Sistema de Monitoração para o Escalonamento de Processos: Estrutura e Métricas de desempenho

Márcio Augusto de Souza
msouza@uepg.br
Departamento de Informática
UEPG – Ponta Grossa- PR

Omar Andrés Carmona Cortez
omar@cefet-ma.br
Departamento Acadêmico de Informática CEFET -
São Luis – MA

Luciano José Senger
ljsenger@uepg.br
Departamento de Informática
UEPG – Ponta Grossa - PR

Regina Helena Carlucci Santana
rcs@icmc.usp.br
Instituto de Ciências Matemáticas e de Computação – USP
São Carlos – SP

Marcos José Santana
rcs@icmc.usp.br
Instituto de Ciências Matemáticas e de Computação – USP
São Carlos – SP

Resumo

Este artigo descreve o PSMS (Process Scheduling Monitoring System – Sistema de Monitoração para o Escalonamento de Processos), uma ferramenta desenvolvida com o intuito de monitorar o escalonamento de processos em sistemas distribuídos. Os dados coletados através da monitoração feita pelo PSMS geram métricas de desempenho, que são escolhidas de acordo com o objetivo do escalonamento de processos, e sobre essas métricas são calculados valores de desempenho. Através da consulta desses valores, o escalonador pode concluir se os seus objetivos de escalonamento estão sendo alcançados e pode, opcionalmente, modificar o algoritmo de escalonamento ou fazer migrações de processo. Este artigo apresenta a estrutura do PSMS, uma análise das métricas de desempenho comumente encontradas em trabalhos científicos relacionados ao escalonamento de processos e um estudo de caso relacionado à monitoração de métricas relacionadas à CPU.

Abstract

This paper presents PSMS (Process Scheduling Monitoring System), a tool for monitoring the process scheduling in distributed systems. The data collected through the monitoring made by PSMS are used for calculating performance metrics chosen according to scheduling goals, and these metrics are used for calculating performance values. With the information returned by PSMS, the scheduler can conclude about the accomplishment of its scheduling goal, and may optionally change or adapt the scheduling algorithm used, allowing better performance. This paper presents the overall structure of PSMS, an analysis of performance metrics commonly found in scientific works related to process scheduling and a case study of its utilization, involving the scheduling of scientific applications and metrics related to CPU utilization.

1 Introdução

O escalonamento de processos é uma atividade computacional que tem grande influência no desempenho de um sistema computacional distribuído, pois uma boa distribuição dos processos entre os processadores disponíveis (que podem variar desde computadores em rede até nós de um computador paralelo) é indispensável para que o sistema distribuído ofereça um bom desempenho.

A definição de desempenho apresentada neste artigo é relacionada ao objetivo do escalonamento de processos. Exemplos de objetivos são: otimizar a utilização dos recursos do sistema, maximizar a vazão (*throughput*), reduzir o tempo médio de resposta (ou *slowdown*), balancear a carga, etc. Muitos objetivos são conflitantes entre

si, de maneira que é muito difícil implementar um algoritmo de escalonamento que seja eficiente para todos os possíveis objetivos [2] [3] [5] [13] [21].

Além dos diferentes objetivos, o sistema pode ser heterogêneo e sua carga de trabalho pode variar durante a sua utilização. O escalonador (ou *software* de escalonamento) deve levar em consideração todos esses fatores nas suas decisões de escalonamento, e o resultado deve ser um sistema com bom desempenho.

Escalonadores modernos devem ser capazes de adaptar ou trocar o algoritmo de escalonamento em casos onde haja problemas de desempenho, e os principais escalonadores dinâmicos fornecem funcionalidades nesse sentido, como *Sun Grid Engine* [18] e *LSF* [15].

Nesse contexto, é muito útil que o escalonador possa ser capaz de avaliar o desempenho do sistema em resposta ao algoritmo de escalonamento utilizado, através da análise de dados coletados no próprio sistema. Essa avaliação de desempenho feita em tempo de execução é desempenhada através de técnicas de monitoração.

