# 帰納的述語を含む分離論理による プログラム検証のためのループ不変式の導出

仲田 壮佑　　中澤 巧爾　（名古屋大学）

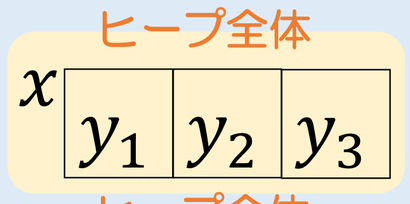
目的。一般の帰納的述語を含むプログラム検証におけるループ不変式の発見の自動化
手法。論理式を抽象化して事後条件を事前条件に一致させることでループ不変式を導出
成果。「畳み込み」「結合」によって抽象化を行い，それを用いたループ不変式の導出法を得た

## 分離論理　ホーア論理の拡張

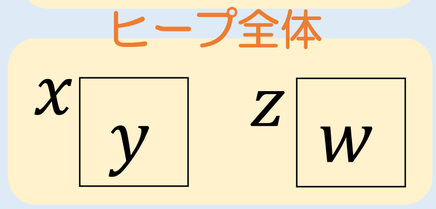
$\Pi ::= t = t \mid t \neq t \mid \Pi \wedge \Pi \mid \mathrm{true}$ 変数の条件

$\Sigma ::= t \mapsto \vec{t} \mid P(\vec{t}) \mid \Sigma * \Sigma \mid \mathrm{emp}$ ヒープの構造の条件

例　$x \mapsto (y_1, y_2, y_3)$




$x \mapsto y * z \mapsto w$




具体的な
{スタック $s$: Stacks
ヒープ　$h$: Heaps
による意味論

Value: $\mathbb{N}$
Locs: Vaule ∪ {nil}
Stacks: Vars → Value
Heaps: Locs $\rightharpoonup_{\mathrm{fin}}$ Value

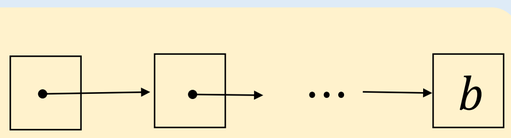
$s, h \vDash x \mapsto (y_1, \cdots, y_n)$
$\Leftrightarrow h = [s(x) \mapsto y_1, \cdots, s(x) + n - 1 \mapsto y_n]$
$s, h \vDash \Sigma_1 * \Sigma_2$
$\Leftrightarrow \exists h_1, h_2. h = h_1 + h_2,\ h_1 \cap h_2 = \emptyset,$
$s, h_1 \vDash \Sigma_1,\quad s, h_2 \vDash \Sigma_2$
$s, h \vDash \cdots$
$\vdots$

## 帰納的述語の例

### 例1　リスト

$\mathrm{ls}(x, y) \stackrel{\mathrm{def}}{=} x \neq y \wedge x \mapsto y$
$\mid x \neq y \wedge \exists z. x \mapsto z * \mathrm{ls}(z, y)$

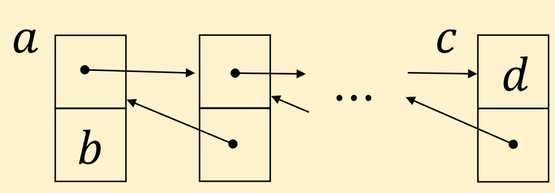
$\mathrm{ls}(a, b)$




### 例2 双方向リスト

$\mathrm{dll}(x, h, y, t) \stackrel{\mathrm{def}}{=} x = y \wedge x \mapsto (h, t)$
$\mid \exists z. x \mapsto (h, z) * \mathrm{dll}(z, x, y, t)$

$\mathrm{dll}(a, b, c, d)$




## 先行研究

[Distefano et al. 2006]
論理式を抽象化し
ループ不変式を導出

[Albarghouthi et al. 2015]
Interpolant を用いた方法

述語は ls のみ
総当り的

## ループ不変式の発見

$\{\mathrm{ls}(c, \mathrm{nil})\}$
$p \coloneqq \mathrm{nil};$
$\mathrm{while}(c \neq \mathrm{nil})($
$n \coloneqq [c];$
$[c] \coloneqq p;$
$p \coloneqq c;$
$c \coloneqq n;$
$)$
$\{\mathrm{ls}(p, \mathrm{nil}) \wedge \cdots\}$

事前条件 ＋ コマンド
事後条件

「ループ本体の事後条件が事前条件に一致」が必要

$\{\mathrm{ls}(p, \mathrm{nil}) * \mathrm{ls}(c, \mathrm{nil})\}$
$n \coloneqq [c];\ [c] \coloneqq p;\ p \coloneqq c;\ c \coloneqq n;$
$\{p \mapsto p' * \mathrm{ls}(p', \mathrm{nil}) * \mathrm{ls}(c, \mathrm{nil})\} \Rightarrow \{\mathrm{ls}(p, \mathrm{nil}) * \mathrm{ls}(c, \mathrm{nil})\}$

