Pontifícia Universidade Católica do Rio Grande do Sul
Faculdade de Informática
Programa de Pós-Graduação em Ciência da Computação

Um *Workflow* Científico para a Modelagem do Processo de Desenvolvimento de Fármacos Assistido por Computador Utilizando Receptor Flexível

Karina dos Santos Machado

Dissertação apresentada como requisito parcial à obtenção do grau de mestre em Ciência da Computação

Orientador: Prof. Dr. Osmar Norberto de Souza

Porto Alegre
2007

Pontifícia Universidade Católica do Rio Grande do Sul

Dados Internacionais de Catalogação na Publicação (CIP)

M149w Machado, Karina dos Santos
Um Workflow científico para a modelagem do processo de desenvolvimento de fármacos assistido por computador utilizando receptor flexível / Karina dos Santos Machado. – Porto Alegre, 2007.
77 f.

Diss. (Mestrado) – Fac. de Informática, PUCRS
Orientador: Prof. Dr. Osmar Norberto de Souza

1. Bioinformática. 2. Dinâmica Molecular. 3. Workflow. 4. Informática. I. Título.

CDD 005.431

Ficha Catalográfica elaborada pelo Setor de Processamento Técnico da BC-PUCRS

Campus Central
Av. Ipiranga, 6681 - prédio 16 - CEP 90619-900
Porto Alegre - RS - Brasil
Fone: +55 (51) 3320-3544 - Fax: +55 (51) 3320-3548
Email: bceadm@pucrs.br
www.pucrs.br/biblioteca

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada "Um Workflow Científico para a Modelagem do Processo de Desenvolvimento de Fármacos Assistido por Computador Utilizando Receptor Flexível", apresentada por Karina dos Santos Machado, como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, Bioinformática e Modelagem Computacional, aprovada em 30/03/2007 pela Comissão Examinadora:

Prof. Dr. Osmar Norberto de Souza – PPGCC/PUCRS
Orientador (a)

Prof. Dr. Duncan Dubugras Alcoba Ruiz – PPGCC/PUCRS

Prof. Dr. João Eduardo Ferreira – USP/SP

Homologada em 04/06/2007, conforme Ata No. 013/2007 pela Comissão Coordenadora.

Prof. Dr. Fernando Luís Dotti
Coordenador.

PUCRS
Campus Central
Av. Ipiranga, 6681 – P. 32 – sala 507 – CEP: 90619-900
Fone: (51) 3320-3611 – Fax (51) 3320-3621
E-mail: ppgcc@inf.pucrs.br
www.pucrs.br/facin/pos

Agradecimentos

Em primeiro lugar agradeço à minha família a quem dedico esse trabalho. Aos meus pais, exemplos de vida, que mesmo longe contribuíram muito para que concluísse mais esta etapa. Ao Renan, meu irmão que sinto muita saudade. À Ana, que além de ser um exemplo de perseverança, sempre me incentivou muito. À Raquel, por tudo que ela fez enquanto estava morando em Porto Alegre, muito obrigada! Sem tua presença aqui, com certeza tudo teria sido bem mais difícil. Obrigada pelas jantas, pela companhia, por dar um jeito em tudo quando eu não tinha tempo, bem... por tudo mesmo! Agradeço a toda minha família, que de alguma forma me ajudou durante o desenvolvimento desse trabalho: meus avós, Didi, meu afilhado amado, tios, tias, primos e primas. Agradeço ao Guilherme, pela companhia e apoio incondicional nesse meu último ano de mestrado.

Agradeço ao PPGCC da PUCRS e à CAPES, essenciais para o desenvolvimento desse trabalho. Agradeço a todos os Professores do programa, em especial ao Professor Duncan, pela atenção dada ao meu trabalho e pelas opiniões sempre válidas. Agradeço muito ao meu orientador, Professor Osmar pela confiança depositada em mim. Tenho certeza de que contribuiu muito para o meu desenvolvimento pessoal e intelectual.

Um agradecimento especial a Evelyn. Tenho certeza que sem a presença dela tudo teria sido bem mais difícil. Obrigada pela orientação, ajuda, disposição, sempre. Agradeço também aos demais membros do LABIO, companhia de trabalho sempre agradável. E por fim, mas não menos importante, agradeço a todos meus amigos queridos.

“A confiança em si mesmo é o primeiro segredo do sucesso.” *(Ralph Waldo Emerson)*

Resumo

O desenho de drogas assistido por computador (CADD) é um processo que envolve a execução seqüencial de diferentes programas, no qual é testado se um determinado ligante (pequena molécula) interage bem com um receptor (geralmente uma proteína ou enzima). Esse processo geralmente é executando com o auxílio de *shell scripts*. Porém a modificação dos parâmetros de entrada e análise dos resultados nesse tipo de abordagem é uma tarefa complexa e que consome muito tempo. Além disso, para considerar a flexibilidade do receptor durante experimentos de *docking*, é necessário que se utilize milhares de *snapshots* do receptor. Neste trabalho, esses *snapshots* são obtidos da trajetória de simulações por dinâmica molecular do receptor. Devido aos desafios associados à manipulação desse grande número de *snapshots* do receptor e à necessidade de um melhor controle sobre os diferentes programas associados a esse processo, esse trabalho apresenta um *workflow* científico para automação do processo de desenvolvimento de drogas assistido por computador, incluindo de forma explícita a flexibilidade do receptor. Os *softwares* JAWE e Shark foram utilizados para modelar e executar respectivamente o *workflow*. Mesmo com essa automação no processo, ainda há problemas relacionados ao número de *snapshots* do receptor que deve ser utilizado. O tempo necessário para a execução de experimentos de *docking*, considerando aproximadamente três mil *snapshots* do receptor, é em torno de 500 horas. Para simplificar e agilizar esse processo, foi desenvolvida uma forma de seleção de *snapshots* baseado na energia livre de ligação (FEB). Assim, durante os experimentos de *docking*, chamados de *docking* seletivo, somente uma parte dos *snapshots* são utilizados: aqueles que obtiveram os melhores resultados de interação em termos de FEB durante um experimento exaustivo com um determinado ligante. Para validar essa implementação e seleção, foram executados experimentos de *docking* com a enzima InhA de *M. tuberculosis* como receptor e cinco ligantes diferentes. Os resultados desses experimentos ilustram a eficiência do *workflow* implementado e da forma de seleção de *snapshots* do receptor que está sendo realizada.

Palavras-chave: Bioinformática, *Drug Design*,*Workflows* Científicos, *Docking* Molecular, Dinâmica Molecular.

Abstract

Computer assisted drug design (CADD) is a process involving the sequential execution of many computer programs, where the output of one program is the input for the next, ensuring that the ligand, a small molecule, optimally binds to its receptor, often a protein or enzyme. This process is usually executed using shell scripts whose input parameters modification and result analysis are complex and time consuming. Moreover, in order to explicitly consider the receptor flexibility during docking experiments, it is necessary to use thousands of receptor snapshots. These snapshots are obtained from a molecular dynamics simulation trajectory of the receptor. Due to the challenges in handling a large number of receptor snapshots and the need of a better execution control of a number of different computer programs, this work presents an integrated scientific workflow solution aiming at the automation of a flexible and efficient CADD process, with explicit inclusion of receptor flexibility. The JAWE and Shark software were used to model and execute the workflow, respectively. Even with automation, there still are problems with the number of receptor snapshots. The time spend during the execution of one process considering about three thousands receptor's snapshots is about 500 hours. To overcome this problem, was developed a procedure for snapshot selection based on the free binding energy (FEB). Thus, during selective docking experiments, a smaller number of snapshots are used: those that had the best results of FEB during an exhaustive experiment with a reference ligand. To validate its implementation and selection we performed docking experiments with the *M. tuberculosis* enzyme InhA as the receptor and five different ligands. Some of theses experiments were extensively performed, considering all receptor snapshots, and some were selectively performed. The results illustrate the efficiency of the proposed workflow and implementation of the CADD process as well as the method for the receptor snapshots selection.

Keywords: Bioinformatics, Drug Design, Scientific Workflows, Molecular Docking, Molecular Dynamic.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

3D	Tridimensional	20
AMBER	*Assisted Model Building with Energy Refinement* - Programa de Simulação por Dinâmica Molecular	22
AutoDock	*Automated Docking* - Programa de *Docking* Molecular	20
BLAST	*Basic Local Alignment Search Tool*	34
CADD	*Computer Assisted Drug Design*	16
DM	Dinâmica Molecular	22
DNA	*Deoxyribonucleic Acid* ou Ácido Desoxirribonucléico, em português	33
ETH	Etionamida	61
FEB	*Free Energy of Binding*	17
InhA	enzima 2-*trans-Enoil ACP(CoA) Reductase de* Mycobacterium tuberculosis	22
IQG607A	Isoniazida pentacianoferrato	59
IQG607B	Diisioniazida tetracianoferrato	60
LABIO	Laboratório de Bioinformática, Modelagem e Simulação de Biossistemas - FACIN/PUCRS	37
NADH	Nicotinamida Adenina Dinucleotídeo, forma reduzida	22
NCBI	*National Center for Biotechnology Information*	34
PDB	*Protein Data Bank*	20
PIW	*Promoter Identification Workflow*	33
RMSD	*Root Mean Square Deviation* ou Desvio Médio Quadrático, em português	46
SPDBV	*Swiss PDB Viewer*	28

SPEC	*Standard Performance Evaluation Corporation*	58
TCL	Triclosan	61
TI	*Tecnologia da informação*	24
WAPI	*Workflow APIs and Interchange Formats*	23
WFMC	*Workflow Managment Coaliation*	23
WfMS	*Workflow Managment System*	24
XPDL	XML *Process Definition Language*	28

Sumário

1 Introdução

Segundo Luscombe *et al.* [2] uma das principais características em Bioinformática é a coleção, organização e interpretação de grandes quantidades de dados. Para alcançar esse objetivo a comunidade científica vem, cada vez mais, utilizando computadores para execução e análise de seus experimentos. Geralmente esses experimentos, chamados *in silico*, correspondem à composição de vários programas em seqüência, onde a saída de um é utilizada como entrada de outro e a execução dos mesmos é realizada manualmente ou utilizando *shell scripts*.

Este tipo de abordagem geralmente possui grande deficiência em questões como heterogeneidade e natureza distribuída dos dados devido a formatos específicos de entrada e saída das ferramentas disponíveis [3]. Além disso, a execução manual de programas seqüenciais ou o uso de *shell scripts* apresentam problemas relacionados à clareza, flexibilidade, registros de uso e do fluxo dos dados e manutenção do processo.

Uma proposta atraente para a descrição desse tipo de experimento, de acordo com Wainer [4], seria o uso de *workflows* científicos, pois além de fornecer o apoio necessário ao ciclo de execução e análise, torna possível a criação de um ambiente com independência entre as diversas aplicações científicas. Esses *workflows* envolvem seqüências de passos que podem lidar com acessos a bancos de dados, mineração de dados, análise de dados e muitos outros passos envolvendo tarefas computacionais [4].

Da Biologia se sabe que macromoléculas (receptores)[1], como por exemplo, proteínas, enzimas e DNA não são rígidas em seu ambiente celular. Por esse motivo, essa flexibilidade deve ser explicitamente considerada durante o processo de desenvolvimento de novas drogas (ou em inglês, *drug design*). Uma das principais etapas desse processo de *drug design* é o *docking* molecular no qual, a interação de uma pequena molécula (ligante)[2] com um determinado receptor é computacionalmente testada e avaliada [5].

Experimentos de *docking* molecular podem ser executados por diferentes *softwares*, como por exemplo: o AutoDock3.05 [6], o FLEXX [7] e o DOCK4.0 [8]. A maioria desses *softwares* trata a flexibilidade do ligante, mas apresentam dificuldades em considerar a flexibilidade do receptor [5]. Os *softwares* que consideram a flexibilidade do receptor, fazem isso somente de uma maneira muito limitada. Com o objetivo de contornar esse problema e incluir uma representação mais realista da flexibilidade natural dos receptores durante os experimentos de *docking*, nós consideramos um conjunto de *snapshots* [3] do receptor gerados por uma simulação

[1] No texto, proteínas, macromoléculas e receptor são utilizadas com o mesmo signifi cado.
[2] Ligantes, inibidores e pequenas moléculas são tratados como sinônimos no texto.
[3] Estruturas instantâneas, conformações e *snapshots* são tratados como sinônimos no texto.

da dinâmica molecular onde, para cada possível conformação do receptor, um experimento de *docking* deve ser executado e analisado.

Os passos envolvidos na execução de experimentos de *docking* dessa forma, considerando a flexibilidade do receptor oriunda de simulações por dinâmica molecular geralmente são executados manualmente, no qual os parâmetros, assim como a seqüência de execução, são definidas pelo usuário. Conseqüentemente, se alguém desejar continuar o trabalho de outro membro do mesmo grupo ou reexecutar o processo com diferentes ligantes e/ou proteínas, encontrará muitas dificuldades no que diz respeito ao ajuste dos parâmetros de entrada e ao acompanhamento e registro de todo o processo.

Com base no problema exposto acima, a proposta deste trabalho é modelar o processo de desenvolvimento de fármacos assistido por computador (ou em inglês, *Computer Assisted Drug Design* - CADD) considerando o receptor flexível, utilizando *workflows* científicos e implementando *shell scripts* e programas que executem efetivamente e de forma automática as seqüências das atividades descritas pelo *workflow*. Esse modelo permitirá que *dockings* moleculares que considerem a flexibilidade explícita tanto do ligante como do receptor sejam facilmente executadas, sendo então a flexibilidade natural das moléculas biológicas consideradas nos experimentos de *docking*. Além disso, com a automatização do processo, é possível a inclusão de uma etapa de seleção de *snapshots* para que não seja necessária a consideração de todas as conformações do receptor, reduzindo o tempo necessário para se analisar a interação receptor-ligante.

1.1 Justificativa

Um método de *docking* molecular ideal seria aquele que permitisse que ambos, ligante e receptor, explorassem todos os seus graus de liberdade conformacionais. Como alternativa para a consideração da flexibilidade da macromolécula em experimentos de *docking* molecular, atualmente têm sido feitos experimentos de *docking* que utilizam, cada um deles, uma estrutura instantânea da proteína (*snapshot*) [9], gerada a partir de simulações por dinâmica molecular utilizando o *software* AMBER6.0 [10], como por exemplo, o trabalho apresentando por Kua *et al.* [9].

Esse processo, na maioria das vezes, é executado de forma manual ou com no máximo a ajuda de alguns *shell scripts*. Porém, esse tipo de abordagem apresenta muitos problemas como, por exemplo, a dificuldade de definição da ordem correta em que cada etapa deve ser executada (o que torna muito difícil a compreensão do processo como um todo), a monitoração da execução do processo, a execução do mesmo utilizando parâmetros diferentes, entre outros problemas.

Sendo assim, com base na idéia da funcionalidade de *workflows* científicos, optou-se por utilizá-los na definição e execução de todas as etapas do processo de desenvolvimento de fármacos assistido por computador, considerando a flexibilidade da macromolécula.

Além dos problemas mencionadas acima, cada simulação por dinâmica molecular gera no mínimo em torno de 5000 *snapshots*. A consideração de todas essas possíveis conformações do receptor nos experimentos de *docking* consiste em uma tarefa com um alto custo computacional. Desse modo, o presente trabalho também busca desenvolver uma maneira efetiva de selecionar *snapshots* do receptor, de forma que a utilização de todos os *snapshots* gerados na etapa de simulação por dinâmica molecular não seja necessária para se analisar a interação receptor-ligante.

1.2 Motivação

Atualmente o processo de CADD considerando receptor flexível é desenvolvido de forma manual, no qual, se alguém desejar continuar o trabalho de outro membro de um mesmo grupo, ou se quiser reiniciar o processo utilizando uma proteína ou ligante diferentes sem conhecer a seqüência exata dos passos, vai encontrar muitas dificuldades, principalmente em relação à definição dos parâmetros de entrada dos *softwares* e à seqüência com que esses devem ser executados. Do mesmo modo, é muito complicado acompanhar a execução do processo como um todo sem se saber que etapa está sendo executada e como a mesma foi parametrizada.

Além disso, considerar a flexibilidade do receptor nesses experimentos de *docking* deve-se ao fato de que se sabe que as macromoléculas não são rígidas em seu ambiente celular [11]. Esta flexibilidade deve ser considerada na busca de novos inibidores e no entendimento da associação com determinado ligante, via *docking* molecular, que geralmente consideram a estrutura do receptor rígida. A consideração da estrutura do receptor flexível por parte dos *softwares* de *docking* é uma tarefa impraticável devido ao processamento que seria necessário, pois receptores geralmente são moléculas grandes, compostas por um grande número de átomos que podem se movimentar de diferentes formas e em diferentes sentidos.

Assim, se esse processo for realizado de forma automática, sua execução poderá ser feita por mais tempo sem que haja necessidade de intervenção humana, permitindo que a interação de muitos ligantes com uma proteína alvo seja analisada de forma mais rápida, aumentando a chance de se encontrar um bom ligante em menos tempo. Ademais, o processo se torna mais flexível e com uma seqüência de passos pré-definida, não havendo problema em membros do mesmo grupo compartilharem o mesmo experimento, por exemplo.

Porém, após as simulações por dinâmica molecular, muitos *snapshots* do receptor são gerados e a utilização de todos eles em experimentos de *docking* seria muito custosa e levaria muito tempo para se analisar a interação receptor-ligante. Por esse motivo, adicionando ao *workflow* uma etapa de seleção dos *snapshots* gerados na simulação por dinâmica molecular, utilizando inicialmente um critério bem simples por energia livre de ligação - FEB (*Free Energy of Binding*), o tempo necessário para analisar cada interação entre um receptor flexível e um ligante se reduz consideravelmente.

Além das motivações já descritas, ainda pode-se acrescentar que o uso de *workflows* para solucionar problemas relacionados com *drug design* tem aumentando, principalmente entre algumas empresas farmacêuticas [12]. Assim, o trabalho contribuirá para a comunidade científica em geral, já que uma integração entre os *softwares* de simulação por dinâmica molecular e de *docking* molecular permite que os experimentos considerando o receptor flexível sejam mais simples de serem executados. Além do mais, o uso de critérios de qualidade como uma maneira de selecionar instâncias do processo a serem executadas poderá ser implementada em outros processos que apresentam as mesmas características.

1.3 Objetivos e Contribuições

- Estudar a modelagem de processos utilizando *workflows* científicos. Esse estudo envolve o aprendizado da terminologia envolvida e principalmente das principais ferramentas já disponíveis;
- Modelar o processo de CADD considerando o receptor flexível, utilizando *workflows* científicos. Para que isso seja possível, é necessário fazer um *docking* molecular seqüencial utilizando estruturas instantâneas (*snapshots*) de uma macromolécula geradas a partir de simulações por dinâmica molecular. Essa é uma alternativa para considerar a estrutura da proteína flexível, já que o AutoDock3.05 [6] (programa utilizado para realizar os experimentos de *docking*) considera somente o ligante flexível;
- Implementar *shell scripts* e programas que efetivamente integrem os *softwares* que são utilizados e automatize o processo de CADD por completo;
- Desenvolver critérios de qualidade para pré-selecionar os experimentos de *docking* que devem ser efetivamente executados pelo *workflow*. O critério de qualidade escolhido para pré-selecionar *snapshots* será a FEB. Após um experimento ser executado por completo, de maneira exaustiva, considerando todos os *snapshots* gerados na simulação por dinâmica molecular, seus resultados indicam quais os *snapshots* que apresentaram melhor resultado e esses são utilizados em experimentos com outros ligantes. Essa seleção é importante para tornar o processo de *virtual screening* viável de ser realizado considerando o receptor flexível.

1.4 Organização do Documento

Este documento organiza-se da seguinte forma: no Capítulo 2 é feita uma breve descrição de alguns conceitos importantes para o entendimento do trabalho: *drug design*, *docking* molecular

e dinâmica molecular. Após, no Capítulo 3 são descritos conceitos de *workflows*, em especial *workflows* científicos. Esse capítulo descreve as ferramentas que foram selecionadas para o uso nesse trabalho assim como alguns trabalhos relacionados ao uso de *workflows* científicos em Bioinformática. O *worflow* desenvolvido, com a descrição de todas as etapas envolvidas são descritas de forma detalhada no Capítulo 4. Então, no Capítulo 5 5, é apresentado um Estudo de Caso para ilustrar o uso do *workflow* desenvolvido, assim como os resultados referentes ao uso do primeiro critério de qualidade desenvolvido para pré selecionar *snapshots*. Finalmente, o Capítulo 6 apresenta algumas conclusões e os trabalhos ainda a serem realizados.

2 Referencial Teórico

Este capítulo tem por objetivo definir conceitos diretamente relacionados com este trabalho, a fim de contextualizá-lo. A Seção 2.1 descreve o processo de Drug Design e conceitua *docking* molecular e dinâmica molecular.

2.1 Drug Design, Docking Molecular e Dinâmica Molecular

Devido ao avanço da biologia molecular e de ferramentas de simulação *in-silico*, o planejamento de medicamentos passou a ser feito de maneira mais lógica, o que é chamado de Desenho Racional de Drogas [13]. Esse processo é baseado em análises teóricas das interações entre pequenas moléculas e receptores [14]. Segundo Bajorath [15] é necessário testar várias combinações de interações entre ligantes (pequenas moléculas) e receptores (proteínas alvo) para se obter um medicamento.

