L_022

計算の検証の分散処理に関する一提案
A Proposal on Distributed Processing for Verification of Computation

茂木 篤† 山本 宙‡ 辻 秀一‡
Atsushi Mogi Hiroshi Yamamoto Hidekazu Tsuji

1. はじめに

本研究では効率的に分散処理できない計算であってもその計算結果の検証作業は効率的に分散処理できることに着目し，検証作業の分散処理等により，検証作業まで考慮に入れた効率的な分散処理手法を提案する．提案手法によりインターネット上の計算機を利用する手法の応用範囲を効率的に分散処理できない問題にまで拡大することが期待できる．

2. 効率的に分散処理出来ない計算

コンピュータに実行させる計算の中には，効率的に分散処理出来ないものが存在する．また，インターネットを介して計算を依頼する場合など，必ずしも信頼出来ない計算機に計算を依頼する場合には計算結果の検証作業が必要となる．

効率的に分散処理出来ない計算には，時系列的に後ろにある計算を行うために，前にある計算を解いておく必要があるという「計算間の依存関係」が存在する．この依存関係により，分散処理が効率的に行えないのである．

本研究ではコストの高い高速計算機とコストの低い低速計算機を使う事ができ，いずれも必ずしも信頼出来ない状況を考える．また，検証方法として複数の計算機に同じ入力の計算を依頼し，同じ出力が得られるかどうか確認する手段を用いる．

この場合，高速計算機に計算の途中結果を報告させ，低速計算機に途中結果からの計算を依頼することで，検証作業を効率的に分散処理することができる．

従来手法と提案手法の計算時間や計算時間を比較するためのパラメータを以下のように定義する．

n ：問題の総ステップ数
P_h ：高速計算機が全体の計算を正しく行う確率

3. 従来手法

3.1 概要と問題点

効率的に分散処理出来ない計算かつ，解の検証が必要な問題に関して，従来手法では高性能なシングルプロセッサを搭載したコンピュータ（高速計算機）を数台占有し，同じ処理を行わせて結果の照合をすることで解の正しさを確保している．

この手法では最低でも高速計算機の計算量が本来の 2 倍必要になる．

3.2 計算時間と計算コストの検証

従来手法のアルゴリズムは，以下のように定義する．また仮定として，高速計算機の数は十分あるものとし，間違った計算結果が偶然一致し計算が終了する可能性は無いものとする．

```
1）2 台の高速系先に同時に同じ計算をさせ，計算結果を記録する
2）while(計算されているすべての計算結果のうち，一致するものがない) {
   別の一台に計算させ，計算結果に追加する
}
3）一致するものが正しい計算結果であるから，答えを返して終了する
```

これらから，従来手法を用いｋ回目(ｋ≧２)の計算で計算終了する確率は以下のようになる．

$$(k-1)P_h^2(1-P_h)^{k-2} \quad \cdots (1.1)$$

また，終了までの計算回数の期待値は，式(1.1)より以下のようになる．

$$\sum_{k=2}^{\infty} k(k-1)P_h^2(1-P_h)^{k-2} \quad \cdots (1.2)$$

式(1.2)より，従来手法の計算コストの期待値は以下のようになる．

$$n\sum_{k=2}^{\infty} k(k-1)P_h^2(1-P_h)^{k-2} \quad \cdots (1.3)$$

計算時間の期待値は，最初の 2 回は同時に行うので以下のようになる．

$$n\left(\sum_{k=2}^{\infty} k(k-1)P_h^2(1-P_h)^{k-2}-1\right) \quad \cdots (1.4)$$

4. 提案手法

4.1 概要

提案手法では問題を時系列で分割する．そして分割された各問題（区間計算）を解く際は，データの入出力にレジュームデータを利用する．レジュームデータとは，決めら

† 東海大学大学院工学研究科情報理工学専攻，
Course of Information Science and Engineering, Graduate School of Engineering, Tokai University
‡ 東海大学情報理工学部情報メディア学科
Department of Information Media Technology, School of Information Science and Technology, Tokai University

れたポイント（レジュームポイント）以降での計算を再現するためのデータのセットである．

レジュームデータを用いる事で，計算の検証作業を分散処理させる事が可能になるとともに，計算失敗時の失敗箇所の特定が出来るようになる．

システムの構成要素はシステムサーバ，高速計算機，分散処理システム（低速計算機の集まり）の3つとなる．

システムサーバは，システムの利用者からの計算依頼の受け付けや，与えられた問題の分割，レジュームデータ（計算結果）の収集や検証等を行う．また，仮定としてこのシステムサーバは信頼できるものとする．

