2J-3 類似事例を用いた並列プログラミング支援

古川 知之　　山崎 勝弘
立命館大学理工学部

1 はじめに

並列プログラミングでは, 負荷均衡の実現とプロセッサ間の情報授受のオーバーヘッドの軽減が最も重要である. そのためにはプロセス間の同期, 相互排除, スレッドなどに習熟している必要があり, 並列プログラミングは非常に困難である. 本稿では, 過去の代表的な並列プログラムの中から最も類似した並列プログラムを検索し, それを再利用することによって並列プログラミングの負担を軽減させる方法について述べる.

2 並列プログラミング

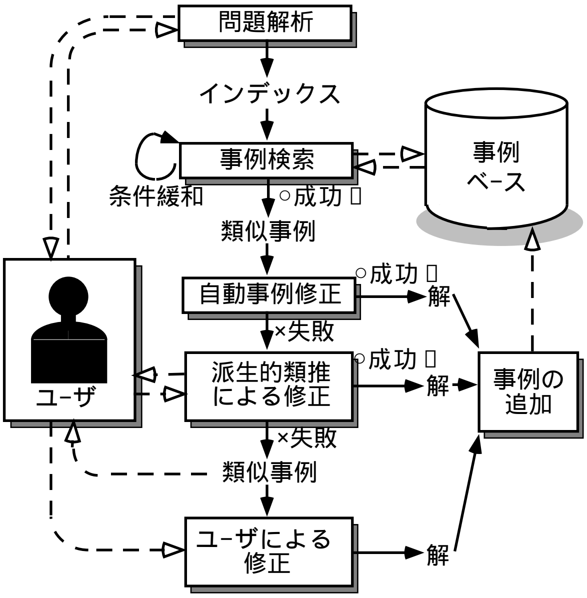

図 1: システム構成

図 1 にシステム構成を示す. システムは複数のインデックスを用いて最も類似した事例を検索する. 完全にマッチする類似事例が検索できないときは, 検索条件を緩めて, とにかく類似した事例を探す. 最善の場合, 非常に類似した事例が検索され, 与えられた問題への変換が自動的になされる. 一方, 最悪の場合でも, 何らかの関連性のある事例が検索されるので, ユーザによる並列プログラミングの手助けになる. 中間の場合には, スケレトンへの情報の追加が自動あるいは手動でなされる. スケレトンとは並列プログラムの骨格を示す最も重要な部分である. スケレトンにはスレッドの使用法, 共有変数の同期, タスク分割などと共に, 変数初期化, 計算, 結果の回収などの一連のプログラミングプロセスが記述されている.

3 並列アルゴリズムの分類

3.1 標準的な分類

- 分割統治法 (Divide and Conquer)
  問題は下位の部分問題に分割され, それら自身がさらに分割されて再帰的に解かれる. 最終結果は部分問題の解を再帰的に結合することにより得られる.
- プロセスネットワーク (Process Networks)
  計算を複数のステージに分け, データがステージを流れる. 入力データの有無によって, ステージは並行に動作する.
- プロセッサファーム (Processor Farms)
  問題は複数の独立な計算に分割され, それらの結果が統合される. 全体の制御は一ヶ所で行われ, 独立した計算の数はプログラマあるいは使用可能なスレーブ数によって決まる.
- 繰り返し変換 (Iterative Transformation)
  オブジェクトは複数の繰り返しステップにより変換され, 局所・広域・結合オペレータが適用される.

3.2 BACS による分類

BACS(Basel Algorithm Classification Scheme) [1] はアルゴリズムの分類, 複雑さの解析などの方法論を明らかにするために提案された.BACS は主にアルゴリズムのトポロジー, プロセス構造, 相互作用の方式, データ分散, 及び実行構造の観点から並列アルゴリズムを分類する.

Parallel Programming using Relevant Cases
Tomoyuki Furukawa and Katsuhiro Yamazaki
Ritsumeikan Univ.

4　事例の表現と例

4.1　一般的表現

(1) インデックス

- 応用 (データ操作, 数値計算, 記号処理, ビジネスデータ処理)
- 仕様 (問題の定義)
- アルゴリズム (分割統治法, プロセスネットワーク, プロセッサファーム, 繰り返し変換)
- **BACS** による分類
- 変数 (共有 / 局所, リダクション, コモン)
- 終了条件 (一定, 条件式)
- 同期 (ミューテックス, バリア, コンディション変数)
- 並列化手法 (パラレルリージョン, パラレルセクション, スレッドライブラリ)

(2) スケレトン

(3) プログラム

(4) 履歴

4.2　レイトレーシング

(1) インデックス

- 数値計算
- $At^2 + Bt + C = 0$ (球)
- プロセッサファーム
- $I^t_{barrier}, Fix_w(C^t)$
- 共有 / 局所
- 一定
- バリア
- スレッドライブラリ

(2) スケレトン

```
pthread_barrier_t A;
画像データ用変数の宣言;
main()
{
    変数の初期化;
    pthread_barrier_init(&A, プロセッサ数);
    for(スレッド数)
        pthread_create(ray, スレッド番号);
    ray((void *)0);
    画像データのファイルへの出力;
}
void *ray(スレッド番号)
{
    pthread_barrier_checkin(&A);
    スレッド番号によるタスクの割り付け;
    for(スレッド番号による計算範囲){
        レイトレーシングの計算部;
        画像データ用変数へのデータの格納;
    }
    pthread_barrier_checkout(&A);
}
```

4.3　2 分木ソート

(1) インデックス

- データ操作
- 要素 (<中央値) ならば左へ
  要素 (>中央値) ならば右へ
- 分割統治法
- C^t
- 共有 / 局所
- 一定
- なし
- スレッドライブラリ

(2) スケレトン

```
スレッド間通信用変数の宣言;
データ用変数の宣言;
void main(void)
{
    変数の初期化 ;
    sort(1);
    全ての処理の終了待ち;
}
void *sort(int no)
{
    if(no < 終了条件){
        中央値の探索;
        pn=no*2;
        pthread_create(sort,pn);
        sort(no*2+1);
    }
    else break;
}
```

5　事例の検索と修正

インデックスは抽象的なものから具体的なものへと並べられ, 優先付されている. インデックスが全て合致するもの, さもなければ優先順位の高いインデックスがより多く合致するものを検索する. 同じクラスのアルゴリズムに対して、類似したスケレトンに自動 / 手動で修正を行う派生的類推による修正が主である.

6　おわりに

現在, 仮想共有メモリ並列マシン KSR 1 を対象に、スケレトンの作成を進めている. 事例ベースの構築と事例の修正法の検討が今後の課題である.

参考文献

[1] Burkhart H.et al.“BACS:Basel Algorithm Classification Scheme”, *Technical Report 93-3*, Universitat Basel,1993.

[2] 山崎, 古川 “事例ベース推論による並列プログラミング支援システム”, 情報処理学会、94-OS-67-14,1994.