Uma Infra-estrutura Modular e Extensível para Manipulação de Binários MIPS

Ricardo Nabinger Sanchez
rnsanchez@wait4.org

César Augusto Missio Marcon
Orientador

Universidade do Vale do Rio dos Sinos
Centro de Ciências Exatas e Tecnológicas

29 de junho de 2007

Roteiro

Compiladores Atuais Devem...

- Aproveitar recursos disponíves em *hardware* específico
- Aplicar diversas técnicas de otimização
- Evitar que elas interfiram negativamente entre si

O Problema

- Desenvolver novas técnicas de otimização
- Refinar as técnicas existentes
- Identificar a interação entre as técnicas
- Evitar que elas interfiram negativamente entre si

Diversos Trabalhos Publicados

Desenvolvimento de técnicas de otimização Tjiang, S., Hennessy, J. *Sharlit—A Tool for Building Optimizers.* ACM SIGPLAN'92.

Interação entre técnicas de otimização Pan, Z., Eigenmann, R. *Fast and Effective Orchestration of Compiler Optimizations for Automatic Performance Tuning.* IEEE CGO'06.

Objetivo Geral

Infra-estrutura extensível capaz de aplicar técnicas de reengenharia de código diretamente sobre binários executáveis.

Objetivos Específicos

1. Infra-estrutura modular e extensível
2. Protótipo em *software* para a arquitetura-alvo ELF/MIPS
3. Módulo de manipulação para validação das técnicas adotadas

Por que alterar diretamente um binário executável?

- Independência da linguagem de alto nível usada
- Quantidade de dados significativamente menor
- Reengenharia de binários:
  - Re-otimização de binários
  - Tradução binária

Por que MIPS?

- Arquitetura RISC[1]
- Semelhanças com outras plataformas RISC
- Diversas ferramentas disponíveis
- Possibilidade de avaliação experimental com *hardware* de prototipação (FPGA[2])

[1] *Reduced Instruction Set Computer*
[2] *Field-programmable Gate Array*

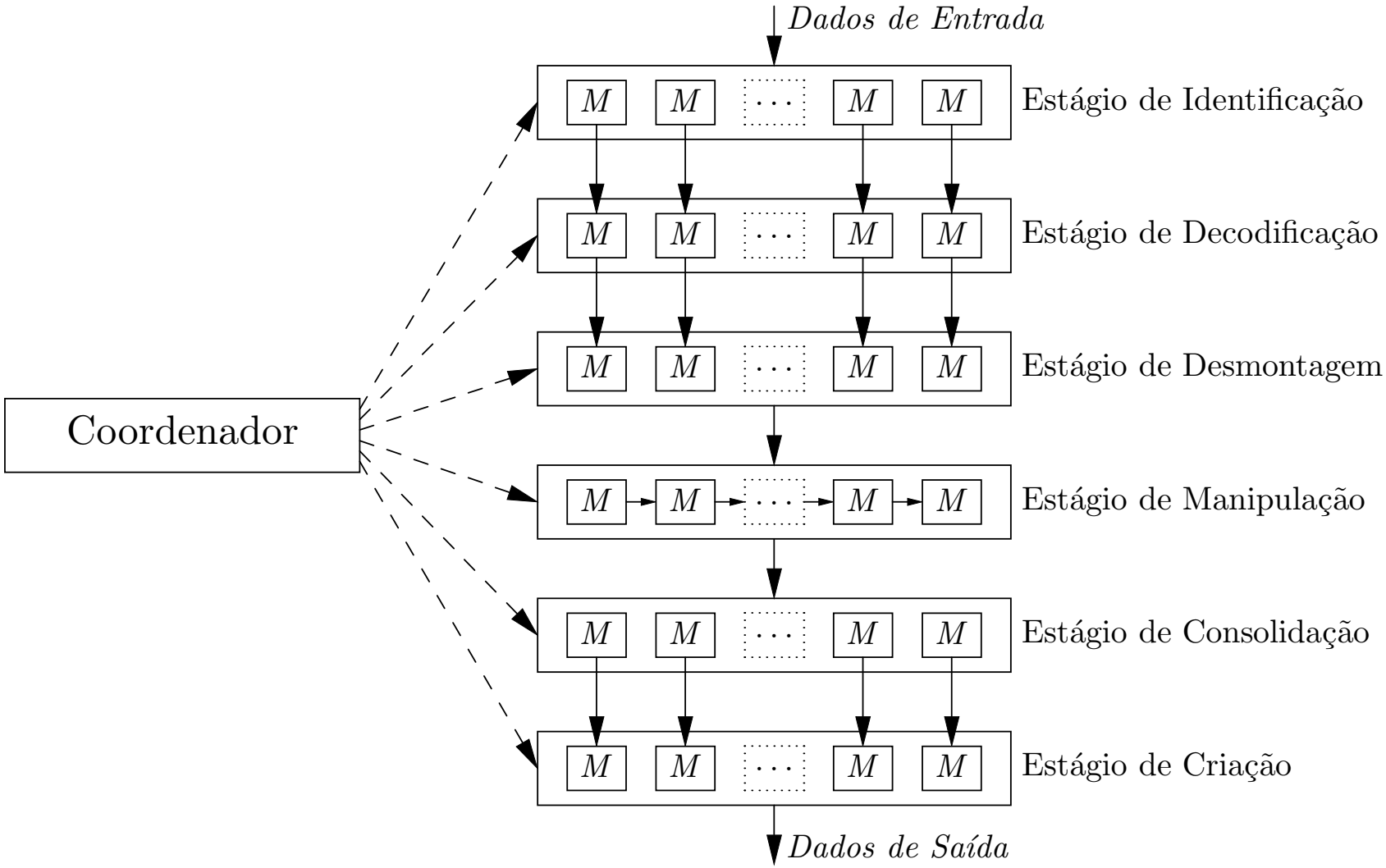
Infra-estrutura Proposta

- Composta por n estágios
  - Isolam detalhes dos dados de entrada, saída, arquitetura-hospedeira e arquitetura-alvo
- Cada estágio é composto por m_{n_i} módulos
  - Implementam funcionalidades específicas

