

統合型並列化コンパイラ・システム

- 概要 -

1 G - 1

村上和彰 音成 幹 福田 晃 富田眞治
(九州大学)

1. はじめに

我々は現在、以下の汎用並列処理システムを開発している。

- ① 可変構造型並列計算機^[1]: 128台のSPARCマイクロプロセッサを128×128クロスバー網で相互結合する、ダイナミック・アーキテクチャを採用した問題適応型の粗粒度レベル並列処理マシン。
- ② 『新風』^[2]: SIMP(単一命令流/多重命令パイプライン)方式に基づく試作機で、4本の均質な命令パイプラインにより命令レベル並列処理を行うスーパースカラ・プロセッサ。
- ③ 『順風』^[3]: ストリームFIFO方式に基づく試作機で、複数本の演算パイプラインによりループ・レベル並列処理を行うベクトル・プロセッサ(さらに、演算パイプラインを複数命令で共有することで、ベクトル要素レベルの並列処理も可能)。

また、これらハードウェア・システムの開発と並行して、並列/分散OS^[4]、並列プログラミング言語“SERVE”^[5]、および、並列プログラミング環境の開発も進めている。

本稿では、その並列プログラミング環境の1つとして、上記ハードウェア・システムに対する統合的なコンパイル環境を提供する統合型並列化コンパイラ(IPC: Integrated Parallelizing Compiler)システムについて、その概要を述べる。

図1に、本システムの構成と処理を示す。統合型並列化コンパイラの“統合型”とは、本コンパイラに対する以下の3つの“統合的”な要件に起因する。

2. Multilingualism

逐次/並列言語を問わず、複数のソース言語をコンパイル可能とする。この要件は、単一プロセッサ用最適化コンパイラでは、フロントエンドをソース言語対応に設け中間言語以下を共用することで、すでに実現している。

しかし、既存の並列化コンパイラでは、その並列性抽出処理がソース言語にかなり依存するため、一般にはソース言語に個別対応となっている。また、source-to-source restructureではプログラム再構成処理以降は共用可能だが、restructure自身はやはりソース言語に個別対応である。

本コンパイラでもプログラム再構成を行うが、そのrestructureは各種ソース言語間で共用可能となるように設計する。そのために、ソース言語依存のフロントエンドからの出力、および、restructureへの入力には、ソースコードの有する明示/暗黙的な並列性を継承可能な中間言語(例えば、プログラムグラフ表現)を用いる。よって、本restructureは、intermediate-to-intermediate restructureとなる。

3. Retargetability

複数の並列処理マシンをオブジェクトコード生成のターゲットとする。これには、現在我々が開発中である前記の3ハードウェア・システム、および、将来開発するであろうハードウェア・システムを含む。

この要件を満足するのは一般に困難であり、また、セルフ・コンパイル環境ではその必要性がないことから、実現例は少ない。

本コンパイラでは、その構成モジュールをretargetabilityに関して次の3種類に分類した上で、全体をモジュール構成する。

- ① ターゲット非依存(TI: Target Independent)モジュール:

ターゲットの属性にまったく依存しない。

- ② パラメータ化ターゲット依存(TDP: Target Dependent/Parameterized)モジュール: ターゲットの属性に依存するものの、当該属性がパラメータ化可能である。

- ③ ターゲット依存(TD: Target Dependent)モジュール: ターゲットの属性に依存し、かつ、当該属性のパラメータ化が困難である。

上記①②のモジュールは各種ターゲットマシン間で共用可能である。一方、③のモジュールは、ターゲットマシンに個別対応のバックエンドとする。

なお、TDPモジュールに対するパラメータとしては、以下のものを設ける。

- i) System Parameters: ターゲットのシステム要因(シングル/マルチ、スカラ/ベクトル、など)。
- ii) Multiprocessor Parameters: マルチプロセッサ要因(プロセッサ数、メモリ構成、相互結合網構成、など)。
- iii) Interconnection Network Parameters: ターゲットの相互結合網要因(トポロジー、ポート構成、バンド巾、など)。
- iv) ISP Parameters: ターゲットの命令セット・アーキテクチャ要因(中間コード-オブジェクトコード対応関係、レジスタ構成、など)。

4. Multilevel Parallelization

プログラムに明示/暗黙的に内在する並列性を、粗粒度レベルから細粒度レベルに至る以下のあらゆるレベルで抽出し活用する。

- ① プロセス・レベル: 並列プログラミング言語を用いてユーザが明示的に指定した並列性。

- ② ブロック・レベル: 基本ブロック、および、その集まりであるマクロブロック間の並列性。

上記①②の並列処理単位(慣例によりタスクと呼ぶ)は、partitioning & scheduling フェーズにおいてプログラム分割により抽出し、タスク・スケジューリングによりその並列性を活かす。

- ③ ループ・レベル: FORTRANのDOループなどに内在する並列性。restructuring フェーズにおいて、ベクトル化、および並行化(doall, doacross, など)することで活かす。

- ④ 命令レベル: 機械命令間の並列性。prepass/postpass optimization フェーズにおける広域コード移動、および、局所コード・スケジューリングにより活用する。