高速計算機の利用は本計算用の 1 台に抑え，解の検証計算用には分散処理システムであるグリッドコンピューティングを用いる．

高速計算機のコストは非常に高いので，占有台数を一台に抑えられる事は大きな利点である．また，グリッドコンピューティングを用いることで，その主な構成要素である低コストで潤沢な低速計算機の利用が出来る事や，余剰計算能力の有効活用といった恩恵を得る事も出来る．

4.2 レジュームデータ

問題の分割数を m とおく．区間計算の先頭から順に 1 から m までのナンバーを付け，それぞれの区間計算の解となるレジュームデータにも同じようにナンバーを付けるようにする．（区間計算 i を解くと，その解としてレジュームデータ i が作成される）

レジュームデータ 0 は，本計算開始前の全体入力などからなる．問題を m 個に分割しているが，最後の問題(区間計算 m)終了後にもレジュームポイントを設定し，レジュームデータ m として記録する．このレジュームデータ m が全体からの出力結果（本計算の解）となる．

検証の状況を保持するための構造体を chkresume と名づける．全体システムは 0 から m までのレジュームポイントそれぞれについて，その区間の計算を行わせた計算機の数だけ chkresume 型データを保持する．これを「検証用データ」と呼ぶ事とし，システムサーバで管理する．

chkresume を構成するフィールドを以下の様に定義する．

id ：計算機の ID
previous ：入力として使ったレジュームデータ i-1
resume ：previous から計算したレジュームデータ i
chk－input ：入力が検証済みか
chk－comp ：計算が検証済みか

4.3 検証までの流れ

システムサーバは初期設定として検証用データの区間 0 の領域にデータを作成する．データの各フィールドの値は，id=システムサーバの ID，previous=(Null)，resume=レジュームデータ 0，chk-input=True，chk-comp=True，とする．

次にシステムサーバはレジュームデータ 0 を一台の高速計算機と，グリッドに参加している複数の低速計算機に与え，計算の結果報告を待つ．

高速計算機はシステムサーバからレジュームデータ 0 を受け取ると区間計算 1 から順に計算を行い，以後チェックポイントに到達するたびに報告をシステムサーバに返す．

報告として返す情報は，依頼された区間のナンバーi（1≦i≦m），計算を担当した計算機の ID，今回その計算機が入力として利用したレジュームデータ i-1，計算結果のレジュームデータ i である．

高速計算機から報告を受け，システムサーバは検証用データの区間 1 の chkresume 型データを作成し報告から得た値（id，previous，resume）を代入する．chk-input，chk-comp の二つのフィールドは False とし，保存する．未検証のレジュームデータ 1 が得られたので，システムサーバはこれを複数の低速計算機に与える．この動作をレジュームデータ m-1 まで繰り返す．

低速計算機ではシステムサーバからレジュームデータ i を受け取ると区間計算 i+1 のみを行う．計算後は高速計算機の時と同じ形式でシステムサーバへの報告を行う．

システムサーバは低速計算機からの報告を受けると，高速計算機の時と同じように chkresume 型データを作成し，検証用データに追加する．

低速計算機の報告から，区間 i に関する新たな検証用データが追加されたら，システムサーバに登録済みの区間 i の検証データ全てと resume，previous の照合を行う．値の一致するものがあれば区間 i の検証は完了とし，一致した検証データの chk-comp フィールドに True を代入する．

区間単体の検証が終わったら，次は区間を通しての確認作業を行う．アルゴリズムは以下のようになる．

1）区間 i=0 から m-1 まで以下を繰り返す
2）chk-input か chk-comp が False なら検証やめループを抜ける
3）chk-input と chk-comp がともに True の時
　3－1）区間 i のデータは確定しているので区間 i に他のデータがあれば消去する
　3－2）次の区間 i+1 の previous が区間 i の resume と一致するものは chk-input フィールドを True にする
4）i=i+1

この作業の結果，複数の低速計算機が同じ答えを出し，高速計算機のデータが消される事もある．仮定として「間違った計算結果が偶然一致することは無い」とおいているので，この場合高速計算機のデータが誤りであるといえる．このような場合は，誤りだと分った区間以降の作業を，別の高速計算機に依頼する．

chkresume 型データの区間 m の要素のうち，chk-input，chk-comp フィールドがともに True のものがあれば，その要素の resume から全体の解を得て計算は終了となる．

5. 今後の課題

Globus toolkit 2.4.3 を用いてシステムを実装し，提案システムの計算時間や，計算コストの検証を行いたいと考えている．

＜参考文献＞
日本アイ・ビー・エムシステムズ・エンジニアリング株式会社 著，“グリッドコンピューティングとは何か Globus Tooklit ではじめるグリッドの基礎”，ソフトバンクパブリッシング株式会社，2004 年 4 月 5 日初版