Frame Inference for Inductive Entailment Proofs in Separation Logic

Quang Loc Le (TU) Jun Sun (SUTD) Shengchao Qin (TU)

TACAS 2018, Thessaloniki - Greece

April 16, 2018

Research Problem

Frame Inference:

- Input: Separation logic formulae A, C
- Question: Does exist a formula ?frame, the "leftover" portions of heap, which makes the following entailment valid?

$$A \models C * ?frame$$

Challenges:

- Unbounded heaps
- Infinite numerical domain

Our Contribution

`S2ENT`, an automatic procedure to infer frame for Inductive Entailment Proofs in Separation Logic.

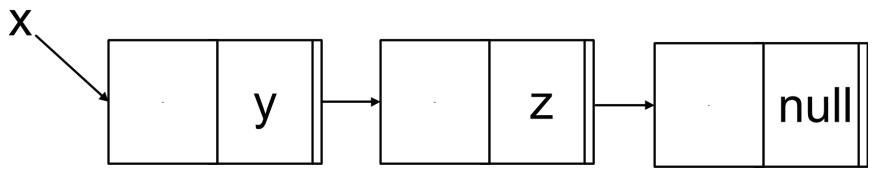
A fragment of Separation Logic

$$
\begin{array}{lll}
\text{Formula} & \Phi & ::= \Delta \mid \Phi_1 \vee \Phi_2 \qquad\qquad \Delta ::= \exists \bar{v}.\ (\kappa \wedge \pi) \\
\text{Spatial formula} & \kappa & ::= \mathtt{emp} \mid x \mapsto c(v_i) \mid \mathtt{P}(\bar{v}) \mid \kappa_1 * \kappa_2 \\
\text{Pure formula} & \pi & ::= \pi_1 \wedge \pi_2 \mid \alpha \mid \phi
\end{array}
$$

- α: Pointer (Dis)Equalities
- ϕ: Presburger arithmetic
- $\mathtt{P}$: inductive predicate. Predicate Definition: $\mathtt{P}(\bar{t}) \equiv \Phi$

Warning: no pointer arithmetic and no magic wand

Inductive Predicates



- Singly-linked list

$$ll(\texttt{root}) \equiv \texttt{emp} \wedge \texttt{root}=\texttt{null} \vee \exists\, r.\ \texttt{root} \mapsto node(r) * ll(r);$$

- Singly-linked list with size property

$$\texttt{lln}(\texttt{root},n) \equiv \texttt{emp} \wedge \texttt{root}=\texttt{null} \wedge n=0 \vee \exists\, q, n_1.\texttt{root} \mapsto \texttt{node}(q) * \texttt{lln}(q,n_1) \wedge n=n_1+1;$$

- Singly-linked list with last pointer and size property

$$\texttt{ll\_last}(\texttt{root},l,n) \equiv \texttt{root} \mapsto \texttt{node}(\texttt{null}) \wedge l=\texttt{root} \wedge n=1 \vee \exists q,n_1.\ \texttt{root} \mapsto \texttt{node}(q) * \texttt{ll\_last}(q,l,n_1) \wedge n=n_1+1;$$

Motivation - Hoare-style & Compositional Verification

Verification Problem: Memory Safety and Functional Correctness.

$\{\,\texttt{lln}(x,n)\wedge n>0\,\}\ \texttt{res}=\texttt{last}(\texttt{x})\ \{\,\texttt{ll\_last}(x,\texttt{res},n)\wedge n>0\,\}$

```
1  node append(node x, node y)
2    requires lln(x,i)*lln(y,j)∧i>0
3    ensures lln(res,i+j);
4  {
5    node t=last(x);

6    t->next=y;

7    return x;}
```

Motivation - Hoare-style & Compositional Verification

VC1: $\texttt{lln}(x,i)*\texttt{lln}(y,j)\wedge i>0 \models \texttt{lln}(x,i)\wedge i>0 * ?frame$

$\{\ \texttt{lln}(x,i)\wedge i>0\ \}\quad \texttt{t} = \texttt{last}(\texttt{x})\quad \{\ \texttt{ll\_last}(x,t,i)\wedge i>0\ \}$

```
1  node append(node x, node y)
2    requires lln(x,i)*lln(y,j)∧i>0
3    ensures lln(res,i+j);
4  { //lln(x,i)*lln(y,j)∧i>0                                   VC1
5    node t=last(x);

6    t->next=y;

7    return x;}
```

Motivation - Hoare-style & Compositional Verification

VC1: $\texttt{lln}(x,i)*\texttt{lln}(y,j)\wedge i>0 \models (\texttt{lln}(x,i)\wedge i>0) * \texttt{lln}(y,j)$

