Arquitetura de Computadores

Computadores com um Conjunto Reduzido de Instruções

Grandes Avanços(1)

- O conceito de família
  - IBM System/360 1964
  - DEC PDP-8
  - Separa arquitetura de implementação
- Unidade de controle Microprogramada
  - Idéia de Wilkes 1951
  - Produzida pela IBM S/360 1964
- Memória Cache
  - IBM S/360 modelo 85 1969

Grandes Avanços(2)

- Microprocessadores
  - Intel 4004 1971
- Pipelining
  - Introduz paralelismo no ciclo de busca e execução
- Múltiplos processadores

O próximo passo- RISC

- Reduced Instruction Set Computer

- Características principais
  - Grande número de registradores de propósito geral
  - ou uso de compiladores para otimizar o uso de registradores
  - Conjunto de instrução limitado e simples
  - Ênfase na otimização do pipeline de instruções

Comparação de processadores

CISC			**RISC**		**Superscalar**	
IBM	DEC VAX	Intel	Motorola	MIPS	IBM	Intel
370/168	11/780	486	88000	R4000	RS/6000	80960
1973	1978	1989	1988	1991	1990	1989
No. of instruction						
208	303	235	51	94	184	62
Instruction size (octets)						
2-6	2-57	1-11	4	32	4	4 or 8
Addressing modes						
4	22	11	3	1	2	11
GP registradores						
16	16	8	32	32	32	23-256
Control memory (k bytes) (microprogramming)						
420	480	246	0	0	0	0

Esforços do CISC

- Custos de software excedem em muito o custo de hardware
- Linguagens de alto nível cada vez mais complexas
- *Gap* semântico levou a:
  - Grandes conjuntos de instruções
  - Mais modos de endereçamento
  - Implementação em hardware comandos de linguagem de alto nível
    - ex. CASE (switch) no VAX

Intenção do CISC

- Facilitar a escrita do compilador
- Melhorar a eficiência da execução
  - Operações complexas em microcódigo
- Dar suporte a HLLs (*High Level Languages*) mais complexas

Características da execução

- Operações realizadas
- Operandos usados
- Sequenciamento da execução
- Estudos foram feitos baseados em programas escritos em HLLs
- Estudos dinâmicos mediram durante a execução do programa

Operações

- Atribuições
  - Movimentação de dados
- Sentenças condicionais(IF, LOOP)
  - Controle da sequência
- Call-return de procedimentos consome muito tempo
- Algumas instruções HLL levaram à máquina codificar operações

Frequência Dinâmica Relativa

	Dynamic Occurrence		Machine Instruction (Weighted)		Memory Reference (Weighted)	
	Pascal	C	Pascal	C	Pascal	C
Assign	45	38	13	13	14	15
Loop	5	3	42	32	33	26
Call	15	12	31	33	44	45
If	29	43	11	21	7	13
GoTo	-	3	-	-	-	-
Other	6	1	3	1	2	1

Operandos

- Principalmente variáveis locais escalares
- Otimização deveria concentrar no acesso de variáveis locais

	Pascal	C	Média
Constante Inteira	16	23	20
Variável Escalar	58	53	55
Array/structure	26	24	25

Chamadas de Procedimentos

- Muito demorada
- Depende do número de parâmetros passados
- Depende do nível de aninhamento
- Maioria dos programas não fazem muitas chamadas seguida de muitos retornos
- Maioria das variáveis são locais
- (c.f. localidade de referência)

Implicações

- Melhor suporte é dado otimizando as características que são mais usadas ou que mais consomem tempo
- Grande número de registradores
  - Referência de operando
- Projeto cuidadoso de pipelines
  - Branch prediction (previsão de desvio) etc.
- Conjunto de instruções simplificado (reduzido)

Grande Banco de Registradores

- Solução em Software
  - Requer que o compilador aloque registradores
  - Alocação é baseada nas variáveis mais utilizadas em um dado momento
  - Requer análise sofisticada do programa
- Solução em Hardware
  - Tem mais registradores
  - Assim mais variáveis estarão em registradores

Registradores para variáveis locais

- Armazenam variáveis locais escalares em registradores
- Reduz o acesso à memória
- Cada chamda de procedimento (função) muda a localidade
- Parâmetros devem ser passados
- Resultados devem ser retornados
- Variáveis de programas devem ser restauradas

Janelas de registrador

- Apenas poucos parâmetros
- Variação limitada da profundidade de chamada
- Uso de múltiplos pequenos conjuntos de registradores
- Cada chamadas é usado um conjunto diferente de registradores
- Retornos levam à troca de conjunto de registradores previamente utilizado

Janelas de registrador cont.

