PROCESSAMENTO PARALELO DE BIOSSEQÜÊNCIAS

Aluno: Guilherme Lemos Mazie
Orientador: Sérgio Lifschitz

Introdução

Durante os 12 meses de pesquisa foram realizados diversos pequenos projetos além da pesquisa principal sobre o processamento paralelo de biosseqüências. Foram feitas implementações de diferentes estratégias para processamento paralelo. A partir de tais execuções, foram obtidos relatórios contendo informações sobre a similaridade de um dado conjunto de seqüências de entrada com um banco de dados, ou melhor, uma base de seqüências biológicas. Para o processamento dos dados foi utilizado um cluster de 32 máquinas do Departamento de Informática da PUC-Rio.

Utilizando o paralelismo, consegue-se um considerável ganho de tempo na comparação das seqüências e, por isso, estratégias dessa natureza são criadas com crescente freqüência. Este método se utiliza do fato de que um conjunto de computadores conectados a um servidor de controle processa comparações de forma mais eficiente do que um único computador, mesmo que muito mais moderno do que os do cluster (conjunto de computadores).

Foram implementadas e testadas cinco estratégias: mpiBLAST [2], Replicada [1], Fragmentada [1], Sob Demanda [1] e Corretiva [1], sendo as quatro últimas desenvolvidas no Laboratório de BioInformática da PUC-Rio.

A estratégia Replicada consiste em replicar todo o banco de dados em todas as máquinas do cluster, de forma que todo o conteúdo da base encontra-se em cada nó. Em seguida, cada seqüência do arquivo de entrada é enviado para um nó. Quando um nó termina a sua comparação, ele recebe outra seqüência.

A estratégia Fragmentada Pura consiste em dividir o banco de dados em tamanhos relativamente iguais e distribuir cada fragmento para um nó do cluster. Em seguida, a seqüência de entrada é comparada com cada fragmento em cada nó separadamente e as respostas são uniformizadas pelo servidor.

A estratégia Fragmentada Corretiva consiste numa estratégia mais complexa. Nela, o banco é fragmentado e cada nó recebe os fragmentos mas trabalha com um fragmento principal. A seqüência de entrada é enviada a todas as máquinas do cluster e a comparação é feita com cada fragmento principal. Caso um nó se torne ocioso, ou seja, tenha acabado todas as suas comparações com o fragmento principal, ele passa a trabalhar com um fragmento secundário, reduzindo o trabalho de uma outra máquina que ainda esteja em processamento.

A estratégia Fragmentada Sob Demanda, assim como a Corretiva, recebe a base de dados fragmentada e cada nó possui um fragmento principal. As seqüências do arquivo de entrada são enviadas uma a uma para cada nó e comparadas com seu fragmento principal, gerando uma resposta que é enviada para o servidor. Caso uma máquina se torne ociosa, ela recebe uma seqüência ainda não trabalhada por um nó em operação e realiza a comparação com um fragmento secundário correspondente ao principal da máquina ainda em operação.

Além dessa tarefa principal, foi desenvolvida uma ferramenta de fragmentação e formatação de base de dados em C, de forma que seja o mais portátil possível.

Tais resultados, relatórios e ferramentas estão todos disponíveis no site [3] do Laboratório de Bioinformática, atualizado e renovado durante a Iniciação Científica.

Novas Estratégias de Processamento

É grande o volume de dados provenientes dos projetos da áreas de bioinformática e biologia molecular. Diante de tantos dados, se torna imprescindível que ferramentas

computacionais agilizem o processo de análise na procura de informações que possam ser relevantes aos biólogos, como pode-se ver na figura 2.1.

