分離論理におけるエンテイルメント証明器の入力に対する制限の緩和

青木　洸佑（名古屋大学），中澤　巧爾（名古屋大学），木村　大輔（東邦大学）

はじめに

帰納的述語を含む分離論理のエンテイルメント判定問題のための証明体系CSLIDω[Tastuta+19]

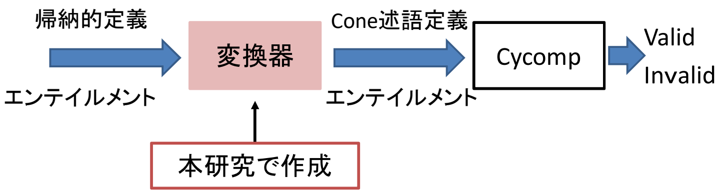
- ➢ 健全かつ完全，決定可能
- ➢ [Kimura+19]が自動証明器Cycompを実装

◆ 問題点：一定の条件(cone条件)を満たす述語のみが対象

- ➢ エンテイルメント判定器の国際競技会SL-COMP 2019のベンチマークの総問題数312問中cone条件を満たす問題は15問のみ

そこで，本研究では

- cone条件を満たす述語への同値変換手法を提案，実装
- SL-COMP 2019のベンチマークによるCycompの評価，実験



論理式(シンボリックヒープ)とエンテイルメント

- 論理式

t (term) $::= x \mid nil$

Π, Π'(pure part) $::= t = t \mid t \neq t \mid \Pi \wedge \Pi'$

Σ, Σ'(spatialpart) $::= t \mapsto (t_1, \ldots, t_n) \mid \mathrm{Pr}(t_1, \ldots, t_n) \mid emp \mid \Sigma * \Sigma'$

ψ(symbolic heap)$::= \exists x_1, \ldots, x_g. (\Pi \wedge \Sigma)$

Pr: 帰納的述語

- 帰納的述語は定義節$\phi_i(x, y_1, \ldots, y_n)$の集合で定義
  - ➢ $\mathrm{Pr}(x, y_1, \ldots, y_n) =_{def} \phi_1(x, y_1, \ldots, y_n) \mid \ldots \mid \phi_m(x, y_1, \ldots, y_n)$
- 意味論

$Val \ = N$

$Loc \ = N^+$(非負整数の集合)

$Store = Var \rightarrow Val$

$Heap = Loc \rightarrow_{fin} Val^n$

(s, h)　：現在のメモリ状態

$s(x)$　：変数xの値

$h(a)$　：アドレスaのメモリセルの値

- $s, h \vDash F$の解釈
  - ➢ $s, h \vDash x \mapsto (y_1)$ iff $\{[\![x]\!]_s\} = dom(h)$ and $h([\![x]\!]_s) = [\![y_1]\!]_s$
  - ➢ $s, h \vDash P * Q$ iff $\exists h_0, h_1. h_0 \# h_1, h_0 * h_1 = h, s, h_0 \vDash P$ and $s, h_1 \vDash Q$
  - ➢ $s, h \vDash emp$ iff $Dom(h) = 0$

Etc.

> エンテイルメント$\psi \vdash \phi_1, \ldots, \phi_n$が正しい $\Leftrightarrow_{def}$ $\forall s, h (s, h \vDash \psi \rightarrow \exists i. s, h \vDash \phi_i)$

帰納的述語のcone条件

定義節：$\phi(x, \vec{y}) \equiv \exists \vec{z}. \left(\Pi \wedge x \mapsto (\vec{u}) * *_{i \in I} P_i(z_i, \vec{t_i})\right)$

- $\{z_i | i \in I\} = \vec{z}$　　(strong establishment),
- $\vec{z} \subseteq \vec{u}$　　(decisiveness)

変換器の手続き（入力例ls(x,y)）

1. 述語Pの各定義節を帰納的述語を含んでいるものと含んでいないものに分ける。(帰納的述語を含む定義節のみによって新たな述語P'を定義する)

