Efeito da Execução Especulativa no Desempenho e Balanceamento de Arquiteturas Super Escalares

Eliseu M. C. Filho
Edil S. T. Fernandes
Programa de Engenharia de Sistemas e Computação
COPPE/Universidade Federal do Rio de Janeiro
Caixa Postal 68511 21945-970 Rio de Janeiro, RJ
e-mail: {eliseu,edil}@cos.ufrj.br

Resumo

Execução especulativa é um dos principais mecanismos encontrados nos atuais processadores super escalares de alto desempenho. A execução especulativa torna mais eficiente a exploração do paralelismo a nível de instrução, antecipando a execução de instruções através das fronteiras dos blocos básicos. Neste trabalho, a execução especulativa é caracterizada através de um parâmetro arquitetural, a *profundidade de especulação*. É avaliado o efeito da profundidade de especulação sobre o desempenho e o balanceamento de arquiteturas super escalares. Os resultados indicam ganhos de desempenho de até 60% com profundidades de especulação elevadas. É também mostrado como os recursos da arquitetura devem ser dimensionados de forma que este ganho seja de fato concretizado.

Abstract

Speculative execution is a key concept in high-performance superscalar processors. It improves the exploitation of instruction-level parallelism by anticipating the execution of instructions across the boundaries of basic blocks. In this work, speculative execution is characterized by means of an architectural parameter referred to as the *speculation depth*. We evaluate the impact of the speculation depth on the performance and resource balancing of a typical superscalar architecture. The results indicate that performance gains of up to 60% can be achieved with a large speculation depths. It is also shown how the processor's resources should be dimensioned in order to obtain this improvement.

1 Introdução

Microprocessadores super escalares de alto desempenho tais como IBM-Motorola PowerPC 604 [1] e PowerPC 620 [2] e MIPS R10000 [3] incorporam mecanismos de suporte à execução especulativa. Em microarquiteturas sem execução especulativa, o despacho de instruções é bloqueado a cada instrução de desvio, voltando a operar somente após a resolução do desvio. Ao contrário, com execução especulativa, o despacho e a execução prosseguem com as instruções ao longo de um dos ramos do desvio, escolhido por um mecanismo de previsão de desvios. Caso esta previsão se revele correta, os resultados produzidos especulativamente alteram em definitivo o estado da computação; caso contrário, os resultados são descartados e o despacho é retomado com as instruções no ramo correto do desvio. A execução especulativa torna mais eficiente a exploração do paralelismo a nível de instrução, ao permitir a execução antecipada de instruções além das fronteiras de blocos básicos.

Na execução especulativa, existe um custo decorrente do cancelamento das instruções em ramos previstos incorretamente. Este custo depende da taxa de acerto do mecanismo de previsão de desvios. Mecanismos de previsão dinâmica de desvios [4], como os empregados nos processadores acima mencionados, fornecem uma taxa média de acerto de 85%. Novos mecanismos de previsão de desvios, como os descritos em [5] e [6], atingem taxas de acerto de até 97%. Com estas taxas de acerto, o custo associado a erros de previsão torna-se bastante pequeno, e o desempenho da arquitetura fica determinado principalmente pela eficiência do mecanismo de execução especulativa.

Além do efeito sobre o desempenho, a execução especulativa possui implicações sobre o balanceamento de recursos da arquitetura. Os recursos da unidade de execução (unidades funcionais, *buffers*, filas) devem ser dimensionados de forma adequada, para fazer face ao maior fluxo de instruções decorrente do não bloqueio do despacho. Estudos anteriores sobre configuração de recursos em arquiteturas super escalares ([7], [8], [9]) não consideram especificamente o efeito da execução especulativa sobre as exigências de recursos. No entanto, este relacionamento é crucial na obtenção de uma arquitetura corretamente balanceada.

Este trabalho apresenta resultados quanto ao efeito da execução especulativa sobre o desempenho e sobre o balanceamento de uma arquitetura super escalar típica. A execução especulativa será caracterizada por um parâmetro arquitetural denominado *profundidade de especulação*. A profundidade de especulação é definida como sendo o número de desvios não resolvidos através dos quais a execução especulativa prossegue, antes que o despacho seja bloqueado. Assim, mais precisamente, estaremos investigando o relacionamento entre a profundidade de especulação, o desempenho e o dimensio de recursos. Os atuais processadores super escalares adotam diferentes profundidades de especulação: a profundidade é dois desvios no PowerPC 604; quatro desvios no PowerPC 604 e no MIPS R10000; e 16 desvios no SPARC64 [10].

