ANÁLISE QUANTITATIVA DE ARQUITETURAS PIPELINE

Paulo Fernandes*
Roland Teodorowitsch**
Philippe Navaux***

Curso de Pós-Graduação em Ciência da Computação
CPGCC - DI - II - UFRGS
C.P. 1501 - 90.001 - Porto Alegre - RS - BRASIL
Telef. (0512) 21-8499 - Telex (051) 2680 CCUF

RESUMO

Este trabalho define um modelo para análise de desempenho de arquiteturas pipeline. A análise proposta consiste na obtenção de parâmetros quantitativos relevantes ao desempenho de uma determinada arquitetura definida tanto pelo hardware quanto pelo software que nela será executado. A partir de um método de análise semelhante, referenciado em bibliografia, busca-se uma maior precisão através da construção de um modelo mais detalhado para descrever as arquiteturas pipeline.

ABSTRACT

This work defines a model for performance avaluation of pipeline architectures. The proposed analysis obtains quantitative parameters concerning the software and the hardware of a pipelined architecture. This analysis is based on a previous developed method which attempts to give more detailed results.

---

* Bacharel em Ciências da Computação (UFRGS)
** Bacharel em Informática (PUCRS)
*** Doutor em Informática (Grenoble-França)

Este trabalho é parcialmente apoiado pelas seguintes instituições brasileiras: CNPq, FINEP, CAPES e SID-Informática

1. Introdução

A importância da avaliação de desempenho no projeto de arquiteturas com capacidade de processamento paralelo é indiscutível. Definir a melhor configuração de elementos de processamento (EP's), as estruturas físicas a utilizar e a distribuição de processamento empregada são tarefas vitais para a obtenção de uma arquitetura eficiente. No entanto, a comparação qualitativa entre diferentes opções de implementação não pode ser feita automaticamente, pois envolve julgamento de valores quantitativos. Neste sentido, é proposta uma análise quantitativa de arquiteturas do tipo pipeline que define valores numéricos para alguns parâmetros considerados relevantes ao desempenho destas arquiteturas. A partir destes valores quantitativos é possível fazer um julgamento qualitativo das diversas opções de implementação, segundo critérios próprios de cada projetista.

O método proposto, desenvolvido a partir de uma análise referenciada em bibliografia, apresenta um modelo mais detalhado para descrever as arquiteturas pipeline.

Neste artigo, descreve-se inicialmente as arquiteturas do tipo pipeline, apresentando o escopo de aplicação da análise proposta. A seguir são apresentadas as análises para o caso ótimo e para o modelo proposto (refinado). Por fim, o método é exemplificado e são discutidos resultados e restrições do método.

2. Arquiteturas Pipeline

As arquiteturas de computadores com capacidade de processamento paralelo definidas como pipeline possuem características específicas que podem ser resumidas numa expressão: "processamento sobreposto". A exemplo de uma linha de produção em uma fábrica, a idéia básica consiste em particionar uma tarefa em sub-tarefas menores que podem ser executadas por diferentes EP's. Tarefas sucessivas são enfileiradas numa linha de processamento (pipe) com diversos EP's e são executadas de maneira sobreposta a nível de sub-tarefas. Desta forma se

tivermos mais de uma tarefa a ser executada estas tarefas podem ter seu tempo total de execução sobreposto através da distribuição de sub-tarefas aos diferentes EP's.

A análise proposta neste artigo é aplicável a arquiteturas pipeline classificadas, segundo Händler [HAN 77] como aritméticos, ou seja, pipelines com funções de execução. Na classificação proposta por Ramamoorthy e Li [RAM 77] enquadram-se como multifunção (com capacidade de executar funções diferentes), dinâmicos (com capacidade de reconfiguração da ligação entre seus EP's durante a execução) e vetoriais. Esta escolha justifica-se pelo fato de que todos os pipelines que não se enquadram nesta classificação podem ser modelados como simplificações desta categoria.

