神奈川大学理学部　情報科学科　馬谷研究室

定理証明器 J-bob における構造体向け構造的帰納法

小長谷 俊介 201703399

1　はじめに

Scheme で使用できる定理証明器の一種である J-bob は「The Little Prover」[1] という書籍で用いられている証明器であり，リスト型と木構造に対する帰納法を定理証明できる．

しかし，それ以外の Scheme のデータ型には使用できないため，定理証明器として使用できる範囲が限られている．

そこで本研究では，Racket 言語 (Scheme の一種) が提供する構造体型に対する構造的帰納法の機能を J-Bob に追加する．

2　背景

J-Bob における証明の例として，リストデータに対する帰納法の証明を図 1 に示す．

この例は，1–4 行目で `memb?/remb` という名前の定理を定義し，5 行目以降はその証明をしている．

`memb?/remb0` は，空リストから remb 関数でシンボル'?を取り除いたら，そのリストにはシンボル'?は含まれていないという定理である．

`memb?`は受け取ったリストにシンボル'?があるか調べる関数であり，`remb` は受け取ったリストからシンボル'?を取り除いたリストを作成する関数である．

証明を行っている部分は，式を置き換えるステップ毎に，1 つ目の括弧が書き換える場所を示し，2 つ目の括弧で書き換える内容を示している．

書き換え場所の示し方は，数字が S 式の何番目の要素であるか，QAE がそれぞれ if 式の Question 部，Answer 部，Else 部にあるかを表すものである．

停止性の証明の付いていない関数定義は J-Bob に受理されないため，図 1 のように `memb?`や `remb`

```
(J-Bob/define (defun.remb)
  '(((dethm memb?/remb0 ()
       (equal (memb? (remb '())) 'nil))
     nil
     ((1 1) (remb '()))
     ((1 1 Q) (atom '()))
     ((1 1)
      (if-true '()
        (if (equal (car '()) '?)
         (remb (cdr '())) (cons (car '())
            (remb (cdr '()))))))
     ((1) (memb? '()))
     ((1 Q) (atom '()))
     ((1) (if-true 'nil
     (if (equal (car '()) '?)
       't (memb? (cdr '())))))
     (() (equal-same 'nil))))))
```

図 1　J-Bob による証明の例

のような関数を J-Bob で使用したい場合，別途それぞれに停止性の証明を付ける必要がある．

3　設計

構造的帰納法の機能を J-Bob に追加するにあたって，J-Bob での定理証明にて構造体を利用可能にする事，構造体に関連する関数を J-Bob 内にある既存の組み込み関数リストに追加する事が必須である．これは J-Bob に組み込み関数であると扱わせる事で J-Bob 内では全域関数として利用できるようになるからである．

また，本研究では Racket という言語を利用するが，Racket での構造体に関連する関数は構造

体の名前が関数名に含まれるようになっている．(struct posn (x y)) と定義した構造体を例として挙げると，x に格納されている値を取り出す関数である posn-x 等がある．

このように，作成する構造体によって組み込み関数リストに追加する関数名や個数が変わってくるので，マクロ機能を利用してどのような構造体にも対応できるようにする事が必須である．

こうして J-Bob で構造体，それに関連する関数を利用可能にしてから，構造的帰納法の実装を行う．

帰納法の具体的な手順は構造体によって内容が変わるため，マクロ機能を利用してどのような構造体にも対応できるようにする事が必須である．

4 実装

構造的帰納法の機能を J-bob に追加するにあたって，Racket 言語が提供する構造体を J-bob で使用可能にし，構造体に関する関数を J-bob が内部で管理している組み込み関数のリストに追加するという工程を行う．

本章では実際に作成した構造体に関する関数を J-bob が内部で管理している組み込み関数のリストに追加するマクロのコードを図 2 に示す．

bfunc は組み込み関数のリスト，ubfunc はその中で 1 引数の関数が入っているリスト，bbfunc は 2 引数の関数が入っているリストである．

変更前は組み込み関数のリストが無く，組み込み関数を参照する部分では if 文を利用しそれぞれの組み込み関数と等しいか確認していくという形であった．しかしながら，その場合には新しい関数を組み込み関数として扱うには手間がかかる状態であるため，リストにすることで扱いやすさとコードの理解しやすさを向上させる．

bfunc-add ではマクロ機能を用いて構造体に関連する関数を組み込み関数のリストに入れるという処理を行っている．

```
(define bfunc (box (cons
            '(atom car cdr natp size)
            '(equal cons + <))))

(define ubfunc (box
               (car (unbox bfunc))))

(define bbfunc (box
                (cdr (unbox bfunc))))

(define-syntax (bfunc-add stx)
  (syntax-parse stx
    ((_ stid (fiid ...))
     #:with stid? #`#,(string->symbol
        (string-append (symbol->string
          (syntax->datum #'stid)) "?"))
     #:with stid-fiid (map
  (lambda (fiid) (con-stfi #'stid fiid))
            (syntax->list #'(fiid ...)))
     #`(set-box! bfunc (cons
         (append '(stid? #,@#'stid-fiid)
                 (car (unbox bfunc)))
                 (cdr (unbox bfunc)))))))
```

図 2 J-bob への組み込み関数の登録マクロ

5 まとめ

本研究では，Scheme で使用出来る定理証明器の一種である J-Bob にて Racket 言語が提供する構造体型に対する構造的帰納法のための機能の追加を行い，その結果，帰納法の機能を J-Bob での定理証明を行う時に利用する事が出来るようになった．

この研究によって，J-Bob を定理証明に利用できる状況が増えると期待できる．

本研究の今後の課題としては，より J-Bob の利用可能な状況を増やすためにその他のデータ型に対する帰納法についても本研究と同様に追加していくという事が挙げられる．

参考文献

[1] Daniel P. Friedman and Carl Eastlund, The Little Prover, The MIT Press, 2015.