Pontifícia Universidade Católica de Minas Gerais
Grão-Chanceler: Dom Walmor de Oliveira Azevedo
Reitor: Dom Joaquim Giovani Mol Guimarães
Vice-reitora: Patrícia Bernardes

Pró-reitoria de Pesquisa e Pós-graduação: João Francisco de Abreu
Editora PUC Minas
Diretor: Geraldo Márcio Alves Guimarães
Coordenação editorial: Cláudia Teles de Menezes Teixeira
Assistente editorial: Maria Cristina Araújo Rabelo
Comercial: Maria Aparecida dos Santos Mitraud
Divulgação: Danielle de Freitas Mourão

Revisão: Maria Lina Soares Souza
Capa: Paulo Cruz - Assessoria de Publicidade
Diagramação: José Augusto Barros

Elaborada pela Biblioteca da Pontifícia Universidade Católica de Minas Gerais

P816i Pontifícia Universidade Católica de Minas Gerais.
Iniciação científica: destaques 2008 / Wolney Lobato, Cláudia de Vilhena Schayer Sabino, João Francisco de Abreu (Org.). - Belo Horizonte: Ed. PUC Minas, 2009
712 p.

Bibliografia
ISBN: 978-85-60778-47-8

1. Pesquisa - Belo Horizonte. I. Lobato, Wolney. II. Sabino, Cláudia de Vilhena Schayer. III. Abreu, João Francisco de. IV. Pontifícia Universidade Católica de Minas Gerais. V. Título.

CDU: 001.891

FAPEMIG CNPq

EDITORA PUC MINAS
Rua Pe. Pedro Evangelista, 377 – Coração Eucarístico
30535-490 – Belo Horizonte – MG – Brasil
Fone: (31) 3319.9904 – Fax: (31) 3319.9907
e-mail: editora@pucminas.br – www.pucminas.br/editora

Sumário

Análise de algoritmos de classificação e agrupamento na modelagem e predição de comportamentos em cargas de trabalhos de máquinas paralelas

Lesandro Ponciano dos Santos
João Paulo Domingos Silva
Luís Fabrício Wanderley Góes

Resumo

Neste trabalho, analisa-se a aplicação de algoritmos de classificação e agrupamento na modelagem e predição de comportamentos de tarefas paralelas. O estudo dedica-se à modelagem de comportamentos e à formação de uma base de informações a ser usada por um escalonador reconfigurável de tarefas. Na fundamentação teórica, é apresentada a análise dos principais conceitos relativos a escalonamento paralelo de tarefas, agrupamento e classificação de dados. Através da revisão bibliográfica, são destacados quatro trabalhos extraídos da literatura, que possuem alguma relação com o problema, o objetivo ou a proposta deste trabalho. Para análise e verificação dos experimentos, são realizadas implementações e análises comparativas entre o algoritmo de classificação *low* e *high*, atualmente utilizado no escalonador reconfigurável, e uma combinação dos algoritmos *k-means* e J48. Os testes são realizados com dados de processamento de quatro computadores reais, armazenados durante sete meses. Por fim, conclui-se que a combinação dos algoritmos *k-means* e J48 ocasionam, na média, maior ganho na predição do comportamento de tarefas paralelas.

Palavras-chave: Algoritmo de classificação; Algoritmos de agrupamento; Caracterização de cargas de trabalho; Escalonamento em arquiteturas paralelas.

Abstract

This paper analyzes the application of classification and clustering algorithms in modeling and prediction of parallel jobs behavior. The study proposes behavior modeling and the creation of an information supply to be used by reconfigurable job scheduling. Its theoretical basis consists of an analysis of the main concepts related to parallel jobs scheduling, data clustering and classification. The bibliography review points out four works concerning the issue. Two proposals are presented: Workload Modeling Technique with Classification and Clustering Algorithms, and Parallel Jobs Behavior Checking and Prediction Technique. The analysis and validation of the experiments were carried out through implementations and comparisons between low and high classification algorithm, currently used in reconfigurable scheduling, and a combination of k-means and J48 algorithms. The tests were made with processing data from four real computers, stored during seven months. The conclusion points out that the combination of k-means and J48 algorithms generates higher speedup in predicting parallel jobs behavior.

Key words: Classification algorithm; Clustering algorithms; Workload characterization; Scheduling in parallel architecture.

Introdução

Com a utilização cada vez mais frequente dos computadores nas diversas áreas de atuação humana, têm surgido aplicações complexas, como mapeamento de cadeias de DNA, simulações de cenários no mercado financeiro e cálculos de previsão do tempo, entre outras. Tais aplicações requerem, em geral, alto poder de processamento e realizam mais operações de entrada e saída (FEITELSON, 2002; CIRNE et al., 2007), o que tem exigido melhor desempenho dos recursos de computação.

Para suprir essa necessidade, alguns estudos em arquiteturas paralelas e distribuídas têm buscado a otimização de componentes de software e hardware, como processadores, memórias, redes de comunicação, protocolos de rede, etc. (GÓES; MARTINS, 2004). Dentre esses componentes, destaca-se o escalonador de tarefas. Ele é responsável por escalonar os processos de uma tarefa para os processadores disponíveis na arquitetura paralela.

O processo de escalonamento pode ser realizado com (GÓES; MARTINS, 2004) ou sem (CIRNE et al., 2007) informação sobre a carga de trabalho (conjunto de tarefas). No primeiro caso, as informações das tarefas servem de base para definir políticas de escalonamento adequadas a cada carga de trabalho (GÓES; MARTINS, 2004). No segundo caso, utiliza-se uma mesma política de escalonamento (CIRNE et al., 2007), independentemente da carga de trabalho.

