Republica Federativa do Brasil
Ministerio do Desenvolvimento, Industria e do Comercio Exterior
Instituto Nacional da Propriedade Industrial

(21) **PI 9902725-9 A**

(22) Data de Depósito 13/07/1999
(43) Data de Publicação **06/03/2001**
(RPI 1574)

(51) Int. Cl[7].:
G06F 9/40

(54) Título **PROCESSO DE FORMAÇÃO, MEMORIZAÇÃO E REUSO, EM TEMPO DE EXECUÇÃO, DE SEQUÊNCIAS DE INSTRUÇÕES DINÂMICAS EM COMPUTADORES**

(71) Depositante(s) COPPE/UFRJ- Coordenação dos Programas de Pos Graduação de Engenharia da Universidade Federal do Rio de Janeiro (BR/RJ)

(72) Inventor(es) Felipe Maia Galvão França

(74) Procurador Joubert Gonçalves de Castro & Zuldech Assessoria Empresarial Ltda

(57) Resumo Patente de Invenção "PROCESSO DE FORMAÇÃO, MEMORIZAÇÃO E REUSO, EM TEMPO DE EXECUÇÃO, DE SEQUÊNCIAS DE INSTRUÇÕES DINÂMICAS EM COMPUTADORES" Trata-se de um processo para a formação, memorização e reuso em tempo de execução, de seqüências de instruções dinâmicas com o objetivo de reduzir o numero de instruções dinâmicas executadas por um programa, de forma a eliminar a necessidade de reexecução de instruções redundantes (instruções que reutilizam mais de uma vez os mesmos valores para seus operandos de entrada) inclusas na sequência quando esta apresentar como argumentos de entrada valores ja observados anteriormente e registrados em tabelas de memorização Este processo não necessita efetuar invalidações em entradas das tabelas de memorização ou possui seu uso limitado a cadeias de instruções dependentes O processo reduz o numero de instruções dinâmicas que devam ser executadas pelo programa alvo, reduzindo desta forma o tempo total de execução do dado programa e mantendo-se a correção dos resultados A invenção refere-se também a um dispositivo eletrônico para a realização do processo

PC	N_PC	R_Reg	S1_Reg	S2_Reg	R_Val	S1_Val	S2_Val	I

Relatório Descritivo da Patente de Invenção "PROCESSO DE FORMAÇÃO, MEMORIZAÇÃO E REUSO, EM TEMPO DE EXECUÇÃO, DE SEQUÊNCIAS DE INSTRUÇÕES DINÂMICAS EM COMPUTADORES"

CAMPO TÉCNICO

A presente invenção refere-se a um processo de definição de um sistema computacional digital capaz de formar (selecionar), memorizar e reusar, em tempo de execução (dinamicamente), seqüências de instruções dinâmicas, eliminando assim a necessidade de se reexecutar tais sequências quando as mesmas apresentarem entradas já observadas anteriormente, e consequentemente as mesmas saídas; ressalta-se que: instruções dinâmicas são caracterizadas por serem instâncias decorrentes de instruções estáticas em tempo de execução; seqüência de instruções dinâmicas é definida como sendo uma sucessão de instruções dinâmicas, sucessão esta, ordenada pelo fluxo de execução; entrada de uma sequência de instruções dinâmicas é definida como sendo o conjunto dos valores atribuídos aos operandos fonte de cada instrução dinâmica pertencente à seqüência; e saída de uma seqüência de instruções dinâmicas é definida como sendo o conjunto dos valores atribuídos a cada operando de destino de cada instrução dinâmica pertencente à sequência. A invenção visa a concepção de um circuito eletrônico digital que realize o referido processo.

TÉCNICAS ANTERIORES

A redução do tempo de execução de programas usando memorização de resultados em computadores, é basicamente efetuado através da Memorização e consequente Reuso de Subrotinas, Funções e Instruções Dinâmicas, objetivando assim a redução do número total de instruções necessárias à execução de programas, mantendo-se porém a correção dos

resultados. Alguns procedimentos para Memorizar e Reusar Subrotinas, Funções e Instruções Dinâmicas já existem, entretanto não foi explorado nenhum processo para Formação, Memorização e Reuso de seqüências de instruções dinâmicas onde o objetivo é Formar (através de seleção), Memorizar e Reusar em tempo de execução, seqüências de instruções dinâmicas formadas a partir da reexecução de instruções dinâmicas com valores de entrada já observados.

