UM SIMULADOR DATAFLOW IMPLEMENTADO EM UMA REDE DE TRANSPUTERS

ÁLVARO GARCIA NETO
Depto de Física e Ciência dos Materiais - USP e Universidade de Manchester
Caixa Postal 369 - 13.560, São Carlos, SP
A.P.W. BOHM, M.C. KALLSTROM
Universidade de Manchester
Department of Computer Science, Manchester, M13 9PL UK

RESUMO

Um simulador "dataflow" está sendo desenvolvido para execução em um processador paralelo baseado em transputers (T-Rack). Uma discussão sobre a necessidade de simuladores paralelos é feita, e um modelo computacional restrito (DF-1) é definido. O projeto Percival e alguns de seus produtos, um processador MIMD e um ambiente de programação Occam são descritos, bem como detalhes de implementação e primeiros resultados.

ABSTRACT

A dataflow simulator for execution in a transputer based parallel processor is being developed. A discussion about the need for parallel simulators is carried and a simple computational model is defined. The ParSiFal project and some of its products, a MIMD processor and an Occam software package are described, as well as some implementation details and first results.

1. INTRODUÇÃO

Simulação é uma poderosa ferramenta de pesquisa em arquitetura de computadores. As características do desempenho de computadores nunca construídos nem sempre são inteiramente preditas teoricamente, e nos estágios iniciais de desenvolvimento, quando a construção de protótipos ainda é prematura, há poucas outras ferramentas disponíveis.

Mesmo quando uma particular arquitetura alcança maturidade, a simulação ainda apresenta duas vantagens:

a) O grau de informação sobre sub-sistemas individuais é maior. No protótipo a obtenção de informação extra quase sempre implica em circuitos mais complexos.

b) São mais flexíveis à alterações. Experimentação via modificações estruturais quase sempre é inviável em protótipos, podendo entretanto ser efetuada no simulador pela inclusão de código adicional.

No caso de arquiteturas paralelas, as exigências computacionais tornam as simulações bastante lentas, quando não impossíveis. Isso motivou o interesse no desenvolvimento de sistemas de alto desempenho voltados à simulação, como o Projeto Percival.

Um simulador para a "Manchester Multi-Ring Dataflow Machine" |1| está sendo desenvolvido, para execução em um processador MIMD, desenvolvido como parte do Projeto Percival. Os propósitos do simulador são o estudo de tráfego e distribuição de trabalho em computadores a fluxo de dados, e a comparação das diferenças de comportamento entre simuladores sequenciais e paralelos. Esse simulador permite experiências com programas bem mais complexos que os até então possíveis em simuladores convencionais.

2. TERMINOLOGIA

Simulações podem ser efetuadas a nível de BLOCO, de COMPONENTE ou de PORTAS. A temporização (uma abstração do conceito usual de tempo) pode ser CONTÍNUA, onde o relógio forma uma sequência aritmética, crescente, contínua e independente do comportamento das unidades sob simulação, ou DISCRETA, onde o relógio é incrementado monotonicamente mas não continuamente, sendo os intervalos definidos pela ocorrência de eventos. Um simulador é chamado ACOPLADO se o relógio de simulação é

mantido o mesmo em todas as unidades do sistema, ou DESACOPLADO caso contrário.

Nesse artigo, processos são classificados como MONÁDICOS se têm um canal de entrada e um de saída ou DIÁDICOS se têm dois canais de entrada e dois de saída.

3. NECESSIDADE DE SIMULADORES PARALELOS

Um simulador acoplado, contínuo e a nível de bloco, escrito em Pascal |2| tem sido a base de nossa pesquisa em arquitetura a fluxos de dados até o momento. As exigências de memória e capacidade de processamento são tão acentuadas que mesmo em computadores razoavelmente velozes (Sun 3/280+Sun 3/60, 4+3 MIPS) uma simulação chega a dias. Como exemplo, a memória desses computadores (8+16 Mbytes) não permite a simulação de multiplicação de matrizes maiores que 40x40.

