L_023

ユーティリティグリッドに向けた計算機状態管理手法の提案
Configuration Time Reduction by Machine State Management for Utility Grid Systems

小倉 章嗣† 荒木 拓也†
Shoji Ogura Takuya Araki

1．背景

近年、サービスやソフトウェア、計算機などを必要に応じて契約、利用する、ユーティリティグリッドが注目されている[1]。ユーティリティグリッドでは、多数の計算機資源を複数のサービスが共有する。それぞれのサービスは、必要な時に必要な量だけ計算機資源を要求、利用することで、維持費の削減、およびサービス要求の急激な増大への対処が可能となる。

計算機を利用するためには、計算機に対して複数の設定を行なう必要があり、この設定作業に時間がかかることが問題となる。計算機を利用するためには一般的に、OS のインストール、設定、ライブラリのインストール、アプリケーションのインストール、データの配置などが必要である。これらの手順を全て行なうと数時間かかることもあり、すぐに計算機を利用することができない。

計算機が利用可能になるまでの時間を短縮するために、現在はホットスタンバイ手法が用いられている[2]。ホットスタンバイ手法では、計算機をすぐに利用できる状態に事前に設定しておく。このように設定された計算機であれば、計算機の利用要求を受けてから、すぐに計算機資源を利用することが可能になる。

しかし、ユーティリティグリッドのような、対象となるサービスが多い状況でホットスタンバイ手法を適用した場合、必要な計算機資源数が増大してしまうという問題がある。これは、ホットスタンバイ手法では、対象となるサービス毎に計算機を待機させておく必要があり、対象となるサービスが増えると、それに比例して必要な待機計算機数も多くなるためである。

2．ユーティリティグリッドに向けた計算機状態管理手法

本論文では、ユーティリティグリッドにおいて必要な計算機数が増大する問題に対処するために、複数サービス向けに計算機を設定することによって、計算機設定時間を短縮しつつも必要な計算機数を抑える手法、Advanced Standby Management (ASM)を提案する。ホットスタンバイでは前述の通り、サービスの数に比例して、必要な計算機数が多くなる。このような状況の場合、ASM では複数のサービスに対応できる状態に空き計算機を維持することで、空き計算機が少ない状況でも多くのサービスに対して計算機設定時間の短縮を可能にする。

例を用いて ASM を説明する。Web サーバを利用するサービスでは、OS のインストール、OS の設定、Web サーバのインストール、設定、データの配置の設定が必要となる。ホットスタンバイでは、複数のユーザが存在する場合、それぞれのユーザに対してデータ配置まで完了した状態の計算機を用意するため、多くの計算機が必要となる。これを防ぐために、ASM では計算機の設定を途中（Web サーバを設定してあるが、データの配置は行なっていない状態など）で止めておき、複数のユーザに振り向けられるように計算機を維持しておくこと。これにより、必要な計算機数を減らす。

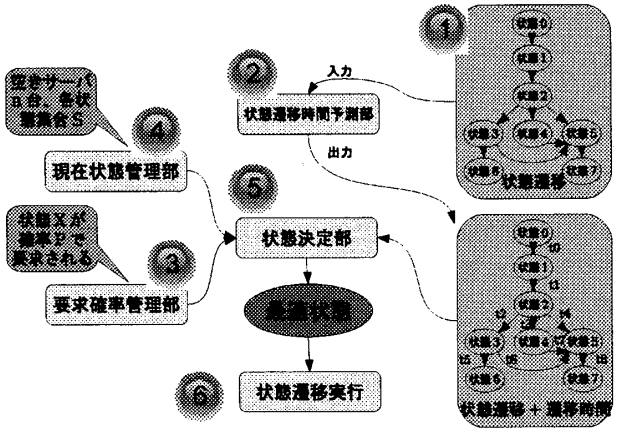

図 1 ASM の処理フロー

計算機設定時間を短縮するために、設定するべき計算機の状態を決定する必要がある。ここで、計算機の状態とは、「Web サーバを設定した状態」など、計算機が特定の設定をされた状態のことを指すものとする。計算機の状態を、複数のサービスに振り向けられる状態に設定しただけでは、計算機設定時間を短縮できるとは限らない。例えば、多くのサービスがある中でも、特定のサービスが頻繁に使われるような状況では、そのサービス向けにホットスタンバイを用意した方が短時間で利用可能な状態になる可能性が高まる。このような状況に対応できるように、計算機の状態を決定する必要がある。