Os procedimentos para Reuso de Subrotinas e Funções geralmente apresentam como obstáculo principal o processo de Seleção de tais Subrotinas e Funções. A Seleção de Subrotinas e Funções a serem memorizadas é efetuado de maneira estática, sendo feita manualmente observando-se a codificação e semântica do programa, ou com auxílio de estatísticas de execução, ou assistida pelo compilador, ou incrementalmente. A aplicação deste procedimento não é genérico e apresenta-se de forma muito particular e dependente da existência de tais elementos em um dado programa. Os procedimentos para Seleção, Memorização e o Reuso de instruções dinâmicas, aplica Memorização e Reuso a partir de esquemas particulares baseados em tabelas de Memorização com um determinado número de entradas, e que memoriza as instruções dinâmicas, onde no melhor caso em condições particulares para todos os esquemas, reusa-se no máximo quatro (4) instruções dinâmicas de uma só vez.

Nesta invenção apresentamos um processo simples e eficiente para Selecionar, Memorizar e Reusar, em tempo de execução, sequências de instruções dinâmicas de quaisquer tamanho (onde o tamanho de uma seqüência de instruções dinâmicas é referência à quantidade de instruções dinâmicas inclusas na seqüência), e em comparação aos trabalhos anteriores, obtém-se um maior percentual de reuso a partir da aplicação do processo.

Os Processos existentes para Formação, Memorização e Reuso aplicados a programas são:

1. Software Function Memoization - Onde em um dado programa, as Subrotinas ou Funções candidatas a serem reusadas são previamente selecionadas de forma estática, e as funções de Memorização e Reuso são efetuadas por trechos de código inseridos no programa alvo.

2. Hardware Function Memoization - Onde em um dado programa, as Subrotinas ou Funções candidatas a serem reusadas são previamente selecionadas de forma estática, e as funções de Memorização e Reuso são efetuadas por hardware ativado por instruções inseridas no código do programa alvo.

3. Hardware Instruction Memoization - Onde em programas em geral: a Seleção, Memorização e o Reuso, são efetuados sobre instruções dinâmicas e através de hardware em tempo de execução. Os esquemas existentes para efetuar o reuso de instruções diferem entre si basicamente: pelas informações armazenadas nas tabelas de Memorização, pelo método de acesso às tabelas, e pelas invalidações efetuadas.

3.1. O esquema Sv reusa instruções que apresentam valores para os operandos (registradores) de entrada, já anteriormente observados ;

3.2. O esquema Sn reusa instruções baseado nos nomes dos operandos (registradores), isto é, uma instrução reusará um valor instanciado a um operando, se o operando que o armazena não for alterado;

3.3. O esquema Sn+d reusa instruções como Sn, porém manipula também cadeias de instruções dependentes, mais especificamente, cadeias de instruções que possuem entre si dependências verdadeiras de dados;

3.4. O esquema Memoing aplica reuso de forma restrita e exclusiva à instruções com grandes latências de execução (multiplicação, divisão, etc...), sendo válido apenas para instruções nas quais o custo de reuso é menor que o custo de execução (considerando que o custo é medido em tempo).

4. Hardware Basic Block Reuse - Onde em programas em geral: a Seleção, Memorização e o Reuso, são efetuados sobre blocos básicos, i.e., sequências de instruções delimitadas por um único ponto de entrada e um único ponto de saída. Os blocos básicos são selecionados parcialmente e previamente marcados estaticamente pelo compilador e/ou dinamicamente por hardware. Durante a execução de um programa, os blocos básicos serão memorizados com seus respectivos escopos de entrada e de saída, incluindo instruções de escrita/leitura em memória que deverão ser executadas para validar o reuso. Este método restringe a memorização a trechos de código com fluxo de execução bem definido.

Todos os esquemas apresentados possuem utilização restrita à instruções isoladas ou à blocos básicos.

DESCRIÇÃO DA INVENÇÃO

O processo objeto desta invenção permite o emprego de hardware, tipicamente um circuito eletrônico digital, em tempo de execução para a Formação, Memorização e o Reuso de sequências de instruções dinâmicas. Neste processo caracteriza-se uma sequência de instruções dinâmicas como sendo constituída tipicamente (seqüências contendo apenas uma (1) instrução podem também ser tratadas pelo processo) por duas (2) instruções. O objetivo do processo é reusar diretamente os resultados provenientes da reexecução de instruções dinâmicas que apresentem entradas já memorizadas anteriormente, ao invés de se reexecutar todas as

instruções da sequência, deste modo reduz-se o número total de instruções executadas de um dado programa, obtendo-se desta forma uma redução do tempo total de execução.

