クラウド環境での進化計算用グリッドシステムの並列化効率評価

信州大学　○杉山聖貴，松村嘉之，　広島大学　保田俊行，大倉和博

要 旨

進化計算は様々な工学的最適化問題に適用されている。しかし、最適化問題が複雑で大規模になるほど、計算時間が膨大になる．本研究では並列分散計算資源にクラウドを用いて，進化計算用グリッドシステムを構築し，効率的に計算時間を短縮することを目的とする．本稿では，不均質な計算機環境（ヘテロ環境）において，動的スケジューリング手法が効果的であることの一例を小規模な計算機実験より示し，目的達成の足掛かりとする．

1. 緒言

進化計算は最適化問題に対して頑健であり，様々な工学的問題に適用されている．しかし，取り扱う問題が複雑で大規模になればなるほど，計算機実行時間が延びる．これに対し，近年では進化計算手法が持つ本来の並列・分散性に着目し，複数の計算機をネットワークで接続して並列化を行い，計算時間を短縮する方法が用いられている．

本研究では，グリッド計算のように不均質（ヘテロジニアス，以下ヘテロ）な並列分散環境を有効利用することで，進化計算を高速化させる進化計算用グリッドシステムの構築を目指す．また，並列分散計算資源として，クラウドコンピューティング[1]（以下クラウド）に注目する．クラウドは，サービス（計算資源の提供）が，ネットワークにより共有されたインフラ上に，マルチテナント型で構築されているものである．クラウドサービスは従量課金制であり，サービスプロバイダを仲介することにより，管理の手間やサーバの維持費の節約ができる．また，アジリティやスケーラビリティに優れているため，必要に応じたリソースを速やかに利用することができる．

このようなグリッド・クラウド環境において並列分散計算を行う際に考慮すべき項目として，セキュリティ，不均質性，不確実性，並列性などが挙げられる[2,3]．本研究では，特に並列性と不均質性に着目し，クラウド環境および研究室内に設置された物理計算機環境において，どれだけ進化計算を効率的に並列実行できるかを検証する．

本稿では，ヘテロな計算機環境において，静的なスケジューリングより，動的なスケジューリングである Fuzzy Load Balancing が効果的であることの一例を小規模な計算機実験により示し，大規模なクラウド環境において，状況に応じた適応的 Load Balancing を備えた進化計算用グリッドシステムの設計の足掛かりとする．

2. 進化計算用グリッドシステム

並列計算を実現する手段として，グリッドコンピューティング（以下グリッド）がある．グリッドとは地理的・組織的に広範囲に分散した資源を対象にして，それらを協調的に動作させ，問題を処理させるための技術である[4]．ここで言う資源とは，広域ネットワーク上の計算資源や情報資源，センサや実験装置，人的資源などを指し，グリッドではそれらを仮想化・統合し，必要に応じて仮想計算機や仮想組織を動的に形成する．そのため，不均質な性能の計算機を扱えるため，新しい計算資源を容易に追加することが可能である．

その計算資源として利用するクラウドでは，複数台の物理計算機上でネットワークを構築し，分散処理技術・仮想化技術を用いてネットワーク上に仮想計算機群を構築する．それにより，計算資源の利用効率や追加性の向上が期待されている．そのクラウド上に生成された仮想計算機群にグリッドを実装することで進化計算用グリッドシステムが利用可能となる．クラウド環境での進化計算用グリッドシステムの概要図を図 1 に示す．CloudStack を用いたクラウド環境上において，計算粒度が一定の進化計算を実装したシステムに対して，90%以上の並列化効率を確認している[5].

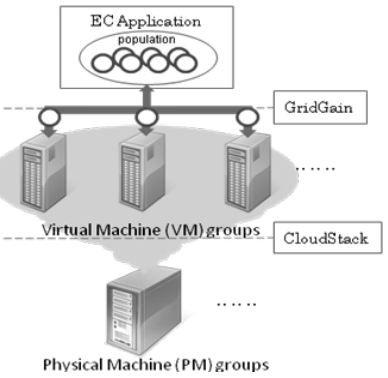

Fig. 1 Grid System for EC on Cloud Environment

3. タスクスケジューリング

グリッド環境中のタスクを実行する複数の計算機の計算負荷は時々刻々と変化するため，ユーザが使用する計算機でのスケジューリングには，動的スケジューリングが適している．本稿では，Fuzzy Load Balancing[6]と呼ばれる動的スケジューリングの手法の性能を検証する．Fuzzy Load Balancing は，隔世代交代モデルにおいて前の世代の評価時間，通信遅延時間から各計算機の許容能力を Fuzzy 制御によって決め，割当てるタスク数を計算機ごとに決定する．Fuzzy Load Balancing のアルゴリズムの流れを以下に示す．

1) 初期世代では同数のタスクを全ワーカに割当てる．
2) 各ワーカでタスクを計算する．
3) 各ワーカの相対評価時間，相対通信遅延時間を求める．
4) 相対時間をメンバーシップ関数に適合させ，従属度を求める．
5) 従属度に制御規則から対応する重みを掛け，加重平均してワーカの許容能力を求める．
6) 各ワーカの許容能力と前世代に割当てたタスク数から次世代のタスク数を求める．
7) 終了条件を満たさない時は 2)に戻る．

その比較対象として，タスクを並列ノード数分，均等に割当てる基本的な静的タスクスケジューリング手法である均等割当てを用いる．

4. 計算機実験

本実験では，表 1 に示した 2 つのスペックの計算機が混在しているヘテロな状況におけるタスクスケジューリングを対象としている. 最大並列ノード数は 5 ノードであり，並列計算をさせる際，必ず 1 台低速 PC が含まれている．

