Balanceamento de Carga em uma Implementação Distribuída do Problema dos N-Corpos

Isabel Cristina Mello Rosseti
Noemi Rodriguez
rosseti,noemi@inf.puc-rio.br
PUC-RioInf.MCC29/98 Junho, 1998

Abstract
Load balancing has become a fundamental factor in the effective use of distributed-memory environments for parallel computation. This paper studies the application of load-balancing algorithms to a parallel simulation of the N-body problem. The Matrix Iterative Model is used as an unifying basis for the implementation of a load-balancing library. Experimental design and analysis techniques are used to examine application performance with the use of this library.

Keywords
Load balancing, load balancing algorithms, matrix iterative model, N-body problem.

Resumo
O balanceamento de carga é um fator fundamental no uso efetivo de ambientes de memória distribuída para execução de aplicações paralelas. Este trabalho estuda a aplicação de três algoritmos de balanceamento a uma simulação paralela de um sistema de N-corpos. Um modelo analítico unificador, o Modelo Iterativo da Matriz, é usado como base para a implementação de uma biblioteca única de balanceamento de carga. O trabalho utiliza técnicas de projeto e análise de experimentos para examinar o desempenho da aplicação com uso dessa biblioteca.

Palavras-chave
Balanceamento de carga, algoritmos de balanceamento de carga, Modelo Iterativo de Matriz, problema dos N-corpos.

1 Introdução

A implementação de problemas paralelos em ambientes de memória distribuída vem se tornando cada vez mais comum. Até alguns anos atrás, a velocidade de comunicação entre processadores muitas vezes fazia com que a comunicação tivesse um custo alto demais neste tipo de ambiente. Hoje, muitas redes locais ou *switches* de comunicação oferecem velocidades convidativas para aplicações paralelas. No entanto, estas aplicações muitas vezes apresentam uma eficiência abaixo da desejável devido ao desbalanceamento de carga entre os diversos processadores no sistema.

Nesse trabalho, o problema de balanceamento de carga é estudado no contexto de uma aplicação paralela particular; descreve-se a utilização de três algoritmos de balanceamento na simulação paralela da evolução de um sistema de N-corpos, baseada no algoritmo de Barnes-Hut. Além de seu interesse próprio, indicado pelo número de trabalhos que a tratam [GKS94, Vel94, BN94], essa aplicação ainda pode ser interpretada como representante de um conjunto mais amplo de aplicações distribuídas, como discutido na seção 3.

Franklin e Govindan propõem um modelo [FG96], denominado *Modelo Iterativo de Matriz* (MIM), que permite representar uma gama bastante grande de algoritmos de balanceamento. O MIM foi utilizado neste trabalho como base para a organização da implementação do balanceamento de carga. Essa abordagem permitiu que fossem minimizadas as diferenças entre as implementações dos diversos algoritmos. Foi desenvolvida uma biblioteca onde é enfatizada a separação da funcionalidade comum a todos os algoritmos.

A seção 2 apresenta a terminologia utilizada e discute, de forma resumida, algumas questões relevantes para o problema de balanceamento de carga. A seção 3 descreve o problema dos N-corpos e a simulação paralela utilizada. Na seção 4, discute-se a implementação da biblioteca de balanceamento. A seção 5 apresenta os resultados dos experimentos realizados. Finalmente, a seção 6 discute algumas conclusões deste trabalho.

2 Balanceamento de Carga

O balanceamento de carga em um sistema distribuído é o processo de redistribuição de tarefas entre nós com o objetivo de utilizar efetivamente os recursos computacionais do sistema, isto é, evitar que existam recursos ociosos enquanto existem tarefas a serem executadas.

Um esquema de balanceamento de carga pode ser classificado, conforme sua relação com o estado corrente de carga do sistema, em estático, dinâmico ou adaptativo. Os esquemas de balanceamento de carga estáticos independem do estado do sistema. Os dinâmicos consideram o estado do sistema e os esquemas adaptativos mudam seus parâmetros ou políticas para refletir as mudanças do estado do sistema [LLC96].

Três componentes importantes do balanceamento de carga adaptativo e dinâmico são: a política de transferência, a política de informação e a política de localização. A política de transferência determina se uma tarefa deve ser transferida. A política de informação determina quais informações serão usadas para fazer as redistribuições de carga e como elas serão distribuídas entre os participantes. Isto é, determina as regras para coletar

informações sobre os nós participantes do sistema. Finalmente, a política de localização é responsável por achar os nós de destino para onde as tarefas serão transferidas [LLC96].

2.1 Modelo Iterativo de Matriz

O Modelo Iterativo de Matriz (MIM), proposto em [FG96], é um modelo analítico que pode ser usado para representar um conjunto de algoritmos de balanceamento de carga. O modelo descreve a distribuição de tarefas entre os processadores devido ao algoritmo de balanceamento.

O modelo assume que:

(a) a aplicação pode ser dividida em um grande número de tarefas (comparado ao número de processadores);

(b) as tarefas são idênticas, com relação aos custos computacionais e de comunicação, e podem ser executadas por qualquer processador disponível;

(c) os processadores podem ter diferentes poderes computacionais;

(d) a aplicação pertence à classe das aplicações iterativas síncronas. Essa classe abrange algoritmos que geralmente possuem as seguintes características: a execução segue por um número determinado de iterações e o trabalho associado com cada iteração é distribuído entre os processadores participantes do sistema; Ao fim de cada iteração, os nós fazem uma sincronização e, então, iniciam a próxima iteração.

O conjunto de processadores disponíveis é denotado por $P = \{P_1, ..., P_p\}$, onde p é o número de processadores.

O MIM descreve a distribuição das tarefas de cada processador do sistema, em cada uma das iterações k ($k = 1, 2, ...$), quando o algoritmo de balanceamento de carga é invocado. O estado do sistema em k é representado por um vetor de distribuição de tarefas $w(k)$, onde $w_i(k)$ é o número de tarefas no processador P_i em k.

A distribuição de tarefas na iteração $(k+1)$, $w(k+1)$, pode ser expressa como função de:

1. $w(k)$, a distribuição de tarefas em k;
2. a transferência de tarefas devido ao balanceamento de carga invocado em k;
3. a chegada de tarefas antes da *(k + 1)-ésima* invocação do algoritmo de balanceamento de carga;
4. a saída de tarefas antes da *(k + 1)-ésima* invocação do algoritmo de balanceamento de carga.

O modelo de transferência dinâmico para cada processador, P_i, pode ser representado por:

$w_i(k+1) = w_i(k) -$ *tarefas enviadas por* P_i *em* k *devido ao balanceamento de carga +* *tarefas recebidas por* P_i *em* k *devido ao balanceamento de carga +* *chegada de tarefas durante* $[k, k+1] -$ *saída de tarefas durante* $[k, k+1]$ {2.1}

Quando um algoritmo de balanceamento de carga é invocado, cada processador decide se ele quer participar do balanceamento. A participação do processador no balanceamento de carga em k é representado pela matriz $A(k)$. Se o processador P_i decide participar do balanceamento de carga, o elemento $A_{ii}(k)$ recebe o valor 1; todos os outros elementos recebem o valor 0.

Cada processador que decide participar do balanceamento de carga tem que descobrir o destino das tarefas a serem transferidas. Essa decisão de migração de tarefas é representada por uma matriz $p \times p$, $M(k)$, onde $M_{ij}(k)$ indica a quantidade de tarefas, α, que o processador P_i decide enviar para P_j. A decisão de migração fica necessariamente restrita à estrutura imposta pelo algoritmo de balanceamento de carga ou pela topologia do sistema. Define-se $H(k)$ como a matriz de topologia. $H(k)$ é uma matriz $p \times p$ que descreve os caminhos de migração possíveis. O elemento $H_{ij}(k) = 1$ se P_i e P_j trocam tarefas. Caso contrário, $H_{ij}(k) = 0$. Portanto, na construção de $M(k)$, $M_{ij}(k) \neq 0$ se $H_{ij}(k) = 1$.

A equação {2.1} pode ser reescrita como segue:

$w_i(k+1) = w_i(k) - \sum_{j \neq i} M_{ij}(k) A_{ii}(k) w_i(k) + \sum_{j \neq i} M_{ji}(k) A_{jj}(k) w_j(k) + \lambda_i(k) - \mu_i(k)$ {2.2}

onde $\lambda(k)$ e $\mu(k)$ são vetores que denotam a taxa de chegada e a taxa de saída, respectivamente.

As duas somatórias representam a atividade de balanceamento de carga. A primeira corresponde à soma de todas as tarefas transferidas de P_i para os outros processadores e a segunda representa a soma de todas as tarefas transferidas para P_i.

2.2 Algoritmos de Balanceamento de Carga Apresentados Segundo o MIM

Em [FG96], os procedimentos de balanceamento de carga *random*, *difusão* e *redistribuição completa* foram adaptados ao MIM. Nessa subseção, os algoritmos de balanceamento de carga *random*, *threshold* e *shortest* serão adaptados a esse modelo. Esses foram os algoritmos escolhidos para os experimentos com a simulação paralela dos N-corpos, por representarem políticas bastante distintas de informação e de localização.

- *random* → nesse algoritmo, descrito em [ELZ86], [FG96], [LLC96] e [MD94], um processador P_i decide enviar uma fração α do excesso de suas tarefas para outro processador, P_e, quando o número de tarefas atribuídas a ele, $w_i(k)$, é maior que um valor pré-definido, $w_i^*(k)$. Assim, o processador P_i envia $\alpha(w_i(k) - w_i^*(k))$ tarefas, se $(w_i(k) > w_i^*(k))$. O processador de destino P_e é escolhido randomicamente, do conjunto de processadores, para o envio dessa carga;

- *threshold* → nesse algoritmo, descrito em [ELZ86], [LLC96] e [MD94], um processador P_i decide enviar uma fração α do excesso de suas tarefas para outro proces-

sador, P_e, quando o número de tarefas atribuídas a ele, $w_i(k)$, é maior que um valor pré-definido, $w_i^*(k)$.

O processador de destino P_e é escolhido randomicamente do conjunto de processadores. Se o número de tarefas atribuídas a P_e, $w_e(k)$, for menor que o valor de *threshold*, $w_e^*(k)$, transfere-se $\alpha(w_i(k) - w_i^*(k))$ tarefas para P_e. Caso contrário, repete-se o processo de seleção de outro nó;

- *shortest* → nesse algoritmo, descrito em [ELZ86] e [LLC96], um processador P_i decide enviar uma fração α do excesso de suas tarefas para outro processador, P_e, quando o número de tarefas atribuídas a ele, $w_i(k)$, é maior que um valor pré-definido, $w_i^*(k)$.

  Um conjunto de processadores é escolhido randomicamente do sistema. Desse conjunto, obtém-se o processador, P_e, com o menor número de tarefas atribuídas a ele. Se o número de tarefas atribuídas a P_e, $w_e(k)$, for menor que o valor de *threshold*, $w_e^*(k)$, transfere-se $\alpha(w_i(k) - w_i^*(k))$ tarefas para P_e. Caso contrário, repete-se o processo de seleção de um conjunto de nós.

A escolha do vizinho para o envio das tarefas excedentes no processador sobrecarregado é a única diferença existente entre os 3 algoritmos. No algoritmo *random*, o vizinho é selecionado randomicamente. No algoritmo *threshold*, o vizinho é escolhido randomicamente, mas o envio das tarefas depende do nível de carga desse vizinho. Já no *shortest*, o vizinho com o menor número de tarefas é escolhido do conjunto de processadores selecionado randomicamente, mas o envio das tarefas depende do nível de carga desse vizinho.

Para esses algoritmos, um elemento da matriz de participação de processadores, $A_{ii}(k)$, é igual a 1 se $(w_i(k) > w_i^*(k))$. Todos os outros elementos de A são nulos. O elemento $M_{ie}(k) = \alpha$ e $M_{ij}(k) = 0$ se $j \neq e$.

A equação {2.2} pode ser reescrita como:

$$w_i(k+1) = w_i(k) - \alpha A_{ii}(k)(w_i(k) - w_i^*(k)) + \sum_{j \neq i} A_{jj}(k) M_{ji}(k)(w_j(k) - w_j^*(k)) + \lambda_i(k) - \mu_i(k) \quad \{2.3\}$$

2.3 Considerações sobre o Balanceamento de Carga

Os algoritmos de balanceamento de carga adaptativos que têm sido examinados na literatura [ELZ86], [FG96], [LLC96] e [MD94] coletam informações de estado do sistema e, baseados nessas informações, procuram fazer a melhor redistribuição da execução das tarefas de uma aplicação [ELZ86].