帰納的述語を含む定義節

$ls(x, y) \coloneqq (x = y \wedge emp)$
$\quad \mid \exists z(x \neq y \wedge x \mapsto z * ls(z, y))$

$\Rightarrow$

$ls(x, y) \coloneqq (x = y \wedge emp)$
$\quad \mid ls'(x, y)$
$ls'(x, y) \coloneqq \exists z(x \neq y \wedge x \mapsto z * ls(z, y))$

2. P'の定義節の述語展開を行う

$ls(x, y) \coloneqq (x = y \wedge emp)$
$\quad \mid ls'(x, y)$
$ls'(x, y) \coloneqq \exists z(x \neq y \wedge x \mapsto z * ls(z, y))$

$\Rightarrow$

$ls(x, y) \coloneqq (x = y \wedge emp)$
$\quad \mid ls'(x, y)$
$ls'(x, y) \coloneqq \exists z(x \neq y \wedge z = y \wedge x \mapsto z * emp)$
$\quad \mid \exists z(x \neq y \wedge x \mapsto z * ls'(z, y))$

3. P'の定義節の簡略を行う

$ls(x, y) \coloneqq (x = y \wedge emp)$
$\quad \mid ls'(x, y)$
$ls'(x, y) \coloneqq \exists z(x \neq y \wedge z = y \wedge x \mapsto z * emp)$
$\quad \mid \exists z(x \neq y \wedge x \mapsto z * ls'(z, y))$

$\Rightarrow$

$ls(x, y) \coloneqq (x = y \wedge emp)$
$\quad \mid ls'(x, y)$
$ls'(x, y) \coloneqq (x \neq y \wedge x \mapsto y)$
$\quad \mid \exists z(x \neq y \wedge x \mapsto z * ls'(z, y))$

4. エンテイルメントを展開した定義に従って場合分け

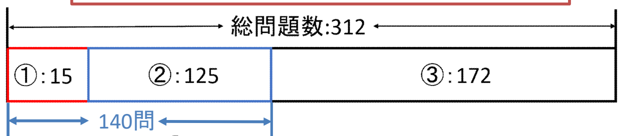
$ls(x, y) * ls(y, nil) \vdash ls(x, nil)$

$\Rightarrow$

$emp * emp \vdash emp \vee ls'(x, nil)$
$emp * ls'(x, nil) \vdash emp \vee ls'(x, nil)$
$ls'(x, nil) * emp \vdash emp \vee ls'(x, nil)$
$ls'(x, y) * ls'(y, nil) \vdash emp \vee ls'(x, nil)$

変換器の有効性

SL-COMP 2019 のベンチマーク(312問)を使って評価



①：元々cone条件を満たしていた
②：変換器によってcone条件を満たした
③: cone条件を満たすように変換できなかった

②の例

$DLL(w, x, y, z) \coloneqq (w = z \wedge x = y) \wedge (emp) \mid \exists u. (w \mapsto (x, u) * DLL(u, w, y, z))$

変換

$DLL(w, x, y, z) \coloneqq (w = z \wedge x = y) \wedge (emp)$
$\quad \mid DLL'(w, x, y, z)$
$DLL'(w, x, y, z) \coloneqq (w = y) \wedge (w \mapsto (x, z))$
$\quad \mid \exists u. (w \mapsto (x, u) * DLL'(u, w, y, z))$

Cycompの実験

今回の実験環境

- メモリ　8GB
- CPU Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz 3.60GHz

Cone条件を満たした問題140問をCycompに入力（180秒でtimeout）

180秒以内で解けたもの	失敗(Timeout　または stack_overflow)	誤答
24	110 （timeout:25 stack_overflow:84）	6 (Cycompのバグと考えられる)

考察

- 素朴な場合分けが多いため計算時間がかかったりスタックオーバーフローになる

課題

- SL-COMP 2019 のベンチマークでまだConeにできそうな述語が存在
- Cycompの修正と効率化