Segundo Kuntz [13], esse processo envolve basicamente quatro etapas:

1. A primeira etapa do processo de *drug design* consiste em isolar um alvo específico, chamado receptor (geralmente uma proteína) [16]. A partir da análise computacional da estrutura 3D dessa proteína (a estrutura da proteína é determinada por cristalografia por difração de raios X ou ressonância magnética nuclear [17] e armazenada em um banco de dados estrutural como o *Protein Data Bank* - PDB [18]) é possível apontar prováveis regiões de ligação, por exemplo, regiões onde uma pequena molécula (ligante) pode se ligar a esse receptor;

2. Baseado nas prováveis regiões de ligação identificadas no passo anterior, é selecionado um conjunto de possíveis candidatos, chamados ligantes (usualmente pequenas moléculas que podem ser buscadas em Bancos de Dados de compostos como o ZINC [19]) que podem se ligar a essa região no receptor. As diferentes conformações que determinado ligante pode assumir dentro do sítio de ligação de uma determinada proteína são simuladas por *softwares* de *docking* como AutoDock 3.05 [6], FLEXX [7] e DOCK4.0 [8];

3. Os ligantes que teoricamente obtiveram melhores resultados nas simulações são experimentalmente sintetizados e testados;

4. Baseado nos resultados experimentais, o medicamento é gerado ou o processo retorna ao passo 1.

De acordo com Lengauer e Rarey [20], *docking* molecular é a base para o processo de CADD, fornecendo estimativas sobre a afinidade e maneira que certa proteína e ligante estão interagindo. No processo de *docking* molecular, a interação ligante-receptor é virtualmente testada e analisada por um algoritmo no qual o ligante assume diferentes orientações e conformações dentro do sítio de ligação do receptor. Um grande número de diferentes interações deve ser testado para identificar o melhor encaixe do ligante no sítio de ligação da estrutura alvo ou receptor. Esta informação é computada em termos da energia livre de ligação (FEB), que quanto mais negativa, melhor a interação ligante-receptor. Na Figura 1 é apresentado o processo de *docking* em três dimensões. Uma molécula de ligante (NADH em ciano) se liga ao sítio da molécula receptor (a proteína InhA [21]).

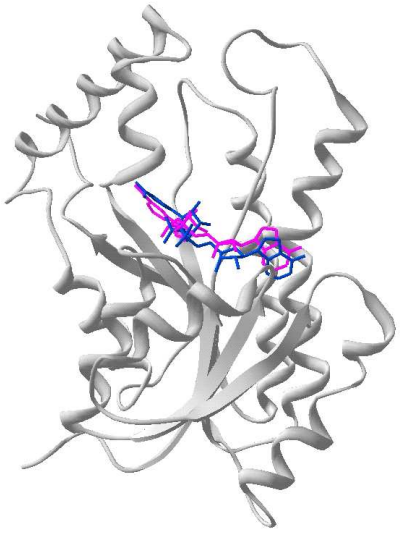


Figura 1 – Representação esquemática do processo de *docking* em 3D. A proteína é representada na forma de *ribbons* em cinza e o ligante em linhas. A conformação ideal do ligante aparece em magenta e a conformação do ligante, ao final de um experimento de *docking*, aparece em ciano.

Entretanto, a limitação dos *softwares* de *docking* geralmente ocorre quando se deseja considerar a flexibilidade explícita do receptor. Atualmente existe um grande número de alternativas para incorporar pelo menos parte da flexibilidade do receptor, por exemplo: tratar como flexíveis algumas cadeias laterais do receptor [5], utilizar um pequeno conjunto de cadeias laterais rotacionáveis [22], simular grandes movimentações usando métodos harmônicos [23], entre outras possibilidades. Porém, de acordo com Carlson [24], o uso de várias estruturas diferentes do receptor tem se caracterizado como a melhor alternativa para incorporar a flexibilidade do receptor nos experimentos de *docking*. Sendo assim, uma maneira de simular a flexibilidade do receptor nos experimentos de *docking* é utilizar um conjunto de possíveis conformações do receptor geradas por meio de simulação por dinâmica molecular [25].

Segundo Sali [26], estudos de sistemas biológicos eram inicialmente limitados à observação e interpretação de dados experimentais. Porém, técnicas experimentais somente descrevem características macroscópicas que são as mesmas características apresentadas por um conjunto

de moléculas. Entretanto, de acordo com van Gunsteren e Berendsen [27], com o avanço de técnicas experimentais tornou-se possível uma visão mais detalhada de diversos processos biológicos pelo acesso a propriedades atômicas de macromoléculas biológicas, como proteínas. O acesso a esse tipo de informação permitiu o desenvolvimento de estudos de simulação por dinâmica molecular que têm por objetivo simular o movimento natural de átomos em moléculas, como proteínas.

A Figura 2 ilustra as mudanças conformacionais durante a trajetória de simulação por dinâmica molecular (DM) da enzima InhA de *Mycobacterium tuberculosis* [21].

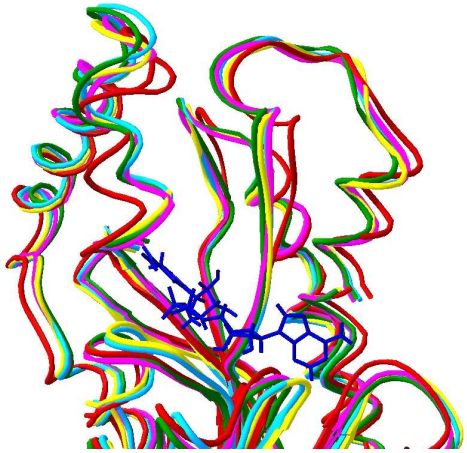


Figura 2 – Flexibilidade do sistema InhA-NADH em diferentes momentos ao longo de uma simulação por DM. Cada cor representa a conformação da proteína em um instante diferente: em vermelho a estrutura cristalográfi ca, em magenta a estrutura média de 0 a 500 ps (1 ps = 10^{-12} s), em verde de 550 a 1000 ps, em ciano de 1050 a 1500 ps e em amarelo de 1550 a 2000 ps. Em azul, em linhas, está o ligante NADH.

A proteína em estudo nesse trabalho é a InhA [21], a enzima 2-*trans-Enoil ACP(CoA) Reductase* de *Mycobacterium tuberculosis*, cujo estudos de simulação por DM complexada com a coenzima Nicotinamida Adenina Dinucleotídeo, forma reduzida (NADH) já foram previamente realizados [28] utilizando o *software* AMBER6.0 [10].

2.2 Considerações Finais

Esse capítulo apresentou os principais conceitos biológicos envolvidos no trabalho: *drug desing*, *docking* molecular e dinâmica molecular com o objetivo de facilitar o entendimento do restante do trabalho. O próximo capítulo introduz a área de *workflows* com enfoque especial em *workflows* científicos. São apresentados a terminologia básica da área, os tipos de *workflow*, as características dos *workflows* científicos, as ferramentas que foram selecionadas para serem utilizadas no trabalho e por fim, alguns trabalhos da área de Bioinformática que utilizaram *workflows* científicos como ferramenta para solução de seus problemas específicos.

3 Workflow

O presente capítulo descreve a terminologia básica da área de *workflows*. São descritos os principais conceitos: *workflows*, sistemas gerenciadores de *workflow*, processo, definição de um processo, atividades, transições, instâncias de processos e atividades, participantes do *workflow* e WAPI. Após, são abordados os tipos de *workflow*, destacando-se *workflows* científicos, no qual são apresentadas suas principais características e etapas. Logo a seguir, as ferramentas selecionadas para o desenvolvimentos do trabalho são descritas e os motivos que levaram a essa escolha são listados. Por fim, são apresentados dois trabalhos na área de Bioinformática, que assim como o nosso trabalho, utilizaram *workflows* científicos para resolução de seus problemas.

3.1 Terminologia Básica

A WFMC, um consórcio de empresas provedoras de soluções *workflow*, define em *Workflow Managment Coaliation - Terminology and Glossary* [29] a terminologia básica para entendimento de *workflows*. A Figura 3 mostra os principais termos envolvidos assim como a relação entre eles.

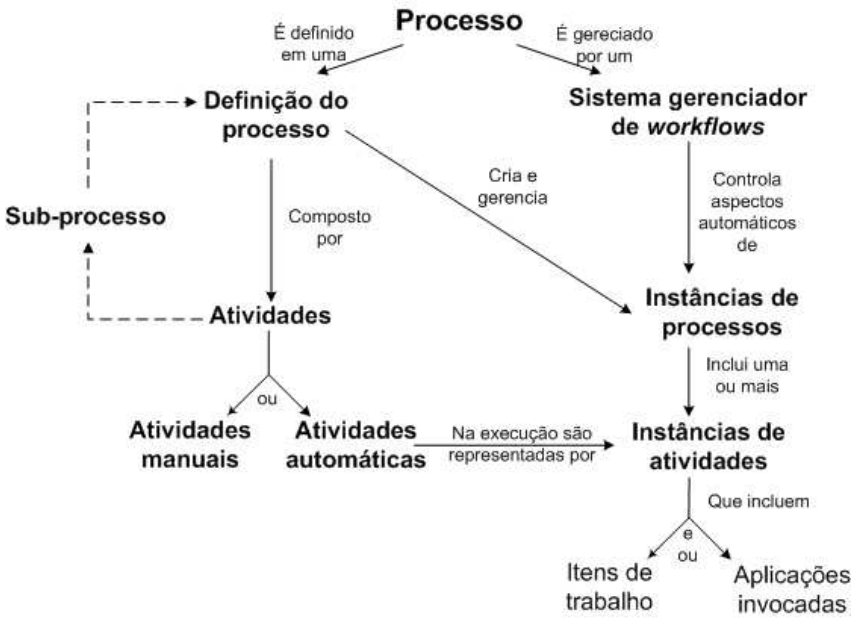


Figura 3 – Terminologia básica da área de *workflows*.

Workfbw: Segundo a WFMC [29], *workflow* pode ser definido como "a automação do processo de negócio, completamente ou em parte, na qual documentos, informações ou tarefas são passadas de um participante a outro de acordo com um conjunto de regras".

WfMS: *Workflow Managment System* ou em português, Sistema Gerenciador de *Workflows* é "um sistema que define, cria e gerencia a execução de *workflows* por meio do uso de um *software*, executando um ou mais *workflow engines* capazes de interpretar a definição do processo, interagir com os participantes do *workflow* e, quando necessário, invocar o uso de ferramentas de TI e aplicações".

Processo: Um conjunto de um ou mais procedimentos ou atividades relacionadas que, conectadas, atingem um objetivo comum, normalmente dentro de um contexto organizacional. A Figura 4 apresenta um exemplo de um processo composto por oito atividades. Quando um processo é chamado por outro processo, este é chamado de **sub-processo** ou ***subfbw***.

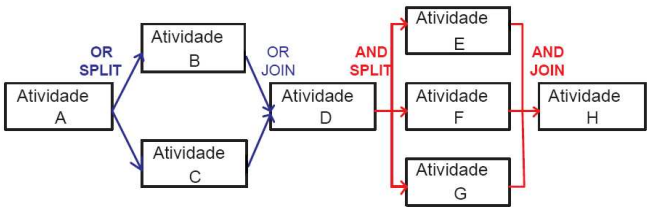


Figura 4 – Exemplo de um processo composto por 8 atividades. Em vermelho estão marcadas transições do tipo AND JOIN e AND SPLIT e em azul estão indicadas transições do tipo OR JOIN e OR SPLIT.

Defi nição de um processo: Um processo é descrito por uma definição de processo que consiste em uma rede de atividades e seus relacionamentos. Devem-se informar o início e o fim do processo, informações a respeito de cada uma de suas atividades e participantes, ferramentas que estão associadas, etc.

Atividade: Corresponde a um passo do processo. Uma atividade pode ser manual ou automática. Atividades manuais não são gerenciadas pelo *workflow*. Atividades automáticas precisam ser executadas por humano e/ou sistema e não sofrem intervenções durante sua execução. Podem ser de dois tipos: itens de trabalho (ou em inglês, *work item*), onde tarefas são alocadas a algum participante do *workflow* e aplicações invocadas, geralmente ferramentas ou aplicações utilizadas como suporte à execução de uma determinada atividade.

Transições: As transições correspondem aos momentos durante a execução do processo no qual uma atividade completa sua execução e o controle passa para a próxima atividade. Dessa forma, as transições definem a ordem com que as atividades devem ser executadas. Em alguns casos podem ser condicionais, onde a próxima atividade a ser executada é determinada pelo resultado da avaliação, em tempo de execução, de uma determinada condição. A Figura 4 ilustra um processo composto por 8 atividades, com diferentes tipos de transições entre elas:

1. Atividade A: Partindo de A, em azul, há transições de subdivisão chamadas OR-SPLIT. Após a execução da atividade A é que o WfMS decide, dependendo de uma determinada

condição, qual das atividades será executada: B OU C. Somente uma dessas atividades é executada;

2. Para que o fluxo de execução do processo passe para a Atividade D, é necessário que B ou C terminem sua execução, por isso, dessas atividades partem transições do tipo junção chamadas OR-JOIN;

3. Após a execução da atividade D, há transições do tipo sub-divisão AND-SPLIT. Esse tipo de transição faz com que as atividades E, F e G sejam executadas simultaneamente;

4. A Atividade H, condicionada por uma junção do tipo AND-JOIN somente será executada quando E, F e G tiverem terminado sua execução;

Instância (de um processo ou atividade): Cada instância representa uma *thread*[1] diferente de execução de um processo ou atividade. Cada vez que um processo ou uma atividade é invocada, são criadas instâncias gerenciadas pelo WfMS. As instâncias de atividades podem ser invocadas de duas maneiras: seqüencialmente, onde somente uma atividade é instanciada por vez ou simultaneamente, onde duas ou mais atividades são instanciadas ao mesmo tempo e executadas em paralelo. As atividades que são executadas em paralelo, geralmente iniciam por um AND-Split e terminam com um AND-Join.

Participantes do *workfbw*: Esse termo geralmente se aplica a humanos, mas conceitualmente pode também representar máquinas. Os participantes são identificados diretamente durante a definição do processo e podem ser: humano, máquina, regra ou unidade organizacional.

WAPI: WAPI é uma abreviação para *Workflow APIs and Interchange Formats*, publicado pela WFMC. É constituído de especificações que permitem a comunicação entre diferentes componentes dos WfMS e aplicações externas. Assim, aplicações externas ao *workflow* podem ser invocadas de dentro do mesmo.

3.2 Tipos de *Workflows*

Embora as definições acima se refiram a processos de negócio, *workflow* não é somente aplicada a esse tipo de processo. Segundo Wainer *et al.* [4] *workflows* podem também ser classificados como: *workflows ad-hoc* e *workflows* científicos. Bolcer e Taylor [30] descrevem como primeiro propósito de *workflows ad-hoc* informar usuários dos problemas correntes e dos dados necessários para executar cada uma de suas tarefas. Esse tipo de *workflow* pode executar seu trabalho de uma maneira *ad-hoc* no qual as soluções e objetivos são alcançados utilizando diferentes ferramentas e os dados são imediatamente disponibilizados.

[1] *Threads* é a divisão de um programa em duas ou mais tarefas executando simultaneamente.

3.3 *Workflows* Científicos

Segundo Santos [31], *workflows* científicos são voltados para as aplicações científicas que demandam alto poder computacional de áreas de pesquisa como biologia, física, astronomia, geologia entre outras. *Workflows* científicos geralmente adquirem dados de diferentes experimentos, por exemplo: dados gerados de modelos computacionais ou obtidos após análises estatísticas sobre conjuntos de dados. Além disso, não podem ser completamente definidos antes que sejam iniciados. Assim, o resultado de tarefas que estão sendo executadas a cada momento é que decidem os próximos passos. Essa falta de conhecimento sobre o processo antes que o mesmo seja efetivamente executado implica em algumas características específicas na modelagem de *workflows* científicos. A principal é assumir que os modelos não são completos e mudam a todo momento (Wainer *et al.* [4]).

Por causa dessas características, *workflows* científicos e de negócio apresentam muitas diferenças. As principais apresentadas por Meyer *et al.* [32] e Weske *et al.* [33] , são:

- *Workflows* de negócio são modelados para atender a um processo relativamente fixo, porém no caso de *workflows* científicos a definição do *workflow* envolve a tomada de diversas decisões, análises e, muitas vezes, trabalho em equipe;
- Enquanto *workflows* de negócio são orientados pelo fluxo de controle das atividades, *workflows* científicos são orientados pelo fluxo de dados;
- *Workflows* de negócio requerem poucas mensagens de coordenação e troca de documentos e dados entre as atividades enquanto que *workflows* científicos utilizam muitos dados, geralmente derivados de diferentes fontes e nenhum documento é modificado;
- Para *workflows* científicos tanto as respostas positivas quanto as negativas precisam ser analisadas e por isso devem ser armazenadas;
- Em *workflows* de negócio o modelo desenvolvido não é alterado durante a execução do *workflow* devido aos resultados obtidos após a execução de cada etapa. Já a definição de *workflows* científicos é um processo dinâmico, influenciado muitas vezes por resultados obtidos durante a execução, gerando constantes mudanças no fluxo de execução.

As principais etapas envolvidas no desenvolvimento de um *workflow* científico, segundo Wainer *et al.* [4] são:

1. **Modelagem do *workfbw*:** o *workflow* é definido como um conjunto de atividades agrupadas, utilizando programas já existentes para executar essa etapa;
2. **Execução do *workfbw*:** a execução do *workflow* é feita geralmente por meio da execução de cada uma de suas atividades, na qual a saída de uma atividade corresponde à entrada

da próxima. São comuns de aparecerem em *workflows* científicos atividades como testes no fluxo de execução, desvios, restrições, tratamentos de erros, entre outras;

3. **Visualização dos resultados:** os resultados, tanto parciais quanto finais, podem ser visualizados e analisados, e ainda utilizados para a geração de ajustes e de novas re-execuções do *workflow*;

4. **Finalização do *workflow*:** pode-se considerar como último passo da modelagem de um processo utilizando *workflows* científicos o momento em que o experimento atingir o seu objetivo, produzindo resultados relevantes e, assim, algum tipo de conhecimento científico poderá ser inferido desses resultados.

Como as características do processo que desejávamos modelar e automatizar correspondem às características de *workflows* científicos, como por exemplo, a demanda de alto poder computacional, o uso de diferentes *softwares* e a manipulação de muitos e diferentes dados, optou-se por esse tipo de *workflow* para ser desenvolvido no presente trabalho.

3.4 Ferramentas Utilizadas Para Modelagem e Execução do *Workflow* Desenvolvido

Baseado nas etapas envolvidas no desenvolvimento de um *workflow* científico descritas na Seção3.3, a Figura 5 relaciona cada etapa com as ferramentas selecionadas para executá-las.

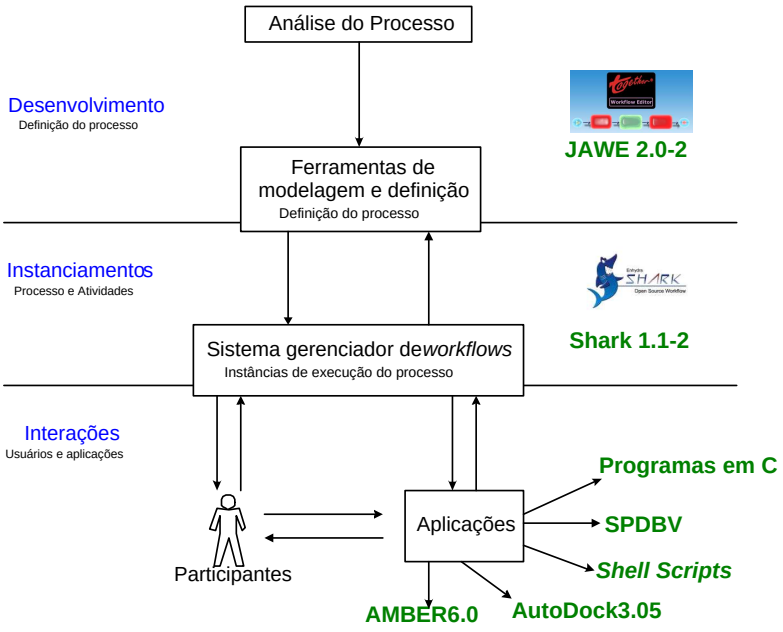


Figura 5 – Relação entre as etapas envolvidas no desenvolvimento de um *workflow* e as ferramentas selecionadas para executá-las.

Para modelar o *workflow* desenvolvido foi selecionada a ferramenta JAWE 2.0-2 [1] e para executá-lo, selecionou-se o *Enhydra Shark* 1.1-2 [34]. Os *softwares* AMBER6.0 [10], AutoDock3.05 [6], SPDBV [35] (*Swiss PDB Viewer*), Programas em C e *shell scripts* são aplicações que se comunicam com instâncias de execução do *workflow*.

As principais razões para essa escolha são:

- Tratam-se de ferramentas que são *softwares* livres cuja distribuição não tem custos;
- São ferramentas que recebem constante atualização e que os erros são corrigidos a cada nova versão;
- São *softwares* robustos e já utilizados em outros trabalhos científicos, como o trabalho apresentado por Barretto *et al.* [36];
- Os *softwares* AutoDock3.05 (utilizado na execução dos *dockings* e o AMBER6.0 (utilizado nas simulações por dinâmica molecular) executam em ambiente Linux assim como o Shark e JAWE, que funcionam corretamente nesse ambiente;
- São ferramentas cujo uso é relativamente simples, tanto para a modelagem quanto para a execução do *workflow* e que a execução de programas externos (como por exemplo, a execução de alguns programas do pacote do AutoDock3.05) pode ser feita utilizando funcionalidades oferecidas pelas próprias ferramentas, como o uso de ToolAgents [29].

3.4.1 Ferramenta JAWE 2.0-2

JAWE á uma ferramenta visual para a criação e gerenciamento de definições de processos. Seu arquivo de saída final é um arquivo XPDL (XML *Process Definition Language*) que pode ser interpretado em tempo de execução por diferentes WfMS [1]. A descrição de como se trabalha utilizando o JAWE segue as especificações da WFMC.

Segundo Mehta e Barter [1], utilizando a ferramenta JAWE é possível somente o desenvolvimento de *workflows* baseados em atividades (em inglês, *Activity-based workflow*) que são aqueles compostos de atividades a serem executadas a fim de alcançar um objetivo. Há os *workflows* baseados em entidades (em inglês, *Entity-based workflow*), que são aqueles cujo foco são conjuntos de documentos, e os estágios envolvidos para completá-los. Ex: publicação de documentos na Web.