計算機の状態を決定する指標としては、設定時間の期待値、分散、最大値などを最小にすることが考えられる。これらの指標は利用者の要求によって変わるため、それぞれが扱える必要がある。初期の試作では、利用要求に対して平均的に短い時間で対応できることが重要であると考え、計算機設定時間の期待値を最小にすることを指標とした。

計算機設定時間の期待値を最小にする状態を決定するためには、計算機利用要求確率、計算機設定時間、計算機の状態遷移規則といった情報が必要である。計算機利用要求確率、計算機設定時間については、過去の計算機利用要求、計算機設定履歴からの予測値を用いる。また、計算機の状態遷移規則は、個々のサービスの設定を行なう手順から生成する。

ASM の動作手順は、次の 6 つに分けられる(図 1)。まず、計算機の設定手順をまとめ、一つの状態遷移にすること、2 つ目は、各状態遷移の遷移時間を予測すること、3 つ目は、資源利用要求の到着確率を予測すること、4 つ目は、計算機のインストール済みアプリケーションなどを確認し、計算機の状態を把握すること、5 つ目は、上記の情報から

†日本電気(株) インターネットシステム研究所

最適な状態を計算すること、6 つ目は、求められた最適な状態に計算機を変更することである。

３．評価

本節では、ASM による計算機設定時間短縮の効果を評価した結果を示す。

我々は、提案手法を実現するシステム、Advanced Standby Manager を試作した。本試作では、状態管理部分、状態計算部分を C++で実装し、また、各計算機を操作する部分はシェルスクリプト呼び出しとして実装した。また、シミュレーションベースの評価を行なうために、資源としてダミーの資源を利用できるようにした。ダミーの資源は、設定された分布の計算機設定時間を算出する。

評価では、3 つの異なる計算機設定方法を比較する。それぞれ、空き計算機が全て OS 無しの状態に設定されている場合(CI)、ホットスタンバイ状態を利用要求確率の比率に応じて決定し、設定している場合(HS)、ASM を用いて計算機の状態を計算して設定している場合(ASM)である。また、評価指標は、計算機設定時間の平均値とする。

シミュレーションでは、Web 三層モデル、複数のユーザがいる環境を想定した。Web 三層モデルは、Web サーバ(Web)、アプリケーションサーバ(AP)、データベースサーバ(DB)からなる。各層はそれぞれ独立したハードウェア上で動作しており、各層のサーバが独立して利用要求を受けるものとする。また、5 組の独立した利用者が存在すると仮定し、利用していたサーバを他のユーザが利用する際には、データ保護の観点から、ディスクフォーマット、OS の再インストールを行なうものとした。

シミュレーションでは、与えられた要求発生確率に従って要求を生成し、その要求に対して計算機を設定する時間を評価した。実験シナリオで利用した要求発生確率を、表１に示す。この確率は、システムが要求を受け取った時に、どの要求であるかを表す。各状態遷移にかかる時間(設定時間)は、事前に設定された値から+10 秒までの範囲で一様分布をとる。設定された状態遷移、および状態遷移時間を、図２に示す。グラフの中のノードが計算機状態を表し、エッジ中に書かれている数字は、状態遷移時間(秒)である。