Essa inadequação motiva um simulador paralelo multi-processado, cujos principais aspectos são:
a) Implementação em nível adequado à granularidade do processador paralelo a ser empregado.
b) Implementação que permita distribuição adequada de carga entre processadores, sem penalização excessiva para comunicação e sincronização.
c) Implementação de um modelo temporal compatível tanto com o processador paralelo hospedeiro como com o processador a fluxo de dados a ser simulado.
d) Implementação de um modelo de troca de informação que não force sincronização artificial ao processador a fluxo de dados a ser simulado.

Como tanto o processador hospedeiro como o a ser simulado são assíncronos, o simulador é desacoplado. As características de "dataflow" sugerem o uso de temporização discreta; a granularidade do paralelismo dos transputers sugere um simulador a nível de bloco. O resultado final a ser alcançado, um simulador desacoplado, discreto e a nível de bloco, apresenta dois problemas: a manutenção de eventos ordenados e a identificação global e unívoca do instante de ocorrência de um evento.

4. O MODELO "DATAFLOW" RESTRITO

Nossa estratégia foi implementar o simulador para um modelo computacional simplificado (DF-1). Dessa forma pudemos nos concentrar nas especificações do simulador e não nas dificuldades de fidelidade para com o paradigma "dataflow" estabelecido. O modelo DF-1 é uma forma restrita de fluxo de dados com instruções de apenas uma entrada. Apesar do reduzido repertório (figura 1), o DF-1 é suficientemente

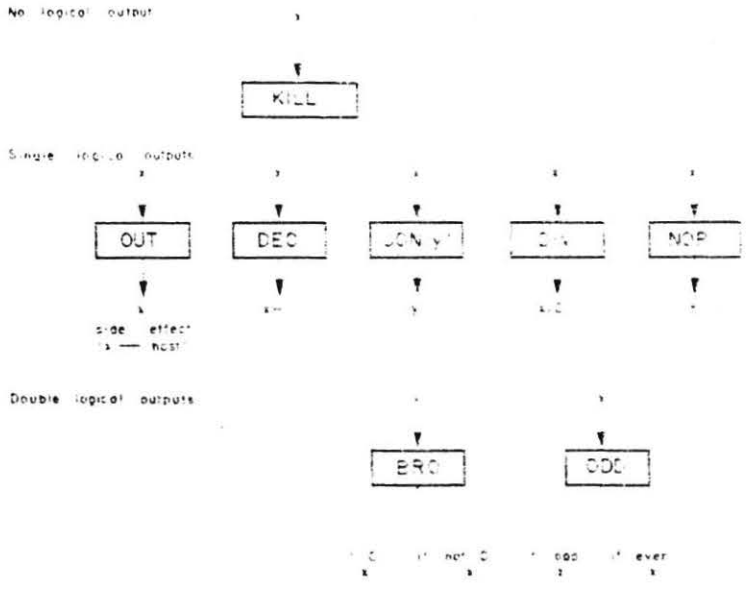

Figura 1 - Repertório de Instruções DF-1

expressivo para codificar programas com as propriedades tipicamente associadas a rotinas "dataflow".
Programas DF-1 são grafos cíclicos de dependência de dados, cada nó representando uma instrução e cada arco indicando a próxima operação. As intruções produzem zero, um ou dois resultados, que podem ser direcionados para até duas outras. Por exemplo, uma instrução DEC (decremente) gera um único resultado (o valor de entrada menos um), que pode ser direcionado para até duas outras.

O processador virtual DF-1 possui um processador de entrada (HOST) e uma certa quantidade de elementos processadores (PEs) conectados por uma rede de interconexão (figura 2). Informação é trocada entre os módulos em pacotes de dados, com os valores a serem processados, ou pacotes de mensagens, com comandos e dados de monitoração.

O processador hospedeiro é responsável pelo acesso a arquivos e interface do usuário. A interconexão é implementada como um anel de elementos chaveadores (SW). Cada PE é um anel composto por três sub-sistemas: uma fila de pacotes (TQ), uma memória de instruções (NS) e uma unidade de processamento (PU). Cada uma dessas unidades é similar às da "Manchester Dataflow Machine" |3| : TQ fornece dados à NS, que apõe a cada pacote a instrução a ser executada e os transfere a um PU disponível, que executa a instrução requerida e gera um ou mais pacotes-resultado, direcionando-os à TQ.