Claramente, a complexidade dessa coleta de informações introduz alguns pontos interessantes:

1. o efeito do trabalho adicional na coleta das informações → o uso de um algoritmo de balanceamento de carga adaptativo depende criticamente do trabalho adicional requerido para coletar as informações de estado do sistema. Esse trabalho pode influir diretamente no tempo de execução de uma aplicação submetida ao procedimento de balanceamento. A introdução de uma excessiva quantidade de trabalho para coleta

de informações pode anular os benefícios de uma redistribuição da execução das tarefas de uma aplicação;

2. o efeito de uma redistribuição das tarefas de uma aplicação → algoritmos de balanceamento de carga complexos contam com informações detalhadas a respeito do estado do sistema. Esse tipo de informação gasta muito tempo para ser coletada e não consegue determinar a sobrecarga ou subcarga de um nó num futuro próximo. Devido a isso, uma redistribuição das tarefas de uma aplicação feita por um algoritmo de balanceamento de carga complexo que supõe estar próxima da redistribuição ótima, pode ser, de fato, muito ruim.

A partir dos pontos levantados anteriormente, surge a seguinte questão com relação aos algoritmos de balanceamento de carga adaptativos: qual é o nível desejável de complexidade desses algoritmos? Em [ELZ86], [MD94] e [FG96] foi mostrado que:

- algoritmos de balanceamento de carga adaptativos simples – algoritmos com pouca ou nenhuma coleta de informações de estado, como *threshold* e *random*, respectivamente – produzem uma melhoria no desempenho em relação ao caso onde não há a redistribuição das tarefas;
- algoritmos de balanceamento de carga adaptativos simples, como *threshold* e *random*, têm o desempenho melhor que algoritmos de balanceamento de carga que coletam uma grande quantidade de informação de estado do sistema e que tentam fazer a "melhor" redistribuição da execução, como *shortest*, *difusão* e *redistribuição completa*;
- esses resultados são válidos para um grande intervalo de parâmetros do sistema como, por exemplo, o seu tamanho.

Em [ELZ86] conclui-se que algoritmos de balanceamento de carga adaptativos simples são mais interessantes do ponto de vista prático e, de modo geral, não existem evidências de que custos de coletar e usar um volume grande de informações de estado tragam algum benefício real. Entretanto, muitos algoritmos de balanceamento de carga adaptativos complexos têm sido propostos na literatura [MD94] e [LLC96], visando a obtenção do mesmo tempo de execução dos algoritmos simples e, além disso, uma melhor redistribuição das tarefas de uma aplicação.

3 Problema dos N-Corpos

Técnicas clássicas de N-corpos estudam a evolução de um sistema de N partículas (ou corpos) onde existe uma força (por exemplo, a força gravitacional) entre pares de partículas. Essas técnicas são amplamente aplicadas a problemas de astrofísica, de dinâmica dos fluidos, entre outros.

A simulação sequencial discutida nesse trabalho está baseada no algoritmo de Barnes-Hut, descrito em [CT92], [Vel94] e [BH86], para estudar a evolução de um sistema de corpos sob a influência da atração gravitacional Newtoniana. Os corpos consistem de

uma massa, posições e velocidades iniciais e eles estão distribuídos sobre um domínio físico finito. Essa simulação segue sobre um número pré-determinado de iterações, onde, em cada iteração, há o cálculo das forças em cada corpo e há a atualização de sua posição e de outros atributos.

Na simulação paralela do problema dos N-corpos usada neste trabalho, devida a [FG93] e [FG96], o domínio físico é dividido em pequenas regiões e cada região é alocada a um processador. Inicialmente, os domínios são distribuídos de forma que cada processador tenha o mesmo número de partículas. Cada processador é responsável por todos os cálculos associados com as partículas de sua região. Chama-se de tarefa o conjunto de cálculos associados a uma única partícula.

Para calcular as forças em cada partícula, é necessário conhecer a posição de outras partículas e isso envolve troca de informação entre pares de processadores. Nessa fase, cada processador troca $log\ p$ mensagens com os outros $p - 1$ processadores.

Depois da troca de informações, cada processador calcula a força resultante em cada partícula de sua região e atualiza suas posições e velocidades. O processador verifica quais de suas partículas atravessaram as fronteiras de seu domínio e, então, transfere as partículas que não pertencem mais a ele para os processadores apropriados.

Essa transferência causa um desequilíbrio na distribuição de carga entre os processadores. O algoritmo de balanceamento de carga é invocado ao fim de cada iteração. O balanceamento de carga consiste em redesignar domínios para cada processador, tal que o número de partículas associadas aos processadores esteja balanceado.

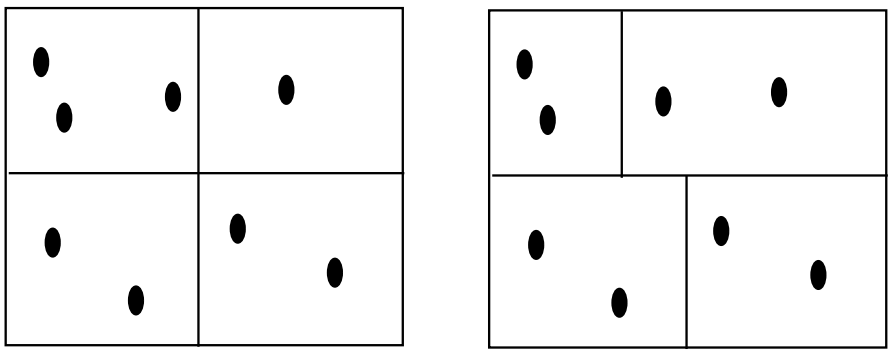

Figura 1: Simulação dos N-corpos antes e após a invocação do algoritmo de balanceamento de carga

É interessante observar que a evolução de um sistema de N-corpos pode ser usada para modelar a evolução da execução de um sistema operacional distribuído, onde novas tarefas surgem e outras terminam de maneira arbitrária. A mudança de posição de uma partícula de uma região para outra pode ser interpretada como uma tarefa terminando em um processador e uma nova tarefa aparecendo na fila de execução do segundo processador. Em um sistema distribuído real, obviamente o número total de tarefas não seria fixo, e

nem todas as tarefas seriam iguais. Ainda assim, a observação do comportamento do sistema de n-corpos parece poder trazer insumos importantes para o balanceamento em um sistema operacional distribuído.

4 Biblioteca de Balanceamento de Carga

Utilizando o MIM como base, podemos capturar as partes comuns entre os vários algoritmos de balanceamento de carga, descrevendo um algoritmo de balanceamento genérico através dos seguintes passos:

1. calcula matriz A (define participacao de processadores)
2. calcula matriz H (determina vizinhacas correntes)
3. calcula matriz M (determina destino de tarefas transferidas)
4. executa trocas de carga entre processadores descritas por A, H e M

Os três primeiros passos tratam das matrizes $A(k)$, $H(k)$ e $M(k)$, e foram implementados por funções $MatrixA$, $MatrixH$ e $MatrixM$, respectivamente. O último passo foi implementado pela função $ExchangeLoad$.

É necessário enfatizar que todos os algoritmos de balanceamento de carga que podem ser descritos pelo MIM se encaixam nessa mesma descrição do algoritmo geral. Para esses algoritmos, as implementações de $MatrixH$ e $ExchangeLoad$ são idênticas. Já as implementações de $MatrixA$ e $MatrixM$ são especiais para cada algoritmo de balanceamento. Do ponto de vista da aplicação, a escolha de algoritmo de balanceamento é transparente. Ao final de cada iteração da simulação, a aplicação chama um procedimento de balanceamento de carga, que basicamente chama as funções $MatrixA$, $MatrixH$, $MatrixM$ e $ExchangeLoad$. No momento de ligação, o programador escolhe qual instância específica de de $MatrixA$ e $MatrixM$ ele deseja utilizar (através de um arquivo *Makefile*).

Para estender a biblioteca desenvolvida com um novo algoritmo, seria necessário apenas desenvolver uma nova implementação das funções $MatrixM$ e $MatrixA$. Em particular, nos três algoritmos estudados a implementação de $MatrixA$ utilizada foi a mesma, uma vez que todos apresentam a mesma política de transferência, baseada no fato do número corrente de tarefas ultrapassar um valor pré-definido (essa política é bastante comum nos assim chamados algoritmos *iniciados pelo remetente*).

5 Análise de Resultados

Os experimentos foram realizados na máquina paralela IBM SP/2 do Núcleo de Computação Eletrônica (NCE) da Universidade Federal do Rio de Janeiro (UFRJ). Os dados foram obtidos utilizando 4 nós da máquina de maneira exclusiva, isto é, a simulação foi executada sem o compartilhamento do poder computacional dos nós com nenhuma outra aplicação. Cada experimento foi repetido 10 vezes.

Foram usadas no experimento simulações com 4096 e 16384 partículas. Para os valores inferiores a 4096, a paralelização do sistema não é vantajosa, pois o tempo de resposta da simulação paralela é similar ao tempo gasto no algoritmo sequencial (tabela 1). Essa similaridade do tempo paralelo é devido ao tempo gasto na fase de troca de informações entre os processadores. Conforme mencionado na seção 3, cada processador troca $log\ p$ mensagens com os outros $p - 1$ processadores. Se o tempo gasto com processamento é pequeno, como acontece com números baixos de partículas, o custo da fase de comunicação faz com que haja um aumento do tempo total de execução, em relação a simulação sequencial. Quando o número de corpos cresce, o custo de comunicação é compensado pelo tempo gasto no cálculo das forças resultantes nos corpos (uma das etapas integrantes da fase de processamento).

Todas as posicões e velocidades iniciais das partículas seguem a Distribuição Uniforme.

A fim de validar a implementação descrita na seção 4, serão analisados os tempos gastos nas execuções da simulação paralela antes e após a inclusão da biblioteca de balanceamento desenvolvida. Para fazer essa análise, foram realizados 2 projetos de experimentos [Jai91]. Esses projetos serão tratados nas subseções seguintes.

partículas	*sequencial*	*paralela*	*partículas*	*sequencial*	*paralela*
	3682	3633		142696	78010
	3664	3749		142936	76935
	3692	4069		143042	77664
	3659	3858		143220	79344
512	3684	4155	4096	143259	76861
	3627	3507		143240	73949
	3699	3445		143127	77941
	3693	3819		143264	72659
	3650	312.95		144464	79549
	3667	412.92		143424	73439
	18590	18498		1114267	472471
	17815	18299		1109664	483067
	18343	18229		1111704	496344
	18475	19835		1127939	473277
2048	18316	18129	16384	1111381	461799
	18318	18676		1112772	468958
	18341	19227		1114053	479472
	18031	18733		1127399	487782
	18264	18922		1114257	493110
	18565	19133		1113257	498025

Tabela 1: Tempos (em ms) de cada uma das 10 execuções das simulações, sequencial e paralela

5.1 Análise das Alternativas de Balanceamento de Carga

Nessa subseção, o projeto com 2 fatores [Jai91] é usado para selecionar qual foi a simulação paralela com redistribuição de tarefas, gerada a partir da biblioteca de balanceamento, mais eficiente.

Os fatores a serem considerados na análise desse sistema são o algoritmo de balanceamento de carga e o tipo de balanceamento utilizado. Foram implementados 3 algoritmos de balanceamento: *random*, *threshold* e *shortest*. Já o fator balanceamento pode assumir um dos 2 níveis:

(a) com balanceamento de carga sem certeza de envio (CBS) → essa opção é usada quando se quer medir o tempo de execução da simulação paralela do problema dos N-corpos com o uso da biblioteca de balanceamento de carga. Nessa opção, a redesignação de domínios feita no algoritmo de balanceamento ou vai aumentar ou vai diminuir a fronteira de seu domínio de um valor pré-determinado, l, e vai fazer as modificações nas fronteiras desse valor. Nesse caso, não se sabe a quantidade de partículas que um nó com sobrecarga enviará. Um processador sobrecarregado pode diminuir de l a sua fronteira e, mesmo assim, continuar nesse estado, se a nova fronteira que lhe foi atribuída não é suficiente para o envio do excesso de suas partículas. Ou esse processador pode tornar-se subcarregado porque, com a mudança de sua fronteira, ele pode enviar, para o seu vizinho escolhido, mais corpos que o necessário. Assim, a redesignação de domínios não garante o equilíbrio do número de corpos associados aos processadores sobrecarregados. É importante observar que esse balanceamento não garante que o sistema vai ficar equilibrado ao fim da iteração;

(b) com balanceamento de carga com certeza de envio (CBC) → essa opção é usada quando se quer medir o tempo de execução da simulação paralela do problema dos N-corpos com o uso da biblioteca de balanceamento de carga. Nesse opção, a redesignação de domínios feita no algoritmo de balanceamento faz uma ordenação de partículas e, a partir dessa ordenação, calcula-se o valor l. Com esse valor calculado, faz-se as modificações nas fronteiras. Nesse caso, sabe-se a quantidade de partículas que um nó com sobrecarga enviará. Um processador sobrecarregado diminui de l a sua fronteira, envia o excesso de corpos e torna-se um nó balanceado. Assim, a redesignação de domínios garante o equilíbrio do número de corpos associados aos processadores sobrecarregados. É importante observar que esse balanceamento não garante que o sistema vai ficar equilibrado ao fim da iteração, pois um processador que estava subcarregado pode, depois de receber o excesso de tarefas de outro processador, tornar-se sobrecarregado.

