Modelo de Memória Reconfigurável para Sistemas Paralelos

Dulcinéia O. da Penha[1], Henrique C. de Freitas[2], Carlos A. P. S. Martins[3]
Laboratório de Sistemas Digitais e Computacionais[1,2,3], Instituto de Informática[2,3], Programa de Pós-Graduação em Engenharia Elétrica[1,3]
Pontifícia Universidade Católica de Minas Gerais
dulcineia@pucmg.br[1] cota@pucminas.br[2] capsm@pucminas.br[3]

Resumo

Arquiteturas paralelas atuais possuem modelo de memória único e estático. Entretanto, as cargas de trabalho de um sistema computacional possuem características distintas e às vezes até divergentes. Conseqüentemente, a escolha de um modelo de memória ideal é difícil e envolve custo, desempenho, disponibilidade, entre outros fatores. Neste trabalho, propomos um modelo de memória reconfigurável para sistemas computacionais paralelos chamado RMA (Reconfigurable Memory Access). Este modelo de memória visa flexibilidade e adaptabilidade na utilização de sistemas de memória de computadores paralelos. Para verificação e análise do modelo proposto, modelamos uma Rede de Petri para o modelo PRAM (Parallel Random Access Machine), uma para o MP-RAM (Message-Passing Random Access Machine) e outra para o RMA. Modelamos dois tipos de cargas de trabalho e simulamos em cada uma das Redes. Analisamos e comparamos os resultados, usando equações matemáticas (modelos analíticos). A partir dos resultados obtidos concluímos que a utilização do modelo RMA em sistemas paralelos proporciona uma melhor adequação do modelo de memória (espaço de endereçamento) do sistema ao modelo de acesso à memória de cada carga de trabalho. Além disso, traz ganho de desempenho na execução de cada aplicação e também ganho de desempenho médio do sistema.

1. Introdução

O funcionamento e o desempenho dos elementos de processamento (EPs) de um sistema paralelo é fortemente relacionado com o seu modelo de memória e com as características de acesso à memória da aplicação paralela executada no sistema. Um modelo computacional é uma descrição lógica ou conceitual de um computador, abstraindo aspectos de tecnologia e hardware [1]. A maioria dos modelos computacionais, analíticos ou operacionais, é descrita considerando, além de outros aspectos e elementos, o modelo ou comportamento da memória e/ou a arquitetura de memória.

Cada aplicação normalmente é desenvolvida baseada em um modelo único e estático, de acesso à memória. Diferentes classes de aplicações possuem diferentes características de acesso à memória que são inerentes à lógica de solução do problema e ao algoritmo da solução. Estas características indicam o modelo de memória mais adequado para cada classe de aplicação. Entretanto, não existe um modelo de memória ideal que atenda a todas as classes de aplicações do ponto de vista de adequação, desempenho e eficiência. Isso acontece porque normalmente um sistema possui cargas de trabalho com características distintas e algumas vezes até divergentes. Além disso, certas aplicações possuem características que fazem com que o modelo de memória influencie significativamente em seu desempenho.

O **problema** motivador da nossa pesquisa é, então, o fato de não existir um modelo de memória único, ideal, para todas as aplicações já que cada classe de aplicação pode possuir um modelo de acesso à memória mais adequado às suas características, em termos de desempenho.

A nossa proposta de **solução** para este problema é um Modelo de Memória Reconfigurável, denominado RMA (*Reconfigurable Memory Access*). Um sistema paralelo implementando o modelo proposto permite a execução de aplicações com diferentes modelos de memória, através da reconfiguração do comportamento do seu sistema de memória. O nosso **objetivo**, neste artigo, é apresentar a versão analítica do Modelo RMA e os resultados obtidos através da simulação do modelo.

Este modelo de memória reconfigurável tem um comportamento múltiplo e dinâmico, e pode ser configurado de acordo com o modelo de acesso à memória da carga de trabalho em execução. Com isso, objetivamos uma melhor adequação do modelo de memória (espaço de endereçamento) com o modelo de acesso à memória de cada carga de trabalho, ganho de desempenho na execução de cada aplicação e também melhor desempenho médio do sistema.

A Computação Reconfigurável [3] é um novo e revolucionário paradigma. Seus conceitos podem aumentar a flexibilidade, adaptabilidade e desempenho dos sistemas [4] [5]. Um sistema reconfigurável é composto por blocos construtivos interconectados. O comportamento do sistema é definido pela topologia de interconexão entre estes blocos e pelo funcionamento lógico deles. Então, o

comportamento do sistema pode mudar dinamicamente de acordo com sua forma [5].

A versão analítica do RMA foi modelada através de Redes de Petri [6] e equações matemáticas (modelagem analítica) e foi verificada e analisada através de simulação da Rede de Petri modelada.

Desenvolvemos uma Rede de Petri para o modelo PRAM, uma para o MP-RAM e outra para o RMA. Modelamos dois tipos de cargas de trabalho e simulamos em cada uma das Redes. Comparamos e analisamos os resultados, usando equações matemáticas (modelagem analítica).