Este artigo propõe um sistema de monitoração, que tem como foco o escalonamento de processos, para a avaliação do desempenho de sistemas distribuídos, chamado PSMS (*Process Scheduling Monitoring System* – Sistema de Monitoração do Escalonamento de Processos). De acordo com seu objetivo de escalonamento, o escalonador envia uma requisição de monitoração para o monitor, que será responsável por selecionar a métrica de desempenho adequada e coletar os dados necessários no sistema para calcular essa métrica. O monitor então retorna ao escalonador um valor que representa o desempenho do sistema.

Uma boa avaliação de desempenho só é possível com a escolha de métricas de desempenho corretas. Dentro desse contexto, este artigo também apresenta uma análise de métricas de desempenho comumente encontradas em trabalhos científicos relacionados à avaliação de desempenho do escalonamento de processos, enfatizando a sua adequação à abordagem de monitoração proposta neste artigo. Faz-se também uma discussão mais aprofundada sobre métricas de desempenho relacionadas à CPU, as quais são utilizadas no estudo de caso apresentado neste artigo.

Este artigo é organizado em 6 seções. A seção 2 apresenta alguns trabalhos relacionados envolvendo monitoração, escalonamento de processos e métricas de desempenho. Uma discussão sobre a relação entre métricas de desempenho e o escalonamento de processos é apresentada na seção 3. Na seção 4, o PSMS é apresentado. A seção 5 apresenta um estudo de caso da atuação do PSMS e, finalmente, a seção 6 apresenta os comentários finais e conclusões deste trabalho.

2 Trabalhos Relacionados

Existem muitas ferramentas que fornecem uma infra-estrutura para coletar e analisar dados obtidos no sistema, e esses dados podem ser usados para ajustar aplicações e *software*s do sistema para que obtenham melhor desempenho.

Os dados obtidos por essas ferramentas podem ser coletados em diversos níveis: pela monitoração de eventos do núcleo (*kernel*) do sistema operacional, como MAGNET [11] e Paradyn [14]; pela monitoração de aplicações e seus padrões de acesso aos recursos do sistema, como Autopilot [16]; pela monitoração da utilização dos recursos do sistema, como NWS [4] [19]. Adicionalmente, algumas ferramentas, como Autopilot e CODE [17], fornecem uma infraestrutura para a tomada de decisões que permita modificar o comportamento de aplicações distribuídas baseado nos dados coletados.

O escalonamento de processos é mais eficiente e consistente quando é feito levando em consideração dados coletados diretamente no sistema, e ferramentas de monitoração como Autopilot, NWS e PSMS fornecem informações que podem ser diretamente utilizadas por escalonadores.

Independente da ferramenta que esteja sendo utilizada, uma boa avaliação de desempenho depende da escolha de métricas de desempenho adequadas. Existe uma carência de trabalhos científicos que estudem a relação entre métricas de desempenho e o escalonamento de processos. Duas exceções são o trabalho de Feitelson e Rudolph [8], que mostra uma análise sobre a adequação de métricas a tipos de sistema computacionais, e o trabalho de Feitelson [9], que discute as métricas de desempenho em relação às suas características de convergência.

A próxima seção apresenta uma discussão sobre a utilização de métricas de desempenho para avaliar o escalonamento de processos, enfatizando a adequação dessas métricas a uma monitoração *on-line*, a qual é utilizada pelo PSMS, que será discutido na seção 4.

3 Métricas de Desempenho Aplicadas ao Escalonamento de Processos

Uma boa avaliação de desempenho, que permita obter resultados conclusivos, depende da correta escolha da métrica de desempenho a ser utilizada. Dentro do contexto definido neste artigo, que compreende a avaliação de desempenho do escalonamento de processos através de monitoração, o papel da métrica é fundamental, pois é a partir dela que pode se concluir a respeito da qualidade ou não de um escalonamento de processos.

Como exposto na seção anterior, existem poucos trabalhos que discutam a relação entre métricas de desempenho e o escalonamento de processos. Um exemplo é o trabalho de Feitelson e Rudolph [8], onde é apresentada uma análise das métricas de desempenho mais comuns usadas para avaliar o escalonamento de processos, discutindo a sua adequação a diferentes tipos de sistemas computacionais

Relacionar métricas de desempenho a tipos de sistemas computacionais é bastante útil, mas não é adequado à abordagem de monitoração descrita neste artigo, que considera que todo escalonamento de processos possui um objetivo. Portanto, a escolha da métrica deve estar atrelada a esse objetivo. Por exemplo, se um determinado sistema tem como objetivo minimizar a vazão, então este deve ser avaliado através da métrica vazão, independente do tipo de sistema computacional.

