Um Multicomputador com Rede de Interconexão Dinâmica

Thadeu B. Corso e **Joni da S. Fraga**

Laboratório de Controle e Microinformática (LCMI)
Departamento de Engenharia Elétrica
Universidade Federal de Santa Catarina
CEP 88040-900 CP 476 Florianópolis SC Brasil
E-mail: thadeu@inf.ufsc.br, fraga@lcmi.ufsc.br

Resumo

Neste artigo é proposto um ambiente para execução de redes de processos comunicantes resultante da associação de um modelo de multicomputador a um mecanismo de comunicação que, em conjunto, são responsáveis por uma drástica redução dos problemas de comunicação em relação aos problemas encontrados em ambientes multicomputadores convencionais. Inicialmente, são descritos modelos de multicomputadores correntes, com redes de interconexão estáticas e dinâmicas, enfatizando os problemas na comunicação decorrentes dos mecanismos de comunicação associados. A seguir, introduz-se o ambiente proposto, constituído pelo multicomputador Crux e o pelo mecanismo de comunicação com atribuição de canais físicos por demanda. Procede-se, então, à avaliação do ambiente proposto, comparando-o com ambientes convencionais, no que diz respeito ao desempenho, à flexibilidade e à simplicidade. Finalmente, são apresentadas as perspectivas de trabalhos futuros envolvendo o ambiente proposto e as conclusões sobre o trabalho já realizado.

Palavras-chave

multicomputador, rede de interconexão, mecanismo de comunicação, avaliação de desempenho.

1. Introdução

Redes de processos comunicantes designam programas paralelos compostos por processos que cooperam exclusivamente por troca de mensagens que fluem entre eles através de canais lógicos de comunicação. Em sua expressão mais restrita, um programa paralelo desse tipo pode exibir uma topologia estática durante toda sua execução. Em sua expressão mais geral, ele pode exibir variações topológicas resultantes da criação dinâmica de processos e de canais.

Multicomputadores designam máquinas que resultam de projetos coerentes de integração de nós de mesma natureza, fisicamente próximos, tipicamente contidos em um único móvel. Cada nó é constituído minimamente de um processador, memória privativa e canais de comunicação. Essas máquinas diferem entre si principalmente pelas características de suas redes de interconexão. Os nós dos multicomputadores podem ser unidos através de canais físicos permanentes constituindo redes de interconexão estáticas. Alternativamente, canais associados aos nós dos multicomputadores podem ser conduzidos através de circuitos de chaveamento eletrônico cuja manipulação permite a atribuição de canais físicos temporários (num

procedimento que pode ser chamado de *conexão de canais físicos*) constituindo redes de interconexão dinâmicas. Cada canal físico (também dito *canal direto*) de um multicomputador serve exclusivamente aos dois nós (ditos nós vizinhos) por ele conectados.

A execução de uma rede de processos comunicantes em um multicomputador depende da resolução do problema do *mapeamento de processos* (correspondente à atribuição de processos a nós) e do *mapeamento de canais lógicos* (correspondente à atribuição de canais lógicos a canais físicos). Em conjunto, esses problemas podem ser referidos como o mapeamento de *redes lógicas* — redes de processos comunicantes — sobre *redes físicas* — redes de interconexão dos multicomputadores. Neste texto, não é considerada a atribuição simultânea de mais de um processo a cada nó.

O mapeamento de uma rede lógica, obtido pela atribuição de cada um de seus processos a um nó distinto e de cada um de seus canais lógicos a um canal físico distinto, na qual nós não são compartilhados por processos e canais físicos não são compartilhados por canais lógicos, pode ser chamado de *correspondência perfeita*.

Quando a correspondência perfeita é possível, às comunicações entre processos correspondem sempre comunicações diretas entre nós vizinhos. Para qualquer rede de processos comunicantes considerada, a correspondência perfeita determina a máxima simplificação das comunicações e o tempo de execução mínimo.

Quando a correspondência perfeita não é possível, o mapeamento das redes lógicas visando sua execução eficiente é uma tarefa mais complexa [1]. Em relação ao mapeamento de processos, nas redes de interconexão dinâmicas ela é arbitrário enquanto que nas redes de interconexão estáticas ela deve considerar os padrões de comunicação dos processos das redes lógicas com o objetivo de explorar fenômenos de localidade das comunicações [2]. Em relação ao mapeamento de canais lógicos, nas redes de interconexão dinâmicas ela depende da multiplexação de canais físicos para cada pacote de tamanho fixo transportado enquanto que nas redes de interconexão estáticas ela origina o roteamento de mensagens entre nós não-vizinhos.