2. Modelos de Memória

A maioria dos modelos computacionais, analíticos ou operacionais, é descrita considerando o modelo ou comportamento da memória, a arquitetura de memória e/ou as características de acesso à memória. Por exemplo, os modelos analíticos RAM (*Random Access Machine*), PRAM, e MP-RAM [1] são descritos considerando o número de elementos de processamento e o comportamento da memória. Máquinas RAM e PRAM possuem uma única memória. No caso de máquinas PRAM, como existe mais de um elemento de processamento, essa memória única é compartilhada entre os EPs. A máquinas PRAM possuem ainda as variantes EREW (*Exclusive Read Exclusive Write*), CREW (*Concurrent Read Exclusive Write*), CRCW (*Concurrent Read Concurrent Write*), e ERCW (*Exclusive Read Concurrent Write*). Estas variantes são regras para resolver conflitos de acesso à memória, uma vez que ela é compartilhada com todas os EPs do sistema. Já as máquinas com modelo MP-RAM, além de possuírem mais de um EP, possuem memórias privativas de cada EP, não compartilhadas com os outros EPs, e uma topologia de interconexão entre os EPs. Neste caso, a comunicação entre os elementos de processamento é realizada através de passagem de mensagem.

Sob a visão operacional, vários modelos computacionais também consideram o comportamento da memória, como é o caso dos modelos UMA (*Uniform Memory Access*), NUMA (*Non Uniform Memory Access*), COMA (*Cache Only Memory Access*) e NORMA (*NO Remote Memory Access*) [2]. Em máquinas UMA, todos os acessos à memória são realizados gastando uma mesma quantidade de tempo para todos os processadores. Já em máquinas NUMA os acessos à memória não precisam gastar o mesmo tempo para todos os processadores. Modelos NUMA podem ser divididos em NC-NUMA (*Non-Coherent Cache* NUMA) e CC-NUMA (*Coherent Cache* NUMA). Em máquinas COMA, todos os endereços de memória são estruturados como caches. Já máquinas NORMA são compostas por múltiplos nodos e memórias locais privadas, e não existe espaço de endereçamento compartilhado. A comunicação entre os processos é realizada através de passagem de mensagens. Os outros modelos (UMA, NUMA e COMA) precisam de hardware especial para manter um espaço de endereçamento único e compartilhado, habilitando todos os processadores a acessar qualquer parte da memória. Entretanto, os mecanismos de acesso à memória destes sistemas são distintos.

Cada um destes modelos descritos anteriormente tem características únicas e comportamento estático. Além disso, nenhum deles é ideal para todas as aplicações, já que cada classe de aplicação possui um modelo de acesso à memória mais adequado às suas características, em termos de desempenho. Isto significa que um determinado modelo de memória é ideal para alguma aplicação, mas trará perda de desempenho para outras que possuam características diferentes de acesso à memória. Algumas aplicações nem poderiam ser executadas sobre sistemas com determinados modelos de memória.

A nossa proposta de solução para o problema levantado é um modelo de memória reconfigurável, com comportamento múltiplo e dinâmico. Este modelo torna o sistema mais flexível e permite a execução de diferentes tipos (classes) de aplicações obtendo um melhor desempenho médio do sistema.

3. Trabalhos Correlatos

No estudo do estado da arte, não encontramos trabalhos correlatos que sigam o mesmo caminho de solução que propomos para o problema da adequação do modelo de memória do sistema paralelo ao modelo de acesso à memória da aplicação. Todos os trabalhos analisados abordam problemas parecidos, mas propõem soluções em caminhos diferentes. Apresentamos aqui alguns destes trabalhos e descrevemos brevemente suas propostas de solução.

Ide [7] propõe uma arquitetura de memória que suporta programação com variáveis compartilhadas em aglomerado de computadores. A arquitetura provê memória compartilhada entre os nodos do sistema através de placas reconfiguráveis e software de passagem de mensagem. Os autores propõem a implementação de uma arquitetura reconfigurável, restrita para aglomerados e específica para redes neurais. A principal diferença da nossa proposta de solução é que propomos um modelo de memória reconfigurável para computadores paralelos, que permite a execução de diferentes tipos de aplicações, visando ganho de desempenho.

Dreier [8] propõe Rthreads, um sistema de software de memória compartilhada-distribuída que suporta compartilhamento de variáveis globais em aglomerados de computadores com memória distribuída fisicamente. Os autores propõem uma implementação em software para prover um ambiente de memória compartilhada virtualmente sobre um ambiente de memória distribuída fisicamente. É uma proposta semelhante aos sistemas DSMs (*Distributed-*

Shared Memory). Assim como a proposta de Ide, esta é executada sobre um modelo de memória fixo (memória distribuída).

Outros ambientes DSM são propostos em [9] [10] [11] [12], entre outros. Ramanjuan [13] apresenta uma nova abordagem para aglomerados de computadores para processamento paralelo. Este trabalho é baseado na idéia de que interconexões com alta velocidade podem formar a base de um novo caminho para a construção de um computador paralelo com memória compartilhada a partir de um aglomerado de computadores. Neste sistema a rede implementa a memória compartilhada, *Network Shared Memory*. Para a implementação deste sistema, cada estação de trabalho tem um cartão de interface (NMI – *Network Memory Interface*), que é usado para a interconexão dos elementos de processamento das estações do computador paralelo. Os NMIs juntos implementam a rede com memória compartilhada, ou seja, o espaço de endereçamento coerente de todos os processadores do aglomerado. Conceitualmente a rede com memória compartilhada consiste de um conjunto de bancos de memória implementados dentro dos NMIs que são conectados juntos em um anel por *links* de alta velocidade de comunicação. Esta arquitetura possui modelo de memória estático.