- Três áreas dentro de um conjunto de registrador
  - Registradores de parâmetros
  - Registradores locais
  - Registradores Temporários

Sobreposição de Janelas de Registradores

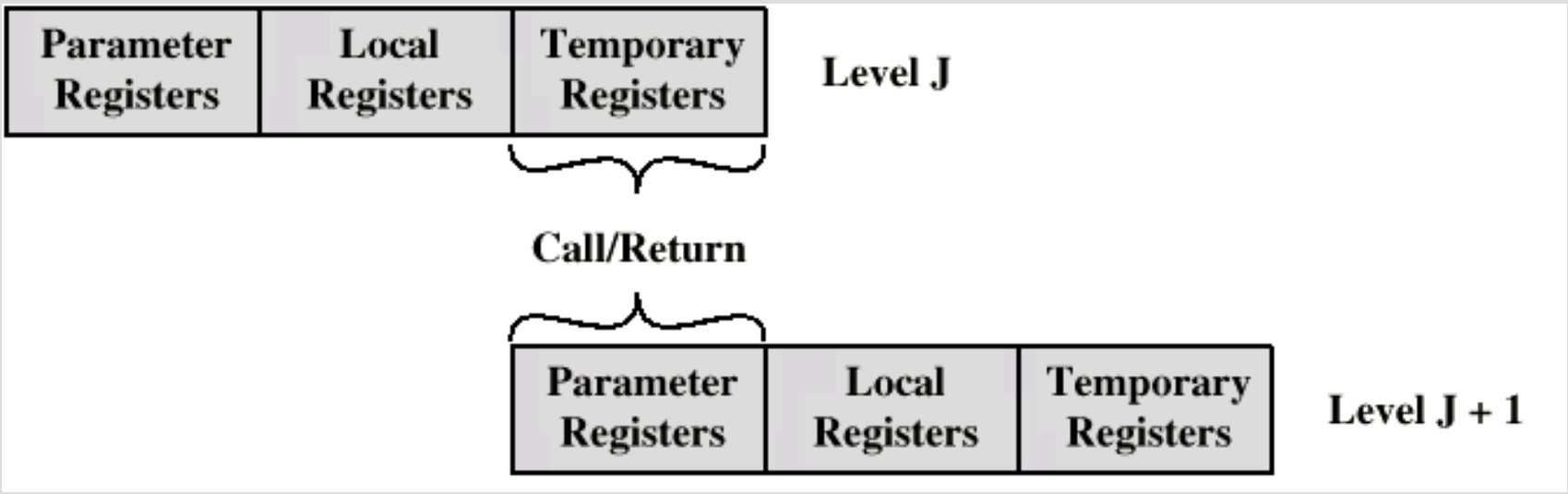
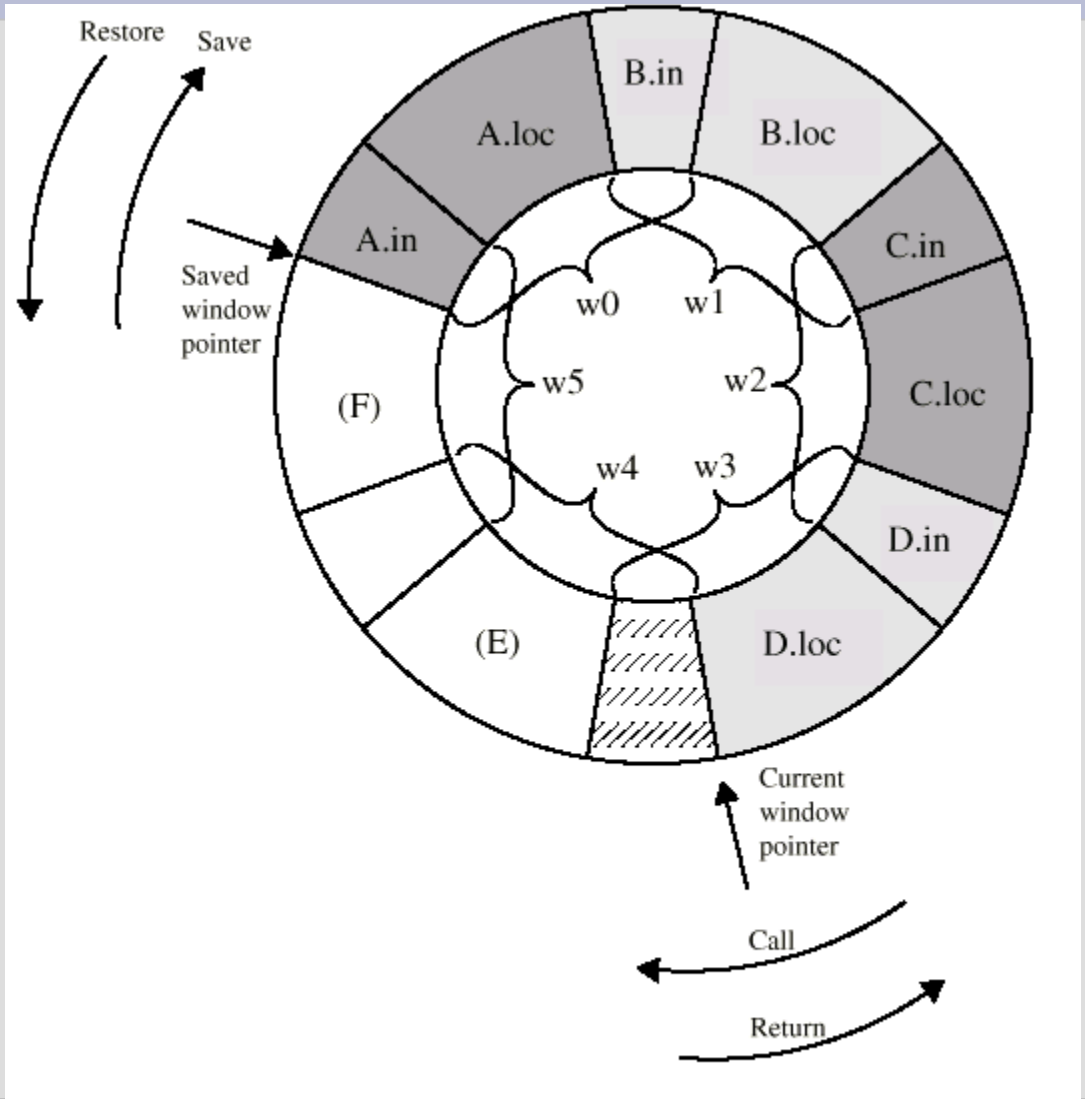