O multicomputador Crux, proposto neste artigo, guarda alguma semelhança estrutural com o Computing Surface [3] que lhe serviu de inspiração inicial. Entretanto, eles representam modelos radicalmente diferentes em relação ao modo de operação, resultantes das funcionalidades distintas de seus componentes, como ficará evidenciado adiante.

O Computing Surface é um de multicomputador reconfigurável cujo modo de operação é caracterizado por ciclos compostos de uma fase de configuração seguida de uma fase de operação efetiva. Em sua configuração básica, cada nó possui quatro canais conectados a um *crossbar* que recebe de uma fonte externa, através de um canal especial, comandos de conexão/desconexão de canais físicos. Além disso, todos os nós estão conectados também a um barramento de controle cujo principal objetivo é o de sincronizar os nós antes de cada fase de configuração. O Computing Surface opera entre fases de configuração de forma similar às redes estáticas. Apesar disso, ele pode ser bastante flexível porque as redes podem ser construídas para responder às características dos programas paralelos.

O tema de fundo deste artigo é o da exploração do paralelismo real provido pelos multicomputadores. Sua motivação fundamental parte da visão dessas máquinas como um ambiente natural para a expressão física das redes de processos comunicantes. O objetivo fundamental é o de propor um ambiente multicomputador de propósito geral que permita o

compartilhamento simultâneo de seus recursos físicos entre várias redes de processos comunicantes de topologia variável, garantindo a correspondência perfeita sempre que possível por questões de desempenho e a comunicação direta em todos os casos por questões de simplicidade.

O texto deste artigo é composto por 5 seções. Na seção 2 introduz-se o ambiente proposto, constituído pelo multicomputador Crux e o pelo mecanismo de conexão de canais físicos por demanda. Na seção 3 procede-se à avaliação do ambiente multicomputador proposto, comparando-o com ambientes convencionais, no que diz respeito ao desempenho, à flexibilidade e à simplicidade. Na seção 4 são apresentadas as perspectivas de trabalhos futuros envolvendo o ambiente proposto. Na seção 5 são apresentadas algumas conclusões sobre o ambiente proposto.

2. Ambiente Multicomputador Proposto

O multicomputador Crux pretende suportar a execução simultânea de várias redes de processos comunicantes de topologia variável pela reunião dos seguintes componentes, mostrados na figura 1:

- Um número expressivo de *nós de trabalho* (cada um dos quais possuindo um processador com memória privativa e vários canais de comunicação).
- Um *nó de controle* para gerir a conexão/desconexão de canais físicos entre nós de trabalho (além da alocação/liberação de nós de trabalho).
- Uma *rede principal* (um *crossbar*) para veicular mensagens entre nós de trabalho através de canais físicos diretos, configurada pelo nó de controle.
- Uma *rede auxiliar* (um barramento) específico para veicular mensagens entre os nós de trabalho e o nó de controle.

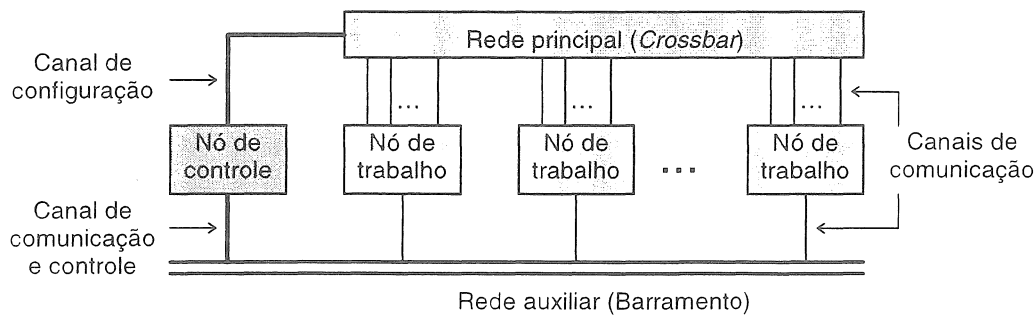

Figura 1: Arquitetura do multicomputador Crux.