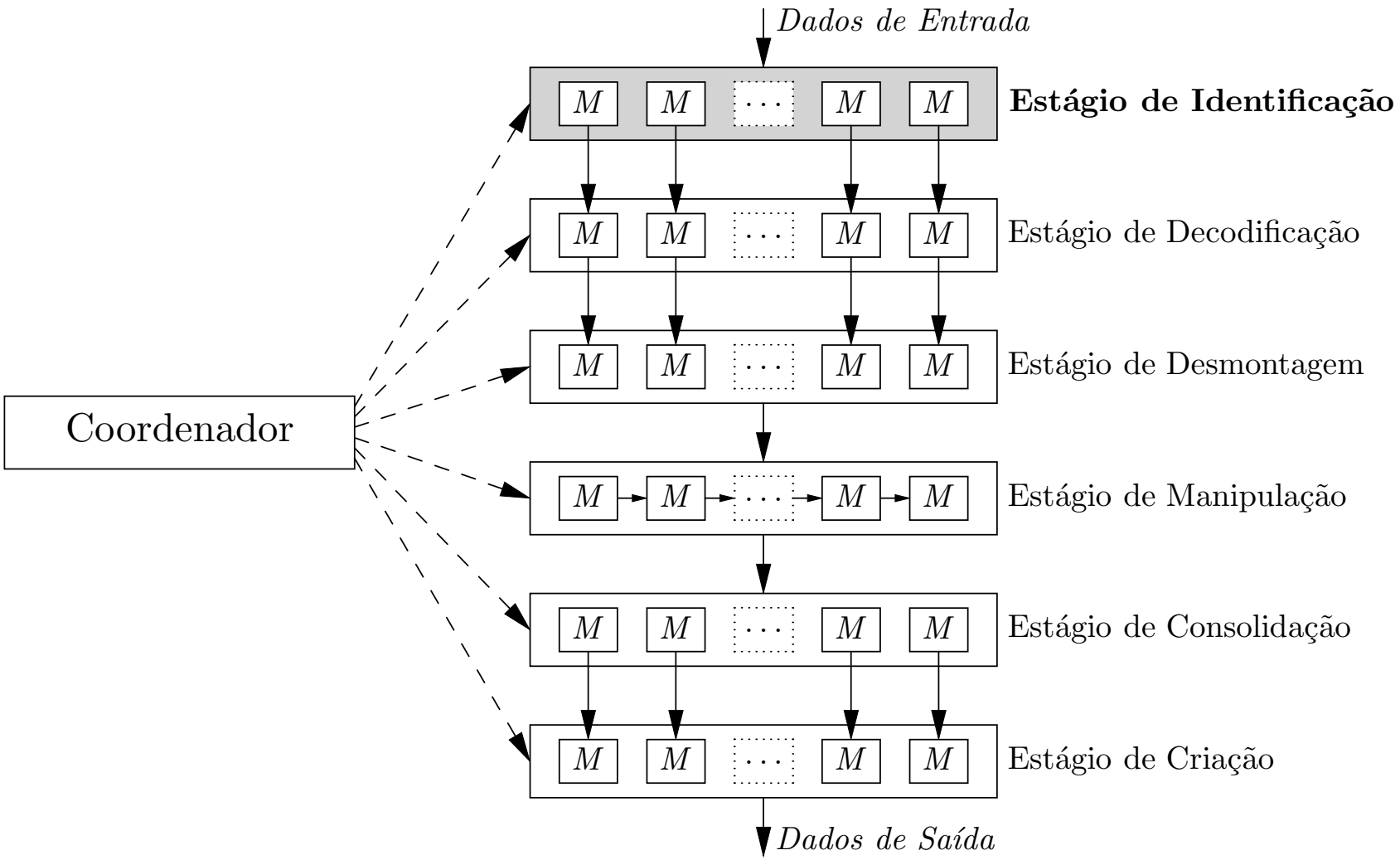
Funcionamento Básico

- Um coordenador determina o fluxo de trabalho
- Os dados de saída de um estágio servem de entrada para o próximo estágio
- A saída do último estágio é o resultado final

Visão Geral


Dados de Entrada
Coordenador
M
Estágio de Identificação
Estágio de Decodificação
Estágio de Desmontagem
Estágio de Manipulação
Estágio de Consolidação
Estágio de Criação
Dados de Saída

Estágio de Identificação



Estágio de Identificação

- Módulos são aplicados seqüencialmente
- Algum deles reconhece o formato dos dados de entrada
- Este determina os módulos subseqüentes:
  - Decodificação
  - Desmontagem