Os principais componentes do JAWE são:

1. Pacote: Quando muitos processos compartilham as mesmas aplicações e são executados pelos mesmos participantes, esses podem ser reunidos em um pacote. Quando se deseja definir um novo processo, deve-se antes definir as características do pacote que conterá o processo. A Figura 6 mostra a tela principal do JAWE. Nessa tela está aberto um

pacote que contém diversos processos, entre eles os que serão utilizados como exemplo de modelagem do JAWE. À esquerda pode ser visto a descrição do pacote chamado “Test” e todos os processos contidos nele. Em destaque aparece o processo “Do math”;

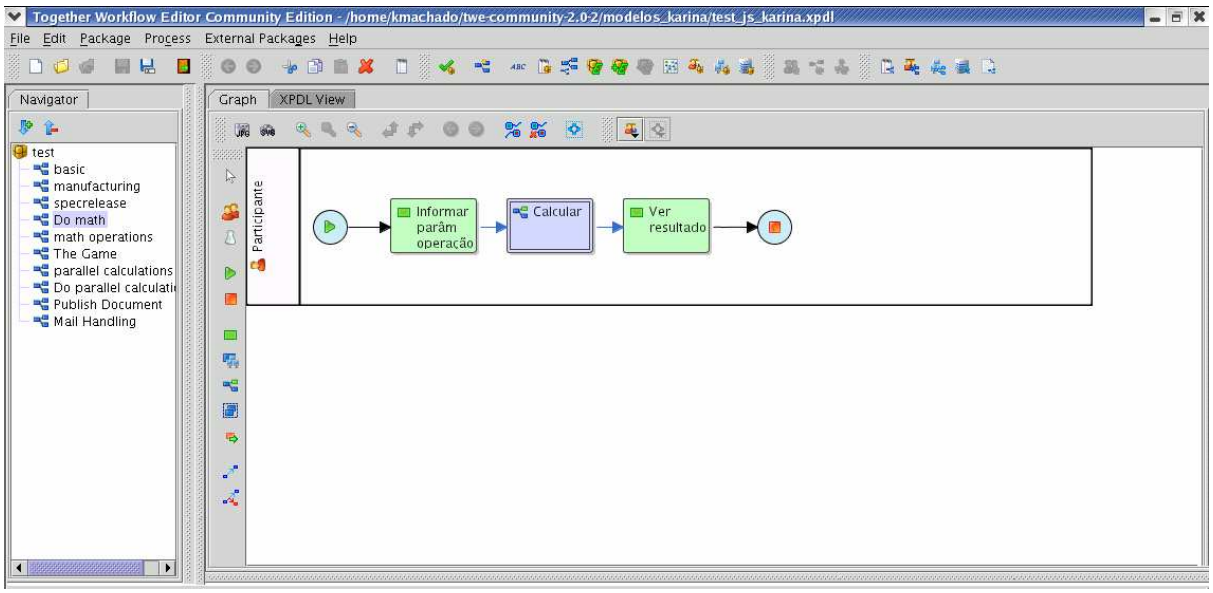


Figura 6 – Tela principal do JAWE 2.0-2 mostrando um processo composto por 3 atividades, sendo uma delas um *subflow*.

2. Processo: A definição de processo já foi feita na Seção 3.1. O processo definido deve conter a definição de cada uma das atividades, transações, aplicações envolvidas na execução do processo. O processo que aparece em destaque na Figura 6 é chamado “Do math” e serve para executar uma operação matemática. Esse processo será executado por algum participante do processo (como pode ser visto na figura) e é composto por 3 atividades: “Informar parâm operação”, “Calcular” e “Ver resultado”.

3. Atividades: Na modelagem de processos utilizando o JAWE, os diferentes tipos de atividades do processo são representadas por retângulos de diferentes cores:

   - As atividades em verde-escuro são aquelas que necessitam da intervenção humana para serem executadas (no JAWE, um pouco diferente do conceito apresentado pela WFMC, atividades manuais são aquelas executadas manualmente pelo usuário, que fazem parte do *workflow*, mas precisam de intervenção humana para terminar sua execução);
   - As caixas em roxo representam *subflows*, sub-processos pertencentes ao mesmo pacote do processo que os utiliza e compostos por atividades, transições e aplicações próprias (não precisando estar relacionados com o processo que o utiliza). O *subflow* mostrado na Figura 6 é descrito na Figura 7. Esse *subflow* é utilizado para efetivamente executar a operação matemática informada pelo usuário na atividade

anterior. Dependendo do tipo de operação uma das atividades “somar”, “subtrair”, “multiplicar” ou “dividir” é executada;

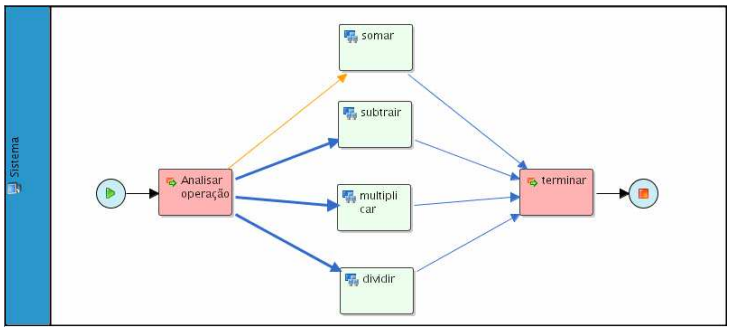


Figura 7 – Exemplo de processo modelado no JAWE 2.0-2. Esse processo verifica o tipo de operação informada (soma, subtração, multiplicação ou divisão) e executa a mesma.

- As atividades verde-claro são aquelas executadas pelo sistema, no qual não há a necessidade de intervenção humana durante a execução. Essa atividades sempre estão relacionadas com aplicações externas que são definidas no JAWE e executadas pelo WfMS por meio do uso de WAPI. Exemplos dessas atividades são as atividades “somar”, “subtrair”, “multiplicar” ou “dividir” que compõem o *subflow* descrito na Figura 7. Cada uma dessas atividades executa uma função escrita em JAVA que realiza a operação correta;
- As atividades em rosa que aparecem no *subflow* são aquelas utilizadas pare direcionar o fluxo de execução, nenhuma ação é efetivamente executada por elas.

4. Transições: As transições correspondem às ligações entre as atividades. Os diferentes tipos de transições possíveis de serem modeladas no JAWE são as mesmas descritas pela WFMC: transições seqüenciais e interativas e as do tipo AND/OR SPLIT ou AND/OR JOIN. O processo descrito no *subflow* da Figura 7 contém a atividade “Analisar operação” de onde partem 4 transições do tipo OR-SPLIT, que são transições condicionais, que executam corretamente a operação informada pelo usuário;

5. Participantes: No processo que está sendo utilizado como exemplo, há dois tipos de participantes envolvidos: No processo “Do math” há o “Participante”, que deverá interagir com o processo e no *subflow* há o “Sistema”, pois todas as atividades são executadas automaticamente pelo sistema sem a necessidade de intervenção humana;

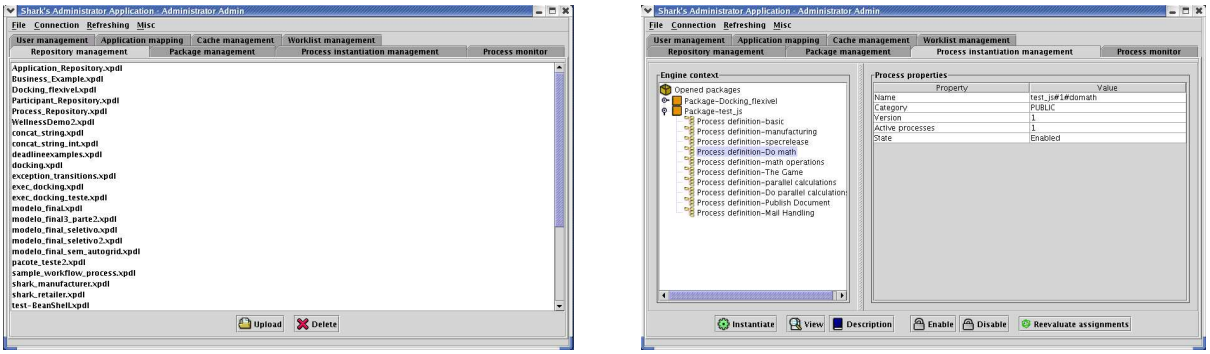
3.4.2 Ferramenta *Enhydra Shark* 1.1-2

O *Enhydra Shark* é um *workflow engine* escrito na linguagem JAVA. Por ter um código aberto, pode ser facilmente extensível. Seu padrão de implementação, assim como o JAWE,

são as especificações WFMC utilizando XPDL (sem nenhuma extensão que seja proprietária) para definição de seus processos e os WFMC *ToolAgents* APIs para a execução de aplicações de dentro do *workflow* [37]. O *Enhydra Shark* pode ser utilizado tanto de forma independente, como de dentro de outra aplicação escrita em JAVA (onde somente algumas funcionalidades podem ser utilizadas em separado), ou ainda como aplicação em *Web Services*.

Para utilizar o *Enhydra Shark* o usuário deve se logar no sistema [37]. Inicialmente pode entrar como administrador, cuja senha padrão é informada nos manuais do programa. Após, para cada um dos processos a serem executados, ele associa os usuários cadastrados no *Shark* aos participantes descritos na definição do processo e, assim, cada usuário que se logar ao sistema receberá em seu *work list* as atividades que deve executar.

Após se logar, a tela inicial do *Enhydra Shark* é a tela mostrada na Figura 8a, na qual estão listados todos os processos (que devem ser arquivos XPDL) já gravados em um repositório de dados da ferramenta. Se for um processo novo, a primeira atividade será fazer *up-load* (carregamento) do mesmo para esse repositório. Após, um dos processos do repositório é lido e pode ser instanciado para ser executado (tela mostrada na Figura 8b).



(a) (b)

Figura 8 – (a) Tela inicial do Shark 1.1-2, após o usuário ter conectado corretamente. Nessa tela o usuário atualiza o processo que deseja executar. (b) Após, o usuário instância o processo que deseja executar.

Após o usuário ter instanciado o processo, os seguintes passos durante a execução do processo “Do math” explicado na Seção anterior, são:

1. O processo inicia sua execução, a primeira atividade está ativa e pronta para se executada (Figura 9a);

2. O usuário indica se deseja atualizar os parâmetros necessários para execução da próxima atividade (Figura 9b);

3. Se o usuário optar por “*YES*” na atividade anterior, os dois valores a serem utilizados na operação e o tipo de operação a ser executada devem ser informados e essa atividade termina sua execução (Figura 9c);

4. A próxima atividade é o *subflow* executado pelo sistema (conforme mostra a Figura 7), logo ele inicia e termina sem intervenção do usuário, ficando a próxima atividade disponível na lista de atividades (*work list*) do usuário em Figura 9d;

5. A atividade é executada pelo usuário e o resultado da operação pode ser visto (Figura 9e);

6. Por fim, a Figura 9f mostra o processo em seu estado final, após todas as atividades terem sido executadas.

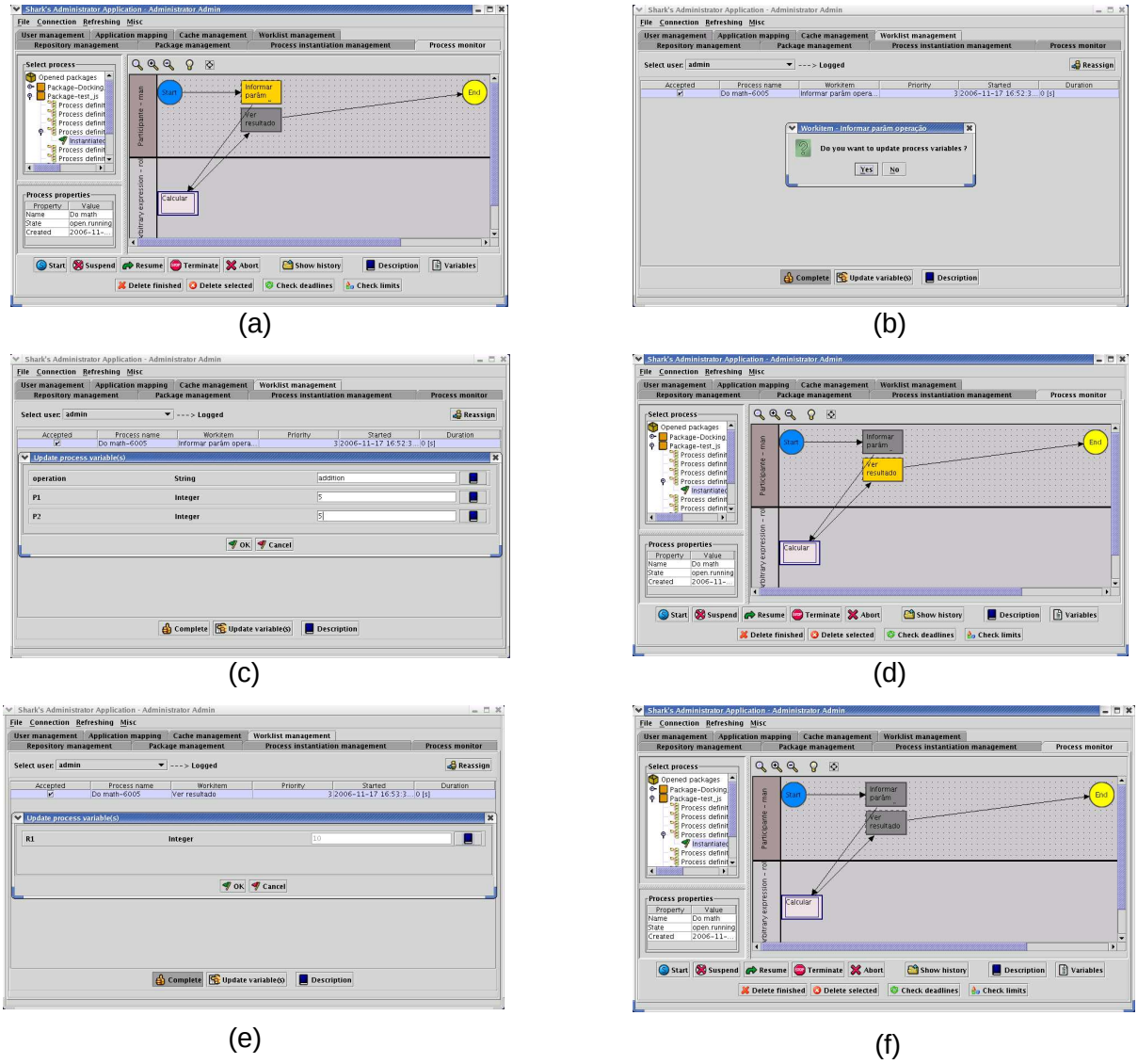


Figura 9 – (a) Processo no início de sua execução. A primeira atividade é executada e as variáveis necessárias são atualizadas em (b) e (c). Em (d) a última atividade fica disponível e em (e) a mesma é executada (a segunda atividade não aparece porque é executada pelo sistema). (f) O processo no final de sua execução, no qual nenhuma atividade está disponível para ser executada.

3.5 Trabalhos Relacionados: *Worlflows* Científicos Aplicados à Bioinformática

A aplicação de *workflows* científicos em Bioinformática tem aumentado muito nos últimos anos, especialmente em *drug design*, como mostra a entrevista de Watson [12]. Alguns exemplos do uso de *workflows* científicos em Bioinformática podem ser vistos nos trabalhos de Weske *et al.* [33] e Ludäscher *et al.* [38].

O trabalho apresentado por Ludäscher *et al.* [38] descreve uma arquitetura que foi desenvolvida com o objetivo de fornecer suporte ao desenvolvimento de *workflows* científicos. Utilizando essa arquitetura, Ludäscher *et al.* [38] modelou o processo de identificação de promotores de genes, gerando o que chamou de PIW - *Promoter Identification Workflow*. O trabalho de Georgakopoulos *et al.* [39] também utilizou esse processo de identificação de promotores de genes para validar o uso da arquitetura de desenvolvimento *workflows* científicos modelada em seu trabalho.

Biólogos estão freqüentemente pesquisando como um organismo responde a mudanças no seu ambiente, expressas por meio do comportamento dos seus genes. Por exemplo, descobrir se o nível de expressão de um conjunto de genes diminui muito na presença de radiação. A tecnologia de *microarrays* de DNA é utilizada para determinar o nível de expressão de um conjunto de genes. Primeiro, uma amostra do DNA é exposta às mudanças no ambiente que causam transcrições de certos genes. Esses genes são marcados com cores fluorescentes como pode ser visto no passo 1 da Figura 10 (adaptada de Georgakopoulos *et al.* [39]). Quanto maior o número de pontos fluorescentes, mais alto o nível de expressão do gene. A seguir, um biólogo seleciona um subconjunto de genes que seja parecido com os genes que obtiveram um maior nível de expressão. A partir desse momento o PIW é executado para identificação de promotores de genes parecidos com os genes selecionados.

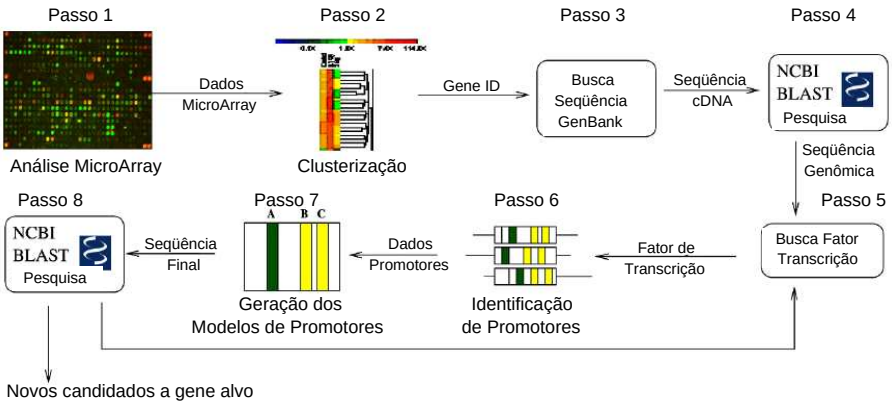


Figura 10 – *Workflow* para identificação de promotores de genes.

Resumidamente, o processo inicia pela determinação do conjunto de genes que vai se buscar

por genes similares (Passos 1 e 2) por meio do uso de algoritmos de clusterização. Esses genes são identificados pelo GeneID, no qual, para cada GeneID, procura-se genes (utilizando um banco de dados como o GenBank [40] do NCBI) com regiões promotoras similares a esses (Passo 3) (esse busca por genes com regiões promotoras similares é realizada utilizando-se uma ferramenta como o BLAST [41](Passo 4) que encontra seqüências similares). Nos passos 4 e 5 é identificado o fator de transcrição dentro de cada região promotora. A partir desses dados, é gerado um conjunto de modelos de promotores para cada gene. Por fim, são procurados em um banco de dados de genes, aqueles que contenham seqüências similares a dos modelos de promotores. Todo o processo está ilustrado na Figura 10 (Ludäscher *et al.* [38]).

Para Ludäscher *et al.* [38], a partir de uma análise nesse modelo já se pode extrair algumas características típicas de *workflows* científicos, como por exemplo, o uso de diversas bases de dados já existentes, assim como de ferramentas já desenvolvidas que precisam ser combinadas em uma certa seqüência para que os passos desse processo sejam executados corretamente.

O Estudo de Caso apresentado por Weske *et al.* [33] trata-se de um *workflow* para modelar o processo de agrupamento de fragmentos de DNA utilizado no melhoramento do seqüenciamento de genomas. Esse estudo foi utilizado para validar uma ferramenta desenvolvida para a definição e execução de *workflows* científicos e pode ser resumido da seguinte maneira: sabendo-se que toda a informação genética de um organismo é armazenada em seqüências de nucleotídeos (sendo uma das formas o DNA), encontrar e interpretar uma seqüência de bases de um organismo é uma tarefa importante e fundamental em biologia molecular. Atualmente, pequenas seqüências de DNA podem ser geradas semi-automaticamente, utilizando recursos específicos. Essas seqüências são conhecidas como fragmentos de DNA. Entre os fatores que dificultam o estudo sobre seqüências de DNA está o fato de existirem muitos erros dentro das seqüências. Por causa desses erros, uma seqüência de bases fornecida como resultado de um serviço de seqüenciamento não está totalmente correta. Uma tarefa dos cientistas é corrigir esses erros e assim produzir um conjunto de dados com um maior nível de qualidade. Essa tarefa é executada por meio de procedimentos de validação pré-definidos com a ajuda do conhecimento de especialistas.

De acordo com Weske *et al.* [33], pode-se modelar o sub-processo de agrupamento de fragmentos, último passo no processo de seqüenciamento de DNA, utilizando-se *workflows*. O *workflow* científico que representa esse sub-processo pode ser visto na Figura 11. A tarefa "Geração de fragmentos inicial" especifica os experimentos científicos que irão extrair informações da seqüência de uma dada molécula (isso porque há diversas maneiras dessa seqüência ser obtida) e gera uma série de fragmentos. No passo seguinte, os fragmentos gerados são unidos para formar a seqüência que se deseja obter ao final do experimento. O resultado dessa atividade é avaliado por um humano. Se o humano validá-la, a análise da seqüência pode ser feita. Caso contrário, novos fragmentos devem ser gerados e unidos para formar diferentes seqüências que se aproximem da seqüência a qual se objetiva chegar.

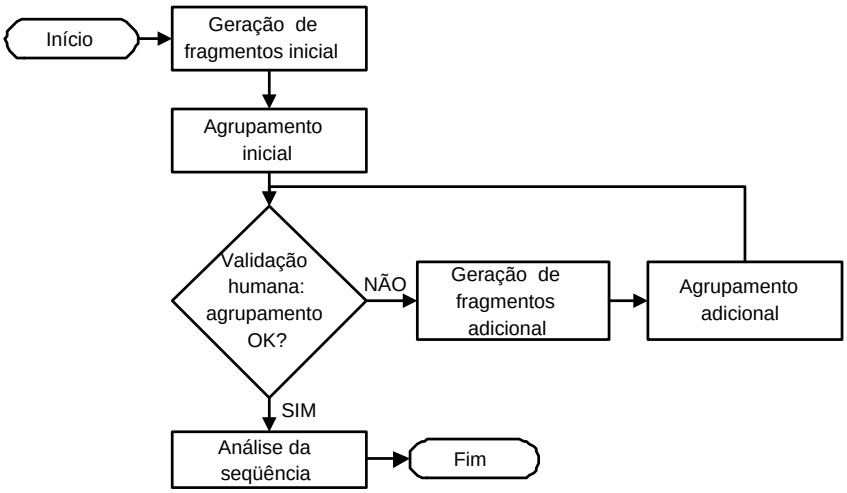


Figura 11 – Agrupamento de fragmentos de DNA visto como um *workflow* científico.