$\{\ \texttt{lln}(x,i)\wedge i>0\ \}\quad \texttt{t} = \texttt{last}(\texttt{x})\quad \{\ \texttt{ll\_last}(x,t,i)\wedge i>0\ \}$

```
1  node append(node x, node y)
2    requires lln(x,i)*lln(y,j)∧i>0
3    ensures lln(res,i+j);
4  { //lln(x,i)*lln(y,j)∧i>0                                   VC1
5    node t=last(x);

6    t->next=y;

7    return x;}
```

Motivation - Compositional Verification

FRAME RULE

$$\frac{\{\ \mathtt{lln}(x,i)\wedge i>0\ \}\quad \mathtt{t=last(x)}\quad \{\ \mathtt{ll\_last}(x,t,i)\wedge i>0\ \}}{\{\ \mathtt{lln}(x,i)*\mathtt{lln}(y,j)\wedge i>0\ \}\quad \mathtt{t=last(x)}\quad \{\ \mathtt{ll\_last}(x,t,i)*\mathtt{lln}(y,j)\wedge i>0\ \}}$$

```
1  node append(node x, node y)
2    requires lln(x,i)*lln(y,j)∧i>0
3    ensures lln(res,i+j);
4  {//lln(x,i)*lln(y,j)∧i>0
5    node t=last(x);
     //ll_last(x,t,i)*lln(y,j)∧i>0
6    t->next=y;

7    return x;}
```

Motivation - Compositional Verification

VC2:

$$\texttt{ll\_last}(x,t,i)*\texttt{lln}(y,j)\wedge i>0 \models \exists q.\, t\mapsto\texttt{node}(q) * \texttt{lsegn}(x,t,i-1)*\texttt{lln}(y,j)\wedge i>0$$

```
1  node append(node x, node y)
2    requires lln(x,i)*lln(y,j)∧i>0
3    ensures lln(res,i+j);
4  {//lln(x,i)*lln(y,j)∧i>0
5    node t=last(x);
   //ll_last(x,t,i)*lln(y,j)∧i>0
6    t->next=y;
   // lsegn(x,t,i−1)*t↦node(y)* lln(y,j)∧i>0
7    return x;}
```

Motivation - Compositional Verification

VC3:

$$\texttt{lsegn}(\texttt{res},t,i-1)*t\mapsto\texttt{node}(y)*\texttt{lln}(y,j)\wedge i>0 \models \texttt{lln}(\texttt{res},i+j) * \texttt{emp}$$

```
1  node append(node x, node y)
2    requires lln(x,i)*lln(y,j)∧i>0
3    ensures lln(res,i+j);
4  {//lln(x,i)*lln(y,j)∧i>0
5    node t=last(x);
   //ll_last(x,t,i)*lln(y,j)∧i>0
6    t->next=y;
   //lsegn(x,t,i−1)*t↦node(y)*lln(y,j)∧i>0
7    return x;}
```

// lsegn(res,t,i−1)∗t↦node(y)∗ lln(y,j)∧i>0

Motivation - Compositional Verification

$$VC1: \quad \texttt{lln}(x,i)*\texttt{lln}(y,j)\wedge i>0 \models \texttt{lln}(x,i)\wedge i>0) * \texttt{lln}(y,j)$$

$$VC2: \quad \texttt{ll\_last}(x,t,i)*\texttt{lln}(y,j)\wedge i>0 \models \exists q.\texttt{t}\mapsto\texttt{node}(q) * \texttt{lsegn}(x,t,i-1)*\texttt{lln}(y,j)$$

$$VC3: \quad \texttt{lsegn}(\texttt{res},t,i-1)*t\mapsto\texttt{node}(y)*\texttt{lln}(y,j)\wedge i>0 \models \texttt{lln}(\texttt{res},i+j) * \texttt{emp}$$

$\{\ \texttt{lln}(x,n)\wedge n>0\ \}\ \texttt{res} = \texttt{last}(\texttt{x})\ \{\ \texttt{ll\_last}(x,\texttt{res},n)\wedge n>0\ \}$

```
1  node append(node x, node y)
2    requires lln(x,i)*lln(y,j)∧i>0
3    ensures lln(res,i+j);
4  { // lln(x,i)*lln(y,j)∧i>0                                   VC1
5    node t=last(x);
   // ll_last(x,t,i)*lln(y,j)∧i>0                               VC2
6    t->next=y;
   // lsegn(x,t,i−1)*t↦node(y)* lln(y,j)∧i>0
7    return x;}
   // lsegn(res,t,i−1)*t↦node(y)* lln(y,j)∧i>0                  VC3
```