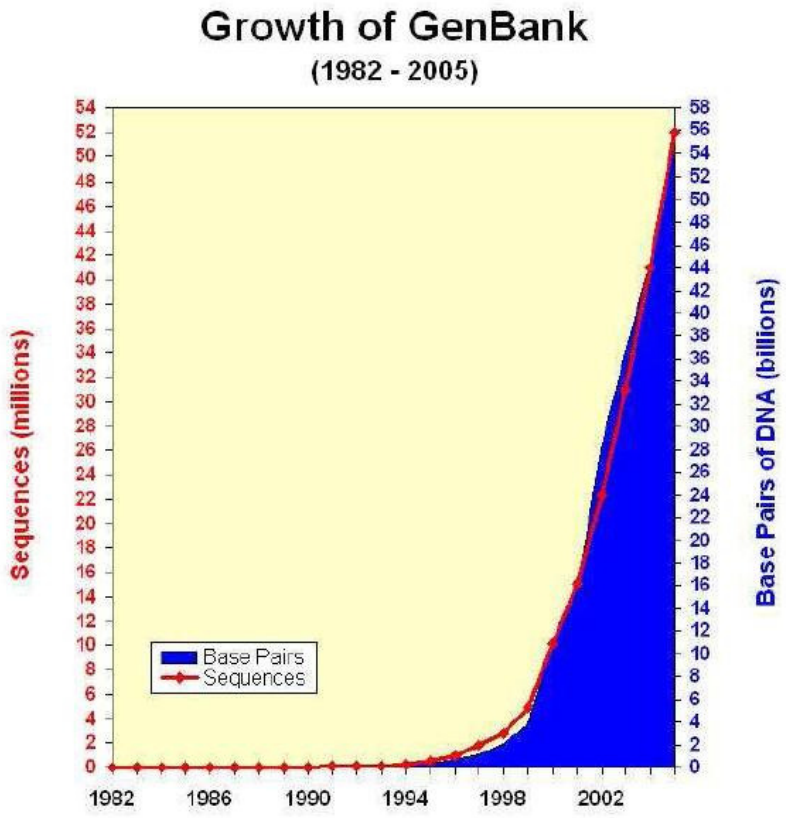

Figura 2.1: Crescimento do volume de dados genéticos.

Por isso, foram desenvolvidos no Laboratório de Bioinformática da PUC-Rio novos métodos de processamento paralelo de biosseqüências, que receberam os as seguintes denominações: Replicada, Fragmentada Pura, Sob Demanda e Fragmentada Corretiva [1].

Tais estratégias foram implementadas com o objetivo de tirar maior proveito da avançada capacidade de processamento paralelo onde, como base, tinha-se somente a ferramenta mpiBLAST [2].

Durante esse projeto de Iniciação Científica, essas estratégias já previamente implementadas foram otimizadas e, sobretudo, testadas, comparando-as com outras ferramentas desenvolvidas.

A. Estratégia Replicada

Se tratando de estratégias em que a base de dados se encontra replicada, uma solução considerada de fácil implementação é, a partir de uma lista de seqüências de consulta, dividi-la para que cada estação de trabalho processe parte desta lista. Contudo, este tipo de abordagem pode gerar desbalanceamento de carga se não for feito um tratamento adequado da forma de dividir a lista de seqüências de consulta.

Por isso, as estratégias utilizadas no Laboratório de Bioinformática se utilizou de um algoritmo de fragmentação dos bancos de dados, ordenando todas as seqüências de consulta

usando o algoritmo circular ("Round-robin") ao enviar uma seqüência para cada máquina. Esta estratégia favorece o balanceamento, considerando que cada máquina receberá a mesma quantidade de dados para processamento. Contudo, outros fatores de desbalanceamento como o nível de similaridade entre as seqüências e a base de dados, a capacidade de processamento de cada máquina e o número de processamentos externos não são tratados por esta estratégia.

Ao analisarmos o processo de execução em ambientes paralelos da ferramenta BLAST utilizando bases de dados replicadas, percebe-se que qualquer seqüência de consulta pode ser transferida para qualquer máquina arbitrária e o processamento irá ocorrer normalmente. Isso facilita o controle no balanceamento de carga, favorecendo a reutilização dos recursos disponíveis. A estratégia Replicada consiste em replicar todo o banco de dados em todas as máquinas do cluster, de forma que todo o conteúdo da base encontra-se em cada nó. Em seguida, cada seqüência do arquivo de entrada é enviado para um nó. Quando um nó termina a sua comparação, ele recebe outra seqüência. Esse método é relativamente pouco eficiente pois, apesar da troca de informações entre o servidor e os nós ser pequena, o custo de pré-processamento é muito alto pois o tempo que se leva para replicar a base é grande e, além disso, demanda alto custo de acesso ao disco ao comparar uma seqüência com a base de dados completa, pois em muitos dos casos são maiores que a memória primária.

B. Estratégia Fragmentada

Nessa estratégia, todo o banco de seqüências é distribuído de forma que esteja replicado entre as diversas máquinas de forma fragmentada. Ou seja, uma base de dados copiada em cada estação de trabalho, mas composta de pedaços com diferentes conteúdo e similar tamanho. Denomina-se esta alocação de replicada com fragmentos primários, ilustrada na Figura 2.2. O termo primário caracteriza um subgrupo de fragmentos alocados em cada estação, que serão prioritariamente utilizados. Cada estação contém distintos fragmentos primários e os fragmentos nãao primários são denominados secundários.

Assim, cada estação de trabalho é responsável, inicialmente, por somente uma parte da base de dados. Uma importante característica dessa alocação do banco é que, embora uma máquina tenha disponível todos os fragmentos da base de dados (primários e secundários), no início do processamento somente serão utilizados os fragmentos primários. Na medida em que for necessário, a fim de manter o sistema balanceado, os fragmentos secundários poderão ser utilizados.