3.6 Considerações Finais

Nesse capítulo são descritos os conceitos básicos da área de *workflow*. São abordadas as principais características de *workflows* científicos assim como alguns trabalhos relacionados com Bioinformática, onde *workflows* foram utilizados na resolução de problemas. As ferramentas utilizadas no desenvolvimento do presente trabalho são também descritas nesse capítulo.

O próximo capítulo apresenta o trabalho desenvolvido, descrevendo o *workflow* científico que foi modelado a fim de se automatizar o processo de desenvolvimento de fármacos assistido por computador, considerando a flexibilidade do receptor. Cada uma das etapas envolvidas e os programas desenvolvidos para executá-las são explicados nesse capítulo.

4 Resultados: Desenvolvimento do Workflow

Neste capítulo é descrito o *workflow* desenvolvido. Inicialmente, justifica-se o uso de *workflows* científicos para a solução do problema de consideração da flexibilidade de proteínas em experimentos de *docking* molecular. Após, a modelagem completa desenvolvida é descrita, onde cada uma das etapas envolvidas no processo são explicadas detalhadamente. Junto à explicação de cada uma dessas etapas é descrita a maneira como as mesmas eram executadas antes do desenvolvimento desse trabalho.

A consideração da flexibilidade da macromolécula em experimentos de *docking* molecular não é trivial. Enquanto a flexibilidade do ligante pode ser facilmente levada em conta nestes experimentos, a flexibilidade do receptor é difícil de ser tratada devido à complexidade do sistema (muitos graus de liberdade envolvidos), que é ocasionada pelo seu número elevado de átomos. Como alternativa, atualmente experimentos de *docking* têm sido executados utilizando, cada um deles, uma estrutura instantânea da proteína (*snapshot*) [9], gerada a partir de simulações por dinâmica molecular.

Na maioria das vezes esse processo é executado manualmente ou com a ajuda de *shell scripts*. Porém, se executado dessa forma, têm-se problemas para definir a ordem correta em que as etapas deverão ser executadas, executar o processo utilizando parâmetros diferentes, monitorar a execução do mesmo, etc. Por esses motivos e com base na idéia da funcionalidade de *workflows* científicos, optou-se por utilizá-los na definição e execução de todas as etapas do processo de desenvolvimento de fármacos assistido por computador, considerando a flexibilidade da macromolécula.

Como foi descrito na Seção 3.4, para a modelagem do *workflow* escolheu-se a ferramenta JAWE 2.0-2 [1] e para a sua execução selecionou-se o *Enhydra Shark* 1.1-2 [34].

O modelo final do *workflow* desenvolvido está ilustrado na Figura 12. Nesse modelo, cada caixa corresponde a uma atividade executada no processo. As cores das caixas mostram o tipo de execução daquela atividade (como foi explicado na Seção 3.4.1).

4.1 Etapa Prévia - Dinâmica Molecular

A etapa prévia consiste em executar as simulações por dinâmica molecular do sistema proteína-ligante ou apenas proteína. Como já foi explicado na Seção 2.1, a simulação por dinâmica molecular, neste caso, visa simular a flexibilidade natural de proteínas.

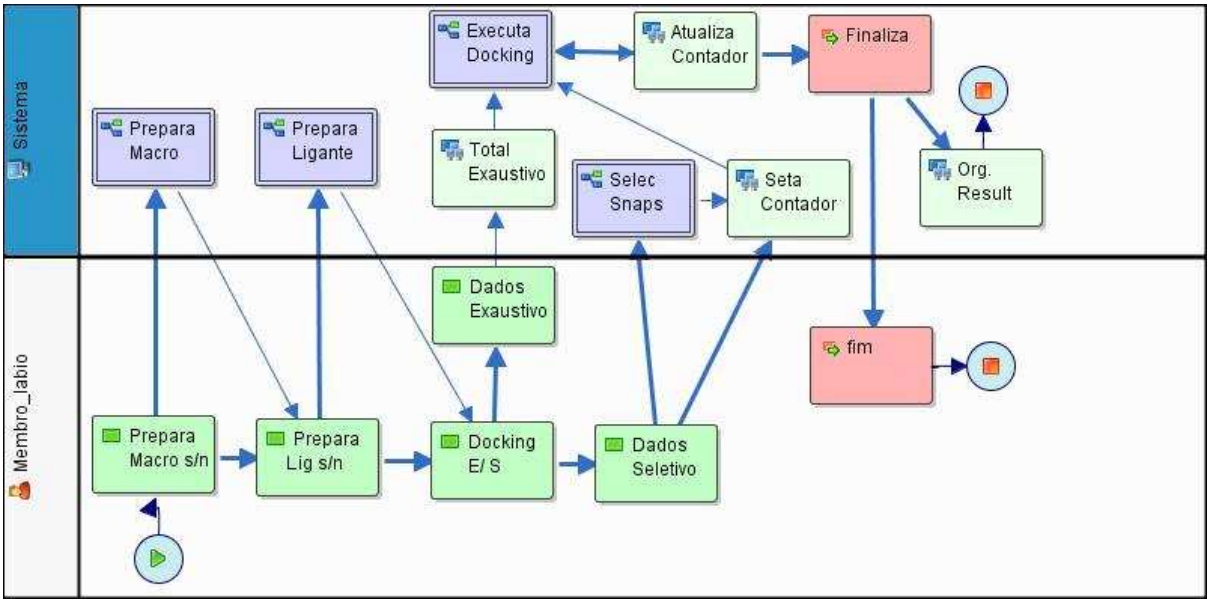


Figura 12 – Modelagem final do processo de desenvolvimento de fármacos assistido por computador considerando o receptor flexível. Esse modelo foi desenvolvido utilizando o *software* JAWE 2.0-2 [1].

Em uma simulação por dinâmica molecular, uma seqüência de *snapshots* é gerada e armazenada, correspondendo às diferentes conformações que determinada proteína apresenta em função do tempo. Essa etapa não está incluída no *workflow*, pois somente precisa ser executada uma vez para cada proteína e não haverá a necessidade de executá-la mais vezes com parâmetros diferentes.

Neste trabalho o sistema InhA-NADH foi simulado por dinâmica molecular utilizando o programa AMBER 6.0 [10]. Essa simulação foi realizada em um trabalho de doutorado desenvolvido no LABIO [28]. Neste trabalho observou-se que a enzima InhA da *Mycobacterium tuberculosis* pode ser considerada uma molécula flexível.

Essa etapa do processo é a única cuja maneira de execução não foi alterada durante o desenvolvimento do presente trabalho. Após a execução da simulação por dinâmica molecular, já de posse de todas as possíveis conformações que a macromolécula pode assumir em função do tempo, todas as demais etapas podem ser executadas pelo *workflow* desenvolvido.

4.2 Primeira Etapa - Preparação da Macromolécula

A primeira etapa do processo de CADD, considerando a flexibilidade da macromolécula, consiste na preparação da mesma para ser utilizada nos experimentos de *docking*. Para isso, é necessária a execução de duas atividades: processamentos dos arquivos de saída da simulação por dinâmica molecular, gerando arquivos compatíveis com os softwares de *docking* e consideração de apenas parte dos *snapshots* da macromolécula gerados pela DM (a simulação por DM é realizada utilizando-se intervalos de tempo muito pequenos entre a geração de dois *snapshots*

consecutivos. Como não há a necessidade de utilização de *snapshots* tão próximos em experimentos de *docking*, intervalos de tempo maiores devem ser considerados para a seleção das conformações da macromolécula a serem consideradas para o *docking*).

No *workflow* desenvolvido, como pode ser visto na Figura 12, antes da execução dessa etapa, o usuário é questionado sobre a execução ou não da preparação dos arquivos da macromolécula a partir dos resultados da DM. Pois, uma vez que os arquivos da dinâmica já tenham sido processados e considerados para determinada macromolécula (ou para um determinado trecho da simulação da dinâmica molecular da mesma), eles não precisam ser refeitos, tornando dispensável à execução desta etapa.

Um *subflow* é responsável pela execução dessa etapa no *workflow* conforme mostra a Figura 13. O *subflow* compreende duas atividades: “Execução PTRAJ” e “Remove PDBs”. Essas duas atividades estão descritas logo a seguir.

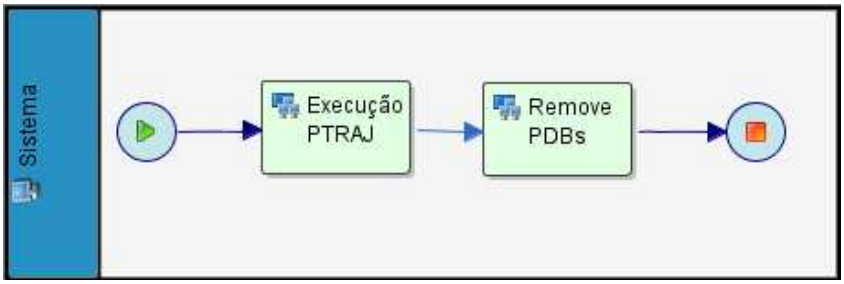


Figura 13 – *Subflow* para executar a etapa de preparação da macromolécula.

4.2.1 Execução do PTRAJ

Nessa etapa, os arquivos da dinâmica que representam os *snapshots* da macromolécula gerados anteriormente (etapa prévia) são transformados em um formato que poderá ser utilizado futuramente nos experimentos de *docking*, ou seja, são transformados em arquivos do tipo PDB da macromolécula com desconsideração do ligante, dos contra-íons e das moléculas de água presentes nos arquivos da dinâmica molecular. Para realizar essa tarefa é utilizado um *software* do próprio pacote AMBER6.0 [10], chamado PTRAJ.

O PTRAJ é um programa utilizado para processar e analisar conjuntos de coordenadas 3D lidas de uma série de arquivos de coordenadas de entrada. Para cada conjunto de coordenadas lido, uma seqüência de ações pode ser executada (em uma ordem que deve ser especificada) de acordo com configurações pré-estabelecidas. Após o processamento de toda a entrada, arquivos de trajetória podem ser escritos, como por exemplo, no formato PDB (Case *et al.* [10]).

A Figura 14 mostra trechos dos arquivos utilizados pelo PTRAJ e do arquivo resultante.

A Figura 14a mostra um trecho de um arquivo de saída da simulação por dinâmica molecular composto por um conjunto de coordenadas em seqüência, no qual não é possível determinar

```
INHA + NADH de M. tubewrculosis [1ENY,27-JAN-1995] Residues 1-268
  64.433  26.825 128.851  65.342  26.818 129.291  64.348  27.752 128.459  63.721
  26.799 129.567  64.230  25.744 127.845  65.119  25.789 127.216  64.261  24.354
 128.504  63.380  24.273 129.141  64.290  23.475 127.859  65.158  24.297 129.120
  62.919  25.939 127.160  61.875  26.090 127.794  62.964  25.857 125.847  63.851
```

(a)

```
INHA + NADH de M. tubewrculosis [1ENY,27-JAN-1995] Residues 1-268
 31481    23 29437  2080  4635  2833  7940  3718     0     0 59005  9407
  2080  2833  3718    79   196   143    39     1     0     0     0     0
     0     0     0     1    71     0
N   H1  H2  H3  CA  HA  CB  HB1 HB2 HB3 C   O   N   H   CA  HA2 HA3 C   O   N
H   CA  HA  CB  HB2 HB3 CG  HG  CD1 HD11HD12HD13CD2 HD21HD22HD23C   O   N   H
```

(b)

```
ATOM      1  N   ALA     1      64.433  26.825 128.851  0.00  0.00
ATOM      2  H1  ALA     1      65.342  26.818 129.291  0.00  0.00
ATOM      3  H2  ALA     1      64.348  27.752 128.459  0.00  0.00
ATOM      4  H3  ALA     1      63.721  26.799 129.567  0.00  0.00
ATOM      5  CA  ALA     1      64.230  25.744 127.845  0.00  0.00
```

(c)

Figura 14 – (a) Trecho de um arquivo de saída da simulação por dinâmica molecular. (b) Trecho do arquivo de topologia utilizado pelo PTRAJ. (c) Exemplo de arquivo PDB gerado após a execução do PTRAJ.

que coordenada corresponde a que átomo. Esse arquivo é organizado dessa maneira para economizar espaço de armazenamento, já que para análises feitas pelo próprio AMBER6.0, esses dados não são necessários, e se forem, serão buscados automaticamente. Na Figura 14a as coordenadas x, y e z do primeiro átomo da primeira conformação de um determinado conjunto de conformações estão marcadas em rosa, as coordenadas do segundo átomo estão em amarelo, do terceiro em verde, e assim por diante, até o último átomo da última conformação de um determinado conjunto de conformações.

A Figura 14b corresponde a um trecho do arquivo de topologia utilizado pelo PTRAJ para conseguir transformar adequadamente os arquivos de entrada em um determinado formato de saída. Esse arquivo contém a listagem com os nomes de cada átomo, de cada resíduo, o total de átomos da proteína (que corresponde ao total de átomos de cada conformação), entre outras características. Na Figura 14b está marcado em azul o nome dos 5 primeiros átomos da proteína.

A Figura 14c mostra um exemplo de arquivo PDB gerado após a execução do PTRAJ. Esse arquivo contém a descrição de cada átomo de uma determinada conformação da proteína e pode ser utilizado nos experimentos de *docking*.

Para executar o PTRAJ é necessário:

1. Ler o arquivo de topologia (Figura 14b): esse arquivo serve como guia na separação dos arquivos de coordenadas, organizando corretamente os dados sobre cada uma das conformações da proteína em função do tempo;

2. Determinar a lista de arquivos de entrada: utilizando-se o comando $trajin$, especifica-se o nome de cada um dos arquivos de entrada. Um exemplo de formato do arquivo de entrada pode ser visto na Figura 14a;

3. Opcionalmente especificar o arquivo de saída: com o comando $trajout$. Esses arquivos de saída podem ser de diferentes formatos. No caso do presente trabalho, utiliza-se sempre como formato para o arquivo de saída o formato PDB (Figura 14c), pois esse formato pode ser utilizado nos experimentos de *docking* molecular;

4. Especificar uma lista de ações a serem executadas com cada arquivo de entrada. Existem diversas possibilidades de ações a serem executadas, como por exemplo: *center* (especifica onde deve ser posicionado o centro de massa de cada estrutura 3D), *strip* (especifica quais átomos ou moléculas do sistema molecular que foram utilizados na simulação, mas que não se deseja que apareçam nos arquivos de saída, por exemplo, moléculas de água), etc.

Antes do desenvolvimento do *workflow* havia um *shell script* com os comandos de execução do PTRAJ. Um exemplo desse *shell script* está na Figura 15 no qual estão listados os comandos contidos no *shell script* que são: os comandos de entrada (arquivos da dinâmica de 0 a 200 ps em 4 pacotes de 50 ps), o nome e formato dos arquivos de saída (nome: mdcp.pdb, formato: PDB), os comandos a serem executados (*center*, *image* e *strip*). Cada vez que alguns desses parâmetros tivessem que ser alterados, o *shell script* precisava ser reeditado. Se fosse necessário incluir mais arquivos de entrada, os comandos de $trajin$ deveriam ser manualmente adicionados.

```
#!/bin/csh -f
trajin  ~/INHA_NADH/DINAM/CRD/mdcp_0050ps.crd.gz
trajin  ~/INHA_NADH/DINAM/CRD/mdcp_0100ps.crd.gz
trajin  ~/INHA_NADH/DINAM/CRD/mdcp_0150ps.crd.gz
trajin  ~/INHA_NADH/DINAM/CRD/mdcp_0200ps.crd.gz
trajout mdcp.pdb pdb nobox
center :1-268 mass origin
image origin center
strip :269-9407
go
```

Figura 15 – Exemplo de execução do PTRAJ.

Para execução dessa etapa pelo *workflow* foi desenvolvido um programa em linguagem C. Esse programa é executado por meio de um aplicativo oferecido pelo *software Shark* 1.1-2 [34], que permite a execução de programas externos dentro do *worflow*.

O programa desenvolvido solicita ao usuário as seguintes informações:

- Localização dos arquivos gerados na etapa de simulação por DM;
- Localização dos parâmetros do PTRAJ;
- Intervalos de tempo que devem ser considerados no experimento;
- Tamanho dos pacotes em que os arquivos resultantes de simulação por dinâmica molecular foram agrupados. Normalmente esse tamanho é de 50 ps, ou seja, a cada 50 ps de simulação, os *snapshots* gerados são agrupados em um arquivo do tipo CRD;

- Resíduos de aminoácidos que devem ser considerados;
- Nome dos arquivos de entrada e saída;
- Informações sobre como gerar a estrutura média da proteína, por exemplo, que intervalo de tempo da simulação por DM que deve ser considerado para isso.

Com essas informações, o programa desenvolvido gera um *shell script* que efetivamente executa o PTRAJ utilizando os parâmetros gerados com base nas informações do usuário. Um exemplo dos parâmetros informados para cada um dos itens são mostrados na Tabela 1:

Tabela 1 – Exemplo de parâmetros de execução do PTRAJ.

Parâmetro	Valor
Localização dos arquivos da dinâmica	/DINAM/CRD/
Localização dos parâmetros do ptraj	/DINAM/parm_new_02.top
Intervalo de tempo que deve ser considerado no experimento	Inicio = 50 ps Fim = 3100 ps
Tamanho dos pacotes	50 (50 em 50 ps)
Resíduos que devem ser considerados	1-268
Nome dos arquivos de entrada e saída	Entrada: mdcp Saída: mdcp.pdb

Sendo assim, da forma como essa etapa está sendo realizada atualmente, o processo não fica fixo em um intervalo de tempo a ser considerado ou a um nome de arquivo de entrada ou saída. Antes do desenvolvimento desse trabalho qualquer mudança nos parâmetros de execução do PTRAJ deveria ser feita editando manualmente um *shell script* e procurando nele os parâmetros que fossem necessários ser alterados. Da mesma forma, a adição de novos trechos da dinâmica a serem considerados como entrada do PTRAJ é feita automaticamente utilizando somente os valores do intervalo de tempo a ser considerado, não necessitando mais da inclusão manual de cada um desses comandos no *shell script*.

A definição dos arquivos PDB da estrutura média da proteína, utilizada posteriormente para determinar a posição inicial do ligante nos experimentos de *docking*, pode ser calculada pelo PTRAJ. Apesar de poder calcular-se a estrutura média da proteína para qualquer intervalo de tempo que se determine, estipulou-se que esta estrutura média seria calculada sobre a chamada fase de produção da simulação por DM[1].

Todos os dados informados nessa etapa são necessários nas etapas seguintes, e por isso são armazenados em um arquivo de configurações, de forma que o usuário não precisará informá-los nem editá-los posteriormente durante o restante do processo.

[1]Fase de produção corresponde à etapa da simulação na qual o sistema já está equilibrado.

4.2.2 Considerando Arquivos PDBs da Macromolécula para Serem Utilizados nos Experimentos de *Docking*

Considerando que os parâmetros informados ao processo são os mostrados na Tabela 1, cada um dos arquivos do tipo *crd.gz*, utilizados como entrada no PTRAJ, correspondem a um arquivo compactado que compreende um pacote de tempo de 50 ps da simulação. Como durante a simulação por dinâmica molecular a cada 0.5 ps é gerado um *snapshot* da proteína, ao final de 3100 ps têm-se um total de 6200 *snapshots* da proteína e por conseqüência, 6200 arquivos PDBs gerados na execução do PTRAJ (se forem utilizados na etapa anterior os parâmetros da Tabela 1, no qual o início do experimento de *docking* deve ser o primeiro pacote de 50 ps e o final o último pacote, de 3100 ps).

Porém não é necessário executar experimentos de *dockings* de estruturas tão consecutivas (a conformação da estrutura 3D não muda muito em intervalos tão curtos de 0.5 ps). Assim, somente metade dos PDBs serão considerados nos experimentos de *docking*, por exemplo, aqueles PDBs cujo número seja par (e correspondem a um tempo de simulação inteiro, 1 ps, 2 ps, e assim por diante), sendo a outra metade, a dos PDBs ímpares (que correspondem às conformações da proteína nos tempos de 0.5 ps, 1.5 ps, 2.5 ps, etc.), desconsiderada para poupar processamento. Assim, os PDBs serão considerados de 1 em 1 ps e não de 0.5 em 0.5 ps.

Para executar essa etapa foi desenvolvido um programa que desconsidera metade dos arquivos PDBs gerados, removendo os arquivos PDBs do diretório que o *workflow* utiliza como fonte de dados, usando para isso os parâmetros que foram armazenados na etapa de Execução do PTRAJ. Assim o usuário não precisa informar novamente qual é o trecho de simulação a ser considerado, o nome dos arquivos de entrada e saída, etc.

Antes do desenvolvimento do *workflow*, essa etapa não era executada. Essa consideração de somente metade dos arquivos PDBs era feita juntamente com a execução dos experimentos de *docking*. Dessa maneira, muitos arquivos PDBs que não eram utilizados ficavam ocupando espaço em disco, aumentando também a chance de *dockings* desnecessários serem executados.

4.3 Segunda Etapa - Preparação do Ligante

Durante a segunda etapa do processo é feita a preparação do ligante que será utilizado durante os experimentos de *docking*. Assim como na preparação da macromolécula, essa etapa também compreende a execução de duas atividades: a atividade intitulada “Editout - SPDBV” na qual o usuário posiciona o ligante em sua posição de início no *docking*, e a atividade “Gera Mol2 Ligante” em que o arquivo PDB do ligante, posicionado corretamente onde se deseja iniciar o *docking*, recebe as cargas elétricas parciais correspondes de cada átomo. Essa etapa foi modelada utilizando o *subflow* mostrado na Figura 16.