No simulador, cada módulo (HOST, SW, TQ, NS e PU) é um processo concorrente, comunicando com os vizinhos através de estruturas de dados compartilhadas de acesso controlado denominadas canais.

5. O PROJETO PERCIVAL

O projeto Percival (em inglês ParSiFal, acrônimo para "Parallel Simulation Facility") é um empreendimento conjunto das Universidades de Cambridge, Manchester e Politécnica de Londres, FEGS Ltd, GEC Research Ltd, INMOS Ltd e Logica Cambridge Ltd, sob os auspícios de um programa governamental de incentivo à computação (Alvey). Dentre seus produtos há um processador paralelo (T-Rack) e um ambiente de programação (TDS).

5.1. T-Rack

O T-Rack é um computador MIMD empregando transputers INMOS, um microprocessador rápido dotado de conexão que permite a construção de sistemas com centenas de processadores |4| . O T-Rack possui 64 transputers T414 ou T800, cada um com até 2 Mbytes de memória, conectados por uma rede configurável. O sistema foi projetado na Universidade de Manchester, e está operacional desde meados de 1987. Vários T-Racks podem ser interligados formando computadores ainda maiores.

5.2. O Modelo Occam e TDS

Occam é uma linguagem de paralelismo explícito próxima ao hardware do transputer |5| , baseada em Processos Sequenciais Comunicantes |6| . Uma de suas características principais, a

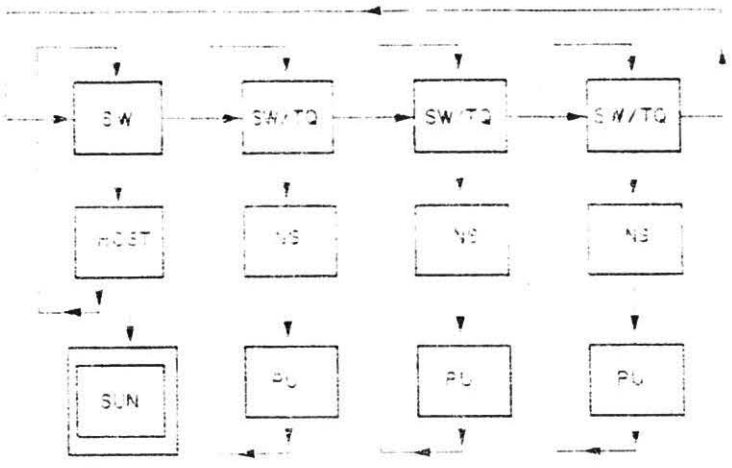

Figura 2 - Processador Virtual DF-1

comunicação e sincronização por canais, é um mecanismo natural para a implementação da troca de mensagens típica do "dataflow". O conjunto reduzido de primitivas de controle de fluxo e de paralelismo facilita a verificação formal da correção de algorítmos paralelos. A INMOS advoga que o transputer foi projetado especificamente para execução eficiente de programas Occam.

TDS |7| é um ambiente de programação composto por um editor, um compilador Occam 2, e utilitários de carregamento, configuração e monitoração de uma rede de transputers, executável em Suns, transputers e microcomputadores e gerando código para qualquer uma dessas máquinas. Atualmente usamos uma versão beta.

6. TEMPORIZAÇÃO

É possível resolver um dos problemas do simulador implementando um relógio global. Isso entretanto força sincronização de todos os processos, demasiado caro num sistema projetado para simular uma máquina assíncrona. Duas das três funções do relógio global no simulador sequencial são agora conduzidas de outra forma: sincronização é efetuada via canais; paralelismo (antes simulado pela execução de varias rotinas com o relógio suspenso) ou é provido pelo T-Rack, ou simulado por chaveamento de tarefas em "time-sharing" internamente ao transputer. A função restante, caracterização de eventos, é efetuada carimbando-se todos os pacotes no instante de sua produção com o valor do relógio.

Cada processo possui um relógio local. Para garantir monotonicidade, esse relógio é verificado a cada recepção de pacote, e se o carimbo chegado for maior que o valor corrente, a diferença é considerada tempo inativo e o relógio atualizado. Assim, em processos monádicos, como TQ, NS e PU e partes do HOST, o tempo é uma grandeza monotônica, crescente e contínua por partes.

