3
Similaridade e tamanho da seqüência de consulta no BLAST

Quando se planeja construir aplicativos que utilizam um agrupamento de computadores no intuito de paralelizar ou distribuir processamento, se faz importante analisar o tempo que cada máquina permanece trabalhando. Do contrário, a utilização do agrupamento pode não contribuir efetivamente. Por exemplo, algumas máquinas podem se manter ociosas em grande parte do tempo, enquanto outras estão sobrecarregadas na tentativa de terminar o processamento.

No trabalho de dissertação aqui proposto, uma das estratégias de melhorias é exatamente manter equivalente o processamento entre as máquinas, desta forma, toda a execução paralela poderá ser o mais balanceado possível. No caso da ferramenta BLAST, além dos fatores de desequilíbrio típicos a toda estratégia paralela, como processos externos e diferentes configurações de *hardware*, é importante analisarmos os fatores de desvio específicos à execução do BLAST. Consideramos o termo desvio (*skew*) como sendo a alteração, seja de tempo ou processamento, de uma ou algumas máquinas em relação às outras.

O objetivo deste Capítulo é discutir as principais características e os motivos que levam aos desvios de desempenho que podem alterar o balanceamento de carga durante a execução paralela da ferramenta BLAST. Em particular, também é feito um estudo comparando a influência dos tamanhos das seqüências e os graus de similaridade existentes entre seqüências, no que diz respeito ao tempo total de execução. Espera-se, assim, justificar adequadamente as decisões tomadas quanto às estratégias de execução paralela com distribuição da carga que serão propostas neste trabalho de dissertação.

3.1 Fatores de desbalanceamento do BLAST

As estratégias de paralelismo em geral devem levar em consideração que nem sempre serão aplicadas em ambientes computacionais dedicados ou exclusivos, e em específico, além dos fatores existentes na grande maioria de abordagens paralelas, existem outros que geram desbalanceamento e são intrínsecos à comparação de seqüências utilizando a ferramenta BLAST.

Durante a execução do algoritmo BLAST existem algumas fases que se destacam, como: geração de palavras, busca de combinações das palavras na base de dados e extensões destas palavras. Juntas estas fases são responsáveis por 90% de todo o funcionamento do programa (32). Além disto, o BLAST ainda mascara as seqüências, acessa toda a base de dados, calcula análises estatísticas e ordena as seqüências com maior similaridade. Para maiores detalhes de funcionamento do BLAST, veja o livro (19).

Na fase de geração de palavras e combinações com a base de dados, as seqüências são quebradas em pequenas palavras e combinadas com as seqüências do banco. Seqüências de consulta com tamanhos diferentes geram distintos números de palavras, por exemplo, uma seqüência com 100 caracteres, gera 98 palavras de 3 caracteres cada. Enquanto uma seqüência com 500 caracteres, gera 497 palavras de 4 caracteres. Quanto maior for o número de palavras, maior será o número de combinações (*matchings*) perfeitas na primeira fase e, conseqüentemente, maiores as tentativas de extensões na fase posterior.

Na fase de extensão as combinações são processadas para se obter regiões de alta paridade. Uma extensão ocorre até o ponto em que a pontuação de similaridade não é inferior a um dado valor limite. Sendo assim, se espera que quanto mais similar a seqüência de consulta for com as seqüências da base de dados, maior será o custo das extensões.

Desta forma, percebemos que a ferramenta BLAST possui dois fatores implícitos que podem gerar desbalanceamento: número de palavras que combinam perfeitamente com a base de dados e custo de extensão destas palavras. Uma seqüência pode ter maior comprimento ou grau de similaridade em relação às outras seqüências, forçando um desvio de processamento.

Como há interesse em atender ambientes de larga-escala, considera-se

importante a análise destes impactos que podem gerar desequilíbrios de carga durante o processamento entre várias máquinas. Assim sendo, no restante deste Capítulo nos empenharemos em analisar e comparar a diferença de tempo exigido na execução da ferramenta BLAST entre distintas seqüências, variando o grau de similaridade, o tamanho e o tipo de seqüências.

Nosso principal objetivo é comparar o desvio causado entre similaridade e tamanho, e onde estes desvios têm maior impacto, sejam em seqüências de nucleotídeos ou aminoácidos.