4. Modelo de Memória Reconfigurável

O modelo analítico de memória RMA (*Reconfigurable Memory Access*) pode ser reconfigurado de acordo com as características da carga de trabalho em execução para comportar-se de acordo com as características de memória de um modelo PRAM, MP-RAM, ou outros modelos existentes. Ou seja, o RMA pode ser configurado para mapear cada um destes modelos ou ainda para mapear outros modelos existentes e suas combinações.

Se considerarmos principalmente o modelo ou comportamento de memória dos modelos de computadores PRAM e MP-RAM, estes modelos podem ser representados como apresentado nas Figuras 1 (a) e 1 (b), respectivamente.

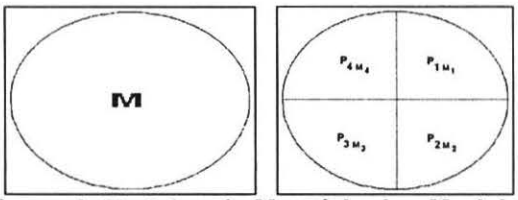

Figura 1. Modelos de Memória dos Modelos de Computadores.(a) PRAM (b) MP-RAM.

O modelo PRAM é constituído por uma única memória, compartilhada entre todos os EPs. O MP-RAM é constituído pelas memórias privadas dos EPs, e a sua topologia de interconexão é representada pela linha de divisão entre os processadores e suas respectivas memórias.

Como podemos observar nas Figura 1 (a) e 1 (b), a ligação ou interconexão entre os processadores é realizada através de uma memória compartilhada (modelo PRAM) ou através de uma rede de interconexão (modelo MP-RAM).

Baseado nestes dois modelos, apresentamos o modelo RMA Analítico, na Figura 2. A linha de divisão central representa o Bloco (ou mecanismo) de Reconfiguração do sistema, que pode ser reconfigurado para tornar o comportamento da memória como um sistema PRAM ou MP-RAM e adaptando-o às características da carga de trabalho em execução. Além disso, este Bloco de Reconfiguração é composto por Blocos de Gerenciamento de Memória em cada máquina do sistema. Os modelos PRAM e MP-RAM apresentam comportamento estático, enquanto o RMA é dinâmico, podendo ser reconfigurado de acordo com as características da carga de trabalho.

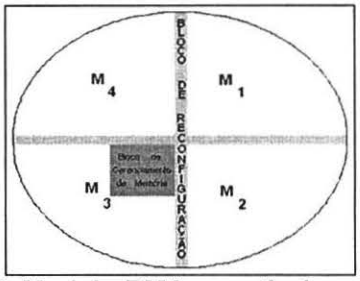

Figura 2. Modelo RMA a partir dos modelos de memória de PRAM e MP-RAM.

5. Modelagem através de Redes de Petri

Existem três técnicas bem conhecidas de verificação de sistemas: modelagem analítica, simulação e medição [14]. Experimentos reais sempre precisam de um ambiente real e os ambientes de execução e testes podem ser muito caros e até mesmo não existirem, como é o caso deste trabalho.

Por este motivo e pelo fato de que propomos uma versão analítica do RMA, a verificação do modelo será realizada através de Redes de Petri e equações matemáticas (modelagem analítica) e ainda através de simulação da Rede de Petri.

A Figura 3 apresenta a Rede de Petri que modela um sistema PRAM EREW com dois processadores (EPs) após a execução de cargas de trabalho com características diferentes. Apesar de as cargas de trabalho serem diferentes (número de acessos locais e remotos diferentes), o resultado da simulação para ambas foi o mesmo já que no modelo PRAM, acessos locais e remotos são tratados igualmente. A Rede de Petri inicial possui um *Token* ou Ficha no *Place* ou Lugar "Memória Disponível" e nos "Geradores de Acessos" de cada processador. Neste sistema, cada processador tem acesso à memória compartilhada do sistema.

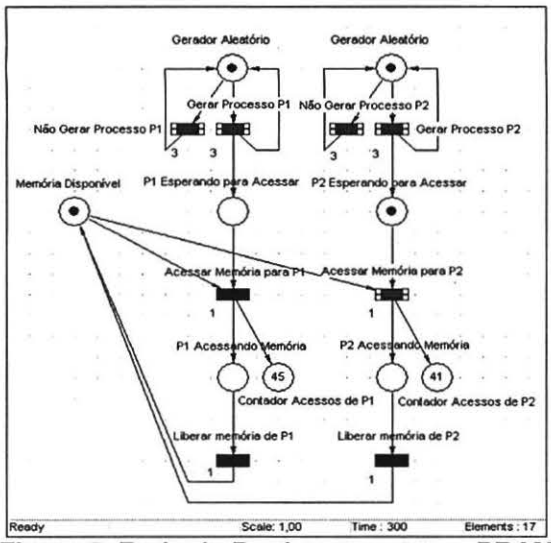

Figura 3. Rede de Petri que modela o PRAM após a execução de duas cargas de trabalho diferentes CT1 e CT2.

Todo acesso a qualquer posição da memória é simples, realizado através de operações de leitura e escrita. Cada acesso gerado em um processador vai para uma fila e só precisa que a memória esteja disponível para ser realizado.