Assim, é objetivo da presente invenção prover um processo para Formação, Memorização e Reuso de seqüências de instruções dinâmicas, composto basicamente por quatro (4) etapas distintas:

(i) SELEÇÃO DO CONJUNTO DE INSTRUÇÕES PARA A APLICAÇÃO DO PROCESSO

A aplicação do processo será restrito a um subconjunto do conjunto de instruções do computador alvo, podendo ser ampliado ou diminuído, dependendo de suas características arquiteturais. As instruções básicas manipuladas pelo processo são classificadas em grupos de acordo com sua funcionalidade, estes grupos são definidos como grupo de instruções aritméticas, grupo de instruções lógicas, grupo de instruções de controle de fluxo de execução, e grupo de instruções para manipulação de memória local e principal. As sequências de instruções dinâmicas serão compostas por instruções pertinentes a estes grupos e a união de tais grupos será denominada como sendo o Conjunto de Seleção.

(ii) DEFINIÇÃO E REPRESENTAÇÃO DAS TABELAS DE MEMORIZAÇÃO

As tabelas de memorização serão definidas e representadas de acordo com as características arquiteturais do computador alvo, esta definição fará considerações ao modelo de uma arquitetura load/store (onde os operandos das instruções estarão sempre em registradores internos ao computador alvo), entretanto poderá ser aplicada a outros modelos de arquitetura. Seja Memo_table_I uma tabela que armazenará instruções dinâmicas já executadas, onde cada instrução armazenada estará instanciada com os seus

respectivos valores para os operandos de entrada e resultados de saída decorrente de sua execução.

Seja Memo_table_T uma tabela que armazenará informações sobre sequências de instruções dinâmicas onde todas as instruções da sequência já foram executadas e estão ou estiveram armazenadas em Memo_table_I, objetivando assim a aplicação de Reuso.

Seja Buffer_T uma tabela que armazenará temporariamente elementos de Memo_table_I consecutivamente em suas entradas, i.e. armazenará a sequência de instruções dinâmicas em formação.

Descrição dos campos de uma entrada de Memo_table_I da Figura (1):

PC - Endereço em memória da instrução.

N_PC - Endereço em memória da próxima instrução a ser executada.

R_Reg - Registrador que armazena o resultado da operação.

S1_Reg - Registrador que armazena o primeiro operando da instrução.

S2_Reg - Registrador que armazena o segundo operando da instrução.

R_Val - Valor do resultado produzido pela execução da instrução.

S1_Val - Valor do primeiro operando da instrução.

S2_Val - Valor do segundo operando da instrução.

I - Se 0 - indica que a instrução possui um operando em registrador e o segundo operando é imediato.

1 - indica que a instrução possui os dois operandos em registradores.

Descrição dos campos de uma entrada de Memo_table_T da Figura (2):

PC_F - Endereço em memória da primeira instrução da sequência.

NPC_L - Endereço em memória da próxima instrução a ser executada após a última instrução da seqüência.

IS_Tags - Registradores usados pelo escopo de entrada.

OS_Tags - Registradores usados pelo escopo de saída.

IS_Val - Valores dos registradores indicados em IS_Tags.

OS_Val - Valores dos registradores indicados em OS_Tag.

(iii) PROCEDIMENTOS PARA INDEXAÇÃO E ATUALIZAÇÃO DAS TABELAS DE MEMORIZAÇÃO

As tabelas de memorização serão indexadas por chaves de acordo com os campos da referida tabela. Uma entrada em Memo_table_I será indexada pela associação dos campos PC + S1_Val + S2_Val. Uma corrente instrução dinâmica instanciada para execução será encontrada em Memo_Table_I, se o seu endereço em memória (PC) e os valores associados a seus operandos de entrada, coincidirem respectivamente com os campos PC, S1_Val e S2_Val de Memo_table_I. Uma entrada em Memo_table_T será indexada pela associação dos campos PC_F + IS_Tags + IS_Val. Uma corrente instrução dinâmica instanciada para execução será encontrada em Memo_Table_T, se o seu endereço de memória (PC) combinar com o campo PC_F de alguma entrada em Memo_table_T e nesta entrada os valores dos registradores do arquivo de registradores do computador alvo indicados pelo campo IS_Tags (da mesma entrada selecionada por PC) coincidirem com os campos IS_Val (da mesma entrada) que correspondem ao escopo de entrada da sequência armazenada. O Buffer_T possuirá apenas a função de armazenamento temporário da sequência identificada, este armazenamento temporário possuirá como objetivo, extrair da sequência completa, somente informações relevantes ao Reuso da mesma e armazenamento destas informações em Memo_table_T. As descrição das operações de atualização das Tabelas (Memo_table_I e Memo_table_T) será baseada em uma política de FILA (outras