3.2 Trabalhos relacionados à similaridade

Alguns dos trabalhos que abordam o balanceamento de carga da ferramenta BLAST em ambiente paralelos, não assumem os desvios obtidos com as diferentes seqüências de processamento. Justamente por considerarem mais relevantes os fatores gerais de desbalanceamento, como: processos externos e diferentes configurações de *hardware*. Isto justifica as poucas publicações na literatura que abordam o tema. Contudo, mostraremos durante este capítulo as conseqüências dos desvios intrínsecos comentados.

Em (6), é feita uma análise do desempenho da ferramenta BLAST utilizando computadores multiprocessados, com memória compartilhada. Entre as análises apresentadas, o trabalho mostra a sensibilidade que o processamento BLAST tem ao variar tamanho e conteúdo de diferentes seqüências. O trabalho também apresenta uma curva linear de crescimento à medida que o tamanho da seqüência de consulta aumenta. Contudo não são mostrados resultados obtidos quando há alteração na similaridade entre as seqüências de consulta, e qual o impacto no processamento quando utilizamos seqüências mais similares.

Já em (12) os autores comentam (sem maiores explicações) sobre o desvio de processamento não ser influenciado somente pelo tamanho da seqüência de consulta, mas também pela diferença de similaridade. Para evitar o desbalanceamento, sugere-se que blocos de seqüências sejam enviados às máquinas que disponham da mesma base de dados.

Utilizando uma abordagem em bases puramente fragmentadas, o trabalho (21) argumenta que o desvio de tempo gerado pela diferença de similaridade entre as seqüências pode se anular, já que no mesmo grupo de seqüências enviadas a uma máquina, algumas exigirão mais tempo de proces-

samento do que outras.

Em contrapartida, a pesquisa apresentada em (9) evidencia o desvio obtido com a diferença de similaridade entre as consultas. Para isso a mesma base de dados é replicada entre diversas máquinas e em seguida são enviadas para cada máquina uma mesma quantidade de caracteres em seqüências de consulta, mas com composição distinta. Com este teste foi possível notar o desbalanceamento obtido, justificando o desvio causado pela similaridade.

3.3 Testes experimentais

Não é nosso objetivo neste Capítulo comprovar que existe um desvio de tempo originado pela diferença de similaridade, pois isso já foi feito pelos trabalhos listados. Da mesma forma com relação ao tamanho da seqüência de consulta. Nosso escopo se estende à análise e quantificação do desvio de tempo causado pela similaridade, em relação ao tamanho da seqüência de consulta e quando estes desvios acontecem. Para tal, identifica-se alguns experimentos que podem embasar nossa argumentação.

3.3.1 Métodos

Para nossos testes utilizamos dois ambientes. No primeiro uma única máquina é utilizada para avaliarmos o tempo de execução quando diferentes seqüências de consulta são comparadas a uma mesma base de dados. O ambiente para testes aqui considerado consiste de uma máquina Pentium 4 com um único processador de 3GHz, um espaço de disco de 80 GB e uma memória RAM de 1GB. Fizemos uso da versão 6 do Fedora Linux (*kernel* 2.6.11), e utilizamos a versão 2.2.16 da implementação NCBI BLAST, que em todas as execuções mantivemos os parâmetros padrões. No intuito de evitar qualquer influência de processamento por processos externo, mantivemos a máquina em modo exclusivo(*single user*) e desligamos todos os serviços desnecessários e consumidores de recursos do sistema operacional.

Acreditamos que observar a execução entre as várias seqüências de consulta individualmente, e em uma única máquina, irá permitir uma análise do desbalanceamento causado pela similaridade e tamanho em ambientes

paralelos.

No segundo ambiente de teste, utilizamos um agrupamento de computadores (detalhado no Apêndice 1) de máquinas homogêneas e sem processamento concorrente, a fim de mostrar o efetivo desbalanceamento causado em ambiente paralelo.

Com o intuito de comparar a possível diferença de desvio no tempo de processamento, utilizamos tanto o aplicativo BLASTN (comparação de nucleotídeos contra dados de nucleotídeos) como BLASTP (comparação de aminoácidos contra dados de aminoácidos), pertencentes à família de programas BLAST(mais detalhes em (19)).

Nossos testes consideram principalmente a diferença nas seqüências de consulta, variando o tamanho e a similaridade (composição da seqüência). Quanto a este último item, consideramos seqüências similares quando são copiadas (trechos ou toda a seqüência) da própria base de dados e, de forma oposta, as não similares, são obtidas a partir de um programa de geração de seqüências aleatórias. Para este último caso, desenvolvemos um programa que gera uma seqüência de tamanho determinado por parâmetro, com todos os caracteres gerados a partir de funções aleatórias.