A continuação do artigo está organizada da seguinte forma. Na Seção 2 é descrito o modelo de arquitetura super escalar considerado neste estudo, enquanto na Seção 3 é apresentada a metodologia adotada na condução dos experimentos. Na Seção 4 são apresentados e discutidos os resultados. A Seção 5 finaliza o artigo com as conclusões.

2 Descrição do Modelo Super Escalar

O modelo super escalar aqui considerado é semelhante às microarquiteturas de processadores super escalares reais com execução especulativa. Como mostrado na Figura 1, o modelo é organizado em torno de um *pipeline* com sete estágios: busca, previsão, decodificação, despacho, expedição, execução e resultado.

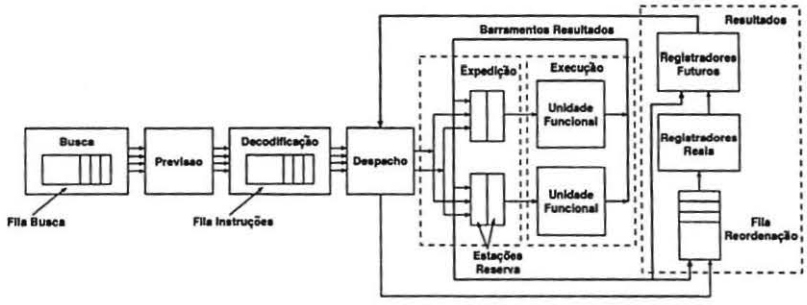

Figura 1: Organização do modelo super escalar.

O estágio de busca transfere instruções de uma memória *cache* para a fila de busca. O número máximo de instruções que podem ser acessadas em um mesmo ciclo é denominado *largura de busca*. O estágio de previsão transfere instruções da fila de busca para a fila de instruções, detectando instruções de desvio. Este estágio prevê o resultado de desvios condicionais através de um mecanismo de previsão dinâmica de desvios. O estágio de despacho envia instruções decodificadas, em ordem, da fila de instruções para as estações de reserva. Durante o despacho, este estágio verifica bits de reserva associados aos registradores-fonte de cada instrução. Se o bit estiver ativo, o registrador correspondente é destino de uma instrução despachada anteriormente, mas ainda não completada. Neste caso, um *tag* associado ao registrador indica a estação de reserva que armazena tal instrução. Este *tag* é copiado para a estação de reserva, ao invés do conteúdo do registrador. O estágio de despacho também ativa os bits de reserva associados aos registradores destino de cada instrução, e atualiza o *tag* destes registradores com a identificação da estação de reserva alocada à instrução. O número máximo de instruções que podem ser despachadas simultaneamente em um mesmo ciclo é denominado *largura de despacho*.

O estágio de expedição escalona instruções dinamicamente, operando de acordo

com o algoritmo de Tomasulo [11] (mecanismo idêntico é encontrado no PowerPC 604 e 620). O estágio de execução contém as unidades funcionais. O estágio de resultado envia o resultado de uma instrução completada, juntamente com o *tag* da estação de reserva que contém a instrução, para o conjunto de registradores e para as estações de reserva. O resultado é armazenado no registrador e nas estações de reserva com *tags* coincidentes. Instruções aguardando nas estações de reserva podem se tornar aptas para expedição após cada escrita de resultados. Resultados e *tags* são enviados através de barramentos de resultados.

Uma fila de reordenação e um conjunto de registradores futuros [12] são empregados para suportar execução especulativa (novamente, suporte semelhante é adotado em processadores reais). O estágio de despacho lê operandos-fonte a partir dos registradores futuros. Para cada instrução, o estágio de despacho aloca uma entrada na fila de reordenação. O estágio de resultado armazena resultados nos registadores futuros e nas entradas apropriadas da fila de reordenação. Este estágio transfere o resultado da fila de reordenação para um registrador no conjunto de registradores reais (ver Figura 1) somente quando a entrada com este resultado atingir a cabeça da fila de reordenação. Este procedimento garante que os registradores reais não serão modificados por instruções executadas especulativamente ao longo de ramos previstos incorretamente, já que o desvio previsto incorretamente precede (na fila de reordenação) as instruções executadas especulativamente.

Instruções de desvio são executadas, em ordem, por uma unidade de desvios dedicada. No caso de um erro de previsão, a unidade de desvios bloqueia o estágio de despacho, limpa as filas de busca e de instruções, e redireciona o estágio de busca para o ramo correto do desvio. O despacho permanece bloqueado até que o desvio previsto incorretamente chegue à cabeça da fila de reordenação. Neste momento, o estágio de resultado descarta as instruções que ainda estejam ocupando as estações de reserva, as unidades funcionais e a fila de reordenação, copia o conteúdo dos registradores reais para o registradores futuros, e libera o estágio de despacho.

