2009 年 12 月 7 日 情報システム基盤学専攻合同輪講発表資料

Resource Allocation using Virtual Clusters

著者： Mark Stillwell1 David Schanzenbach Frederic Vivien Henri Casanova
出典： *Proceedings of the 2009 9th CCGrid,Pages 260-267,2009*
発表者： 所属:本多研究室 学籍番号:0953016 氏名:西川優

1 概要

現在, クラスタは大規模な科学技術計算, 大規模データ処理, ホスティングサービスの分野において, 多く用いられるようになってきている. しかし物理ホストの場所をとる問題や電力消費量が多い問題が欠点として挙げられる.

そこで本研究ではクラスタを構成する各物理ホストの CPU やメモリの使用率を考慮しつつジョブを各物理マシンに振り分けることで少ない物理ホスト数でも最大限のパフォーマンスを得ることで上記の問題の解決していくことを目標としている. それにより物理ホスト数の削減や各物理ホストの処理効率の向上などの効果が期待できる.

2 研究における前提条件

本研究では以下のような前提条件を設けている.

●クラスタは同種の物理ホストで構成されているとする.

●各ジョブを各物理ホストに割り当てるかをスケジュールしている間に処理すべきジョブの数が増減するケースは想定しないものとする.

●各ジョブはそれぞれ一つのバーチャルマシン（VM）インスタンスしか要求しないものとする.

●各ジョブ Ja には, 以下で定義されるメモリニーズ,CPU ニーズの情報が付与される.

$$Ja\ の\ CPU\ ニーズ = \frac{Ja\ の\ CPU\ 処理時間}{Ja\ の処理時間} \quad (1)$$

$$Ja\ のメモリニーズ = \frac{Ja\ の使用するメモリ容量}{Ja\ を処理する物理ホストのメモリ容量} \quad (2)$$

●各ホストに割り当てられるジョブの合計メモリニーズは 1 以下であるとする.

3 提案

本研究では, 既存のソフトウェアを用いてクラスタのシンプルなシステムアーキテクチャを提案し, かつ効果的にジョブを各物理ホストに割り当てるためのスケジューリングアルゴリズムを提案する.

3.1 システムアーキテクチャ

提案するシステムのアーキテクチャは図 1 のようになる.

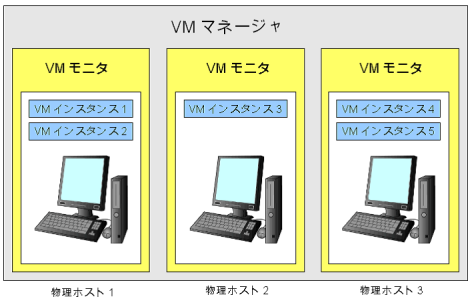

図 1: システムアーキテクチャ

1 台の物理ホストは 1 つの VM モニタに管理されており, 複数の VM インスタンスは 1 つの VM モニタに管理されている.VM インスタンスはジョブを処理する仮想計算資源であり,VM モニタは物理ホスト内で生成される各 VM インスタンスに割り当てられた CPU 使用率やメモリ使用率を決定, または変更することができる. すべての VM モニタは一つの VM マネージャによって管理され,VM マネージャは各 VM モニタを通してすべての物理マシンの VM インスタンスにおける CPU 使用率, メモリ使用率を決定できる. また VM インスタンスを他の物理ホストへ移動することができる.

3.2 スケジューリングアルゴリズム

ジョブを各物理マシンに割り当てるためのアルゴリズムの評価指標としてイールドの概念を提案する. イールドは以下のように定義する.

$$Ja\ のイールド = \frac{Ja\ の処理時間における\ CPU\ 処理時間の割合}{Ja\ の\ CPU\ ニーズ} \quad (3)$$

● Randomized Rounding(RRND)

まず最初に各ホストと各ジョブの対応表を作成し, 線形計画法を用いて最小イールドが最も高い解を算出し, 表の各値に入れていく. 次に求めた各値に丸め処理を行い,0 か 1 に分ける. ジョブ Ja とホスト Ha の組み合わせが 1 の場

合,Ha に Ja を割り当てる. このようにして一度対応表通りに Ha と Ja を振り分け, 振り分け可能な場合はでその組み合わせでジョブを割り当てる. 合計メモリニーズが 1 以上のホストが存在する場合は対応表における解の値を調節し, 再度繰り返す.

● Randomized Rounding with No Zero probability(RRNZ)

RRND の改良アルゴリズムとして提案されている. 違う部分は線形計画法を用いて算出した対応表に 0 の値が存在した場合,0 の代わりに非常に小さな値（本研究では 0.01 を使用）を入れる.

