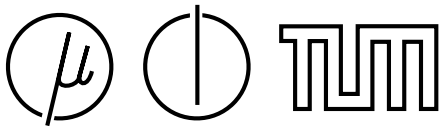
Separation Logic + Superposition Calculus =

Heap Theorem Prover

Juan A. Navarro Pérez and Andrey Rybalchenko



Fakultat für Informatik
Technische Universität München

Separation Logic + Superposition Calculus =

Heap Theorem Prover

1

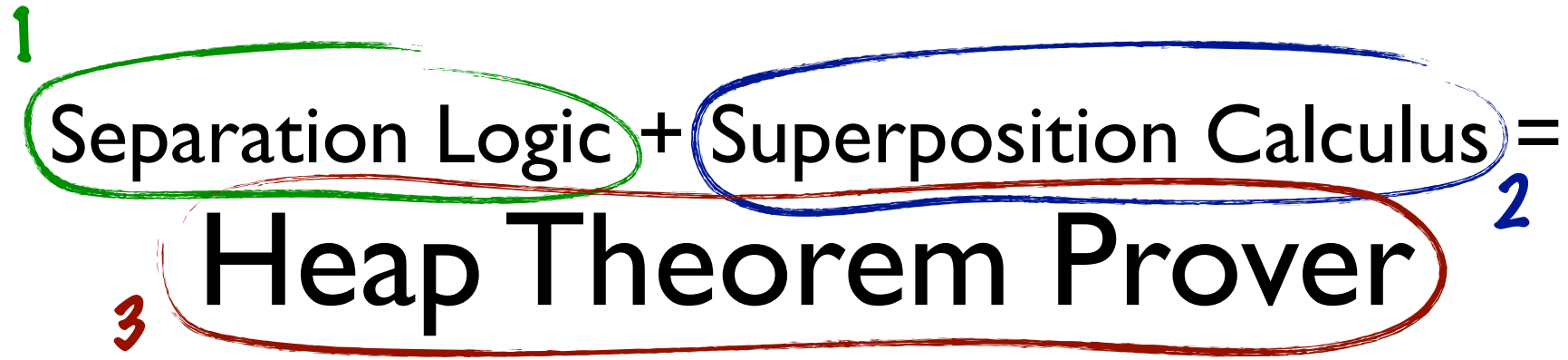
Separation Logic + Superposition Calculus =