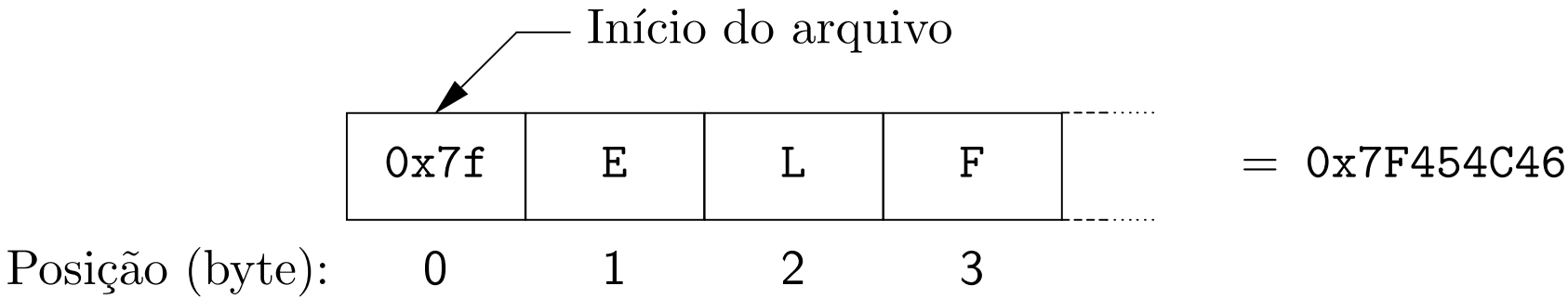
Principais Técnicas de Identificação

Estágio de Identificação

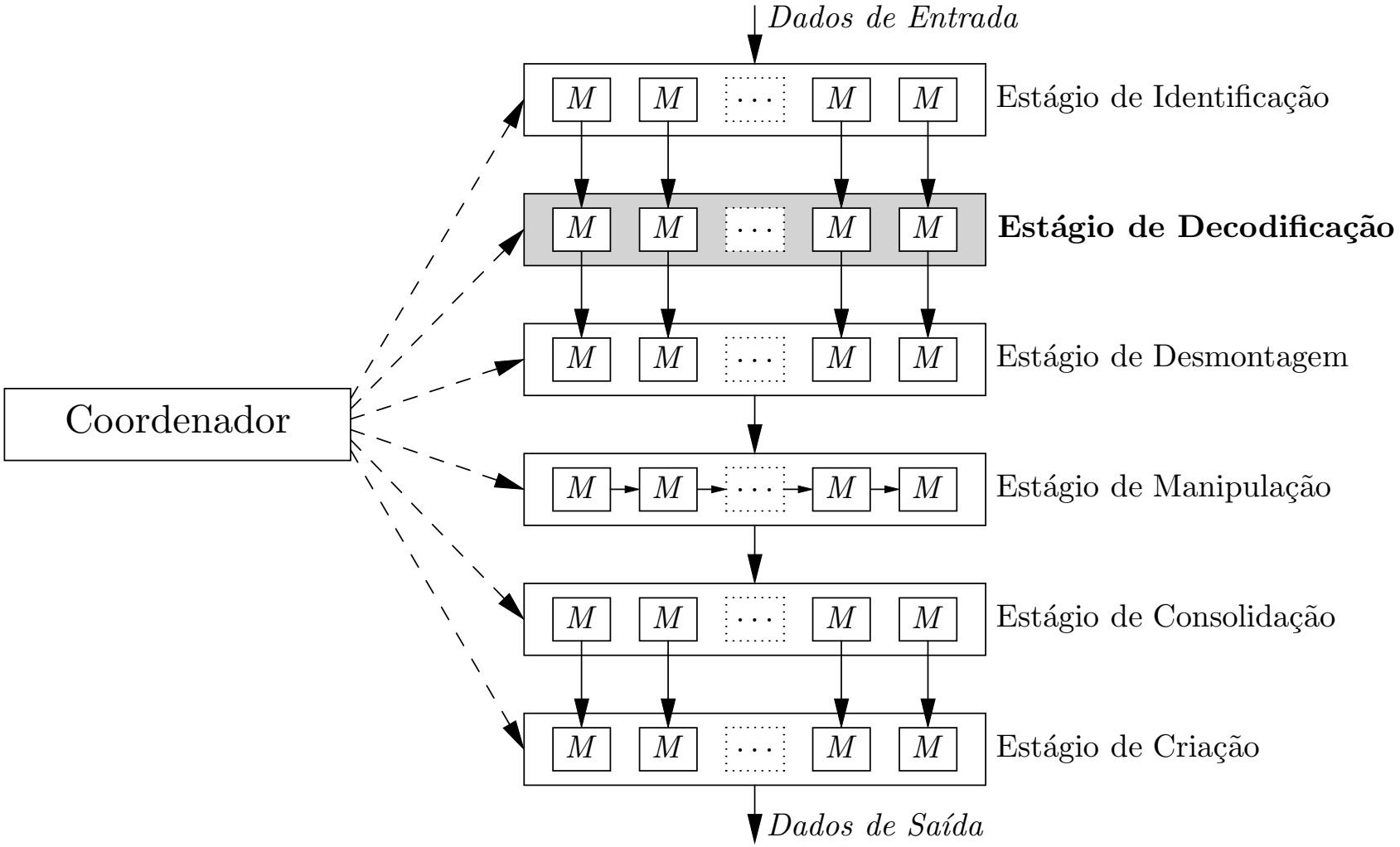
- Nome do arquivo: geralmente a extensão do arquivo
- Assinatura: número mágico definido pelo autor
- Inferência: análise (de parte) do conteúdo do arquivo
- Meta-dados: algum elemento externo indica o formato
- Mista: combina duas ou mais técnicas