Diagrama de Buffer Circular



Operação do Buffer Circular

- Quando uma chamada é feita, o ponteiro da janela atual é movido para mostrar a janela ativa no momento
- Se todas as janelas estão sendo usadas, uma interrupção é gerada e a janela mais velha é salva em memória
- Um ponteiro de janela salva indica onde a próxima janela salva deve ser restaurada

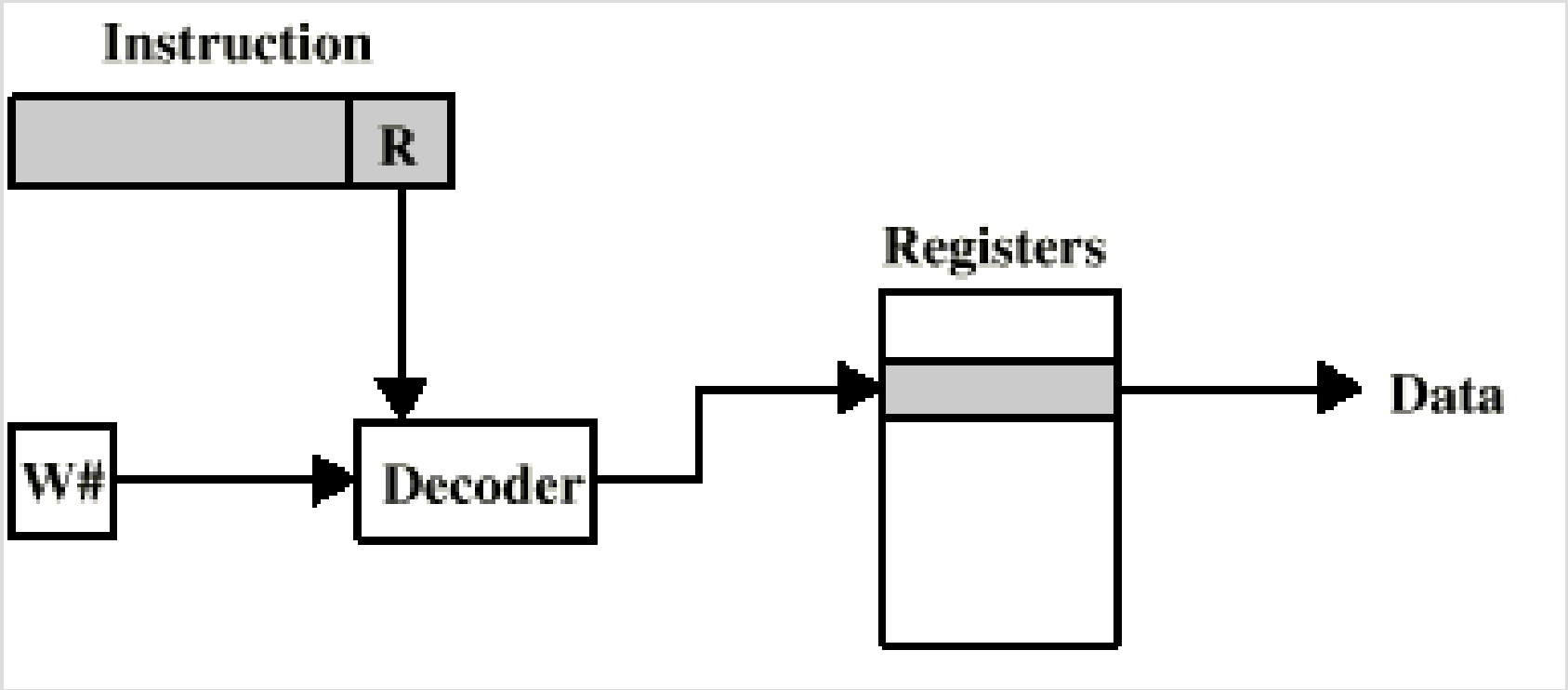
Variáveis Globais

- Alocadas pelo compilador em memória
  - Ineficiente para variáveis frequentemente acessadas
- Tem um conjunto de registradores para variáveis globais