Heap Theorem Prover

1

Separation Logic + Superposition Calculus = 2

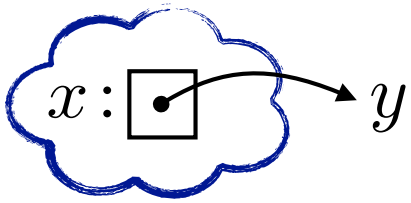
Heap Theorem Prover


1
Separation Logic + Superposition Calculus =
2
3
Heap Theorem Prover

Separation Logic

$\mathsf{next}(x, y)$

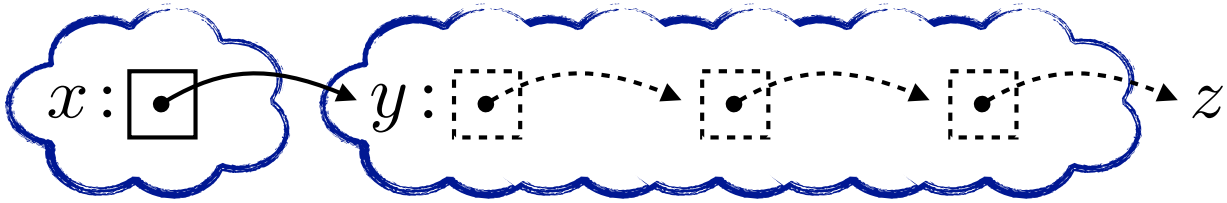
Heap
Memory



Separation Logic

$\mathsf{next}(x, y)$ $\mathsf{lseg}(y, z)$

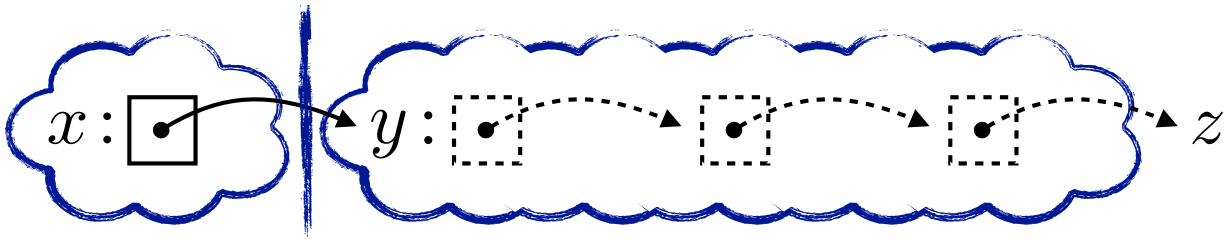
Heap Memory



Separation Logic

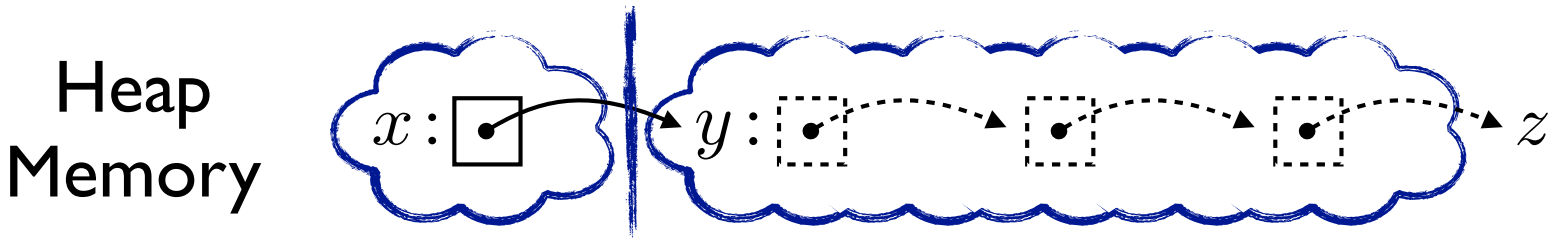
$\mathsf{next}(x, y) * \mathsf{lseg}(y, z)$

Heap Memory



Separation Logic

$\mathsf{next}(x, y) * \mathsf{lseg}(y, z)$



Symbolic Execution with Separation Logic *(Berdine et al. 2005)*

- Reasoning about heap structure and aliasing
- Calculus for entailments of the form: $\Pi \wedge \Sigma \rightarrow \Pi' \wedge \Sigma'$
- Search for a sequence of inferences that yields a proof

Superposition Calculus

$$\frac{L_1 \vee \cdots \vee L_n \vee x \simeq y \qquad R_1 \vee \cdots \vee R_m \vee x \simeq z}{L_1 \vee \cdots \vee L_n \vee R_1 \vee \cdots \vee R_m \vee y \simeq z}$$

Superposition Calculus

$$\frac{L_1 \vee \cdots \vee L_n \vee x \simeq y \qquad R_1 \vee \cdots \vee R_m \vee x \simeq z}{L_1 \vee \cdots \vee L_n \vee R_1 \vee \cdots \vee R_m \vee y \simeq z}$$

Superposition Calculus

$$\frac{L_1 \vee \cdots \vee L_n \vee x \simeq y \qquad R_1 \vee \cdots \vee R_m \vee x \simeq z}{L_1 \vee \cdots \vee L_n \vee R_1 \vee \cdots \vee R_m \vee y \simeq z}$$

- Equality reasoning with paramodulation *(Robinson & Wos 1969)*
- Superpostion inference rules *(Knuth & Bendix 1970)*
- Handbook of Automated Reasoning *(Nieuwenhuis & Rubio 2001)*

Superposition Calculus

$$\frac{L_1 \vee \cdots \vee L_n \vee x \simeq y \qquad R_1 \vee \cdots \vee R_m \vee x \simeq z}{L_1 \vee \cdots \vee L_n \vee R_1 \vee \cdots \vee R_m \vee y \simeq z}$$