Com o propósito de obter resultados confiáveis, cada execução foi repetida pelo menos quatro vezes e, dependendo do teste realizado, após cada execução o *cache* era limpo, a fim de que a execução não se beneficiasse por dados já disponíveis em memória. Notamos que devido os procedimentos adotados, a diferença de tempo das repetições de uma mesma execução eram mínimas.

3.3.2 Resultados e Análises

Em nosso primeiro teste variamos o tamanho das seqüências de consulta entre 100 a 10.000 caracteres, utilizando dois grupos de seqüências: similares e aleatórias. Estas execuções foram feitas tanto para o BLASTN quando para o BLASTP, e as tomadas de tempo para cada programa podem ser vistas nas Figuras 3.1 e 3.2, respectivamente. Para todas as execuções (usando BLASTN e BLASTP), utilizamos duas base de dados de 700 MB (menor que o tamanho da memória RAM de 1 GB), uma de nucleotídeo e outra de aminoácido.

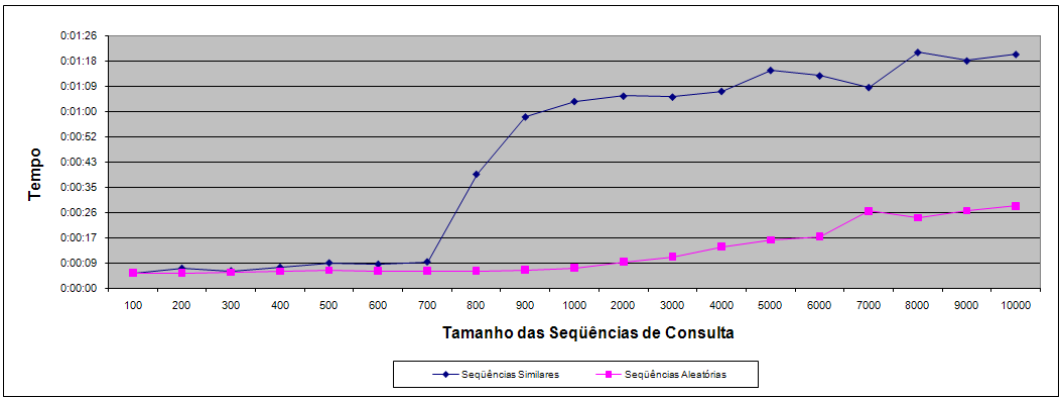

Figura 3.1: *Execução do BLASTN para seqüências aleatórias e similares utilizando uma base de dados menor que a memória RAM.*

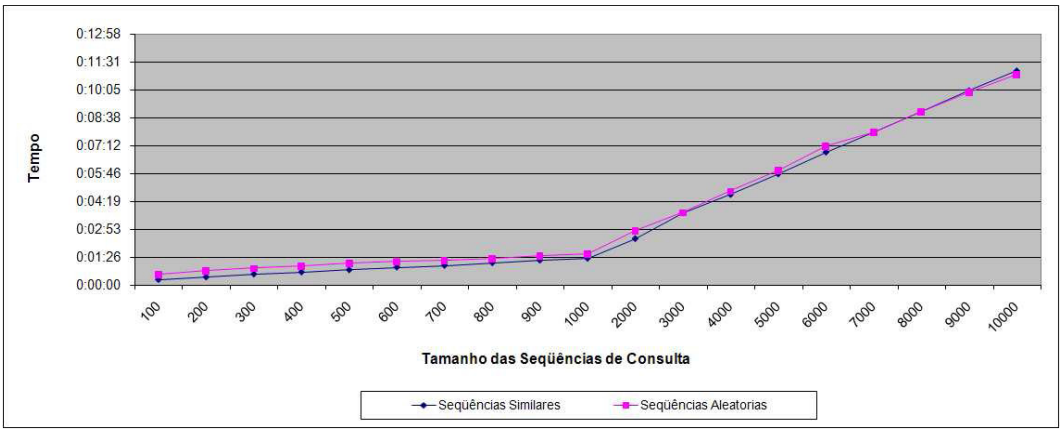

Figura 3.2: *Execução do BLASTP para seqüências aleatórias e similares utilizando uma base de dados menor que a memória RAM.*

De acordo com a Figura 3.1, se percebe que existe pouca variação de tempo quando as seqüências variam o tamanho entre 300 e 900 caracteres. Diferentemente do fator de similaridade, pois segundo o gráfico se mostra mais sensível.