表１評価で用いた要求確率

	User0	User1	User2	User3	User4
WEB	0.034	0.069	0.046	0.055	0.055
AP	0.060	0.106	0.116	0.119	0.119
DB	0.008	0.061	0.045	0.069	0.038

図３に評価結果を示す。図３のグラフは、各手法において、71 回目の試行から、100 回目の試行までの計算機設定時間の平均値を示したものである。グラフの横軸は、空き計算機数を示す。複数回試行を重ねた後の値を示しているのは、回数を重ねる毎に要求確率の推定値がより高精度になっていくため、回数を重ねた後の値で比較した方が、より良い結果が得られるためである。

他手法と比較した結果、ASM では空き計算機数が少ない状況においても、計算機設定時間の平均値が小さく、空き計算機数が２台の状況では、約 2 割、設定時間が短いことがわかる。これは、空き計算機数が少ない状況では、ASM が複数のアプリケーション、ユーザに計算機を振り向けられる状態に計算機を設定し、複数の要求に平均的に短時間で応えられたためと考えられる。

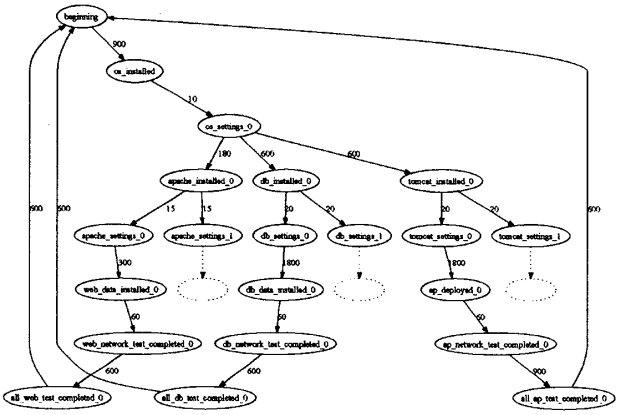

図２　評価で用いた計算機の状態遷移

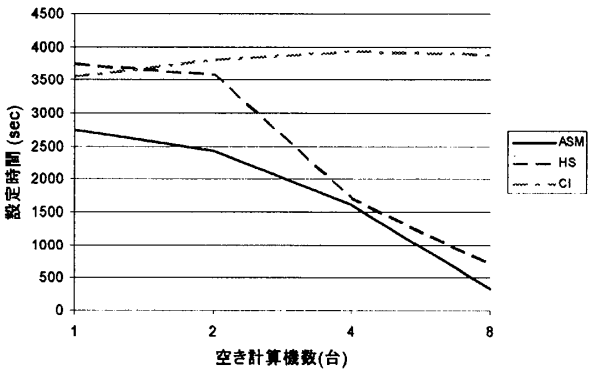

図３　空き計算機数に対する平均設定時間

４．まとめと今後の課題

本論文では、ユーティリティグリッドにおいて計算機設定時間を短縮するための手法である、Advanced Standby Management (ASM) を提案、評価した。ユーティリティコンピューティングにおいて、設定されるサービスが多い状況で、従来の設定時間短縮手法を適用した場合、必要な計算機資源数が増大してしまうことが問題となっていた。本提案では、すぐに利用できる状態に計算機を設定するのではなく、複数のサービス向けに設定可能な状態に計算機を維持することによって、必要な計算機数を抑えながらも、計算機設定時間を短縮することを可能にした。我々は本提案を実現するプロトタイプシステム Advanced Standby Manager を作成した。評価の結果、特定のシナリオでは従来手法に比べて計算機設定時間の平均値を約 2 割低減できることを確認した。

今後は、実計算機での評価、他の評価値の利用検討、要求確率推定値の高精度化を行なっていく。

参考文献

[1] Sun Grid Compute Utility, http://jp.sun.com/back/2005/0323/feature/

[2] K. Appleby,et al. Oceano - SLA Based Management of a Computing Utility.In IEEE/IFIP International Symposium on Integrated Network Management, 2001.