Um dos problemas na utilização de escalonadores com informação é a obtenção de informação confiável: se o usuário fornece informações erradas, o desempenho do escalonador fica comprometido (LEE et al., 2004). Uma possível solução para esse problema é a caracterização de rastros (históricos de execução de tarefas), isto é, a construção de um modelo de desempenho das tarefas a ser utilizado como informação confiável para o escalonador (FEITELSON, 2002).

Neste trabalho, destaca-se o escalonador com informação Reconfigurable Gang Scheduling Algorithm (RGSA). Esse escalonador utiliza as seguintes informações sobre as tarefas: (i) intervalo de chegada; (ii) tempo de execução; (iii) número de processos. A partir dessas informações, ele define e aplica as melhores políticas de escalonamento para a carga de trabalho. No entanto, sua aplicação em ambiente real ainda esbarra na falta de mecanismos eficazes de obtenção e modelagem das informações sobre a carga de trabalho.

Um grupo de algoritmos que pode ser aplicado na modelagem de cargas de trabalho são os algoritmos de agrupamento. Agrupamento[1] é um método de organização e análise de dados, geralmente aplicado na etapa de mineração de dados,[2] no processo de obtenção de conhecimento de bases de dados (Knowledge Discovery in Databases – KDD) (FAYYAD; SHAPIRO; SMYTH, 1996).

Uma observação a ser feita é que este trabalho de diplomação visa contribuir com um projeto maior – a implementação e teste do escalonador RGSA em ambiente real. Trata-se, pois, de um trabalho complementar àquele. Sua contribuição ao projeto maior consiste na

1. Agrupamento, do inglês *Clustering*.
2. Mineração de dados, do inglês *Data Mining*.

análise da aplicação de algoritmos de agrupamento no processo de caracterização de carga de trabalho para esse escalonador. Propõe-se aqui a modelagem de comportamentos de cargas de trabalho por meio de algoritmos de agrupamento e a utilização desse modelo como informação confiável para o escalonador.

O RGSA foi proposto por Góes e Martins (2004), que realizaram testes com cargas sintéticas. Um dos problemas para a verificação da proposta em ambiente real é a falta de uma técnica de caracterização capaz de extrair, de rastros de computadores paralelos reais, modelos com as informações confiáveis de que o RGSA necessita.

Santos e Góes (2007) propõem a caracterização de cargas para o escalonador RGSA baseada em um método que classifica as tarefas, pelo tempo de execução e número de processos, em H (high - alto) e L (low - baixo), utilizando a mediana como divisor dessas classes. H e L são invariáveis e não há prova formal de que elas sejam a melhor forma de classificar as tarefas, tendo ocorrido situações em que o algoritmo não obteve resultados satisfatórios.

Este trabalho propõe a utilização de algoritmos de agrupamento das tarefas, os quais, pelas suas similaridades, permitiriam gerar grupos dinâmicos, que variam de acordo com as características da carga de trabalho, o que possibilitaria uma melhor classificação das tarefas dessa carga de trabalho. Seu principal objetivo é, pois, aplicar algoritmos de agrupamento na modelagem de tarefas de cargas de trabalho de máquinas paralelas, para melhorar o desempenho das arquiteturas paralelas. Com esse processo de caracterização, as informações extraídas são agrupadas em um modelo de ocupação pelo tempo (tempo de execução) e espaço (números de processos) do sistema caracterizado. Por meio dessas informações, o escalonador RGSA pode identificar e aplicar a melhor política de escalonamento para os grupos específicos de tarefas submetidas ao sistema.

A monografia está organizada da seguinte forma: na seção 2, faz-se a fundamentação teórica, em que são definidos e analisados os principais conceitos relacionados à área em estudo; na seção 3, é apresentada a revisão bibliográfica e feita a apresentação e análise dos principais trabalhos relacionados; na seção 4, é proposta uma técnica de modelagem de tarefas paralelas por meio de algoritmos de classificação e de agrupamento; na seção 5, são apresentados os materiais e os métodos; na seção 6, são analisados os resultados obtidos nos experimentos; por fim, na seção 7, faz-se a conclusão do trabalho.

Fundamentação teórica

As arquiteturas paralelas são sistemas computacionais compostos de várias unidades de processamento e, geralmente, classificados em multiprocessador e multicomputador (HENNESSY; PATTERSON, 2003; TANENBAUM, 2001). Os multiprocessadores possuem vários processadores que compartilham uma mesma memória e utilizam barramento como sistema local de interconexão, daí serem chamados de "fortemente acoplados" (TANENBAUM, 2001). Como exemplo de multiprocessadores, pode-se citar a arquitetura de multiprocessadores simétricos (SMP). Já os multicomputadores possuem vários nodos de processamento, cada um com sua memória e são ditos fracamente acoplados, uma vez que utilizam um sistema de redes para interconexão (TANENBAUM, 2001). Como exemplo de multicomputadores, pode-se citar as redes de estações de trabalho (NOWs) ou aglomerados de computadores.

Outro modelo de multicomputação é a Grade Computacional (*Grid*). Grade é uma arquitetura com dispersão geográfica (distribuída), heterogeneidade de *software* e dispositivos de *hardware*. Ela recebe a submissão de aplicações, mapeia-as e escalona-as entre os dispositivos de processamento disponíveis – cada dispositivo possui seu escalonador local que cuida do escalonamento das tarefas dessa aplicação (TANENBAUM, 2001). Uma Grade Computacional pode ser formada, por exemplo, de SMPs e diversos outros tipos de computadores, paralelos ou não, conectados através da *internet* visando compartilhar seus potenciais de processamento e armazenamento (FOSTER *et al.*, 1999).

Em um sistema em grade, o escalonador de aplicações é responsável por decidir, por exemplo, as aplicações que obterão acesso aos recursos computacionais, a quantidade de recursos destinados a cada aplicação e a localização desses recursos. Essas decisões são afetadas por diferentes fatores, como carga de trabalho do sistema, diversidade de aplicações e necessidades dos usuários.