Em processos diádicos, devido aos ciclos que possibilitam e à ausência de prioridade nas suas entradas, não é possível garantir o consumo dos pacotes na ordem de seus carimbos temporais |8| . A solução desse problema não é fácil, pois o processo SW é inerentemente diádico e a imposição de entradas com prioridade leva a situações de travamento. Além disso, uma das características de modelo de fluxo de dados de Manchester são os anéis (ciclos).

Para reduzir o fenômeno de ultrapassagem (consumo de pacotes fora de ordem), os processos SW têm temporização diferente (figura 3): a SW não carimba pacotes, e o processo TQ foi incorporado a sua saída. A disciplina da TQ foi alterada de FIFO para fila ordenada por carimbo. Assim, pacotes ultrapassados tem oportunidade de retornar a suas posições originais antes de serem processados.

Essa estratégia reduz ultrapassagem, mas não a elimina. Parâmetros, como tamanho da fila e latência dos processos, alteram a magnitude do problema. Atualmente trabalhamos no estabelecimento de parâmetros que reduzam o fenômeno. Dados preliminares sugerem resultados lógicos (ordem de produção de respostas)

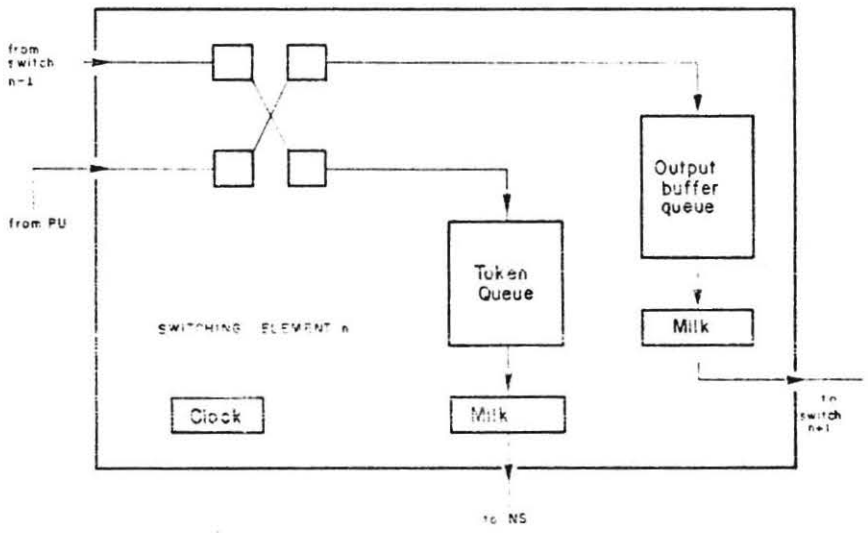

Figura 3 - O elemento chaveador (SW)

bastante compatíveis com os do simulador convencional, mesmo com parâmetros que causam elevados níveis de ultrapassagem.

7. IMPLEMENTAÇÃO

Cada módulo da máquina DF-1 (figura 3) é codificado como um processo Occam e mapeado em um transputer. Alguns processos são decompostos em sub-processos que simulam paralelismo por "time-sharing" em uma única UCP.
Canais unidirecionais entre os módulos são mapeados na rede. Algumas filas FIFO adicionais são inseridas para evitar travamento do tipo "store-and-forward". Processos de gerenciamento de filas são sensíveis a travamento, por exemplo, ficando bloqueados quando tentando enviar um pacote que removeria a causa do bloqueio. Como Occam não permite o compatilhamento de dados inter-processos, a fila é local a um único processo, e seu bloqueamento enquanto tentando transmitir impede que outros pacotes sejam nela inseridos, causando a propagação do travamento.

Para evitar propagação de travamento, um esquema mestre/escravo é implementado, com um processo escravo, chamado ORDENHADOR, sempre solicitando dados à fila. Caso seja impossível enviar um pacote, é o ordenhador quem fica bloqueado, e não o gerenciador. Assim, novos pacotes podem sempre ser inseridos.