Além de serem relacionadas aos objetivos de um escalonamento, as métricas de desempenho devem ser adequadas ao tipo de monitoração utilizada, seja ela *on-line* ou *off-line*. O PSMS, por exemplo, é um monitor *on-line*, e as métricas são interpretadas no momento em que são coletadas.

Por exemplo, para monitoração *off-line*, uma métrica bastante utilizada é o tempo de resposta. Porém, para uma monitoração *on-line*, é preferível utilizar a métrica *slowdown*, que fornece uma visão instantânea da carga de um processador. Por exemplo, um valor de *slowdown* igual a 2, medido em um instante qualquer, significa que o tempo de execução das aplicações nesse instante está duas vezes maior do que o seu tempo de execução em um sistema disponível.

Diminuir o tempo de resposta é um objetivo bastante comum entre os algoritmos de escalonamento, conforme concluem Feitelson e Rudolph no trabalho citado nesta seção. Porém, mesmo que o objetivo seja diminuir o tempo de resposta, nem sempre a métrica tempo de resposta (ou *slowdown*) será a mais adequada para uma avaliação *on-line* desse objetivo.

Por exemplo, os trabalhos de Giné et al. [12] e Batat e Feitelson [1] afirmam que um bom escalonamento de processos, que leve em consideração a utilização de memória, deve minimizar a quantidade de paginação feita pelas aplicações do sistema. Para provar que um algoritmo de escalonamento direcionado à memória é melhor do que outro que seja direcionado à CPU, por exemplo, executam-se aplicações *memory-bound* e mostra-se que o tempo de resposta delas melhora quando se utiliza o algoritmo de escalonamento direcionado a minimizar a paginação.

Nesse caso, o algoritmo de escalonamento é validado (provou-se que ele garante melhor desempenho) através da métrica tempo de resposta, mas essa métrica não é a melhor escolha para monitorar o desempenho desse algoritmo durante sua utilização em sistemas computacionais. Para garantir que os tempos de resposta sejam bons, é importante garantir que a paginação seja minimizada. Então, um monitor *on-line* deve, idealmente, monitorar a quantidade de paginação do sistema, para garantir que o algoritmo de escalonamento esteja escalonando aplicações de maneira que se minimize a paginação.

Conclui-se então que dois fatores muito importantes a serem considerados na escolha de uma métrica de desempenho para monitoração são: o objetivo do escalonamento e o tipo de monitoração (*on-line* ou *off-line*) necessária.

O PSMS é um monitor *on-line*, baseado no objetivo da monitoração, e baseia as suas escolhas de métricas nesses dois fatores. Na próxima seção são discutidos a estrutura e principais componentes do PSMS.

4 Sistema de Monitoração Para o Escalonamento de Processos

O escalonamento de processos em um sistema distribuído é sempre desempenhado com um determinado objetivo em mente. Dessa maneira, avaliar o desempenho de um algoritmo de escalonamento consiste em analisar o quanto este objetivo foi alcançado [2] [7] [13] [21].

Essa avaliação é útil, pois o escalonador pode fazer uma decisão errada sobre o algoritmo de escalonamento, ou um algoritmo pode se tornar ineficiente por causa de uma variação na carga de trabalho. Nesse contexto, o

PSMS fornece a funcionalidade necessária para avaliar o quanto o objetivo de escalonamento está sendo alcançado. Com essa informação, o escalonador pode detectar problemas de desempenho e possivelmente corrigir esses problemas, por exemplo, através de ajustes no algoritmo de escalonamento ou através de migrações de processo.

O PSMS utiliza métricas comumente usadas para avaliar o escalonamento de processos [6] [8]. Exemplos das métricas fornecidas pelo PSMS são: vazão, percentual de utilização de um recurso do sistema e *slowdown*. Adicionalmente, o PSMS pode ser estendido pela inclusão de novas métricas de acordo com o desejo do administrador do sistema.