Uma aplicação paralela é composta de várias tarefas (*jobs*). Define-se carga de trabalho como um conjunto de tarefas executadas em uma arquitetura paralela, geralmente expressa em um rastro (*logs* ou históricos de execução de tarefas) (GÓES; MARTINS, 2004). As tarefas são constituídas de processos e descrevem os passos para solucionar um problema. As tarefas que compõem uma aplicação paralela são executadas simultaneamente em várias unidades de processamento.

As informações da carga de trabalho podem ser fornecidas pelo usuário, mas para assegurar a confiabilidade da informação – fundamental para um bom escalonamento (JAIN, 1991) –, é mais seguro gerar modelos de predições através da caracterização de rastros.

Quando se utiliza a caracterização para obter informações de uma carga de trabalho, geralmente, trabalha-se com grandes bases de dados. Em um problema prático, é necessário reduzir o volume desses dados, o que pode ser feito gerando um modelo que preserve as características mais relevantes. Jain (1991) mostra que, através da caracterização de cargas de trabalho, é possível construir modelos de sistemas e utilizá-los como informação na melhoria de desempenho.

Os modelos de cargas de trabalho apresentam muitas vantagens em relação às cargas de trabalho real. Uma delas é que podem ser usados para simular um sistema real e realizar análises e desenvolver sistemas melhorados. Em um projeto de caracterização, os parâmetros que representarão a carga de trabalho variam com a finalidade do estudo Jain (1991).

Uma das técnicas de caracterização é o agrupamento, que, em mineração de dados, consiste basicamente em reunir um conjunto de dados por suas características naturais, de modo que os mais similares fiquem no mesmo grupo e os menos similares fiquem em grupos distintos (WITTEN; FRANK, 2005; LAROSE, 2006). Desse modo, todo elemento de um grupo possui maior similaridade com outro elemento do mesmo grupo do que com qualquer elemento de outro grupo (TAN; STEINBACH; KUMAR, 2006).

No processo de agrupamento, a busca da melhor solução no conjunto de soluções viáveis é de complexidade de tempo exponencial $O(2^n)$, uma vez que consiste na avaliação exaustiva de todas as configurações de agrupamentos possíveis (TAN; STEINBACH; KUMAR, 2006). Desse modo, soluções heurísticas têm sido propostas a fim de obter soluções aproximadas com baixa ordem de complexidade. As heurísticas existentes podem ser divididas em hierárquicas e de particionamento (TAN; STEINBACH; KUMAR, 2006).

Nos algoritmos tradicionais de agrupamento hierárquico, os grupos são formados gradativamente através de aglomerações ou divisões de elementos, o que gera uma hierarquia de grupos, normalmente representada por uma árvore (WITTEN; FRANK, 2005). Essa classe de algoritmo pode apresentar abordagem de aglomeração (*bottom-up*) ou divisão (*top-down*).

Na abordagem de aglomeração (*bottom-up*), cada elemento do conjunto é inicialmente associado a um grupo distinto, e novos grupos são formados pela união dos grupos existentes. Essa união ocorre de acordo com alguma medida que forneça informações sobre a similaridade entre eles. Já na abordagem de divisão (*top-down*), inicialmente tem-se um único grupo contendo todos os elementos do conjunto e, a cada passo, são efetuadas divisões formando grupos de tamanhos menores, mas mantendo a similaridade entre seus elementos.

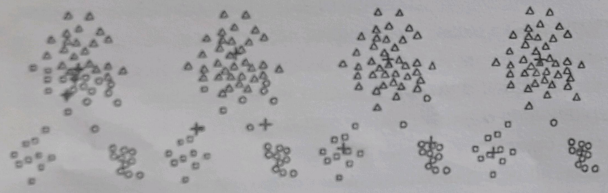
Nos algoritmos de agrupamento por particionamento, o conjunto de elementos é dividido em k subconjuntos, podendo k ser conhecido ou não, e cada configuração obtida é avaliada através de uma função-objetivo (WITTEN; FRANK, 2005; TAN; STEINBACH; KUMAR, 2006). Caso a avaliação do agrupamento indique que a configuração não atende ao problema em questão, uma nova configuração é obtida por meio da migração de elementos entre os grupos, e o

processo continua de forma iterativa até que algum critério de parada seja alcançado. Esse processo é conhecido como otimização iterativa (WITTEN; FRANK, 2005; TAN; STEINBACH; KUMAR, 2006).

O *k-means* é um algoritmo de agrupamento por particionamento em que o elemento representativo de um grupo é o centroide. O centroide possui um valor médio para os atributos considerados, relativo a todos os elementos do grupo (TAN; STEINBACH; KUMAR, 2006). A técnica *k-means* é amplamente utilizada na literatura, na busca de soluções para agrupamento. No algoritmo 1, são apresentados os principais passos de execução do algoritmo *k-means* e, na Figura 1, é apresentado um exemplo de 4 iterações do algoritmo para um conjunto de dados arbitrário.

ALGORITMO 1 - Visão geral do algoritmo *k-means* (adaptado de TAN; STEINBACH; KUMAR, 2006).

1: Selecione k pontos para centroides iniciais.
2: Faça.
3: Forme k grupos e nomeie para cada grupo um centroide mais representativo.
4: Recalcule o centroide de cada grupo.
5: Enquanto centroides mudarem.



(a) Iteração 1 (b) Iteração 2 (c) Iteração 3 (d) Iteração 4

Figura 1 - Mudança dos centroides e definição dos *clusters* no algoritmo *k-means*. Fonte: Adaptado de Tan, Steinbach e Kumar (2006).