O mecanismo de comunicação com atribuição de canais físicos por demanda, que constitui o núcleo do suporte de execução das redes de processos comunicantes no multicomputador Crux, é composto pelos dois níveis mostrados na figura 2. Nesse mecanismo, a rede principal é utilizada para veicular as mensagens dos programas paralelos através de canais diretos entre os nós de trabalho onde executam seus processos. A rede auxiliar é utilizada para resolver de forma simples e eficaz o problema da conexão dos canais físicos da rede principal, veiculando mensagens do suporte de execução, entre os nós de trabalho e o nó de controle, com os pedidos de conexão/desconexão de canais físicos da rede principal. Dessa forma, antes de

proceder à comunicação efetiva entre dois nós de trabalho, é verificada a existência de um canal direto entre eles através da rede principal. Nesse caso, a comunicação pode ser iniciada imediatamente; caso contrário, ela deve ser precedida de um pedido de conexão.

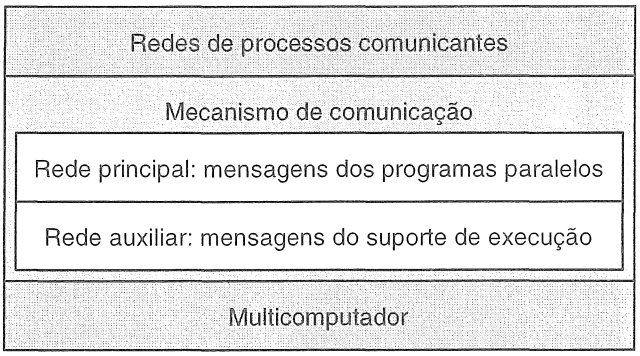

Figura 2: Mecanismo de comunicação com atribuição de canais físicos por demanda.

O mecanismo de comunicação com atribuição de canais físicos por demanda é implementado por um componente central executado no nó de controle e por componentes locais executados em cada nó de trabalho, incorporados ao núcleo do sistema operacional. O componente central é responsável pela operação da rede principal, conectando e desconectando canais físicos em resposta a pedidos emanados dos nós de trabalho. Os componentes locais, acessíveis através de uma pequena coleção de operadores de comunicação, interagem com o componente central para a realização do mecanismo de dois níveis descrito.

O mecanismo de comunicação com atribuição de canais físicos por demanda é um procedimento completamente geral uma vez que a duração dos canais (até sua desconexão, também por demanda) pode variar desde o tempo de transporte de uma único pacote até o da execução completa de um programa paralelo.

Para complementar o suporte de execução, um mecanismo de alocação/liberação de nós de trabalho por demanda associado à criação/destruição de processos também é provido. Da mesma forma que o mecanismo de comunicação, esse serviço é fornecido por um componente central executado no nó de controle e por componentes locais executados em cada nó de trabalho, incorporados ao núcleo do sistema operacional.

3. Análise do Ambiente

A eficiência do ambiente proposto pode ser avaliada tanto em relação ao desempenho da rede do multicomputador Crux quanto em relação à flexibilidade e à simplicidade do mecanismo de comunicação com atribuição de canais físicos por demanda.

3.1. Desempenho

Para avaliar o multicomputador Crux no que diz respeito ao desempenho de sua rede de interconexão, procedeu-se à sua comparação com uma representante qualificada entre as redes estáticas (o torus) e outra entre as redes dinâmicas (o *crossbar*), usando técnicas de simulação discreta. Entre outras alternativas razoáveis de redes estáticas, a escolha do torus deveu-se não

apenas ao seu bom desempenho relativo a outras redes estáticas mas também à uniformidade de composição dos seus nós, que independe do número de nós da rede [4]. (Observe-se que o Computing Surface se comporta como uma rede estática durante cada fase de operação.) Entre as redes dinâmicas, a escolha do *crossbar* deveu-se exclusivamente ao seu desempenho superior em relação às demais redes dinâmicas [5]. Considerou-se que os nós dos multicomputadores comparados possuem um processador principal, memória privativa e um controlador de comunicação composto de um processador especializado e canais de entrada e de saída independentes.