O Lugar "Memória Disponível" controla o acesso à memória. Neste modelo, acessos à memória são exclusivos (EREW). Deste modo, o Lugar "Memória Disponível" possui apenas uma Ficha. As características de acesso à memória da carga de trabalho do sistema são geradas (e controladas) através de geradores de acessos. Contadores de acessos são utilizados para análise dos resultados, comparações e conclusões.

As Figuras 4 e 5 apresentam a Rede de Petri que modela um sistema MP-RAM com duas máquinas (EPs) após a execução de duas cargas de trabalho com características diferentes. Apresentamos as figuras das execuções das duas cargas de trabalho para facilitar a visualização do número de acessos, já que a Rede de Petri do modelo é complexa. A Rede de Petri inicial possui um *Token* ou Ficha no *Place* ou Lugar "Memória Disponível" e nos "Geradores de Acessos" de cada processador (de acordo com a carga de trabalho desejada). Neste sistema, cada máquina pode acessar somente a sua memória local privada. Acessos a dados remotos são realizados através de passagem de mensagem. Acessos locais são simples, realizados através de operações de leitura e escrita.

Cada acesso local gerado vai direto para a fila (de acessos) da memória. Cada acesso remoto gerado é enviado para a fila de mensagens da máquina remota e desta para a fila de acessos (ainda da máquina remota). Após acessar o dado, a máquina remota envia uma resposta para a máquina local que requisitou o dado. Esta resposta vai para a fila de acessos locais da máquina que originou o acesso. Após este acesso local, a passagem de mensagem foi concluída. A Rede de Petri do modelo MP-RAM não modela o meio de interconexão. Ela se limita a modelar apenas o acesso à memória. A influência do meio de interconexão é considerada através da Equação 2, que será explicada adiante. A escolha de não modelar o atraso (ou caminho) do meio de interconexão através de Lugares e Transições deve-se ao tamanho e complexidade do modelo.

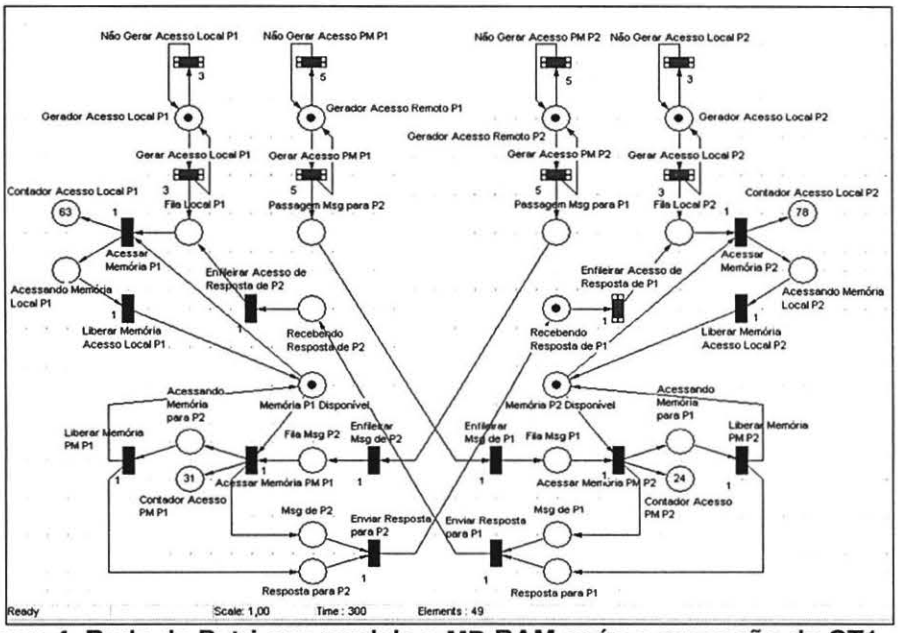

Figura 4. Rede de Petri que modela o MP-RAM após a execução de CT1.

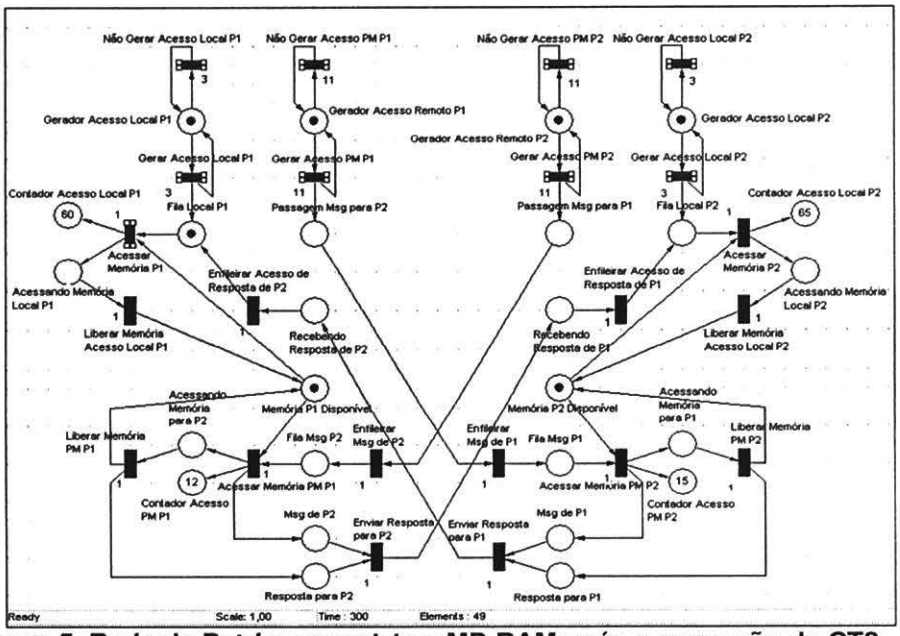

Figura 5. Rede de Petri que modela o MP-RAM após a execução de CT2.