- Equality reasoning with paramodulation *(Robinson & Wos 1969)*
- Superpostion inference rules *(Knuth & Bendix 1970)*
- Handbook of Automated Reasoning *(Nieuwenhuis & Rubio 2001)*

reasoning about **aliasing** = reasoning about **equality**

Separation Logic + Superposition Calculus =

Heap Theorem Prover

reasoning about
heap structure

reasoning about
aliasing/equality

Separation Logic + Superposition Calculus =

Heap Theorem Prover

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$$
$$\rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial	Pure
$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$	$c \not\simeq e$
$\neg\mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$	

Heap Theorem Prover

$$c \not\approx e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\approx e$

Heap Theorem Prover

$$c \not\approx e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\approx e$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$$

$$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

$$a \simeq b \vee \mathsf{lseg}(a,b) * \mathsf{lseg}(a,a) * \mathsf{next}(a,d) * \mathsf{lseg}(d,e)$$

Pure

$$c \not\simeq e$$

$$a \simeq b \vee a \simeq c$$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

$a \simeq b \vee \mathsf{lseg}(a,b) * \mathsf{lseg}(a,a)$

$* \mathsf{next}(a,d) * \mathsf{lseg}(d,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

$a \simeq b \vee \mathsf{lseg}(a,b) * \mathsf{next}(a,d) * \mathsf{lseg}(d,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

$a \simeq b \vee \mathsf{lseg}(a,b) * \mathsf{next}(a,d) * \mathsf{lseg}(d,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg\mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

$\mathsf{lseg}(a, a) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

$\mathsf{lseg}(a,a) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg\mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

$\mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg\mathsf{lseg}(a, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

$\mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(a,c) * \mathsf{lseg}(c,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

$\mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(a,c) * \mathsf{lseg}(c,e)$

$c \simeq e \vee \neg\mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

$\mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$c \simeq e \vee \neg\mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e) \rightarrow \mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$$

Spatial

$\mathsf{lseg}(a,b) * \mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$\neg\mathsf{lseg}(b,c) * \mathsf{lseg}(c,e)$

$\mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

$c \simeq e \vee \neg\mathsf{lseg}(a,c) * \mathsf{next}(c,d) * \mathsf{lseg}(d,e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

$c \simeq e$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

$c \simeq e$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

$c \simeq e$

Heap Theorem Prover

$$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e) \rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

$c \simeq e$

Heap Theorem Prover

$c \not\simeq e \wedge \mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

Theorem $\rightarrow \mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Spatial

$\mathsf{lseg}(a, b) * \mathsf{lseg}(a, c) * \mathsf{next}(c, d) * \mathsf{lseg}(d, e)$

$\neg\mathsf{lseg}(b, c) * \mathsf{lseg}(c, e)$

Pure

$c \not\simeq e$

$a \simeq b \vee a \simeq c$

$a \simeq b$

$c \simeq e$

Heap Theorem Prover

split the *entailment* **into:**
    pure clauses, a *positive spatial clause*, and a *negative spatial clause*
repeat
    repeat
        search for a *model* **of** the *pure clauses*
        return "Theorem" if no *model* **is found**
        normalize the *positive spatial clause* **w.r.t.** the *model*
        derive more *pure clauses* **by** well-formed inferences
    until the *pure clauses* **reach a fixpoint**
    return "Invalid" if the *model* **is a counterexample**
    normalize the *negative spatial clause* **w.r.t.** the *model*
    apply unfolding inferences **and resolve away** the *spatial clauses*
    return "Invalid" if the *spatial clauses* **don't match**
forever