Nas simulações realizadas, foram assumidos os seguintes mecanismos de comunicação, específicos para cada rede:

- Torus - As mensagens são divididas em *flits* (*flow control digits*) de tamanho reduzido (1 *byte* nas simulações efetuadas). Todos os *flits* de uma mesma mensagem seguem a mesma rota entre o nó origem e o nó destino. Tão logo o primeiro *flit* (o único que carrega informações de roteamento) alcança um nó intermediário, ele pode ser transferido ao seguinte. Quando não existem canais de saída livres, os *flits* podem ser armazenados nos nós intermediários que devem ter capacidade para acomodar até uma mensagem inteira para cada canal de entrada. Essa técnica de roteamento é conhecida como *virtual cut-through* [6]. Apenas as rotas com distância mínima são utilizadas. A escolha desse mecanismo deveu-se ao seu desempenho superior em relação ao clássico *store-and-forward*, embora a vantagem não seja tão expressiva como se costuma acreditar [7].
- *Crossbar* - As mensagens são divididas em pacotes de tamanho moderado (16 *bytes* nas simulações efetuadas). Um árbitro embutido na rede permite a transferência de pacotes, realizando a conexão e a desconexão de um canal direto entre o nó origem e o nó destino para cada pacote. Esse é o mecanismo natural para essa rede.
- Crux - Cada mensagem é transferida de uma só vez entre o nó origem e o nó destino como um bloco monolítico através de um canal direto da rede principal. Mensagens de controle de tamanho reduzido (2 *bytes* nas simulações efetuadas) trafegam pela rede auxiliar sempre que um pedido de conexão deva preceder a comunicação efetiva. Esse é o mecanismo com atribuição de canais físicos por demanda.

Nas simulações realizadas, foram assumidas as seguintes propriedades gerais, independentes das redes:

- Os nós geram fluxos contínuos de mensagens (eles estão sempre aptos a iniciar a emissão da próxima mensagem assim que acabam de emitir a anterior), segundo densidades que variam entre uma e quatro mensagens sendo emitidas simultaneamente a partir de cada nó.
- Os nós destino são escolhidos conforme a distribuição uniforme.
- As mensagens possuem tamanhos que seguem a distribuição exponencial.
- A velocidade de transmissão dos canais é de 1 *byte* por unidade de tempo.

Os parâmetros considerados foram o número de nós, o tamanho das mensagens e a densidade das mensagens. Em cada simulação, variou-se apenas um desses parâmetros e mediu-se os tempos mínimo, médio e máximo de trânsito das mensagens, além dos desvios padrão associados, mas apenas os tempos médios são mostrados nas figuras 3, 4 e 5.

A figura 3 apresenta os resultados das simulações para modelos com 25, 36 e 49 nós, considerando mensagens com tamanho médio de 512 *bytes* e densidade de emissão de mensagens igual a 1.

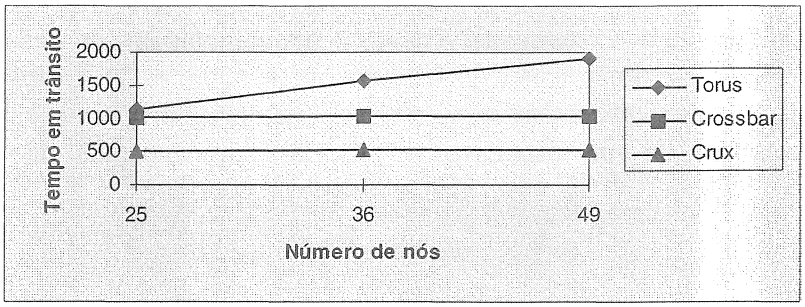

Figura 3.

A figura 4 apresenta os resultados das simulações para modelos com 36 nós, considerando mensagens com tamanho médio de 512 *bytes* e densidades das mensagens iguais a 1, 2, 3 e 4.

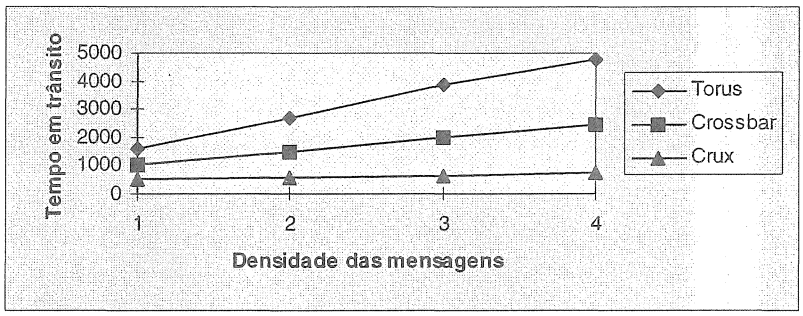

Figura 4.