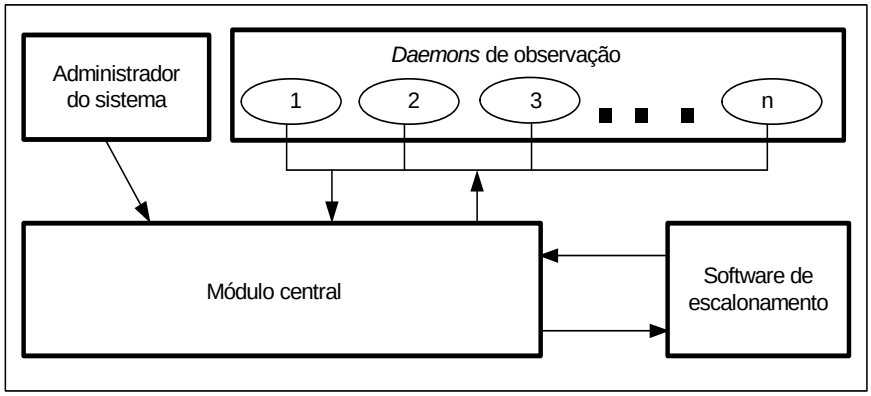

Figura 1. Estrutura do PSMS

O PSMS, cuja estrutura principal é mostrada na figura 1, é composto por um módulo central, que é responsável pelo controle de todas as funções do monitor e também pela comunicação com o escalonador (*software* de escalonamento), e por **n** *daemons* de observação (considerando que o sistema distribuído possua **n** processadores), que coletam dados locais aos processadores sendo monitorados. Os processos *daemon* são bastante simples, pois eles apenas iniciam a coleta dos dados requisitados pelo módulo central, e praticamente não causam sobrecarga no sistema.

O PSMS interage com o software de escalonamento, o qual é responsável por requisitar monitorações e receber um valor de desempenho, o qual é calculado a partir das métricas medidas nos *daemons*.

O valor de desempenho é calculado através da utilização de regras de decisão, as quais são aplicadas sobre o valor das métricas. O administrador do sistema tem a opção de definir novas regras de decisão e também novas métricas, e pode acrescentá-las ao sistema, de acordo com as suas necessidades.

O PSMS possui algumas regras de decisão já definidas, como por exemplo, a regra do coeficiente de variação, que será usada no estudo de caso. Essa regra consiste em calcular o coeficiente de variação dos valores das métricas medidas no sistema. Define-se um valor máximo para esse coeficiente, e o escalonador é alertado sobre um possível problema de desempenho caso esse valor seja ultrapassado.

A próxima seção apresenta um exemplo da utilização do PSMS e de como os resultados obtidos por ele podem ser usados.

5 Estudo de Caso

O estudo de caso mostrado neste artigo envolve os seguintes elementos: o protótipo do PSMS, um *benchmark CPU-bound* usado para calcular a métrica *slowdown* e duas aplicações compactas disponíveis no *Nasa Parallel Benchmarks* – BT e LU [20]. Essas aplicações compactas simulam aplicações encontradas em ambientes de pesquisa e são exemplos de aplicações *CPU-bound*. Aplicações científicas como BT e LU geralmente executam por horas, mas, para facilitar os experimentos mostrados neste artigo, as duas aplicações executam por minutos.

O sistema utilizado no estudo de caso é uma rede de computadores com três processadores executando o sistema operacional *Linux* e é descrito na tabela 1. Os processadores são comparados pelos seus fatores de desempenho (valores que representam a capacidade computacional do processador, e são usados para normalizar os valores das métricas em sistemas heterogêneos). O fator de desempenho é calculado através de *benchmarks* específicos do PSMS, e mostra que o processador P3 tem o dobro da capacidade de P1, ou seja, executa aplicações aproximadamente na metade do tempo.

Tabela 1. Configuração do Sistema Computacional Utilizado

Processador	Fator de Desempenho da CPU
P1	110
P2	150
P3	280

Para o experimento mostrado neste artigo, considera-se que o objetivo a ser alcançado pelo escalonador é balancear a carga do sistema, com ênfase na utilização da CPU. Também, considera-se que o escalonador de processos é organizado de maneira a balancear a carga no momento em que faz suas decisões de escalonamento, ou seja, a heterogeneidade do sistema é considerada durante o escalonamento. Esse fator é importante visto que todos os escalonadores modernos procuram balancear a carga do sistema enquanto distribuem os processos.