As características das arquiteturas pipeline são abordadas, neste trabalho, segundo sua relevância frente ao desempenho. Apresenta-se a seguir algumas características das arquiteturas pipeline a fim de aprofundar o conhecimento destas e embasar o estudo do seu desempenho. Maiores detalhes sobre as características citadas podem ser encontrados no capítulo 3 de [HWA 85].

Entre as características básicas da arquitetura temos: o número de tarefas executáveis (funções) e o número de estágios (EP's). Para cada tarefa temos: o tempo total de execução e a tabela de reserva de estágios. Para cada estágio: o tempo individual de execução e o tempo de transferência para os demais estágios. Para uma seqüência de operações: as tarefas (funções) envolvidas e o número de operandos vetoriais associados a cada tarefa.

O número de estágios, bem como o número de funções são informações de fácil compreensão. A sua escolha é uma das principais decisões a tomar em um projeto. O tempo individual de execução de um estágio é um conceito bastante simples e envolve o tempo físico de execução das operações referentes a este estágio. Entre cada dois estágios existe ainda um tempo de transmissão (transferência) de informações. O tempo total de execução de uma

determinada tarefa é a soma dos tempos de execução de cada estágio com o total dos tempos de transferência entre estes estágios. A tabela de reserva de uma função especifica os estágios que serão utilizados pela função e em que ordem deverão ser utilizados. Portanto, o cálculo do tempo total de execução de uma função é feito com base nas informações desta tabela. Um exemplo de tabela de reserva de estágios é mostrado na figura 3.

3. Análise Quantitativa de Arquiteturas Pipeline

Para efeitos de organização de informações os parâmetros de observação da análise são classificados em três níveis: estrutural, físico e comportamental. No nível estrutural, são definidas as características topológicas da arquitetura. No nível físico, as informações referentes a influência física dos dispositivos utilizados na implementação do hardware. E, por fim, no nível comportamental, são definidas as características esperadas para o software, quer seja o software de controle básico dos dispositivos físicos, quer seja o software da programação a ser aplicada na arquitetura.

Esta seção irá dividir-se em análise do caso ótimo e análise refinada. Na primeira, descreve-se a análise de arquiteturas pipeline desconsiderando a indução de tempos de espera no sequenciamento real das tarefas de uma determinada aplicação. Na análise refinada, considera-se não só os tempos de espera, mas busca-se um método de cálculo mais detalhado para o tempo de execução de uma tarefa, afim de obter resultados mais próximos da realidade.

3.1. Análise do Caso ótimo

A análise proposta por Hwang e Briggs em [HWA 85] (caso ótimo) refere-se a arquiteturas pipeline multifunção, estáticos e com capacidade de processamento de instruções vetorias (figura 1). Para esta análise é considerado um sub-conjunto dos parâmetros de observação citados na seção anterior.

Classificando-os conforme os níveis citados, temos:

- nível estrutural: número de estágios (k);

- nível físico:tempo total de execução de uma instrução (T);

- nível comportamental: número de instruções em uma aplicação (n) e para cada instrução da aplicação, tamanho do vetor de operandos utilizado (N_i).

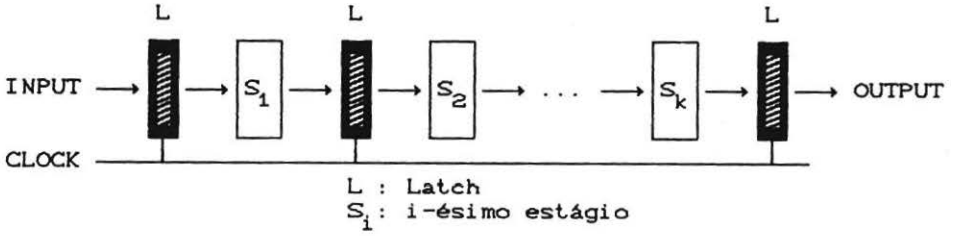


Figura 1 - Representação genérica de arquiteturas pipeline analisáveis pelo método de [HWA 85]