Heap Theorem Prover

split the *entailment* **into:**
    pure clauses, a *positive spatial clause*, and a *negative spatial clause*
repeat
    repeat
        search for a *model* **of** the *pure clauses*
        return **"Theorem"** **if** no *model* **is found**
        normalize the *positive spatial clause* **w.r.t.** the *model*
        derive more *pure clauses* **by** well-formed inferences
    until the *pure clauses* **reach a fixpoint**
    return **"Invalid"** **if** the *model* **is a counterexample**
    normalize the *negative spatial clause* **w.r.t.** the *model*
    apply unfolding inferences **and resolve away** the *spatial clauses*
    return **"Invalid"** **if** the *spatial clauses* **don't match**
forever

no search

Heap Theorem Prover

split the *entailment* **into:**
    pure clauses, a *positive spatial clause*, and a *negative spatial clause*
repeat
    repeat
        search for a *model* **of** the *pure clauses*
        return **"Theorem"** **if** no *model* **is found**
        normalize the *positive spatial clause* **w.r.t.** the *model* **no search**
        derive more *pure clauses* **by** well-formed inferences **no search**
    until the *pure clauses* **reach a fixpoint**
    return **"Invalid"** **if** the *model* **is a counterexample**
    normalize the *negative spatial clause* **w.r.t.** the *model*
    apply unfolding inferences **and resolve away** the *spatial clauses*
    return **"Invalid"** **if** the *spatial clauses* **don't match**
forever

Heap Theorem Prover

split the *entailment* **into:**
    pure clauses, a *positive spatial clause*, and a *negative spatial clause*
repeat
    repeat
        search for a *model* **of** the *pure clauses*
        return "Theorem" if no *model* **is found**
        normalize the *positive spatial clause* **w.r.t.** the *model* **no search**
        derive more *pure clauses* **by** well-formed inferences **no search**
    until the *pure clauses* **reach a fixpoint**
    return "Invalid" if the *model* **is a counterexample**
    normalize the *negative spatial clause* **w.r.t.** the *model* **no search**
    apply unfolding inferences **and resolve away** the *spatial clauses*
    return "Invalid" if the *spatial clauses* **don't match**
forever

Heap Theorem Prover

split the *entailment* **into:**
    pure clauses, a *positive spatial clause*, and a *negative spatial clause*
repeat
    repeat
        search for a *model* **of** the *pure clauses*
        return "Theorem" if no *model* **is found**
        normalize the *positive spatial clause* **w.r.t.** the *model* **no search**
        derive more *pure clauses* **by** well-formed inferences **no search**
    until the *pure clauses* **reach a fixpoint**
    return "Invalid" if the *model* **is a counterexample**
    normalize the *negative spatial clause* **w.r.t.** the *model* **no search**
    apply unfolding inferences **and resolve away** the *spatial clauses* **no search**
    return "Invalid" if the *spatial clauses* **don't match**
forever

Heap Theorem Prover

split the *entailment* **into:**
    pure clauses, a *positive spatial clause*, and a *negative spatial clause*
repeat
    repeat
        search for a *model* **of** the *pure clauses* — **search**
        return "Theorem" if no *model* **is found**
        normalize the *positive spatial clause* **w.r.t.** the *model* — **no search**
        derive more *pure clauses* **by** well-formed inferences — **no search**
    until the *pure clauses* **reach a fixpoint**
    return "Invalid" if the *model* **is a counterexample**
    normalize the *negative spatial clause* **w.r.t.** the *model* — **no search**
    apply unfolding inferences **and resolve away** the *spatial clauses* — **no search**
    return "Invalid" if the *spatial clauses* **don't match**
forever