Na Figura 3.1 fica claro o impacto entre similaridade e tamanho. Por exemplo, uma seqüência de 800 bases de nucleotídeo pode gastar mais tempo do que uma seqüência de 10.000 bases com pouca ou nenhuma similaridade.

Ao compararmos as Figuras 3.1 e 3.2, percebemos que os efeitos da similaridade estão muito mais associados à execução do BLASTN do que utilizando o BLASTP. Este último, por outro lado, se mostrou muito mais sensível ao tamanho, inclusive mostrando uma curva com coeficiente angular muito maior, a partir da seqüência de tamanho 1.000. Com estas figuras,

podemos concluir que o fator de desvio que poderá intensificar o desbalanceamento de carga vai depender do programa a ser utilizado. Os motivos destas ocorrências serão vistos mais adiante.

3.3.3 Reduzindo a memória disponível

Em nosso segundo teste aplicamos a mesma estratégia na variação das seqüências de consulta (um grupo de seqüências similares variando o tamanho de 100 à 10.000 caracteres, e outro grupo com seqüências geradas aleatoriamente) para os programas BLASTP e BLASTN. Contudo, diminuímos o tamanho da memória RAM para 512 MB e aumentamos o tamanho da base de dados para 2 GB (tanto para nucleotídeo como para aminoácido). Este teste tem o objetivo de verificar se o custo de similaridade e tamanho estão associados ao acesso de disco rígido. Como pode ser observado nas Figuras 3.3 e 3.4, execuções para BLASTN e BLASTP, respectivamente, o desvio de tempo ocorreu da mesma forma que executando o BLAST com uma base de dados menor que a memória primária.

Na Figura 3.3, o custo para uma seqüência com maior similaridade será sempre igual ou superior, em muitos casos até mesmo em relação a uma seqüência de tamanho bem maior.

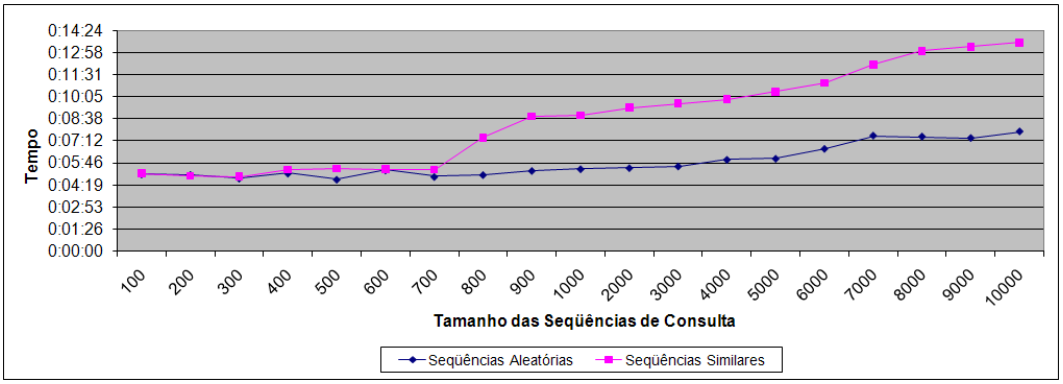

Figura 3.3: *Execução do BLASTN para seqüências aleatórias e similares utilizando uma base de dados maior que a memória RAM.*

