

グリッド環境に適した組み合わせ最適問題の解法

日大生産工(院) 長尾 健太郎
日大生産工 松田 聖

1 はじめに

近年, 広い地域に配置された計算資源を用いて分散・並列計算を行うグローバルコンピューティングに関する研究が盛んに行われるようになってきた. 本稿では, その中の代表的な一つであるグリッドコンピューティングについて研究を行った. グリッドコンピューティングは広域ネットワーク上に分散配置された計算資源を仮想的な構成の高性能計算機(メタコンピューティング)とみだてて分散・並列計算を行う計算システムである. グリッドコンピューティングシステムにおいては, ユーザ認証, 通信, 遠隔計算機上でのプロセス生成などの様々な要素技術が必要になる. 本研究では, グリッドコンピューティングのミドルウェアとして標準となっている Globus Metacomputing Toolkit(Globus) を採用する.

本研究では Globus を用いて構築したグリッドシステム上で組み合わせ最適化問題を並列分散処理することで処理時間の短縮を目的とする. 適応する問題として, 組み合わせ最適化問題のひとつである, 巡回セールスマン問題 (TSP: Traveling Salesman Problem) を取り上げる. しかし, 従来問題をどの様に分散処理を行い, どの様に協調して解くのが大きな課題である. そこで本研究では, 巡回セールスマン問題の経路を探索木に変換し, 経路長の比較を行う制約条件を持たせ, 処理実行中に制約条件を更新することによる時間短縮を試みた.

2 Grid Computing とは

グリッドとは, ネットワークを介して複数のコンピュータを結ぶことで仮想的に高性能コンピュータをつくり, 利用者はそこから必要なだけ処理能力や記憶容量を取り出して使うシステムである. 複数のコンピュータに並列処理を行わせることで, 一台一

台の性能は低くとも高速に大量の処理を実行できるようになる. ビジネス利用や学術研究など, 多くの可能性が模索され, 実現に向けてさまざまな試みが行なわれている.

3 Globus Toolkit

Globus Toolkit は, オープンソフトウェアツールキットである. そのため, 世界中の多くのプロジェクトや開発者が, Globus Toolkit の開発に貢献している. Globus が提供するサービスは必要に応じて個別に利用する事ができるようになっており, 既存アプリケーションへのインクリメンタルな導入が可能である.

4 組み合わせ最適化問題

組み合わせ最適化問題は数理計画問題の一つの分野であり, 「与えられた制約条件のもとで, 最大の評価をえられるような組み合わせを求める」という問題であり, 様々な場面で取り扱われている. 組み合わせ最適化問題には巡回セールスマン問題, スケジューリング問題, ナップサック問題, 資源配分問題など実社会において広い応用範囲を持っているが, 実社会で求められる大規模な問題を解くには膨大な計算量が必要である. しかし, 組み合わせ最適化問題の多くが独立性の高い部分問題に分割しやすいという性質を持つため, 並列化によって高速化が期待できる.

4.1 巡回セールスマン問題

巡回セールスマン問題 (TSP) とは組み合わせ最適化問題の代表的な問題であり, 複数の都市をそれぞれ一度だけ訪問して出発点の都市に戻る巡回経路の中で経路長が最短のものを求める問題である.

solving combinatorial optimizations problem efficiently
in grid environment

Kentarou NAGAO¹ and Satoshi MATSUDA

巡回する都市の数を m とすると, 巡回経路の数は,

$$\frac{(m-1)!}{2}$$

と表せ, 下表のように

表 1: 探索数における経路数の増加

都市数	経路数
2	1
3	3
4	12
5	60
6	360
:	:
10	1814400
:	:
31	4.11×10^{33}
:	:
m	$(m-1)! \div 2$

図 1: 探索数における経路数の増加

従って, TSP を単純に計算するのは不可能である.

4.2 並列分散処理手法

与えた TSP 問題に対して以下の様な処理を行う. TSP 問題をまず, スタート地点から常に最短経路を取るルートを探査し, 得られた結果を比較基準として保存する. その後, 残った巡回経路を探索木に変換し, 探索木ごとに分割し各端末に配布する. 配布された探索木を反復進化法にて探索を行う. この時, 作成した比較基準を超える経路が発見された場合, その探索を打ち切り, 別の巡回経路を探査する. また, 比較基準よりも経路長が短いものが発見された場合, それを新たな比較基準とし, 別端末にもその比較基準を配布する.

5 システム構成

本研究では, 表 2 のように計 11 台でシミュレーションを行った. 検証する TSP の都市数は 10 都市より行った. 1 台で行ったシミュレーションは 10 都市から 15 都市, グリッド環境では, 15 都市から 20 都市のシミュレーションを行った. 以下にシステムの概要図を記載する.

表 2: 使用環境スペック

使用端末	CPU 数	メモリ容量
統括サーバ	Pen4 2.4GHz	2GB
探索 PC(同一スペック)	PenM 1.6GHz	512MB

6 シミュレーション結果

シミュレーションにより得られた結果を以下に示す.

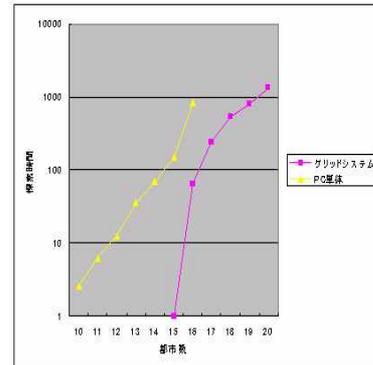


図 2: 検証結果

PC 一台の処理は指数関数的に増加しているが, グリッド環境では, その兆候が見られない. これは, 探索木を分割し各端末で分散処理を行い, 比較基準を持たせた事により, 大幅に経路探索の数を減らす事ができたと推測できる.

7 まとめ

本研究では, グリッドシステム上で並列分散処理を行うことで処理時間の短縮を目的としていた, シミュレーション結果からみると, 本来ならば指数関数的に増加するはずのグラフが, グリッド環境ではその傾向が見られなかった. その事から, 本研究での目的は達成できたとと言える. 今後は検証 PC や都市の数を増やし, さらにシミュレーションを続けていく.

参考文献

- [1] 日本アイ・ピー・エムシステム・エンジニアリング株式会社著 (2004) 『グリッドコンピューティングとは何か』, サイエンス社