5.1.1 Modelo do Projeto 2-Fatorial

Considere um projeto com a níveis do fator A, b níveis do fator B e r replicações do experimento com cada um desses ab níveis. Nesse caso, o modelo é dado por:

$y_{ijk} = \mu + \alpha_j + \beta_i + \gamma_{ij} + e_{ijk}$

$i = 1,...,b$

$j = 1,...,a$

$k = 1,...,r$ {5.1}

onde:

- y_{ijk} é a resposta na k-ésima repetição do experimento com o fator A no nível j e com o fator B no nível i;
- μ é a resposta média do sistema;
- α_j é o efeito do fator A no nível j;
- β_i é o efeito do fator B no nível i;
- γ_{ij} é o efeito da interação entre o fator A no nível j e o fator B no nível i;
- e_{ijk} é o erro experimental obtido na k-ésima replicação do experimento com o fator A no nível j e com o fator B no nível i.

O modelo supõe que:

- os efeitos são calculados de forma que sua soma seja igual a zero
  $\sum_{j=1}^{a} \alpha_j = 0$
  $\sum_{i=1}^{b} \beta_i = 0$
- as interações entre os fatores são calculados de forma que a soma de suas linhas e colunas seja igual a zero
  $\sum_{i=1}^{b} \gamma_{i1} = \sum_{i=1}^{b} \gamma_{i2} = ... \sum_{i=1}^{b} \gamma_{ia} = 0$
  $\sum_{j=1}^{a} \gamma_{1j} = \sum_{j=1}^{a} \gamma_{2j} = ... \sum_{j=1}^{a} \gamma_{bj} = 0$
- os erros são calculados de forma que sua soma seja igual a zero
  $\sum_{k=1}^{r} \sum_{j=1}^{a} \sum_{i=1}^{b} e_{ijk} = 0$

Os valores medidos são dispostos em uma matriz $a \times b$ onde cada dimensão corresponde a um fator considerado. O conjunto de repetições de um dado experimento pode ser visto como ocupando a mesma célula dessa matriz. Dessa forma, cada célula da matriz conterá r replicações do mesmo experimento.

Calculando-se a média dos experimentos de uma célula da matriz, obtém-se que:

$\bar{y}_{ij.} = \mu + \alpha_j + \beta_i + \gamma_{ij}$ {5.2}

O ponto (.) indica que a média é feita entre todas as repetições da célula.

Seja y_{ijk} o elemento (i,j,k) dessa matriz. Substituindo cada y_{ijk} da matriz pela equação {5.1}, obtêm-se abr equações. Adicionando as abr equações, obtém-se a seguinte equação para a resposta média do sistema:

$\mu = 1/abr \sum_{k=1}^{r} \sum_{j=1}^{a} \sum_{i=1}^{b} y_{ijk}$ {5.3}

A fração da equação {5.3} é conhecida como média global do sistema e é denotada por $\bar{y}_{...}$.

A média da coluna j é dada por:

$$\bar{y}_{.j.} = 1/br \sum_{k=1}^{r} \sum_{i=1}^{b} y_{ijk} \quad \{5.4\}$$

Substituindo a equação {5.1} na equação {5.4}, tem-se que:

$$\alpha_j = \bar{y}_{.j.} - \mu = \bar{y}_{.j.} - \bar{y}_{...} \quad \{5.5\}$$

Analogamente, β_i é dado por:

$$\beta_i = \bar{y}_{i..} - \mu = \bar{y}_{i..} - \bar{y}_{...} \quad \{5.6\}$$

Substituindo a equação {5.6}, {5.5} e {5.3} em {5.2}, obtém-se que:

$$\gamma_{ij} = \bar{y}_{ij.} - \bar{y}_{i..} - \bar{y}_{.j.} + \bar{y}_{...} \quad \{5.7\}$$

Finalmente, e_{ijk} é dado por:

$$e_{ijk} = y_{ijk} - \bar{y}_{ij.} \quad \{5.8\}$$

A variação total da observação y_{ijk}, SST, em um projeto 2-fatorial é devida aos fatores considerados, a interação entre os fatores e aos erros experimentais. Para calculá-la, eleva-se a equação {5.1} ao quadrado. Assim:

$$SST = SSA + SSB + SSAB + SSE$$

onde:

$$SSO = \sum_{i,j,k} \mu^2 = abr\mu^2$$

$$SSA = \sum_{i,j,k} \alpha_j^2 = br \sum_j \alpha_j^2$$

$$SSB = \sum_{i,j,k} \beta_i^2 = ar \sum_i \beta_i^2$$

$$SSAB = \sum_{i,j,k} \gamma_{ij}^2 = r \sum_{i,j} \gamma_{ij}^2$$

A variação total pode, portanto, ser dividida em duas partes, SSA, SSB e $SSAB$, e SSE, que representam as partes explicável e inexplicável da variação, respectivamente.

A alta porcentagem da variação explicável indica que o modelo foi empregado de maneira correta. Essa porcentagem é dada pela soma das seguintes percentagens:

% da variação explicada pelo fator A = $100(SSA/SST)$

% da variação explicada pelo fator B = $100(SSB/SST)$

% da variação explicada pela interação AB = $100(SSAB/SST)$

A porcentagem da variação não explicada pelo modelo é dada por:

% da variação não explicada pelo modelo = $100(SSE/SST)$

5.1.2 Considerações sobre as Alternativas de Balanceamento de Carga

As análises do projeto 2-fatorial, descritas em [Ros98], revelaram que a simulação com o algoritmo *shortest* obtiveram melhor desempenho que as simulações com os algoritmos *random* e *threshold* (ver tabela 2). Esses resultados são coerentes com as conclusões

de [ELZ86], considerando-se que a superioridade do algoritmo *shortest* pode ser explicada pela simplicidade do sistema testado. Como somente 4 processadores participaram do sistema, ficou mais fácil escolher, em cada iteração que esse algoritmo era invocado, os processadores que estavam menos sobrecarregados. Essa facilidade na escolha garantiu uma boa redistribuição das tarefas da aplicação e, portanto, diminuiu os tempos do algoritmo em relação aos demais.

Os valores obtidos na execução da simulação paralela com os algoritmos de balanceamento *random*, *difusão* e *redistribuição completa*, discutidos em [FG96], também estão em concordância com o que foi concluído em [ELZ86]. Nas execuções descritas em [FG96], prova-se que, para 16 processadores, o algoritmo *random* é 12% mais rápido que o algoritmo de *difusão*, e 30% mais rápido que o algoritmo de *redistribuição completa*. Para um sistema pequeno, com 2 ou 4 processadores participantes, as simulações com os algoritmos *random* e *difusão* têm, praticamente, o mesmo tempo de execução. Conforme discutido no parágrafo anterior, a simulação com o algoritmo de balanceamento *difusão* só obtém o mesmo tempo de execução do algoritmo *random* porque um processador sobrecarregado descobre, mais facilmente, a porcentagem de tarefas a serem recebidas por cada um de seus vizinhos.

Foi observado, também, que a simulação paralela com balanceamento de carga sem certeza de envio obteve melhor desempenho que a simulação paralela com balanceamento de carga com certeza de envio. Essa constatação pode ser explicada da seguinte maneira: quando ocorre um desequilíbrio nas tarefas dos processadores participantes da simulação paralela com balanceamento de carga com certeza de envio, os processadores sobrecarregados ordenam seus corpos e, depois dessa ordenação, diminuem suas regiões de um valor calculado a partir da ordenação e enviam uma quantidade do excesso de seus corpos para os processadores subcarregados (que aumentam suas regiões do referido valor para poder receber esses excessos). É interessante observar que, além da ordenação ser uma operação dispendiosa, esse balanceamento não garante que, ao final da iteração, os processadores do sistema fiquem com o número de tarefas equilibrado, pois, como foi mencionado anteriormente, um processador que estava subcarregado pode, depois de receber o excesso de tarefas de outro processador, tornar-se sobrecarregado.
Na simulação com balanceamento de carga sem certeza de envio, os processadores sobrecarregados diminuem suas regiões de um valor constante e transferem os corpos que não pertencem mais a eles para os processadores subcarregados (que aumentam suas regiões dessa mesma constante para poder receber esses corpos). Da mesma forma que a simulação com certeza de envio, esse balanceamento não garante que o sistema vai ficar equilibrado ao fim da iteração, pois um processador que estava subcarregado pode, depois de receber o excesso de tarefas de outro processador, tornar-se sobrecarregado. Entretanto, nesse balanceamento não se gasta tempo na fase de ordenação de corpos.

Considerando-se que o objetivo desse trabalho é estudar como o balanceamento de carga pode melhorar o desempenho da aplicação, a simulação paralela com o algoritmo de balanceamento *shortest* sem certeza de envio será comparado com a simulação paralela original na subseção 5.2.

5.2 Análise do Balanceamento de Carga na Simulação Paralela

Nessa subseção, o projeto com 1 fator [Jai91], o balanceamento de carga, é usado para comparar os desempenhos da simulação paralela do problema dos N-corpos antes e após a inclusão da biblioteca de balanceamento desenvolvida. Esse fator pode assumir um dos 2 níveis:

(a) sem balanceamento de carga (SBC) → essa opção é usada quando se quer medir o tempo de execução da simulação paralela do problema dos N-corpos sem a intervenção da biblioteca de balanceamento de carga;

(b) a alternativa selecionada no 1º projeto realizado, isto é, o algoritmo *shortest* sem certeza de envio.

O modelo do projeto 1-fatorial é uma simplificação do modelo do projeto 2-fatorial porque, quando o fator algoritmo de redistribuição do projeto 2-fatorial é desconsiderado, os dois modelos são similares. Devido a essa similaridade, pode-se adaptar o modelo descrito na subseção 5.1.1 para realizar a análise descrita nessa subseção.

A análise do projeto de experimentos 1-fatorial, descrita em [Ros98], revelou que a simulação paralela com o algoritmo de balanceamento de carga *shortest* sem certeza de envio obteve o melhor desempenho em relação à simulação paralela pura. Essa conclusão pode ser explicada da seguinte maneira: quando ocorre um desequilíbrio nas tarefas dos processadores participantes da aplicação paralela original, os processadores que estão subcarregados devem ficar esperando, na fase de sincronização, pelos processadores sobrecarregados, em todas as iterações restantes da simulação, aumentando, assim, o tempo de execução do sistema. Na simulação com balanceamento de carga, tenta-se manter um número de corpos equilibrado entre todos os processadores desse sistema, a fim de minimizar os tempos gastos, desnecessariamente, na fase de sincronização dos processadores e, diminuindo, portanto, os tempos gastos nas execuções das simulações com balanceamento.

partículas	*random*		*threshold*		*shortest*	
	CBS	CBC	CBS	CBC	CBS	CBC
4096	54114	63796	54857	61890	52374	59071
	50372	62724	55212	60784	50682	59066
	55502	62734	55947	62596	56388	58037
	57293	62759	54754	61605	54198	57322
	53624	61480	55507	62143	53122	59579
	56261	64509	56000	62029	52851	56409
	52960	61250	54763	61296	52838	59092
	58263	62655	52372	62123	50273	59079
	58802	64102	55125	65343	54158	58672
	55198	62099	57006	63997	54322	56364
16384	365677	408124	369368	414846	352101	423809
	355702	407559	392752	437041	342705	417687
	344331	425001	378279	409581	348132	404408
	377706	401598	381000	405884	342948	418022
	361188	408636	357954	408678	330032	422257
	360509	408736	357644	409825	338566	429812
	341438	409079	370662	418337	345639	422233
	339157	407894	373869	416590	342104	429029
	365278	409351	373769	408300	345361	421760
	347464	424015	354003	409617	350124	411258

Tabela 2: Tempos (em ms) de cada uma das 10 execuções da simulação paralela antes e após a inclusão dos algoritmos de balanceamento de carga

Fator	*Variação Percentual (%)*	*Fator*	*Variação Percentual (%)*
Algoritmo	14.11	Algoritmo	2.16
Balanceamento	68.29	Balanceamento	85.90
Alg/Bal	1.77	Alg/Bal	0.47
Erros	15.83	Erros	12.27

Tabela 3: Variação percentual dos efeitos considerados para o sistema com 4096 corpos e com 16384 corpos

Fator	*Variação Percentual (%)*	*Fator*	*Variação Percentual (%)*
Balanceamento	97.086	Balanceamento	98.217
Erros	2.914	Erros	1.783

Tabela 4: Variação percentual do fator balanceamento para o sistema com 4096 corpos e com 16384 corpos

6 Conclusões

Com um maior entendimento do problema do balanceamento de carga em ambientes distribuídos, adquirido no desenvolvimento desse trabalho, foi possível estabelecer um paralelo entre as características básicas de alguns algoritmos de balanceamento, e desenvolver uma biblioteca básica extensível que serve para implementar uma grande variedade desses algoritmos.