Basicamente, podem-se definir três métricas principais relacionadas à CPU, que são: *slowdown* de aplicações *CPU-bound*, percentual de utilização da CPU (percentual de tempo em que uma CPU mantém-se ocupada durante um determinado período de tempo) e tamanho da fila de processos da CPU [8][10].

Os valores das métricas de *slowdown* e percentual de utilização têm a vantagem de serem independentes de sistema operacional, enquanto que o tamanho da fila da CPU depende do tipo de sistema operacional e da maneira que se implementa o acesso à CPU. Além disso, as métricas *slowdown* e percentual de utilização levam em conta apenas a influência das aplicações que acessam a CPU, enquanto que a métrica tamanho da fila da CPU sofre influência de quaisquer aplicações que estejam executando no processador, independente de estarem usando a CPU ou não.

A métrica percentual de utilização da CPU tem a desvantagem de não representar a carga da CPU em determinadas situações. Por exemplo, em um sistema que esteja executando apenas aplicações *CPU-bound*, uma CPU com dois processos possuirá o mesmo percentual de utilização que uma CPU com apenas um processo, ou seja, 100%. Por isso, a métrica padrão do PSMS para medir a utilização de CPU é o *slowdown*.

A métrica *slowdown* é calculada através da execução de *benchmarks* (aplicações sintéticas inseridas no sistema pelo próprio PSMS).

Neste experimento, consideram-se duas aplicações paralelas distintas (BT e LU, ambas compostas por três processos). Suponha que o sistema esteja disponível, e as duas aplicações paralelas sejam escalonadas aproximadamente no mesmo instante. O escalonador, considerando as diferenças de capacidade de CPU entre os três processadores escalona três processos em P3, dois em P2 e somente um em P1, e as seis possibilidades de escalonamento são mostradas na tabela 2. O problema aqui é que LU termina antes, e o escalonamento feito inicialmente torna-se ineficiente.

Tabela 2. Distribuição de Processos.

	P1	P2	P3
Distribuição A	1 LU	2 LU	3 BT
Distribuição B	1 LU	1 BT e 1 LU	2 BT e 1 LU
Distribuição C	1 BT	2 LU	2 BT e 1 LU
Distribuição D	1 LU	2 BT	1 BT e 2 LU
Distribuição E	1 BT	1 BT e 1 LU	1 BT e 2 LU
Distribuição F	1 BT	2 BT	3 LU

A figura 2 mostra os valores de coeficiente de variação calculados no sistema para todas as seis diferentes possibilidades de escalonamento das duas aplicações, calculados em intervalos de 4 minutos. Considerando que ambas são submetidas ao mesmo tempo (tempo 4), todas as seis possibilidades podem ocorrer com igual probabilidade.

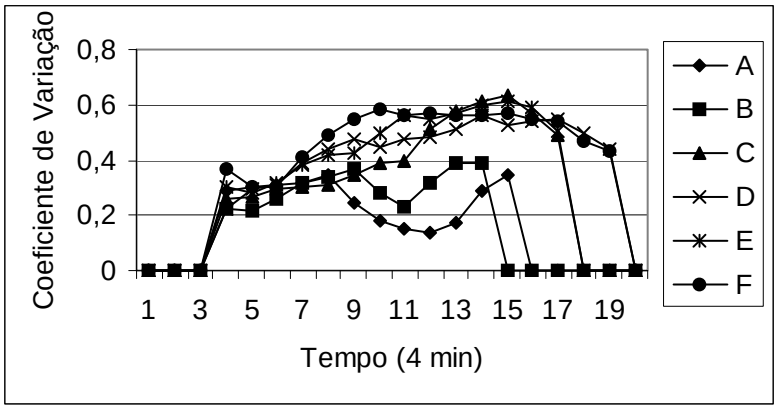

Figura 2. Valores de Coeficiente de Variação

Como pode ser visto na figura 2, todas as seis possibilidades de distribuição têm bom desempenho até o tempo 7, quanto LU termina. A partir do tempo 7, até que o sistema esteja disponível novamente, somente duas distribuições oferecem bom desempenho. As outras quatro degradam o desempenho do sistema, e elas poderiam requerer uma migração de processos para balancear a carga do sistema novamente.