適切な抽象化「畳み込み」「結合」をする

## 畳み込み

帰納的述語定義に従って定義の１つを含むものをその述語に置き換える

$a \neq b \wedge a \mapsto c' * \mathrm{ls}(c', b)$

畳み込み

$\mathrm{ls}(a, b)$

## 結合

複数の述語を１つにまとめる

$\mathrm{ls}(x, y) := x \neq y \wedge x \mapsto y \mid \cdots$

$\mathrm{ls}(a, b') * \mathrm{ls}(b', \mathrm{nil})$

述語定義の一つに置き換えて畳み込みができないか試す

$a \neq b' \wedge a \mapsto b' * \mathrm{ls}(b', \mathrm{nil})$

畳み込み

$\mathrm{ls}(a, \mathrm{nil})$

結合

$\mathrm{ls}(a, b') * \mathrm{ls}(b', \mathrm{nil}) \vDash \mathrm{ls}(a, \mathrm{nil})$ をチェック
証明器[仲田ら 2016 JSSST]を利用

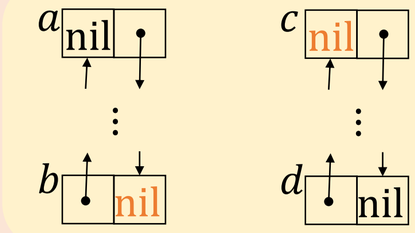
## 実装による実行例

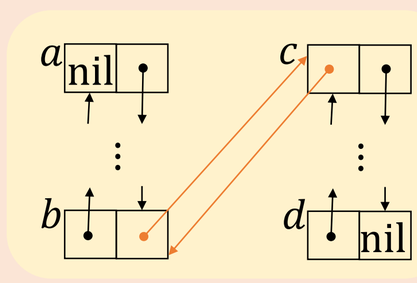
入力：帰納的述語定義／事前条件 $\Pi; \Sigma$／コマンド $c$
出力：事後条件 $\Pi'; \Sigma'$

帰納的述語定義： dll （再掲）

$\mathrm{dll}(x, h, y, t) := x = y \wedge x \mapsto (h, t)$
$\mid \exists z. x \mapsto (h, z) * \mathrm{dll}(z, x, y, t)$

$\{\mathrm{dll}(a, \mathrm{nil}, b, \mathrm{nil}) * \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil})\}$
$t \coloneqq a;$
$n \coloneqq [a].\mathrm{right};$
$\mathrm{while}(n \neq \mathrm{nil})($
$t \coloneqq n;$
$n \coloneqq [n].\mathrm{right};$
$)$
$[t].\mathrm{right} \coloneqq c;$
$[c].\mathrm{left} \coloneqq t;$




ループ不変式

$n = \mathrm{nil} \wedge t = b \wedge a \neq \mathrm{nil} \wedge t' \neq \mathrm{nil} \wedge t \neq \mathrm{nil} \wedge \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil}) * t \mapsto (t', \mathrm{nil}) * \mathrm{dll}(a, \mathrm{nil}, t', t),$
$a \neq \mathrm{nil} \wedge t' \neq \mathrm{nil} \wedge t \neq \mathrm{nil} \wedge \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil}) * t \mapsto (t', n) * \mathrm{dll}(n, t, b, \mathrm{nil}) * \mathrm{dll}(a, \mathrm{nil}, t', t),$
$n = \mathrm{nil} \wedge t = a \wedge a = b \wedge \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil}) * a \mapsto (\mathrm{nil}, \mathrm{nil}),$
$n = z'_0 \wedge t = a \wedge \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil}) * a \mapsto (\mathrm{nil}, z'_0) * \mathrm{dll}(z'_0, a, b, \mathrm{nil}),$
$n = \mathrm{nil} \wedge t = b \wedge t \neq \mathrm{nil} \wedge \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil}) * a \mapsto (\mathrm{nil}, t) * t \mapsto (a, \mathrm{nil}),$
$t \neq \mathrm{nil} \wedge \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil}) * a \mapsto (\mathrm{nil}, t) * t \mapsto (a, n) * \mathrm{dll}(n, t, b, \mathrm{nil}),$
$n = \mathrm{nil} \wedge t = b \wedge t' \neq \mathrm{nil} \wedge t \neq \mathrm{nil} \wedge \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil}) * t \mapsto (t', \mathrm{nil}) * \mathrm{dll}(a, \mathrm{nil}, t', t),$
$t' \neq \mathrm{nil} \wedge t \neq \mathrm{nil} \wedge \mathrm{dll}(c, \mathrm{nil}, d, \mathrm{nil}) * t \mapsto (t', n) * \mathrm{dll}(n, t, b, \mathrm{nil}) * \mathrm{dll}(a, \mathrm{nil}, t', t)$

$\{ n = \mathrm{nil} \wedge t = b \wedge a \neq \mathrm{nil} \wedge t' \neq \mathrm{nil} \wedge t \neq \mathrm{nil} \wedge \mathrm{dll}(a, \mathrm{nil}, t', b) * b \mapsto (t', c) * c \mapsto (b, z') * \mathrm{dll}(z', c, d, \mathrm{nil}) \}$ ∨（他3式）

ループ不変式の導出を自動化