A figura 5 apresenta os resultados das simulações para modelos com 36 nós, considerando mensagens com tamanhos médios de 128, 256, 384 e 512 *bytes* e densidade de emissão de mensagens igual a 1.

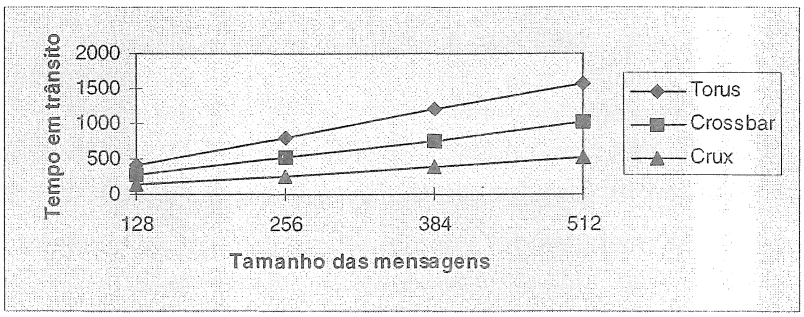

Figura 5.

Os resultados das simulações representam valores típicos do desempenho relativo das redes consideradas, selecionados de um conjunto muito maior de simulações omitidas neste texto. Como apreciação global, as figuras 3, 4 e 5 determinam três faixas desempenho nitidamente distintas, na qual o Crux se posicionou na faixa superior, o *crossbar* na faixa intermediária e o torus na faixa inferior, para as variáveis selecionadas e dentro dos limites estabelecidos para elas. Da figura 3, verifica-se que os desempenhos do Crux e do *crossbar* independem do número de nós enquanto que o do torus decresce com o aumento do número de nós. Esse resultado reflete o fato de que tanto no Crux quanto no *crossbar* as comunicações são sempre diretas enquanto que no torus a distância média entre os nós cresce com o número de nós. Das figuras 4 e 5, verifica-se que o Crux é menos sensível que o *crossbar* que, por sua vez, é menos sensível que o torus às variações de tamanho e de densidade das mensagens.

Entre as simulações cujos resultados foram omitidos, incluem-se as que envolveram mensagens criadas seguindo a distribuição exponencial. Nessas simulações foram medidos os tempos entre a criação de cada mensagem no nó origem e a sua recepção no nó destino. Para densidades moderadas, comparáveis com as apresentadas acima, os resultados foram equivalentes. Para densidades maiores, verificou-se um rápido crescimento dos tempos medidos em todos os modelos.

3.2. Flexibilidade

Para avaliar a flexibilidade relativa de diversos ambientes multicomputadores, pode-se comparar os procedimentos necessários ao mapeamento das redes lógicas em cada um deles:

- *Execução de uma rede lógica estática única para a qual é possível estabelecer a correspondência perfeita.*
  No torus, o mapeamento deve ser planejado antes da carga do programa paralelo à partir do conhecimento da topologia da rede lógica. No Computing Surface, o mapeamento também deve ser planejado para permitir que o programa de configuração construa a rede física antes da carga do programa paralelo. No *crossbar* (com árbitro embutido), a atribuição de processos a nós é determinada arbitrariamente pela duração dos processos e a de canais lógicos a canais físicos é estabelecida para cada pacote. No Crux, a atribuição de processos a nós também é determinada arbitrariamente pela duração dos processos e a de canais lógicos a canais físicos é estabelecida pela duração do canal lógico.
  Em relação à atribuição de canais lógicos a canais físicos, o árbitro do *crossbar* deve ser acionado a cada pacote enquanto que o mecanismo de conexão de canais físicos por demanda do Crux é acionado apenas para a criação/destruição de cada canal lógico. Além disso, no *crossbar*, o compartilhamento de canais físicos exige o tratamento de pacotes.

- *Execução de uma única rede lógica estática para a qual não é possível estabelecer a correspondência perfeita.*
  No torus e no Computing Surface, o mapeamento ainda deve ser planejado antes da carga do programa paralelo quando se quer tirar proveito de fenômenos de localidade através de uma adaptação conveniente da rede lógica à rede física. No *crossbar* e no Crux, os procedimentos não são afetados por essa nova situação. Entretanto, no Crux o mesmo mecanismo de atribuição de canais físicos por demanda deve agora ser acionado sempre que atribuição de um canal lógico a um canal físico deva ser alterada.