Frame Inference for Inductive Entailment Proofs

$$VC1: \quad \mathtt{lln}(x,i) * \mathtt{lln}(y,j) \wedge i>0 \models \mathtt{lln}(x,i) \wedge i>0) * \mathtt{lln}(y,j)$$

$$VC2: \quad \mathtt{ll\_last}(x,t,i) * \mathtt{lln}(y,j) \wedge i>0 \models \exists q.\mathtt{t} \mapsto \mathtt{node}(q) * \mathtt{lsegn}(x,t,i-1) * \mathtt{lln}(y,j)$$

$$VC3: \quad \mathtt{lsegn}(\mathtt{res},t,i-1) * t \mapsto \mathtt{node}(y) * \mathtt{lln}(y,j) \wedge i>0 \models \mathtt{lln}(\mathtt{res},i+j) * \mathtt{emp}$$

How to deal with infinite domains?

Frame Inference for Inductive Entailment Proofs

Three Key Ideas:

1. Frame Inference: shape and then pure properties
   - Unknown predicates as place-holders

$$E_0: \quad \texttt{ll\_last}(x,t,i) * \texttt{lln}(y,j) \wedge i > 0 \models \exists q.\texttt{t} \mapsto \texttt{node}(q) \; * \; \texttt{U}(x,t,y,q)$$

   - Generate relational constraints $\mathcal{R}$ such that

> E_o is valid if $\mathcal{R}$ is satisfied.

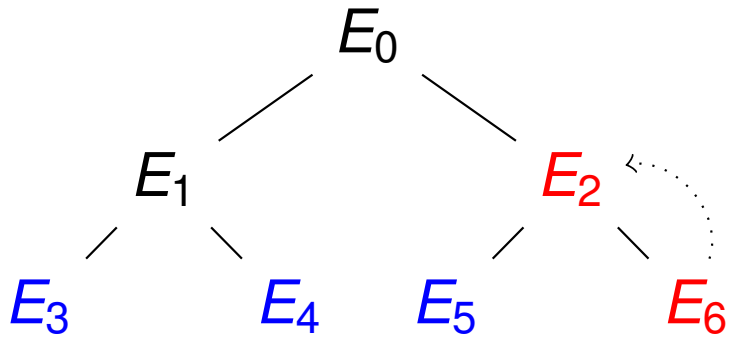
2. New cyclic proof system with Cyclic Cut rule
3. Predicate Normalization using Lemma Synthesis

Proofs by Induction

Proving Property P by induction on natural number n

- Base case: prove that P holds when $n = 0$
- Inductive case:
  - $\frac{\text{induction hypothesis (IH)}:}{P \text{ holds when } n = k}$
  - prove that P holds when $n = k+1$

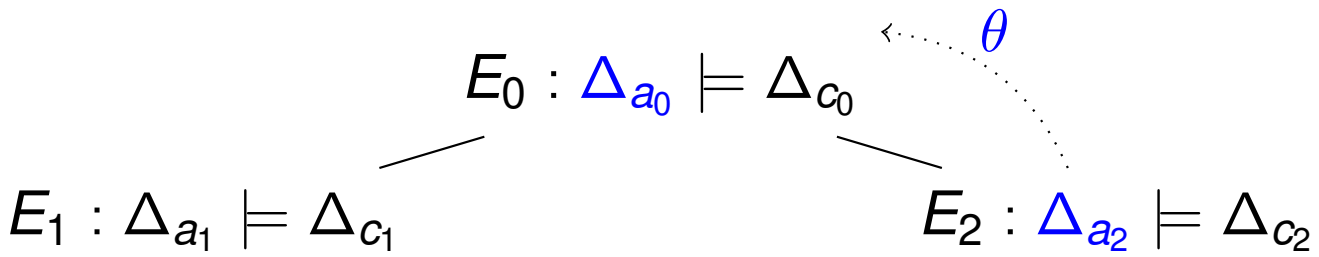
Cyclic Proof (J. Brotherston - UCL)