Assim como acontece antes da execução da preparação da macromolécula, o usuário deve informar se esse *subflow* deve, ou não, ser executado, pois uma vez que o ligante já tenha sido preparado para o *docking* essa etapa torna-se dispensável de ser executada novamente.

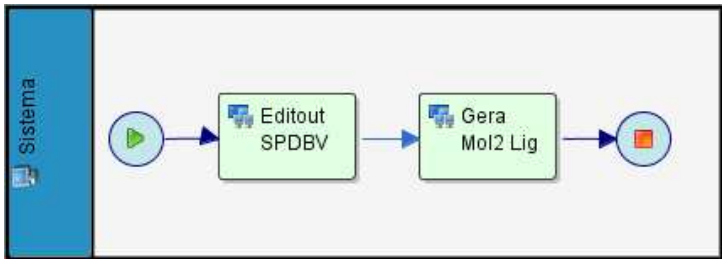


Figura 16 – *Subflow* para executar a etapa de preparação do ligante para o experimento de *docking*.

4.3.1 Editout - SPDBV

Essa etapa realiza duas operações descritas a seguir:

1. Um programa desenvolvido em linguagem C é executado pelo *workflow* para separar o arquivo *Editout.pdb*. Esse arquivo foi preparado para a etapa de execução da dinâmica molecular e contém as coordenadas e cargas da proteína, do ligante, dos contra-íons, e das moléculas de água do sistema molecular. O programa então separa esse arquivo em:

   - *Editout-onlyP.pdb* - contém somente informações da proteína;
   - *Ligante.pdb* - contém somente informações do ligante.

   Durante a execução desse programa pelo *workflow*, o usuário deve informar a localização do arquivo *Editout.pdb* e o nome do ligante (tornando o processo mais flexível no caso da necessidade de se executar o processo com diferentes proteínas e/ou diferentes ligantes). Assim, o programa separa corretamente os átomos da proteína e do ligante, não utilizando o restante dos átomos do sistema molecular.

   Antes do desenvolvimento desse programa, essa etapa era executada manualmente, editando o arquivo o *Editout.pdb*, recortando cada trecho e armazenando nos arquivos correspondentes.

   Pode acontecer das cargas da proteína e/ou do ligante não serem buscadas do arquivo *Editout.pdb*. Nesse caso, basta que o usuário não preencha a localização do arquivo, que o programa não executa essa atividade.

2. A segunda tarefa executada durante essa etapa consiste em editar o arquivo PDB do ligante e a estrutura média da macromolécula na fase de produção no programa SPDBV [35]. O SPDBV é um aplicativo que oferece uma interface amigável que permite que

várias moléculas possam ser analisadas ao mesmo tempo. Mutações, pontes de hidrogênio, ângulos e distâncias entre os átomos são facilmente obtidas utilizando sua interface gráfica [35].

Com essas duas estruturas 3D editadas no SPDBV, o usuário posiciona o ligante no sítio de ligação da proteína, na posição que ele deseja que seja a posição inicial deste ligante nos experimentos de *docking*. A edição desses dois arquivos no SPDBV é feita pelo *workflow* automaticamente, e o usuário somente precisa se preocupar em posicionar o ligante e armazenar este novo arquivo PDB.

A Figura 17 mostra as duas etapas dessa atividade: (a) as duas estruturas, do ligante e da proteína são editadas no SPDBV, (b) o ligante já posicionado pelo usuário em sua posição inicial dentro da proteína, que será sua posição inicial nos experimentos de *docking*.

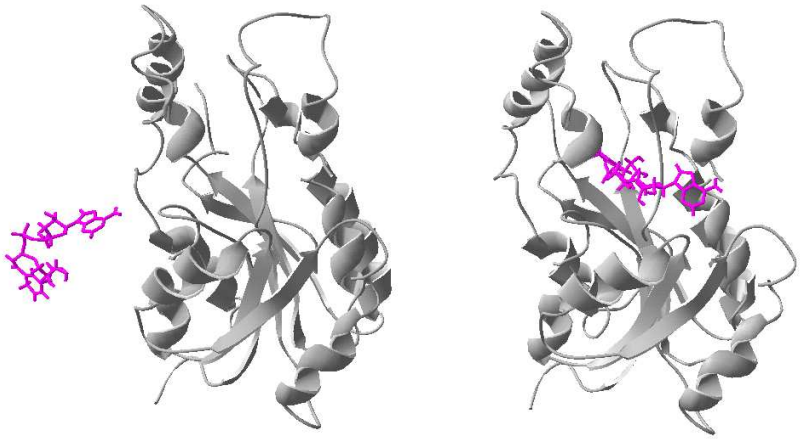


Figura 17 – (a) Estrutura 3D da proteína (InhA) e do ligante (NADH) editadas no SPDBV. (b) O NADH em sua posição inicial para o *docking* na InhA.

4.3.2 Gera Arquivo MOL2 do Ligante

Essa etapa é a última atividade envolvida na preparação do ligante para os experimentos de *docking*. Nesse momento é necessário transformar o arquivo PDB do ligante, já colocado em sua posição inicial de *docking* na proteína, em um arquivo PDBQ, que corresponde a um arquivo do ligante que contém, além das coordenadas de cada átomo, suas respectivas cargas.

Para que isso seja possível, o arquivo PDB deve antes ser transformado em um arquivo MOL2 (alguns ligantes já têm um arquivo MOL2 disponível para uso). Para transformar o PDB em MOL2, há 2 maneiras:

- Utilizando o programa MOE [42]: o PDB do ligante é aberto, a hibridização correta dos

átomos das moléculas é determinada e atribuem-se as cargas. Porém, esse *software* não é livre e precisa de licença para ser utilizado;

- Utiliza-se os arquivos MOL2 anteriormente preparados, substitui-se as coordenadas x, y e z que estão atualmente no MOL2 pelas coordenadas x, y e z do arquivo PDB do ligante em sua posição inicial na proteína. Esta troca era feita de forma manual o que exigia algum trabalho. Por exemplo, a molécula do ligante NADH é composta por 52 átomos e as coordenadas dos 52 átomos eram substituídas manualmente.

Para resolver esse problema, foi desenvolvido um programa que solicita ao usuário que informe a localização e o nome do arquivo PDB do ligante na posição inicial para o *docking* e do arquivo MOL2 original desse ligante. Esses arquivos correspondem ao primeiro e segundo quadros com trecho de arquivos mostrados na Figura 18, respectivamente.

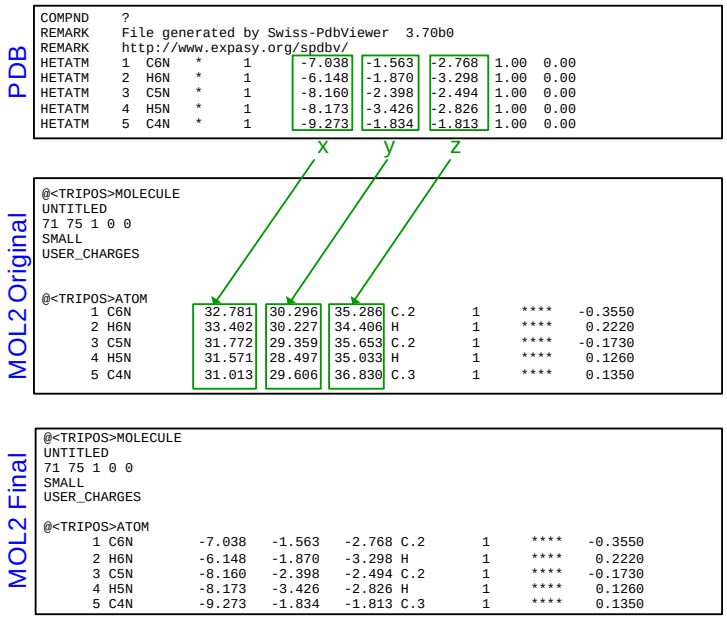


Figura 18 – Ilustração do processo de substituição das coordenadas do arquivo MOL2 pelas coordenadas do arquivo PDB do ligante posicionado para o *docking*.

O programa então lê o arquivo PDB e MOL2 e vai substituindo as coordenadas x, y e z de cada átomo do arquivo PDB no arquivo MOL2 (conforme mostra a Figura 18). Assim, ao final da execução do programa, as coordenadas dos átomos no arquivo MOL2 são iguais às do arquivo PDB (que corresponde ao terceiro quadro com trecho de arquivo mostrado na Figura 18).

A partir do arquivo MOL2, o arquivo PDBQ é gerado utilizando o programa *deftors* do AutoDock3.05 [6]. Segundo Morris *et al.* [43] esse programa define todas as torções que devem ser permitidas durante o processo de *docking*. Nesse formato final - PDBQ - o ligante está pronto para ser utilizado nos experimentos de *docking*.

4.4 Terceira Etapa - Seleciona o Tipo de Docking - Seletivo/Exaustivo:

O *workflow* desenvolvido permite que dois tipos de *dockings* sejam executados: o *docking* seletivo e o *docking* exaustivo. Após a preparação da macromolécula e do ligante para os experimentos de *docking*, o usuário deve informar qual será o tipo de *docking* a ser executado (Atividade “Docking E/S” no modelo final que pode ser visto no início do capítulo na Figura 12).

4.4.1 *Docking* Exaustivo

Se a opção for pela execução de um experimento de *docking* exaustivo significa que o experimento será executado utilizando os *snapshots* de forma seqüencial. Esse experimento exaustivo pode utilizar todos os *snapshots* gerados durante a etapa da simulação por dinâmica molecular, ou somente parte deles (que corresponde a um determinado intervalo de tempo da simulação).

Afim de tornar a execução do *workflow* flexível, é solicitado ao usuário que informe: os *snapshots* inicial e final que ele deseja utilizar e o chamado ponto de início. Os *snapshots* inicial e final indicam o intervalo de tempo da simulação que está sendo considerado no experimento e o ponto de início indica onde deve iniciar a execução do experimento. Esse valor de ponto de início é útil no caso em que haja necessidade de reiniciar a execução do processo, por exemplo, no caso de interrupção do mesmo devido a algum problema na máquina ou no *software* de execução (*Shark* 1.1-2 [34]). Ainda é importante ressaltar que a definição desses limites para execução do experimento permite que ele seja subdividido em partes que podem ser executadas em máquinas diferentes, em paralelo.

Após a definição dos limites para execução, o *workflow* calcula o total de execuções que a etapa de *docking* deve ter (pois depende dos valores dos *snapshots* inicial e final). Assim, a execução dos experimentos de *docking* se inicia, e enquanto todos os *snapshots* não tenham sido utilizados, o *subflow* de execução dos experimentos de *docking* permanece executando.

É importante ressaltar que, para que os resultados de um experimento de uma proteína com um determinado ligante possam ser utilizados como critério de seleção no caso de um experimento de *docking* seletivo (explicado logo a seguir), pelo menos uma vez, todos os *snapshots* da proteína devem ter sido utilizados no *docking*, gerando assim, uma tabela que contém os valores de energia livre de ligação (FEB), desvio médio quadrático da posição inicial (RMSD) e Tempo correspondente a cada *snapshot* da simulação por dinâmica molecular da proteína.

Como funciona a execução de cada experimento de *docking* e todos os passos que estão envolvidos serão explicados posteriormente na Seção 4.5.

4.4.2 *Docking* Seletivo

Após o usuário ter executado pelo menos uma vez o *docking* exaustivo para uma determinada proteína e ligante, é possível que a execução de novos experimentos considerando a mesma proteína, porém ligantes diferentes, seja feita de forma seletiva. Nesse tipo de execução não são utilizados todos os *snapshots* da simulação e sim somente parte deles, selecionados de acordo com um determinado critério de qualidade. Assim, o tempo necessário para analisar a interação ligante-proteína é reduzindo consideravelmente.

Um exemplo do tempo que é necessário para executar um experimento desse tipo, de forma exaustiva, foi o experimento de *docking* utilizando a proteína InhA [21] e o ligante IQG607A. Esse experimento foi executado simultaneamente em 7 máquinas do cluster Ombrófila (Pentium 3 1000 MHz, 512 MB de memória RAM - localizado no CPAD-PUCRS) e despendeu aproximadamente 100 horas de execução ininterrupta. Se tivesse sido realizado em uma máquina apenas, teriam sido necessários em torno de 700 horas (ou 30 dias) para o término dessa execução. Imaginando ainda que, se o objetivo de um certo trabalho for analisar a interação de uma proteína com um Banco de Dados de 1 milhão de ligantes, e se para cada execução fossem necessários esses 30 dias para terminar, a execução desse trabalho tornaria-se completamente inviável.

Sendo assim, devido a essa necessidade de reduzir o tempo de execução necessário para analisar cada interação proteína-ligante e baseado em alguns critérios de qualidade, essa etapa de *docking* seletivo tem por objetivo selecionar *snapshots* para serem utilizados nos experimentos de *docking* de determinado ligante, fazendo a seleção e execução dos experimentos diretamente de dentro do *workflow* desenvolvido.

Até o momento foi utilizado um critério de qualidade para selecionar os *snapshots* de uma proteína gerados pela trajetória por dinâmica molecular. Esse critério é baseado na idéia de que, se o resultado do *docking* utilizando determinado *snapshot* obteve um bom valor de FEB, ou seja, se obteve uma FEB bem negativa (que significa que a proteína com determinada conformação interagiu bem com o ligante) e um valor de RMSD pequeno (que significa que o ligante permaneceu dentro do sítio de ligação após o *docking*), é possível que este mesmo *snapshot*, interagindo favoravelmente com um ligante parecido com o primeiro (pertencentes à mesma classe de ligantes), também apresente um bom valor de energia e RMSD. Assim, se os *snapshots* cujo resultados apresentaram os melhores valores de energia resultantes de um experimento exaustivo e cujo valor de RMSD não excedeu um certo limite forem utilizados em experimentos seletivos, há uma boa possibilidade de se encontrar bons resultados sem a necessidade de executar o experimento considerando todos os *snapshots* da trajetória de DM da proteína.

Essa etapa do processo compreende a execução da atividade “Dados Seletivo” (Figura 12) e do *subflow* mostrado na Figura 19. O *subflow* é composto por duas atividades: “Prep sel.

snaps.", que gera as entradas necessárias para uma execução correta da segunda atividade "Selec Snapshot" que efetivamente executa a seleção dos *snapshots*.

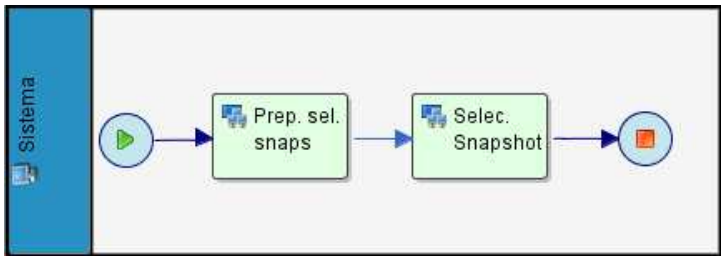


Figura 19 – *Subflow* utilizado para selecionar os *snapshots* a serem utilizados na etapa de *docking* seletivo.

Essa etapa do processo é executada de maneira geral, englobando as atividades citadas acima, conforme mostra o fluxograma da Figura 20, sendo cada etapa descrita a seguir:

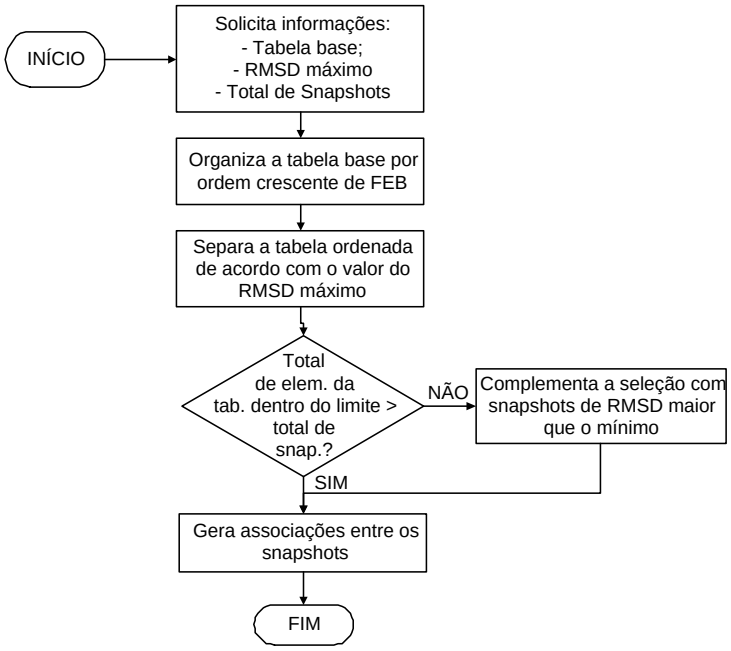


Figura 20 – Fluxograma que representa a seqüência de passos executada pelo programa que realiza a seleção dos *snapshots*.

1. O primeiro passo é executado pela atividade "Dados Seletivo" (Figura 12). Durante a execução desta atividade pelo *workflow* é solicitado ao usuário que informe o total de *snapshots* que ele quer selecionar, o valor do RMSD máximo que será permitido na seleção, e o local e nome da tabela de resultados que ele deseja utilizar como base para fazer a seleção (chamada tabela-base). A geração das tabelas bases ocorrem durante a execução dos experimentos de *docking* exaustivo, conforme será explicado na Seção 4.5. Um exemplo de tabela-base pode ser visto na Figura 21a;

2. Nesse segundo passo, a tabela-base selecionada é ordenada em ordem crescente de acordo com a FEB, como mostra a Figura 21b. A partir daí, as etapas são executadas pelo *subflow* “Selec Snapshot”. Na primeira atividade, “Prep sel. snaps.” o *workflow* prepara a entrada para o *shell script* que executa a seleção, utilizando os parâmetros informados no passo anterior. Após a seleção é efetivamente executada;

3. A tabela já ordenada é separada em duas tabelas de acordo com o RMSD informado como máximo: uma tabela contém os resultados cujo RMSD permanece dentro do limite de 0 até RMSD máximo informado e outra tabela contém os resultados cujo valor do RMSD ultrapassa o valor máximo estipulado (Figura 21c);

4. Se o total de elementos da tabela com os valores dentro do limite do RMSD ultrapassar ou for igual ao total de *snapshots* que devem ser selecionados, a seleção está pronta (Figura 21d), caso contrário, os resultados cujo RMSD excede o limite são utilizados pra complementar o total de *snapshots* a serem selecionados;

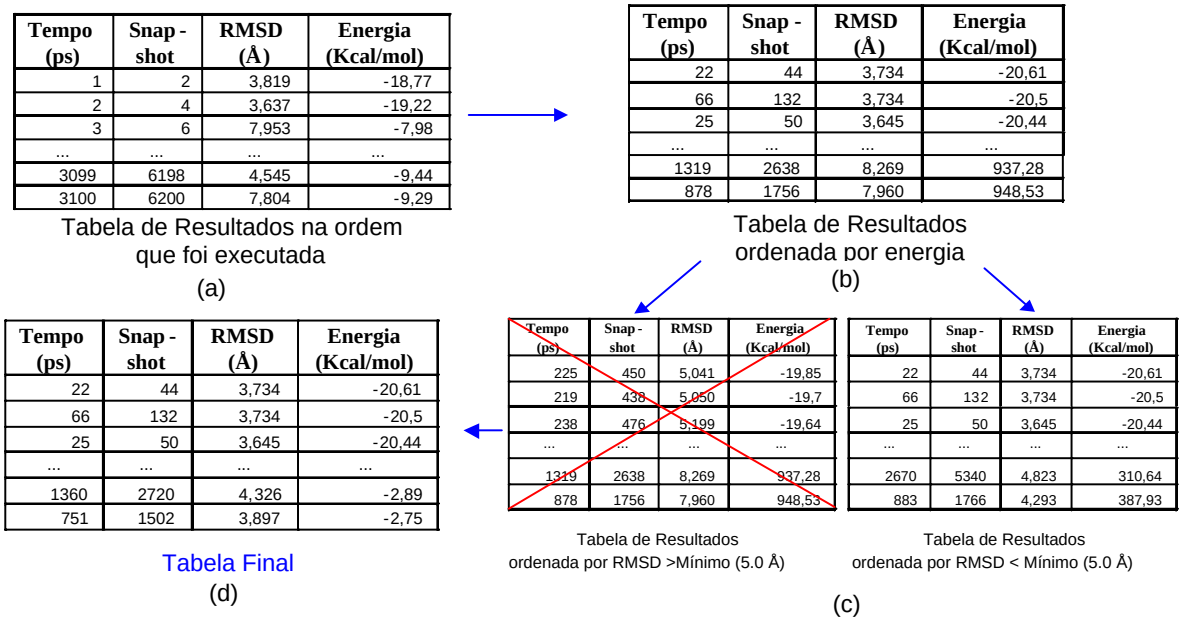


Figura 21 – Seqüência de passos executados durante a seleção dos *snapshots*.

5. É então gerado um *shell script* que determina associações entre os arquivos da proteína que devem ser utilizados nos experimentos de *docking* com os arquivos efetivamente chamados pelo *Shark1.1-2*. Essas associações são necessárias devido a dificuldade na leitura de dados externos no *Shark1.1-2*[2] . Os dados que precisariam ser lidos seriam os

[2]A leitura de dados externos ao *Shark1.1-2* necessita da criação de uma classe JAVA, porém a inclusão de novas classes no *Shark* faz parte de uma funcionalidade que está com problemas na versão 1.1-2 justamente a que está sendo utilizada no desenvolvimento do presente trabalho. Uma nova versão do *Shark* já está disponível para uso, porém ainda em fase de testes. Assim que estiver consolidada, esta passará a ser utilizada, desde que as funções que precisamos funcionem corretamente

snapshots a serem utilizados a cada experimento de *docking* que fosse ser executado (com base nos *snapshots* selecionados).