Ou seja, a modelagem do meio de interconexão no MP-RAM traz a necessidade de modelar o meio de acesso dos acessos locais e daqueles realizados através do RMA. A modelagem do meio de acesso ou de interconexão não auxilia na descrição da proposta, uma vez que não influencia ou interfere no modelo ou na forma de acesso. A influência no desempenho foi modelada através da atribuição de pesos para cada um dos modelos.

Assim como no modelo PRAM, as características de acesso à memória da carga de trabalho do sistema são geradas (e controladas) através de geradores de acessos. Contadores de acessos remotos mostram o número de passagens de mensagens realizadas e contadores de acessos locais mostram o número de acessos locais. No entanto, o número de acessos locais gerados pelos processos em execução na máquina é obtido a partir da subtração do número de passagens de mensagens do total de acessos local. Isto é feito porque cada mensagem recebida de outra máquina requisitando um dado gera um acesso local na máquina remota para então responder à mensagem. O número de acessos locais gerados por processos locais pode ser calculado através da Equação 1, onde AL é o número de acessos locais gerados por processos locais, CAL (Contador Acesso Local) é o número total de acessos locais realizados (gerados por processos locais e por acessos para passagem de mensagem), e CAPM (Contador Acesso PM) é o número de passagem de mensagens realizadas.

$$AL\ P_i = CAL\ P_i - CAPM\ P_j \qquad \text{(Eq1.)}$$

As Figuras 6 e 7 apresentam a Rede de Petri que modela o RMA com duas máquinas (EPs) após a execução de duas cargas de trabalho com características diferentes. Como no modelo MP-RAM, apresentamos as figuras das execuções das duas cargas de trabalho para facilitar a visualização do número de acessos (informação quantitativa), já que a Rede de Petri do modelo é complexa. A Rede de Petri inicial possui um *Token* ou Ficha no *Place* ou Lugar "Memória Disponível" e nos "Geradores de Acessos" de cada processador (de acordo com a carga de trabalho desejada). Neste modelo o sistema pode ser reconfigurado para comportar-se como um sistema PRAM (com variáveis compartilhadas) ou MP-RAM (com passagem de mensagem). O acesso à memória local acontece como no modelo MP-RAM apresentado anteriormente.

Se o sistema for configurado para comportar-se como MP-RAM, cada EP tem acesso direto à sua memória local e usa passagem de mensagem para acessar dados remotos. Neste caso, o sistema comporta-se como na Rede de Petri do modelo MP-RAM, conforme descrito anteriormente. Se o sistema for configurado para comportar-se como PRAM, acessos locais são realizados diretamente. No entanto, como a estrutura da memória do RMA é distribuída, acessos remotos não podem ser realizados diretamente.

Desta forma, um mecanismo do modelo RMA realizará o compartilhamento da memória entre os EPs. Entretanto, espera-se que o *overhead* acrescentado por esse compartilhamento seja significativamente menor do que o de passagem de mensagem. Isto porque o compartilhamento deve ser realizado no nível da arquitetura do hardware e não precisará percorrer muitas camadas de protocolo e software.