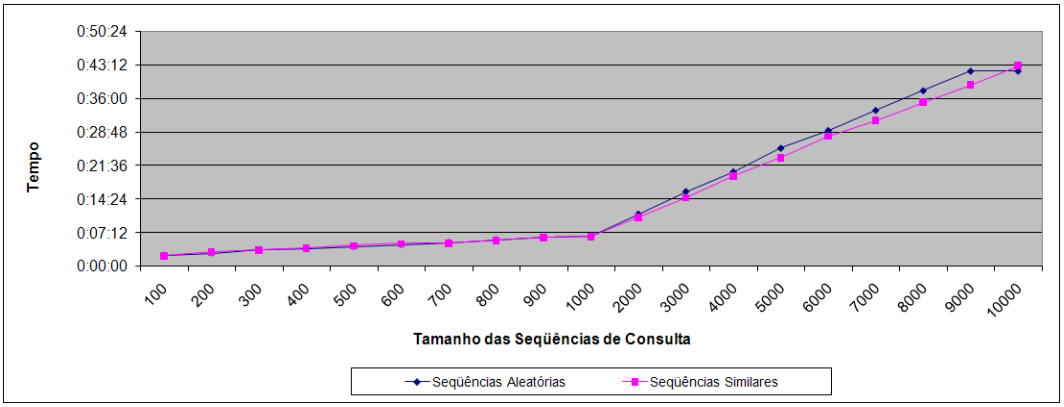

Figura 3.4: *Execução do BLASTP para seqüências aleatórias e similares utilizando uma base de dados maior que a memória RAM.*

3.3.4 Diferença entre BLASTP e BLASTN

Na Figura 3.4 percebemos que a similaridade praticamente não provocou desvio de tempo em relação ao tamanho das seqüências quando utilizado o BLASTP. O mesmo ocorreu no teste da Figura 3.2 utilizando uma base de dados menor que a memória RAM. Situação completamente adversa quando utilizamos o programa BLASTN. O principal motivo para isso é o tamanho das palavras utilizadas em cada programa. No BLASTN uma palavra possui 11 caracteres, enquanto o BLASTP possui somente 3. Logo, este último tem maior probabilidade de combinações e extensões durante a execução do algoritmo BLAST, resultando assim em uma similar quantidade de tempo ne comparação, seja para seqüências similares ou aleátorias.

Por essa razão, se tratando de nucleotídeo, seqüências similares irão gerar maiores números de palavras com combinações, em inglês *word hits*, em relação às seqüências aleatórias e, conseqüentemente, o BLASTN irá consumir mais tempo para análise e extensões. Em seqüências menos similares, o número de *word hits* e o tempo necessário para análise e extensão é muito menor.

Para proteínas a diferença no número de *word hits* é muito menor entre seqüências similares e aleatórias, pois como a definição do tamanho de uma palavra é de somente 3 caracteres, o número final de *word hits* pouco será influênciado pela similaridade da seqüência de consulta.

No caso do programa BLASTP, existe um outra fase de processamento que pode aumentar o número de *word hits*, que é o acréscimo das palavras

conhecidas por vizinhança (*neighbourhood*). Estas são palavras obtidas quando se altera uma letra da *word hit* e, mesmo assim, a pontuação para a combinação permanece alta (19). Esta nova palavra é incluída ao grupo de palavras inicialmente utilizadas. Desta forma, o número total de palavras utilizadas para combinações e extensões na base de dados será o total de *word hits* e palavras de vizinhança obtidas. Em nossos testes percebemos que a inclusão das palavras *neighbourhood* não influenciou o processamento entre as seqüências similares e aleatórias.

A visualização desta situação nos programas BLASTP e BLASTN é facilitada ao analisarmos o número de extensões obtidas com sucesso para cada consulta. Este valor é recolhido do rodapé do relatório de saída do BLAST e pode ser visualizado nas Figuras 3.5 e 3.6 para as mesmas seqüências utilizadas nos testes anteriores.

A Figura 3.5 apresenta as extensões obtidas com sucesso para seqüências similares e para as geradas aleatoriamente. A diferença no número de extensões entre seqüências aleatórias e similares é extremamente grande. Já para proteínas quase não há discrepância no número de extensões obtidas.

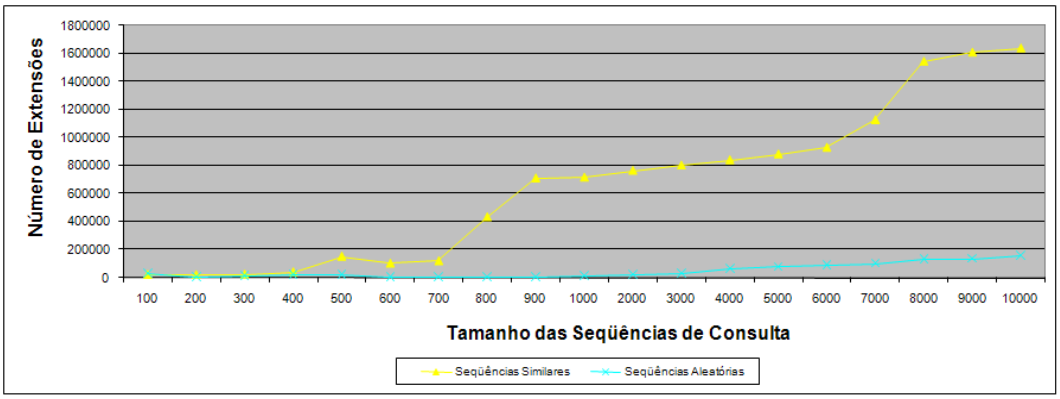

Figura 3.5: *Número de extensões obtidas com sucesso pelo BLASTN para seqüências aleatórias e seqüências similares.*