políticas de atualização podem ser utilizadas), i.e., havendo entradas livres nas tabelas, estas poderão acomodar novas instruções dinâmicas ou informações sobre as sequências de instruções dinâmicas a serem inseridas, caso contrário a entrada mais antiga (ou seja, a que foi inserida a mais tempo) será expurgada para ceder espaço à nova inserção requisitada, sendo a última inserção considerada a mais recente.

(iv) ALGORITMO DE FORMAÇÃO, MEMORIZAÇÃO E REUSO DE SEQUÊNCIAS DE INSTRUÇÕES DINÂMICAS

O algoritmo para Formação, Memorização e Reuso de sequências de instruções dinâmicas baseia-se em dois (2) procedimentos de busca efetuados sobre as tabelas Memo_table_I e Memo_table_T. A realização destas buscas podem ocorrer concorrentemente entre si e ainda, assincronamente e/ou subdivididas em vários estágios de um pipeline paralelo aos estágios da arquitetura do computador alvo.

Sejam as Tabelas Memo_table_I, Memo_table_T, e Buffer_T. Para cada instrução a ser executada faça:

Busca em Memo_table_I

(1) Verificar se a instrução corrente pertence ao conjunto de Seleção;

(2) Se (1) for falso então qualquer seqüência em formação é finalizada em Buffer_T, e as informações extraídas sobre a sequência armazenada em Buffer_T, serão armazenadas em Memo_table_T, e Buffer_T será liberado para o armazenamento de uma nova sequência; fim do algoritmo senão continue;

(3) Verificar se a instrução corrente instanciada com seus operandos está armazenada em Memo_Table_I;

(4) Se (3) for falso então incluir a instrução corrente instanciada em Memo_table_I e qualquer seqüência em formação é finalizada em Buffer_T e as informações

extraídas sobre a sequência armazenada em Buffer_T, serão armazenadas em Memo_table_T, e Buffer_T será liberado para o armazenamento de uma nova sequência; fim do algoritmo senão a instrução em questão é redundante, e irá iniciar ou ser acrescentada à formação da seqüência em curso, sendo incluída em Buffer_T.

Busca em Memo_Table_T.

(1) Verificar se a instrução corrente pertence ao conjunto de Seleção;

(2) Se (1) for falso então não efetuar busca em Memo_Table_T ; fim do algoritmo senão continue;

(3) Verificar se o PC (contador de programa) da instrução corrente, ocorre no campo PC_F de alguma entrada de Memo_table_T;

(4) Se (3) for verdadeira então selecionar a primeira entrada na qual o escopo de entrada indicado por IS_Tags e IS_Val corresponda ao estado do arquivo de registradores da máquina;

(5) Se (4) for verdadeira (i.e.,existe pelo menos uma entrada válida) então considerando a entrada encontrada, carregar o PC do computador alvo com o valor do campo NPC_L e atualizar o arquivo de registradores do computador alvo com o escopo de saída dado pelos campos OS_Tags e OS_Val.

EXEMPLO

Para a Formação das sequências de instruções dinâmicas a serem memorizadas, será considerado o trecho de código apresentado na Figura 3, onde para cada instrução instanciada (com valores atribuídos a seus operandos) pela execução deste trecho de código, estas serão armazenadas em entradas na Memo_table_I (se a referida instrução pertencer ao Conjunto de Seleção) como mostra a Figura 4. Uma instrução será considerada redundante se em uma reexecução desta instrução, esta apresentar uma instância já