Esse *shell script* para gerar essas associações, ao ser executado pelo *Shark*, relaciona os arquivos dos *snapshots* a serem chamados pelo *Shark* com os arquivos efetivamente gravados em disco, conforme mostra a Tabela 2. Se o usuário informou 1000 *snapshots* para serem utilizados no *docking* seletivo, teríamos as seguintes associações: considerando que o melhor resultado foi para o *snapshot* 44, esse ficará associado ao primeiro arquivo PDB a ser utilizado nos experimentos de *docking* ("mdcp.pdb.1"), o segundo melhor foi o *snapshot* 132, que ficará associado ao segundo PDB e assim por diante até o milésimo PDB que, segundo a Tabela 2 corresponde ao *snapshot* 1502.

Tabela 2 – Tabela de associações para os arquivos dos *snapshots*.

Arquivo Associado	Arquivo Original
mdcp.pdb.1	mdcp.pdb.44
mdcp.pdb.2	mdcp.pdb.132
mdcp.pdb.3	mdcp.pdb.50
...	...
mdcp.pdb.999	mdcp.pdb.2720
mdcp.pdb.1000	mdcp.pdb.1502

Assim, para o *Shark*, a execução do *docking* seletivo seguirá o mesmo princípio do *docking* exaustivo, uma vez que utilizará *snapshots* aparentemente seqüenciais (os arquivos associados). Assim, não há a necessidade de modificar os *shell scripts* de execução dos experimentos de *docking* ao executar *dockings* seletivos.

Antes do desenvolvimento do *workflow* não existia essa etapa no processo pois não havia a possibilidade de execução de *dockings* seletivos.

4.5 Quarta Etapa - Executa o Docking e Gera a Tabela de Resultados

Antes de explicar essa etapa de execução do *workflow*, é necessário uma melhor descrição da funcionalidade do AutoDock3.05 [6]. De acordo com Morris *et al.* [6], o AutoDock3.05 foi desenvolvido para oferecer um procedimento automático para predição da interação entre ligantes e macromoléculas alvo. Em qualquer procedimento de *docking* molecular, há duas variáveis que precisam ser consideradas: o nível de robustez e corretude do procedimento e a demanda computacional do mesmo. O procedimento ideal encontra a energia mínima global de interação entre o ligante e a macromolécula, explorando todos os graus de liberdade (DOF) disponíveis para o sistema.

Essa etapa de execução do *docking* é executada da mesma maneira se o *docking* for exaustivo ou seletivo e pode ser vista na Figura 22. Para a execução de cada etapa foram desenvolvidos *shell scripts* e programas na linguagem C que, utilizando os parâmetros de entrada passados pelo *Shark*, executam corretamente cada uma das etapas descritas abaixo:

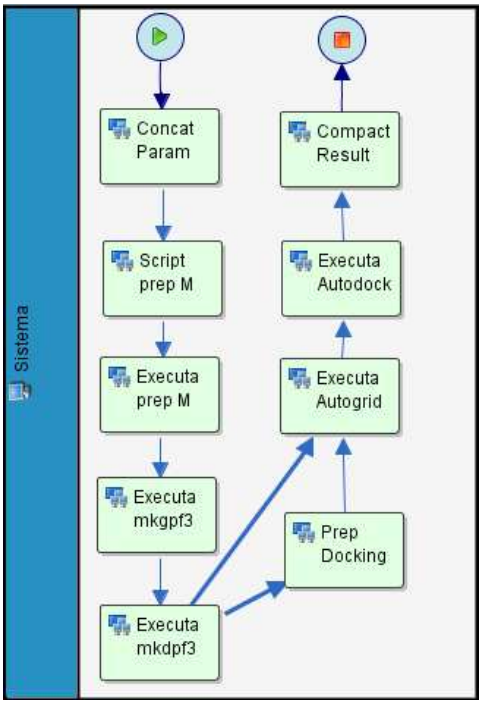


Figura 22 – *Subflow* para executar efetivamente os experimentos de *docking* utilizando o AutoDock3.05.

4.5.1 Concatenação dos Parâmetros

A primeira atividade desse *subflow* é chamada “Concat Param”. Ela executa a concatenação do contador, que indica qual será o próximo *snapshot* a ser utilizado, aos parâmetros que são necessários para execução de cada uma das atividades. Assim, a cada execução, cada uma das atividades do *subflow* executa com parâmetros diferentes, que correspondem justamente a cada um dos *snapshots*;

4.5.2 Preparação dos Arquivos PDB (*snapshots*) da Macromolécula para o *Docking*

A preparação de cada arquivo PDB para ser utilizado no *docking* é composta por 2 etapas:

1. Deve ser executado um programa que foi desenvolvido anteriormente em FORTRAN, responsável por atribuir as cargas de cada átomo da macromolécula a cada um dos arquivos PDB que correspondem aos *snapshots*. As cargas de cada átomo da macromolécula, no

caso do presente trabalho, são retiradas do arquivo “Editout_onlyP.pdb”, que foi separado do arquivo “Editout.pdb” na etapa de preparação do ligante (Seção 4.3);

2. É executado o programa *addsol* do AutoDock3.05 [6] que prepara esse arquivo PDB já com as cargas (arquivo PDBQ) para então ser utilizado no *docking*. Segundo Morris *et al.* [43] o programa *addsol* especifica parâmetros de solvatação atômica para cada átomo da macromolécula, gerando um arquivo PDBQS. Esse arquivo PDBQS é que pode ser utilizado como entrada para execução dos programas Autogrid e Autodock explicados a seguir.

   Essas duas etapas levam em consideração diferentes *snapshots* a cada execução. Por esse motivo é composto por duas atividades: a “Script prep M” na qual é gerado um *shell script* de preparação para cada *snapshot* e a “Executa prep M” na qual finalmente o arquivo da proteína está pronto para ser utilizado no *docking*.

4.5.3 Executa Mkgpf3

Esse programa do AutoDock3.05 gera o arquivo *Input.gpf* que será utilizado como parâmetro de entrada na execução do *autogrid*.

4.5.4 Executa Mkdpf3

Assim como o *mkgpf3*, o *mkdpf3* gera o arquivo *Input.dpf*, utilizado como entrada para execução do *autodock*. Esse arquivo determina as características dos experimentos de *docking*.

4.5.5 Edita Preparação do *Docking*

Essa etapa é executada somente uma vez, sempre na primeira execução do *subflow*. Durante essa atividade, o arquivo gerado na etapa anterior, de preparação para o *docking*, é editado diretamente pelo *Shark* utilizando o editor *nedit*. Dessa forma, o usuário pode alterar alguns dos parâmetros contidos nesse arquivo, como por exemplo: o nome do arquivo que descreve a macromolécula (que deve ser alterado para um nome padrão pois o mesmo arquivo *Input.dpf* será utilizado em todos os experimentos de *docking*), o tipo de algoritmo de *docking* a ser utilizado: *simulated anneling - SA* ou *genetic algorithm - GA*, o número de passos que devem ser executados em cada experimento, entre outros.

4.5.6 Executa AutoGrid

O *autogrid* define um mapa de *grids* para cada um dos tipos de átomos do ligante. Esse mapa corresponde a uma matriz 3D de pontos igualmente espaçados, centrado em alguma região de interesse da macromolécula em estudo. Cada ponto dentro do mapa de *grids* armazena a energia potencial de um átomo de prova em relação a todos os átomos na macromolécula. Assim, durante o *docking* esses valores de energia são utilizados para reduzir os cálculos que são necessários para se chegar ao valor final da FEB para cada experimento. Um exemplo do mapa de *grids* pode ser visto na Figura 23 (adaptada de Morris *et al.* [43]).

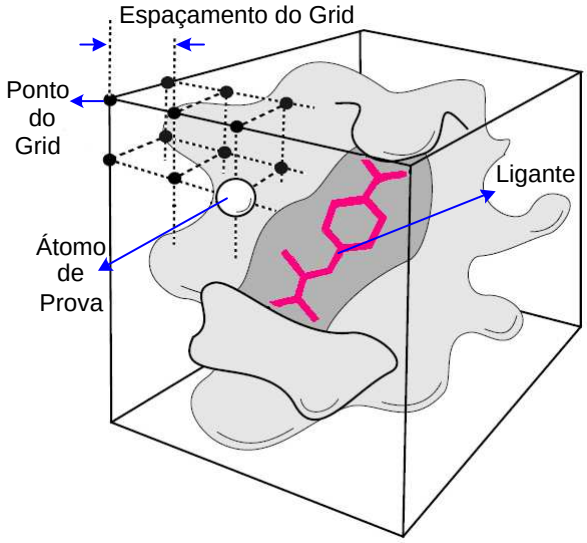


Figura 23 – Exemplo de um mapa de *grids*.

4.5.7 Executa Autodock

O *autodock* avalia a interação entre o receptor e o ligante. São executadas n interações, cada uma com o ligante em uma posição diferente. De acordo com um escore, dado na forma de intensidade de energia de interação, avalia-se se o ligante e determinada conformação da proteína apresentaram uma interação favorável.

Ao final da execução, o *autodock* gera um arquivo de saída que contém as informações a respeito de cada experimento de *docking*. Um trecho desse arquivo pode ser visto na Figura 24, contendo as seguintes informações:

- O arquivo contém um cabeçalho com informações sobre o programa e dados referentes à simulação, como por exemplo: quais são os arquivos e parâmetros de entrada do experimento, algumas características sobre a máquina na qual o experimento foi executado,

```
LOWEST ENERGY DOCKED CONFORMATION from EACH CLUSTER
                ____________________________________________________

Residue number will be set to the conformation's cluster rank.

MODEL        7
USER    Run = 7
USER    Cluster Rank = 1
USER    Number of conformations in this cluster = 2
USER
USER    RMSD from reference structure       = 6.078 A
USER
USER    Estimated Free Energy of Binding    =   -9.91 kcal/mol  [=(1)+(3)]
USER    Estimated Inhibition Constant, Ki   =   +5.46e-08       [Temperature = 298.15 K]
USER
USER    Final Docked Energy                 =   -9.91 kcal/mol  [=(1)+(2)]
USER
USER    (1) Final Intermolecular Energy     =   -9.91 kcal/mol
USER    (2) Final Internal Energy of Ligand =   +0.00e+00 kcal/mol
USER    (3) Torsional Free Energy           =   +0.00e+00 kcal/mol
USER
USER
USER    DPF = NADH.I16Tmacro_SA01.dpf
USER    NEWDPF move           NADH.pdbq
USER    NEWDPF about          -2.256000 6.264000 4.018000
USER    NEWDPF tran0          -6.320467 7.380286 6.343469
USER    NEWDPF quat0          0.643646 0.403506 -0.650309 134.467765
USER
USER                Rank         x       y       z       vdW    Elec      q     RMS
ATOM      1  A6N ***     1      -5.508   1.228  -0.426    -0.45  -0.07  -0.133  6.078
ATOM      2  A5N ***     1      -5.315  -0.184  -0.408    -0.56  -0.01  -0.047  6.078
```

Figura 24 – Trecho do arquivo de saída do *autodock*. Os valores que são lidos e armazenados para uso posterior estão marcados na fi gura, em magenta o RMSD e em ciano a FEB.

etc. Esses valores não aparecem no exemplo de arquivo mostrado na Figura 24 pois se encontram no início desse arquivo de saída do *autodock*, trecho não abordado na figura;

- Descrição do ligante: átomos que o compõem, torções aplicadas a cada átomo, etc.;
- Listagem dos cálculos executados durante a simulação;
- Parâmetros sobre o método utilizado: *Genetic Algorithm* ou *Simulated Anneling*;
- Energia e localização de cada átomo envolvido na simulação em cada um dos passos executados;
- Tabelas com os resultados dos passos executados. Essas tabelas são ordenas por ordem crescente de FEB. Os resultados da primeira tabela listada é que são os que são buscados e armazenados na tabela de resultados que é gerada para cada experimento.

Esses valores são armazenados e posteriormente utilizados para análise e seleção de *snapshots* para outros experimentos de *docking*.

4.5.8 Finaliza Execução

Após a realização dos cálculos de afinidade proteína-ligante pelo *autodock*, é preciso armazenar os valores do arquivo de saída do *autodock* importantes para a análise desses resultados.

Sendo assim, durante essa atividade é executado um programa desenvolvido em linguagem C que vai armazenando, em um arquivo texto, as informações conforme mostra a Tabela 3:

Essa tabela contém informações a respeito de cada um dos experimentos executados: na coluna 1, o tempo (em ps) que corresponde a qual *snapshot*, a coluna 2 indica qual é esse *snapshot* (devido à desconsideração de parte deles, o tempo e o *snapshot* não são equivalentes), na coluna

Tabela 3 – Exemplo da tabela de resultados gerada em uma execução de experimentos de *docking* exaustivo do sistema IQG607A - InhA.

Tempo (ps)	Snap shot	RMSD Å	FEB (Kcal/mol)	Tempo *autogrid* (min)	Tempo *autodock* (min)
1	2	6.298	-9.94	4:50.02	10:22.50
2	4	6.167	-10.16	4:06.81	10:08.61
3	6	6.099	-10.14	4:15.85	10:15.97
4	8	4.906	-9.92	4:25.54	10:11.99
5	10	5.267	-10.25	4:05.40	10:08.66
...	...	...	...	...	...
3098	6196	3.719	-9.74	4:25.49	9:52.37
3099	6198	3.931	-9.96	4:39.90	9:41.25
3100	6200	3.989	-9.75	4:30.58	10:00.94

3 tem-se o valor do RMSD que indica o quanto o centro de massa do ligante se distanciou do seu ponto inicial após o *docking*, a coluna 4 contém o valor da FEB que corresponde à energia final resultante da interação proteína-ligante e as colunas 5 e 6 mostram os tempos despendidos na execução do *autogrid* e *autodock* respectivamente.

Pode acontecer de, no final da execução do experimento pelo *autodock*, o número de passos estabelecidos pelo usuário não terem sido suficientes para a convergência do resultado da interação de determinada conformação da proteína com o ligante. Ou, nos primeiros passos de execução do experimento, o *autodock* já consegue prever que determinada conformação da proteína não obterá bons resultados (por exemplo, durante todos os primeiros passos, o resultado da interação foi um valor de FEB positivo, que significa que pode estar havendo colisão entre os átomos). Nesses casos, o arquivo de saída do *autodock* não conterá os valores de energia nem RMSD finais, mas de alguma forma essa conformação deve aparecer na tabela de resultados, pois é necessário saber como cada uma das conformações interagiu com o ligante. Assim, quando durante a leitura desse arquivo de saída não forem encontrados esses valores, na tabela de resultados que está sendo gerada é gravado para FEB um valor alto de 100.000 *kcal/mol* e para RMSD um valor também absurdo de 1000 Å. Dessa forma, essas conformações, quando ordenadas pela energia, aparecerão como última opção para serem utilizadas nos experimentos seletivos. Também assim, facilmente se identifica as conformações que não tiveram sua interação testada com sucesso com o ligante.

Além de gerar a tabela de resultados, o *shell script* executado por essa atividade compacta o arquivo de saída do *autodock* (diminuindo o espaço em disco ocupado pelos resultados) e exclui os arquivos que são úteis somente a cada experimento que está sendo executado.

No caso de *dockings* seletivos, há a necessidade ainda de renomear o arquivo de saída do *autodock*, uma vez que o nome do mesmo é definido pelos parâmetros de execução do *Shark*, seguindo uma ordem seqüencial. Assim, o arquivo de saída, que após o término da execução do *a utodock* chamava-se, por exemplo, *Result-1.dlg.gz* deve passar a se chamar *Result-<primeiro*

snapshot da seleção>.dlg.gz e assim por diante.

Antes do desenvolvimento do *workflow* toda essa etapa era executada utilizando um *shell script* que continha praticamente os mesmos passos executados por esse *subflow*. Porém, quaisquer mudanças na proteína, ligante, número total de experimentos a serem executados, etc. deveriam ser manualmente alteradas no *shell script*, sendo necessário para isso que o usuário conhecesse sobre programação e sobre *shell scripts* para que conseguisse realizar corretamente as alterações. Em separado, pelo menos uma vez, todos os passos deveriam ser executados manualmente para que os parâmetros do *autodock* fossem configurados e para se ter certeza de que todos os comandos do *shell script* foram modificados de forma correta.

Ademais, a tabela de resultados, que agora é gerada juntamente com a execução de cada experimento de *docking*, não era feita antes do desenvolvimento do *workflow*. Um *shell script*, que também chamava programa escritos em FORTRAN, executava a leituras dos resultados dos *dockings*, que eram armazenados em arquivos de texto, um contendo os valores de melhor FEB de cada experimento e outro os valores de RMSD destes experimentos. Porém , como estavam em arquivos separados, sem identificação do *snapshot* correspondente, tinha-se dificuldade em analisar os dados. Ainda, como acontecia com o *shell script* de execução de experimentos, mudanças na proteína, ligante ou no número de experimentos deveriam ser manualmente alteradas.

4.5.9 Considerações Finais

Este capítulo apresentou toda a modelagem e implementação do *workflow* científico desenvolvido, descrevendo detalhadamente cada uma de suas etapas. Em paralelo, foi sendo mostrado como essas atividades eram executadas antes do desenvolvimento deste trabalho. Com base nas diferenças entre como as atividades são executadas atualmente e como elas eram realizadas anteriormente, já é possível perceber grandes vantagens no uso do *workflow* para execução de todo esse processo, tornando-o muito mais robusto e flexível.

O próximo capítulo apresenta um Estudo de Caso realizado afim de validar o *workflow* desenvolvido. Foram feitos experimentos utilizando a proteína InhA e os ligantes NADH (*docking* exaustivo), IQG607A (*dockings* exaustivo e seletivo), IQG607B (*docking* seletivo), TCL (*dockings* exaustivo e seletivo) e ETH (*docking* seletivo). Esse capítulo descreve como foram realizados os experimentos e analisa os resultados encontrados nos *dockings* seletivos e exaustivos com o objetivo de mostrar a eficácia dos resultados obtidos.

5 Resultados: Estudo de Caso

Esse capítulo tem por objetivo descrever o Estudo de Caso realizado visando validar a modelagem e implementação do *workflow* científico desenvolvido (Capítulo 4), assim como validar o primeiro critério de qualidade que foi definido para ser utilizado na seleção de *snapshots* para a realização de *dockings* seletivos. Neste capítulo também são descritos os resultados que foram obtidos em todos os experimentos, mostrando que os resultados dos experimentos de *docking* seletivo são muito satisfatórios.

5.1 Execução da Simulação por Dinâmica Molecular

Apesar da etapa de simulação por DM do receptor não ser realizada durante a execução do *workflow*, é importante que se conheça a mesma, pois esta descreve a maneira como foram gerados todos os *snapshots* do receptor utilizados nos experimentos descritos logo a seguir.

De acordo com Schroeder *et al.* [44], a enzima InhA de *Mycobacterium tuberculosis* pode ser considerada uma molécula flexível e foi escolhida como modelo de receptor no presente trabalho. A flexibilidade dessa enzima pode ser vista na Figura 2 (Capítulo 2). Essa flexibilidade explícita foi obtida em uma simulação por dinâmica molecular gerada pelo módulo SANDER do AMBER6.0 [10] como previamente descrito em [44].

A simulação por DM foi feita por 3100 ps e os *snapshots* instantâneos foram gerados a cada 0.5 ps e salvos em arquivos de pacotes de 50 ps (um total de 100 *snapshots* são armazenados em cada arquivo de saída da dinâmica). Assim, um total de 6200 *snapshots* do receptor foram gerados. É importante ressaltar que, durante a etapa de preparação da macromolécula (Seção 4.2), metade desses *snapshots* foram desconsiderados porque não há necessidade de se utilizar estruturas tão consecutivas nos experimentos de *docking*.

5.2 Descrição dos Experimentos

Para esse Estudo de Caso foram realizados os experimentos descritos na Tabela 4, no qual: a primeira coluna mostra o número do experimento, a segunda o nome de cada um dos ligantes, a terceira o número de átomos de cada um deles, a quarta o tipo de experimento executado, se foi um *docking* exaustivo (considerando a trajetória completa da simulação por DM) ou seletivo

(utilizando somente parte dos *snapshots* da simulação por DM), a quinta o tempo médio despendido na execução de um experimento de *docking* para cada um dos ligantes (em minutos) e a sexta coluna contém o tempo total de execução aproximado de cada experimento, em horas. A coluna 7 indica o local em que cada experimento foi executado: máquinas do LABIO (PC 1 ou PC 2) ou o Cluster Ombrófila do CPAD/PUCRS, juntamente com os valores de *benchmark* de cada uma dessas máquinas. Os *benchmarks* foram obtidos com base em resultados de testes de performance disponibilizados pela *Standard Performance Evaluation Corporation* - SPEC [45] [1]. Esses valores correspondem à razões entre um tempo de referência e o tempo efetivamente gasto na execução de determinada aplicação. Como são executadas diferentes aplicações, é feita uma média entre os resultados obtidos com cada uma delas e esses valores é que são disponibilizados pela SPEC. Entre todas as possibilidades disponíveis, selecionamos as médias de resultados de execução de aplicações de ponto flutuante (mais próximas das operações realizadas pelo AutoDock3.05) para máquinas com configurações muito semelhantes às utilizadas na execução dos nossos experimentos.

Tabela 4 – Descrição dos experimentos de *docking* realizados.

Experim.	Ligante	Número átomos[a]	Tipo execução	Tempo médio execução 1 *docking*	Tempo total (hs) (aprox.)	Local/ *Benchmarks*
1	NADH	52	Exaust.	15 min	120	Cluster/1848[b]
2	IQG607A	24	Exaust.	13 min	100	Cluster/1848
3	IQG607A	24	Selet.	-	-	-
4	IQG607B	35	Selet.	11 min	200	PC 1/590
5	TCL	18	Exaust.	09 min	500	PC 1/590
6	TCL	18	Selet.	06 min	100	PC 2/850
7	ETH	13	Selet.	04 min	80	PC 2/850

[a]Total de átomos após a fusão dos hidrogênios não polares a seus respectivos átomos pesados, feito na preparação do ligante para o *docking*.