Os experimentos para este trabalho foram realizados com um simulador *trace-driven* que reproduz a operação do modelo super escalar. Este simulador recebe como entrada os valores dos principais parâmetros arquiteturais e produz diversas estatísticas sobre o desempenho da arquitetura. O simulador aceita programas compilados para a arquitetura SPARC Versão 7 [13]. Os *traces* foram gerados por um outro simulador, que reproduz a operação de uma implementação escalar da arquitetura SPARC. As latências adotadas para as instruções são idênticas às do PowerPC 604.

3 Metodologia

Avaliando o Efeito sobre o Desempenho. Para avaliar a relação entre execução especulativa e desempenho, empregamos uma arquitetura cujos recursos foram dimensionados de forma a não limitar o desempenho. Esta configuração foi determinada, em uma fase preliminar, da seguinte maneira. Com valores elevados para a largura de busca e para os tamanhos das filas de busca e de instruções, foi medida a proporção de bloqueios no despacho provocados por insuficiência na largura de despacho, no número de unidades funcionais e de estações de reserva e no tamanho da fila de reordenação. O número de barramentos de resultados foi tomado como igual ao de unidades funcionais. A configuração ótima obtida é descrita na Tabela 1. Usando esta configuração, foi medido o desempenho para diferentes valores da profundidade de especulação.

Tabela 1: Parâmetros arquiteturais na avaliação de desempenho.

Parâmetro	Valor
Largura busca	8 instruções/ciclo
Fila busca	64 entradas
Fila instrução	64 entradas
Largura despacho	8 instruções/ciclo
Unidades funcionais	8 unidades de inteiros
	4 unidades de ponto-flutuante
Estações de reserva	16 por unidade funcional
Barramentos resultados	8 barramentos
Fila reordenação	1024 entradas

Como mencionado na Seção 1, o impacto da execução especulativa sobre o desemepnho é influenciado pela taxa de acerto do mecanismo de previsão de desvios. Para levar em consideração este fator, foram considerados dois diferentes mecanismos de previsão de desvios na avaliação do efeito sobre o desempenho. O primeiro usa uma BHT (*Branch History Table*) com 512 entradas e um contador crescente-descrescente de dois bits como autômato de previsão [4]. Para os programas de teste usados, este mecanismo apresentou uma taxa média de acerto de 87%. Pan, So e Rahmeh [5], bem como Yeh e Patt [6], propuseram mecanismos de previsão dinâmica de desvios em dois níveis, que melhoram a taxa média de acerto de previsão capturando correlações existentes entre desvios. Yeh e Patt reportam taxas de acerto de até 97% [6]. Para aproximar o efeito da execução especulativa na presença de taxas de acerto quase ideais, foi também considerado um mecanismo perfeito de previsão de desvios. Os resultados obtidos com a previsão perfeita indicam o limite superior do efeito da execução especulativa sobre o desempenho.

Avaliando a Influência sobre o Balanceamento. Sob este aspecto, serão aqui apresentadas as exigências da execução especulativa sobre o número de estações de reserva (em [14] são apresentados resultados mostrando as exigências sobre outros recursos da arquitetura, e.g., a fila de reordenação) . Para evitar um número excessivo de bloqueios do despacho por falta de recursos, o número de estações de reserva deve ser escolhido de modo a acomodar o fluxo de instruções resultante de uma certa combinação da largura de despacho e da profundidade de especulação. Assim, o objetivo é encontrar o balanceamento entre a profundidade de especulação, a largura de despacho e número de estações de reserva que minimize a freqüência de bloqueios do despacho.

Este balanceamento foi investigado para as configurações de arquitetura relacionadas na Tabela 2. Estas configurações são semelhantes às encontradas em processadores super escalares reais. Largura de despacho de quatro instuções/ciclo são encontradas no PowerPC 604 e 620. O R10000 inclui duas unidades de inteiros e uma unidade de memória, enquanto o PowerPC 604 e 620 possuem três unidades de inteiro e uma de memória. Para as configurações na Tabela 2, foi medida a freqüência de bloqueios do despacho em função do número de estações de reserva e da profundidade de especulação. Estas medidas foram realizadas para arquiteturas com mecanismo de previsão de desvios real (BHT).

Tabela 2: Configurações na avaliação do balanceamento.