C. Fragmentada Corretiva

Na Estratégia Corretiva, o processamento é dividido em tarefas de acordo com o número de estações de trabalho, de forma com que cada estação receba uma tarefa. Quando uma máquina terminar o processamento da tarefa que lhe foi designada, pode ainda existir máquinas processando outras tarefas, então a máquina ociosa poderá contribuir recebendo o pedaço de uma tarefa que se prolonga em outra máquina. O objetivo aqui é corrigir a alocação inicial de tarefas, envolvendo todas as tarefas para completar o BLAST, feita no início do processamento. O primeiro passo da Estratégia Corretiva é dividir a tarefa a fim de que cada pedaço seja enviado para uma estação de trabalho.

Após uma estação de trabalho ter processado a tarefa recebida, se tornando ociosa, a máquina gerente toma conhecimento disto e aciona o módulo de realocação. Esse é responsável por verificar qual é a estação mais lenta e qual a quantidade da tarefa deve ser realocada. Podem existir duas formas de realocar uma tarefa. Na primeira, o módulo de realocação decide somente que algumas seqüências serão processadas por um único fragmento na máquina ociosa. Na segunda forma, um fragmento inteiro será processado por todas as seqüências na máquina ociosa. Cabe lembrar que no caso dessa estratégia e por conta da alocação da base de dados replicada com fragmentos primários, não existe o tráfego de

seqüências ou fragmentos no processo de realocação, pois esses já estão disponíveis em todas as máquinas. Existem apenas mensagens de metadados sendo enviados, informando e definindo a tarefa a ser executada. A fim de facilitar o entendimento desta estratégia, a Figura 2.3 mostra o seu funcionamento e procedimentos adotados.

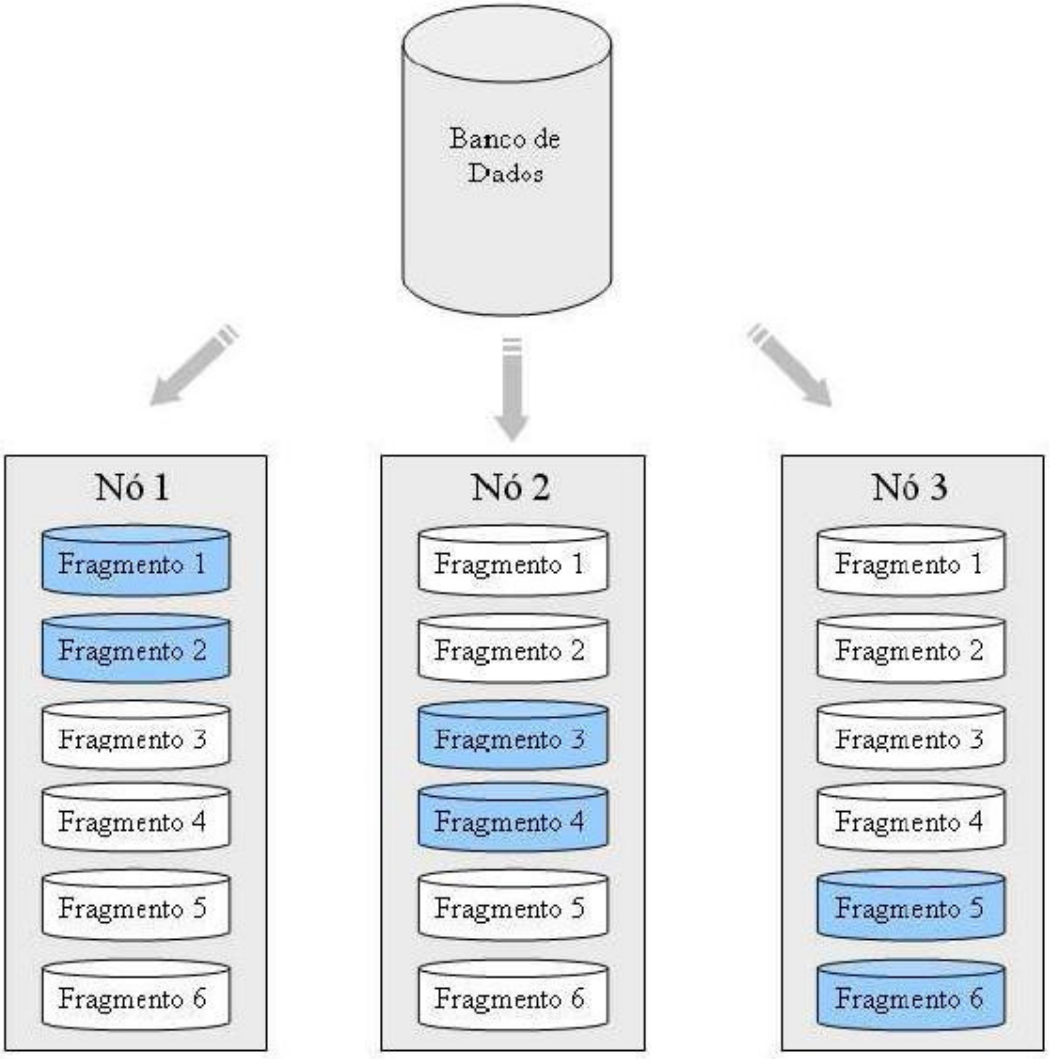

Figura 2.2: Exemplo da alocação da base de dados com replicação total com fragmentos para três máquinas. Em cada máquina de trabalho o banco inteiro está fragmentado em distintas partições e, em destaque, estão os fragmentos primários para cada estação.

