2014 年 1 月 6 日 情報システム基盤学合同輪講発表資料

ARiA:A Protocol for Dynamic Fully Distributed Grid Meta-Scheduling

著者： Amos Brocco, Apostolos Malatras, Ye Huang, Béat Hirsbrunner
出典： *Proceedings of The 30th international Conference on Distributed Computing Systems, IEEE, 2010, pp86-95*
発表者： 高性能コンピューティング学講座 本多・近藤研究室　1353029 松下 明史

1　はじめに

今日、地理的に分散したノード (計算資源) を接続し、高いスループットを得るグリッドコンピューティングが注目されている。このグリッドコンピューティングには様々な形態が存在するが、本論文では、複数のノードで依存関係がないジョブが一定間隔で発生する形態を想定している。この形態でより高いスループットを得るためには、新たに発生したジョブを他のノードへ転送することで、ジョブの発生から完了までの時間を短縮することが重要となる。そして、この方法を実現するには、メタスケジューリングの改良が不可欠である。

本論文では、ジョブを高速かつ適切に分配するメタスケジューリング手法、ARiA Protocol を提案し、シミュレーションで評価を行った。

2　メタスケジューリング

メタスケジューリングとは、ローカルスケジューリングの前に行う、ジョブを実行するノードを決める機構のことである。このメタスケジューリングには、ジョブの転送先を決めるノードの違いから大きく 2 つの手法に分けられる。

1 つ目は、メタスケジューリングに必要な情報や転送したいジョブを管理ノードと呼ばれるノードに集め、そこで転送先を決定する集中型である。この集中型ではグリッド全体の通信量を抑えることができる。

2 つ目は、各ノードがメタスケジューリングに必要な情報を集め、ジョブの転送先を決定する分散型である。グリッド全体の通信量は多くなるが、耐故障性や拡張性において集中型より優れている。

ちなみに、ローカルスケジューリングとは、各ノードが保持するジョブの実行順を決める機構のことである。本研究では、ジョブの投入時刻が早い順に実行する FCFS(First-Come-First-Served) や予測実行時間が短い順に実行する SJF(Shortest-Job-First) などを用いる。

3　目的

適切なメタスケジューリングを行うには、グリッド全体の状態などを把握しなければならない。しかし、これらの情報の取得や、転送先を決定する処理がグリッド全体のスループット低下の原因になってはならない。そのため、適切なノードへ転送する信頼性と共に高速な動作も求められる。

本研究の目的は、近年のグリッドシステムを構成するノード数が増加している傾向を考慮し、拡張性が高い分散型のメタスケジューリングを用いて、高速かつ低負荷となる手法を提案する。

4　提案手法

グリッドコンピューティング上で分散型メタスケジューリングを行う機構、ARiA Protocol を提案する。この提案手法では、図 1 で示した ACCEPT、REQUEST、INFORM、ASSIGN の 4 つのメッセージのみ使用し、必要とする情報と通信回数を減らすことで高速にメタスケジューリングを行う。

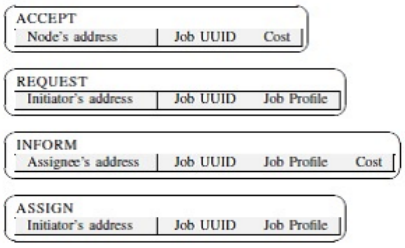

図 1: 提案手法で用いられるメッセージと内容

4.1　Job Submission Phase

ジョブの投入と投入されたノードでのジョブの初期処理を行う段階である。まず、投入されたノードにおいてジョブは固有の識別子 (Job UUID) と、ジョブの内容や条件等を表したデータ (Job Profile) を生成する [2]。そして、このジョブの条件に合うノードを見つけるため、前述のデータに発行元のアドレスを追加した REQUEST メッセージをグリッド上のノードへ通知し、所定の時間まで待機する。

4.2　Job Acceptance Phase

前段階で発行された REQUEST メッセージの情報を元に、受信したノードで Cost を求める段階である。Cost とは、具体的に対象ノードが既に保持しているジョブの締め切り時間 (deadline) と予測完了時間 (ETC) の 2 つの値から求める値 NAL である。以下に、NAL を求める式を示す。

$$NAL_{cost}(j) = \sum_{job \in Q'} \delta(job, Q') * |\gamma_{job}|$$

$$Q' = Q \cup \{j\}$$
$$\gamma_{job} = deadline_{job} - ETC_{job}$$
$$\delta(job, Q') = \begin{cases} -1 & \gamma_{job} \geq 0, job \in S \\ 0 & \gamma_{job} \geq 0 \wedge w \in S : \gamma_w < 0 \\ 1 & otherwise \end{cases}$$