Denominação	Configuração
D4-I2-M1	largura despacho = 4, 2 unidades inteiras, 1 unidade memória
D4-I3-M1	largura despacho = 4, 3 unidades inteiras, 1 unidade memória
Parâmetros comuns em todas configurações	
Largura busca	8 instruções/ciclo
Fila de busca	64 entradas
Fila de instruções	64 entradas

Programas de Teste. Nos experimentos, foram usados oito programas inteiros e seis programas de ponto flutuante dos conjuntos SPEC92 e SPEC95, listados na Tabela 3. Os programas inteiros foram compilados com o compilador gcc 2.7.2, e os programas de ponto flutuante com o compilador fgcc 0.5.18, todos para o sistema operacional SunOS 4.1.4. Os *traces* empregados cobrem a execução de 20 milhões de instruções. O número de instruções de desvios executadas dentro deste total pode ser visto na Tabela 3.

Tabela 3: Programas de teste

Programas inteiros	Desvios executados	Programas ponto-flutuante	Desvios executados
go	2,155,140	applu	122,335
m88ksim	3,313,582	fpppp	171,995
cc1	3,954,491	tomcatv	194,841
compress	2,242,992	swim	106,095
ijpeg	4,338,049	mgrid	1,334,134
vortex	3,604,077	turb3d	1,113,499
espresso	3,543,002		
eqntott	3,781,182		

4 Resultados

4.1 Efeito sobre o Desempenho

As Figuras 2 e 3 mostram o efeito da execução especulativa (caracterizada pela profundidade de especulação) sobre o desempenho de programas inteiros, para a previsão de desvios com BHT e previsão de desvios perfeita, respectivamente. Como métrica de desempenho, foi tomada a porcentagem de redução do número total de ciclos de execução, em relação a uma arquitetura-referência que não suporta execução especulativa (i.e., o despacho é bloqueado a cada instrução de desvio encontrada). Esta arquitetura-referência apresenta uma organização similar à descrita na Seção 2, a menos dos componentes relacionados com execução especulativa (registradores futuros e fila de reordenação). A arquitetura-referência possui configuração de recursos idêntica à da arquitetura especulativa avaliada (Tabela 1).

Para previsão de desvios com BHT, a redução média do número de ciclos é de 28% com uma especulação através de dois desvios (como no PowerPC 604), e de 40% para uma profundidade de quatro desvios (como no PowerPC 620 e MIPS R10000). Uma redução média de 48% é obtida elevando a profundidade de especulação para oito desvios. Os únicos programas que não se beneficiam com uma profundidade de especulação elevada foram o `ijpeg` e o `eqntott`, para os quais o ganho de desempenho se estabilizou com profundidades de especulação de três e cinco desvios respectivamente. Os ganhos adicionais obtidos com especulações além de oito desvios são menos pronunciados. No entanto, no caso de quatro programas (`go`, `m88ksim`, `compress` e `espresso`, reduções no número de ciclos entre 55% e 60% são obtidas para uma profundidade de especulação de 10 desvios. A Figura 3 mostra que profundidades de especulação elevadas (acima de oito desvios) podem produzir ganhos sensíveis na presença de um mecanismo de previsão mais preciso. Para uma profundidade de especulação de oito

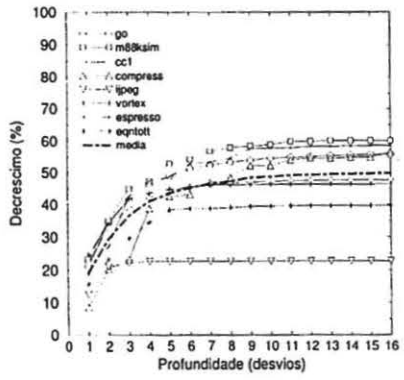

Figura 2: Inteiros, previsão com BHT

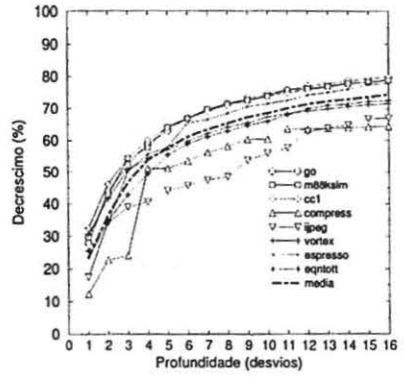

Figura 3: Inteiros, previsão perfeita

desvios, a redução média no número total de ciclos é de 65%. Uma redução média no número de ciclos de 70% é obtida para uma profundidade de especulação de 12 desvios.