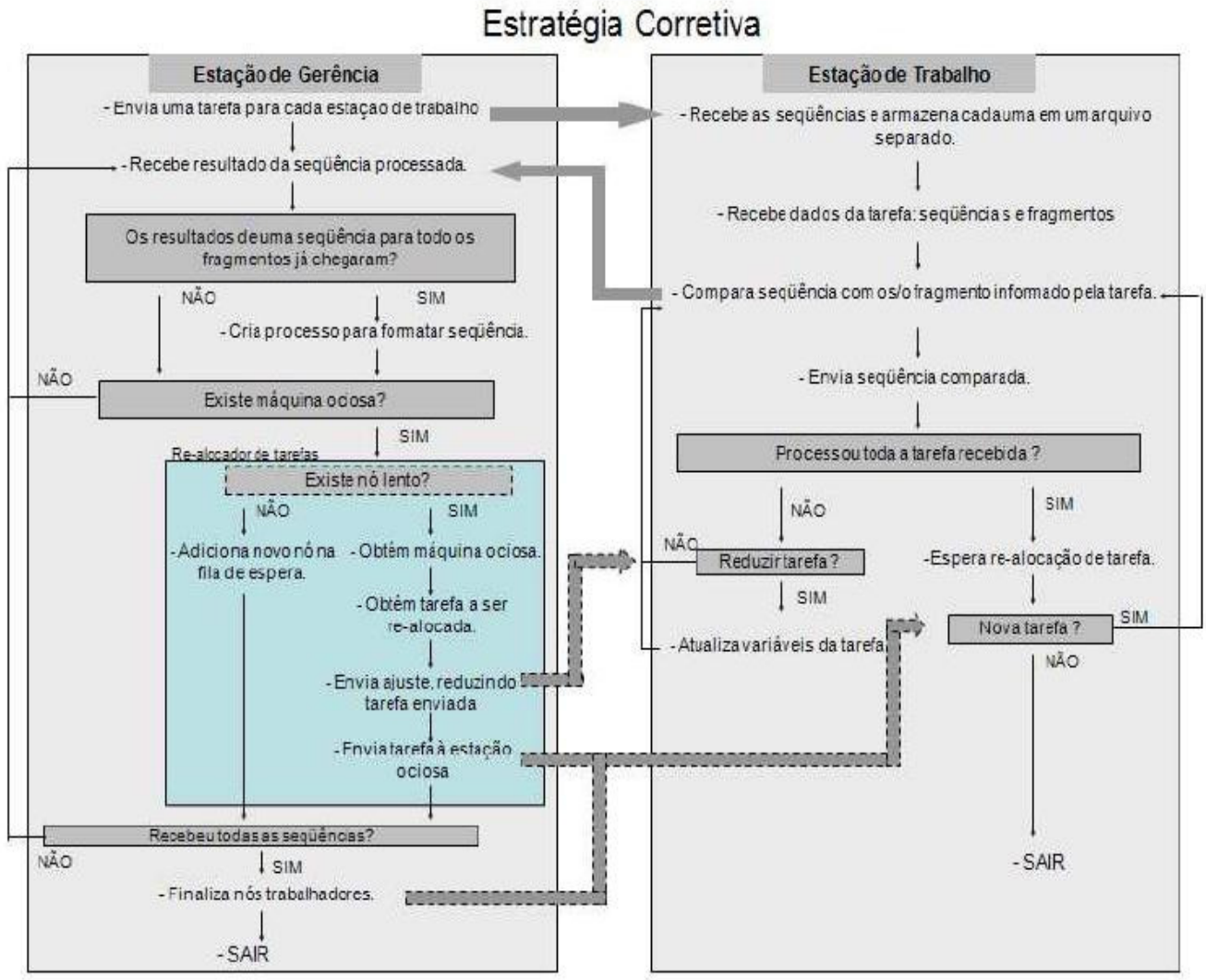

Figura 2.3: Fluxograma de funcionamento da estratégia Corretiva.

D. Estratégia Sob Demanda

Considerando que a base de dados esteja alocada de forma replicada com fragmentos primários, o objetivo da Estratégia Sob Demanda é enviar tarefas para serem processadas na medida em que uma estação de trabalho se encontre ociosa, buscando assim maior reutilização de recursos disponíveis. Tanto para seqüências como para fragmentos, existe um intervalo entre seqüências e fragmentos de uma dada tarefa. Após uma máquina executar uma tarefa recebida, o resultado será enviado à estação de gerência e a máquina estará ociosa aguardando outra tarefa. O processamento permanecerá enviando tarefas para as estações ociosas até que não tenha mais tarefas a serem executadas. Ao observar essa estratégia, percebemos que uma tarefa é dividida em várias outras que serão executadas paralelamente. Para facilitar o entendimento, a Figura 2.4 contém o fluxograma mostrando o funcionamento da estratégia sob demanda.