Exemplo de Identificação por Assinatura

Estágio de Identificação



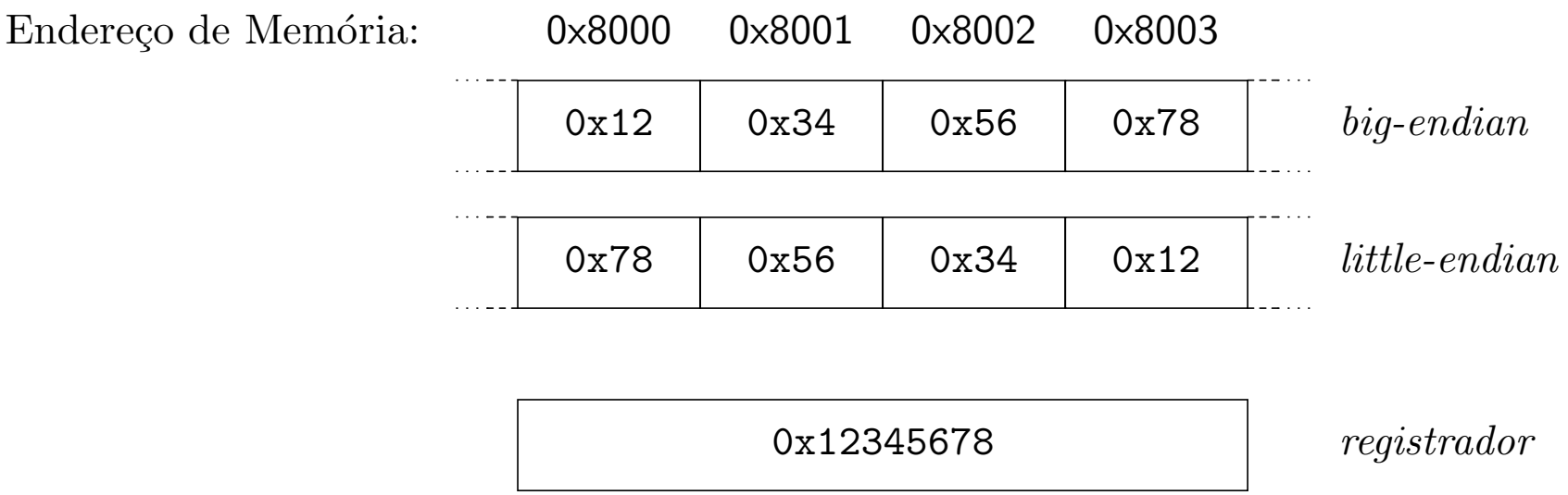
Estágio de Decodificação



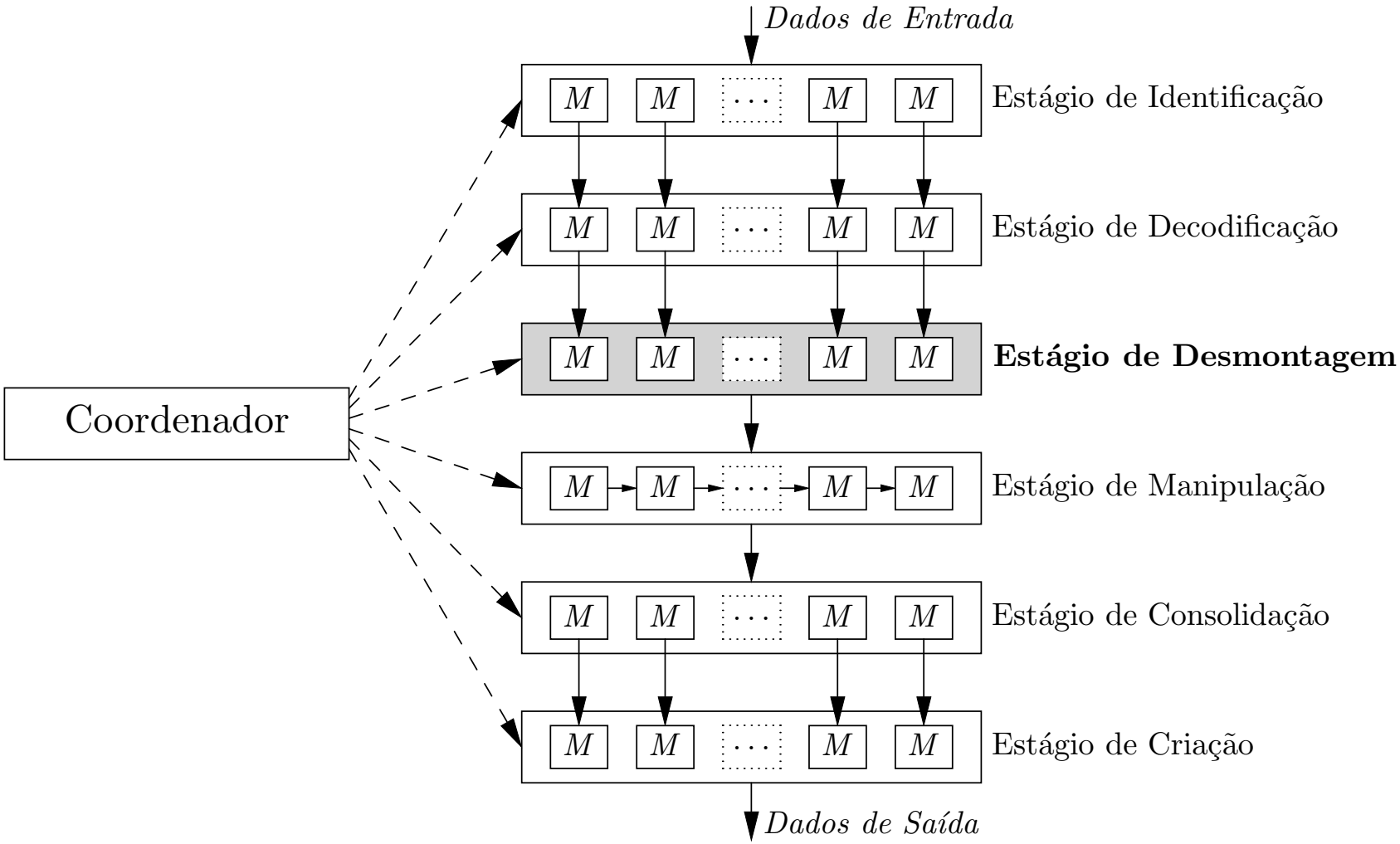
Estágio de Decodificação

- Abstrai detalhes do arquivo-objeto do restante da infra-estrutura
- Extrai segmentos do arquivo-objeto que contenham instruções
- Popula uma representação virtual de memória
- Lida com algumas questões da arquitetura (ordem de bytes)

Ordem de Bytes
Estágio de Decodificação



Estágio de Desmontagem



Estágio de Desmontagem

- Opera sobre uma representação virtual de memória
- Converte instruções de máquina para uma representação manipulável
- Computa o CFG[3] do código durante a desmontagem
- Lida diretamente com características da arquitetura

[3] *Control Flow Graph*

Representação Intermediária
Estágio de Desmontagem

- Modelada como um CFG
- Nodos do CFG representam blocos básicos
- Cada bloco básico contém uma lista de instruções
- Vértices do CFG representam desvios
- Desvios podem ser tomados ou não

Abordagens Dinâmicas de Desmontagem

Estágio de Desmontagem

- Execução assistida
- Acesso a todas as informações disponíveis
- Cobertura de desmontagem depende da entrada fornecida ao programa
- Tempo de desmontagem depende da execução do programa