Outra técnica de caracterização é a classificação. Classificação é o método de mineração de dados que consiste em uma função de aprendizado que mapeia os dados segundo classes predefinidas

(FAYYAD; SHAPIRO; SMYTH, 1996). Nesse método, um dos atributos que compõe a base de dados é utilizado como classificador, portanto há que analisar qual dos atributos melhor representa o registro. O processo de classificação não é exato, algum nível de diferença deve ser tolerado (HAN; KANBER, 2006).

O processo de classificação é bastante complexo, porque consiste em identificar uma característica ou grupo de características que melhor descreve um conjunto de dados. Nesse contexto, podem-se empregar diversas técnicas ou algoritmos como: classificação por indução de regras ou árvores de decisão, redes bayesianas, redes neuronais e algoritmos genéticos, entre outros.

Revisão bibliográfica

Neste estudo, são destacados os seguintes trabalhos: Góes e Martins (2004), Feitelson (2002) e Santos e Góes (2007).

Góes e Martins (2004) propõem e desenvolvem o algoritmo reconfigurável de escalonamento paralelo de tarefas ou RGSA (*Reconfigurable Gang Scheduling Algorithm*). O objetivo do RGSA é obter um escalonamento de tarefas capaz de se adaptar às variações das arquiteturas paralelas e às cargas de trabalho. Para alcançar esse objetivo, o RGSA necessita de informações confiáveis sobre as tarefas da carga de trabalho. Este trabalho é complementar ao de Góes e Martins (2004) e tem como objetivo obter modelos de comportamentos de tarefas a serem usados na obtenção de informações para o escalonador RGSA.

Um fato importante a ser considerado na análise e modelagem de comportamentos de cargas de trabalho é que uma carga pode ser modelada em distribuições significativamente diferentes das que seriam observadas em intervalos mais longos; em locais diferentes, essas distribuições também diferem uma das outras (FEITELSON, 2002).

Essa constatação baseia-se na análise dos desvios máximo e médio das distribuições em relação a intervalos de tempo e rastros dife-

rentes (FEITELSON, 2002). O estudo foi norteado pelo objetivo de mensurar variações e gerar modelos simplificados representativos de cargas reais. Os resultados indicam que, em uma carga de trabalho, existem dados que apresentam razoável semelhança entre si se observados em pequenos intervalos de tempo e diferenças maiores se observados em períodos mais longos. Isso indica que ocorrem variações na carga de trabalho ao longo do tempo.

Essas observações motivam o desenvolvimento de sistemas adaptáveis, capazes de identificar períodos de semelhança e variações em uma carga de trabalho e de se ajustar a ela na medida em que tais comportamentos são identificados (FEITELSON, 2002). Para melhorar o desempenho do escalonador de tarefas, é importante construir modelos de informação que lhe permitam adaptar-se às mudanças de comportamento da carga de trabalho.

Santos e Góes (2007) propõem uma técnica de caracterização de cargas de trabalho capaz de extrair, de um rastro, modelos de carga de trabalho com as informações de que o escalonador RGSA necessita. Em seu trabalho, são analisadas 66.560 tarefas, execuções do ano de 1995 do rastro LANL-CM5-1994-3. Como conclusão, é apresentada a eficácia da técnica proposta na geração de modelos representativos das tarefas executadas nos períodos analisados.

A técnica de caracterização proposta por Santos e Góes (2007) extrai modelos de comportamentos de rastros de supercomputadores. Uma das características da técnica é utilizar o tempo de submissão, tempo de execução e o número de processos como únicas informações de cada tarefa da carga de trabalho. No processo de caracterização, as tarefas são classificadas, quanto ao tempo de execução e quanto ao número de processos, em H (alto, do inglês *high*) e L (baixo, do inglês *low*), o que gera quatro combinações possíveis (HH, HL, LL e LH), como é apresentado na Figura 2. Neste trabalho, essa técnica de caracterização é chamada de algoritmo *low* e *high*.

A classificação das tarefas é muito importante no processo de caracterização da carga de trabalho, porque consiste no primeiro processo de identificação de semelhanças a que as tarefas são submetidas. Uma classificação ruim *reflete negativamente em toda a caracterização. Em síntese, o algoritmo low e high pressupõe a existência*, em uma carga de trabalho, de quatro classes de tarefas cujos limites são definidos pela mediana do tempo de execução e do número de processos, como apresentado na Figura 3. Essa classificação desencadeia os demais processos da caracterização.

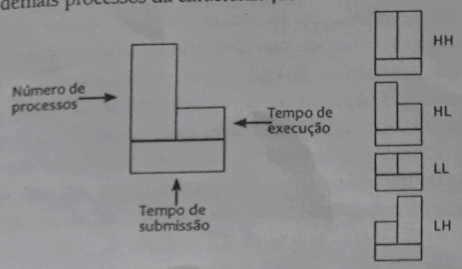


FIGURA 2 - Representação gráfica da classificação das tarefas – Algoritmo *low* e *high*. Fonte: Baseado em Santos e Góes (2006).

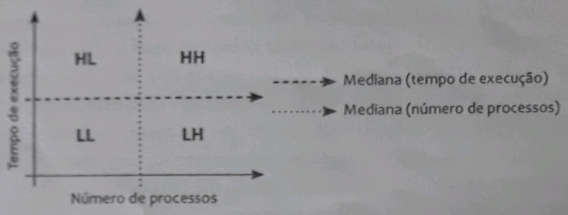


Figura 3 - Representação da classificação através da mediana (Algoritmo *low* e *high*). Fonte: Dados da pesquisa.