armazenada em Memo_table_I, neste caso a instrução é candidata a iniciar ou complementar a formação de uma sequência de instruções dinâmicas, sendo esta instrução armazenada sequencialmente em Buffer_T (que armazena uma sequência de instruções redundantes). A Figura 5 apresenta uma sequência de instruções redundantes, seqüência esta, armazenada em Buffer_T e delimitada pelo intervalo de endereços [104,116]. A Figura 6 apresenta uma nova entrada incluída em Memo_table_T a partir da seqüência armazenada em Buffer_T (Figura 5), esta nova entrada irá armazenar informações sobre a sequência formada, este procedimento corresponde à fase de memorização da sequência de instruções dinâmicas. O Reuso da sequência do exemplo, ocorrerá quando houver uma reexecução da instrução 104 e nesta reexecução os valores dos operandos do escopo de entrada da sequência coincidirem com os valores dos respectivos operandos no arquivo de registradores do computador alvo, ou seja, se existir uma entrada em Memo_table_T com o valor 104 no campo PC_F e o arquivo de registradores do computador alvo possuir em seus registradores identificados pelo campo IS_Tags da mesma entrada, os mesmos valores armazenados no campo IS_Val também da mesma entrada Figura 7. Para este caso, considerando a entrada de Memo_table_T já identificada, a ação de reuso será procedida pela escrita do campo NPC_L no PC do computador alvo, alterando assim o fluxo de execução, e rescrevendo os registradores do arquivo de registradores do computador alvo indicados pelo campo OS_Tags com os respectivos valores armazenados no campo OS_Val, restaurando assim o contexto da máquina para a execução da instrução subsequente à seqüência reusada. A partir do exposto as instruções de uma sequência redundante deixarão

de ser executadas, pois foram reusadas todas de uma só vez, procedendo-se com o apresentado na Figura 8.

VANTAGENS DO PROCESSO REIVINDICADO

O processo de Formação, Memorização e Reuso de sequências de instruções dinâmicas aqui descrito é eficiente na exploração do comportamento repetitivo de trechos de programas que são frequentemente reexecutados a partir das mesmas entradas, sob os seguintes aspectos:

(i) Não é necessário identificar, formar ou selecionar estaticamente trechos com propensão à repetição.

(ii) Reusar uma seqüência de instruções dinâmicas equivale a reusar todas as instruções desta seqüência de uma única vez, de modo que o custo de se executar todas as instruções da seqüência é reduzido ao custo de execução de uma única instrução da sequência, para os casos em que o custo de reuso de uma (1) instrução é igual ao custo de execução da mesma instrução (custo medido em tempo).

(iii) É dispensável a instanciação de todas as instruções de uma sequência de instruções dinâmicas que serão executadas, pois o reuso de sequências é baseado em escopos de entrada, e estes determinam o fluxo e os resultados de todas as instruções a serem executadas, não é necessária a execução que quaisquer instrução pertinete a seqüência para validar o reuso da mesma.

(iv) O reuso de sequências de instruções dinâmicas não é restrito a quantidade de acessos que possam ser feitos simultaneamente às instruções dinâmicas isoladamente em uma tabela de memorização.

(v) O processo apresentado reduz a carga de hardware e gargalos de execução para efetuar: (a) Renomeação de Registradores, desde que as instruções de uma sequência reusada que não foram reexecutadas, não

necessitaram renomear seus registradores de destino aliviando assim a unidade de renomeação; (b) Predição de Desvios, desde que os desvios inclusos em tais sequências reusadas serão antecipadamente resolvidos reduzindo assim a latência de resolução de desvios; (c) Demanda por Unidades Funcionais, desde que evitando-se a reexecução de instruções de uma sequência redundante evita-se a alocação de Unidades Funcionais para as instruções da sequência; (d) Resolução de Dependências Verdadeiras de Dados, desde que instruções dependentes que estejam inclusas em uma dada sequência redundante serão executadas de uma só vez, ou seja, no mesmo ciclo .

(VI) Outros tipos de instruções podem ser incluídas no processo, de modo que ele permite aperfeiçoamento incremental. Implementação adaptável a qualquer computador alvo, desde que 99,6% das seqüências de instruções dinâmicas capturadas e reusadas pelo processo (medidas em testes) apresentaram para o tamanho dos escopos de entrada e saída, individualmente, um valor menor ou igual à quatro (4) registradores.

PI9902725

REIVINDICAÇÕES

1. Processo para a Formação, Memorização e Reuso, em tempo de execução, de sequências de instruções dinâmicas, caracterizado por reusar, em tempo de execução, sequências de instruções dinâmicas contendo, não exclusivamente, uma ou mais instruções do tipo controle de fluxo de execução, i.e., instruções que têm como ação característica principal modificar o registrador contador de programas do computador alvo.