Abordagens Estáticas de Desmontagem

Estágio de Desmontagem

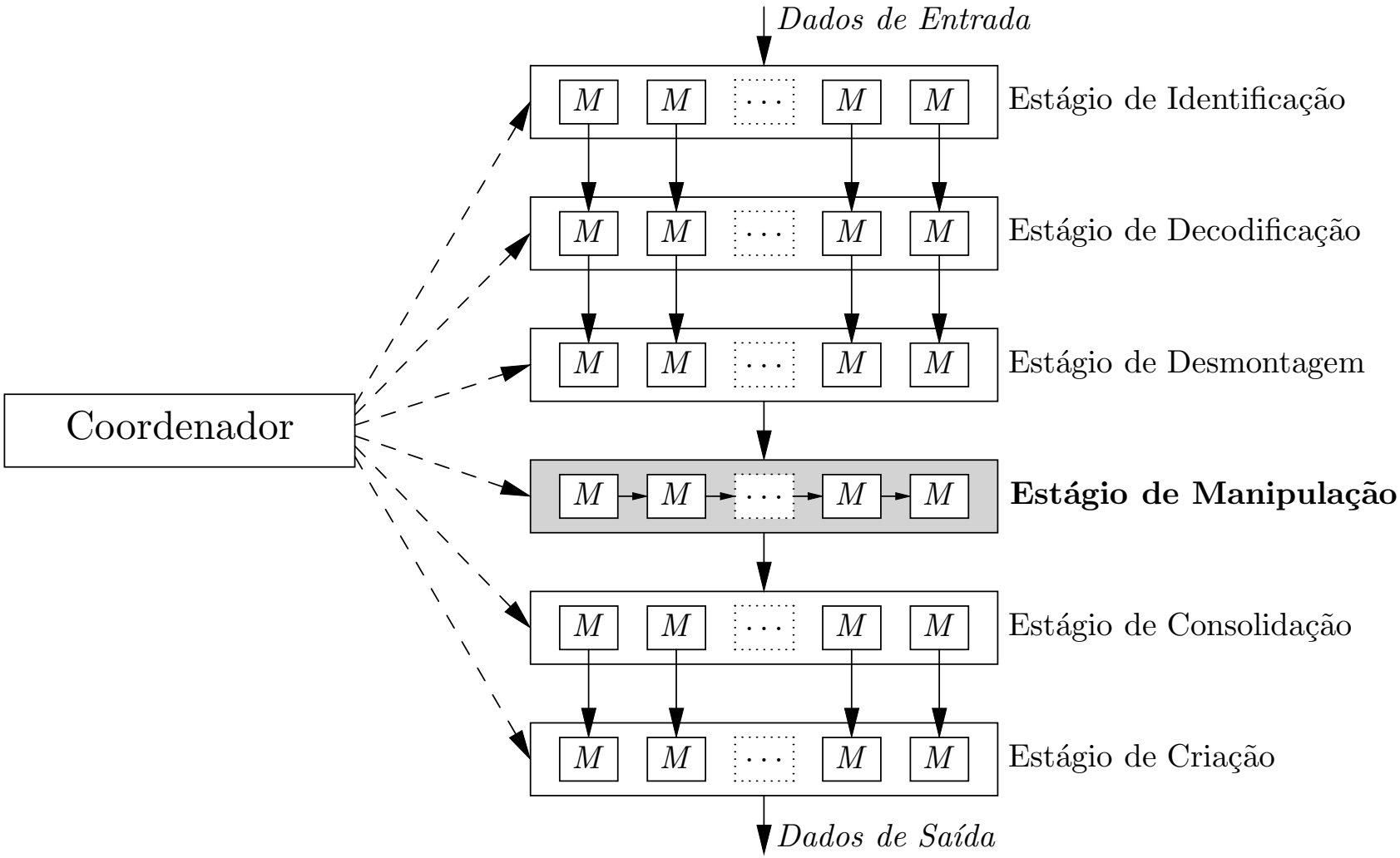
- Sem contexto de execução
- Acesso às informações disponíveis estaticamente
- Pode empregar análise ou execução simbólica
- Cobertura e tempo de desmontagem dependem da técnica adotada

Caminhamento Recursivo (*Recursive Traversal*)

Estágio de Desmontagem

- Desmonta de acordo com o fluxo do código
- Análise parcial das instruções, aumentando a complexidade
- Capaz de evitar dados presentes no fluxo de instruções
- Cobertura de desmontagem depende da abordagem (dinâmica ou estática)
- Comum o aparecimento de sombras

Estágio de Manipulação

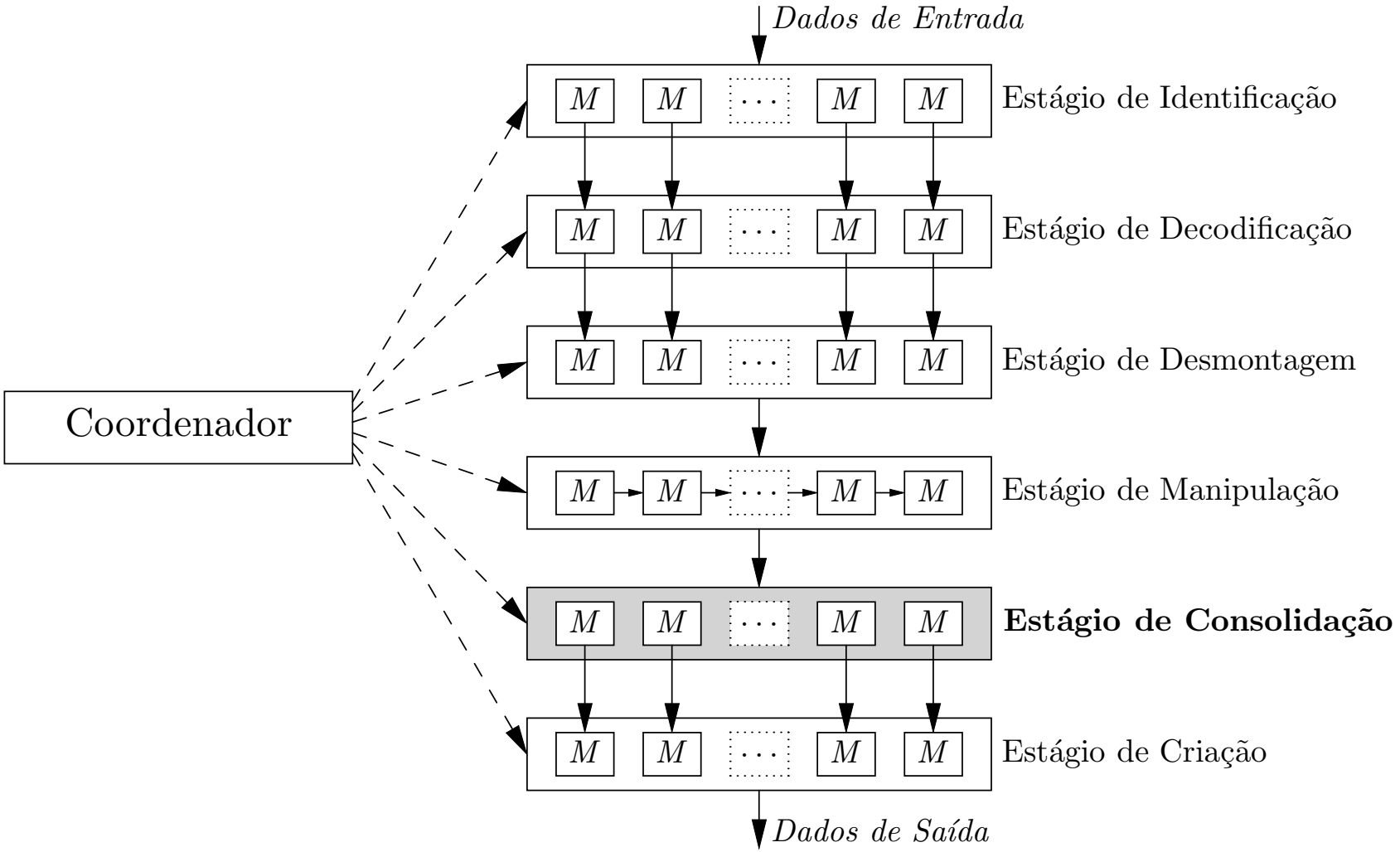


Estágio de Manipulação

- Cada módulo implementa uma técnica de reengenharia específica
- Aplicados seqüencialmente
- Atuam diretamente no CFG
- Módulos devem zelar pela consistência dos dados:
  - CFG
  - Representação virtual de memória
- Se necessário, DFGs[4] dos blocos básicos relevantes são computados