Para detectar esse problema, o PSMS pode ser configurado para alertar o escalonador no momento onde os coeficientes de variação alcançam certo valor. Neste experimento, utilizou-se como limite o valor 0,40 (zero vírgula quarenta), pois esse valor apresentou resultados satisfatórios em todos os estudos de caso desempenhados durante o desenvolvimento do PSMS. Porém, esse valor pode ser modificado para se adaptar a sistemas computacionais específicos.

A figura 2 mostra que os coeficientes de variação das distribuições D, E e F possuem valor superior a 0,40 no tempo 8 (e no tempo 10, a distribuição C ultrapassa também 0,40), e nesse momento o PSMS pode alertar o escalonador sobre um problema de desempenho. Através de migrações de processo, por exemplo, o escalonador pode balancear a carga do sistema, permitindo que os processos restantes sejam executados mais rapidamente.

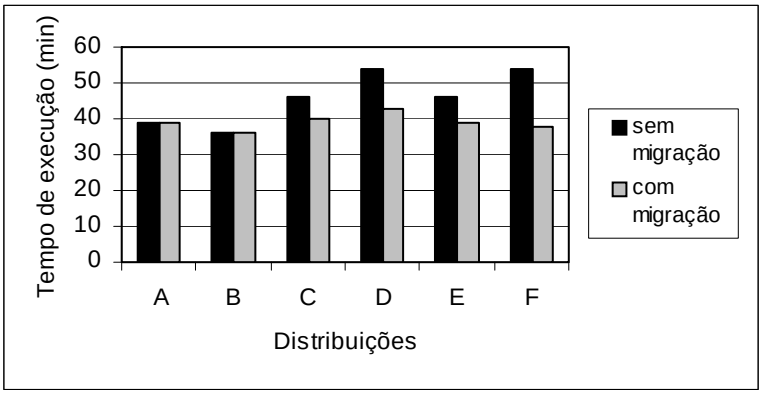

Figura 3. Tempos de Execução de Todas as Distribuições

A figura 3 mostra o tempo final de execução para todas as distribuições com e sem as migrações de processo. Os resultados mostram o melhor caso, onde o escalonador faz uma migração de processos assim que ele é alertado pelo PSMS. Neste experimento, o tempo de migração de processos foi considerado muito pequeno quando comparado ao tempo total de execução de uma aplicação científica *CPU-bound*, e por isso foi ignorado. Entretanto, é muito importante notar que fazer ou não uma migração de processos não é uma decisão do PSMS, e essa abordagem é mostrada neste artigo como um exemplo de uma ação que pode ser tomada pelo escalonador em caso de problema de desempenho.

O experimento mostrado neste artigo demonstra que diferentes escalonamentos podem gerar grandes diferenças de desempenho, e mesmo um bom escalonamento de processos pode falhar dependendo das características da carga de trabalho. Portanto, é bastante útil que o escalonador de processos possa estar continuamente analisando o seu desempenho, e o PSMS é uma ferramenta que permite ao escalonador manter-se informado sobre o desempenho do sistema em resposta ao escalonamento de processos.

6 Considerações Finais

Este artigo apresentou a estrutura geral do PSMS, uma análise das métricas de desempenho utilizadas por ele e também um experimento exemplificando a sua utilização. Como discutido ao longo deste artigo, é bastante útil fazer uma avaliação do desempenho do sistema em resposta ao escalonamento de processos, e o PSMS fornece a infra-estrutura necessária para os escalonadores avaliarem o quanto os seus objetivos de escalonamento estão sendo alcançados.

É muito importante que a escolha da métrica de desempenho seja criteriosa para que o PSMS retorne informações corretas, e essas métricas devem ser direcionadas ao objetivo do escalonamento de processos e também devem ser adequadas a uma monitoração *on-line*, conforme discutido na seção 3. No caso de avaliar a utilização da CPU, a métrica *slowdown* se mostrou bastante adequada.

A seção 5 apresentou um estudo de caso que mostra a viabilidade da utilização da infra-estrutura definida pelo PSMS. Esse experimento mostra que é possível que um algoritmo de escalonamento, mesmo quando concebido visando um determinado objetivo, pode ser prejudicado por diversos fatores, como por exemplo, uma mudança na carga de trabalho.

