Erlangからの高位合成のための関数レベル並列化

Function Level Parallelization for High-Level Synthesis from Erlang

若林 秀和 [1] 石浦 菜岐佐 [1] 吉田 信明 [2] 神原 弘之 [2]
Hidekazu Wakabayashi Nagisa Ishiura Nobuaki Yoshida Hiroyuki Kanbara

関西学院大学 理工学部 [1] School of Science and Technology, Kwansei Gakuin University
京都高度技術研究所 [2] ASTEM RI/KYOTO

1 はじめに

近年, 益々高機能化する組込みシステムの仕様をいかに記述し, そこからハードウェアをいかに効率的に設計するかが重要な課題となっている. これに対する一アプローチとして, 文献 [1] では Erlang から高位合成を行う手法を提案している. この方式では, プロセスレベルでの並列化および命令レベルでの並列化は実現されるが, 1 プロセス内の関数は逐次的にしか実行できない. 本手法では, プログラムの変換によって関数レベルの並列化を実現する方法を提案する.

2 Erlang からの高位合成

Erlang は, 並行処理指向の関数型言語であり, 並行に動作する複数のプロセスにより対象システムの動作を記述する. プロセス間のデータ共有は非同期のメッセージ通信により行う. Erlang プログラムはコンパイラによって仮想機械 BEAM のバイトコードに変換され, インタープリタにより実行される. 文献 [1] は, BEAM コードから CDFG を生成することにより高位合成を行っている.

図 1 (a) のコード例では, 2–3 行目で呼び出し元のプロセス ID (`Pid`) と 5 つの入力値をメッセージとして受け取り, 7 行目で計算結果 `Res` を返送している. 4–5 行目の 2 つの `mex` 間に依存関係はないが, コンパイラが生成する BEAM コードは逐次的であるため, ここから生成されるハードウェアでも `mex` は並列に実行されない.

3 プログラムの変換による関数レベル並列化

本稿では, プログラムレベルでの変換によって可能な関数の実行を並列化する手法を提案する. これは関数の評価をプロセスで実行し, 引数や返り値の授受をメッセージ通信で行うことにより実現する. 例えば図 1 (a) は (b) のよう変換する. (b) の 4–5 行目の `p_mex` と `q_mex` は `mex` を評価するプロセスであり, `p_crt` は `crt` を評価するプロセスである. 入力をメッセージとして受け取り, 必要な入力が揃ったら `mex` を計算する. ここで `mex` の実行結果は直接 `p_crt` に送られる. `p_crt` は `p_mex` と `q_mex` からの結果および `decrypt` のからのデータ全てが揃った時点で計算を開始し, 結果は直接呼び出し元のプロセスに返す.

`mex` を評価するプロセスは (c) のように書ける. 新着メッセージを受信した際, それが `c` のデータであって, `c` が未受信ならばそれを `C` に受信する (2–3 行目). 必要な入力が全て受信済みであれば `mex` を計算し, 結果 (`MP`) を `p_crt` に送信する (4–6 行目). 必要な入力が揃っていなければ, 7 行目で `C` を受信済みとして次のメッセージを待つ.

(b) の `decrypt` で複数のデータを処理する際には, `p_crt` が i 番目のデータを処理するのと並列に `p_mex` と `q_mex` が i+1 番目のデータを処理するというパイプライン的な実行も可能となる.

以上の変換は Erlang 又は BEAM のレベルで行う.

表 1 実行時間の比較

	(実行回数)	逐次処理 [s]	本手法 [s]	(プロセス数)
RSA	(10 回)	7.3	6.0	(3)
MMM	(1000 回)	5.8	5.0	(3)
Matrix	(10000 回)	2.1	1.4	(4)

(Intel Xeon E3-1276 3.60GHz(4 コア) 32 GiB メモリ)

4 実験結果

本変換手法の並列化効果を確認するため, マルチコア CPU 上で実行時間を計測する実験を行った. 結果を表 1 に示す. RSA は RSA の復号, MMM はモンゴメリ乗算, Matrix は行列の乗算である. 本手法により実行時間が短縮できることがわかる. プロセス数に比例した高速化が達成されないのは, メッセージ通信や各種並列化のオーバーヘッドのためと考えられる.

5 むすび

本稿では, Erlang からの高位合成のための関数レベル並列化手法を提案した. 本手法に基づく変換系の実装が今後の課題である.

謝辞 本研究は一部 JSPS 科研費 16K00088 および 16K01207 の助成による.

参考文献

[1] H. Takebayashi, N. Ishiura, K. Azuma, N. Yoshida, and H. Kanbara: "High-Level Synthesis of Embedded Systems Controller from Erlang," in *Proc. SASIMI 2016*, pp. 285–290 (Oct. 2016).

```
1 decrypt() ->
2  receive
3    {Pid,{C,P,Q,DP,DQ}} ->
4      MP = mex(C,DP,P),
5      MQ = mex(C,DQ,Q),
6      Res = crt(MP,MQ,P,Q),
7      Pid ! Res,
8      decrypt()
9  end.
```

(a) 逐次処理

```
1 decrypt() ->
2  receive
3   {Pid,{C,P,Q,DP,DQ}} ->
4    p_mex ! {c,C}, p_mex ! {dp,DP}, p_mex ! {p,P},
5    q_mex ! {c,C}, q_mex ! {dq,DQ}, q_mex ! {q,Q},
6    p_crt ! {p,P}, p_crt ! {q,Q}, p_crt ! {pid,Pid},
7    decrypt()
8  end.
```

(b) 並列化

```
 1 p_mex(C0,DP0,P0) ->
 2   receive
 3     {c,C} when C0 is empty ->
 4      if DP0 and P0 exist ->
 5        MP = mex(C,DP0,P0), p_crt ! {mp,MP},
 6        p_mex([],[],[]);
 7      true -> p_mex(C,DP0,P0)
 8      end;
 9     {dp, DP} when DP0 is empty ->
10       ...
11   end.
```

(c) 関数を評価するプロセス

図 1 プログラムの変換例