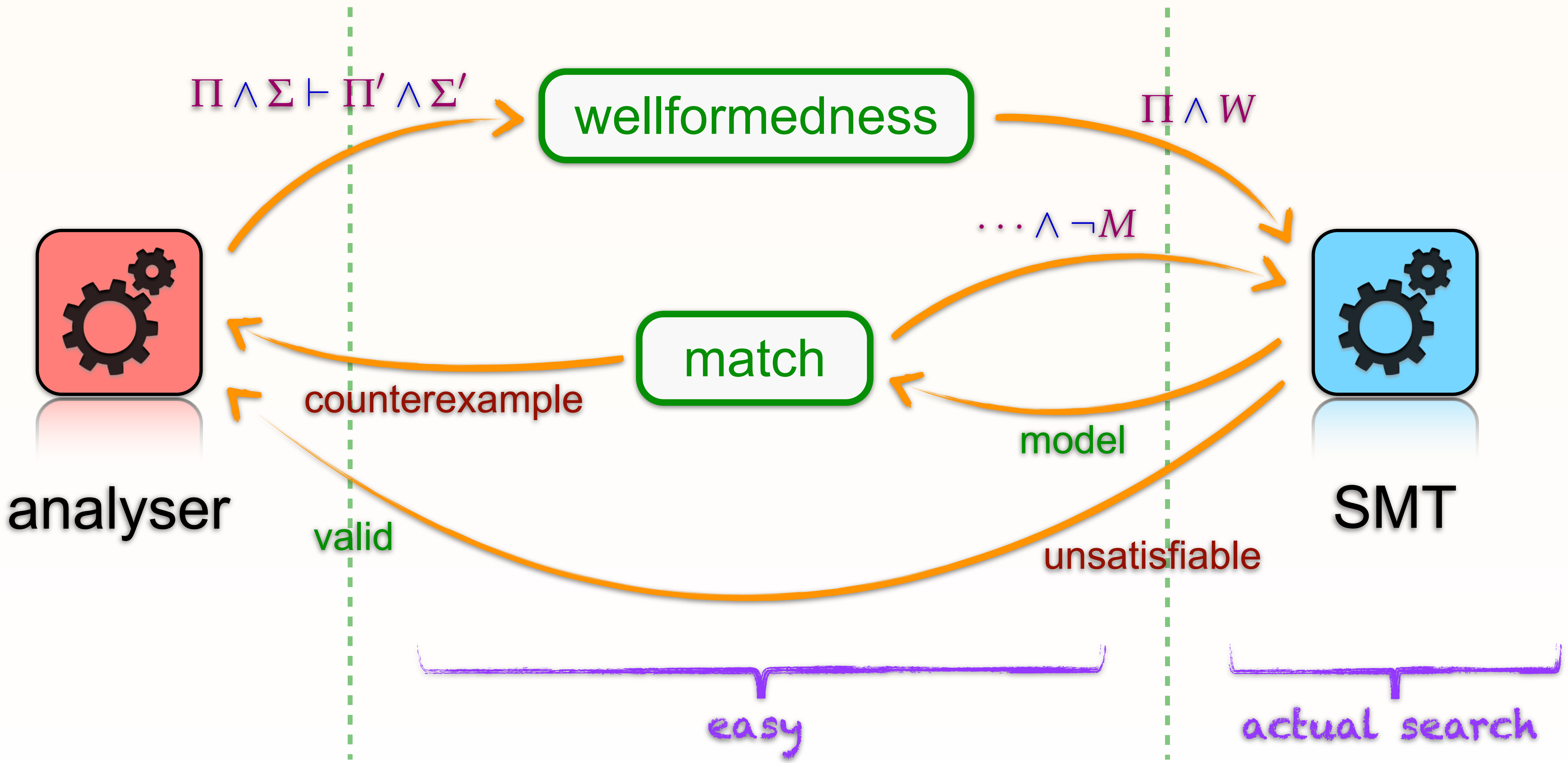
SMT

Theorem proved

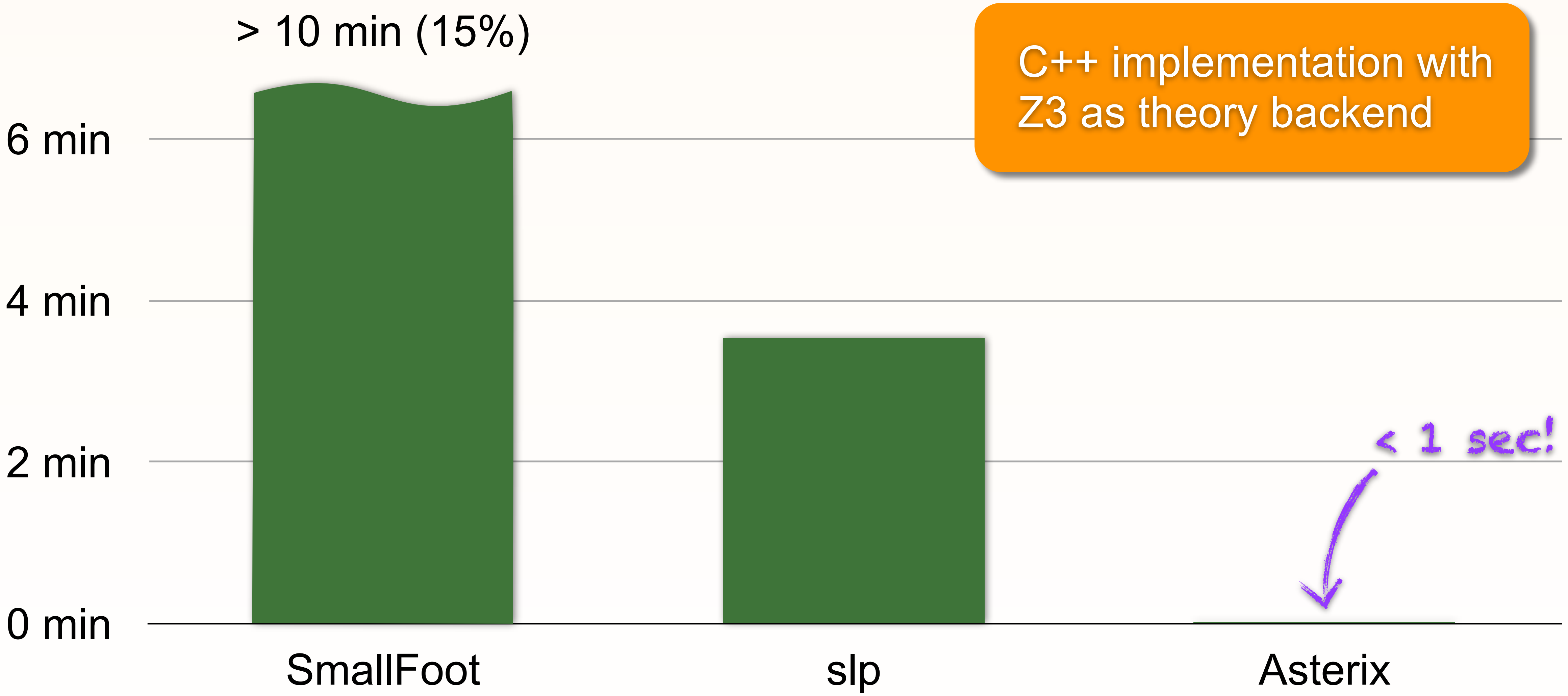
$$i = i' + 1 \wedge \mathsf{lseg}(p, q', i') * \mathsf{next}(q', q) * \mathsf{lseg}(q, \mathsf{nil}, n - i' - 1)$$
$$\vdash \mathsf{lseg}(p, q, i) * \mathsf{lseg}(q, \mathsf{nil}, n - i)$$



Algorithm


$\Pi \wedge \Sigma \vdash \Pi' \wedge \Sigma'$
wellformedness
$\Pi \wedge W$
$\cdots \wedge \neg M$
match
counterexample
model
analyser
SMT
valid
unsatisfiable
easy
actual search

Implementation



C++ implementation with Z3 as theory backend

Time to solve 8x ‘clones’ of typical verification conditions

Conclusions

- We designed a decision procedure for Separation Logic entailments
- It leverages on existing SMT checker technology for theory reasoning
- An efficient implementation is provided
- Full text available at arXiv:1303.2489

Future work

- Integration with a static analyser
- Support for user defined predicate definitions (e.g. inductive data types)
- Reasoning about sharing and overlap in sub-structures (e.g. DAGs)