- *Execução simultânea de várias redes lógicas dinâmicas.*
  No torus e no Computing Surface, o mapeamento não pode ser planejado antes da carga dos programas paralelos pelo desconhecimento prévio das topologias das redes lógicas. Nessa situação o aproveitamento de fenômenos de localidade se torna uma tarefa bastante complexa. No *crossbar* e no Crux, os procedimentos não são afetados por essa nova situação. Em relação à situação anterior, ampliam-se consideravelmente as dificuldades de uso das redes físicas estáticas e do Computing Surface em relação ao *crossbar* e ao Crux.

A duração de um canal físico em uma rede de interconexão estática é igual ao tempo de existência da própria rede. A duração de um canal físico de um *crossbar* corresponde ao tempo de transporte de um pacote de tamanho fixo. A duração de um canal físico no Computing Surface eqüivale ao tempo de cada fase de operação. Em nenhum desses casos a duração dos canais físicos coincide com a dos canais lógicos. A duração de um canal físico no Crux, operando com o mecanismo de comunicação com atribuição de canais físicos por demanda, pode ser exatamente igual à duração do canal lógico quando o número de canais lógicos dos processos não supera o número de canais físicos dos nós onde eles executam. Além disso, esse mecanismo é suficientemente geral para tratar de maneira uniforme a multiplexação de canais físicos quando o número de canais lógicos dos processos supera o número de canais físicos dos nós onde eles executam. Em suma, a flexibilidade superior do Crux resulta da maior facilidade de adaptação das redes lógicas porque as redes físicas podem ser construídas em tempo de execução em resposta as exigências das redes lógicas, inclusive as que envolvem criação dinâmica de processos e de canais lógicos.

3.3. Simplicidade

Para avaliar a simplicidade relativa de diversos ambientes multicomputadores, pode-se confrontar os procedimentos envolvidos nas comunicações através das redes físicas em cada um deles. Nas redes estáticas, as comunicações envolvem roteamento de mensagens. Entre fases de configuração, o Computing Surface se comporta como uma rede estática. No *crossbar*, as comunicações envolvem o acionamento do árbitro da rede a cada pacote transportado e o tratamento de pacotes. No Crux, o algoritmo elementar de conexão de canais físicos por demanda, que permite a comunicação direta através de mensagens completas. A simplicidade superior do Crux resulta da eliminação do roteamento de mensagens e do tratamento de pacotes que implicam em uma redução drástica da complexidade das comunicações, podendo ser considaradas justificativas suficientes para validar a proposta.

Quando se considera que as funções de controle providas pelo suporte de execução durante a operação normal dos multicomputadores envolvem mensagens, pode-se introduzir os termos *rede de controle* e *rede das aplicações*. No torus, no Computing Surface e no *crossbar*, a mesma rede física deve ser compartilhada entre a rede de controle e as rede das aplicações enquanto que no Crux existe uma rede de controle independente da rede das aplicações. Essa separação também contribui para a simplicidade superior da proposta por constituir um elemento natural de estruturação de um sistema operacional para o Crux.

3.4. Inconvenientes

A proposta aqui apresentada se limita, pelo menos a curto prazo, a multicomputadores de porte médio, por causa do moderado número de canais dos *crossbars* disponíveis atualmente. Entretanto, o aumento de demanda por *crossbars* de maiores dimensões poderá no futuro reduzir consideravelmente seu custo comercial. Apesar disso, a flexibilidade do mecanismo de comunicação com atribuição de canais físicos por demanda permite considerar o uso de vários *crossbars* independentes, a cada um dos quais seria conectado um canal físico distinto de cada nó. Se essa solução reduz a gama de topologias em relação ao *crossbar* único, ela permite empregar *crossbars* de menores dimensões [8].

O Crux possui dois componentes compartilhados críticos: o nó de controle e a rede auxiliar. Tanto suas principais qualidades como seus principais inconvenientes decorrem desses componentes. O nó de controle e a rede auxiliar podem se transformar em gargalos se o ambiente for usado por redes lógicas que envolvam comunicações com mensagens curtas que exijam freqüentes mudanças de atribuições de canais físicos. (Esse fenômeno, previsível desde a concepção da arquitetura do multicomputador Crux, foi efetivamente comprovado em simulações específicas). Além disso, em relação à tolerância a falhas, uma interrupção de funcionamento do nó de controle ou da rede auxiliar compromete todo o ambiente. A falha de um nó de trabalho, em compensação, pode ser isolada com facilidade.