5. おわりに

以上、統合型並列化コンパイラ・システムの開発思想を中心に、その構成および処理の概要を述べた。現在、ネットワーク・シンセシスとコード・スケジューリングを主体に開発を進めている。これらについては、後続発表論文を参照されたい。

参考文献

- [1] 森ほか: 可変構造型並列計算機のPE間メッセージ通信機構, 情処論文誌, vol.30, no.12 (1989年12月)。
- [2] 久我ほか: SIMP方式に基づく『新風』プロセッサの低レベル並列処理アルゴリズム, 情処論文誌, vol.30, no.12 (1989年12月)。
- [3] 弘中ほか: ストリームFIFO方式に基づくベクトル・プロセッサ『順風』, 信学技報, CPSY89-39 (1989年8月)。
- [4] 福田ほか: 可変構造型並列計算機の並列/分散OS, 情処研報, 89-OS-43-8 (1989年6月)。
- [5] 廣谷ほか: 並列プログラミング言語SERVEの設計, 情処39全大論文集, 2Q-1 (1989年10月)。

An Integrated Parallelizing Compiler System :
Overview

Kazuaki MURAKAMI, Miki OTONARI, Akira FUKUDA
and Shinji TOMITA
Kyushu University

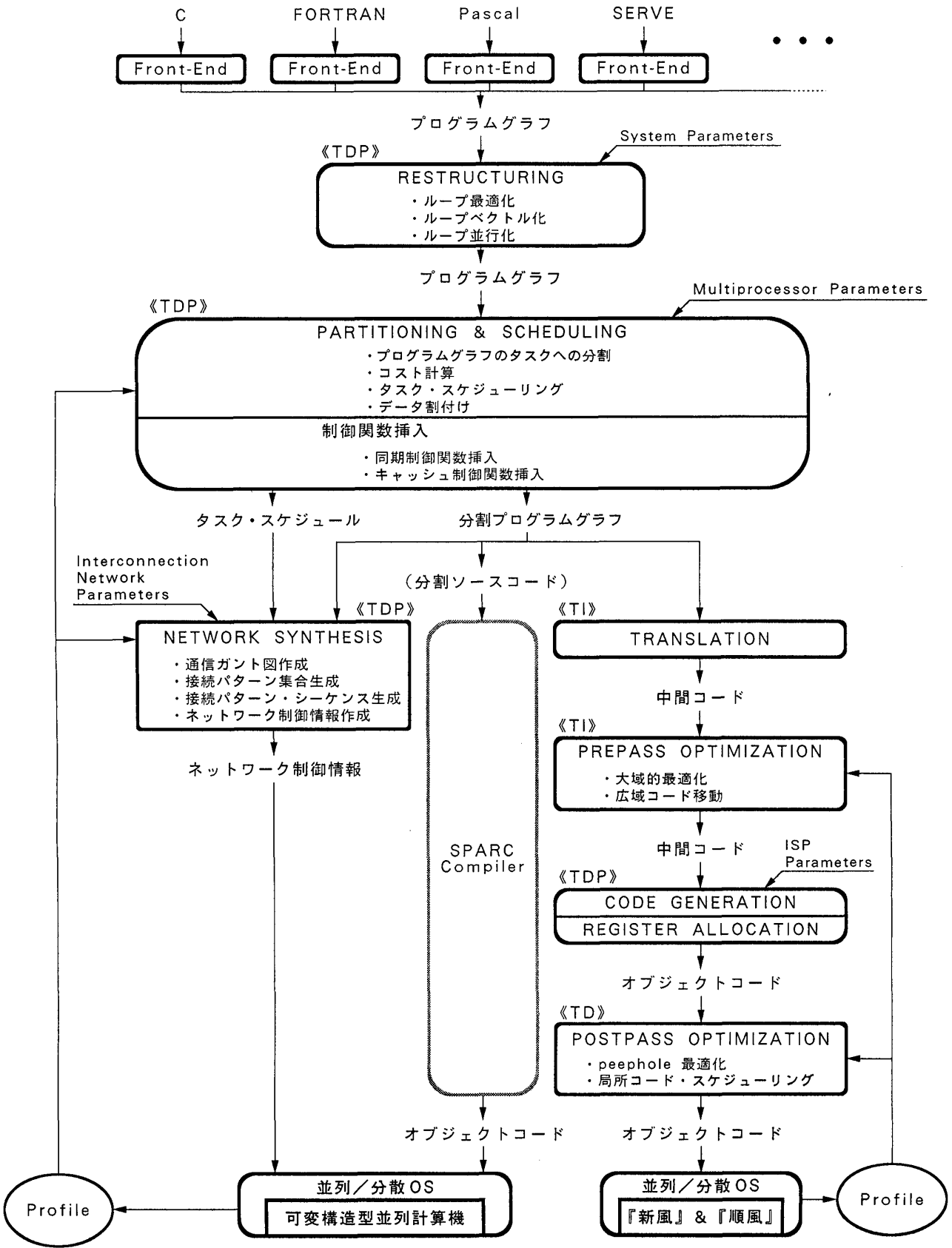


図1. 統合型並列化コンパイラ・システムの構成とその処理