Informalmente, nos experimentos realizados, a execução da simulação paralela do problema dos N-corpos com a biblioteca de balanceamento foi, em média, 40% mais rápida que a execução da simulação paralela original. Isso se deve ao fato de que, quando ocorre um desequilíbrio de partículas na simulação paralela pura, o trabalho adicional desse desequilíbrio se propaga até o fim da execução da referida aplicação. Já na simulação com a biblioteca de balanceamento, quando ocorre um desequilíbrio de corpos, o algoritmo de balanceamento é invocado, a fim de equilibrar, entre todos os processadores do sistema, a quantidade de partículas, impedindo que o referido trabalho adicional aconteça. Assim, o uso da biblioteca de balanceamento na simulação paralela é imprescindível para obter uma melhoria no desempenho dessa simulação.

O fato da execução da simulação paralela com um algoritmo de balanceamento de carga relativamente complexo, *shortest*, ser melhor que a execução da referida simulação com os algoritmos de balanceamento considerados simples, *random* e *threshold*, pode ser explicado pelo tamanho do sistema. Em geral, para sistemas com 2 ou 4 processadores, os algoritmos de balanceamento de carga complexos além de obterem o mesmo tempo de execução dos algoritmos simples, fazem uma melhor redistribuição de tarefas da aplicação. Quando o sistema cresce, como discutido em [FG96], os algoritmos complexos têm custos adicionais na tentativa da melhor redistribuição de tarefas, o que não ocorre com os algoritmos simples. Em sistemas pequenos, com 2 ou 4 processadores, se o objetivo é estudar como o balanceamento de carga pode melhorar o desempenho da aplicação, é aconselhável, pelos resultados obtidos, o uso de um algoritmo de balanceamento complexo. Em sistemas com mais de 4 processadores, o uso de algoritmo de balanceamento simples pode se tornar mais interessante.

O tempo de execução da simulação paralela com o algoritmo *shortest* sem certeza de envio é, em média, 10% mais rápido que o tempo de execução da referida aplicação com o algoritmo *shortest* com certeza de envio (essa observação é válida para os algoritmos de balanceamento de carga *random* e *threshold*). Isso se deve ao fato de que, quando ocorre um desequilíbrio de corpos na simulação com o algoritmo *shortest* com certeza de envio, os processadores sobrecarregados gastam muito tempo na fase de ordenação de seus corpos,

o que não acontece na simulação paralela com o algoritmo *shortest* sem certeza de envio. É necessário enfatizar que nenhuma das duas alternativas de balanceamento de carga garante um número de partículas equilibrado ao final da iteração, pois um processador que estava subcarregado pode, depois de receber o excesso de corpos de outro processador, tornar-se sobrecarregado. Essa possibilidade é uma decorrência inevitável da localidade de decisões nos algoritmos estudados.

Agradecimentos

Agradecemos ao Núcleo de Computação Eletrônica (NCE) da UFRJ, por permitir e apoiar a utilização de sua máquina paralela, e ao Prof. Celso Ribeiro, pelas sugestões dadas durante a condução deste trabalho.

Referências Bibliográficas

[BH86] J. Barnes and P. Hut. A Hierarchical $O(NlogN)$ Force Calculation Algorithm. *Nature*, 324:446–449, 1986.

[BN94] G. Blelloch and G. Narlikar. A practical comparison of n-body algorithms. In *Dimacs implementation challenge workshop*, 1994.

[CT92] K. M. Chandy and S. Taylor. *An Introduction to Parallel Programming*. Jones and Bartlett Publishers, Boston, 1992.

[ELZ86] D. L. Eager, E. D. Lazowska, and J. Zahorjan. Adaptive Load Sharing in Homogeneous Distributed Systems. *IEEE Transactions on Software Engineering*, 12(5):662–675, 1986.

[FG93] M. A. Franklin and V. Govindan. The N-body Problem: Distributed System Load Balancing and Performance Evaluation. In *Proc. 6th International Conference on Parallel and Distributed Computing Systems*, 1993.

[FG96] M. A. Franklin and V. Govindan. A General Matrix Iterative Model for Dynamic Load Balancing. *Parallel Computing*, 22:969–989, 1996.

[GKS94] A. Grama, V. Kumar, and A. Sameh. N-Body Simulations Using Message Passing Parallel Computers. In *SIAM Conference on Parallel Processing*, 1994.

[Jai91] R. Jain. *Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling*. Digital Equipment Corporation, John Wiley & Sons, inc., Littleton, Massachusetts, 1991.

[LLC96] G. Lee, H. Lee, and J. Cho. A Prediction-based Adaptive Location Policy for Distributed Load Balancing. *Journal of Systems Architecture*, 42:1–18, 1996.

[MD94] K. B. Mahieddine and P. M. Dew. A Periodic Symmetrically Initiated Load Balancing Algorithm for Distributed Systems. *Operating Systems Review*, 28(1):66–77, 1994.

[Ros98] I. C. M. Rosseti. Uma Biblioteca para Balanceamento de Carga em Ambientes Distribuídos. Dissertação de Mestrado, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Março de 1998.

[Vel94] E. F. Velde. *Concurrent Scientific Computing*. Spring Verlag, New York, 1994.

A Projeto de Experimentos

Antes de conceituar projeto de experimentos é necessário definir fatores e níveis. As variáveis que afetam o tempo de resposta de um experimento são chamadas de fatores. Os valores que um fator pode assumir são chamados de níveis.

A finalidade de um projeto de experimentos é obter, dos dados produzidos pelo menor número de execuções do sistema (ou experimentos), o máximo de informações a respeito de seu comportamento [Jai91].

O projeto de experimentos consiste de um conjunto de especificações, a saber:

(a) o número de experimentos;

(b) as variáveis (e suas combinações) que afetam o tempo de resposta de um experimento;

(c) o número de repetições (ou replicações) de um experimento.

Nesse capítulo, são realizados 2 projetos de experimentos:

- análise das alternativas de balanceamento de carga → nesse caso, um projeto com 2 fatores é usado para selecionar qual foi a simulação paralela com redistribuição de tarefas, gerada a partir da biblioteca de balanceamento, mais eficiente.
  Os fatores a serem considerados na análise desse sistema são o algoritmo de redistribuição de tarefas e o tipo de balanceamento utilizado. Foram implementados 3 algoritmos de balanceamento: *random*, *threshold* e *shortest*. Já o fator balanceamento pode assumir um dos 2 níveis:

  (a) com balanceamento de carga sem certeza de envio (CBS) → essa opção é usada quando se quer medir o tempo de execução da simulação paralela do problema dos N-corpos com o uso da biblioteca de balanceamento de carga. Nessa opção, a redesignação de domínios feita no algoritmo de balanceamento ou vai aumentar ou vai diminuir a fronteira de seu domínio de um valor pré-determinado, l, e vai fazer as modificações nas fronteiras desse valor. Nesse caso, não se sabe a quantidade de partículas que um nó com sobrecarga enviará. Um processador sobrecarregado pode diminuir de l a sua fronteira e, mesmo assim, continuar nesse estado, se a nova fronteira que lhe foi atribuída não é suficiente para o envio do excesso de suas partículas. Ou esse processador pode tornar-se subcarregado porque, com a mudança de sua fronteira, ele pode enviar, para o seu vizinho escolhido, mais corpos que o necessário. Assim, a redesignação de domínios não garante o equilíbrio do número de corpos associados aos processadores sobrecarregados. É importante observar que esse balanceamento não garante que o sistema vai ficar equilibrado ao fim da iteração;

  (b) com balanceamento de carga com certeza de envio (CBC) → essa opção é usada quando se quer medir o tempo de execução da simulação paralela do problema dos N-corpos com o uso da biblioteca de balanceamento de carga. Nesse opção, a redesignação de domínios feita no algoritmo de balanceamento faz uma ordenação de partículas e, a partir dessa ordenação, calcula-se o valor l. Com

esse valor calculado, faz-se as modificações nas fronteiras. Nesse caso, sabe-se a quantidade de partículas que um nó com sobrecarga enviará. Um processador sobrecarregado diminui de l a sua fronteira, envia o excesso de corpos e torna-se um nó balanceado. Assim, a redesignação de domínios garante o equilíbrio do número de corpos associados aos processadores sobrecarregados. É importante observar que esse balanceamento não garante que o sistema vai ficar equilibrado ao fim da iteração, pois um processador que estava subcarregado pode, depois de receber o excesso de tarefas de outro processador, tornar-se sobrecarregado.

O projeto de experimentos 2-fatorial com replicações, descrito em [Jai91], é usado, geralmente, para analisar os diferentes fatores considerados no sistema.

O modelo do projeto 2-fatorial utilizado nessa etapa é baseado nos fatores algoritmo de redistribuição de tarefas (A) e o tipo de balanceamento de carga (B). O fator A pode assumir 3 níveis. Já o fator B tem 2 níveis. Nos testes apresentados posteriormente, cada experimento é repetido 10 vezes.

- análise do balanceamento de carga na simulação paralela → nesse caso, um projeto com 1 fator, o balanceamento de carga, é usado para comparar os desempenhos da simulação paralela do problema dos N-corpos antes e após a inclusão da biblioteca de balanceamento desenvolvida. Esse fator pode assumir um dos 2 níveis:

  (a) sem balanceamento de carga (SBC) → essa opção é usada quando se quer medir o tempo de execução da simulação paralela do problema dos N-corpos sem a intervenção da biblioteca de balanceamento de carga;

  (b) a alternativa selecionada no 1º projeto realizado.

  O projeto de experimentos 1-fatorial com replicações, descrito em [Jai91], é usado, geralmente, para comparar as diferentes alternativas de um único fator.

  O modelo do projeto 1-fatorial utilizado nessa etapa é baseado, unicamente, no tipo de balanceamento de carga (B). Esse fator pode assumir 2 níveis. Nos testes apresentados posteriormente, cada experimento é repetido 10 vezes.

  Esse modelo é uma simplificação do modelo do projeto 2-fatorial porque, quando o fator algoritmo de redistribuição do projeto 2-fatorial é desconsiderado, os dois modelos são similares. Devido a essa similaridade, pode-se desenvolver um modelo geral do projeto 2-fatorial e adaptá-lo aos dois tipos de projetos usados nesse capítulo.

Nas subseções A.1 a A.6, o modelo do projeto 2-fatorial será explicado.

A.1 Modelo

Considere um projeto com a níveis do fator A, b níveis do fator B e r replicações do experimento com cada um desses ab níveis. Nesse caso, o modelo é dado por:

$$y_{ijk} = \mu + \alpha_j + \beta_i + \gamma_{ij} + e_{ijk}$$

$i = 1,...,b$

$j = 1,...,a$

$k = 1,...,r$ $\{A.1\}$

onde:

- y_{ijk} é a resposta na k-ésima repetição do experimento com o fator A no nível j e com o fator B no nível i;
- μ é a resposta média do sistema;
- α_j é o efeito do fator A no nível j;
- β_i é o efeito do fator B no nível i;
- γ_{ij} é o efeito da interação entre o fator A no nível j e o fator B no nível i;
- e_{ijk} é o erro experimental obtido na k-ésima replicação do experimento com o fator A no nível j e com o fator B no nível i.