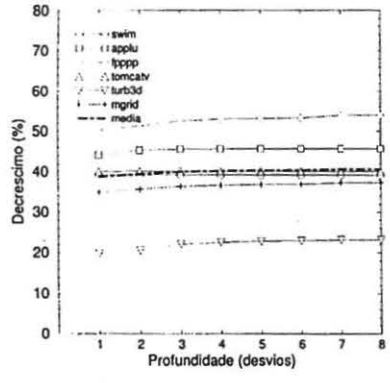

Figura 4: Ponto flutuante, previsão BHT

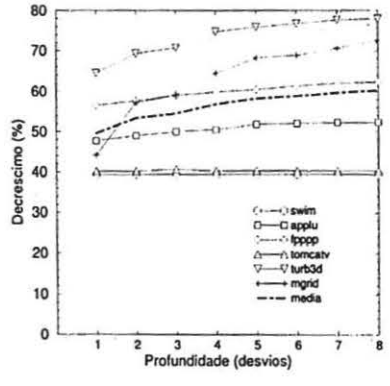

Figura 5: Ponto flutuante, previsão perfeita

As Figuras 4 e 5 mostram o efeito da execução especulativa sobre o desempenho dos programas de ponto flutuante, para previsão com BHT e previsão perfeita, respectivamente. Na previsão com BHT, observa-se uma redução média de 49% no número total de ciclos para especulação através de um desvio, sendo que aumentos na profundidade de especulação não produzem ganhos adicionais significativos. Na maioria dos programas (com exceções abaixo mencionadas), este comportamento deve-se à pequena densidade de desvios executados (veja Tabela 3), típica programas de ponto flutuante

[15]. Devido a esta característica, na maioria dos casos um desvio é resolvido antes que novas instrucoes de desvio cheguem para despacho. Assim, a simples introdução de um único nível de especulação é suficiente para eliminar a maior parte dos bloqueios de despacho devido à instruções de desvio. Este fato é demonstrado pela redução média inicial de 40% obtido com a introdução de um nível de especulação (para programas inteiros, este degrau inicial foi de no máximo 25%).

Comportamento semelhante é observado para a arquitetura com previsão de desvios perfeita (Figura 5), exceto em dois casos. Para os programas `mgrid` e `turb3d`, ganhos adicionais são obtidos para profundidades de especulação de até oito desvios. A razão para esta diferença está na maior contagem dinâmica de desvios apresentada por estes dois programas (ver Tabela 3). O mesmo comportamento não é observado para o mecanismo de previsão com BHT porque, coincidentemente, a BHT apresenta uma baixa taxa de acerto para estes dois programas (78% para `mgrid` e 43% para `turbo3d`). Neste caso, o custo de anulação de instruções passa a dominar, anulando o efeito da especulação. Com os erros de previsão retirados (na previsão perfeita), o benefício de uma maior profundidade de especulação torna-se visível.

4.2 Efeito sobre o Balanceamento

O principal benefício obtido com a execução especulativa (ou, mais precisamente, com uma maior profundidade de especulação) é a redução dos bloqueios do despacho causados por desvios pendentes (i.e., não resolvidos). Entretanto, este benefício pode não se realizar caso o despacho de instruções seja bloqueado freqüentemente por falta de recursos. Assim, a ocorrência de bloqueios do despacho é o fator central a ser considerado na avaliação do balanceamento entre especulação e recursos.

Em arquiteturas semelhantes a aqui considerada, bloqueios do despacho são provocados por um seguintes fatores: (1) não existe estação de reserva disponível; (2) não existe entrada disponível na fila de reordenação; (3) a profundidade de especulação é insuficiente; (4) foi encontrado um desvio previsto incorretamente; e (5) a fila de instruções está vazia. Bloqueios do despacho devido a profundidade de especulação insuficiente ocorrem sempre que o número de desvios pendentes é igual a profundidade e um novo desvio chega para despacho. Erros de previsão provocam o bloqueio do despacho porque instruções do ramo correto não podem ser despachadas enquanto o estado da arquitetura não for corrigido.

Bloqueios do despacho foram quantificados medindo a freqüência de ciclos com despacho nulo, i.e., ciclos em que nenhuma instrução é despachada. Para as configurações listadas na Tabela 3, apresentamos dois conjuntos de resultados, que diferem no modo como ciclos despacho nulo foram medidos:

- no primeiro conjunto (Figuras 6 e 7 a seguir), ciclos com despacho nulo foram

medidos em função da profundidade de especulação e do número de estações de reserva por unidade funcional de inteiros (estações de inteiros). O número de estações de reserva por unidade funcional de memória (estações de memória) e o tamanho da fila de reordenação foram fixados em valores altos;

- no segundo conjunto (Figuras 8 e 9), ciclos com despacho nulo foram medidos em função da profundidade de especulação e do número de estações de memória. O número de estações de inteiros foi fixado no melhor valor determinado a partir dos resultados anteriores, e o tamanho da fila de reordenação foi mantido em um valor alto.