Alguns testes de avaliação paralela

A fim de avaliar a conseqüência do desvio de similaridade em execuções paralelas propomos um terceiro teste. Neste analisamos o desbalanceamento obtido em processamento paralelo quando utilizadas seqüências similares e aleatórias com tamanhos iguais. Para isso fizemos uso do agrupamento com

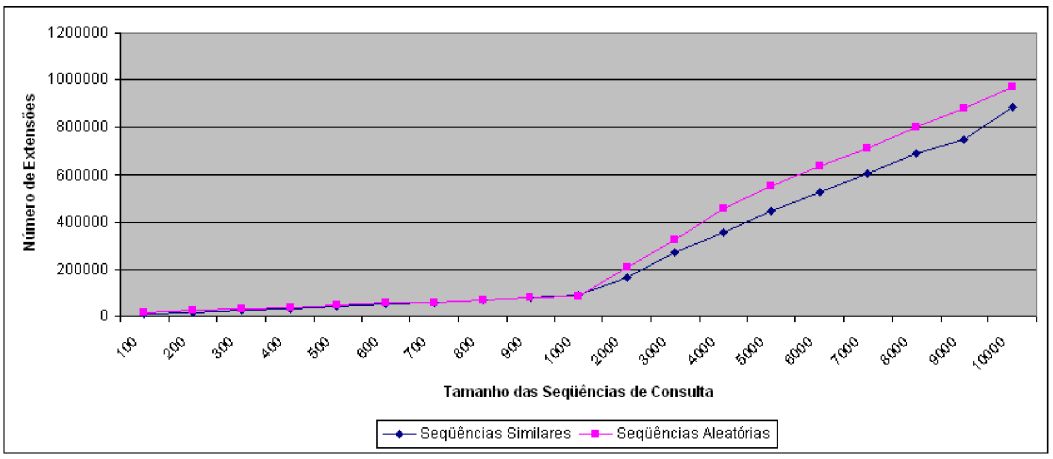

Figura 3.6: *Número de extensões obtidas com sucesso pelo BLASTP para seqüências aleatórias e seqüências similares.*

8 máquinas, e utilizamos a estratégia Fragmentada alocando um fragmento da base de dados em cada máquina. Utilizamos esta estratégia paralela pois, dentre as comentadas aqui, ela é a única que não busca o balanceamento de carga, possibilitando que ao final do processamento possamos verificar o desbalanceamento ocorrido.

Nas Figuras 3.8 e 3.7 é possível observar os desvios obtidos em ambientes paralelos. Na Figura 3.8 o desequilíbrio é proporcionalmente muito maior que na Figura 3.7, e nos permite verificar os efeitos do desvio originados pela similaridade. Segundo estas duas figuras, os desvios comentados anteriormente podem degradar consideravelmente o ambiente de processamento paralelo, justificando assim, as preocupações quanto ao desbalanceamento de carga.

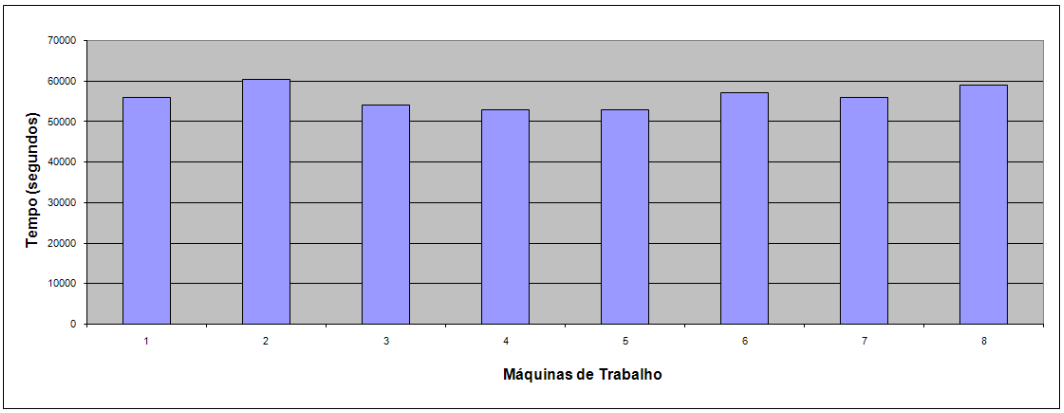

Figura 3.7: *Execução Paralela para estratégia Fragmenta usando seqüências de consulta aleatórias.*