2. Processo para a Formação, Memorização e Reuso, em tempo de execução, de sequências de instruções dinâmicas, caracterizado por selecionar um subconjunto reduzido, quando comparado à outras metodologias correlatas, de tipos de instruções candidatas ao reuso pelo computador alvo que exclui os tipos de (i) instruções de acesso à memória principal; (ii) instruções envolvendo operandos representados em ponto-flutuante; e (iii) instruções privilegiadas (tipicamente de suporte ao sistema operacional); podendo no entanto esta seleção ser ampliada ou diminuída, dependendo das características arquiteturais do computador alvo, sem comprometer o funcionamento correto ,neste caso não necessariamente com a mesma eficiência, do referido processo de formação, memorização e reuso, em tempo de execução, de sequências de instruções dinâmicas compostas por instruções pertinentes aos tipos restantes, i.e., tipicamente tipos de (iv) instruções aritméticas, (v) instruções lógicas e deslocamentos posicionais, e (vi) instruções de controle de fluxo de execução, chamadas e retorno à subrotinas.

3. Processo para a Formação, Memorização e Reuso, em tempo de execução, de sequências de instruções dinâmicas, de acordo com a reivindicação 1, caracterizado por definir o uso concorrente de três tipos de tabelas de

memorização diferentes, sejam: (i) Memo_table_I um tipo de tabela que armazenará instruções dinâmicas já executadas, onde cada instrução armazenada estará instanciada com os seus respectivos valores para os operandos de entrada e resultados de saída decorrente de sua execução; (ii) Memo_table_T um tipo de tabela que armazenará informações sobre sequências de instruções dinâmicas, onde todas as instruções da tabela já foram executadas e estão ou estiveram armazenadas em Memo_table_I objetivando assim a aplicação de Reuso, e; (iii) Buffer_T um tipo de tabela que armazenará temporariamente elementos de Memo_table_I consecutivamente em suas entradas, i.e., armazenará a sequência de instruções dinâmicas em formação.

4. Processo para a Formação, Memorização e Reuso, em tempo de execução, de sequências de instruções dinâmicas, de acordo com a reivindicação 3, caracterizado por proceder na indexação e atualização das tabelas de memorização, de modo que as tabelas de memorização serão indexadas por chaves de acordo com os campos da referida tabela; uma entrada em Memo_table_I será indexada pela associação dos campos PC + S1_Val + S2_Val e uma corrente instrução dinâmica para execução será encontrada em Memo_Table_I se o seu endereço em memória (PC) e os valores associados aos seus operandos de entrada coincidirem respectivamente com os campos PC, S1_Val e S2_Val de Memo_table_I; uma entrada em Memo_table_T será indexada pela associação dos campos PC_F + IS_Tags + IS_Val e uma corrente instrução dinâmica para execução será encontrada em Memo_Table_T se o seu endereço em memória (PC) combinar com o campo PC_F de alguma entrada em Memo_table_T e nesta entrada os valores dos registradores do arquivo de registradores do computador alvo indicados pelo campo IS_Tags (da mesma entrada selecionada por PC) coincidirem

com os campos IS_Val (da mesma entrada) que correspondem ao escopo de entrada da seqüência armazenada; o Buffer_T possuirá apenas a função de armazenamento temporário da sequência identificada e este armazenamento temporário possuirá como objetivo extrair da seqüência de instruções dinâmicas somente informações relevantes ao reuso da mesma e armazenamento destas informações em Memo_table_T, visto as considerações anteriores.

5. Processo para a Formação, Memorização e Reuso, em tempo de execução, de seqüências de instruções dinâmicas, de acordo com a reivindicação 4, caracterizado por dispensar o uso de qualquer mecanismo de invalidação sobre entradas nos três tipos de tabelas utilizados.