Observe que o parâmetro T é independente da instrução e, portanto, igual para todas. Não obstante, quando temos uma instrução vetorial ($N_i > 1$) o tempo T refere-se a execução da operação sobre um dos operandos. A diferença entre estas operações e as operações escalares ($N_i = 1$) é que as operações escalares não podem ter seu tempo de processamento sobreposto. Para instruções vetoriais, esta análise (caso ótimo) supõe que a execução sobre os elementos do vetor de operandos é sobreposta com a diferença de um estágio entre cada disparo de execução.

Observe ainda que outras simplificações foram feitas. Por exemplo, desconsidera-se o tempo de transferência entre os estágios e a possibilidade de reconfiguração. O tempo de transferência pode ser incluído no modelo somando-se o mesmo ao tempo médio de execução por estágio.

A primeira informação a ser calculada dos parâmetros citados é o tempo médio de execução de cada estágio (t):

$$t = T / k$$

O tempo para finalização da instrução i é obtido considerando-se k ciclos para preenchimento dos estágios do processador pipeline e a seguir $N_i - 1$ ciclos para execução dos operandos vetoriais restantes. Isto é resumido pela fórmula abaixo:

$$T_i = (k + N_i - 1) . t$$

Após a carga inicial do processador pipeline, o índice de desempenho (throughput - W) da arquitetura é dado pela razão entre o número de estágios e o tempo total de execução de uma tarefa, ou seja, o inverso do tempo dispendido em um estágio da arquitetura.

$$W = k / T = k / (t . k) = 1 / t$$

O tempo de execução de uma seqüência de n instruções de vetores, por sua vez é obtido somando-se os tempos gastos com a execução de cada instrução, ou seja:

$$Tp = \sum_{i=1}^{n} T_i = t . [(k - 1) . n + \sum_{i=1}^{n} N_i]$$

Para estabelecer o ganho de velocidade (speed-up) da execução da arquitetura pipeline sobre a execução sequencial é preciso que o tempo da execução sequencial (Ts) sobre as mesmas n instruções seja definido.

$$Ts = T . \sum_{i=1}^{n} N_i$$

Desta forma o ganho de velocidade (speed-up - S) é dado pela razão entre o tempo de execução sequencial e o tempo de execução do pipeline.

$$S = Ts / Tp$$

Maiores detalhes sobre a análise do caso ótimo podem ser encontrados em Hwang e Briggs [HWA 85].

3.2. Análise Refinada

A análise exposta anteriormente desconsidera diversos aspectos das arquiteturas pipeline fornecendo, em geral, resultados muito acima dos valores reais encontrados. Esta análise tem validade nos casos onde o interesse é apenas comparativo entre diferentes arquiteturas. Por outro lado, quando pretende-se comparar hardwares semelhantes com opções de software distintas esta análise não é suficientemente verossímil. Aspectos como o tempo perdido para reconfiguração e recarga do pipeline não podem ser desprezados quando é necessário avaliar a adaptabibilidade entre o software e o hardware. Neste sentido propõe-se uma análise que possa diferenciar ainda mais os aspectos referentes à adaptação entre software e hardware. São incluídas também especificações mais precisas para fatores já considerados. Por exemplo, na análise refinada existe a possibilidade de definição de tempos para a transferência entre os estágios do processador pipeline.