Detectar essa mudança no sistema é bastante útil, e no exemplo mostrado neste artigo, sugere-se que sejam feitas migrações de processo, mas diversos outros enfoques podem ser seguidos, como por exemplo, ajustar o algoritmo de escalonamento para se adaptar à nova situação do sistema.

O PSMS foi projetado de uma maneira modular, facilitando a inclusão de novos componentes e permitindo a ele incorporar novos desenvolvimentos obtidos pela pesquisa em escalonamento de processos. Sua flexibilidade é útil para um grande número de situações de escalonamento, de maneira que o PSMS, além de sua utilidade prática como auxiliar de escalonadores, também é uma ferramenta útil para auxiliar a pesquisa em escalonamento de processos de um modo geral.

7 Referências

[1] Batat, A.; Feitelson, D. (2000) Gang scheduling with memory considerations. Proceedings of the 14th Intl. Parallel Distributed Processing Symposium, pp. 109-114.

[2] Baumgartner, K.M., e Wah, B. W. (1991) "Computer Scheduling Algorithms: Past, Present and Future", Information Sciences, Elsevier Science Pub. Co., v.57, pp. 319-345.

[3] Beguelin, A., Geist, A., Dongarra, J., Jiang, W., Manchek, R., e Sunderam, V. (1994) "PVM: Parallel Virtual Machine. A User's Guide and Tutorial for Networked Parallel Computing", The MIT Press.

[4] Berman, F., Wolski, R., Casanova, H., Cirne, W., Dail, H., Faerman, M., Figueira, S., Hayes, J., Obertelli, G., Schopf, J., Shao, G., Smallen, S., Spring, S., Su, A., e Zagorodnov, D. (2003) "Adaptive Computing on the Grid Using AppLeS", IEEE Transactions on Parallel and Distributed Systems (TPDS), 14(4), pp. 369--382.

[5] Dumitrescu, C.L., e Foster, I. (2005) "GangSim: a simulator for grid scheduling studies", IEEE International Symposium on Cluster Computing and the Grid.

[6] Feilteson, D.G., e Rudolph, L. (1995) "Parallel Job Scheduling: Issues and Approaches", IPPS'95 Workshop on Job Scheduling Strategies for Parallel Processing, Lecture Notes in Computer Science v.949, April.

[7] Feiltelson, D.G., e Rudolph, L. (1996) "Toward Convergence in Job Schedulers for Parallel Supercomputers", IPPS' 96 Workshop on Job Scheduling Strategies for Parallel Processing, Lecture Notes in Computer Science v.1162, April.

[8] Feiltelson, D.G., e Rudolph, L. (1998) "Metrics and Benchmarking for Parallel Job Scheduling", Lecture Notes in Computer Science, n. 1459, p. 1-24.

[9] Feiltelson, D.G. (2001) "Metrics for Parallel Job Scheduling and their Convergence", Lecture Notes in Computer Science, n. 2221, p. 188-200.

[10] Ferrari, D., e Zhou, S. (1987) "An Empirical Investigation of Load Indices for Load Balancing Applications", Proceedings of Performance'87, the 12th Int'l Symposium on Computer Performance Modeling, Measurement and Evaluation, pp. 515-528.

[11] Gardner, M.K., Feng, W., Broxton, M., Engelhart, A., e Hurwitz, G. (2003) "MAGNeT: A Tool for Debugging, Analyzing and Adapting Computing Systems", Proc. Of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid.

[12] Giné, F.; Solsona, F.; Hernandez, P.; Luque, E. (2001) Coscheduling under memory constraints in a NOW environment. Lecture Notes on Computer Science, vol. 2221.

[13] Krallmann, J.; Schwigelshohn, U. (1999) On the design and evaluation of job scheduling algorithms. in. FEITELSON D.; RUDOLPH, L. eds. Job Scheduling Strategies for Parallel Processing, Springer-Verlag, pp. 17-42.

[14] Miller, B.P., Callaghan, M.D., Cargille, J.M., Hollingsworh, J.K., Irwin, R.B, Karavanic, K.L., Kunchithapadam, K., e Newhall, T. (1995) "The Paradyn Parallel Performance Measurements tools", IEEE Computer, November.

[15] Platform Computing Corporation. (2001) "LSF Administrator's Guide".