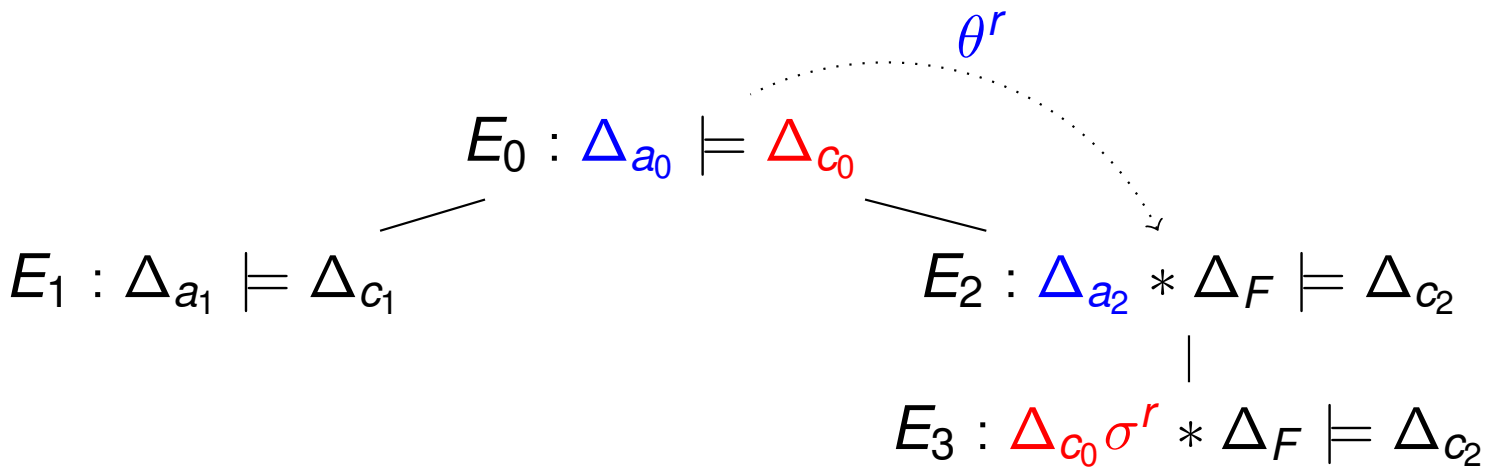
Automatically detect IH E_2

- Weaken E_6 to E'_6
- Find θ s.t. $E'_6\theta \equiv E_2$
- Global soundness condition

Cyclic Proofs



(a) $\texttt{Cyclic}_{SL}$- J. Brotherston: Full back-link



(b) Our Proposal: Partial back-link

Proposed Cyclic Cut Rule

$$\text{CUT}\ \frac{A \models B \qquad B \models C}{A \models C}$$

$$\text{CUT}\ \frac{\Delta_{a_0}\theta^r * \Delta_{a_F} \models \Delta_{c_0}\theta^r * \Delta_{a_F} \qquad E_3 : \Delta_{c_0}\theta^r * \Delta_{a_F} \models \Delta_{c_2}}{E_2 : \Delta_{a_2} * \Delta_{a_F} \models \Delta_{c_2}}$$

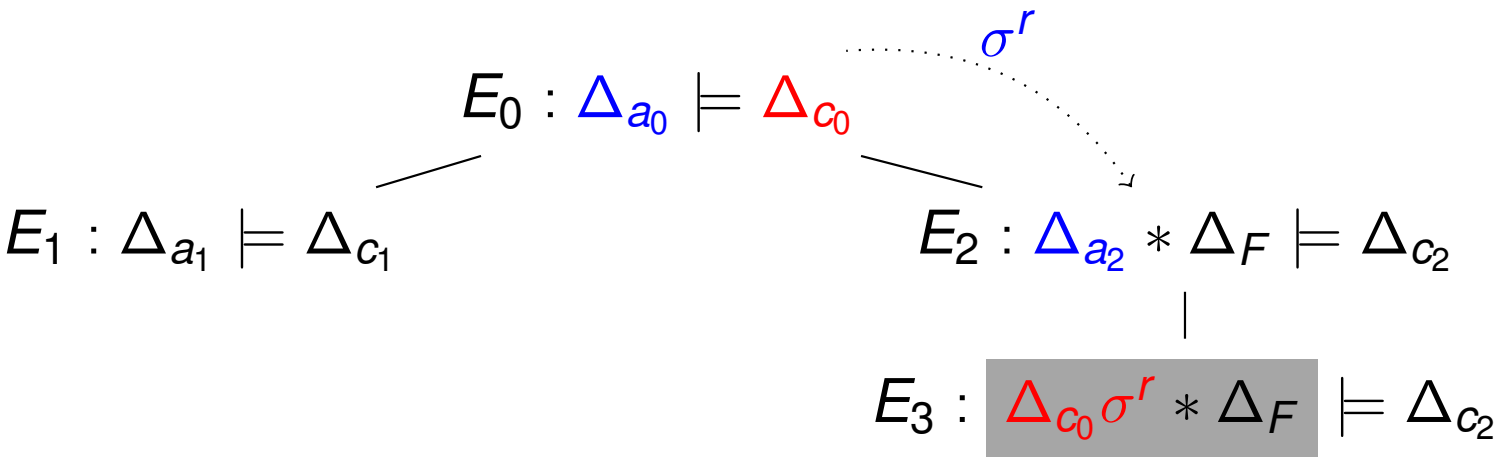


Figure : Our Proposal: Partial back-link for Cut Inference.

