平成１４年度　情報工学専攻修士論文要旨

坂部 研究室	氏　　名	吉仲 晃一
論 文 題 目	Join 算法と JAVA の融合言語 JoinJAVA における 同期通信機能に関する研究	

近年必要性が増してきているマルチスレッドプログラミングのための言語として、JoinJAVA 言語が提案されている。これは、並行プロセスを記述するのに適した計算モデルである Join 算法の表記法を利用したものであり、同期処理や通信処理の記述に Join 算法の記述を、それ以外の記述に JAVA による記述を行うようにして Join 算法と JAVA 言語を融合させた言語である。この JoinJAVA 言語を実装し、動作させる目的で JoinJAVA コンパイラが作成されているが、このコンパイラが扱うのは非同期チャネルのみであり、本来 Join 算法や JoinJAVA 言語において定義されている同期チャネルについては組み込まれていない。同期チャネルには非同期チャネルにはない、値を返すという働きがあるため、同期チャネルを用いることで関数呼び出しや実行結果の返却などの処理を容易に記述することができるようになる。また、同期チャネルを含む Join 算法の記述を非同期チャネルのみに変換するマクロ展開が提案されているが、これはいまだ実装されていない。またマクロ展開によって値を返すためのチャネルが新たに作られ、それによって内部通信が行われるため通信量が増加し効率が良くない。

本論文では、JoinJAVA コンパイラで同期チャネルを扱えるように拡張する。その際、JoinJAVA コンパイラのプリプロセッサとしてマクロ展開を実装する。また、マクロ展開によって新しいチャネルが作成されないようにして効率を向上させるために、新たな変換規則を提案しこの規則が正しいことを証明する。その後、変換規則の実装を行い、その評価を行う。

変換規則は、典型的な形をした同期チャネルを含むプロセスにおいて、同期チャネルを含むプロセスへ変換する規則である。具体的には `let` 構文によるチャネル呼び出しの記述 P_A を、関数呼び出し後の記述 P_B へと変換する。証明の際には、P_A と P_B をそれぞれマクロ展開によって非同期チャネルのみのプロセス P_{MA}、P_{MB} へと変換し、P_{MA} の方を動作意味論によって内部通信の２ステップを動作させて P'_{MA} へと到達させる。その後、P'_{MA} と P_{MB} が bisimilarity equivalence であることを示す。

提案した変換規則をプリプロセッサの前処理として実装しても問題ないと考えられるため、この規則の実装・評価を行った。評価手法として、変換規則を適用する場合としない場合で JoinJAVA コンパイラから吐き出される JAVA プログラムの実行時間の測定を行った。その結果、変換規則によって時間が大幅に短縮されていることが分かった。