[b]Cada PC do *cluster* tem *benchmark* = 231, como são 7 PCs executando ao mesmo tempo, tem-se um *benchmark* aproximado de 7 vezes esse valor, totalizando 1848.

Onde:

- **Cluster Ombrófila:** 7 PCs Pentium 3 1000MHz com 256 Mb de memória RAM;
- **PC LABIO 1:** 1 PC Pentium 4 1400MHz com 512 Mb de memória RAM;

[1]O SPEC [45] é um consórcio, sem fins lucrativos cujos membros, são principalmente desenvolvedores de *hardware* e *software*, estudantes, clientes e consultores da área. A missão principal da SPEC é o desenvolvimento de *benchmarks* tecnicamente confiáveis e objetivos para múltiplos sistemas operacionais e ambientes, incluindo computação de alto-desempenho, *Web-services*, etc. Em 2000, a SPEC disponibilizou o CPU2000, um *benchmark* de CPU composto por 19 aplicações. Esse *benchmark* passou a ser utilizado por um grande número de diferentes pessoas, que foram enviando seus resultados de teste à SPEC. Por esse motivo, atualmente existem registros de teste de performance para as mais diferentes máquinas. Esses testes foram realizados utilizando aplicações que executam operações matemáticas com inteiros ou com ponto-flutuante. Esses valores permanecem disponíveis no site da SPEC e podem ser utilizados.

- **PC LABIO 2:** 1 PC Pentium 4 2400MHz com 1Gb de memória RAM.

5.2.1 Experimento com o Ligante NADH

Um *docking* exaustivo do ligante NADH na enzima InhA, utilizando os *snapshots* gerados na simulação por DM desta macromolécula foi realizado previamente por nosso grupo no LABIO [28]. Os arquivos de saída dos *dockings* foram utilizados para a elaboração de uma tabela semelhante à mostrada na Tabela 3 (Capítulo 4). Esta tabela, contendo os valores de FEB e RMSD para os *dockings* do NADH nos 3100 *snapshots* da InhA, foi utilizada como base para a seleção dos *snapshots* a serem utilizados nos experimentos de *docking* seletivo.

Dos 3100 *snapshots* da macromolécula utilizados neste experimento de *docking* exaustivo, foram selecionados 1000 *snapshots*, correspondentes àquelas conformações da macromolécula que apresentaram os 1000 melhores valores de FEB, com valor de RMSD menor que 5,0 Å.

A execução deste *docking* exaustivo durou cerca de 120 horas e foi realizado no cluster Ombrófila localizado no CPAD-PUCRS.

O ligante NADH é composto por 71 átomos (que corresponde ao total de átomos antes da fusão dos hidrogênios não polares a seus respectivos átomos pesados, feito na preparação do ligante para o *docking*, sua estrutura após essa preparação para o *docking* é composta por 52 átomos) pode ser considerado uma molécula ligante grande. Portanto, espera-se que ele ocupe boa parte da cavidade de ligação da InhA.

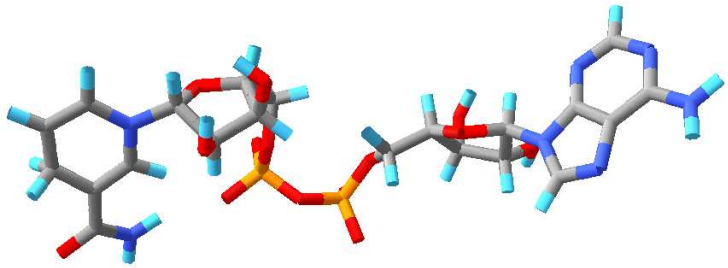


Figura 25 – Estrutura 3D do ligante NADH.

5.2.2 Experimentos com o Ligante IQG607A

O ligante Isoniazida Pentacianoferrato (IQG607A - Figura 26), uma molécula menor que o NADH, é composta por 28 átomos antes da preparação do ligante para o *docking* e 24 átomos, após. Com esse ligante foram realizados 2 experimentos.

Inicialmente, foi realizado um *docking* exaustivo deste ligante na InhA. A realização desse experimento consumiu aproximadamente 100 horas. Como este experimento foi realizado no

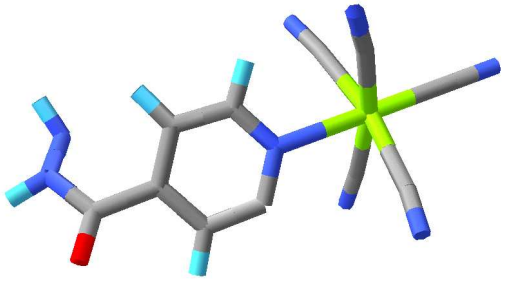


Figura 26 – Estrutura 3D do ligante IQG607A.

cluster Ombrófila, não foi possível utilizar o *workflow* desenvolvido em substituição aos *scripts* previamente utilizados no experimento com o NADH.

Os resultados do experimento de *docking* exaustivo do IQG607A com a InhA foram registrados. A partir daí, foi realizada uma seleção de 1000 *snapshots* pela avaliação e seleção daquelas 1000 conformações da macromolécula (InhA) que apresentaram os melhores valores de FEB (valores mais negativos) no experimento de docking exaustivo InhA-NADH, e cujos valores de RMSD não fossem maior do que 5,0 Å. Ou seja, dos resultados do docking exaustivo dp IQG607A, foram selecionados 1000 resultados: os 1000 melhores resultados do docking InhA-NADH. Isso foi feito para analisar se os resultados do docking seletivo seriam coincidentes, e portanto representativos do docking exaustivo.

5.2.3 Experimento com o Ligante IQG607B

Um *docking* seletivo com o ligante Diisioniazida-tetracianoferrato (IQG607B), composto por 43 átomos (esse número corresponde ao total de átomos antes da preparação do ligante para o *docking*, durante os experimentos, após a preparação do mesmo, esse ligante permanece com 35 átomos), foi realizado como descrito na Seção 5.2.2. A estrutura desse ligante pode ser vista na Figura 27.

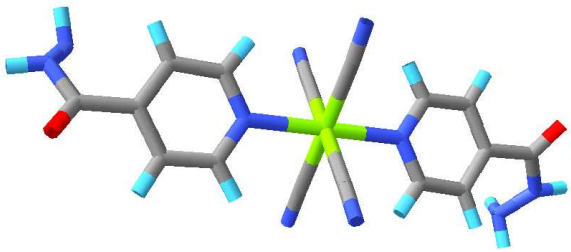


Figura 27 – Estrutura 3D do ligante IQG607B.

Esse experimento levou aproximadamente 200 horas de execução em um PC no LABIO, utilizando o *workflow* desenvolvido.

5.2.4 Experimentos com o Ligante TCL

Utilizando o ligante Triclosan (TCL), também uma molécula pequena, formada por 24 átomos (antes da preparação para o *docking*, após, o ligante tem 18 átomos)(Figura 28) foram realizados 2 experimentos. Num primeiro momento foi realizado um *docking* seletivo usando os mesmos *snapshots* utilizados no experimento com o IQG607B (1000 *snapshots* que apresentaram melhor valor de FEB ao interagir com o NADH). A duração desse experimento foi de aproximadamente 100 horas, utilizando um PC do LABIO e, assim como no caso do IQG607B, o *workflow* desenvolvido foi utilizado.

Num segundo momento testou-se a eficiência do *workflow* desenvolvido para a execução também de experimentos exaustivos e a análise da representatividade dos resultados do *docking* seletivo para o ligante TCL. O experimento exaustivo foi realizado considerando os 3100 *snapshots* da trajetória de simulação por dinâmica molecular da InhA em um PC no LABIO e durou cerca de 500 hs.

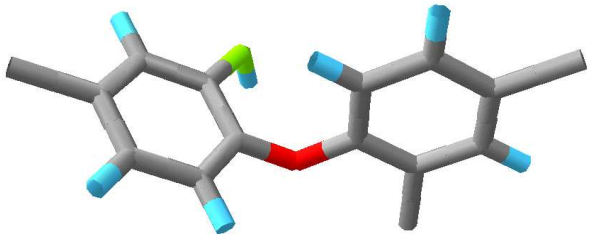


Figura 28 – Estrutura 3D do ligante TCL.

5.2.5 Experimento com o Ligante ETH

Por fim, foi executado um experimento de *docking*, também seletivo, utilizando o ligante Etionamida (ETH - Figura 29). Esse ligante é uma molécula pequena composta por 21 átomos antes da preparação do ligante para o *docking* e 13 átomos depois da preparação (estrutura utilizada nos experimentos de *docking*). A mesma seleção dos experimentos anteriores foi utilizada nesse experimento. Dentre os ligantes utilizados nos experimentos, esse é o único que, sabidamente, necessita estar associado ao NADH para se ligar a InhA. A duração deste experimento foi de aproximadamente 80 horas e também utilizou o *workflow* desenvolvido.

5.3 Resultados Obtidos

Os resultados obtidos com os experimentos estão listados na Tabela 5.

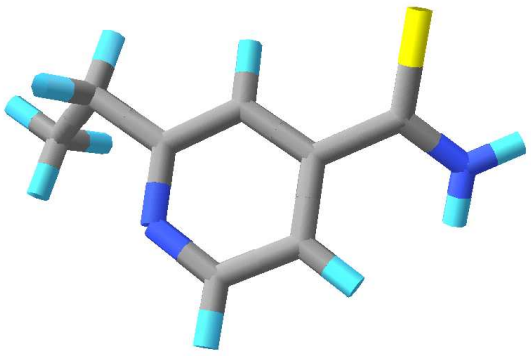


Figura 29 – Estrutura 3D do ligante ETH.

Tabela 5 – Resumo dos resultados dos experimentos de *docking* realizados.

Experim.	Ligante	Média FEB(-) *kcal/mol*	Total Result. FEB(-)	Total Result. FEB(+)	Total Result. Não Dock	Média RMSD (Å)
1	NADH exaust.	-12,9 $\pm$ 4,2	2822	278	0	5,3 $\pm$ 2,2
2	IQG607A exaust.	-9,9 $\pm$ 0,6	3041	0	59	5,0 $\pm$ 1,5
3	IQG607A selet.	-9,9 $\pm$ 0,5	991	0	9	4,8 $\pm$ 1,4
4	IQG607B selet.	-10,6 $\pm$ 1,2	933	21	46	4,5 $\pm$ 1,5
5	TCL exaust.	-8,9 $\pm$ 0,3	2836	0	264	6,8 $\pm$ 1,8
6	TCL selet.	-8,9 $\pm$ 0,3	921	0	79	6,7 $\pm$ 1,7
7	ETH selet.	-6,7 $\pm$ 0,3	984	0	16	5,2 $\pm$ 2,4

A coluna 1 descreve o número do experimento, a coluna 2, o nome de cada um dos ligantes utilizados nos respectivos experimentos. A coluna 3 refere-se a média e o desvio padrão da FEB, considerando somente os resultados na qual a FEB final foi negativa. Os resultados na qual a FEB final foi positiva, ou não convergiu, não foram considerados. As colunas 4, 5 e 6 contabilizam o total de experimentos cuja FEB foi negativa, positiva ou cuja execução foi interrompida (talvez por não apresentar convergência com o protocolo de *docking* utilizado), respectivamente. A soma dos valores dessas 3 colunas indica o total de experimentos realizados com cada um dos ligantes (3100 para *dockings* exaustivos e 1000 para *dockings* seletivos). A coluna 7 lista a média e o desvio padrão do RMSD encontrado em cada um dos experimentos (quanto maior os valores de RMSD e seu desvio padrão, maior a variação da posição final de menor energia do ligante em relação a sua posição inicial).

- O experimento de *docking* exaustivo do NADH confirmou a boa afinidade da enzima InhA por este ligante [46], apresentando uma média de energia livre de ligação (FEB) de $-12,9 \pm 4,2$ *kcal/mol*: a maior afinidade identificada entre os ligantes testados.

  Por se tratar de uma molécula grande, a posição final de ligação do NADH não varia muito dentro do seu sítio de ligação na proteína, portanto, não apresenta valores muito altos de RMSD (RMSD = $5,3 \pm 2,2$ Å). A Figura 30 mostra a estrutura média 3D da InhA na

representação de *ribbons* cinza e três posições diferentes do ligante após os experimentos de *docking*.

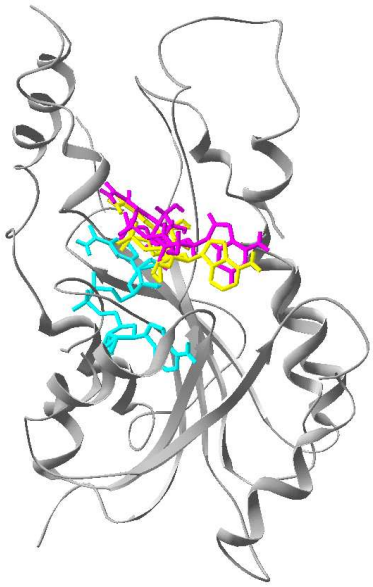


Figura 30 – Estrutura 3D da InhA e 3 posições finais do ligante NADH: em magenta a posição inicial, em amarelo a posição final com o *snapshot* 4578 e em ciano a posição final com o *snapshot* 1456.

A posição inicial do ligante em seu sítio de ligação é representada pela estrutura em magenta. Esta posição inicial do ligante é a mesma para todos os *snapshots* utilizados no *docking* deste ligante. Em amarelo está representada resultado do experimento com o *snapshot* 4578[2] onde a posição final do ligante tem um baixo valor de RMSD de $1,2$ Å (isto é, encontra-se ligado próximo à posição inicial) e FEB = $-15,9$ *kcal/mol*. Esta posição de ligação corresponde à posição de ligação do NADH observada na estrutura 3D da enzima InhA complexada com o NADH, determinada por cristalografia de raios X [21]. Em ciano, está representada a posição final do ligante que apresenta um valor alto de RMSD de $6,1$ Å (distante da posição inicial) que corresponde ao *snapshot* 1456 (FEB = $-16,7$ *kcal/mol*).

- O *docking* exaustivo do ligante IQG607A apresentou uma boa média de energia livre de ligação, com um desvio padrão bem pequeno (FEB = $-9,9 \pm 0,6$ *kcal/mol*), sugerindo que a energia de ligação do IQG607A não varia muito entre os experimentos. Utilizando o programa VMD [47] para visualização das estruturas 3D, foi possível analisar, de forma seqüencial, a posição final do ligante em cada um dos experimentos. Essa análise demonstrou que todas as conformações finais de *docking* do ligante, que apresentam

[2] A simulação por DM da InhA gerou 6200 *snapshots*, em intervalos de 0.5 em 0.5 ps. Porém, como eram estruturas muito consecutivas, metade delas foi desconsiderada, todas aquelas com numeração ímpar, que correspondiam aos *snapshots* em tempos do tipo 0.5 ps, 1.5 ps, 2.5 ps, e assim por diante. Assim, a numeração dos *snapshots* corresponde a valores que variam de 0 a 6200, de 2 em 2 (por exemplo, os *snapshots* 0, 2 e 4 correspondem aos tempos de simulação da dinâmica de 0, 1 e 2 ps respectivamente). Totalizando por isso, 3100 *snapshots* a serem utilizados nos experimentos de *docking*.

as melhores energias livres de ligação (energias mais negativas), encontram-se dentro do sítio de ligação.

A Figura 31 mostra a estrutura 3D da proteína e três diferentes posições do ligante ao final do *docking* exaustivo: a posição de referência em magenta, uma posição cujo valor de RMSD foi baixo (RMSD = $2,3$ Å) que corresponde ao *snapshot* 5774, em ciano, o ligante encontra-se distante de sua posição inicial, com um RMSD de $9,4$ Å (*snapshot* 3258). Ambos os resultados escolhidos para a representação das possíveis localizações finais do ligante na InhA apresentaram bons valores de energia de $-10,0$ *kcal/mol* e $-9,6$ *kcal/mol*, respectivamente.

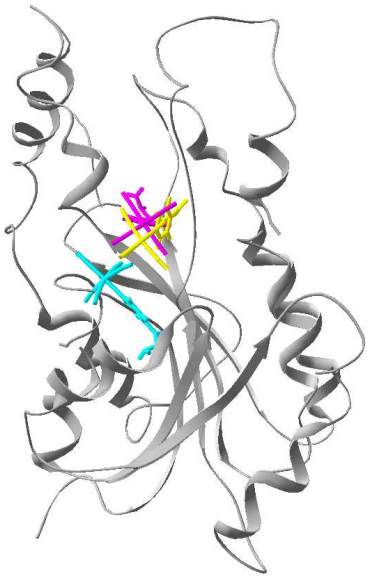


Figura 31 – Estrutura 3D da InhA em cinza e 3 posições finais do IQG607A: em magenta sua posição inicial, em amarelo a posição final com o *snapshot* 5774 e em ciano a posição final com o *snapshot* 3258.

Como esse experimento foi realizado de forma exaustiva, foram computados/armazenados os resultados de *docking* para todos os *snapshots* da trajetória (total de 3100 *snapshots*).

Desta extensa lista, foram selecionados somente aqueles *snapshots* correspondentes aos 1000 melhores resultados obtidos no *docking* do NADH (seleção dos 1000 *snapshots* usados para o *docking* seletivo).

A média de energia livre de ligação do IQG607A na InhA, calculada apenas para aquela seleção foi de $-9,9 \pm 0,5$ *kcal/mol*: estatisticamente igual à média sobre o total de *snapshots* que foi de $-9,9 \pm 0,6$ *kcal/mol*. Este resultado demonstra que, para este ligante, a seleção dos *snapshots* baseada nos resultados do *docking* do NADH foi efetiva. Isto é, os 1000 *snapshots* selecionados para agilizar o processo de *docking* são representativos da flexibilidade do receptor, e neste caso, poderiam ser usado para a avaliação da afinidade do ligante em questão.

- O experimento seletivo utilizando o ligante IQG607B apresentou também uma boa média

de energia (FEB= $-10,6 \pm 1,2$ *kcal/mol*). Por tratar-se de uma molécula grande como o NADH, seus valores de RMSD não formam muito grandes, indicando que este ligante não se deslocou muito dentro do sítio de ligação. A análise visual com o VDM demonstrou que, na grande maioria dos experimentos, o ligante permaneceu dentro do sítio de ligação.

A Figura 32 mostra a InhA em cinza juntamente com a posição inicial do IQG607B e duas diferentes posições finais do mesmo após os experimentos de *docking*:

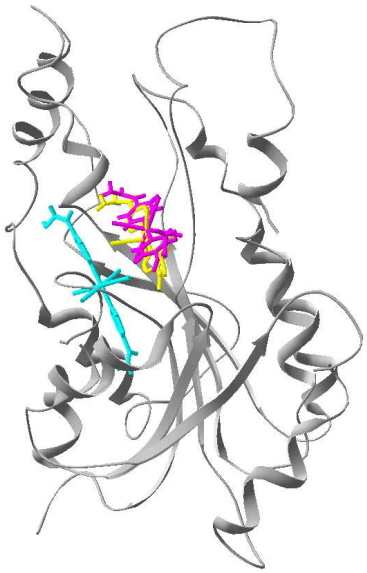


Figura 32 – Estrutura 3D da InhA e 3 posições finais do IQG607B: em magenta sua posição inicial, em amarelo a posição final com o *snapshot* 4442 e em ciano a posição final com o *snapshot* 1742.

Em amarelo, o ligante com um baixo valor de RMSD (RMSD = $1,2$ Å e FEB = $-11,0$ *kcal/mol*) que corresponde a um experimento onde a posição final do ligante (correspondente ao *snapshot* 4442) não se distanciou muito de sua posição inicial, em magenta. Em ciano acontece o oposto, o ligante distanciou-se bastante de sua posição original (*snapshot* 1742), apresentando, por isso, um alto valor de RMSD (RMSD = $8,2$ Å e FEB = $-9,5$ *kcal/mol*).

- O TCL, quando ligado à enzima InhA, apresenta um de seus anéis interagindo com o anel nicotinamida do NADH (conforme estrutura cristalográfica 1P45 [48]), o que o fixa naquela posição. Porém, a ausência do NADH no *docking* possibilita que o TCL ocupe também outras regiões, como uma posição mais externa ao seu local de ligação original e anteriormente ocupada pela porção adenina do NADH, o que justifica os maiores valores de RMSD observados, tanto no experimento exaustivo, quanto seletivo. A média de FEB do experimento exaustivo foi de $-8,9 \pm 0,3$ *kcal/mol*, considerada uma boa média e com muito pouca variação.

  A partir da análise visual com o VMD pode-se concluir que este *docking* seletivo representa adequadamente as formas de ligação que seriam observadas num *docking* exaustivo,

pois, assim como para os outros ligantes, a análise das posições finais do TCL demonstrou que na maioria dos experimentos o ligante permaneceu dentro do sítio de ligação, ou no máximo na posição que deveria ser ocupada pelo NADH, conforme mostra a Figura 33.

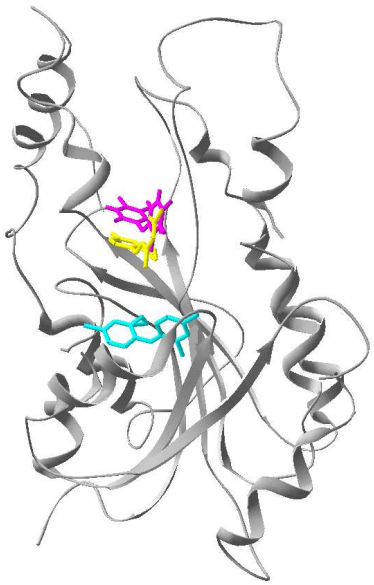


Figura 33 – Estrutura 3D da InhA em cinza e 3 posições finais do TCL: em magenta sua posição inicial, em amarelo a posição final com o *snapshot* 3270 e em ciano a posição final com o *snapshot* 1188.