Nos gráficos que seguem, cada barra empilhada indica a contribuição dos fatores (1), (3), (4) e (5) acima mencionados para a porcentagem de ciclos com despacho nulo, dentro do número total de ciclos. Cada barra corresponde a um certo número de estações de reserva (valor $n\epsilon$ indicado acima da barra), e cada grupo de barras empilhadas corresponde a um certo valor da profundidade de especulação. As porcentagens mostradas representam a média dos programas de inteiros. Em todos os casos, foi usado o mecanismo de previsão de desvios com BHT.

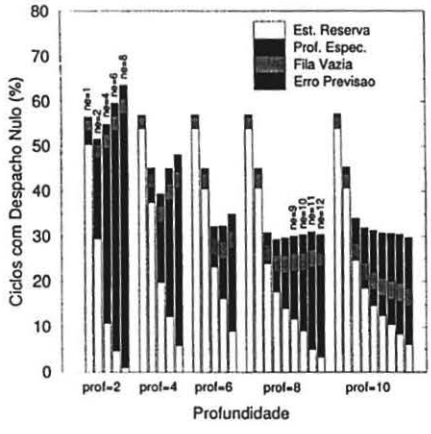

Figura 6: Configuração D4-I2-M1.

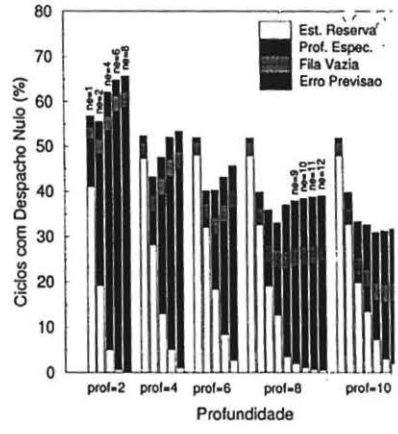

Figura 7: Configuração D4-I3-M1.

As Figuras 6 e 7 mostram os resultados do primeiro conjunto (número de estações de inteiros e profundidade de especulação são as variáveis). Na Figura 6, observe inicialmente o alto número de bloqueios do despacho quando a profundidade de especulação é pequena. A proporção de ciclos com despacho nulo varia entre 39% a 64% do número total de ciclos quando a profundidade de especulação é menor ou igual a quatro desvios. Os componentes que mais contribuem são a indisponibilidade de estações de reserva

e a profundidade de especulação insuficiente. Para pequenas profundidades de especulação, as parcelas relacionadas com erros de previsão e com fila de instruções vazia não ultrapassam 4% do total de ciclos, cada.

Um aspecto importante a ser observado na Figura 6 é o interrelacionamente entre bloqueios do despacho associados com a profundidade de especulação e com o número de estações de reserva. Por exemplo, considere a situação quando a profundidade de especulação é quatro desvios. A porcentagem de ciclos com despacho nulo provocados pela indisponibilidade de estações de reserva decresce de 52% quando existe uma estação, para apenas 5% quando existem oito estações. Por outro lado, com este aumento no número de estações de reserva, a proporção de ciclos com despacho nulo devido a profundidade de especulação insuficiente passa de 0.1% para 30%. Este comportamento decorre do fato que a adição de estações de reserva permite um aumento do número de instruções despachadas por ciclo, incluindo instruções de desvio. À medida que um número maior de desvios é despachado dentro de um certo intervalo de tempo, torna-se mais freqüente os casos em que a profundidade de especulação é atingida antes que o desvio mais antigo seja resolvido.

Note que, se aumentarmos a profundidade de especulação na tentativa de compensar este efeito, os bloqueios devido à indisponibilidade de estações de reserva voltam a aumentar. Assim, o balanceamento adequado não pode ser obtido agindo somente sobre a profundidade de especulação ou o número de estações de reserva, mas sobre estes dois parâmetros simultaneamente. Ambos devem ser ajustados até que a proporção de ciclos com despacho nulo seja minimizada. Para as configurações da Figura 6, isto acontece com uma profundidade de especulação de 8 desvios e com 8 estações de reserva por unidade de inteiros. Um outro mínimo é obtido com uma profundidade de especulação de 10 desvios, mas neste caso são necessárias 12 estações de reserva por unidade de inteiros.