reasoning about
heap structure

reasoning about
aliasing/equality

Separation Logic + Superposition Calculus =

Heap Theorem Prover

reasoning about
heap structure

reasoning about
aliasing/equality

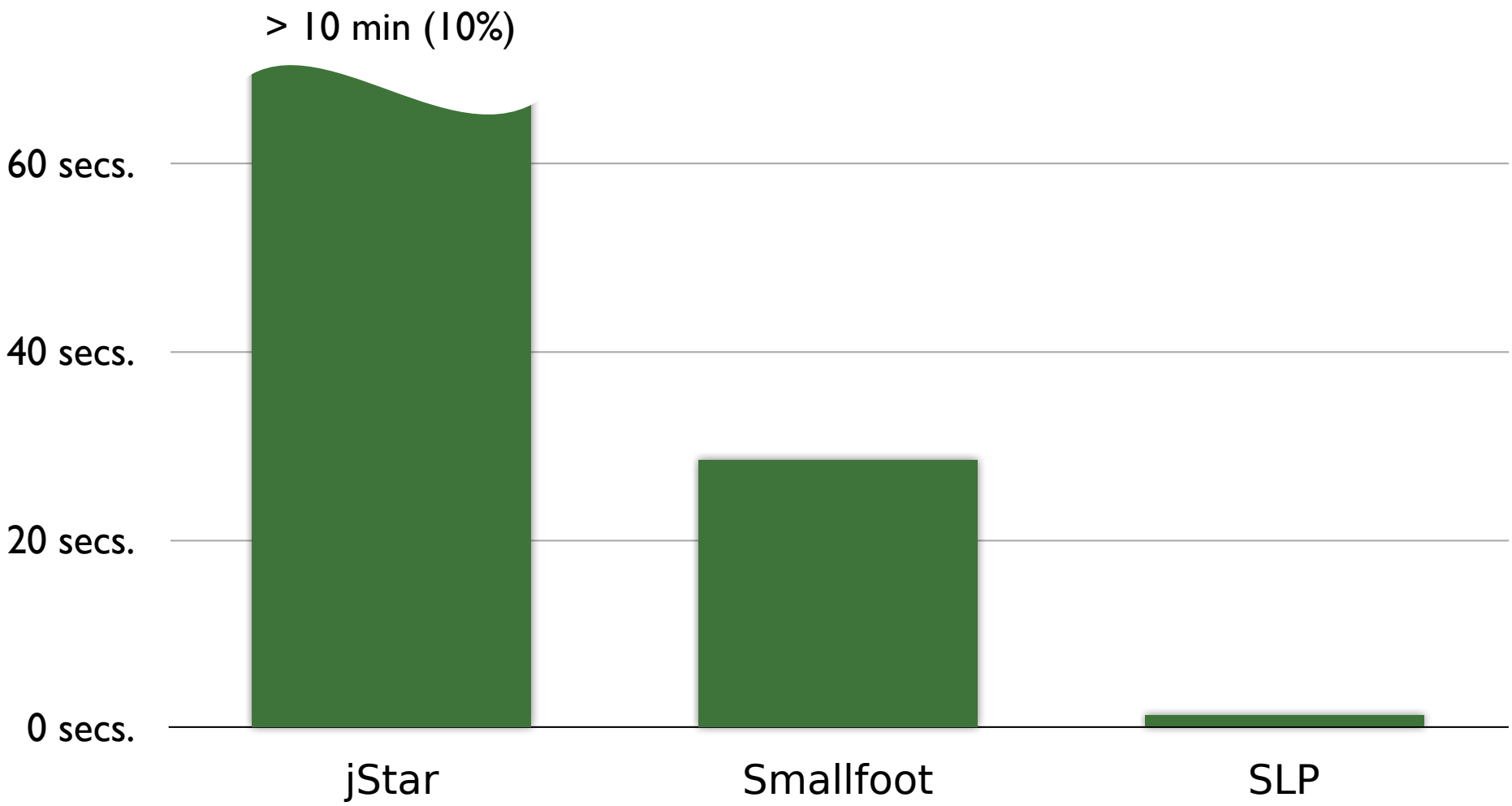
Separation Logic + Superposition Calculus =