6. Processo para a Formação, Memorização e Reuso, em tempo de execução, de sequências de instruções dinâmicas, de acordo com a reivindicação 4, caracterizado por executar uma seqüência algorítmica de formação, memorização e reuso de sequências de instruções dinâmicas baseada em dois (2) procedimentos de busca efetuados sobre as tabelas Memo_table_I e Memo_table_T, onde a realização destas buscas podem ocorrer concorrentemente entre si e, ainda, assincronamente e/ou subdivididas em vários estágios de um pipeline paralelo aos estágios da arquitetura do computador alvo, sendo as tabelas Memo_table_I, Memo_table_T, e Buffer_T, onde para cada instrução a ser executada será feita uma Busca em Memo_table_I do seguinte modo:(1) Verificar se a instrução corrente pertence ao Conjunto de Seleção;(2) se (1) for falso então qualquer sequência em formação é finalizada em Buffer_T, as informações extraídas sobre a seqüência armazenada em Buffer_T serão armazenadas em Memo_table_T, Buffer_T será liberado para o armazenamento de uma nova seqüência e fim do algoritmo; senão continue;(3) Verificar se a instrução

corrente instanciada com seus operandos está armazenada em Memo_Table_I;(4) se (3) for falso então incluir a instrução corrente instanciada em Memo_table_I sendo que qualquer seqüência em formação é finalizada em Buffer_T, as informações extraídas sobre a seqüência armazenada em Buffer_T serão armazenadas em Memo_table_T, Buffer_T será liberado para o armazenamento de uma nova seqüência e fim do algoritmo; senão a instrução em questão é redundante e irá iniciar ou ser acrescentada à formação da seqüência em curso, sendo incluída em Buffer_T e uma Busca em Memo_Table_T será feita do seguinte modo: (1) Verificar se a instrução corrente pertence ao conjunto de Seleção;(2) se (1) for falso então não efetuar busca em Memo_Table_T e fim do algoritmo; senão continue;(3) Verificar se o PC (contador de programa) da instrução corrente ocorre no campo PC_F de alguma entrada de Memo_table_T;(4) se (3) for verdadeira então selecionar a primeira entrada na qual o escopo de entrada indicado por IS_Tags e IS_Val corresponda ao estado do arquivo de registradores do computador alvo;(5) se (4) for verdadeira (i.e., existe pelo menos uma entrada válida) então, considerando a entrada encontrada, carregar o PC do computador alvo com o valor do campo NPC_L e atualizar o arquivo de registradores do computador alvo com o escopo de saída dado pelos campos OS_Tags e OS_Val.

7. Processo para a Formação, Memorização e Reuso, em tempo de execução, de seqüências de instruções dinâmicas, de acordo com a reivindicação 6, caracterizado por formar seqüências de instruções dinâmicas a partir da n-ésima repetição, $n \in N^*$, onde N^* representa o conjunto dos números naturais excluindo-se o zero, de qualquer instrução pertencente ao Conjunto de Seleção.

FIGURAS

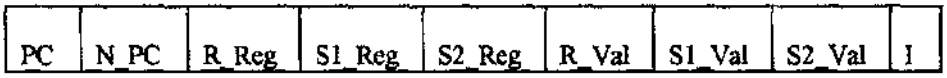

Figura 1

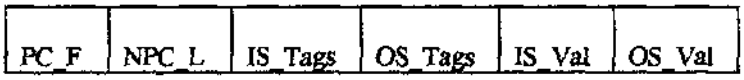

Figura 2

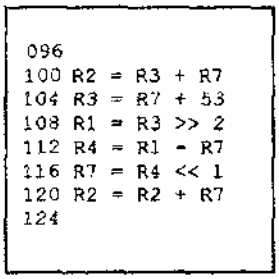

Figura 3

PC	*N_PC*	*R_Reg*	*S1_Reg*	*S2_Reg*	*R_Val*	*S1_Val*	*S2_Val*	*I*
.								
100	104	R2	R3	R7	20	9	11	1
104	108	R3	R7	53	64	11	53	0
108	112	R1	R3	2	16	64	2	0
112	116	R4	R1	R7	5	16	11	1
116	120	R7	R4	1	10	5	1	0
120	124	R2	R2	R7	30	20	10	1
.								