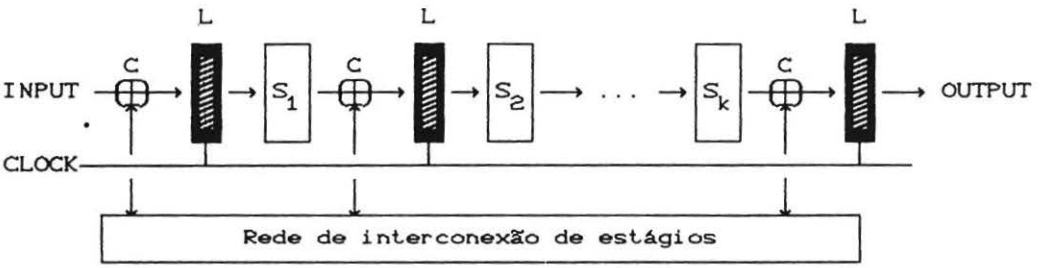
Os parâmetros de observação escolhidos são a seguir descritos de acordo com a classificação adotada. Cabe salientar que alguns parâmetros, como por exemplo a tabela de reserva das funções do pipeline, não são numéricos. Estes influenciam indiretamente a análise, ou seja, não são considerados diretamente nas fórmulas.

- nível estrutural: número de estágios (EP's) e possibilidades de comunicação entre os estágios;

- nível físico: tempo de processamento de cada estágio e tempo de transferência entre cada par de estágios;

- nível comportamental: número de funções executáveis, tabelas de reseva de estágios para cada função, número de operandos vetoriais para cada função e uma seqüência de funções.

A figura 2 representa genericamente as arquiteturas pipeline analisáveis pelo método proposto.



L : Latch
C : Chave de Re-direcionamento
S_i : i-ésimo estágio

Figura 2 - Representação genérica de arquiteturas pipeline analisáveis pelo método proposto

Definidos os parâmetros de observação da análise proposta, resta definir os parâmetros relevantes ao desempenho que serão calculados. Basicamente são calculados os mesmos parâmetros referidos na análise do caso ótimo. Além destes é fornecido um parâmetro que indique as diferenças de desempenho entre o caso ótimo e a análise refinada. Desta forma, temos os seguintes parâmetros quantitativos gerados pelo método de análise:

- índice de desempenho (throughput - W): taxa de instruções executadas por unidade de tempo;

- Ganho de desempenho (speed-up - S): ganho de desempenho (performance) da arquitetura pipeline frente a uma arquitetura sequencial;

- Influências do código (R): possível perda de desempenho frente ao índice de desempenho do caso ótimo.

A sistemática de cálculo destes parâmetros se assemelha em muito àquela utilizada na análise do caso ótimo. A diferença básica reside no fato de que na análise refinada o cálculo do tempo de execução de uma tarefa leva em conta a tabela de reserva

da respectiva função.

A nomenclatura adotada para esta análise utiliza as seguintes notações:

- k : número de estágios do processador pipeline;

- wp : tempo de execução de um estágio;

- wt : tempo de transferência entre dois estágios;

- n : número de funções executáveis

- $TEF_{i,j}$: número de ciclos que a função i usa na execução de j operandos vetoriais;

- N_i : número de operandos vetoriais para a função i;

- Tf_i : tempo total de execução da função i;

- Tp : tempo de execução da seqüência;

- To : tempo de execução ótimo da seqüência;

- Ts : tempo de execução sequencial das funções.

O parâmetros k, wp, wt, n e N_i são fornecidos diretamente como parâmetros de observação (entrada) pelo projetista. Outras informações fornecidas pelo projetista não são usadas diretamente no cálculo, mas sim na construção dos demais parâmetros, como por exemplo a tabela de reserva de estágios de cada função.

O Tempo de Execução de Função (TEF) para N_i operandos é obtido através da análise das tabelas de reserva de estágio, com as devidas considerações de sobreposição de tarefas e execução de funções sobre operandos vetoriais. Ou seja, o TEF deve ser tal, de forma que a execução de dois operandos vetoriais não utilize o mesmo estágio no mesmo ciclo. Assim, podemos calcular o tempo

total de execução para a instrução i:

$$Tf_i = TEF (i , N_i) * (wp + wt)$$

O cálculo de Tp é feito através da soma dos tempos envolvidos para execução das funções contidas na seqüência.

$$Tp = \left[\sum_{i=1}^{l(I)} Tf_i \right]$$

O cálculo do tempo de execução do caso ótimo é feito da mesma forma como foi descrito na seção anterior. Analogamente, o cálculo do tempo de execução sequencial é desenvolvido, desprezando totalmente a possibilidade de sobreposição de tarefas.