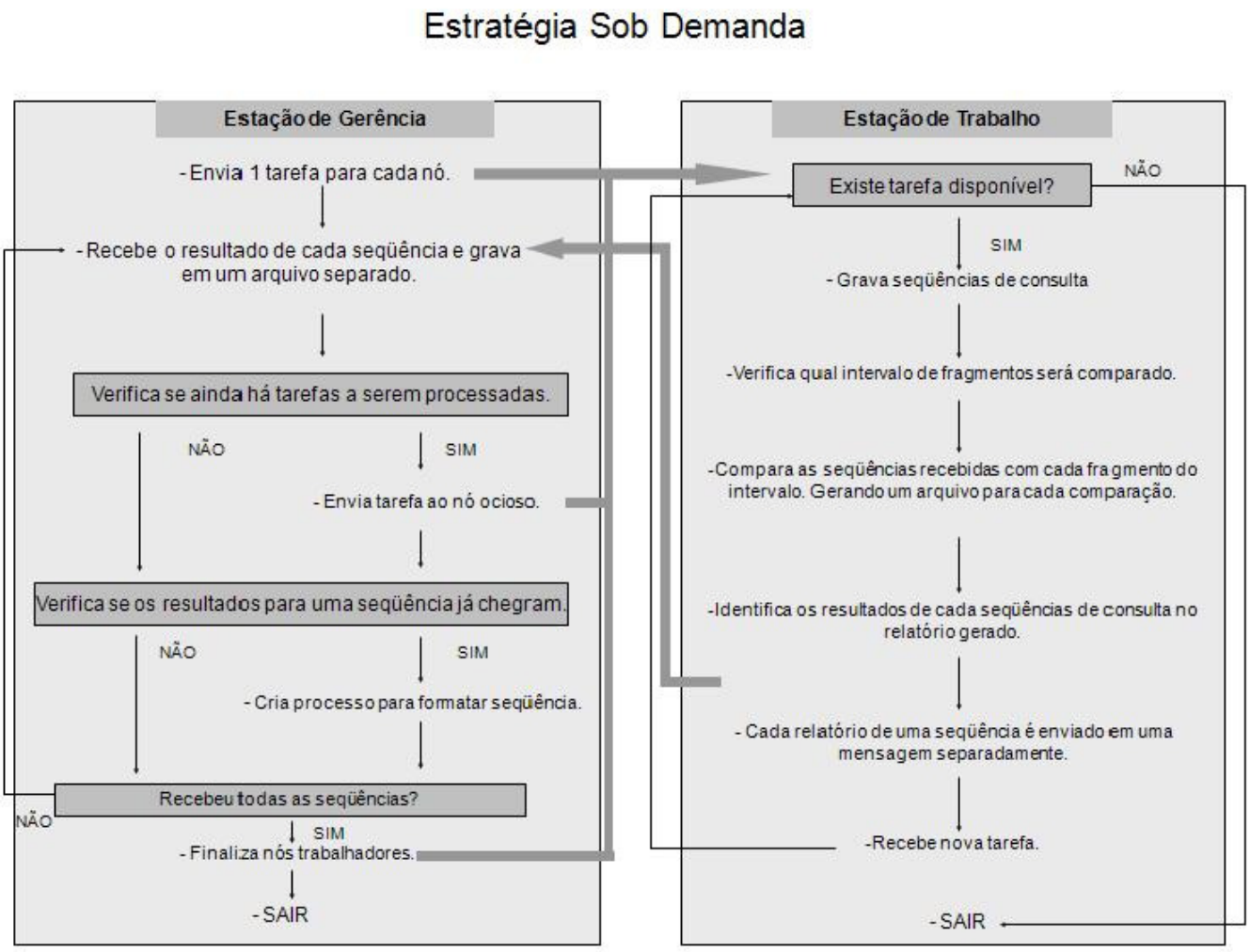

Figura 2.4: Algoritmo de funcionamento da estratégia Sob Demanda.

Em um primeiro momento, preferencialmente, cada máquina receberá tarefas a serem processadas com os respectivos fragmentos primários, até que todas as seqüências de consulta sejam processadas. A partir desse momento, se ociosa, a estação de trabalho recebe seqüências para outros fragmentos, a fim de manter o sistema balanceado, ajudando também a completar a execução do BLAST de forma mais rápida. Durante todo processamento do sistema, sempre que uma estação de trabalho executar uma tarefa, o resultado será enviado a uma única estação de gerência. Esta verifica se a seqüência recebida já foi comparada com todos os fragmentos da base de dados. Em caso positivo, o sistema cria um novo processo filho com o objetivo de montar um único relatório de saída para uma seqüência de consulta.