Note ainda que a parcela relativa aos erros de previsão impede que o número total de ciclos com despacho nulo diminua com profundidades de especulação maiores que 10 desvios. Isto acontece porque, com uma profundidade maior, aumenta o número médio de instruções precedendo o desvio previsto incorretamente na fila de reordenação. Em conseqüência, em média o despacho permanece bloqueado por um intervalo maior até que o estado da arquitetura seja corrigido.

A Figura 7 mostra o comportamento quando é acrescentada uma terceira unidade funcional de inteiros. Comparando com a Figura 6, nota-se que a porcentagem de ciclos com despacho nulo aumenta em cerca de 10% para uma profundidade de especulação menor ou igual a seis desvios, sendo este aumento menor para profundidades de especulação maiores. Neste caso, a porporção de ciclos com despacho nulo é minimizada com uma profundidade de especulação de 10 desvios, e com 8 estações de reserva por unidade de inteiros.

As Figuras 8 e 9 mostram o segundo grupo de resultados (número de estações de memória variável). Neste caso, foram usados valores de profundidade de especulação e número de estações de inteiros que, segundo os resultados anteriores, minimizam a proporção de ciclos com despacho nulo. O objetivo agora é determinar o número de estações de memória, de modo que a parcela de ciclos com despacho nulo relacionada com este recurso seja comparável às parcelas mínimas relacionadas com a profundidade de especulação e com o número de estações de inteiros (estas em torno de 10%, conforme observado nos gráficos das Figuras 6 e 7).

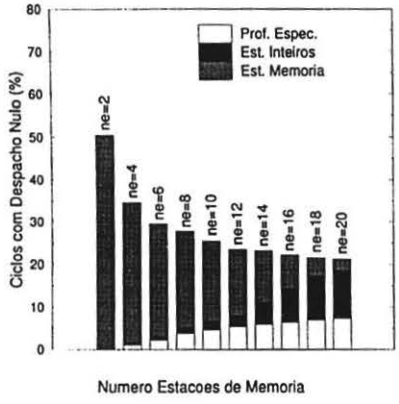

Figura 8: Configuração D4-I2-M1.

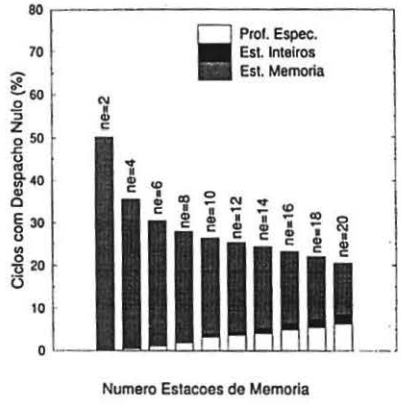

Figura 9: Configuração D4-I3-M1.

A Figura 8 mostra que a proporção de ciclos com despacho nulo provocados pela indisponibilidade de estações de memória fica abaixo de 10% somente se o número de estações de memória for maior que 14 estações. Comportamento semelhante é observado na Figura 9, para configuração com uma unidade de inteiros adicional. A razão para o alto número de estações de memória que se faz necessário está na maior latência das instruções de memória (dois ciclos).

5 Conclusões

Neste trabalho, foi quantificada a influência da execução especulativa sobre o desempenho e sobre o dimensionamento dos recursos de uma arquitetura super escalar típica. Observamos que o desempenho de programas inteiros é sensivelmente afetado pela profundidade de especulação. Usando um mecanismo de previsão dinâmica de desvios com BHT, obtivemos uma redução média na contagem de ciclos de 18% para a profundidade

de especulação igual à do PowerPC 604 (dois desvios) e de 40% para a profundidade de especulação igual à do PowerPC 620 (quatro desvios). Não foram observados ganhos adicionais significativos para níveis de especulação acima de oito desvios. No entanto, os resultados obtidos com o mecanismo perfeito de previsão indicam que o impacto de profundidades de especulação elevadas pode ser mais pronunciado se a previsão for mais precisa. Como limite superior, obtivemos uma redução média no número de ciclos de 70% para uma profundidade de especulação de 12 desvios. Programas de ponto flutuante se mostraram menos sensíveis à profundidade de especulação. Para a previsão de desvios com BHT, foi obtida uma redução média no número de ciclos de 40% ao ser introduzida execução especulativa através de um único desvio; porém, o desempenho manteve-se essencialmente o mesmo para profundidades de especulação maiores.