Figura 4

PC	*N_PC*	*R_Reg*	*S1_Reg*	*S2_Reg*	*R_Val*	*S1_Val*	*S2_Val*	*I*
104	108	R3	R7	53	64	11	53	0
108	112	R1	R3	2	16	64	2	0
112	116	R4	R1	R7	5	16	11	1
116	120	R7	R4	1	10	5	1	0

Figura 5

PC_F	*NPC_L*	*IS_Tags*	*OS_Tags*	*IS_Val*	*OS_Val*
100	124	R7	R1,R3,R4,R7	11	16,64,5,10

Figura 6

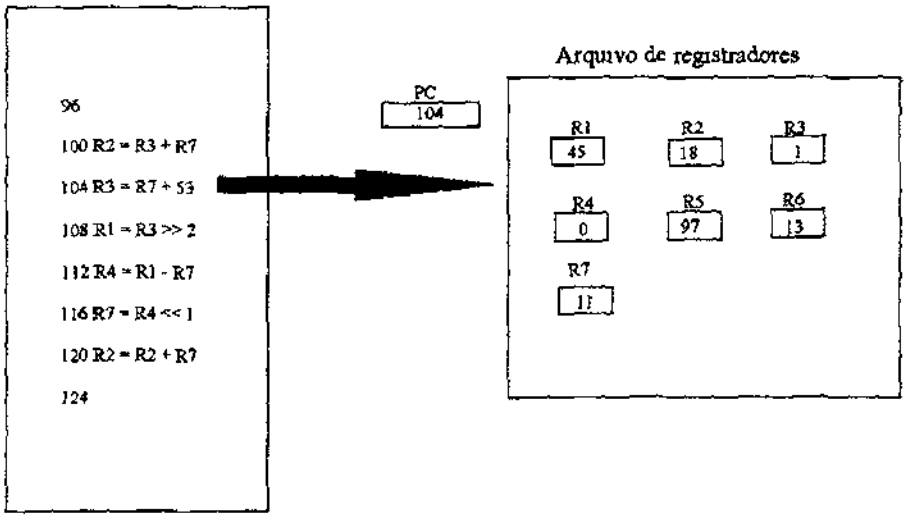

Figura7

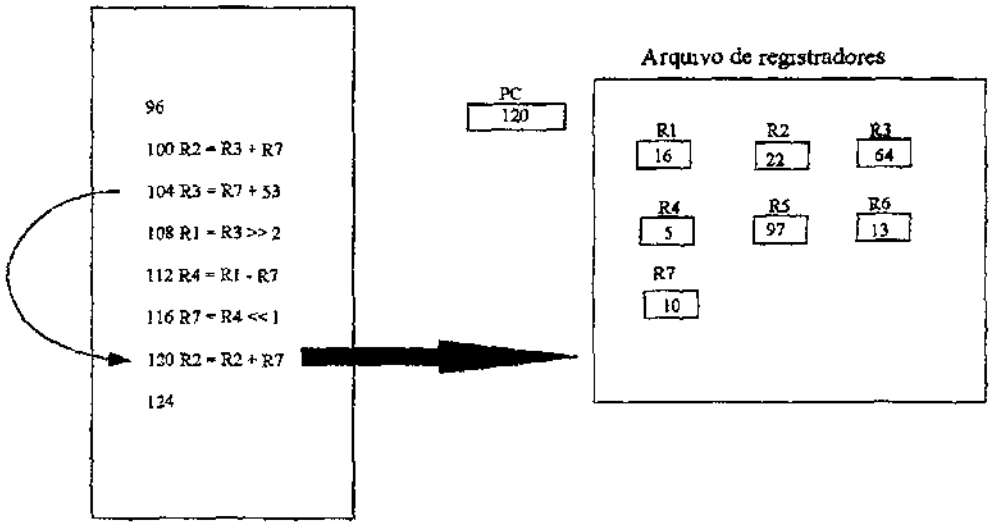

Figura 8

PI9902725

RESUMO

Patente de Invenção "Processo de Formação, Memorização e Reuso, em Tempo de Execução, de Seqüências de Instruções Dinâmicas em Computadores"

Trata-se de um processo para a formação, memorização e reuso, em tempo de execução, de seqüências de instruções dinâmicas com o objetivo de reduzir o número de instruções dinâmicas executadas por um programa, de forma a eliminar a necessidade de reexecução de instruções redundantes (instruções que reutilizam mais de uma vez os mesmos valores para seus operandos de entrada) inclusas na seqüência quando esta apresentar como argumentos de entrada valores já observados anteriormente e registrados em tabelas de memorização. Este processo não necessita efetuar invalidações em entradas das tabelas de memorização ou possui seu uso limitado a cadeias de instruções dependentes. O processo reduz o número de instruções dinâmicas que devam ser executadas pelo programa alvo, reduzindo desta forma o tempo total de execução do dado programa e mantendo-se a correção dos resultados.

A invenção refere-se também a um dispositivo eletrônico para a realização do processo.