Durante os 12 meses de pesquisa, comprovou-se que as duas últimas estratégias descritas, a Fragmentada Corretiva e a Sob Demanda, são superiores às demais, como mostram as figuras 2.5 e 2.6.

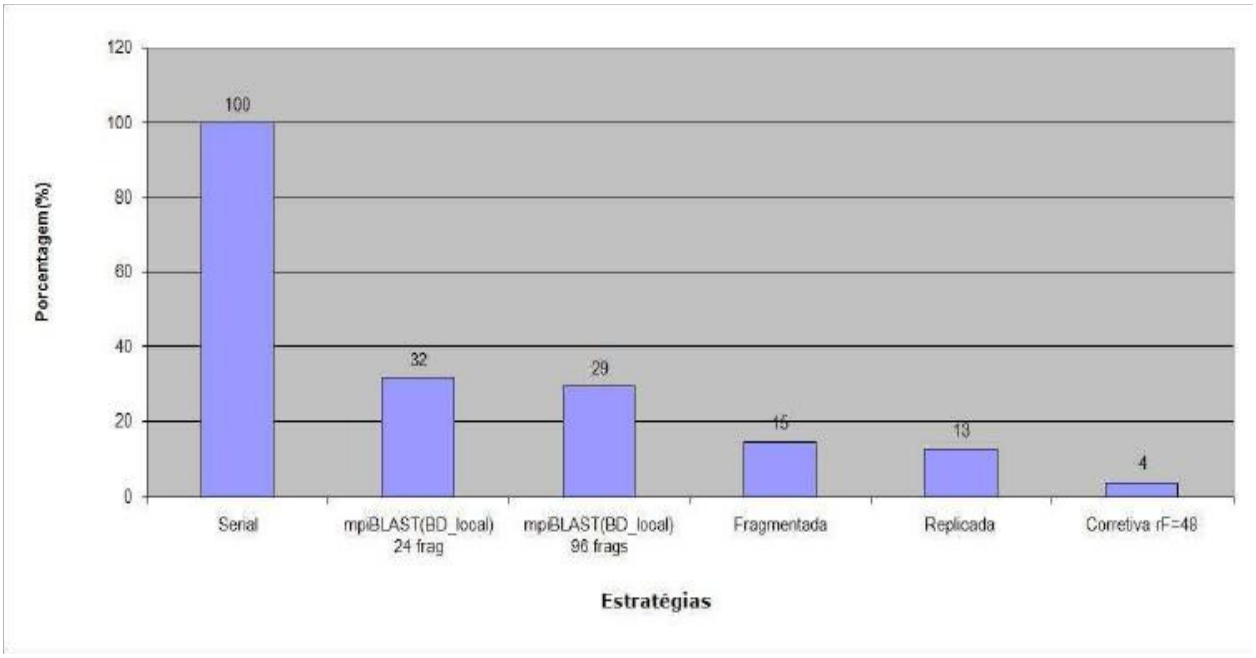

Figura 2.5: Compara a Estratégia Corretiva em relação às várias outras, mostrando a porcentagem em tempo de processamento em relação à estratégia Serial. Em todas as estratégias foram utilizadas as mesmas seqüências de consulta e o mesmo banco nt.