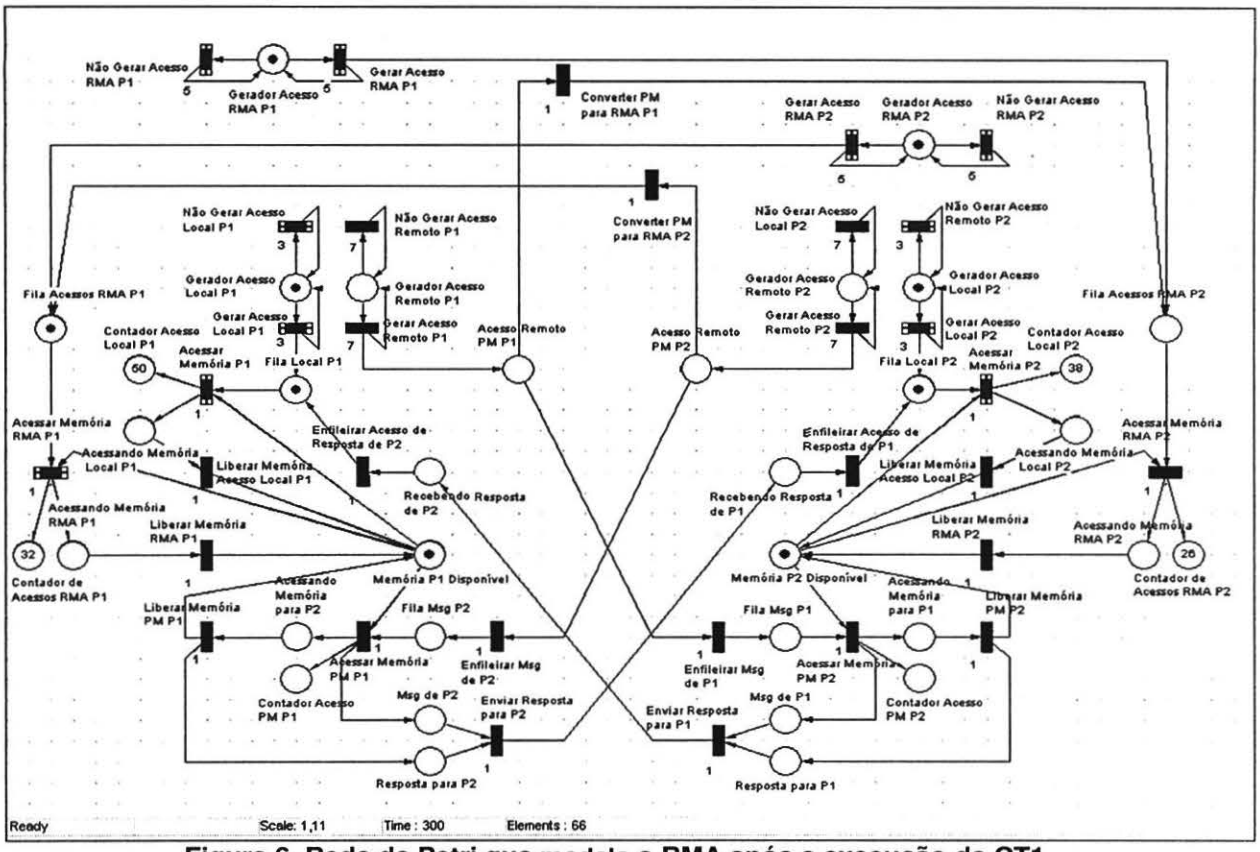

Figura 6. Rede de Petri que modela o RMA após a execução de CT1.

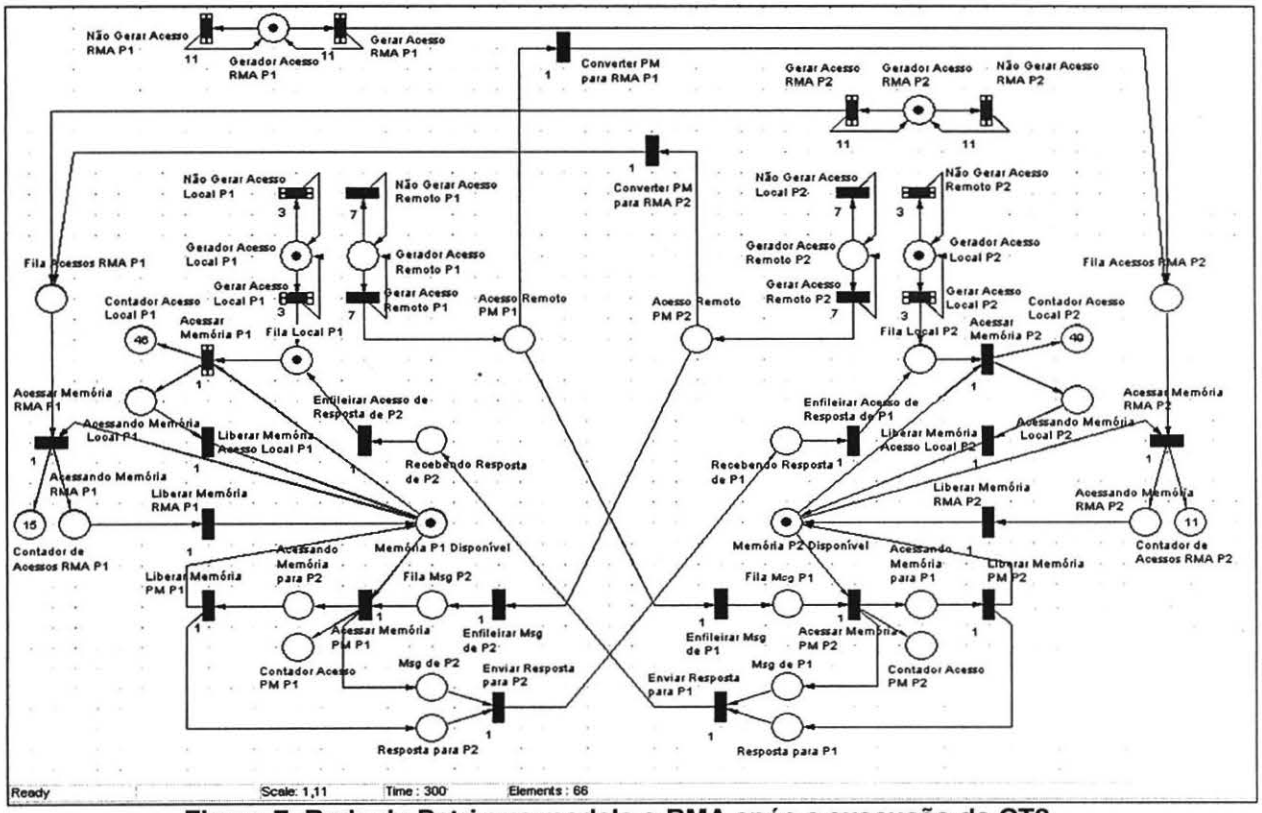

Figura 7. Rede de Petri que modela o RMA após a execução de CT2.