Pela definição do modelo, assume-se que:

- os efeitos são calculados de forma que sua soma seja igual a zero
  $\sum_{j=1}^{a} \alpha_j = 0$
  $\sum_{i=1}^{b} \beta_i = 0$
- as interações entre os fatores são calculados de forma que a soma de suas linhas e colunas seja igual a zero
  $\sum_{i=1}^{b} \gamma_{i1} = \sum_{i=1}^{b} \gamma_{i2} = ... \sum_{i=1}^{b} \gamma_{ia} = 0$
  $\sum_{j=1}^{a} \gamma_{1j} = \sum_{j=1}^{a} \gamma_{2j} = ... \sum_{j=1}^{a} \gamma_{bj} = 0$
- os erros são calculados de forma que sua soma seja igual a zero
  $\sum_{k=1}^{r} \sum_{j=1}^{a} \sum_{i=1}^{b} e_{ijk} = 0$

A.2 Cálculo dos Efeitos

Os valores medidos são dispostos em uma matriz $a \times b$ onde cada dimensão corresponde a um fator considerado. O conjunto de repetições de um dado experimento pode ser visto como ocupando a mesma célula dessa matriz. Dessa forma, cada célula da matriz conterá r replicações do mesmo experimento.

Calculando-se a média dos experimentos de uma célula da matriz, obtém-se que:

$\bar{y}_{ij.} = \mu + \alpha_j + \beta_i + \gamma_{ij}$ $\{A.2\}$

O ponto (.) indica que a média é feita entre todas as repetições da célula.

Seja y_{ijk} o elemento (i,j,k) dessa matriz. Substituindo cada y_{ijk} da matriz pela equação $\{A.1\}$, obtêm-se abr equações. Adicionando as abr equações, tem-se que:

$$\sum_{k=1}^{r}\sum_{j=1}^{a}\sum_{i=1}^{b} y_{ijk} = abr\mu + br\sum_{j=1}^{a}\alpha_j + ar\sum_{i=1}^{b}\beta_i + r\sum_{j=1}^{a}\sum_{i=1}^{b}\gamma_{ij} + \sum_{k=1}^{r}\sum_{j=1}^{a}\sum_{i=1}^{b} e_{ijk} \qquad \{A.3\}$$

Por definição, a soma dos efeitos e da interação entre eles é igual a zero. Se a soma dos erros experimentais em cada coluna é igual a zero, obtém-se:

$$\sum_{k=1}^{r}\sum_{j=1}^{a}\sum_{i=1}^{b} y_{ijk} = abr\mu \qquad \{A.4\}$$

A resposta média do sistema é dada por:

$$\mu = 1/abr\ \sum_{k=1}^{r}\sum_{j=1}^{a}\sum_{i=1}^{b} y_{ijk} \qquad \{A.5\}$$

A fração da equação $\{A.5\}$ é conhecida como média global do sistema e é denotada por $\bar{y}_{...}$. Os 3 pontos (...) indicam que a média é feita em todas as dimensões da matriz. Eles são usados para distinguir a média global da média da coluna, $\bar{y}_{.j.}$, que é obtida fazendo-se a média das respostas pertencentes à coluna j.

A média da coluna j é dada por:

$$\bar{y}_{.j.} = 1/br\ \sum_{k=1}^{r}\sum_{i=1}^{b} y_{ijk} \qquad \{A.6\}$$

Substituindo a equação $\{A.1\}$ na equação $\{A.6\}$, tem-se que:

$$\bar{y}_{.j.} = 1/br\ \sum_{k=1}^{r}\sum_{i=1}^{b}(\mu + \alpha_j + \beta_i + \gamma_{ij} + e_{ijk})$$

$$\bar{y}_{.j.} = 1/br\ (br\mu + br\alpha_j + r\sum_{i=1}^{b}\beta_i + r\sum_{i=1}^{b}\gamma_{ij} + \sum_{k=1}^{r}\sum_{i=1}^{b} e_{ijk}) \qquad \{A.7\}$$

Pela definição do modelo, as somatórias da equação $\{A.7\}$ são iguais a zero. Assim:

$$\bar{y}_{.j.} = \mu + \alpha_j$$

$$\alpha_j = \bar{y}_{.j.} - \mu = \bar{y}_{.j.} - \bar{y}_{...} \qquad \{A.8\}$$

Analogamente, β_i é dado por:

$$\bar{y}_{i..} = \mu + \beta_i$$

$$\beta_i = \bar{y}_{i..} - \mu = \bar{y}_{i..} - \bar{y}_{...} \qquad \{A.9\}$$

Substituindo a equação $\{A.5\}$, $\{A.8\}$ e $\{A.9\}$ em $\{A.3\}$, obtém-se que:

$$\gamma_{ij} = \bar{y}_{ij.} - \bar{y}_{i..} - \bar{y}_{.j.} + \bar{y}_{...} \qquad \{A.10\}$$

A.3 Cálculo dos Erros

A resposta estimada para a alternativa j é dada por:

$$\hat{y}_{ij} = \mu + \alpha_j + \beta_i + \gamma_{ij} = \bar{y}_{ij}$$

A diferença entre a resposta medida e a resposta estimada representa um erro experimental. Calculando-se os erros experimentais para as abr observações, obtém-se que o erro médio deve ser zero porque μ foi calculado assumindo-se que as somas dos efeitos, das interações entre eles e dos erros era igual a zero.

A variância dos erros pode ser estimada pela soma dos quadrados dos erros (SSE):

$$SSE = \sum_{k=1}^{r} \sum_{j=1}^{a} \sum_{i=1}^{b} e_{ijk}^2 \qquad \{A.11\}$$

A.4 Alocação da Variação

A variação total da observação y em um projeto 2-fatorial é devida aos fatores considerados, a interação entre os fatores e aos erros experimentais. Para calculá-la, eleva-se a equação $\{A.1\}$ ao quadrado, para cada observação y:

$$y_{ijk}^2 = \mu^2 + \alpha_j^2 + \beta_i^2 + \gamma_{ij}^2 + e_{ijk}^2 + 2\mu\alpha_j + 2\mu\beta_i + 2\mu\gamma_{ij} + 2\mu e_{ijk} + 2\alpha_j\beta_i + 2\alpha_j\gamma_{ij} + 2\alpha_j e_{ijk} + 2\beta_i\gamma_{ij} + 2\beta_i e_{ijk} + 2\gamma_{ij} e_{ijk}$$

Adicionando os termos correspondentes das abr equações, tem-se que:

$$\sum_{i,j,k} y_{ijk}^2 = \sum_{i,j,k} \mu^2 + \sum_{i,j,k} \alpha_j^2 + \sum_{i,j,k} \beta_i^2 + \sum_{i,j,k} \gamma_{ij}^2 + \sum_{i,j,k} e_{ijk}^2 + \textit{Soma dos produtos} \qquad \{A.12\}$$

Todas as somas dos produtos são iguais a zero devido às seguintes restrições:

(a) a soma dos efeitos é igual a zero;

(a) a soma das interações entre os fatores é igual a zero;

(c) a soma dos erros experimentais é igual a zero.

A equação $\{A.12\}$, expressa em termos das somas dos quadrados, pode ser reescrita como:

$$SSY = SSO + SSA + SSB + SSAB + SSE \qquad \{A.13\}$$

Onde SSY é a soma dos quadrados de y, SSO é a soma dos quadrados da média global, SSA é a soma dos quadrados dos efeitos do fator A, SSB é a soma dos quadrados dos efeitos do fator B e $SSAB$ é a soma dos quadrados dos efeitos da interação entre os fatores. Observa-se que SSO, SSA, SSB e $SSAB$ podem ser reescritos como:

$$\begin{aligned} SSO &= \textstyle\sum_{i,j,k} \mu^2 = abr\mu^2 \\ SSA &= \textstyle\sum_{i,j,k} \alpha_j^2 = br \sum_j \alpha_j^2 \\ SSB &= \textstyle\sum_{i,j,k} \beta_i^2 = ar \sum_i \beta_i^2 \\ SSAB &= \textstyle\sum_{i,j,k} \gamma_{ij}^2 = r \sum_{i,j} \gamma_{ij}^2 \end{aligned} \qquad \{A.14\}$$

A variação total de y (SST) é definida como:

$$SST = \textstyle\sum_{i,j,k}(y_{ijk} - \mu)^2 = \sum_{i,j,k} y_{ijk}^2 \text{ - } abr\mu^2 = SSY - SSO = SSA + SSB + SSAB + SSE \qquad \{A.15\}$$

A variação total pode, portanto, ser dividida em duas partes, SSA, SSB e $SSAB$, e SSE, que representam as partes explicável e inexplicável da variação, respectivamente.

A alta porcentagem da variação explicável indica que o modelo foi empregado de maneira correta. Essa porcentagem é dada pela soma das seguintes percentagens:

% da variação explicada pelo fator A = $100(SSA/SST)$

% da variação explicada pelo fator B = $100(SSB/SST)$

% da variação explicada pela interação AB = $100(SSAB/SST)$

A porcentagem da variação não explicada pelo modelo é dada por:

% da variação não explicada pelo modelo = $100(SSE/SST)$

A.5 Análise da Variância

Na alocação da variação, descrita na subseção A.4, qualquer fator que possui uma alta porcentagem de variação é considerado importante.

Essa importância deveria ser distinguida de significância, que é um termo estatístico. Para determinar se um termo tem um efeito significativo na resposta, compara-se sua contribuição na variação com relação aos erros experimentais do sistema.

O procedimento estatístico que analisa a significância de um ou mais fatores é chamado de análise de variância (ANOVA).

Para entender o ANOVA, considere as soma dos quadrados SSY, SSO, SSA, SSB, $SSAB$ e SSE. Cada uma dessas somas tem um grau de liberdade que corresponde ao número de termos independentes requeridos para calculá-los. Os graus de liberdade para as somas são:

$$
\begin{aligned}
&DFY = abr \\
&DF0 = 1 \\
&DFA = (a-1) \\
&DFB = (b-1) \\
&DFAB = (a-1)(b-1) \\
&DFE = ab(r-1) \qquad \{A.16\}
\end{aligned}
$$

A soma SSY consiste de uma soma de abr termos, e todos eles podem ser escolhidos independentemente. Portanto, SSY tem abr graus de liberdade.

A soma SSO consiste de um único termo, μ^2, que é repetido abr vezes. Uma vez que um valor para μ for escolhido, a soma SSO pode ser calculada. Assim, SSO tem 1 grau de liberdade.

A soma SSA contém a soma de a termos, mas somente $a-1$ desses termos são independentes, visto que a soma de todos os efeitos deve ser igual a zero. Portanto, SSB tem $a-1$ graus de liberdade.

A soma SSB contém a soma de b termos, mas somente $b-1$ desses termos são independentes, visto que a soma de todos os efeitos deve ser igual a zero. Portanto, SSB tem $b-1$ graus de liberdade.

A soma $SSAB$ contém a soma de ab termos, mas somente $(a-1)(b-1)$ desses termos são independentes, visto que a soma dos efeitos de A e dos efeitos de B devem ser iguais a zero. Portanto, $SSAB$ tem $(a-1)(b-1)$ graus de liberdade.

A soma SSE consiste de abr termos dos quais somente $ab(r-1)$ podem ser escolhidos independentemente, já que a soma dos r erros experimentais correspondentes as r replicações de cada experimento deve ser zero. Assim, somente $(r-1)$ erros em cada um dos ab experimentos são independentes.

O F-teste deve ser usado para verificar se SSX é significantemente maior que SSE. Assumindo-se que os erros são normalmente distribuídos, SSX e SSE tem distribuições chi-square. A razão $(SSX/DFX)/(SSE/DFE)$ tem uma distribuição F. Se essa razão calculada for maior que o F[1 - α; DFX, DFE] quantile obtido da tabela de quantiles das F-variáveis (ver, por exemplo, as tabelas A.6, A.7 e A.8 de [Jai91]), SSX é considerado significantemente maior que SSE.

A razão SSX/DFX é chamada de quadrado médio de X (MSX), Analogamente, a razão SSE/DFE é chamada de quadrado médio dos erros (MSE).

A.6 Intervalos de Confiança para os Efeitos

Os valores estimados dos parâmetros do modelo são variáveis randômicas, já que elas estão baseadas em uma amostra. Os resultados obtidos de uma outra amostra poderiam fornecer os parâmetros com valores diferentes. Para minimizar esse problema, a variância dos parâmetros do modelo deve ser estimada a partir dos erros experimentais do sistema.