Nota-se, pela Figura 3, que o algoritmo *low* e *high* considera a dispersão dos dados, mas não se preocupa em minimizar os desvios entre os elementos de uma mesma classe. Além disso, o fato de basear-se em apenas duas retas lineares pode trazer problema. Na situação, por exemplo, em que todas as tarefas têm o mesmo número de processos, mas diferentes tempos de execução, o algoritmo só terá

H ou L número de processos, o que indica que, necessariamente, uma classe ficará com 0 (zero) elemento. Nesse caso, os algoritmos de agrupamento por similaridade tenderiam a desconsiderar o atributo número de processos e a gerar quatro grupos, não vazios, considerando, apenas, similaridades no tempo de execução.

Diferentemente do que propõem Santos e Góes (2007), este trabalho propõe a análise de algoritmos de agrupamento para definição dos grupos de classificação das tarefas em substituição às classes predefinidas H e L. Os algoritmos de agrupamento têm a vantagem de gerar classes de acordo com as características naturais da carga (similaridade), o que dinamiza o processo de classificação de tarefas.

Modelagem de comportamentos de tarefas paralelas por meio de algoritmos de agrupamento e classificação

Nesta seção, é apresentada uma técnica de modelagem de tarefas paralelas para formação de uma base de informações a ser utilizada pelo escalonador RGSA. A proposta, em linhas gerais, consiste em adaptar a técnica de Santos e Góes (2007) para que outros algoritmos, além da classificação Low e High, possam ser utilizados no processo.

O processo de modelagem pode ser descrito em sete estágios, norteados, em alguns, pelas fases do processo de KDD. O processo inicia-se no estágio em que se dispõe dos dados brutos e termina com a formação da base de informações.

Inicialmente, tem-se um rastro (ou *log*). Os rastros consistem em um conjunto de dados sobre as tarefas executadas em uma arquitetura paralela, ao longo do tempo. Geralmente, esses rastros possuem propósito geral, por isso são constituídos de diversas informações sobre tarefas, memória, comunicação entre processos, entre outras. Nas cargas de trabalho, em razão de possíveis falhas ou erros no processo de coleta e/ou armazenamento, podem ocorrer dados inconsistentes, duplicados, valores em branco, entre outros problemas. Assim, faz-se necessária a limpeza dos dados antes de seu processamento.

O segundo estágio do processo consiste na limpeza dos dados. Nesse estágio, são eliminados dados redundantes, com valores faltosos ou zerados e inconsistências como, por exemplo, marcação de tempo descontínuo. Uma vez dada consistência aos dados, é necessário selecionar os dados mais relevantes para a análise.

O terceiro estágio consiste na seleção dos dados, entendida como a eliminação de atributos não relevantes e eliminação de *outliers*. Segundo Góes e Martins (2004), os atributos mais relevantes são tempo de submissão, tempo de execução e o número de processos, portanto são esses os atributos selecionados e utilizados nos demais estágios do processo. Os limiares que definem se um dado é ou não um *outlier* é fornecido como parâmetro ao algoritmo de seleção de dados. Uma vez selecionados os dados, é necessário identificar padrões de similaridade entre eles, o que é feito utilizando-se algoritmos de agrupamento.

O quarto estágio consiste no agrupamento dos dados. No agrupamento, podem-se utilizar diversos algoritmos clássicos, como, por exemplo, o *k-means* ou *k-medoids*. Nesse estágio, é criado um novo atributo para a base de dados chamado Grupo D, e cada registro é relacionado a um dos grupos G0, G1, G2 ou G3 dependendo do que for processado pelo algoritmo de agrupamento. Uma vez realizado o agrupamento, processam-se outros dois estágios, não sequenciais e não excludentes: processamento estatístico e detecção de regras.

O quinto estágio é o de processamento estatístico e consiste na apuração das medidas de tendência central – média, mediana e moda – e das medidas de dispersão – desvio padrão e coeficiente de variação. Essas medidas ajudam a identificar tendências de comportamentos, como assimetria, covariância, e a verificar se há tendência à distribuição gaussiana ou à de Pareto, entre outras. Depois de apurados, esses valores são armazenados no banco (ou base) de informações.

O sexto estágio é o de detecção de regras. Ele consiste no processamento dos grupos de dados para identificar uma árvore de decisão (ou conjunto de regras) gerada em função do atributo Grupo D. Essas

regras são armazenadas na base de informações. Além disso, processa-se também o estágio de classificação em que se geram classes de tarefas em função das regras obtidas no estágio de detecção de regras.

O conceito de base de informações utilizado neste trabalho refere-se a um conjunto de dados estatísticos, regras geradas por agrupamento e classes geradas por classificação obtidas ao longo de todo o processo de modelagem e detecção de comportamentos, como apresentado no Quadro 1. A estrutura de informação utilizada pelo RGSA, atualmente, é formada pelos atributos destacados com sombreado no Quadro 1. Os demais dados são utilizados na constatação de mudanças no comportamento e na análise de predição.

Quadro 1
Exemplo da estrutura de uma base de informações em modelagem de comportamento de tarefas paralelas

Atributos armazenados	Regras armazenadas (exemplo)
Identificação do período, Ex.: mês.	
Número de tarefas submetidas	
Número mínimo de processos	
Número médio de processos	
Número máximo de processos	
Desvio padrão do número de processos	se job.n_processos <= 64
Moda do número de processos	então job.classe = G1;
Mediana do número de processos	senão
Tempo de execução mínimo	se job.n_processos <= 256
Tempo de execução médio	então job.classe = G0;
Tempo de execução máximo	senão
Desvio padrão do tempo de execução	se job.tempo_exe <= 8357
Moda do tempo de execução	então job.classe = G2;
Mediana do tempo de execução	senão job.classe = G3;
Percentual de tarefas do Grupo G0	
Percentual de tarefas do Grupo G1	
Percentual de tarefas do Grupo G2	
Percentual de tarefas do Grupo G3	

Fonte: Dados da pesquisa.

