平成１４年度　情報工学コース卒業研究報告要旨

坂部 研究室	氏　　名	森　　田　　洋　　介
卒業研究題目	Proof-carring code 法のコード記述言語の拡張	

Proof-carring code(PCC) はあるプログラムを配布する際に、プログラムを提供する側はプログラムにそのプログラムが正しいかどうかの証明を添付し、プログラムを使用する側はその証明が正しいかどうかをチェックしてから実行するという方法である。プログラムの正しさを検証するのはアセンブリ言語又は機械語レベルで行うので、プログラムを使用する側は自分の実行環境において、実行する前にプログラムの正しさを検証できる。そのため、例えばプログラムをインターネット上から取得する際に改竄されていたり、ウイルスが紛れ込んでいたりしても実行する前に検証できるので防ぐことができる。

PCC ではアセンブリ言語で書かれたプログラムを、証明可能なコードに変更する規則が必要がある。また、そのコードを証明するための推論規則も必要であり、推論規則を作るためには対象とするプログラムで使われている型が実行環境においてどのように表現されているかを理解しなければならない。

本研究では、PCC によって証明できるデータ構造の種類を増やすためにアセンブリコードの命令を拡張することを目的とする。新しく追加したデータ構造は、配列とスタックである。配列は SUB という配列の n 番目の要素にアクセスできる命令を追加し、スタックは POP というスタックの一番上の要素をを捨てる命令、TOP というスタックの一番上の要素にアクセスできる命令、PUSH というスタックの一番上に要素を追加する命令を追加した。それらの命令を PCC で扱えるように、アセンブリ言語のプログラムを証明できるコードに直す規則、そのコードを証明するための推論規則を再定義した。例えば、任意個の int から配列 int_array を生成する推論規則は次のようになる。

$$\frac{m \vdash e_1 : \tau\_array \ \wedge m \vdash e_2 : int}{\begin{array}{l} m \vdash e_1 : addr \ \wedge \ m \vdash e_1 \oplus 1 : addr \ \wedge \cdots \wedge m \vdash e_1 \oplus (e_2 - 2) : addr \\ \qquad \wedge m \vdash e_1 \oplus (e_2 - 1) : addr \wedge \ sel(m, e) : \tau \ \wedge sel(m, e \oplus 1) : \tau \\ \qquad \wedge \cdots \wedge sel(m, e \oplus (e_2 - 2)) : \tau \ \wedge sel(m, e \oplus (e_2 - 1)) : \tau \end{array}}$$

これは、e_2 個の変数が $addr$(読み書き可能) であり、かつ τ 型であるときにそれらを要素に持つ配列 τ_array と配列の長さを持つ int 型の整数になるという規則である。これらの推論規則を用いて、int 型の要素を持つ配列・スタックを与えたらその要素の和を返すプログラムの型に関する正当性を証明できた。

追加した推論規則の正当性はまだ証明できていない。また、データ構造は配列・スタック以外にも沢山あり、それらに対する命令も考えなければならない。本研究では各データ構造の要素は int 型に限定している。他にも string 型や char 型などに関しても考えなければならない。これらは今後の課題とする。