O sistema pode ainda ser configurado para ser heterogêneo, ou seja, parte da memória compartilhada (utilizando o mecanismo do RMA) e parte distribuída (memória física do sistema). Assim como nos modelos PRAM e MP-RAM, as características de acesso à memória da carga de trabalho do sistema são geradas (e controladas) através de geradores de acessos. Contadores de acessos remotos mostram o número de passagens de mensagens realizadas, contadores de acesso RMA mostram o número de acessos realizados através do mecanismo do RMA e contadores de acessos locais mostram o número de acessos locais. O número de acessos locais gerados pelos processos em execução na máquina é obtido como no modelo MP-RAM, a partir da Equação 1.

6. Resultados Experimentais

Os resultados experimentais foram obtidos através da modelagem usando Redes de Petri de cada um dos modelos PRAM, MP-RAM e RMA. Modelamos dois tipos de cargas de trabalho (CT) com características diferentes e simulamos em cada uma das Redes usando o simulador Visual Object Net [15]. Escolhemos este simulador devido às suas características: facilidade de projeto e simulação rápida de Redes de Petri. Além disso, este é um simulador gratuito e com características de animação que tornam mais fácil a observação do comportamento dinâmico das redes. Comparamos e analisamos os resultados obtidos, usando equações matemáticas (modelos analíticos).

Na modelagem das cargas de trabalho, acessos locais são aqueles acessos realizados na memória local mesmo quando a carga é executada em máquinas com modelo MP-RAM. Do mesmo modo, acessos remotos são aqueles que quando a carga é executada em máquinas MP-RAM são realizados em memória remota.

As cargas de trabalho para a realização dos testes possuem grau de paralelismo alto. Além disso, são caracterizadas por granularidade e demanda por comunicação. A Carga de Trabalho 1 (CT1) possui granularidade fina e demanda por comunicação alta em relação ao processamento (predominantemente acessos remotos). Já CT2 possui granularidade grossa e demanda por comunicação baixa (predominantemente acessos locais).

Para simular estas cargas nas Redes de Petri de cada modelo, são atribuídos tempos (unidades de tempo) a cada tipo de gerador de acesso, que funcionam como taxas de geração de acessos. Como podemos observar na Figura 3, do modelo PRAM, os tempos de geração de acessos são iguais a 3 em todos os geradores, tanto para a CT1 quanto para a CT2. Este tempo pode ser explicado pelo fato de que, em máquinas PRAM, qualquer acesso local ou remoto será realizado na mesma memória comum, compartilhada entre os dois processadores.

Já no caso do modelo MP-RAM e do RMA, os tempos dos geradores são diferentes para caracterizar cada tipo de acesso de cada uma das cargas de trabalho. Como podemos observar na Figura 6, do modelo MP-RAM para CT1, o tempo de geração de acessos locais é 3, e o de acessos remotos é 5, devido à grande demanda por comunicação. Entretanto, para a CT2, que possui pouca demanda por comunicação, o tempo de geração de acessos remotos é 11.

Um sistema como os das Figuras 6 e 7, que utiliza o modelo RMA, pode configurar sua memória através do mecanismo de reconfiguração para comportar-se como distribuída (MP-RAM) ou compartilhada (PRAM). No caso de reconfiguração para compartilhamento atribuímos um *overhead* de comunicação de peso 3, sendo que acessos locais possuem peso 1. Acessos através de passagem de mensagem exigem diferentes acessos à memórias locais e tráfego na rede comum do sistema, e por isso atribuímos peso 5.

Portanto, no caso de CT1 e CT2, que possuem acessos remotos, para uma melhor utilização do sistema, a memória será reconfigurada para ser compartilhada entre os dois EPs. Portanto, em nenhum dos dois casos haverá acessos por passagem de mensagem. Desta forma, não há fichas no Gerador de Acessos PM de nenhum dos EPs. Todo acesso remoto será realizado através da reconfiguração da memória pelo RMA. Para CT1, o tempo de geração de acessos locais é 3 e o de acessos remotos através do RMA é 5. Para CT2, o tempo para locais é 3, e para remotos é 11. Cada simulação foi realizada por 300 unidades de tempo (ut), correspondentes ao tempo de transição na simulação das Redes de Petri.

Para cada modelo o tempo total de acesso é dado pela soma do número de acessos de cada tipo multiplicado pelo peso. O tempo total de acessos do sistema é calculado através da Equação 2, onde TTAS é o tempo total de acesso do sistema. A Tabela 1 mostra o tempo total de acesso de cada modelo para cada carga de trabalho, baseado na Equação 2.

$$TTAS = \sum N.\ de\ Acessos_i * Peso_i \qquad \text{(Eq2)}$$

Tabela 1. Tempo Total de Acesso do Sistema

Tempo Total de Acesso (ut)	Modelos					
	PRAM		MP-RAM		RMA	
	CT1	CT2	CT1	CT2	CT1	CT2
	86	86	416	257	262	173

Analisando as características das cargas de trabalho e os tempos totais de acesso apresentados na Tabela 1, percebemos que o modelo de memória mais adequado (aquele que traz melhores resultados em termos de desempenho) é o modelo PRAM. Isto porque este modelo não introduz *overheads* de comunicação já que os processos se comunicam através de memória compartilhada.