Depois de geradas e armazenadas as regras, são apurados e armazenados os valores dos atributos. Nessa segunda etapa, as tarefas são agrupadas pelo dia da semana e hora do dia em que foram submetidas (Quadro 2).

Quadro 2
Agrupamento pelo dia da semana e hora do dia

		Dia da Semana
Horas do Dia	0	
	1	
	2	
	...	
	22	
	23	

Fonte: Dados da pesquisa.

Na terceira etapa, calcula-se a frequência de tarefas de cada classe, identificando o dia da semana e a hora do dia em que foram submetidas. Neste trabalho, propõe-se estimar a probabilidade de ocorrência de tarefas, de cada classe e em cada hora do dia, utilizando-se o método de aproximação pela frequência, como apresentado na Equação 1, em que *i* é uma hora do dia.

$$P(Classe)_i = \left(\frac{Frequência\_Tarefas\_Classe_i}{Total\_Tarefas_i} \right) * 100$$

Materiais e métodos

No processo de agrupamento (*k-means*) e classificação (J48), utilizou-se a ferramenta de mineração de dados Weka (*Waikato Environment for Knowledge Analysis*) versão 3.5.11. Weka[3] é uma ferramenta *OpenSource* desenvolvida pela Universidade de Waikato. Sua escolha se justifica por dois aspectos: (i) Weka implementa os principais algoritmos de agrupamento de dados (particionamento e aglomeração); (ii) permite a entrada e a saída de dados padronizadas em formato de *log*, o que facilita o processo de comparação dos resultados gerados pelos algoritmos em análise.

Neste trabalho, desenvolveu-se uma ferramenta de processamento em lote para integrar todo o processo descrito na seção 4. Ela executa as seguintes funções: (i) Acessos ao rastro fonte (log) e à base

3. www.cs.waikato.ac.nz/ml/weka

de informações; *(ii)* Limpeza dos dados; *(iii)* Geração de arquivos de entradas de dados para a ferramenta Weka e leitura das saídas geradas; *(iv)* Algoritmo de classificação *low* e *high*. Desenvolveu-se também uma ferramenta de verificação e predição de comportamentos como proposta na seção 5.

As linguagens de programação utilizadas são C++ e Perl.[4] C++ é uma linguagem compilada que suporta o conceito de orientação por objetos, o que facilita a modelagem e implementação das entidades do sistema e facilita a reutilização de código. Utilizou-se a linguagem C++ no formato *Ansi* para que o código-fonte seja portável para todos os compiladores da linguagem. *Perl* é uma linguagem de *scripts*, interpretada, que possui funções específicas de acesso ao *hardware* e manipulação de arquivos, de modo a obter melhor desempenho computacional.

Selecionou-se a ferramenta Dev-C++[5] versão 4.9.9.2 para compilação dos algoritmos em linguagem C++. Dev-C++ é uma ferramenta de distribuição gratuita. Implementações em *Perl* foram interpretadas pelo interpretador *ActivePerl* 5.8.8.2 *Build* 822. *Perl* também é uma ferramenta de distribuição gratuita. Os testes foram realizados em um computador Pentium 4, 1.7 GHz, 983 MHz com 1 GB de memória RAM, *hardware* gerenciado pelo sistema operacional *Microsoft Windows XP Professional* Versão 2002, *Service Pack* 2.

As análises dos resultados deste trabalho são fundadas principalmente em duas métricas: a semelhança e o *speedup* (ou ganho). A semelhança é, em termos percentuais, o quanto um modelo de tarefas é próximo de outro. Neste trabalho, essa métrica é usada para comparar os modelos de dados e os modelos de predição. O *speedup*, por sua vez, é o ganho obtido ao usar um método ou algoritmo em relação a outro. Neste trabalho, o *speedup* será usado para avaliar o ganho obtido na variação do algoritmo de classificação *low* e *high* em relação ao processo *k-means* e J48.

Nos experimentos realizados neste trabalho, utilizaram-se quatro rastros de máquinas paralelas reais: *(i) The San Diego Supercomputer Center Data Star* (SDSC-DS), do computador paralelo IBM p655/p690, instalado no *San Diego Supercomputer Center (SDSC) Data Star*. 1,664 CPUs; (ii) rastro SHARCNET, localizado em Ontário, Canadá. 6,828 CPUs; *(iii)* rastro *High-Performance Computing Center North* (HPS2N), localizado na Suécia. 240 CPUs; *(iv) The Los Alamos National Lab – Connection Machine* (LANL-CM5) 1,024 CPUs.

4. www.activestate.com
5. www.bloodshed.net/devcpp.html

No processo de caracterização e predição, utilizam-se períodos mensais: gera-se o modelo de comportamentos de tarefas de um mês e ele é utilizado na predição do comportamento de tarefas do mês subsequente. Na Tabela 1, são apresentados os números de tarefas de cada carga, em cada período.

Tabela 1
Tarefas caracterizadas por rastro e períodos

Rastro	Ano	Número de tarefas						
		Abr.	Mai.	Jun.	Jul.	Ago.	Set.	Out.
HPC2N	2003	29970	17817	20545	24121	7241	5159	11444
SDSC	2004	5259	8395	8358	5339	7204	8782	9147
SHARCNET	2005	87505	76813	52551	95118	90225	63162	158681
LANLCM5	1995	5790	6831	6216	6969	6126	6303	5271

Fonte: Dados da pesquisa.

Apresentação e análise dos resultados

Uma análise estatística dos modelos gerados consiste em identificar medidas de tendência central e medidas de dispersão no conjunto de dados. Duas medidas de tendência central requerem maior atenção: a mediana, por ser utilizada como *threshold* pelo algoritmo *Low* e *High*, e a média, por ser utilizada em análise de comportamento linear.

