ソフトウェア論講座
坂部研究室

坂部 俊樹　西田 直樹　濱口 毅　橋本 健二

INFORMATION SCIENCE NAGOYA UNIVERSITY 情報システム学専攻

研究概要

本研究室では「高品質のソフトウェア」を効率良く作成するための技法を確立することを目標として研究を行っている．「高品質のソフトウェア」の条件としては，正しいこと，速いこと，メモリ使用量が少ないことはもちろんであるが，構造が整っていていること，再利用に適していること，仕様などのドキュメントがしっかり揃っていることなど，ソフトウェア工学的な条件もあげられる．これらの条件のうち，特に「正しい」という条件に注目し，プログラムが正しい動作をすることを検証する方法，正しいプログラムしかできないようなプログラム構成法などについて研究を行っている．

共同研究者

（2013 年 12 月現在）

- 酒井 正彦　（本研究科計算機数理科学専攻・教授）
- 草刈 圭一朗　（本研究科計算機数理科学専攻・准教授）
- 関 浩之　（本研究科情報システム学専攻・教授）
- 石原 靖哲　（大阪大学・准教授）
- 廣川 直　（北陸先端科学技術大学院大学・准教授）
- Germán Vidal　（バレンシア工科大学・教授）
- Aart Middeldorp　（インスブルック大学・教授）
- Sarah Winkler　（インスブルック大学・ポスドク研究員）
- Cynthia Kop　（インスブルック大学・ポスドク研究員）
- Bernhard Gramlich　（ウィーン工科大学・准教授）
- Karl Gmeiner　（ウィーン工科大学・研究員）
- Stephan Falke　（カールスルーエ工科大学・ポスドク研究員）

主な研究テーマ

- 制約付き項書換え系によるプログラムのモデル化
- 帰納的定理の補題生成機能付き証明器の開発
- ポインタを操作するプログラムの効率的動的検査系
- プログラム逆化手法の開発
- 例外処理を含む関数型プログラムの停止性証明
- 木構造データ変換の問合せ保存性検証
- SMT ソルバーの開発

研究の背景：制約付き項書換え系の研究

近年の計算機の性能向上により論理式の充足可能性を判定する SMT ソルバーの開発およびその利用が盛んになっている．それに伴い，項書換え系に制約系を導入した制約付き項書換え系の研究が盛んになり，様々な体系が提案されその応用が研究されている．

本研究室では近年の制約付き項書換え系の研究の先駆けとして，制約付き書換えの研究で世界をリードしている．

制約付き項書換え系の特徴

- 通常の関数記号とは別に解釈を持つ関数記号と述語記号がある
  - 解釈には既存の SMT ソルバーを利用
  - 以下では解釈を持つ記号を緑で表示
- ユーザ定義の関数のみを書換え規則として表現

CRISYS プロジェクト：C プログラムの int 型上の数値計算関数の正しさの自動検証

```
int sum1(int n){
  int i=0, z=0;
  while( i<=n ){
    z += i;
    i++;
  }
}
```

等価変換 $\Longrightarrow \mathcal{R}_{\mathsf{sum1}} = \left\{ \begin{array}{l} \mathsf{sum1}(n) \to \mathsf{u}(n, \mathsf{s}(0), 0) \\ \mathsf{u}(n, i, z) \to \mathsf{u}(n, \mathsf{s}(i), z + i) \ [\![i \leq n]\!] \\ \mathsf{u}(n, i, z) \to z \ [\![\neg\, i \leq n]\!] \end{array} \right\} \longrightarrow$

仕様

$$\mathsf{sum}(x) = 0 \quad \text{if } x \leq 0$$
$$\mathsf{sum}(x+1) = \mathsf{sum}(x) + (x+1) \quad \text{if } x \geq 0$$

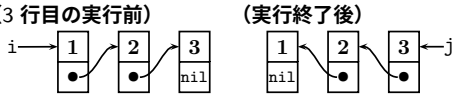
等価変換 $\Longrightarrow \mathcal{R}_{\mathsf{sum}} = \left\{ \begin{array}{l} \mathsf{sum}(x) \to 0 \ [\![x \leq 0]\!] \\ \mathsf{sum}(\mathsf{s}(x)) \to \mathsf{sum}(x) + \mathsf{s}(x) \ [\![x \geq 0]\!] \end{array} \right\} \longrightarrow$

証明したい等式 $\mathsf{sum}(n) = \mathsf{sum1}(n) \longrightarrow$

書換え帰納法 $\longrightarrow$ 成功/失敗

- $\mathsf{sum}(n) = \mathsf{sum1}(n)$ が制約付き項書換え系 $\mathcal{R}_{\mathsf{sum1}} \cup \mathcal{R}_{\mathsf{sum}}$ の帰納的定理ならば，sum1 は仕様を満たす
  - 制約付き等式 $s = t \ [\![\phi]\!]$ は R の帰納的定理：ϕ を真にする任意の基底代入 σ について，$\mathcal{R}$ による $s\sigma$ と $t\sigma$ の計算結果が一致（$s\sigma \leftrightarrow_R^* t\sigma$）
- 制約付き項書換え系の利点：比較演算に起因した証明の発散を回避
- 証明プロセス中に等式生成が発散するが，補題等式 $\mathsf{u}(n, x, y) + \mathsf{s}(n) = \mathsf{u}(\mathsf{s}(n), \mathsf{s}(x), y + x) \ [\![x \leq \mathsf{s}(n)]\!]$ を自動生成し証明に成功

ポインタを操作するプログラムの効率的動的検査系



その場のリスト逆転プログラム

```
1 i:=cons(1,0,2,0,3,0);[i+1]:=i+2;[i+3]:=i+4;
2 {list(α0, i) ∧ α0 = 1 · 2 · 3}
3 j:=nil;
4 while not i=nil do
5    {∃α. ∃β. list(α, i) * list(β, j) ∧ rev(α0) = append(rev(α), β)}
6    k:=[i+1]; [i+1]:=j; j:=i; i:=k od
7 {list(rev(α0), j)}
```

述語 list，関数 rev，append を定義する制約付き項書換え系

$$\left\{ \begin{array}{r} \mathsf{list}(\varepsilon, i) \to \mathsf{emp} \ \wedge \ i = \mathsf{nil} \\ \mathsf{list}(a \cdot \alpha, i) \to (i \mapsto a, \_) * \mathsf{list}(\alpha, [i{+}1]) \\ \mathsf{rev}(\varepsilon) \to \varepsilon \\ \mathsf{rev}(a \cdot \alpha) \to \mathsf{append}(\mathsf{rev}(\alpha), a \cdot \varepsilon) \\ \mathsf{append}(\varepsilon, \alpha) \to \alpha \\ \mathsf{append}(a \cdot \alpha, \beta) \to a \cdot \mathsf{append}(\alpha, \beta) \end{array} \right\}$$

↓

動的検査系

↓

実行中の注釈（論理式）の真偽

（2 : true, 5 : true, 5 : true, 5 : true, 7 : true）