Proposed Cyclic Cut Rule

$$\text{Conseq}\ \frac{\Delta_{a_2} \equiv \Delta_{a_0}\theta^r \qquad \text{Subst}\ \dfrac{IH:\ \ \Delta_{a_0} \models \Delta_{c_0}}{\Delta_{a_0}\theta^r \models \Delta_{c_0}\theta^r}}{*\text{-Intro}\ \dfrac{\Delta_{a_2} \models \Delta_{a_0}\theta^r}{\Delta_{a_2} * \Delta_F \models \Delta_{a_0}\theta^r * \Delta_F}}$$

$$\text{CUT}\ \frac{\Delta_{a_0}\theta^r * \Delta_{a_F} \models \Delta_{c_0}\theta^r * \Delta_{a_F} \qquad E_3 : \Delta_{c_0}\theta^r * \Delta_{a_F} \models \Delta_{c_2}}{E_2 : \Delta_{a_2} * \Delta_{a_F} \models \Delta_{c_2}}$$

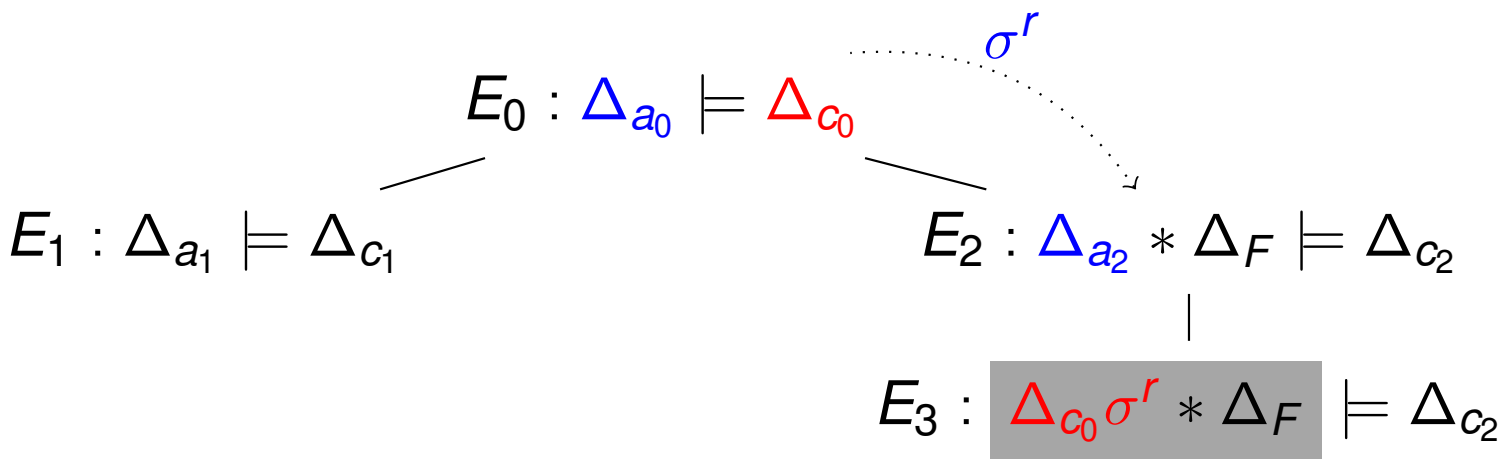


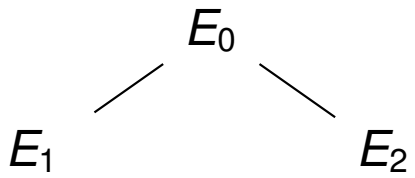
Figure : Our Proposal: Partial back-link for Cut Inference.

Example - VC2

$$E_0$$

$$E_0 : \texttt{ll\_last}(x,t,i) * \texttt{lln}(y,j) \wedge i > 0 \models \exists q.\texttt{t} \mapsto \texttt{node}(q) * \texttt{U}(x,t,q,y)$$

Example - VC2



$E_0 : \texttt{ll\_last}(x,t,i) * \texttt{lln}(y,j) \wedge i>0 \models \exists q.\texttt{t} \mapsto \texttt{node}(q) * \texttt{U}(x,t,q,y)$