[4] *Data Flow Graph*

Estágio de Consolidação

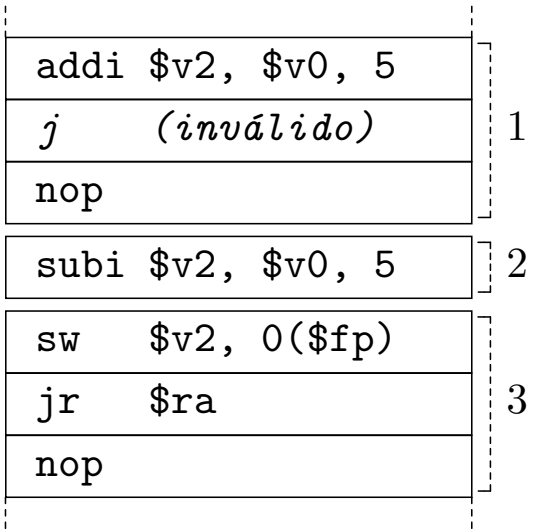


Estágio de Consolidação

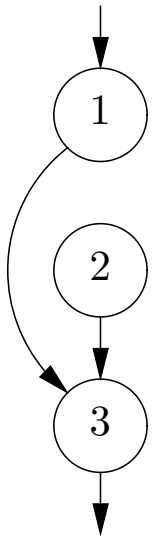
- Linearização do código (possivelmente modificado)
- Ajuste de endereços das instruções de desvio
  - Endereços são corrigidos com a ajuda do CFG
  - Lida com detalhes de arquitetura
- Segmentação da representação virtual de memória