● Greedy(GR)

任意の順番でジョブのリストを作成する. 各ジョブに対して, ホストにすでに割り当てられたすべてのジョブの CPU ニーズの合計の昇順にランク付けする. その後, ジョブのメモリニーズを満たす最初のホストを選択する.

● Sorted-Task Greedy(SG)

GR の改良アルゴリズムとして提案されている.SG は最初にジョブリストをメモリニーズの降順に並べ替え, その後は GR と同じスケジューリングを行う.

● Greedy with Backtracking(GB)

GR にバックトラッキング機能（単に各ジョブと各ホストの組み合わせの中で, 割り当て可能な組み合わせを見つけるまで総当たり方式で試していく機能）を付加したアルゴリズム.

● Sorted Greedy with Backtracking(SGB)

SG と GB を組み合わせたアルゴリズム.

● Multi-Capacity Bin Packing Algorithm8(MCB8)[1]

各ジョブをメモリニーズよりも CPU ニーズの方が高いリストと,CPU ニーズよりもメモリニーズの方が高いリストの二つに分ける. 各リストは CPU ニーズ, メモリニーズの合計の降順にソートされる. 最初に選択された物理ホストの CPU 利用可能割合とメモリ利用可能割合を比べ, 大きい方と対応するリストからジョブを割り当てる.（CPU 利用可能割合が 0.4, メモリ利用可能割合が 0.6 の時はメモリニーズの方が CPU ニーズよりも高いリストからジョブが割り当てられる）これで両方のリストの先頭ジョブが割り当て可能ではない時, 次のホストを選択し同様に繰り返していく.

4 実験と考察

実験では CPU ニーズ, メモリニーズの各変動係数が 0.25,0.75 の正規分布に従うようなジョブ数と各ニーズの組み合わせでジョブを生成し, シミュレーションを行う.

図 2 の結果から, 最小イールドの平均値は MCB8 が最も高い値を出しているということがわかった. また RRNZ は図の中にすら出てこないほど最小イールドが小さい結果になった.

図 3 の結果で 500 ジョブのスケジューリングにかかった時間は greedy アルゴリズムが 15〜20 ミリ秒,MCB8 は 500 ミリ秒,RRND,RRNZ は 650 秒という結果になった.MCB8 はスケジューリングの精度が高い半面, スケジューリング時間が greedy と比べて遅くなってしまうため, 用途に応じて使い分ける必要がある.

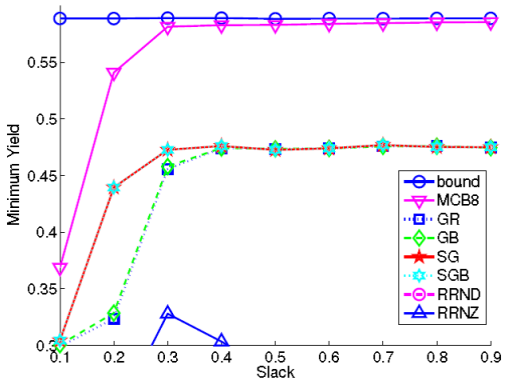

図 2: 最小イールドの平均と空きメモリ領域の関係の推移

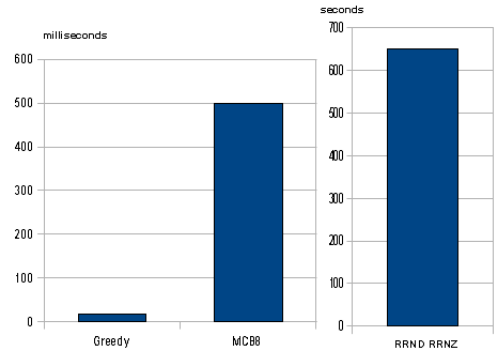

図 3: 各アルゴリズムの実行時間

5 まとめ

本研究ではジョブをクラスタリソースに割り当てるための新しい取り組みを実装した. また, 優れたアルゴリズムを提案し, 実験することでその有用性を示すことに成功した. 今後はジョブ数がスケジューリング中に増える場合やメモリニーズや CPU ニーズを把握していないジョブが混ざっている場合にも対応できるようにしていくことが課題として挙げられる.

参考文献

[1] ”Multi-Capacity Bin Packing Algorithms with Applications to Job Scheduling under Multiple Constraints” Proceedings of the 1999 International Conference on Parallel Processing,Page: 404 Year:1999, William Leinberger,George Karypis,Vipin Kumar