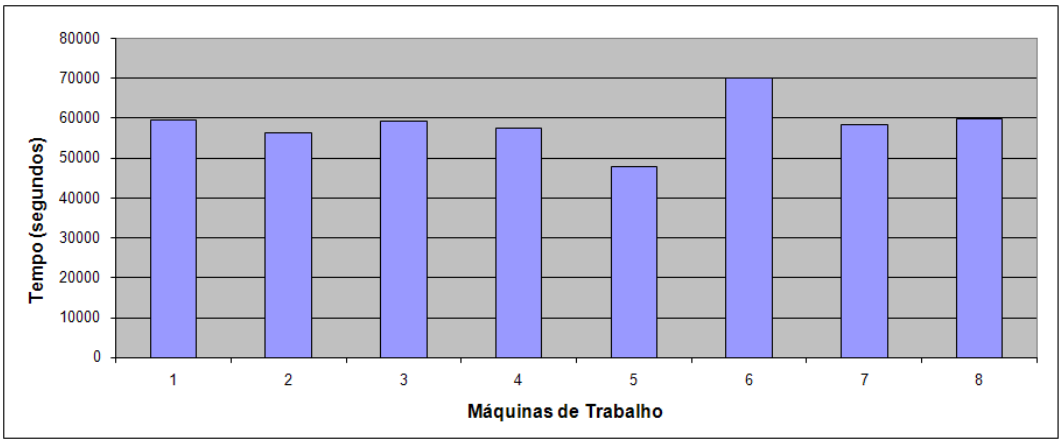

Figura 3.8: *Execução Paralela para estratégia Fragmenta usando seqüências de consulta similares.*

3.4 Conclusão

A proposta para este Capítulo foi analisar o desvio de tempo entre similaridade e tamanho das seqüências de consulta quando comparados a uma mesma base de dados ou distintos fragmentos em um agrupamento de computadores. Para nossos testes utilizamos seqüências ditas similares, retiradas da própria base de dados, e seqüências aleatórias, geradas automaticamente.

Com os testes executados foi possível observar a diferença no desvio de tempo entre os programas BLASTP e BLASTN. A começar pelo tamanho da seqüência que se mostrou um fator de desequilíbrio maior quando utilizado o BLASTP. Os testes para este programa retratam que, quanto maior for a seqüência de consulta, maior será o tempo de execução. Além disso, a influência do tamanho para o desvio de processamento se manteve muito maior do que a influência da similaridade.

Já quando utilizado o BLASTN se percebeu uma sensibilidade maior referente à similaridade das seqüências de consulta. Para os testes utilizando este programa observamos que seqüências menores, porém mais similares em relação à base de dados, podem gastar bem mais tempo de processamento do que seqüências menos similares de tamanho maior.

A principal justificativa encontrada para que os fatores de desvios sejam distintos quando utilizado os dois programas, BLASTN e BLASTP, é que o tamanho da palavra de busca em cada um irá alterar o custo de execução da fase de buscar palavras na base de dados e estender as combinações encon-

tradas.

Também verificamos os impactos ocorridos quando estes desvios de tempo ocorrem em um agrupamento utilizando a estratégia fragmentada. Embora cada fragmento tenha tamanho semelhante, a sua composição é diferente, potencializando os impactos gerados pelo desvio originado pela similaridade. Como resultado, percebemos um aumento considerável no nível de desbalanceamento entre as estações de trabalho.

As conclusões aqui obtidas servem de embasamento para fortalecer nossas estratégias de processamento paralelo. Os fatores de desvio apresentados justificam as opções em tratar o desbalanceamento de forma dinâmica e sob demanda. Pois, embora o desvio de gerado pela diferença de tamanho das seqüências de consulta possa ser tratado por algum pré-processamento em balanceamentos estáticos, o mesmo não ocorre ao desvio de similaridade. Neste último caso o desvio só poderá ser identificado durante a execução.