A partir dos valores já calculados, uma simples aplicação das seguintes fórmulas calcula os parâmetros de saída da análise.

$$W = \sum_{i=1}^{n} N_i / Tp$$

$$S = Ts / Tp \qquad R = Tp / To$$

4. Resultados obtidos

Nesta seção o método é exemplificado através de um caso prático. Para este exemplo, é feita a análise ótima e a análise refinada. Os resultados são exibidos através de gráficos, onde o parâmetro de saída é calculado em função da variação do parâmetro de entrada a ser avaliado. Esta forma de representação, gráficos em forma de curvas, é reconhecidamente de mais fácil interpretação, permitindo ao projetista a análise de um número maior de informações de maneira rápida e simples. Para tanto, definiu-se uma configuração básica e calculou-se o índice de

desempenho (throughput - figura 4) e o ganho de desempenho frente a execução sequencial (speed-up - figura 5) para o tempo de execução por estágio (t ou wp) variando de 1 até 10 unidades de tempo.

Configuração estudada:

- k = 5 estágios ;

- T = k.t
  T = 5.t unidades de tempo por instrução (caso ótimo);

- wp = t unidades de tempo (análise refinada);

- wt = .2 unidades de tempo (análise refinada);

- n = 5 instruções: a,b,c,d,e;

- operandos vetoriais associados a cada instrução: N(a)= 10, N(b)= 6, N(c)= 7, N(d)= 8, N(e)= 4;

- sequência de 5 instruções: a,b,c,d,e.

k \ t	1	2	3	4	5
1	a		b	c	c
2	b	ad	c	b	e
3	c	bc	ad	e	b
4	d	e	e	ad	
5	e				ad

Figura 3 – Tabela de reserva de estágios

Intuitivamente certos resultados são esperados. Por exemplo, é sabido que o aumento do tempo de execução por estágio diminui o índice de desempenho da arquitetura. Enquanto o ganho de desempenho frente a uma arquitetura sequencial aumenta com o

tempo de execução por estágio até um dado limite. Estes fatos podem ser verificados nas curvas das figuras 4 e 5.

A análise dos resultados mostra ainda a diferença entre as análises refinada e ótima. Note-se que a análise refinada apresenta índices de desempenho e ganho de velocidade inferiores ao caso ótimo. O uso das tabelas de reserva de estágios e a inclusão do tempo de transferência entre estágios no cálculo, permite alcançar resultados mais próximos da realidade.