4. Estado Atual e Perspectivas

Enfatizou-se neste artigo as características do multicomputador Crux e do mecanismo de comunicação com atribuição de canais físicos por demanda. Embora, a associação desses componentes constituam o centro da proposta, um ambiente completo para execução de programas paralelos deve fornecer ainda os serviços habituais de um sistema operacional e de uma linguagem de programação.

A proposta apresentada é suportada por um pequeno protótipo do Crux com 9 nós que, se é insuficiente para sua avaliação efetiva em programas paralelos de interesse prático, fornece todos os recursos necessários para o desenvolvimento de *software*, enquanto se aguarda recursos para a construção de um modelo de maior porte.

De fato, o ambiente proposto faz parte de um projeto de pesquisa que já produziu sobre esse protótipo dois importantes componentes para suportar a execução de programas paralelos:

- Um sistema operacional com interface Unix foi implementado segundo o modelo cliente-servidor [9]. Ele se apoia sobre um micronúcleo que provê apenas as funções de gerência de processos além do mecanismo de conexão de canais físicos por demanda.
- Um interpretador da linguagem SuperPascal (na realidade uma extensão do interpretador distribuído por Brinch Hansen [10]) foi implementado com a as seguintes características: cada processo SuperPascal gera um processo Unix e eles se comunicam através do mecanismo de comunicação com atribuição de canais físicos por demanda.

5. Conclusões

Este artigo introduziu um ambiente para execução de redes de processos comunicantes resultante da integração do multicomputador Crux com o mecanismo de comunicação com atribuição de canais físicos por demanda.

Verificou-se a eficiência do ambiente proposto é considerável se comparada com outros ambientes conhecidos, tanto em relação ao desempenho quanto em relação à flexibilidade e à simplicidade. Considerou-se suas limitações decorrentes tanto da tecnologia atual dos *crossbars* quanto em relação à própria arquitetura do Crux.

Enfatizou-se que o ambiente descrito é suportado por um protótipo de dimensões reduzidas sobre o qual já foi possível comprovar na prática suas qualidades relativas à flexibilidade de operação e à simplicidade uso. Espera-se ainda um protótipo de dimensão superior para comprovar seu desempenho, conforme os resultados das simulações, em algoritmos de interesse prático.

Referências

1 Hwang, K., *Advanced Computer Architecture - Parallelism, Scalability, Programmability*, McGraw-Hill, 1993.

2 Reed, D. A., Fujimoto, R. M., *Multicomputer Networks: Message-Based Parallel Processing*, MIT Press, 1987.

3 McDonald, N., in *Past, Present, Parallel: A Survey of Available Parallel Computing Systems*, Springer-Verlag, New York, 1991.

4 Feng, T.-Y., *A Survey of Interconnexion Networks*, IEEE Computer, v. 12, n. 14, p. 12-27, dezembro de 1981.

5 Broomell, G., Heath, J. R., *Classification Categories and Historical Development of Circuit Switching Topologies*, ACM Computing Surveys, v. 15, n. 2, p. 95-133, junho de 1983.

6 Kermai, P., Kleinrock, L., *Virtual Cut-Through: A New Computer Communication Switching Technique*, Computer Networks, v. 3, n. 14, p. 267-286, outubro de 1979.

7 Fausto, L. F., Corso, T. B., Freitas Filho, P. J., *Store-and-Forward versus Wormhole Routing Mechanisms: A Comparative Performance Evaluation in Multicomputer Networks*, Summer Computer Simulation Conference, Arlington, Estados Unidos, julho de 1997.

8 Nicole, D. A., et al., *Switching Networks for Transputers Links*, VIII Occam Users Group, May 1988.

9 Corso, T. B., Fraga, J. da S., *Um Sistema Operacional para um Multicomputador*, relatório técnico, INE-CTC-UFSC, 1997.

10 Hansen, P. B., *The Programming Language Superpascal*, Software - Practice and Experience, v. 24, n. 5, p. 467-483, maio de 1994.