No caso de CT1, que possui muita comunicação, o uso de uma máquina MP-RAM traz perda de

desempenho significativa. Já o uso do RMA alcança um speedup de cerca de 1,9217 em relação ao MP-RAM. Como sistemas de memória compartilhada muitas vezes são difíceis de projetar e construir e não são escaláveis, o uso do modelo RMA é vantajoso. Já para CT2, o uso de MP-RAM não degrada tanto o desempenho, uma vez que a aplicação não tem muita comunicação.

Entretanto, como sistemas paralelos possuem cargas de trabalho com características diferentes, o uso do modelo RMA traz ganho de desempenho médio do sistema.

7. Conclusões

O objetivo do trabalho foi alcançado através da proposta e apresentação do modelo analítico de memória reconfigurável RMA, da simulação usando Redes de Petri e da análise dos resultados obtidos.

A partir da análise dos resultados concluímos que a utilização do modelo RMA em sistemas paralelos proporcionará uma melhor adequação do modelo de memória (espaço de endereçamento) do sistema com o modelo de acesso à memória de cada carga de trabalho. Além disso, trará otimização e ganho de desempenho na execução de cada aplicação e também ganho de desempenho médio do sistema.

Como trabalhos futuros, pretendemos realizar simulações com outras cargas de trabalho com características diferentes. Além disso, estamos desenvolvendo Redes de Petri escaláveis para os modelos de memória. Pretendemos ainda realizar a simulação dos modelos usando Redes de Petri Estocásticas e Coloridas, atribuindo funções de taxa de geração de acessos. O desenvolvimento da versão operacional do modelo RMA, onde consideramos o desenvolvimento mais detalhado dos meios de acesso e interconexão e os parâmetros temporais mais importantes, está em andamento e auxiliará na implementação real do modelo RMA em computadores paralelos.

8. Agradecimentos

Agradecemos à Pro-reitoria de Pesquisa e de Pós-graduação PUC-Minas (ProPPG), Programa de Pós-Graduação em Eng. Elétrica (PPGEE), Lab. de Sistemas Digitais e Computacionais (LSDC), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), PUC-Minas pelo suporte `a nossa pesquisa e infra-estrutura para os experimentos.

9. Referências

[1] Almasi, G. S., e A.A. Gottlieb, "Highly Parallel Computing", 2nd. Edition, Benjamim/Cummings, 1994.

[2] Hwang, K., e Z. Xu, "Scalable Parallel Computing: Technology, Architecture, Programming", McGraw-Hill, 1998.

[3] Martins, C. A. P. S., E. D. M. Ordonez, J. B. T. Corrêa e M. B. Carvalho, "Computação Reconfigurável: conceitos, tendências e aplicações", Jornada de Atualização em Informática 2003, Congresso da Sociedade Brasileira de Computação, Capítulo 8, 2003.

[4] K. Compton , e S. Hauck, "Reconfigurable Computing: A Survey of Systems and Software", In ACM Computing Surveys, June 2002, vol. 34:2, pp. 171-210.

[5] A. Dehon, "The Density Advantage of Configurable Computing", IEEE Computer Society, April 2000, vol. 33, No. 4, pp. 41-49.

[6] G., Bressan, "Modelagem e Simulação de Sistemas Computacionais", LARC-PCS/EPUSP 2002, www.larc.usp.br/conteudo/universo/pcs012/modsim05.pdf

[7] A. N. Ide, e J. H. Saito, "Uma plataforma de desenvolvimento reconfigurável utilizando arquitetura de cluster", I Seminário de Computação Reconfigurável, Instituto de Informática, PUC Minas, Belo Horizonte, 2001.

[8] B. Dreier, M. Zahn, e T. Ungerer, "Parallel and Distributed Programming with Pthreads and Rthreads", Third International Workshop on High-Level Parallel Programming Models and Supportive Environments (HIPS'98), IPDPS 1998 Workshop, Germany, March 1998, pp. 34-40.

[9] R. Butler, A. L. Leveton, e E. L. Lusk, "p4-Linda: A Portable Implementation of Linda", In Proceedings of the Second International Symposium on High Performance Distributed Computing, IEEE Computer Society, Spokane, Washington, July 1994, pp. 50-58.

[10] M. Stumm, e S. Zhou, "Algorithms Implementing Distributed Shared Memory", IEEE Computer Society, May 1990, vol. 23, Issue 5, pp. 54-64.

[11] B. Nitzberg, e V. Lo, "Distributed Shared Memory: A Survey os Issues and Algorithms", IEEE Computer Society, August 1991, vol. 24, Issue 8, pp. 52-60.

[12] J. Nieplocha, R. J. Harrison, e R. J. Littlefield, "Global Arrays: A Portable 'Shared-Memory' Programming Model For Distributed Memory Computers", In Proceedings of Supercomputing '94, Los Alamitos, CA, IEEE Computer Society, 1994, pp. 340-349.

[13] R. S. Ramanjuan, J. C. Bonney, e K. J. Thurber, "Network Shared Memory: A New Approach for Clustering Workstations for Parallel Processing", In Proceedings of Fourth IEEE International Symposium on High Performance Distributed Computing HPDC '95, IEEE Computer Society, August 1995, pp. 48-56.

[14] R. Jain, "The Art of Computer Systems Performance Analyses", John Wiley, 1991.

[15] R. Drath, Visual Object Net ++ . URL: http://www.systemtechnik.tu-Ilmenau.de/~drath/visual_E.htm, Technical University of Ilmenau, Germany, 2004.