Os intervalos de confiança para os efeitos devem ser calculados usando valores da Distribuição Normal para $ab(r-1)$ graus de liberdade (que são os graus de liberdade associados com os erros).

É possível fazer estimativas para os níveis (a partir de $\hat{y}_{ij} = \mu + \alpha_j$ ou de $\hat{y}_{ij} = \mu + \beta_i$) e para as combinações entre eles (a partir de $\sum h_k\alpha_k$ ou de $\sum h_k\beta_k$, onde, por definição, $\sum h_k = 0$).

As estimativas para os níveis isolados podem ser usadas para calcular os intervalos de confiança para as respostas médias.

As estimativas para a combinação dos níveis podem ser usadas para verificar se existe uma diferença significante entre duas alternativas.

O intervalo de confiança para um nível X é dado por:

$$Efeito_X \pm tS_X$$

Onde $Efeito_X$ é a resposta média de X, S_X é o desvio padrão de X e t é o valor da Distribuição Normal.

Os desvios padrões são dados por:

S_e = desvio padrão dos erros = $\sqrt{MSE}$

S_{α_j} = desvio padrão de $\alpha_j = S_e\sqrt{(a-1)/abr}$

S_{β_i} = desvio padrão de $\beta_i = S_e\sqrt{(b-1)/abr}$

$S_{\gamma_{ij}}$ = desvio padrão de $\gamma_{ij} = S_e\sqrt{(a-1)(b-1)/abr}$

S_μ = desvio padrão de $\mu = S_e/\sqrt{br}$

Os efeitos em que os intervalos incluem o zero são significativos; os demais são não significativos.

A fim de se comparar níveis, α_l e α_m, pode-se calcular os intervalos de confiança entre eles, usando a fórmula $\sum h_k \alpha_k$. Para comparar α_l e α_m, deve-se seguir os seguintes passos:

- atribuir valores 1, -1 para α_l e α_m, e 0 para os demais, levando em consideração que $\sum h_k = 0$;
- calcular o valor médio de α_l e α_m, que é dado por $y_{.l.} - y_{.m.}$;
- calcular o desvio padrão de α_l e α_m, que é dado por $S_e\sqrt{\sum h_k^2/br}$.

Na comparação dos níveis α_l e α_m, se o intervalo de confiança inclui o zero, α_l não é superior a α_m e vice-versa. Se o intervalo de confiança não inclui o zero, α_l é superior a α_m.

De maneira análoga, para comparar os níveis β_l e β_m, deve-se:

- atribuir valores 1, -1 para β_l e β_m, e 0 para os demais, levando em consideração que $\sum h_k = 0$;
- calcular o valor médio de β_l e β_m, que é dado por $y_{l..} - y_{m..}$;
- calcular o desvio padrão de β_l e β_m, que é dado por $S_e\sqrt{\sum h_k^2/ar}$.

Na comparação dos níveis β_l e β_m, se o intervalo de confiança inclui o zero, β_l não é superior a β_m e vice-versa. Se o intervalo de confiança não inclui o zero, β_l é superior a β_m.

B Análise das Alternativas de Balanceamento

Nessa seção serão analisados os tempos das simulações do problema do N-corpos com 3 algoritmos de balanceamento de carga, gerados a partir da biblioteca de balanceamento, a fim de se descobrir qual dessas simulações foi a mais eficiente.

As tabelas 5, 6 e 7 mostram o tempo de execução da simulação com os três algoritmos de balanceamento de carga implementados, a saber: *random*, *threshold* e *shortest*.

A simulação foi executada variando-se o balanceamento de carga para os seguintes níveis:

- a simulação com a biblioteca de balanceamento sem a garantia de equilíbrio do número de partículas de cada um dos processadores participantes (CBS);
- a simulação com a biblioteca de balanceamento com a garantia de equilíbrio do número de partículas dos nós do sistema, ao final da iteração (CBC).

Nas subseções seguintes serão realizadas as análises para o sistema com 4096 e 16384 corpos.

Partículas	*Balanceamento*	
	CBS	CBC
4096	54145	63796
	50372	62724
	55502	62734
	57293	62759
	53624	61480
	56261	64509
	52960	61250
	58263	62655
	58802	64102
	55198	62099
16384	365677	408124
	355702	407559
	344331	425001
	377706	401598
	361188	408636
	360509	408736
	341438	409079
	339157	407894
	365278	409351
	347464	424015

Tabela 5: Tempos (em ms) de cada uma das 10 execuções da simulação paralela com o algoritmo de balanceamento de carga *random*

Partículas	*Balanceamento*	
	CBS	CBC
4096	54857	61890
	55212	60784
	55947	62596
	54754	61605
	55507	62143
	56000	62029
	54763	61296
	52372	62123
	55125	65343
	57006	63997
16384	369368	414846
	392752	437041
	378279	409581
	381000	405884
	357954	408678
	357644	409825
	370662	418337
	373869	416590
	373769	408300
	354003	409617

Tabela 6: Tempos (em ms) de cada uma das 10 execuções da simulação paralela com o algoritmo de balanceamento de carga *threshold*

Partículas	*Balanceamento*	
	CBS	CBC
4096	52374	59071
	50682	59066
	56388	58037
	54198	57322
	53122	59579
	52851	56409
	52838	59092
	50273	59079
	54158	58672
	54322	56364
16384	352101	423809
	342705	417687
	348132	404408
	342948	418022
	330032	422257
	338566	429812
	345639	422233
	342104	429029
	345361	421760
	350124	411258

Tabela 7: Tempos (em ms) de cada uma das 10 execuções da simulação paralela com o algoritmo de balanceamento de carga *shortest*

B.1 Análise das Alternativas para o Sistema com 4096 Corpos

A média para cada um dos níveis dos fatores é calculada e apresentada na tabela 8. Essas médias são os tempo previstos pelo modelo.

A média global pode ser interpretada como uma indicação de que o tempo de execução de uma simulação com o algoritmo de balanceamento médio e com o tipo de balanceamento médio é 57826,18 ms.

A tabela 8 também apresenta os efeitos causados pelo fator algoritmo e pelo fator balanceamento considerados isoladamente. Os resultados podem ser interpretados da seguinte maneira:

- fator algoritmo $\rightarrow$ os tempos gastos na execução da simulação com o algoritmo *random* e com o algoritmo *threshold* são, respectivamente, 1200,22 e 931,12 ms superior ao tempo gasto na execução da simulação com o algoritmo de redistribuição médio. Já a execução da simulação com o algoritmo *shortest* é 2131,33 ms inferior ao tempo gasto na execução da simulação com o algoritmo de balanceamento médio;
- fator balanceamento $\rightarrow$ os tempos gastos na execução da simulação com balanceamento sem certeza de envio é 3324,32 ms inferior ao tempo gasto na execução da simulação com o balanceamento médio. Já o tempo gasto na execução da simulação com balanceamento com certeza de envio é 3324,32 ms superior ao tempo gasto na execução da simulação com balanceamento médio.

A tabela 9 apresenta os efeitos causados pela interação entre os dois fatores. Essa interação pode ser interpretada como, por exemplo, para o algoritmo *random* com certeza de envio, uma indicação de que o tempo de execução da simulação com o algoritmo *random* com certeza de envio é 460,08 ms superior ao tempo gasto na execução da simulação com o algoritmo médio com certeza de envio, ou 460,08 ms superior ao tempo gasto na execução da simulação com o algoritmo *random* com balanceamento de carga médio.

Na tabela 10 são apresentadas as somas dos quadrados dos fatores, assim como a variação percentual causada por cada fator e pela interação entre eles.

Observa-se que 68% da variação no tempo de execução é devido ao balanceamento utilizado. O fator algoritmo contribui com 14% na variação do tempo de execução da simulação. A interação entre os fatores teve pouca participação na variação dos resultados.

O modelo proposto não é capaz de explicar 16% da variação no tempo de execução da simulação. Esse percentual é relativamente alto, mas pode ser devido ao fato de que os tempos medidos nas 10 replicações de um mesmo experimento apresentaram variações razoáveis. Essas variações podem ser devidas ao fato de que, apesar da exclusividade do poder computacional das máquinas usadas, a simulação foi executada compartilhando a rede de comunicação entre todos os nós do sistema com outras aplicações. Portanto, esses erros podem ser experimentais e não existem indicativos de uma inadequação do modelo para expressar os tempos medidos.

A tabela 11 apresenta o grau de significância de cada efeito considerado na análise.

Conforme discutido na subseção A.5, para um dado nível de significância, α, um fator X é estatisticamente significativo se o F calculado for superior ao valor tabelado de $F[\alpha,$

grau de liberdade de X, grau de liberdade do erro]. Por indisponibilidade de dados, considerou-se o grau de liberdade do erro igual a 500 (superior ao grau dos erros em todos os testes realizados). Supondo-se que 90% é o nível de significância desejado, os valores de F calculados para os efeitos são superiores aos seus respectivos valores tabelados. Portanto, todos os efeitos são estatisticamente significativos.

Antes de se calcular os intervalos de confiança para o fator considerado, é preciso descobrir o valor da Distribuição Normal (t) a ser usado. Para um nível de significância de 90% ($\alpha = 0,9$), o valor de t[1 - (1 - $\alpha/2$), DFE] é 1,697.

A tabela 12 mostra o intervalo de confiança para o fator balanceamento na análise.

Os efeitos em que os intervalos de confiança correspondentes não incluem o zero são significativos, os demais, são não significativos. Dessa forma, observa-se que os fatores e a interação entre eles são significativos.

A fim de comparar os níveis *random*, *threshold* e *shortest* do fator algoritmo, pode-se calcular os intervalos de confiança entre as alternativas, usando a fórmula $\sum h_k\alpha_k$ (onde $\sum h_k = 0$, por definição):

- atribuindo-se para h os valores h_1 = -1, $h_2 = 1$ e $h_3 = 0$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-636.31270, 1174.51270)$;
- atribuindo-se para h os valores h_1 = -1, $h_2 = 0$ e $h_3 = 1$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-4236.96270, -2426.13730)$;
- atribuindo-se para h os valores $h_1 = 0$, h_2 = -1 e $h_3 = 1$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-3967.86270, -2157.03730)$.

Na comparação dos níveis α_l e α_m, se o intervalo de confiança inclui o zero, α_l não é superior a α_m e vice-versa. Caso contrário, α_l é superior a α_m. Observa-se que o nível *shortest* é superior aos demais níveis.

Analogamente, para comparar os níveis CBS e CBC do fator balanceamento, pode-se calcular os intervalos de confiança entre as alternativas, usando a fórmula $\sum h_k\beta_k$ (onde $\sum h_k = 0$, por definição). Atribuindo-se para h os níveis h_1 = -1 e $h_2 = 1$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-6648.63333 \pm 739.26637) = (-7387.89970, -5909.36696)$.

Na comparação dos níveis β_l e β_m, se o intervalo de confiança inclui o zero, β_l não é superior a β_m e vice-versa. Caso contrário, β_l é superior a β_m. Observa-se que a alternativa CBS é superior a alternativa CBC.

Fazendo-se uma análise geral da tabelas 5, 6 e 7, já era possível observar uma superioridade do nível *shortest* em relação aos demais níveis, e uma superioridade da alternativa CBS em relação a alternativa CBC. Com os cálculos dos intervalos de confiança entre as alternativas, foi possível comprovar a superioridades observadas nessa análise.

Considerando-se que o objetivo desse capítulo é estudar como o balanceamento de carga pode melhorar o desempenho da aplicação, a simulação paralela com o algoritmo de balanceamento *shortest* sem certeza de envio será usado para a análise proposta na seção C.

B.2 Análise das Alternativas para o Sistema com 16384 Corpos

A média para cada um dos níveis dos fatores é calculada e apresentada na tabela 13. Essas médias são os tempo previstos pelo modelo.

A média global pode ser interpretada como uma indicação de que o tempo de execução de uma simulação com o algoritmo de balanceamento médio e com o tipo de balanceamento médio é 385910,15 ms.