Ajuste de Endereços
Estágio de Consolidação

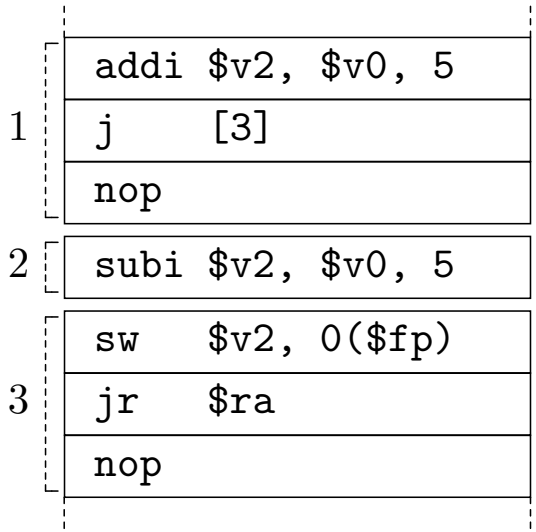
Fluxo de Instruções



CFG

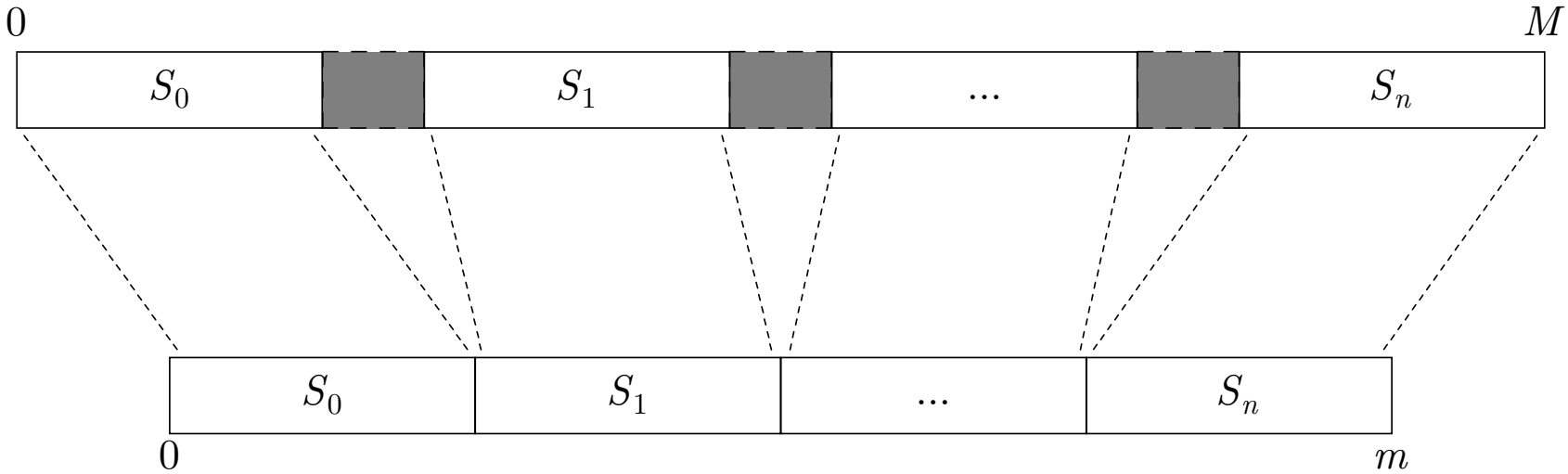


Fluxo de Instruções Corrigido

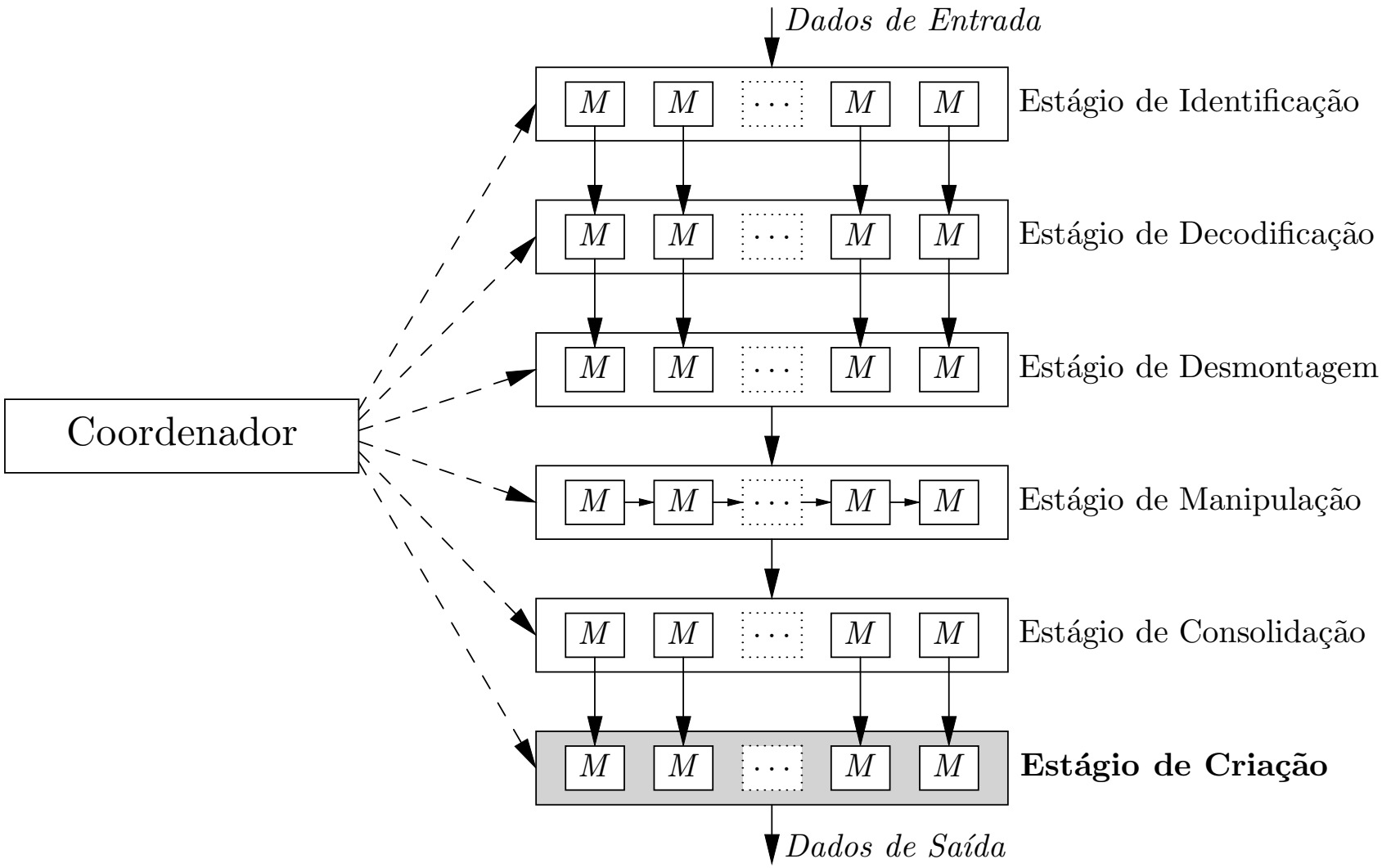


Processo de Segmentação

Estágio de Consolidação



Estágio de Criação



Estágio de Criação

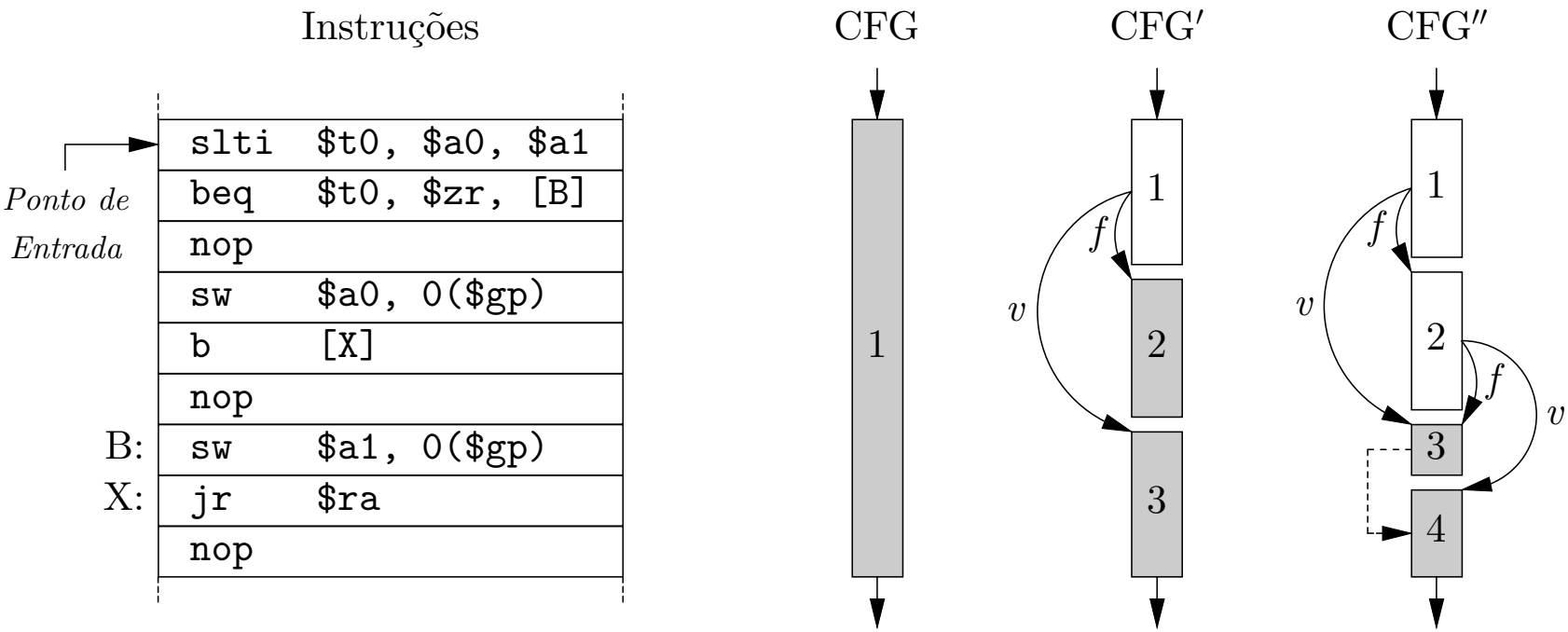
- Recebe todas as informações disponíveis
- Gera um novo arquivo-objeto
- Formato de saída não é necessariamente igual ao de entrada