Na Tabela 2, é apresentada a mediana do número de processos. A análise da mediana ao longo dos períodos indica que, no que tange à distribuição dos valores, as cargas SDSC e SHARCNET só apresentam variação no mês de outubro, permanecendo a mesma nos demais meses. Este é um fator importante, porque indica que as tarefas, quando submetidas ao algoritmo de classificação *low* e *high*,

apresentaram bons resultados uma vez que a estrutura desse algoritmo vai se manter ao longo dos períodos.

Tabela 2
Mediana do número de processos

	Períodos						
Cargas	Abr.	Mai.	Jun.	Jul.	Ago.	Set.	Oct.
HPC2N	2	4	2	2	2	2	8
SDSC	8	8	8	8	8	8	32
SHARCNET	1	1	1	1	1	1	1
LANLCM5	32	32	64	32	32	64	64

Fonte: Dados da pesquisa.

Ainda de acordo com a Tabela 2, a carga HPC2N apresenta variação da mediana em 2, 4 e 8, mas 2 é o valor mais frequente e aparece em quatro meses consecutivos, o que é muito importante, uma vez que o processo de predição é realizado sempre de um mês para outro. Já a carga LANLCM5 apresenta duas variações da mediana em 32 e 64, mas esses valores se mantêm apenas de dois em dois meses, o que pode causar maus resultados, uma vez que os algoritmos trabalham com padrões mensais, e não bimestrais.

Como o intervalo de valores frequentes para o tempo de execução é maior (geralmente de 1 a 10.000) que o do número de processos (geralmente de 1 a 512), é menos provável que as medianas coincidam de um mês para outro, mas é desejada uma pequena diferença para que se percebam semelhanças de comportamento. Na Tabela 3, são apresentados os valores obtidos para mediana do tempo de execução.

Tabela 3
Mediana do tempo de execução

	Períodos						
Rastros	Abr.	Mai.	Jun.	Jul.	Ago.	Set.	Oct.
HPC2N	2448	2498	2479	3028	3351	3629	3359
SDSC	647	423	214	334	992.5	1115	754
SHARCNET	88	85	22215	6484	1305	1201	1463
LANLCM5	226.5	252	394	377	255	606	359

Fonte: Dados da pesquisa.

Pelos resultados da Tabela 3, nota-se que os rastros apresentam comportamentos bem variáveis: as maiores dispersões são percebidas na carga SHARCNET, mas as demais também apresentam dispersões elevadas. Isso pode ser um indicativo de que o algoritmo *low* e *high* pode perder precisão ao utilizar a mediana do tempo de execução de um mês para prever o comportamento das tarefas do mês subsequente.

Nos testes de predição de comportamentos, caracterizaram-se os dados de um mês e armazenaram-se as regras, dados estatísticos e classes obtidas. Em um segundo momento, extraíram-se do rastro os dados das tarefas executadas no mês subsequente ao caracterizado e calculou-se a semelhança desses dados com as informações obtidas na caracterização. Nesse processo obteve-se, conforme as Tabelas 8 e 9, a caracterização realizada pelo Algoritmo *k-means* e J48 gerou maior semelhança em 95,24% das observações do que a caracterização realizada pelo algoritmo *low* e *high*. Destacou-se, na Tabela 9, a borda dos casos em que o algoritmo *k-means* e J48 apresentou resultado inferior.

Tabela 4
Algoritmo *low* e *high* – Semelhança na predição - SDSC – 2004

Mês de caracterização	Mês de predição	Semelhança em função dos dias da semana						
		D	S	T	Q	Q	S	S
Abr.	Mai.	54	46	52	52	62	48	34
Mai.	jun.	60	54	62	60	56	52	68
jun.	jul.	64	68	60	62	46	46	62
jul.	Ago.	62	62	60	64	64	66	60
Ago.	set.	66	62	66	72	64	58	72
set.	out.	72	74	64	70	70	68	68

Fonte: Dados da pesquisa.

No Gráfico 1, é apresentado o ganho médio do algoritmo *k-means* e J48 em relação ao *low* e *high*. Nota-se que ocorreu maior ganho (19,68%) no rastro SHARCNET. Esse rastro possui maior número de tarefas e apresentou menor variação da mediana, média e desvio padrão do número de processos, entre os meses. Por outro lado, o pior resultado foi obtido no rastro LANLCM5, em que ocorreu maior variação dessas medidas.

Tabela 5
Algoritmo *k-means* e J48 – Semelhança na Predição - SDSC – 2004

Mês de caracterização	Mês de predição	Semelhança em função dos dias da semana						
		D	S	T	Q	Q	S	S
Abr.	Mai.	68	62	68	68	66	64	64
Mai.	Jun.	74	68	64	68	74	60	72
Jun.	Jul.	82	82	82	82	74	76	82
Jul.	Ago.	68	70	74	70	62	60	68
Ago.	set.	76	74	80	82	82	80	88
set.	out.	82	86	84	84	82	84	86

Fonte: Dados da pesquisa.

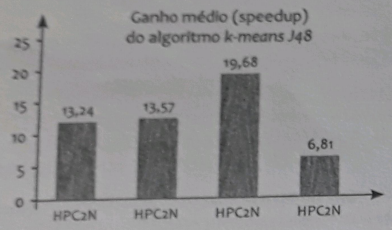


GRÁFICO 1 - Ganho médio obtido na utilização do algoritmo *k-means* e J48 em relação ao algoritmo *low* e *high*

No Gráfico 2, pode-se observar o percentual de acertos obtidos na predição com o algoritmo *k-means* e J48 e com o algoritmo *low* e *high*, no mês de outubro, com o rastro SHARCNET. Percebe-se que o algoritmo *k-means* e J48, na terça-feira, foi pouco melhor que o *low* e *high*, em relação aos outros dias. Isso ocorre porque nesse dia há maior variação entre as tarefas, o que dificulta o agrupamento pelo algoritmo *k-means*.