[16] Ribler, R.L., Simitci, H., e Reed, D.A. (2001) "The Autopilot Performance-Directed Adaptive Control System", Future Generation Computer Systems, Special Issue (Performance Data Mining), 18(1), September.

[17] Smith, W. (2001) "A framework for control and observation in distributed environments", Technical Report NAS-01-006, NASA.

[18] Sun Microsystems, Inc. (2005) "N1 Grid Engine 6 Administration Guide".

[19] Wolski, R. (2003) "Experiences with Predicting Resource Performance On-line in Computational Grid Settings", ACM SIGMETRICS Performance Evaluation Review, Volume 30, Number 4, pp 41--49, Março.

[20] Wong, F.C., Martin, R.P, Arpaci-Dusseau, R.H., e Culler, D.E. (1999) "Architectural requirements and scalability of the NAS parallel benchmarks", Proceedings of the 1999 ACM/IEEE conference on Supercomputing.

[21] Yao Z., Cui-ju L. e Guang-hua S. (2006) "Application-adaptive resource scheduling in a computational grid", Journal of Zhejiang University - Science A, vol. 7, n. 10, Outubro.

MÁRCIO AUGUSTO DE SOUZA
Possui graduação em Ciência da Computação pela Universidade Estadual Paulista Júlio de Mesquita Filho (1994), mestrado em Ciência da Computação e Matemática Computacional pela Universidade de São Paulo (1997) e doutorado em Ciência da Computação e Matemática Computacional pela Universidade de São Paulo (2004). Atualmente é professor adjunto da Universidade Estadual de Ponta Grossa. Tem experiência na área de Ciência da Computação, com ênfase em Computação Paralela Distribuída, atuando principalmente nos seguintes temas: avaliação de desempenho, computação paralela, escalonamento de processos e redes de computadores.

OMAR ANDRÉS CARMONA CORTEZ
Possui graduação em Ciência da Computação pela Universidade Federal do Maranhão (1997), mestrado em Ciências da Computação e Matemática Computacional pela Universidade de São Paulo (1999) e doutorado em Ciências da Computação e Matemática Computacional pela Universidade de São Paulo (2004). Atualmente é professor-doutor do Centro Federal de Educação Tecnológica do Maranhão. Tem experiência na área de Ciência da Computação, com ênfase em Benchmarks, atuando principalmente nos seguintes temas: computação paralela, inteligência artificial, lógica nebulosa, algoritmos evolutivos e estratégias evolutivas.

LUCIANO JOSÉ SENGER
Possui graduação em Bacharelado Em Informática pela Universidade Estadual de Ponta Grossa (1995), mestrado em Ciência da Computação e Matemática Computacional pelo Instituto de Ciências Matemáticas e de Computação (1997) e doutorado em Ciências de Computação e Matemática Computacional pelo Instituto de Ciências Matemáticas e de Computação (2005). Atualmente é professor adjunto da Universidade Estadual de Ponta Grossa. Tem experiência na área de Ciência da Computação, com ênfase em Computação Paralela Distribuída, atuando principalmente nos seguintes temas: computação paralela, redes de computadores, avaliação de desempenho e sistemas distribuídos.

REGINA HELENA CARLUCCI
Possui graduação em Engenharia Elétrica Eletrônica pela Escola de Engenharia de São Carlos

(1980), mestrado em Ciências de Computação pelo Instituto de Ciências Matemáticas de São Carlos (1985) e doutorado em Eletrônica e Computação - University of Southampton (1989). Atualmente é professor associado da Universidade de São Paulo. Tem experiência na área de Ciência da Computação, com ênfase em Avaliação de Desempenho, atuando principalmente nos seguintes temas: avaliação de desempenho, simulação, simulação distribuída, escalonamento de processos e computação paralela.

MARCOS JOSÉ SANTANA
Possui graduação em Engenharia Elétrica Eletrônica pela Escola de Engenharia de São Carlos (1980), mestrado em Ciências de Computação pelo Instituto de Ciências Matemáticas de São Carlos (1985) e doutorado em Eletrônica e Computação - University of Southampton (1989). Atualmente é professor associado da Universidade de São Paulo. Tem experiência na área de Ciência da Computação, com ênfase em Avaliação de Desempenho, atuando principalmente nos seguintes temas: avaliação de desempenho, escalonamento de processos, computação paralela, simulação e balanceamento de carga.