Técnica de Desmontagem Proposta

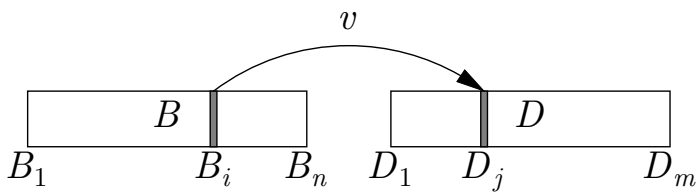
Desmontagem Recursiva Estendida:

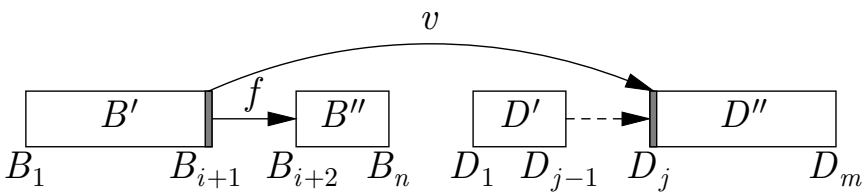
- Semelhante à técnica recursiva tradicional
- Assume que todo o código é um grande bloco básico
- Divide os blocos básicos ao longo da desmontagem
- Preserva código não referenciado estaticamente

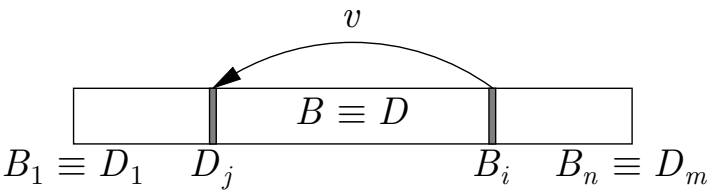
Funcionamento da Técnica

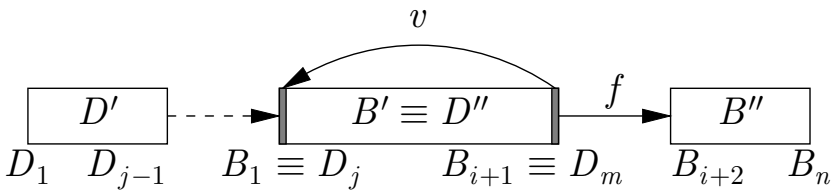


Divisão de Blocos Básicos









ReMIPS

- Ferramenta implementada para validação das propostas
- Contempla a abordagem organizada em estágios e modular
- Arquitetura-alvo ELF/MIPS
- Desenvolvida na linguagem C, em ambiente UNIX
- Licença GPL[5]

[5] GNU *General Public License*

Módulo de Manipulação

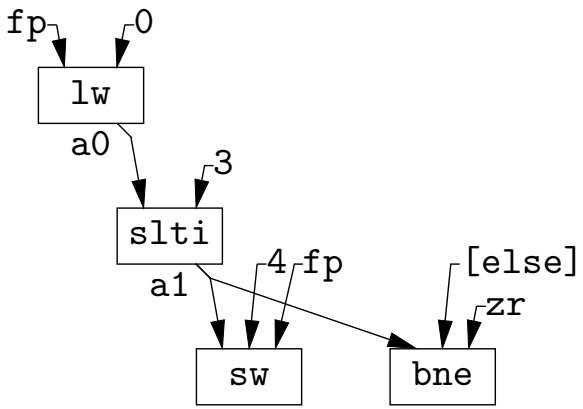
- Implementa técnica de otimização *delayed branch optimization*
- Move computação útil para o *delay slot*
- *Delay slot* é a instrução após alguma de desvio
- Aplicável apenas aos blocos básicos contendo instrução de desvio

Funcionamento da Técnica de Otimização

Bloco Básico

```
lw    $a0, 0($fp)
slti  $a1, $a0, 3
sw    $a1, 4($fp)
bne   $a1, $zr, [else]
nop
```

Fluxo de Dados



Bloco Básico Modificado

```
lw    $a0, 0($fp)
slti  $a1, $a0, 3
bne   $a1, $zr, [else]
sw    $a1, 4($fp)
```

Cenários Avaliados

A. *Bubble-sort* sobre um vetor de tamanho arbitrário
B. *Checksum* para cabeçalhos de pacotes IPv4
C. `switch` (*expr*) com quantidade arbitrária de alternativas
D. `if...then...else`, com uma expressão simples
E. Um programa que não realiza nenhuma operação

Resultados

- Nem todas oportunidades podem ser aproveitadas
- Maior aproveitamento requer realocação de registradores
- Ainda assim, a técnica pôde ser aplicada:
  - 13% menos instruções nos cenários A, B, e C
  - 24% menos instruções nos cenários D e E

Conclusões

- É viável manipular diretamente binários executáveis
- Abordagens estáticas podem investir mais tempo nas modificações
- Garantir a consistência após modificações é o grande desafio
- Muitas possibilidades de aplicação
- Área de reengenharia de código tem atraído pesquisadores e também a indústria

Infra-estrutura
Trabalhos Futuros

- Melhor separação entre os estágios de consolidação e criação
- Extinção do vértice de concatenação
- Representação intermediária genérica

ReMIPS
Trabalhos Futuros

- Módulos adicionais:
  - Formatos de entrada e saída de dados
  - Técnicas de desmontagem
  - Técnicas de manipulação
- Preservar consistência nos casos em que a desmontagem não é completa
- Cenários mais complexos de avaliação
- Porte para C++
- Licença estilo BSD

Publicação em Avaliação

Sanchez, R. N., Marcon, C. A. M. *Uma Infra-estrutura Modular e Extensível para a Manipulação de Binários MIPS*[6].
19th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2007): VIII Workshop em Sistemas Computacionais de Alto Desempenho (WSCAD 2007).

[6] Submetida em 2007-06-25

Fim da Apresentação

Obrigado!
Perguntas?

"If you think it's simple, then you have misunderstood the problem."

— Bjarne Stroustrup