求めた NAL を Cost とし、自身のアドレスとジョブ識別子を負荷した ACCEPT メッセージを REQUEST メッセージ発行元へ送信する。その後、REQUEST メッセージ発行元は、送られてきた Cost の中で一番低い値を提示したノードに対して ASSIGN メッセージを送り、その後ジョブの送受信が実行される。

4.3 Dynamic Rescheduling Phase

この動作は、グリッドの動的な環境変化に応じてジョブの再分配を行う段階である。この再分配の対象となるジョブは、各ノードで実行されるまでの待機時間が長い、もしくは、締め切り時間を違反することが確定しているジョブである。この対象ジョブに関する情報と再分配時点での Cost、対象ジョブを保持しているノードのアドレスをまとめた INFORM メッセージを配信する。INFORM メッセージを受信したノードは Job Acceptance Phase と同様に Cost を求める。そして、INFORM メッセージの Cost より低い値が出たノードのみ ACCEPT メッセージを送り、その中から一番低い値のノードへ再分配する。

5 実験、評価

提案手法 ARiA Protocol をシミュレーションによって評価した。ローカルスケジューリングなどの条件を変化させ、同時に Dynamic Rescheduling Phase の有無による変化に注目して評価した。

図 2、図 3 は各ノードのジョブスケジューリングを、FCFS と SJF のどちらかに統一、または混合した状態でシミュレーションを行った際のジョブ 1 つ当たりの平均待機時間と実行時間、遊休ノード数の変化を表した図である。これらの図から SJF に統一した場合と混合した場合では、予測実行時間が長いジョブを再分配することで平均待機時間と遊休ノード数を大きく減少することができた。

また、図 4 は、様々な条件でシミュレーションを行った際の各メッセージの通信量を示した図である。再分配時に使用する INFORM メッセージで、通信量が大きく上昇しているが、シミュレーション時間やノード数から 1 ノード当たり 149bps しか上昇しない。

6 おわりに

グリッド上で情報量や通信回数を極力減らすことで、高速に動作するメタスケジューリング手法、AR*i*A Protocol を提案し、評価を行った。シミュレーションの結果、様々な条件において再分配の動作がグリッド全体のスループット向上に有効であることが示せた。今後は異なるネットワーク構築法、スケジューリング手法に提案手法を適用して比較されると思われる。

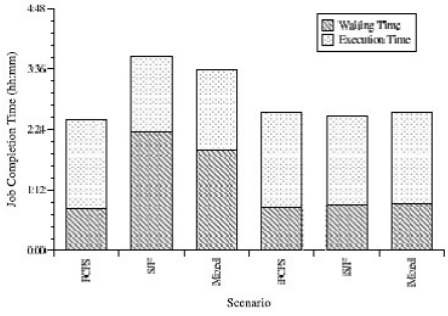

図 2: 各条件におけるジョブの平均待機時間と実行時間

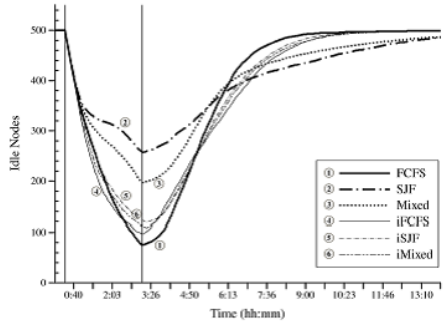

図 3: 各条件における遊休ノード数の変化

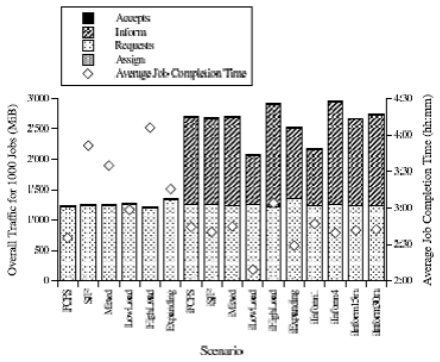

図 4: 各条件におけるメッセージ通信量

参考文献

[1] 1]Amos Brocco, Beat Hirsbrunner “ Service provisioning fremwork for a self-organized grid ” In Proceedings of GridPeer 2009 Workshop, international Conference on Computer Communications and Networks, pp1-6, 2009

[2] Graham R. Nudd, Darren J. Kerbyson, Efstathios Papaefstathiou, Stewart C. Perry, John S. Harper, Daniel V. Wilcox “ Pace:a toolset for the performance prediction of parallel and distributed systems” International Journal of High Performance Computing Applications,pp228-251, 2000