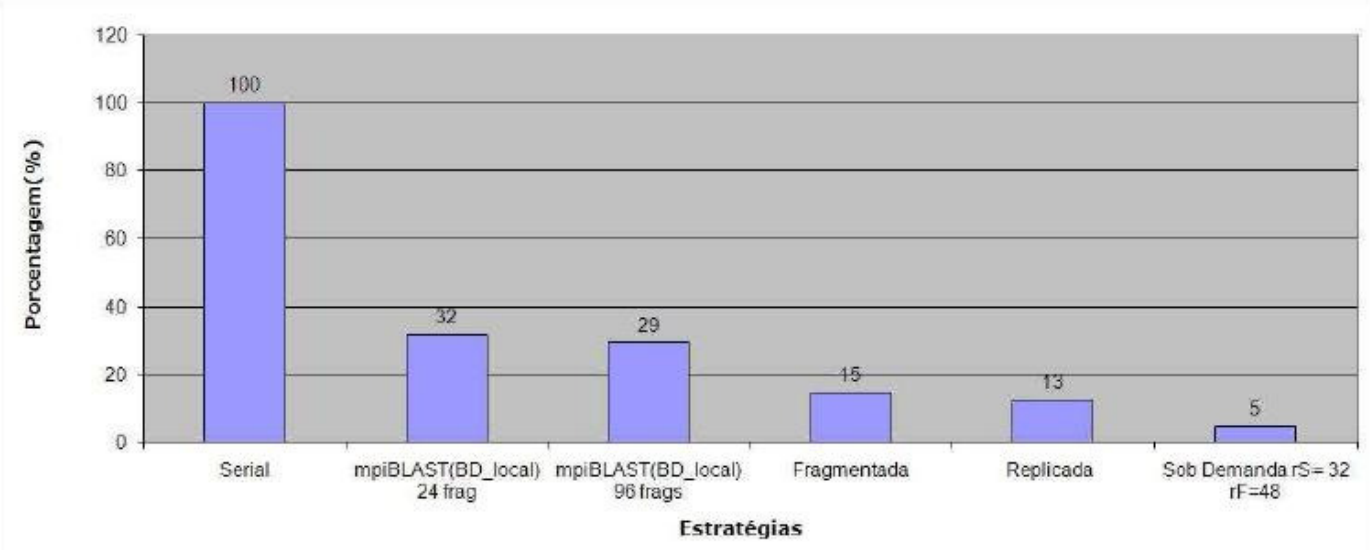

Figura 2.6: Apresenta o resultado da execução de várias estratégias e o desempenho da Estratégia Sob Demanda, mostrando a porcentagem em tempo de processamento em relação à estratégia Serial. Em todas as estratégias foram utilizadas as mesmas seqüências de consulta e o mesmo banco nt.

Ferramenta de Fragmentação

Ao realizar a implementação das novas estratégias, como citado anteriormente, o algoritmo de distribuição circular ("Round-robin") foi utilizado para a distribuição de seqüências pelos nós do cluster. Porém, devido às facilidades de processamento de texto, tal ferramenta foi desenvolvida na linguagem Perl, que apesar de ser amplamente utilizada no meio da bioinformática, não é muito portátil.

Como o objetivo da equipe do Laboratório de Bioinformática é disponibilizar as ferramentas de teste online para utilização, concluimos ser melhor traduzirmos o algoritmo para uma linguagem mais geral e portátil, como C. Dessa forma, uma das tarefas executadas nesses 12 meses de pesquisa foi a implementação dessa ferramenta.

O algoritmo é relativamente básico, mas de não tão simples implementação devido às imensas variedades de tamanhos de bancos de dados e de seqüências de consulta, o que torna o gerenciamento de memória um tanto quanto complicado. A ferramenta é composta de uma função que recebe como parâmetros o banco de dados genético, o número de fragmentos em que ele deve ser dividido, sua natureza (proteico ou nucleotídeos) e um parâmetro que define se os fragmentos criados devem ser fragmentados pela ferramenta do BLAST, formatdb.

Ao realizar-se a execução do programa, ele cria a quantidade respectiva de fragmentos e, seqüência por seqüência, vai lendo-as do banco de dados e escrevendo-as, de forma circular, em cada fragmento, até que todas elas estejam contidas em um dos arquivos criados. A fim de evitar maiores problemas de gerenciamento de memória, todo o processo é realizado a partir de strings estáticos, pois a alocação dinâmica de memória é muito dependente da capacidade da máquina e, caso estejamos tratando de um banco de dados com uma seqüência extremamente grande, a realocação de memória é problemática em diversos aspectos por ser uma função relativamente instável.

mpiBLAST

A ferramenta mais utilizada e divulgada que também se utiliza do processamento paralelo é o mpiBLAST [2]. É uma implementação que tira vantage das capacidades da computação distribuida, como um cluster, através da comunicação MPI e, dessa forma, utiliza, assim como as ferramentas do LabBio, todos os recursos computacionais disponíveis, diferentemente do NCBI BLAST, que só tira proveito de multiprocessadores com memória compartilhada.

Por se basear nos mesmos princípios que nosso laboratório se baseou, essa foi a ferramenta escolhida como base de comparações. Assim, uma das tarefas desse projeto de Iniciação Científica foi a utilização do cluster da PUC-Rio para executar, com o mpiBLAST, os mesmos testes realizados com as estratégias Corretiva e Sob Demanda.

Para tal, foi instalado no cluster a versão 1.5.0beta1 do programa, a mais atual à época da pesquisa. Devido à má documentação e falta de instruções do site da ferramenta, a sua implementação no cluster foi de extrema dificuldade, o que nos levou a mais uma pequena tarefa: o desenvolvimento de um tutorial bem explicado de como implementar a ferramenta num cluster, disponível no site [3] do Laboratório de Bioinformática.

Utilizando as mesmas bases de dados utilizadas com as estratégias desenvolvidas no LabBio, testamos a versão mais recente à época do mpiBLAST, confirmando que nossas estratégias ainda apresentam grande vantagem. Abaixo segue uma tabela comparativa dos resultados.