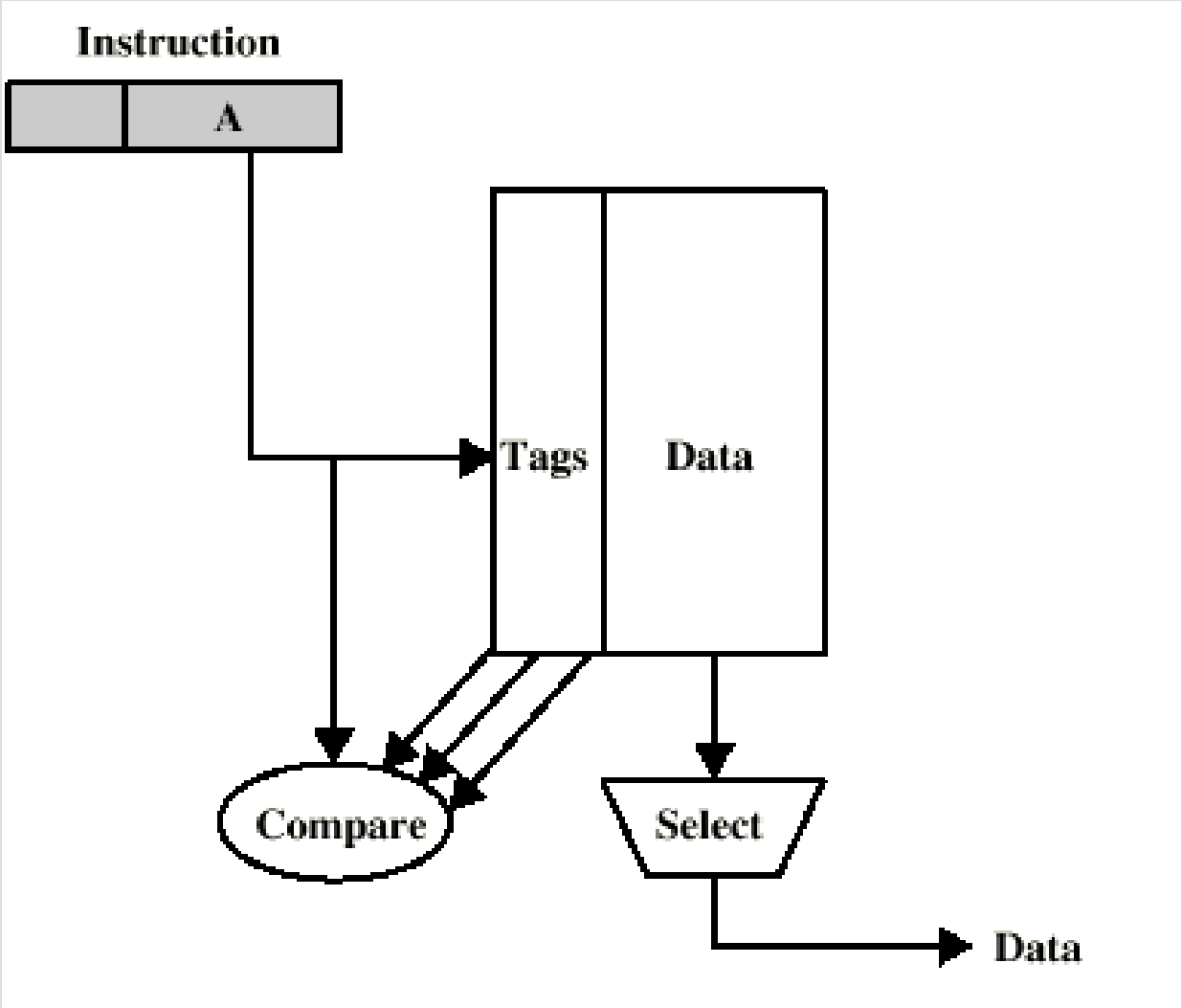
Registradores versus Cache

Grande banco de registradores	**Cache**
Escalares todos locais recentemente usados	Escalares locais
Variáveis Individuais	Blocos de memória
Variáveis globais atribuídas por compilador recentemente usadas	Variáveis globais
Save/restore baseado na profundidade de aninhamento de procedimentos	Save/restore baseado em subst. de cache
Endereçamento de registrador	Endereçamento de memória