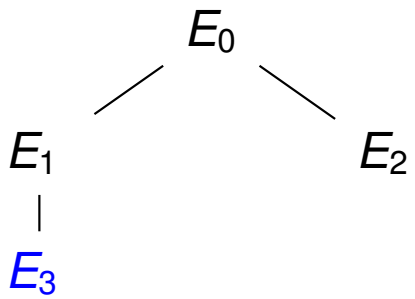
Unfold `ll_last` in LHS

$$\texttt{ll\_last}(\texttt{root},lst,n) \equiv \texttt{root} \mapsto \texttt{node}(\texttt{null}) \wedge \texttt{root}=lst \wedge n=1 \vee \exists q.\ \texttt{root} \mapsto \texttt{node}(q) * \texttt{ll\_last}(q,lst,n-1);$$

$$E_1 : \texttt{t} \mapsto \texttt{node}(\texttt{null}) * \texttt{lln}(y,j) \wedge i>0 \wedge \texttt{x}=t \wedge i=1 \models \exists q.\texttt{t} \mapsto \texttt{node}(v,q) * \texttt{U}(x,t,q,y)$$

$$E_2 : \exists q_1.\ \texttt{x} \mapsto \texttt{node}(q_1) * \texttt{ll\_last}(q_1,t,i-1) * \texttt{lln}(y,j) \wedge i>0 \models \exists q.\texttt{t} \mapsto \texttt{node}(v,q) * \texttt{U}(x,t,q,y)$$

Example - VC2



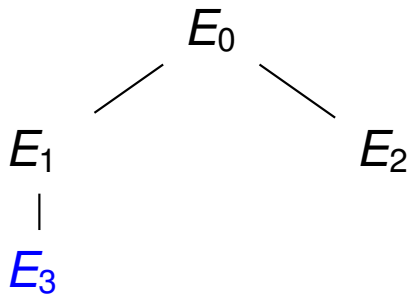
$E_1 : \mathtt{t} \mapsto \mathtt{node}(\mathtt{null}) * \mathtt{lln}(y,j) \wedge i > 0 \wedge \mathtt{x} = t \wedge i = 1$
$\models \exists q.\mathtt{t} \mapsto \mathtt{node}(q) * \mathtt{U}(x,t,q,y)$

Matching: unify $\mathtt{t} \mapsto \mathtt{node}(...)$ both sides

$E_3 : \mathtt{lln}(y,j) \wedge i > 0 \mathtt{x} = t \wedge i = 1 \wedge q = \mathtt{null}$
$\models \mathtt{U}(x,t,q,y)$

$E_2 : \exists q_1 .\ \mathtt{x} \mapsto \mathtt{node}(q_1) * \mathtt{ll\_last}(q_1,t,i-1) * \mathtt{lln}(y,j) \wedge i > 0$
$\models \exists q.\mathtt{t} \mapsto \mathtt{node}(q) * \mathtt{U}(x,t,q,y)$

Example - VC2



$E_3 : \texttt{emp} \wedge \texttt{x}{=}t \wedge i{=}1 * \texttt{lln}(y,j) \wedge i{>}0 \wedge q{=}\texttt{null} \models \texttt{U}(x,t,q,y)$

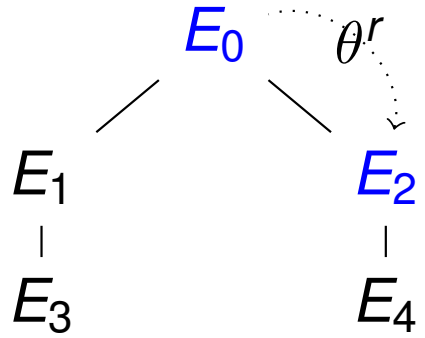
Abduction: synthesize constraint σ_1

$$\sigma_1 : \texttt{lln}(y,\_) \wedge t{=}x \wedge q{=}\texttt{null} \Rightarrow \texttt{U}(x,t,q,y)$$

$$E_2 : \exists q_1 .\ \texttt{x} \mapsto \texttt{node}(q_1) * \texttt{ll\_last}(q_1,t,i{-}1) * \texttt{lln}(y,j) \wedge i{>}0 \models \exists q . \texttt{t} \mapsto \texttt{node}(q) * \texttt{U}(x,t,q,y)$$

Example - VC2

σ_1: $\mathtt{lln}(y,\_) \wedge t{=}x \wedge q{=}\mathtt{null} \Rightarrow \mathtt{U}(x,t,q,y)$



$E_2 : \exists q_1 .\ \mathtt{x}{\mapsto}\mathtt{node}(q_1) * \mathtt{ll\_last}(q_1,t,i{-}1) * \mathtt{lln}(y,j) \wedge i{>}0 \models \exists q.\mathtt{t}{\mapsto}\mathtt{node}(q) * \mathtt{U}(x,t,q,y)$