Heap Theorem Prover

no search

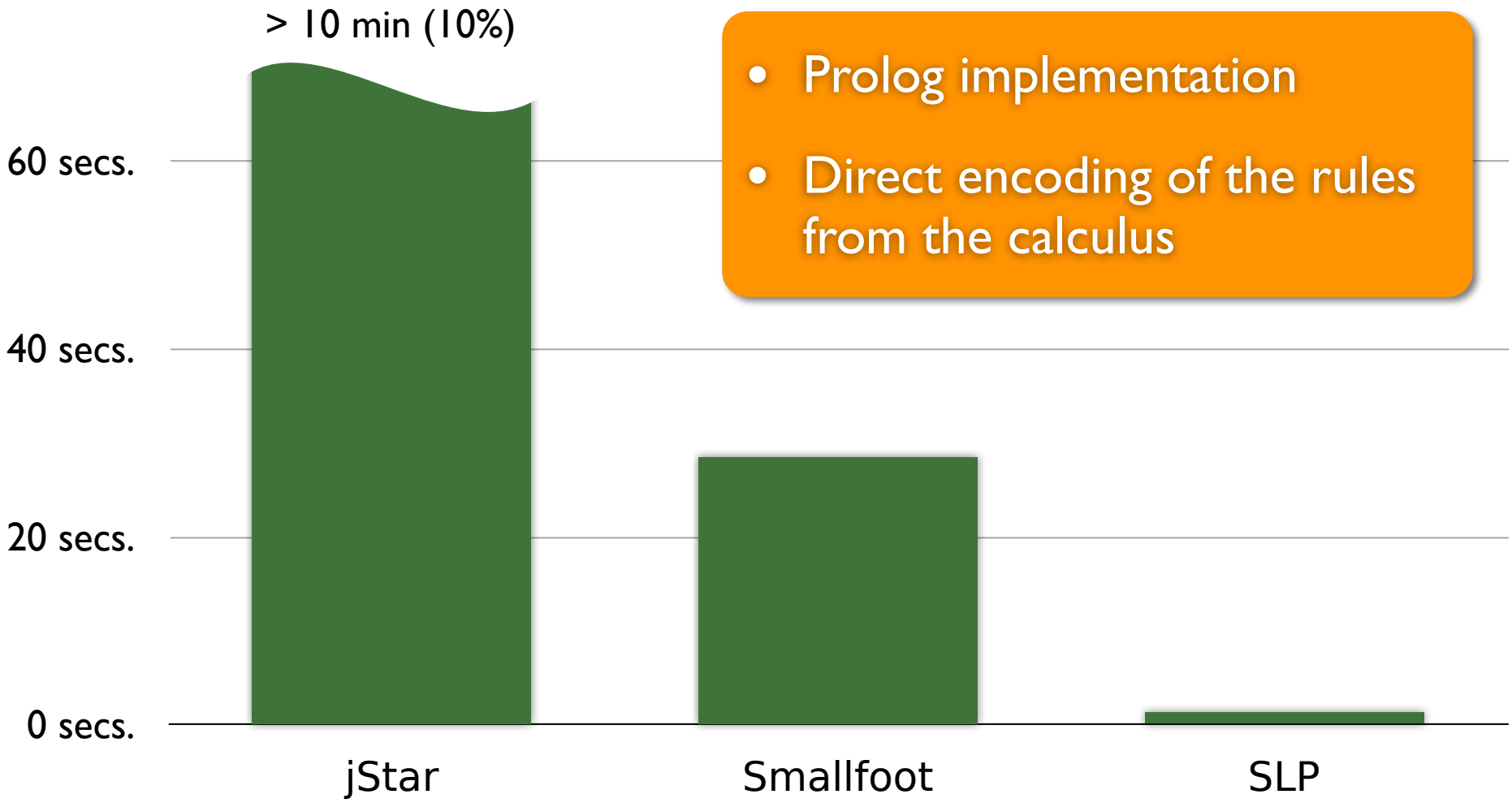
search

Heap Theorem Prover



Time to solve 1 000 synthetic "unfolding" entailments

Heap Theorem Prover



- Prolog implementation
- Direct encoding of the rules from the calculus

Time to solve 1 000 synthetic "unfolding" entailments

What's next?

- Extension with other data structures (e.g. trees)
- Combination with other theories
  - Linear Arithmetic *(Korovin & Voronkov 2007)*
  - SMT *(Baumgartner & Waldmann 2009; de Moura & Bjørner 2009)*
- Verified Software Toolchain *(Appel 2011)*
- Ongoing implementation of a model checker

reasoning about
heap structure

reasoning about
aliasing/equality

Separation Logic + Superposition Calculus =

Heap Theorem Prover

no search

search