Referenciando um escalar - Janela



Referenciando um Escalar - Cache



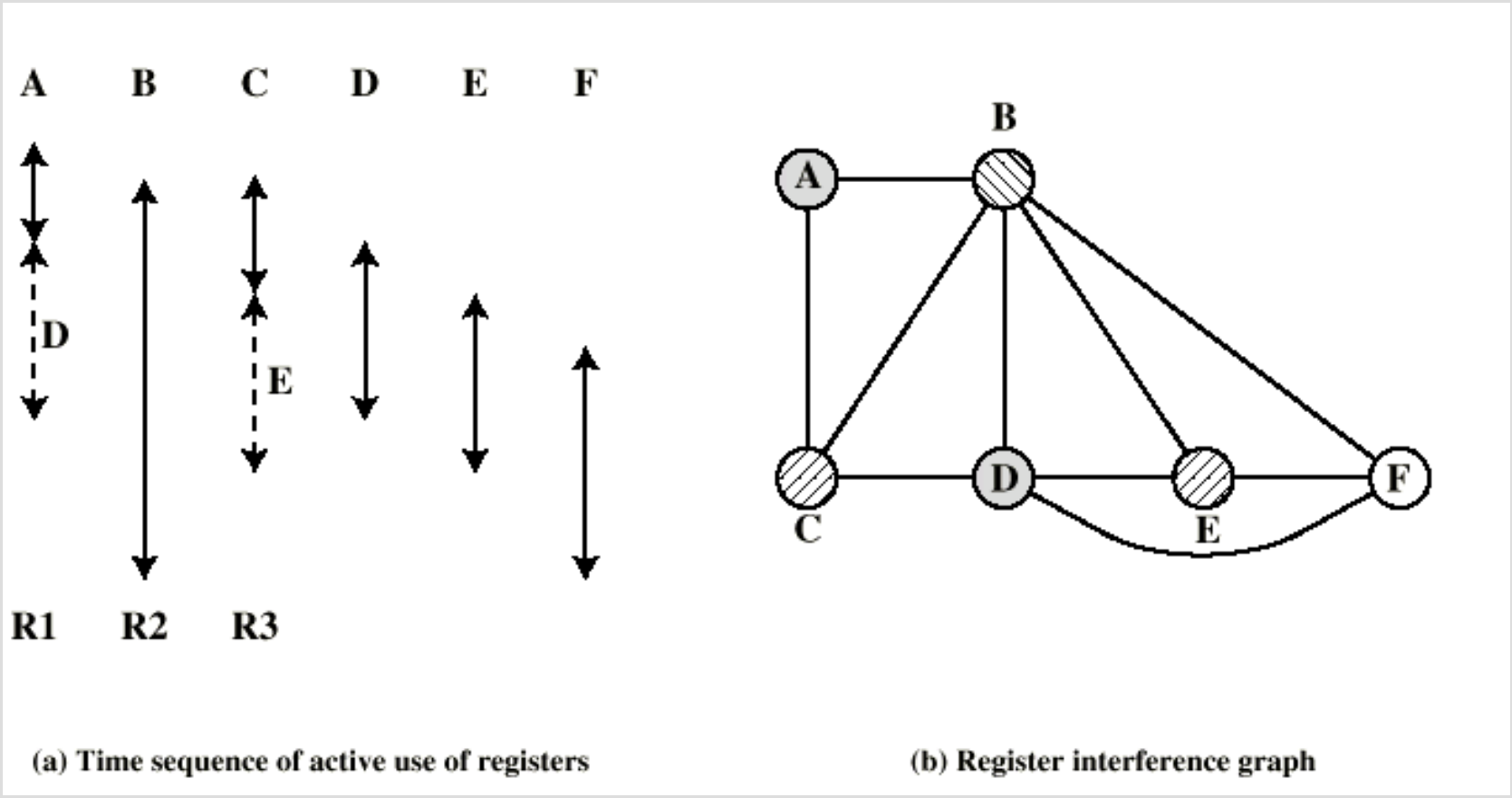
Otimização de Registrador Baseada em Compilador

- Assume pequeno número de registradores (16-32)
- Tarefa de otimização é do compilador
- Programas HLL não tem referências explícitas para os registradores
- Atribui registrador simbólico ou virtual para cada variável candidata
- Mapeia (ilimitados) registradores simbólicos para registradores reais
- Registradores simbólicos que não se sobrepoem podem compartilhar registradores reais
- Se todos os registradores reais acabarem algumas variáveis usam memória

Coloração de Grafos

- Dado um grafo com nós e arestas
- Atribuir uma cor para cada nó
- Nós adjacentes tem cores diferentes
- Use o menor número de cores
- Nós são registradores simbólicos
- Dois registradores que estão ativos no mesmo fragmento de programa são ligados por uma aresta
- Tente colorir o grafo com *n* cores, onde *n* é o número de registradores reais
- Nós que não podem ser coloridos são colocados na memória

Abordagem de Coloração de Grafos



(a) Time sequence of active use of registers

(b) Register interference graph

Por que CISC (1)?