Induction: generate back-link

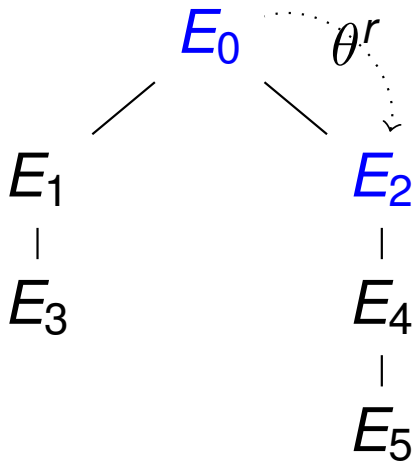
$E_0 : \mathtt{ll\_last}(x,t,i) * \mathtt{lln}(y,j) \wedge i{>}0 \models \exists q.\mathtt{t}{\mapsto}\mathtt{node}(q) * \mathtt{U}(x,t,q,y)$

$\theta^r \equiv [q_1/x; i{-}1/i]$

$E_4 : \exists q_1 .\ \exists q.\mathtt{t}{\mapsto}\mathtt{node}(q) * \mathtt{U}(q_1,t,q,y) * \mathtt{x}{\mapsto}\mathtt{node}(q_1) \wedge i{-}1{>}0 \models \exists q.\mathtt{t}{\mapsto}\mathtt{node}(q) * \mathtt{U}(x,t,q,y)$

Example - VC2

σ_1: $\mathtt{lln}(y,\_)\wedge t{=}x\wedge q{=}\mathtt{null}$
$\Rightarrow \mathtt{U}(x,t,q,y)$



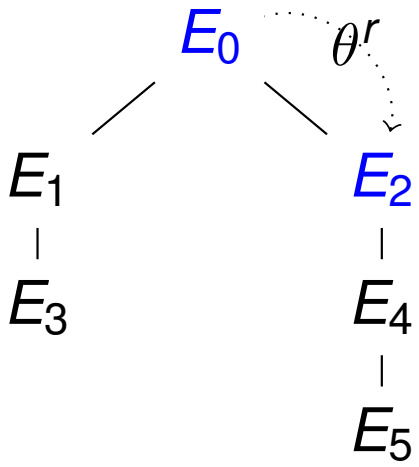
$E_4 : \exists q_1.\ \exists q.\mathtt{t}{\mapsto}\mathtt{node}(q) * \mathtt{U}(q_1,t,q,y) * \mathtt{x}{\mapsto}\mathtt{node}(q_1)\wedge i{-}1{>}0$
$\models \exists q.\mathtt{t}{\mapsto}\mathtt{node}(q) * \mathtt{U}(x,t,q,y)$

Matching: Unify $\mathtt{t}{\mapsto}\mathtt{node}(..)$ in both sides

$E_5 : \exists q_1.\ \mathtt{U}(q_1,t,q,y) * \mathtt{x}{\mapsto}\mathtt{node}(q_1)\wedge i{-}1{>}0$
$\models \mathtt{U}(x,t,q,y)$

Example - VC2

σ_1: $\texttt{lln}(y,\_) \wedge t = x \wedge q = \texttt{null}$
$\Rightarrow \texttt{U}(x,t,q,y)$



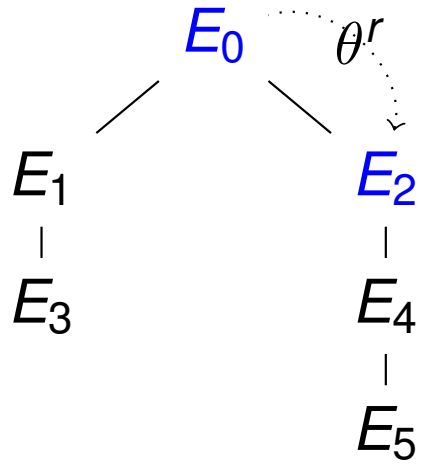
$E_5 : \exists q_1 .\ \texttt{U}(q_1,t,q,y) * \texttt{x} \mapsto \texttt{node}(q_1) \wedge i-1>0 \ \models\ \texttt{U}(x,t,q,y)$

Abduction: synthesize constraint σ_2

$$\sigma_2: x \mapsto \texttt{node}(x) * \texttt{U}(x,t,q,y) \Rightarrow \texttt{U}(x,t,q,y)$$

Example - VC2

σ_1: $\texttt{lln}(y,\_)\wedge t{=}x\wedge q{=}\texttt{null}$
$\Rightarrow \texttt{U}(x,t,q,y)$
σ_2: $x{\mapsto}\texttt{node}(x){*}\texttt{U}(x,t,q,y)$
$\Rightarrow \texttt{U}(x,t,q,y)$



E_0 is valid if $\sigma_1 \wedge \sigma_2$ is satisfied.