A tabela 13 também apresenta os efeitos causados pelo fator algoritmo e pelo fator balanceamento considerados isoladamente. Os resultados podem ser interpretados da seguinte maneira:

- fator algoritmo → o tempo gasto na execução da simulação com o algoritmo *threshold* é 6489,8 ms superior ao tempo gasto na execução da simulação com o algoritmo de redistribuição médio. Já as execuções da simulação com o algoritmo *shortest* e com o algoritmo *random* são, respectivamente, 4010,8 e 2479 ms inferior ao tempo gasto na execução da simulação com o algoritmo de balanceamento médio;
- fator balanceamento → os tempos gastos na execução da simulação com balanceamento sem certeza de envio é 29058,41667 ms inferior ao tempo gasto na execução da simulação com o balanceamento médio. Já o tempo gasto na execução da simulação com balanceamento com certeza de envio é 29058,41667 ms superior ao tempo gasto na execução da simulação com balanceamento médio.

A tabela 14 apresenta os efeitos causados pela interação entre os dois fatores. Essa interação pode ser interpretada como, por exemplo, para o algoritmo *random* com certeza de envio, uma indicação de que o tempo de execução da simulação com o algoritmo *random* com certeza de envio é 1481,26667 ms superior ao tempo gasto na execução da simulação com o algoritmo médio com certeza de envio, ou 1481,26667 ms superior ao tempo gasto na execução da simulação com o algoritmo *random* com balanceamento de carga médio.

Na tabela 15 são apresentadas as somas dos quadrados dos fatores, assim como a variação percentual causada por cada fator e pela interação entre eles.

Observa-se que 86% da variação no tempo de execução é devido ao balanceamento utilizado. O fator algoritmo e a interação entre os fatores tiveram pouca participação na variação dos resultados.

O modelo proposto não é capaz de explicar 12% da variação no tempo de execução da simulação. Esse percentual é relativamente alto, mas pode ser devido ao fato de que os tempos medidos nas 10 replicações de um mesmo experimento apresentaram variações razoáveis. Como já mencionado na subseção B.1, essas variações podem ser devidas ao fato de que, apesar da exclusividade do poder computacional das máquinas usadas, a simulação foi executada compartilhando a rede de comunicação entre todos os nós do sistema com outras aplicações. Portanto, esses erros podem ser experimentais e não existem indicativos de uma inadequação do modelo para expressar os tempos medidos.

A tabela 16 apresenta o grau de significância de cada efeito considerado na análise.

Conforme discutido na subseção A.5, para um dado nível de significância, α, um fator X é estatisticamente significativo se o F calculado for superior ao valor tabelado de $F[\alpha$,

grau de liberdade de X, grau de liberdade do erro]. Por indisponibilidade de dados, considerou-se o grau de liberdade do erro igual a 500 (superior ao grau dos erros em todos os testes realizados). Supondo-se que 90% é o nível de significância desejado, os valores de F calculados para os fatores algoritmo e balanceamento são superiores aos seus respectivos valores tabelados. Portanto, os fatores são estatisticamente significativos. Já a interação entre os fatores não é estatisticamente significativa porque o seu valor de F calculado é inferior ao respectivo valor de F tabelado.

Antes de se calcular os intervalos de confiança para o fator considerado, é preciso descobrir o valor da Distribuição Normal (t) a ser usado. Para um nível de significância de 90% ($\alpha = 0,9$), o valor de t[1 - (1 - $\alpha/2$), DFE] é 1,697.

A tabela 17 mostra o intervalo de confiança para o fator balanceamento na análise.

Os efeitos em que os intervalos de confiança correspondentes não incluem o zero são significativos, os demais, são não significativos. Observa-se que a alternativa *random* do fator algoritmo é não significativa e os demais níveis desse fator são significativos. Já fator balanceamento é significativo para todas as suas alternativas.

A fim de comparar os níveis *random*, *threshold* e *shortest* do fator algoritmo, pode-se calcular os intervalos de confiança entre as alternativas, usando a fórmula $\sum h_k\alpha_k$ (onde $\sum h_k = 0$, por definição):

- atribuindo-se para h os valores h_1 = -1, $h_2 = 1$ e $h_3 = 0$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-15211.159, -2726.441)$;
- atribuindo-se para h os valores h_1 = -1, $h_2 = 0$ e $h_3 = 1$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-7774.159, 4710.559)$;
- atribuindo-se para h os valores $h_1 = 0$, h_2 = -1 e $h_3 = 1$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-16742.959, -4258.241)$.

Na comparação dos níveis α_l e α_m, se o intervalo de confiança inclui o zero, α_l não é superior a α_m e vice-versa. Caso contrário, α_l é superior a α_m. Observa-se que os níveis *shortest* e *random* são superiores ao nível *threshold*.

Analogamente, para comparar os níveis CBS e CBC do fator balanceamento, pode-se calcular os intervalos de confiança entre as alternativas, usando a fórmula $\sum h_k\beta_k$ (onde $\sum h_k = 0$, por definição). Atribuindo-se para h os níveis h_1 = -1 e $h_2 = 1$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-58116.8333 \pm 5096.86478) = (-63213.698, -53019.969)$.

Na comparação dos níveis β_l e β_m, se o intervalo de confiança inclui o zero, β_l não é superior a β_m e vice-versa. Caso contrário, β_l é superior a β_m. Observa-se que a alternativa CBS é superior a alternativa CBC.

Fazendo-se uma análise geral da tabelas 5, 6 e 7, já era possível observar as superioridades dos níveis *shortest* e *random* em relação ao nível *threshold*, e uma superioridade da alternativa CBS em relação a alternativa CBC. Com os cálculos dos intervalos de confiança entre as alternativas, foi possível comprovar a superioridades observadas nessa análise.

Considerando-se que o objetivo desse capítulo é estudar como o balanceamento de carga pode melhorar o desempenho da aplicação, a simulação paralela com o algoritmo de balanceamento *shortest* sem certeza de envio será usado para a análise proposta na seção C.

É necessário enfatizar que o algoritmo *shortest* sem certeza de envio foi escolhido, em lugar do *random* sem certeza de envio, porque, além dele ter sido superior na subseção B.1, esse algoritmo obteve melhor desempenho que o *random* no nível CBS (ver coluna CBS das tabelas 5 e 7).

Balanceamento	*random*	*threshold*	*shortest*	*Média por Linha*	*Efeito Linha*
CBS	55242	55143	53120.6	54501.86667	-3324.31667
CBC	62810.8	62371.6	58269.1	61150.5	3324.31667
Média por Coluna	59026.4	58757.3	55694.85	57286.18333	
Efeito Coluna	1200.2166	931.1166	-2131.3333		

Tabela 8: Média das repetições para cada combinação dos níveis

Balanceamento	*random*	*threshold*	*shortest*
CBS	-460.08333	-289.98333	750.06667
CBC	460.08333	289.98333	-750.06667

Tabela 9: Efeito da interação entre os fatores nos tempos medidos

Fator	*Soma dos Quadrados*	*Variação Percentual (%)*
	SST = 970951290	
	SSY = 2.01603 10^{11}	
	SS0 = 2.0063205 10^{11}	
Algoritmo	SSA = 137001601.4	14.11
Balanceamento	SSB = 663064879.3	68.29
Alg/Bal	SSAB = 17167340.23	1.77
Erros	SSE = 153717469.1	15.83

Tabela 10: Soma dos quadrados e variação percentual para os efeitos considerados

Componente	*Graus de Liberdade*	*Quadrado Médio*	*F Calculado*	*F Tabelado*
Algoritmo	2	68500800.7	24.064	2.31
Balanceamento	1	663064879.3	239.931	2.72
Alg/Bal	2	8583670.115	3.015	2.31
Erros	54	2846619.798		

Tabela 11: Valores para a análise de variância para o projeto de testes propostos, onde $S_e^2 = 2846619,798$

Fator	*Desvio Padrão*	*Intervalo de Confiança*
μ	(57826.18333 ± 369.633)	(57456.55033, 58195.81633)
Algoritmo		
random	(1200.21667 ± 522.740)	(677.47667, 1722.95667)
threshold	(931.11667 ± 522.740)	(408.37667, 1453.85667)
shortest	(-2131.33333 ± 522.740)	(-2654.07333, -1608.59333)
Balanceamento		
CBS	(-3324.31667 ± 369.633)	(-3693.94967, -2954.68367)
CBC	(3324.31667 ± 369.633)	(2954.68367, 3693.94967)

Tabela 12: Intervalos de confiança para os fatores

Balanceamento	*random*	*threshold*	*shortest*	*Média por Linha*	*Efeito Linha*
CBS	355854	370930	343771.2	356851.73333	-29058.41667
CBC	411008.3	413869.9	420027.5	414968.56667	29058.41667
Média por Coluna	383431.15	392399.95	381899.35	385910.15	
Efeito Coluna	-2479	6489.8	-4010.8		

Tabela 13: Média das repetições para cada combinação dos níveis

Balanceamento	*random*	*threshold*	*shortest*
CBS	1481.26667	7588.46667	-9069.73333
CBC	-1481.26667	-7588.46667	9069.73333

Tabela 14: Efeito da interação entre os fatores nos tempos medidos

Fator	*Soma dos Quadrados*	*Variação Percentual (%)*
	SST = 5.9541368 10^{10}	
	SSY = 8.99514 10^{12}	
	SS0 = 8.9355986 10^{12}	
Algoritmo	SSA = 1286989234	2.16
Balanceamento	SSB = 5.0663495 10^{10}	85.90
Alg/Bal	SSAB = 284078080.1	0.47
Erros	SSE = 7306805686	12.27

Tabela 15: Soma dos quadrados e variação percentual para os efeitos considerados

Componente	*Graus de Liberdade*	*Quadrado Médio*	*F Calculado*	*F Tabelado*
Algoritmo	2	643494617	4.756	2.31
Balanceamento	1	5.0663495 10^{10}	374.422	2.72
Alg/Bal	2	142039040.1	1.050	2.31
Erros	54	135311216.4		

Tabela 16: Valores para a análise de variância para o projeto de testes propostos, onde $S_e^2 = 135311216,4$

Fator	*Desvio Padrão*	*Intervalo de Confiança*
μ	(385910.15 ± 2548.432)	(383361.718, 388458.582)
Algoritmo		
random	(-2479 ± 3604.028)	(-6083.028, 1125.028)
threshold	(6489.8 ± 3604.028)	(2885.772, 10093.828)
shortest	(-4010.8 ± 3604.028)	(-7614.828, -406.772)
Balanceamento		
CBS	(-29058.41667 ± 2548.432)	(-31606.849, -26509.985)
CBC	(29058.41667 ± 2548.432)	(26509.985, 31606.849)

Tabela 17: Intervalos de confiança para os fatores

C Análise do Balanceamento de Carga na Simulação Paralela

Nessa seção serão analisados os tempos de execução da simulação paralela do problema dos N-corpos antes e após a inclusão da biblioteca de balanceamento, levando-se em consideração, apenas, o tipo de balanceamento de carga utilizado no sistema.

A simulação foi executada variando-se o balanceamento de carga para os seguintes níveis:

- a simulação paralela sem balanceamento de carga (SBC);
- a simulação com o algoritmo de balanceamento de carga *shortest* sem a garantia de equilíbrio do número de partículas de cada um dos processadores participantes (CBS).

Conforme discutido na seção A, é importante enfatizar que o projeto com 1 fator, proposto nessa seção, é uma simplificação do modelo descrito nas seções A.1 a A.6, e, portanto, não há a necessidade de se desenvolver uma nova modelagem para esse tipo de projeto.

Nas subseções seguintes serão feitas as análises para o sistema com 4096 e 16384 corpos.

C.1 Análise do Balanceamento para o Sistema com 4096 Corpos

A média para cada um dos níveis é calculada e apresentada na tabela 18. Essas médias são os tempos previstos pelo modelo.

A média global pode ser interpretada como uma indicação de que o tempo de execução de uma simulação com um balanceamento de carga médio é 64877,85 ms.

A tabela 19 apresenta os efeitos causados pelo fator balanceamento.

Os resultados podem ser interpretados da seguinte maneira:

- o tempo gasto na execução da simulação sem o balanceamento de carga é 11757,25 ms superior ao tempo gasto na simulação com um balanceamento médio;
- o tempo gasto na execução da simulação com balanceamento de carga sem certeza de envio é 11757,25 ms inferior ao tempo gasto na simulação com um balanceamento médio.

Na tabela 20 são exibidas as somas dos quadrados do fator, assim como a variação percentual causada por ele.

Observa-se que 97% da variação no tempo de execução é devida ao balanceamento de carga utilizado.