- Simplificação do compilador?
  - Disputado…
  - Instruções de máquina complexas são difíceis de aproveitar
  - Otimização mais difícil
- Programas menores?
  - Programa toma menos memória mas…
  - Memória agora está mais barata
  - Pode não ocupar menos bits, apenas parece mais curta na forma simbólica
    - Mais instruções requerem op-codes maiores
    - Referência ao registrador requer menos bits

Características RISC

- Uma instrução por ciclo
- Operações de registrador para registrador
- Poucos e simples modos de endereçamento
- Poucos e simples formatos de instrução
- Projeto em Hardware (nenhum microcódigo)
- Formato de instrução fixa
- Mais esforço/tempo do compilador

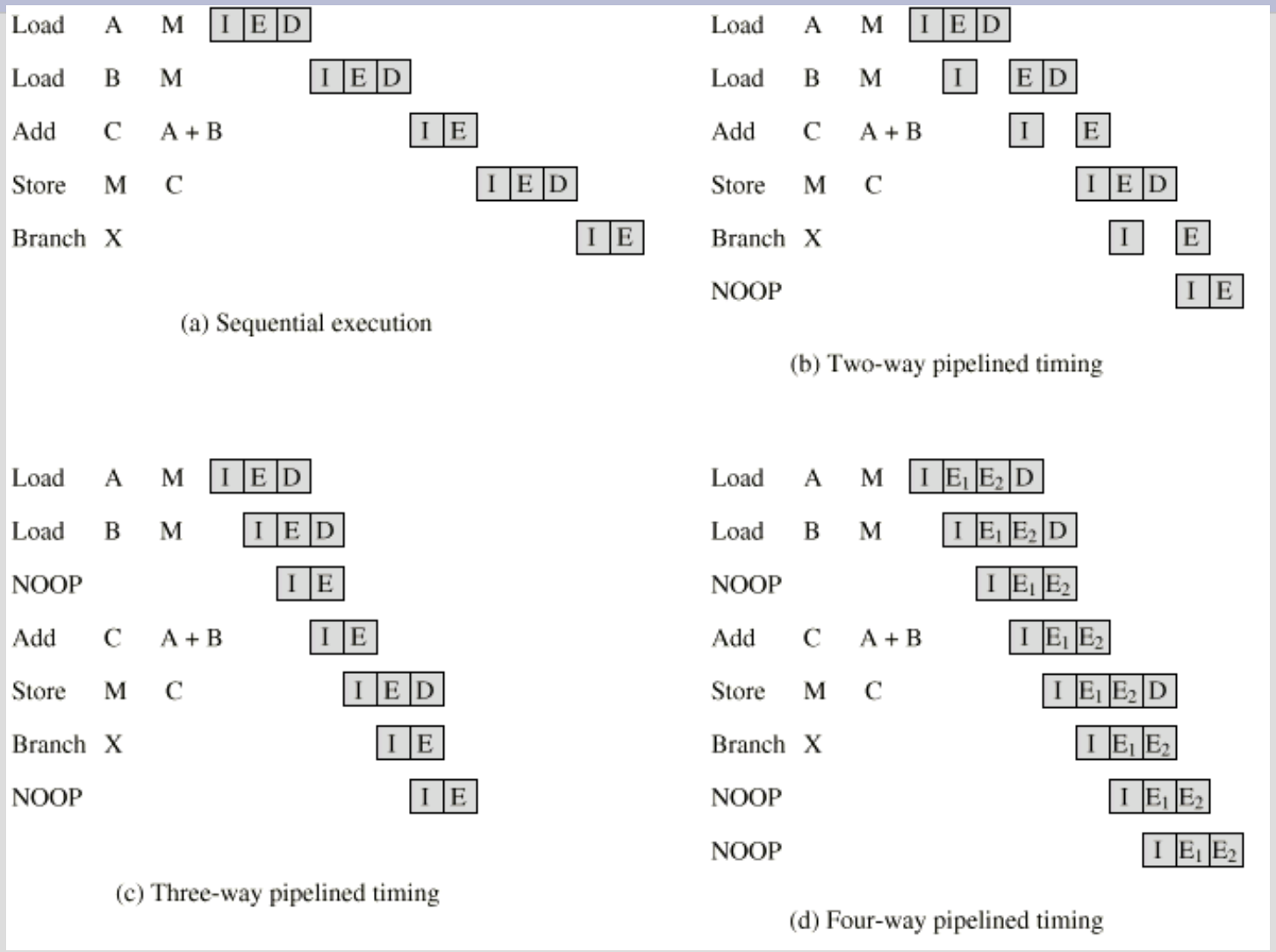
RISC v CISC

- Divisão não muito clara
- Muitos projetos emprestam elementos de ambas filosofias
- ex. PowerPC e Pentium II

RISC Pipelining

- Maioria das instruções são de registrador para registrador
- Duas fases de execução
  - I: Instruction fetch (busca)
  - E: Execute (execução)
    - op. na ALU com entrada e saída em registrador
- Para load e store
  - I: Instruction fetch (busca)
  - E: Execute (execução)