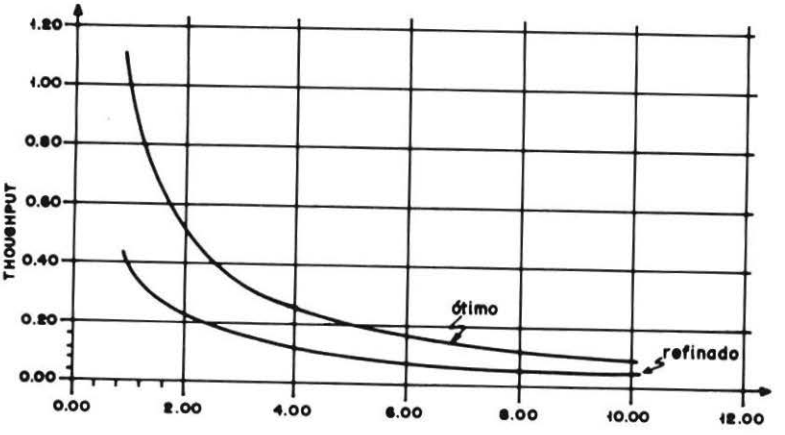


Figura 3 - Throughput x Número de estágios

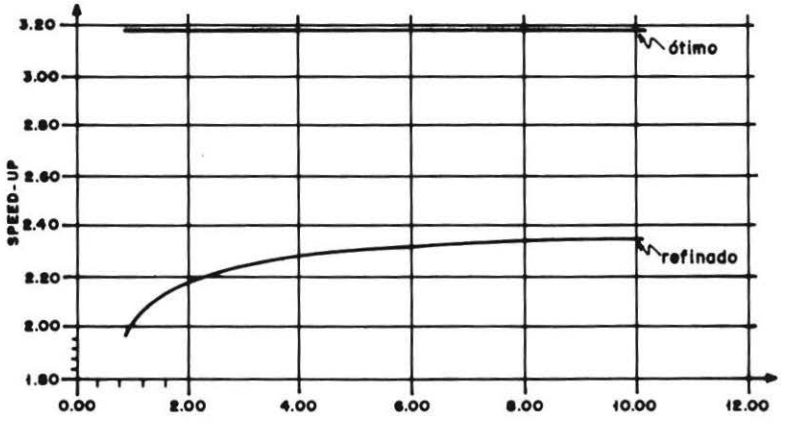


Figura 4 - Speed-up x Número de estágios

Tabela 1— Resultados obtidos

t	Análise refinada		Análise do caso ótimo	
	W	S	W	S
1	0.400	1.998	1.000	3.182
2	0.218	2.179	0.500	3.182
3	0.150	2.248	0.333	3.182
4	0.114	2.283	0.250	3.182
5	0.092	2.305	0.200	3.182
6	0.077	2.320	0.167	3.182
7	0.067	2.331	0.143	3.182
8	0.058	2.339	0.125	3.182
9	0.052	2.345	0.111	3.182
10	0.047	2.350	0.100	3.182

5. Conclusão

Foi demostrado e aplicado um método estocástico para avaliação de desempenho de arquiteturas pipeline, onde o comportamento das mesmas é a única base para análise. Obviamente esta prática só é utilizável em avaliações de sistemas pouco complexos. A dificuldade na elaboração do método para a análise proposta (refinada) comparada a dificuldade de elaboração do método para a análise do caso ótimo demostra o crescimento rápido da complexidade de elaboração de métodos de análise ad hoc.

A análise proposta, desenvolvida de maneira ad hoc, apresenta resultados confiáveis se comparados com os resultados do método apresentado em [HWA 85]. Este fato valida sua utilização para análise de casos reais.

Visando facilitar a utilização do método, no Curso de Pós-Graduação em Ciência da Computação (CPGCC-UFRGS) vem sendo implementada uma ferramenta ,PIPE, para automatizar sua aplicação. Tanto a implementação de PIPE, quanto o modelo de

análise apresentado, estão vinculados ao projeto Avaliação e Desempenho de Máquinas Paralelas (CPGCC-UFRGS) que propõe um ambiente integrado para avaliação de desempenho de máquinas paralelas.

6. Bibliografia

[FER 78] FERRARI, D. Computer systems performance evaluation. Englewood Cliffs, Prentice-Hall, 1978.

[GOE 63] GOETZ, B. E. Quantitative methods: a survey and a guide for managers. New York, McGraw-Hill, 1963.

[HAN 77] HÄNDLER, W. The impact of classification schemes on computer architecture. Proceedings of 1977 international conference on parallel processing, pp. 7-15.

[HWA 85] HWANG, K. & BRIGGS, F. Computer architecture and parallel processing. McGraw-Hill, New York, 1977.

[RAM 77] RAMAMOORTHY, C. & LI, H. Pipeline architectures. ACM Computing surveys, 9 (1): 61-102, New York, Mar. 1977.

[SVO 76] SVOBODOVA, L. Computer performance measurement and evaluation methods: analysis and applications. New York, Elsevier, 1976.