Ao compararmos a contribuição de cada um dos fatores que provocam ciclos com despacho nulo, observamos que o custo associado a profundidade de especulação insuficiente pode ser bem maior que o custo associado a erros na previsão. Nas configurações de arquitetura consideradas, ciclos com despacho nulo consumidos pelas correções de estado não ultrapassaram 12% do total de ciclos. Em contraste, para a configuração com três unidades de inteiros, uma unidade de memória e profundidade de especulação de dois desvios (a configuração do PowerPC 604), a proporção de ciclos provocados por profundidade de especulação insuficiente chega a 30%. Por outro lado, o nível de especulação não pode ser aumentado indefinidamente. A partir de um certo ponto, ciclos com despacho nulo associados a correções de estado passam a dominar. Para previsão de desvios com BHT, observou-se que este limiar situa-se em uma profundidade de especulação de 10 desvios.

É essencial que aumentos na capacidade de especulação de uma arquitetura super escalar sejam acompanhados de um dimensionamento adequado de recursos. Por exemplo, seja a configuração com três unidades de inteiros e uma de memória, com duas estações de reserva por unidade. Aumentando a profundidade de especulação de dois para quatro desvios (equivalente à passagem do PowerPC 604 para o PowerPC 620), a proporção de ciclos com despacho nulo associada à profundidade de especulação insuficiente cai de 30% para menos de 10%. No entanto, mantendo o mesmo número de estações de reserva (como acontece no PowerPC 604 e 620), tal mudança eleva de 18% para 30% a proporção de ciclos com despacho nulo associada com insuficiência de recursos. Aumentar um certo recurso individualmente não resulta em melhoria; recursos e profundidade de especulação devem ser ajustados simultaneamente. Observou-se que na configuração de unidades funcionais semelhante à do PowerPC 604, a ocorrência de ciclos com despacho nulo é minimizada somente com uma profundidade de especulação de 10 desvios, e com 8 estações de inteiros e 20 estações de memória.

6 Referências Bibliográficas

[1] Song, S. P., M. Denman, J. Chang, *The PowerPC 604 RISC Microprocessor*, IEEE Micro, Vol. 14, N. 5, October 1994, pp. 8-17.

[2] Diep, T. A., C. Nelson, J. P. Shen, *Performance Evaluation of the PowerPC 620 Microarchitecture*, Proc. of the 22th International Symposium on Computer Architecture, 1995, pp. 163-175.

[3] MIPS Inc., *The R10000 Microprocessor User's Manual*, 1995.

[4] Lee, J. K. F., A. J. Smith, *Branch Prediction Strategies and the Branch Target Buffer Design*, IEEE Computer, Vol. 17, N. 1, September 1980, pp. 261-294.

[5] Pan, S-T., K. So, J. T. Rahmeh, *Improving the Accuracy of Dynamic Branch Prediction Using Branch Correlation*, Proc. of the 5th Symposium on Architectural Support for Programming Languages and Operating Systems, 1992, pp. 76-84.

[6] Yeh, T.-Y., Patt, Y., *Two-Level Adaptative Training Branch Prediction*, Proc. of the 24th Annual International Symposium on Microarchitecture, 1991, pp. 51-61.

[7] Johnson, M., *Superscalar Microprocessor Design*, Prentice Hall, Englewood Cliffs, NJ, 1991.

[8] Jourdan, S., *Exploring Configurations of Functional Units in an Out-of-Order Superscalar Processor*, Proc. of the 24th International Symposium on Computer Architecture, 1991, pp. 117-125.

[9] Jourdan, S., *An Investigation of the Performance of Various Instruction Issue Buffer Topologies*, Proc. of the 28th Annual International Symposium on Microarchitecture, 1995, pp. 279-284.

[10] Williams, T., N. Patkar, G. Shen, *SPARC64: A 64-b 64-Active-Instruction Out-of-Order-Execution MCM Processor*, IEEE Journal of Solid State Circuits, Vol. 30, N. 11, November 1995, pp. 1215-1226.

[11] Tomasulo, R. M., *An Efficient Algorithm for Exploiting Multiple Arithmetic Units*, IBM Journal of Research and Development, Vol. 11, N. 1, January 1967, pp. 25-33.

[12] Smith, J. E., A. R. Pleszkun, *Implementing Precise Interrupts in Pipelined Processors*, IEEE Transactions on Computers, Vol. 37, N. 55, May 1988, pp. 562-573.

[13] Sun Microsystems, *The SPARC Architecture Manual, Version 7*, Mountain View, CA, 1987.

[14] *The Effect of the Speculation Depth on the Performance of Superscalar Architectures*, Technical Report, 1996.

[15] Knuth, D., *An Empirical Study of FORTRAN Programs*, Software - Practice and Experience, Vol. 1, N. 1, 1971, pp. 105-133.