    - Calcula endereço de memória
  - D: Memória
    - Operação registrador para memória ou memória para registrador

Efeitos do Pipelining



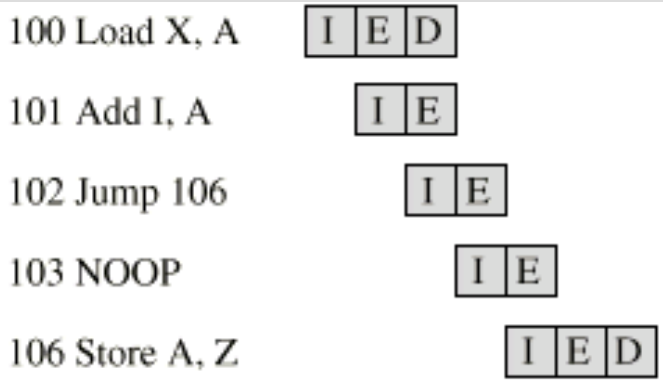
Otimização do Pipelining

- Desvio atrasado
  - Não tem efeito até depois da execução da instrução seguinte
  - Esta próxima instrução é o período (janela) de retardo

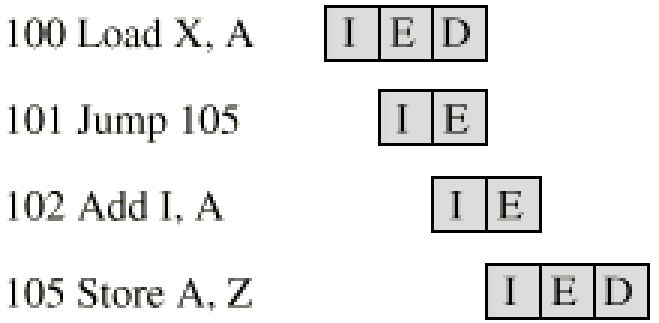
Desvio Normal e Atrasado

Endereço	Normal	Atrasado	Otimizado
100	LOAD X,A	LOAD X,A	LOAD X,A
101	ADD 1,A	ADD 1,A	JUMP 105
102	JUMP 105	JUMP 105	ADD 1,A
103	ADD A,B	NOOP	ADD A,B
104	SUB C,B	ADD A,B	SUB C,B
105	STORE A,Z	SUB C,B	STORE A,Z
106		STORE A,Z	

Uso de Atraso de Desvio



(a) Inserted NOOP



(b) Reversed instructions

Controvérsia

- Quantitativa
  - compara tamanhos de programa e tempos de execução
- Qualitativa
  - examina questões de suporte à linguagem de alto nível e o uso de VLSI
- Problemas
  - RISC e CISC não são diretamente comparáveis
  - Dificuldade em separar efeitos de hardware dos efeitos de compilação
  - Maioria das comparações são feitas em máquinas de “brinquedo” ao invés de máquinas de produção
  - A maioria dos dispositivos comerciais são mistos