	FCO	FPU	FSD	MPI	REPL
N-04-08-500/2000	0:17:15	0:08:40	0:00:55	0:07:13	0:02:07
N-04-08-1000/2000	0:34:35	0:17:21	0:01:57	0:17:35	0:04:21
N-04-08-1500/2000	0:51:57	0:26:00	0:02:57	0:43:38	0:06:35
N-04-08-2000/2000	1:09:24	0:34:39	0:04:13	1:19:22	0:08:45
N-04-08-2000/2500	1:09:40	0:34:51	0:05:40	2:14:38	0:08:47
N-04-24-500/2000	0:51:41	0:08:40	0:02:00	0:08:03	0:02:13
N-04-24-1000/2000	1:43:26	0:17:18	0:04:13	0:21:31	0:04:22
N-04-24-1500/2000	2:35:41	0:25:59	0:06:48	0:48:31	0:06:34
N-04-24-2000/2000	3:27:54	0:34:36	0:09:39	1:30:24	0:08:47
N-04-24-2000/2500	3:29:51	0:34:39	0:10:59	2:38:59	0:08:46
N-08-08-500/2000	0:08:41	0:08:42	0:00:40	0:07:04	0:01:07
N-08-08-1000/2000	0:17:23	0:17:24	0:01:45	0:17:53	0:02:12
N-08-08-1500/2000	0:26:12	0:26:05	0:02:50	1:13:04	0:03:18
N-08-08-2000/2000	0:34:59	0:34:59	0:04:15	1:21:30	0:04:24
N-08-08-2000/2500	0:35:02	0:35:02	0:04:52	2:16:05	0:04:23
N-08-24-500/2000	0:26:06	0:08:42	0:01:51	0:07:54	0:01:06
N-08-24-1000/2000	0:52:07	0:17:49	0:03:16	0:19:30	0:02:12
N-08-24-1500/2000	1:18:27	0:26:02	0:05:23	0:45:33	0:03:18
N-08-24-2000/2000	1:44:59	0:34:47	0:08:32	1:24:10	0:04:24
N-08-24-2000/2500	1:45:06	0:34:54	0:09:03	2:16:24	0:04:25
P-04-08-500/2000	1:32:49	0:46:27	1:21:26	3:43:30	1:15:50
P-04-24-500/2000	1:52:07	0:18:39	1:25:23	4:53:33	1:15:34

FCO corresponde à estratégia Fragmentada Corretiva, FPU à Fragmentada Pura, FSD à estratégia Sob Demanda, MPI ao mpiBLAST [2] e REPL à Replicada. Os nomes anteriores aos tempos estão no seguinte formato: P/N para proteínas ou nucleotídeos; 04/08 representando a quantidade de máquinas utilizadas no cluster para o processamento; 08/24 representando em quantos fragmentos a base de dados foi dividida; e, por último os limites inferior/superior do tamanho das seqüências geradas para a realização dos testes. Cabe ainda salientar que os 20 primeiros testes foram realizados com seqüências geradas a partir da base ecoli.nt, enquanto os dois últimos, a partir da base SwissProt.

É notável a vantagem da estratégia Sob Demanda frente às demais.

Site do LabBio

Em função dos progressos conseguidos com o sucesso dos testes das estratégias desenvolvidas no Laboratório de Bioinformática, ficou evidente a necessidade do desenvolvimento de um site mais completo e mais informativo sobre tudo que foi feito nesses anos de pesquisa. Por isso, mais uma das tarefas desenvolvidas nesses 12 meses foi a atualização do site do LabBio [3] para uma página mais completa, disponibilizando o download do código das estratégias desenvolvidas, da ferramenta de fragmentação e de textos e relatórios das pesquisas realizadas.

Conclusões

Notadamente, as estratégias desenvolvidas no Laboratório de Bioinformática obtiveram desempenho superior ao mpiBLAST, na grande parte dos testes. As estratégias Fragmentada

Pura e Replicada foram as que obtiveram piores desempenhos, nessa ordem, entre as quatro desenvolvidas na PUC. O desempenho da estratégia Fragmentada Pura é sempre o desempenho do nó mais lento, ou melhor, que demorou mais tempo para efetuar suas comparações. A estratégia Replicada, por sua vez, tem um tempo de pré-processamento muito alto, além de não tratar o caso da ociosidade de uma máquina também.

Por outro lado, a estratégia Fragmentada Sob Demanda obteve tempos muito bons, muito inferiores aos apresentados pelo mpiBLAST. Ela evita que um computador se torne ocioso enquanto outro ainda está em processamento, ou seja, o paralelismo está sempre acontecendo, reduzindo o tempo da melhor forma possível.

Referências

1 – SOUSA, Daniel Xavier de. **Estratégias de Balanceamento de Carga para Avaliação Paralela do BLAST com Bases de Dados Replicadas e Fragmentos Primários**. Rio de Janeiro, 2007. 84p. Dissertação de Mestrado, Departamento de Informática, PUC-Rio.
2 – http://www.mpiblast.org
3 – http://www.puc-rio.br/~blast