Nessa implementação, cada PE armazena o mesmo programa, e um único PE recebe o pacote que dispara o processamento. Os pacotes subsequentes são distribuídos aos demais PEs por técnicas de "hash". A análise da distribuição de carga, tanto no simulador como no processador a fluxo de dados sob simulação, é outro assunto em estudo.

Para preservar capacidade de processamento, e também para simplificar a depuração, as rotinas são essencialmente programadas como algorítmos sequenciais. Embora o modelo Occam permita codificar paralelismo até a nível de instrução, a exposição desse tipo de paralelismo não só não aumenta a velocidade de processamento, como também a diminui. Dados à nossa disposição sugerem que o chaveamento de tarefas é feito na ordem de micro-segundos no T414. Apenas "paralelismo útil" é codificado, geralmente na forma de invocação paralela de rotinas.

8. RESULTADOS PRELIMINARES

A performance de DF-1 no T-Rack está sendo comparada com a de um simulador 1-DF sequencial através de um conjunto de programas com características típicas de "dataflow". Um exemplo simples é mostrado na figura 4, que decrementa o valor de x e recicla até zero.

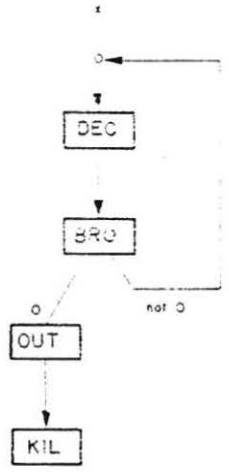

Figura 4 - Um programa 1-DF

A intervalos regulares e programáveis durante a execução do programa, os vários módulos do simulador expontaneamente enviam dados estatísticos ao hospedeiro, que os armazena para pós-processamento e exibição numa ferramenta de monitoramento de sistemas paralelos Tan87 . As grandezas monitoradas incluem comprimentos e taxas de entrada/saída de filas, percentagem de utilização, tempo considerado inativo e o grau de ultrapassagem. Dados preliminares indicam considerável aumento de performance em relação ao simulador convencional executando em um VAX 11/780.

9. CONCLUSÃO

Nosso alvo é construir um simulador completo da máquina "dataflow" de Manchester. Nossa estratégia consistiu em desenvolver primeiramente um modelo restrito (1-DF). Grande cuidado foi exercido para preservar no modelo propriedades características de fluxo de dados. A codificação do 1-DF apresentou muitas oportunidades de experimentação com programação paralela, além de um grande volume de dados. Estão em curso a comparação desses dados com os do simulador convencional e a implementação de um simulador completo.

REFERÊNCIAS

|1| Gurd, J., Kirkham, C.C., Bohm, A.P.W., "The Manchester Dataflow Computing System", Special Topics in Supercomputing 1, North-Holland, Janeiro, 1987.

|2| Teo, Y.M., "Performance Evaluation of a Heterogeneus Multi-Ring Dataflow Machine", Tese de Mestrado, Departamento de Computação, Universidade de Manchester, 1987.

|3| Gurd, J.R., Watson, I, "Data Driven System for High Speed Parallel Computing - Part 2: Hardware Design", Computer Design, 9(7), julho, 1980.

|4| INMOS Ltd: Transputer Reference Manual, 1a. Edição, , INMOS Ltd, 1000 Aztec West, Almondsbury, Bristol, BS12 4SQ, Grã Bretanha, 1984.

|5| May, D., Shepperd, R., "Occam and the Transputer" in Proceedings Workshop on Hardware-Supported of Concurrent Languages in Distributed Systems, IFIP WG10.3, North Holland, pp 19-33, outubro, 1984.

|6| Hoare, C.A.R., "Communicating Sequential Processes", Communications of the ACM, pp 666-667, agosto, 1978.

|7| Logica Cambridge Ltd, "Transputer Development System for Sun Workstations, versão 1.5, Logica Cambridge Ltd, 104 Hills Rd, Cambridge CB2 1LQ, Grã Bretanha, novembro, 1987.

|8| Peacock, J.K., Wong, J.W. e Manning, E.G., "Distributed Simulation Using a Network of Processors", Computer Networks 3, pp 44-56, 1979.