Conclusão

Nesta seção, são discutidos os principais resultados obtidos e destacadas as principais contribuições para a área de modelagem e predição de tarefas.

Os resultados obtidos neste trabalho atendem ao objetivo. Eles indicam que, através dos algoritmos *k-means* e J48, é possível gerar modelos de cargas de trabalho capazes de assegurar maior acerto no

processo de predição de um período futuro, quando comparado com o algoritmo *low* e *high*. Outra observação importante é que a modelagem em função do dia da semana, e com predição de mês para mês, no geral mostrou-se bastante satisfatória e obteve em muitos casos modelos com até 80% de semelhança. As principais contribuições deste trabalho são a técnica proposta e a constatação de que as tarefas executadas nos sistemas paralelos possuem um padrão no número de processos e no tempo de execução e que esse padrão se repete em intervalos mensais, semanais e diários, e também ao longo das horas do dia.

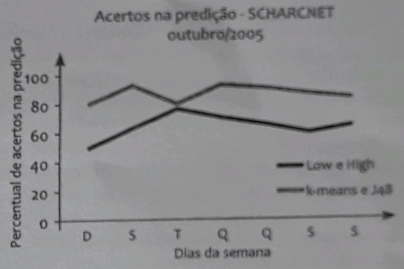


GRÁFICO 2 - Ganho médio obtido na utilização do algoritmo *k-means* e J48 em relação ao algoritmo *low* e *high*

Referências

CIRNE, W. *et al*. On the efficacy, efficiency and emergent behavior of task replication in large distributed systems. *Parallel Computing*, v. 33, n. 3, p. 213-234, 2007.

FAYYAD, U.; SHAPIRO, G. P.; SMYTH, P. Data mining to knowledge discovery in databases. AI Magazine, v. 17, n. 3, p. 37-54, 1996.

FEITELSON, D. Workload modeling for performance evaluation, performance evaluation of complex systems: techniques and tools. In: CALZAROSSA, Mariacarla; TUCCI, Salvatore (Ed.). *Performance evaluation of complex systems*: techniques and tools. New York: Springer Verlag, 2002. p. 114-141.

FOSTER, Ian; KESSELMAN, Carl. *et al*. *The grid*: blueprint for a new computing infrastructure. San Francisco: Morgan Kaufmann, 1999.

GÓES, L. F. W.; MARTINS, C. A. P. S. Reconfigurable gang scheduling algorithm. In: INTERNATIONAL WORKSHOP, 10, New York, 2004. *Job scheduling strategies for parallel processing, lecture notes in computer science*. New York: Etats-Unis, 2004.

HAN, Jiawei; KANBER, Micheline. *Data mining*: concepts and techniques. 2. ed. San Francisco: Elservier, 2006.

HENNESSY, John L.; PATTERSON, David A. *Computer architecture*: a quantitative approach. 3.ed. San Francisco: Morgan Kaufmann, 2003.

JAIN, Raj. *The art of computer systems performance analysis*: techniques for experimental design, measurement, simulation, and modeling. New York: J. Wiley, c1991.

LAROSE, Daniel T. *Data mining methods and models*. New Jersey: John Wiley & Sons, 2006.

LEE, C. B. *et al. Are user runtime estimates inherently inaccurate*? 10th Workshop on Job Scheduling Strategies for Parallel Processing, 2004. Disponível em: www.cs.huji.ac.il/~feit/ parsched/jsspp04/p-04-2.pdf.

MARTINS, C. A. P. S. *et al. Computação reconfigurável*: conceitos, tendências e aplicações. In: JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA, 22, CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 23, 2003.

SANTOS, Lesandro P.; GÓES, Luís F. Técnica de caracterização de cargas de trabalho para extração de informações utilizadas pelo escalonador reconfigurável de tarefas. In: WORKSHOP DE SISTEMAS COMPUTACIONAIS DE ALTO DESEMPENHO (WSCAD), 2007, Gramado. *19 th Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. Gramado: WSCAD, 2007.

TAN, Pang-Ning; STEINBACH, Michael; KUMAR, Vipin. *Introduction to data mining*. 3. ed. Boston: Pearson Addison Wesley, 2006.

TANENBAUM, Andrew S. *Modern operating systems*. 2. ed. New Jersey: Prentice-Hall, 2001.

THE SAN DIEGO Supercomputer Center (SDSC) Blue Horizion log. Disponível em: <http://www.cs.huji.ac.il/labs/parallel/workload/logs.html>. Acesso em: 30 dez. 2007.

WITTEN, I. H.; FRANK, E. (2005). *Data Mining*: Practical machine learning tools and techniques. 2. ed. San Francisco: Morgan Kaufmann, 2005.

Uma proposta de arquitetura de *software* para um provedor de serviços interativos em televisão digital

Gabriel Massote Prado
João Benedito dos Santos Junior

Resumo

Neste capítulo, é apresentado o trabalho – desenvolvido no Laboratório de Televisão Digital da PUC Minas em Poços de Caldas – de especificação e implementação de estratégias para construção de um protótipo de plataforma necessária à implantação de um Provedor de Serviços Interativos (PSI), que possa armazenar, analisar e gerar relatórios baseados nas informações resultantes da interação de telespectadores com aplicações em Televisão Digital Interativa, caracterizadas pelo uso de um canal de retorno via IP (Internet Protocol). Ao apresentar um protótipo de plataforma PSI, esta proposta usufrui das experiências acumuladas com o desenvolvimento da plataforma JiTV (Java Interactive Television), que contempla a produção na emissora, a transmissão, a recepção no terminal de acesso do telespectador e o uso do canal de retorno para interatividade plena.

Palavras-chave: Televisão Digital Interativa; Canal de retorno; Interatividade; Tecnologia Java; Tecnologia XML.