The satisfiability problem is solved by S2SAT (Le et. al - CAV16,17)

Example - VC2

$$\sigma_1: \mathtt{lln}(y,\_)\wedge t{=}x\wedge q{=}\mathtt{null} \;\Rightarrow\; \mathtt{U}(x,t,q,y)$$
$$\sigma_2: x{\mapsto}\mathtt{node}(x){*}\mathtt{U}(x,t,q,y) \;\Rightarrow\; \mathtt{U}(x,t,q,y)$$

1. Predicate Normalization with Lemma Synthesis

$$\mathtt{U}(x,t,q,y) \leftrightarrow \mathtt{lseg}(x,t) * \mathtt{lln}(y,\_) \wedge q{=}\mathtt{null}$$

VC2:

$$\mathtt{ll\_last}(x,t,i){*}\mathtt{lln}(y,j){\wedge}i{>}0 \models \mathtt{t}{\mapsto}\mathtt{node}(\mathtt{null}) * \mathtt{lseg}(x,t){*}\mathtt{lln}(y,\_)$$

2. Extend size property

$$\mathtt{ll\_last}(x,t,i){*}\mathtt{lln}(y,j){\wedge}i{>}0 \models \mathtt{t}{\mapsto}\mathtt{node}(\mathtt{null}) * \mathtt{lsegn}(x,t,m){*}\mathtt{lln}(y,n) \wedge p(i,j,m,n)$$

3. Infer constraints over size property

$$\mathtt{ll\_last}(x,t,i){*}\mathtt{lln}(y,j){\wedge}i{>}0 \models \mathtt{t}{\mapsto}\mathtt{node}(\mathtt{null}) * \mathtt{lsegn}(x,t,i{-}1){*}\mathtt{lln}(y,j)$$

More in the paper

- Soundness of Cyclic Proofs
- Predicate Normalization and Synthesis over Pure Properties
- Implementation: Based on SLEEK (Chin *et. al.*)
- Evaluation on 36 Frame Inference Problems
  - `Cyclic`$_{SL}$: solves 18 problems (shape only and frame is `emp`).
  - `songbird`: solves 25 problems (frame is `emp`).
  - `S2ENT`: solves 36 problems.
- Integrated into S2
  - Compositional Verification of Heap-Manipulating Programs in C library Glib open source (gslist.c, glist.c, gtree.c, gnode.c).

Frame Inference for Cyclic Entailment Proofs

- Infer shape and then pure properties
- New cyclic proof system with Cyclic Cut Rule

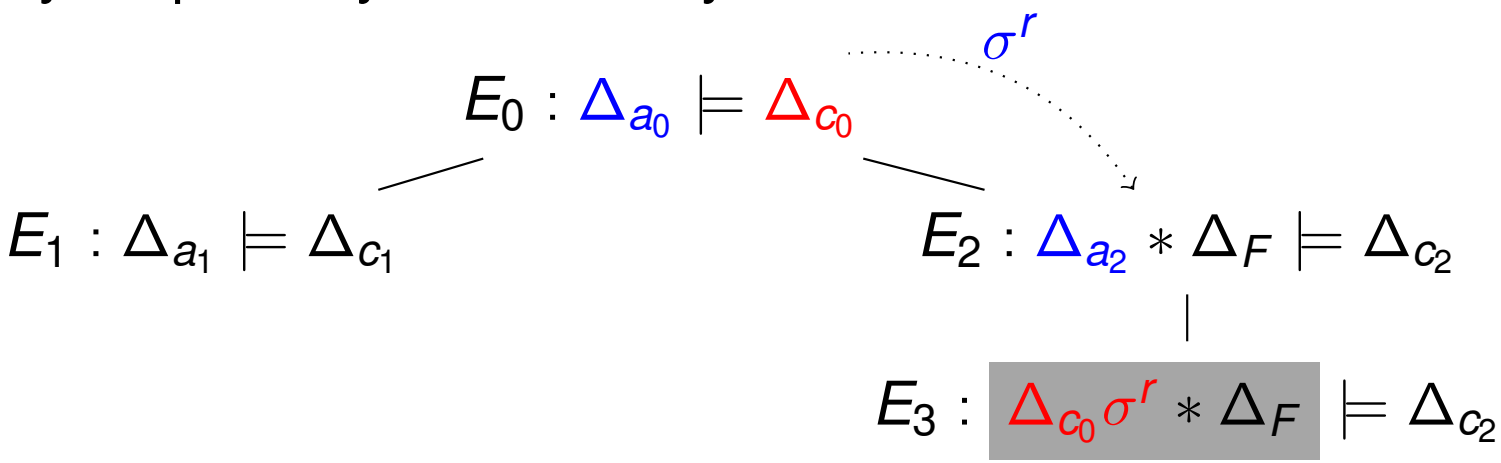


Figure : Our Proposal: Partial back-link for Cut Inference.

- Predicate Normalization using Lemma Synthesis