均等割当てによるスケジューリングと Fuzzy Load Balancing によるスケジューリング手法を比較するため，(15,100)-ES を 100 世代実行し，各世代での評価計算を分散させる．実装する最適化問題には，式(1)に示した計算粒度が一定の関数最適化問題である Ridge 関数を用いる．ただし，Ridge 関数最適化問題だけでは 1 個体の適応度計算はごくわずかであるため，進化計算の工学的最適化問題を想定した負荷として，一定量の LU 分解の計算を適応度計算の後に行う．LU 分解のサイズは 570 行 570 列とし，表 1 の標準 PC で 10 回計算させ，1 個体の評価時間を約 1 秒となるように調整している．一方，低速 PC で同じ計算をさせた時の計算時間は約 3.5 秒である．進化計算の並列化には, グリッドミドルウェアである GridGain (version 2.0.3)[7] を用いた.

$$F(x) = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2 \quad (1)$$

$$(-64 \le x_i \le 64)$$

$$\min(F(x)) = F(0,0,\ldots,0) = 0$$

また, 本実験では, 並列化効率, 高速化率から評価を行う．並列化効率 E(n)は式(2)を用いて算出した.

$$E(n) = \frac{\text{ideal calculation time [ms]}}{\text{actual calculation time [ms]}} \quad (2)$$

1 ノードで全ての処理を行った場合の 1 試行の計算時間を取得した．この計算時間を並列ノード数で割ることにより，各ノードで並列処理した場合の理想計算時間(ideal calculation time)[ms]を求めた．そして，理想計算時間を実計算時間(actual calculation time)[ms]で割り，同期待ち時間や通信時間を含んだ場合の並列化効率を求めた．高速化率 S(n)は式(3)を用いて算出した.

$$S(n) = \frac{\text{actual calculation time in 1 node [ms]}}{\text{actual calculation time in n nodes [ms]}} \quad (3)$$

1 ノードでの実計算時間から n ノードでの実計算時間で割ることにより，逐次処理からどれだけ速くなったかを表す.

式(2)で求めた並列化効率を図 2，式(3)で求めた高速化率を図 3 に示す．並列化効率では，均等割当てが 30%程度であるのに対し，Fuzzy Load Balancing は 60%から 70%を示し，均等割当てに対して 30%以上の向上が確認された．高速化率では，Fuzzy Load Balancing は均等割当てに対して，各並列ノード数で 2 倍以上の高速化を実現することができた.

Table 1 Experimental Environment

	Standard PC	Low Speed PC
CPU	Intel Celeron G530	Intel Celeron 430
Clock Frequency	2.4 GHz	1.8 GHz
OS	Fedora 17	Vine Linux 4.2

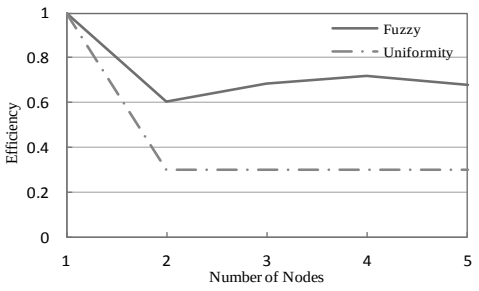

Fig. 2 Efficiency

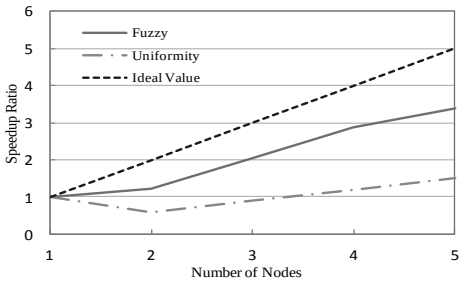

Fig. 3 Speedup Ratio

5. 結言

本稿では，性能の異なる計算機が混在する並列分散環境において，効率良く進化計算の計算時間短縮を実現するために，Fuzzy Load Balancing によるタスクスケジューリングを行った．実験結果より，研究室内の不均質な計算機環境において，均等割当てのおよそ 2 倍の計算時間短縮を実現することができ，効果的な負荷分散を確認することができた．今後はこの結果を踏まえて，クラウド環境において，進化計算の計算粒度が個体ごとに変化する問題に対して，効果的なグリッドシステムの設計を目指していく.

6. 参考文献

[1] P. Mell, T. Grance “The NIST Definition of Cloud Computing”, Special Publication, 800-1457 (2011).

[2] R. Baker, R. Buyya and D. Laforenza “The grid”, International effort in global computing, International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet (SSGRR 2000), I’ Aquila, Rome, Italy. (2000).

[3] 染谷博司, “グリッドでの最適化について”, 研究会「並列計算・乱数・グリッド・HPC」講演資料, 統計数理研究所, 東京, (2003).

[4] I. Foster and C. Kesselman “The Grid”, Blueprint for a new computing infrastructure, Morgan Kaufmann Publishers, (1999).

[5] 杉山, 松村, 保田, 大倉, “CloudStack を用いたクラウド環境での進化計算用グリッドサービス”, 第 57 回システム制御情報学会（SCI’13）, 327-5 (2003).

[6] Lotika Singh, Apurva Narayan, Satish Kumar, “Dynamic Fuzzy Load Balancing on LAM/MPI Clusters with Applications in Parallel Master-Slave Implementations of an Evolutionary Neuro-fuzzy Learning System”, IEEE International Conference on Fuzzy Systems, pp.1782-1788 (2008).

[7] GridGain, http://www.gridgain.com/