O modelo proposto não é capaz de explicar 3% da variação nos tempos de execução. Esse percentual pode ser devido ao fato de que os tempos medidos nas 10 replicações de um mesmo experimento apresentaram variações razoáveis. Como já mencionado na subseção B.1, essas variações podem ser devidas ao fato de que, apesar da exclusividade do poder computacional das máquinas usadas, a simulação foi executada compartilhando a rede de comunicação entre todos os nós do sistema com outras aplicações. Portanto, esses erros podem ser experimentais e não existem indicativos de uma inadequação do modelo para expressar os tempos medidos.

A tabela 21 apresenta o grau de significância do fator considerado na análise.

Conforme discutido na subseção A.5, para um dado nível de significância, α, um fator X é estatisticamente significativo se o F calculado for superior ao valor tabelado de $F[\alpha$, grau de liberdade de X, grau de liberdade do erro]. Supondo-se que 90% é o nível de significância desejado, o valor de F calculado para o fator balanceamento é superior ao seu respectivo valor tabelado. Portanto, esse efeito é estatisticamente significativo.

Antes de se calcular os intervalos de confiança para o fator considerado, é preciso descobrir o valor da Distribuição Normal (t) a ser usado. Para um nível de significância de 90% ($\alpha = 0,9$), o valor de t[1 - (1 - $\alpha/2$), DFE] é 1,734.

A tabela 22 mostra o intervalo de confiança para o fator balanceamento na análise.

Os efeitos em que os intervalos de confiança correspondentes não incluem o zero são significativos, os demais, são não significativos. Dessa forma, observa-se que o fator balanceamento é significativo.

A fim de comparar os níveis SBC e CBS do fator balanceamento, pode-se calcular os intervalos de confiança entre as alternativas, usando a fórmula $\sum h_k\beta_k$ (onde $\sum h_k = 0$, por definição). Atribuindo-se para h os valores h_1 = -1 e $h_2 = 1$, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-23514.5 \pm 11834.8469) = (-35349.3469, -11679.6531)$.

Na comparação dos níveis β_l e β_m, se o intervalo de confiança inclui o zero, β_l não é superior a β_m e vice-versa. Caso contrário, β_l é superior a β_m. Observa-se que o nível CBS é superior ao SBC.

Fazendo-se uma análise geral da tabela 7, já era possível observar uma superioridade do nível CBS em relação ao nível SBC. Com os cálculos dos intervalos de confiança entre as alternativas, foi possível comprovar a superioridade observada nessa análise.

C.2 Análise do Balanceamento para o Sistema com 16384 Corpos

A média para cada um dos níveis é calculada e apresentada na tabela 23. Essas médias são os tempos previstos pelo modelo.

A média global pode ser interpretada como uma indicação de que o tempo de execução de uma simulação com um balanceamento de carga médio é 412600,85 ms.

A tabela 24 apresenta os efeitos causados pelo fator balanceamento.

Os resultados podem ser interpretados da seguinte maneira:

- o tempo gasto na execução da simulação sem o balanceamento de carga é 68829,65 ms superior ao tempo gasto na simulação com um balanceamento médio;
- o tempo gasto na execução da simulação com balanceamento de carga sem certeza de envio é 68829,65 ms inferior ao tempo gasto na simulação com um balanceamento médio.

Na tabela 25 são exibidas as somas dos quadrados do fator, assim como a variação percentual causada por ele.

Observa-se que 98% da variação no tempo de execução é devida ao balanceamento de carga utilizado.

O modelo proposto não é capaz de explicar 2% da variação nos tempos de execução. Esse percentual pode ser devido ao fato de que os tempos medidos nas 10 replicações de um mesmo experimento apresentaram variações razoáveis. Como já mencionado na subseção B.1, essas variações podem ser devidas ao fato de que, apesar da exclusividade do poder computacional das máquinas usadas, a simulação foi executada compartilhando a rede de comunicação entre todos os nós do sistema com outras aplicações. Portanto, esses erros podem ser experimentais e não existem indicativos de uma inadequação do modelo para expressar os tempos medidos.

A tabela 26 apresenta o grau de significância do fator considerado na análise.

Conforme discutido na subseção A.5, para um dado nível de significância, α, um fator X é estatisticamente significativo se o F calculado for superior ao valor tabelado de $F[\alpha$, grau de liberdade de X, grau de liberdade do erro]. Supondo-se que 90% é o nível de significância desejado, o valor de F calculado para o fator balanceamento é superior ao seu respectivo valor tabelado. Portanto, esse efeito é estatisticamente significativo.

Antes de se calcular os intervalos de confiança para o fator considerado, é preciso descobrir o valor da Distribuição Normal (t) a ser usado. Para um nível de significância de 90% ($\alpha = 0,9$), o valor de t[1 - (1 - $\alpha/2$), DFE] é 1,734.

A tabela 27 mostra o intervalo de confiança para o fator balanceamento na análise.

Os efeitos em que os intervalos de confiança correspondentes não incluem o zero são significativos, os demais, são não significativos. Dessa forma, observa-se que o fator balanceamento é significativo.

A fim de comparar os níveis SBC e CBS do fator balanceamento, pode-se calcular os intervalos de confiança entre as alternativas, usando a fórmula $\sum h_k\beta_k$ (onde $\sum h_k = 0$, por definição). Atribuindo-se para h os valores h_1 = -1 e h_2 = 1, tem-se que, para 90% de nível de significância, o intervalo de confiança é $(-137659.3 \pm 53885.31788) = (-191544.61788, -83773.98212)$.

Na comparação dos níveis β_l e β_m, se o intervalo de confiança inclui o zero, β_l não é superior a β_m e vice-versa. Caso contrário, β_l é superior a β_m. Observa-se que o nível CBS é superior ao nível SBC.

Fazendo-se uma análise geral da tabela 7, já era possível observar uma superioridade do nível CBS em relação ao nível SBC. Com os cálculos dos intervalos de confiança entre as alternativas, foi possível comprovar a superioridade observada nessa análise.

SBC	CBS	*Média Global*
76635.1	53120.6	64877.85

Tabela 18: Média das repetições para cada combinação dos níveis

SBC	CBS
11757.25	-11757.25

Tabela 19: Efeito do fator balanceamento nos tempos medidos

Fator	*Soma dos Quadrados*	*Variação Percentual (%)*
	SST = 2847644204.55	
	SSY = 87030352617	
	SS0 = 84182708412.45	
Balanceamento	SSB = 2764658551.25	97.086
Erros	SSE = 82985654.3	2.914

Tabela 20: Soma dos quadrados e variação percentual para o fator considerado

Componente	*Graus de Liberdade*	*Quadrado Médio*	*F Calculado*	*F Tabelado*
Balanceamento	1	2764658551.25	599.6682	3.01
Erros	18	4610314.1278		

Tabela 21: Valores para a análise de variância para o projeto de testes propostos, onde $S_e^2 = 4610314,1278$

Fator	*Desvio Padrão*	*Intervalo de Confiança*
μ	(64877.85 ± 832.52895)	(64045.32105, 65710.37895)
Balanceamento		
SBC	(11757.25 ± 832.52895)	(10924.72105, 12589.77895)
CBS	(-11757.25 ± 832.52895)	(-12589.77895, 10924.72105)

Tabela 22: Intervalos de confiança para o fator balanceamento

SBC	CBS	*Média Global*
481430.5	343771.2	412600.85

Tabela 23: Média das repetições para cada combinação dos níveis

SBC	CBS
68829.65	-68829.65

Tabela 24: Efeito do fator balanceamento nos tempos medidos

Fator	*Soma dos Quadrados*	*Variação Percentual (%)*
	SST = 96470771585.5	
	SSY = $3.50126\ 10^{12}$	
	SS0 = 3404789228414	
Balanceamento	SSB = 94750414382.45	98.217
Erros	SSE = 1720357203.05	1.783

Tabela 25: Soma dos quadrados e variação percentual para o fator considerado

Componente	*Graus de Liberdade*	*Quadrado Médio*	*F Calculado*	*F Tabelado*
Balanceamento	1	94750414382.45	991.3682	3.01
Erros	18	95575400.16944		

Tabela 26: Valores para a análise de variância para o projeto de testes propostos, onde $S_e^2 = 95575400,16944$

Fator	*Desvio Padrão*	*Intervalo de Confiança*
μ	(421600.85 ± 3810.2674)	(417790.5826, 425411.1174)
Balanceamento		
SBC	(68829.65 ± 3810.2674)	(65019.3826, 72639.9174)
CBS	(-68829.65 ± 3810.2674)	(-72639.9174, -65019.3826)

Tabela 27: Intervalos de confiança para o fator balanceamento

D Considerações Finais

Nas subseções B.1 e B.2, foram feitas análises dos tempos de execução da simulação paralela do problema dos N-corpos com os algoritmos de balanceamento de carga *random*, *threshold* e *shortest*. Essas análises revelam que a simulação com o algoritmo *shortest* obteve melhor desempenho que as simulações com os algoritmos *random* e *threshold*. Esses resultados são coerentes com as conclusões de [ELZ86], considerando-se que a superioridade do algoritmo *shortest* pode ser explicada pela simplicidade do sistema testado. Como somente 4 processadores participaram do sistema, ficou mais fácil escolher, em cada iteração que esse algoritmo era invocado, os processadores que estavam menos sobrecarregados. Essa facilidade na escolha garantiu uma boa redistribuição das tarefas da aplicação e, portanto, diminuiu os tempos do algoritmo em relação aos demais.

Os valores obtidos na execução da simulação paralela com os algoritmos de balanceamento *random*, *difusão* e *redistribuição completa*, discutidos em [FG96], também estão em concordância com o que foi concluído em [ELZ86]. Nessas execuções, prova-se que, para 16 processadores, o algoritmo *random* é 12% mais rápido que o algoritmo de *difusão*, e 30% mais rápido que o algoritmo de *redistribuição completa*. Para um sistema pequeno, 2 ou 4 processadores participantes, as simulações com os algoritmos *random* e *difusão* têm, praticamente, o mesmo tempo de execução. Conforme discutido no parágrafo anterior, a simulação com o algoritmo de balanceamento *difusão* só obtém o mesmo tempo de execução do algoritmo *random* porque um processador sobrecarregado descobre, mais facilmente, a porcentagem de tarefas a serem recebidas por cada um de seus vizinhos.

Nas subseções B.1 e B.2, também foi observado que a simulação paralela com balanceamento de carga sem certeza de envio obteve melhor desempenho que a simulação paralela com balanceamento de carga com certeza de envio. Essa constatação pode ser explicada da seguinte maneira: quando ocorre um desequilíbrio nas tarefas dos processadores participantes da simulação paralela com balanceamento de carga com certeza de envio, os processadores sobrecarregados ordenam seus corpos e, depois dessa ordenação, diminuem suas regiões de um valor calculado a partir da ordenação e enviam uma quantidade do excesso de seus corpos para os processadores subcarregados (que aumentam suas regiões do referido valor para poder receber esses excessos). É interessante observar que, além da ordenação ser uma operação dispendiosa, esse balanceamento não garante que, ao final da iteração, os processadores do sistema fiquem com o número de tarefas equilibrado, pois um processador que estava subcarregado pode, depois de receber o excesso de tarefas de outro processador, tornar-se sobrecarregado. Na simulação com balanceamento de carga sem certeza de envio, os processadores sobrecarregados diminuem suas regiões de um valor constante e transferem os corpos que não pertencem mais a eles para os processadores sub-

carregados (que aumentam suas regiões dessa mesma constante para poder receber esses corpos). Da mesma forma que a simulação com certeza de envio, esse balanceamento não garante que o sistema vai ficar equilibrado ao fim da iteração, pois um processador que estava subcarregado pode, depois de receber o excesso de tarefas de outro processador, tornar-se sobrecarregado. Entretanto, nesse balanceamento não se gasta tempo na fase de ordenação de corpos.

Nas subseções C.1 e C.2 foram realizadas análises que revelam que a simulação paralela com o algoritmo de balanceamento de carga *shortest* sem certeza de envio obteve o melhor desempenho em relação à simulação paralela pura. Essa conclusão pode ser explicada da seguinte maneira: quando ocorre um desequilíbrio nas tarefas dos processadores participantes da aplicação paralela original, os processadores que estão subcarregados devem ficar esperando, na fase de sincronização, pelos processadores sobrecarregados, em todas as iterações restantes da simulação, aumentando, assim, o tempo de execução do sistema. Na simulação com balanceamento de carga, tenta-se manter um número de corpos equilibrado entre todos os processadores desse sistema, a fim de minimizar os tempos gastos, desnecessariamente, na fase de sincronização dos processadores e, diminuindo, portanto, os tempos gastos nas execuções das simulações com balanceamento.