A Figura 33 mostra a InhA em cinza (representação em *ribbons*). Em magenta, assim como acontece nas Figuras 31 e 32, está o ligante em sua posição inicial (que é a mesma em todos os experimentos de *docking*). O TCL, em amarelo na Figura 33, é um exemplo de resultado com um baixo valor de RMSD de $1,9$ Å (*snapshot* 3170 e FEB = $-8,3$ *kcal/mol*) e em ciano, um exemplo em que o ligante, após o experimento, permaneceu distante de sua posição inicial, conforme indica seu valor de RMSD = $11,1$ Å. Esse experimentou foi o executado com o *snapshot* 1188 da InhA e apresentou uma FEB de $-8,6$ *kcal/mol*.

Assim como aconteceu com o IQG607A, comparações entre o experimento exaustivo e seletivo do TCL mostram médias de energia e RMSD muito próximas (experimento seletivo: RMSD = $6,7 \pm 1,7$ Å FEB = $-8,9 \pm 0,3$ *kcal/mol* e experimento exaustivo: RMSD = $6,8 \pm 1,8$ Å FEB = $-8,9 \pm 0,3$ *kcal/mol*), indicando que o experimento seletivo representa adequadamente o exaustivo, o que demonstra a eficiência do critério de seleção de *snapshots* escolhida. Assim, a interação entre a InhA-TCL poderia ser analisada somente com o experimento seletivo, que despendeu 4 vezes menos tempo de execução.

- O *docking* do ligante ETH apresentou os menores, mas ainda aceitáveis, valores de energia livre de ligação (FEB = $-6,7 \pm 0,3$ *kcal/mol*). Entretanto os valores de RMSD e o grande desvio padrão observado (RMSD = $5,2 \pm 2,4$ Å) indicam que, por tratar-se de

uma molécula pequena (e, portanto com maior mobilidade), o ligante pode se movimentar bastante dentro do sítio de ligação. Este comportamento já era esperado uma vez que, para este ligante atuar como inibidor da enzima InhA, ele deve estar covalentemente ligado ao NADH [49]. Em muitos casos onde o RMSD apresentou-se mais elevado, o ETH ocupava uma posição que seria originalmente ocupada pelo NADH. A Figura 34 mostra a InhA juntamente com três diferentes posições do ETH. Em magenta está o ligante em sua posição inicial (a mesma em todos os experimentos de *docking*), em amarelo o ligante após um experimento com o *snapshot* 3078 da InhA (RMSD = $2,7$ Å e FEB = $-6,7$ *kcal/mol*) e em ciano o ligante em sua posição final após um experimento entre o mesmo e o *snapshot* 2 da InhA (RMSD = $12,6$ Å e FEB = $-6,2$ *kcal/mol*). Mais uma vez, a utilização seletiva dos *snapshots* no *docking* do ETH conseguiu amostrar as possíveis formas de ligação deste ligante à enzima InhA.

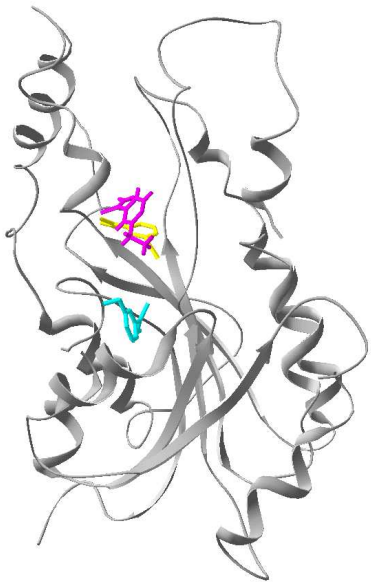


Figura 34 – Estrutura 3D da InhA e 3 posições finais do ETH: em magenta sua posição inicial, em amarelo a posição final com o *snapshot* 3078 e em ciano a posição final com o *snapshot* 2.

Uma síntese dos resultados obtidos por meio dos experimentos tanto exaustivos quanto seletivos pode ser vista nas Figuras 35 e 36. Os gráficos demonstram a eficiência do uso do *workflow* para execução dos experimentos de *docking* uma vez que obtiveram resultados aceitáveis para todos os experimentos, tanto os experimentos realizados no *cluster*, sem o uso do *workflow*, como os executados utilizando o *workflow* desenvolvido, mostrando que ambos são corretos e que o *workflow* executa corretamente de forma mais eficiente e robusta.

Ambos os gráficos foram feitos considerando os 1000 resultados dos experimentos seletivos e os 1000 melhores resultados de *docking* baseados na FEB final dos experimentos exaustivos. Ao reunir esses 1000 resultados de cada experimento, foram retirados aqueles cujos valores de FEB ou RMSD eram inapropriados para qualquer um dos experimentos (ou o experimento de

docking não convergiu para um valor mínimo, ou o valor de FEB era positivo), obtendo ao final, um total de 800 resultados considerados nos gráficos.

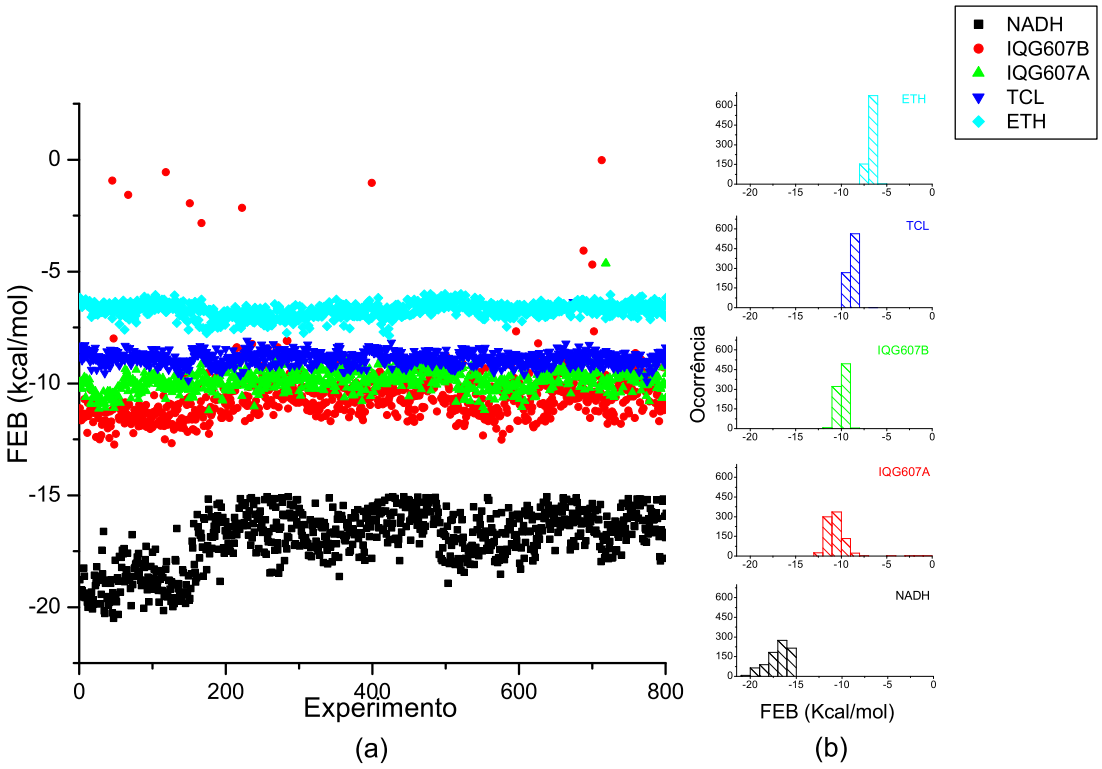


Figura 35 – Gráfi co FEB x histogramas da FEB somente dos 800 melhores valores de FEB de todos os experimentos.

O gráfico à esquerda da Figura 35a plota todos os valores de FEB de cada um dos experimentos. É possível ver faixas bem distintas de FEB. O experimento com o NADH, em preto no gráfico, apesar de apresentar uma FEB com maior variabilidade, conforme histograma à direita na Figura 35b, apresenta os melhores valores de FEB - os valores mais negativos - bem distanciados dos demais experimentos. O experimento com o IQG607A, em vermelho no gráfico, mostra uma faixa de FEB menos variável que o NADH, mas ainda bastante negativa, demonstrando que interage bem com a InhA. Seu histograma mostra uma concentração dos valores de FEB dos experimentos entre os valores -12 e -10 *kcal/mol*. A terceira faixa de valores, em verde no gráfico, corresponde aos valores de FEB resultantes da interação entre a InhA e o IQG607B, onde a grande maioria dos valores de FEB encontra-se entre -11 e -9 *kcal/mol*, variando muito pouco entre os experimentos. Os valores de FEB resultantes da interação entre a InhA e o TCL apresentam-se em uma faixa de energia maior (menos negativa), mas ainda aceitável. Seu histograma também indica uma baixa variabilidade, ou seja, as energias finais de *docking* dos experimentos são muito parecidas. A última faixa de valores, que corresponde ao ligante ETH interagindo com a InhA apresenta a menor média de energia. Seu histograma

concentra todas as ocorrências entre -8 e -6 *kcal/mol*, mas principalmente entre -7 e -6 *kcal/mol*, indicando que, entre os ligantes testados, esse foi o que interagiu de forma menos favorável com a InhA. Esse gráfico reproduz relação semelhante entre os valores de FEB observados experimentalmente [21], [48] e [50].

A Figura 36 compara os resultados finais de RMSD entre os diferentes experimentos de *docking*. O gráfico plota os valores de RMSD médios a cada 20 experimentos de *docking*, juntamente com o desvio padrão do RMSD encontrado nesse trecho (em barras verticais).

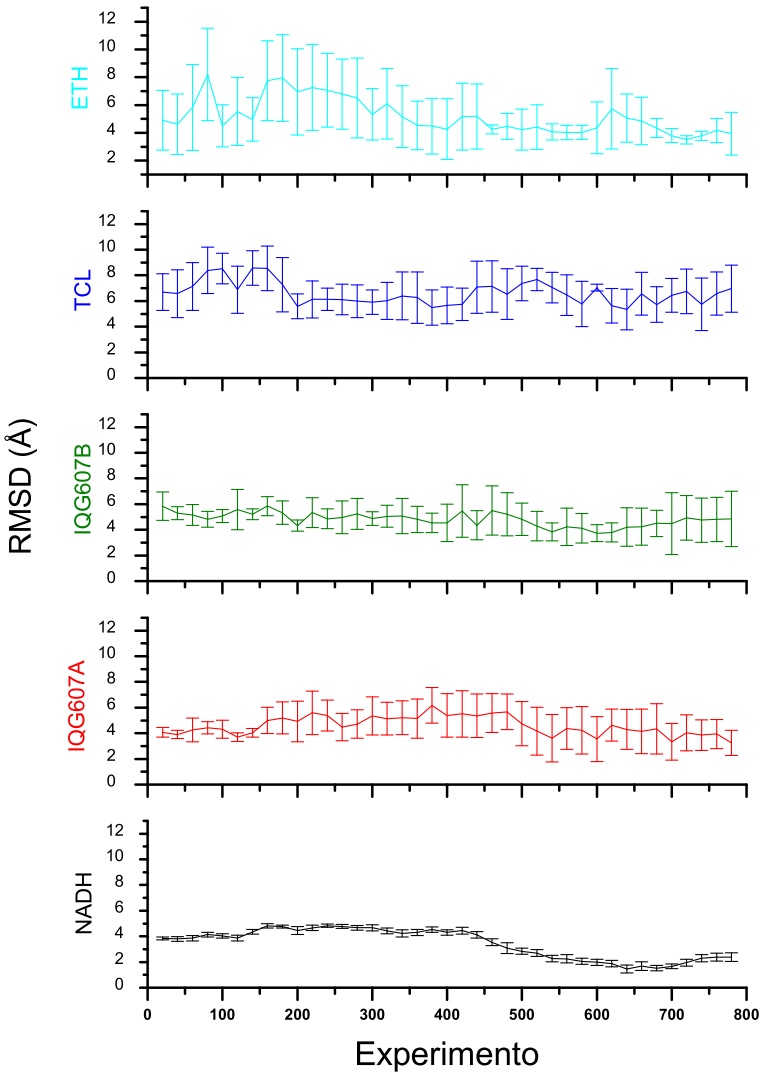


Figura 36 – Gráfico dos valores médios de RMSD a cada 20 experimentos juntamente com o desvio padrão, em barras verticais, em cada trecho.

É possível observar claramente que o experimento InhA-NADH mostra as menores variações de RMSD (menores valores de desvio padrão de RMSD), o que condiz com o esperado, pois se trata de uma molécula grande que não se movimenta muito dentro do sítio de ligação da InhA. O gráfico do IQG607B apresenta a segunda menor variação de desvio padrão do RMSD

entre os experimentos, estando de acordo com o esperado, já que essa é também uma molécula grande composta por 43 átomos, não conseguindo por isso se deslocar muito dentro do sítio de ligação da proteína. O gráfico do IQG607A mostra que o mesmo já apresenta uma maior variação de RMSD, pois trata-se de uma molécula menor que o NADH e IQG607B. Os gráficos do TCL e ETH mostram que estes ligantes são os que mais se movimentam dentro do sítio de ligação na proteína, apresentando tanto um maior valor de RMSD quanto maiores desvios padrões, especialmente o ETH que apresenta as maiores variações de desvio padrão.

5.4 Considerações Finais

Esse capítulo descreveu os experimentos de *docking* realizados assim como os resultados neles obtidos pela aplicação do *workflow* científico desenvolvido no processo de CADD. As análises destes resultados demonstraram que, mesmo aplicando somente esse primeiro critério de qualidade baseado nos melhores resultados de FEB, já foram alcançados resultados com experimentos seletivos que representam adequadamente interações proteína-ligantes.

Os resultados mostraram que o *workflow* executa corretamente os diferentes experimentos, tornando a parametrização de cada um deles uma tarefa simples e eficiente.

Se esse Estudo de Caso fosse realizado utilizando-se somente os *shell scripts* previamente desenvolvidos, os experimentos seletivos seriam realizados de maneira bem mais lenta e complicada ou, talvez, nem poderiam ser realizados. Se, a cada execução de um novo experimento, todos os 70 *shell scripts* tivessem que ser modificados pelo usuário, a análise dos resultados seria mais dificultada.

6 Conclusões

O presente trabalho apresentou o *workflow* científico desenvolvido para automatizar o processo de desenvolvimento de fármacos assistido por computador considerando a flexibilidade do receptor. A consideração dessa flexibilidade é importante para o processo de CADD pois as proteínas não permanecem rígidas em seu ambiente celular. A técnica selecionada para incluir essa flexibilidade da proteína nos experimentos de *docking* foi o uso de um conjunto de possíveis conformações do receptor geradas por meio de simulação por DM. Com a utilização do *workflow* desenvolvido, esse processo pode ser executado de forma mais simples, onde a parametrização de cada uma das etapas ocorre em tempo de execução e não há a necessidade da modificação prévia dos *shell scripts* e programas a cada nova interação receptor-ligante a ser testada.

Porém, a execução de experimentos de *docking* considerando determinado ligante e todas as conformações de determinado receptor necessita de um tempo considerável para ser concluído. Por exemplo, se forem utilizados 3000 *snapshots* da proteína e se o experimento for executado em uma máquina como um Pentium 4 de 3GHz com 1Gb de memória RAM, seriam necessários em média (considerando que cada experimento despendesse 8 minutos) 400 hs de execução ininterrupta. Por esse motivo, se tornou imprescindível o desenvolvimento de algum tipo de seleção dos *snapshots* de forma que nem todos precisassem ser utilizados.

A técnica de seleção de *snapshots* escolhida para ser utilizada nesse trabalho é baseada na energia livre de ligação (FEB). Dessa forma, após um experimento exaustivo, aqueles *snapshots* da macromolécula que resultaram nos melhores resultados de *docking* com determinado ligante são utilizados para o *docking* de outros ligantes pertencentes à mesma classe que o primeiro. Assim, somente são executados experimentos de *docking* considerando esses *snapshots* da proteína.

6.1 Principais Contribuições

As principais contribuições do presente trabalho foram:

- Com o *workflow* desenvolvido esse processo de desenvolvimento de fármacos assistido por computador utilizando receptor flexível passou a ser executado de forma simples e tornou a análise e armazenamento dos resultados mais eficiente e organizada. Assim, mais

experimentos podem ser realizados, onde qualquer usuário tem condições de parametrizar a execução do processo de acordo com suas necessidades;

- Redução do tempo necessário para a análise de uma interação receptor flexível - ligante com a inclusão de uma etapa de seleção de *snapshots* a esse processo;

- Com base nos resultados dos experimentos descritos no Estudo de Caso, o critério de seleção escolhido demonstrou-se eficiente para o *docking* dos ligantes eleitos para teste. Comparações entre os resultados de experimentos exaustivos e seletivos para um mesmo ligante (com, por exemplo, o IQG607A) mostram que a seleção representa adequadamente o conjunto completo de *snapshots*. Dessa forma, este trabalho mostra que, para ligantes de uma mesma classe, não é necessária a utilização de todos os *snapshots* da DM, pode-se apenas selecionar alguns *snapshots* para a representação do conjunto de conformações possíveis para a macromolécula. Portanto, o *workflow* desenvolvido ainda apresenta uma grande flexibilidade, pois pode ser aplicado tanto para a execução de *dockings* exaustivos quanto para *dockings* seletivos, sendo as duas maneiras executadas da mesma forma dentro do *Shark*.

6.2 Trabalhos Futuros

As principais atividades que se pretende implementar no trabalho já desenvolvido de forma a torná-lo mais eficiente são:

- Apesar dos experimentos realizados mostrarem que a seleção pela energia se mostra eficiente, utilizando-se ligantes de uma mesma classe, ainda são muitos *snapshots* que precisam ser utilizados (foram utilizados 1000 *snapshots* durante os experimentos seletivos). Se utilizarmos tantos *snapshots* em experimentos com milhares de ligantes, o tempo necessário para terminar a execução dos experimentos é muito grande (se para um ligante o tempo para executar o *docking* seletivo é em torno de 100 hs, se forem utilizados 100 ligantes, já são necessários 10000 hs ou 417 dias de processamento em um PC com uma configuração semelhante aos PC LABIO 1 e PC LABIO 2 utilizados nos nossos experimentos).

  Sendo assim, é essencial para que a flexibilidade do receptor seja considerada em muitos experimentos com diferentes ligantes, que os *snapshots* da mesma sejam selecionados de maneira mais eficiente, reduzindo os *snapshots* a serem utilizados em cada experimento. Assim, é necessário que sejam estudadas diferentes técnicas de seleção de *snapshots*. Uma possibilidade é aplicar a abordagem mostrada em Singh *et al.* [51] (utilizada na seleção de ligantes) à seleção de *snapshots*. Para isso, os resultados de experimentos exaustivos devem ser armazenados em um banco de dados. Durante o desenvolvimento

de um trabalho no LABIO foi desenvolvido um sistema que armazenava os resultados de experimentos de *docking* em um banco de dados MySQL. Esse sistema precisa ser aperfeiçoado para suportar muitos acessos e armazenamento de muitas informações. Após, utilizando esses dados armazenados e aplicando técnicas de mineração de dados, descobrir relações entre os *snapshots*, ou então, classificá-las em grupos de acordo com algum critério como, por exemplo, a presença de determinadas interações. A partir dessas classificações ou regras selecionar os *snapshots* que melhor representam a flexibilidade da proteína para determinada classe de ligantes;

- Fazer melhorias no *workflow* desenvolvido de forma a tornar alguns de seus parâmetros mais flexíveis. Essas melhorias serão úteis quando experimentos de *Virtual Screening* forem ser executados utilizando o *workflow* desenvolvido (execução de muitos experimentos considerando diferentes proteínas e/ou ligantes);
- Executar mais experimentações utilizando o *workflow* desenvolvido;
- Com os resultados das experimentações, analisar a qualidade dos critérios de seleção já desenvolvidos e os que serão implementados em atividades futuras.

Referências

[1] N. Mehta and R. H. Barter. Design document for jawe2openflow project. Technical Report UCRL-TR-206044, Lawrence Livermore National Laboratory (LLNL), University of California, 2004. Disponível em: http://www.llnl.gov/tid/lof/documents/pdf/310223.pdf, acessado em junho de 2006.

[2] N. M. Luscombe, D. Greenbaum, and M. Gerstein. What is Bioinformatics? a Proposed Definition and Overview of the Field. *Methods of Information in Medicine*, 4:346–358, 2001.

[3] M. Chagoyen, M. Kurul, P. De-Alarcón, J. Carazo, and A. Gupta. Designing and Executing Scientific Workflows with a Programmable Integrator. *Bioinformatics*, 20:2092–2100, 2004.

[4] J. Wainer, G. V. Weske, and C. B. Medeiros. Scientific Workflow Systems. In *Proceedings of the NFS Workshop on Workflow and Process Automation in Information Systems: State-of-the-art and Future Directions*, pages 1–5, Athens, Georgia, USA., 1996.

[5] A. R. Leach. Ligand docking to proteins with discrete side-chain flexibility. *J. Mol. Biol.*, 235:345–356, 1994.

[6] D. Goodsell, G. Morris, and A. Olson. Docking of Flexible Ligands: Applications of AutoDock. *J. Mol. Recognit.*, 9:1–5, 1996.

[7] M. Rarey, B. Kramer, T. Lengauer, and G. Klebe. A Fast Flexible Docking Method Using an Incremental Construction Algorithm. *J. Mol. Biol.*, 261:470–489, 1996.

[8] T. J. A. Ewing, S. Makino, A. G. Skillman, and I. D. Kuntz. DOCK4.0: Search Strategies for Automated Molecular Docking of Flexible Molecule Databases. *J. Comp. Aided. Mol. Des.*, 15:411–428, 2001.

[9] J. Kua, Y. Zhang, and A. McCammon. Studying Enzyme Binding Specificity in Acetylcholinesterase Using a Combined Molecular Dynamics and Multiple Docking Approach. *J. Am. Chem. Soc.*, 124:8260–8267, 2002.

[10] D. A. Case, T. E. Cheatham III, T. Darden, H. Gohlke, R. Luo, K. M. Merz Jr., A. Onufriev, C. Simmerling, B. Wang, and R. Woods. The AMBER Biomolecular Simulation Programs. *J. Computat. Chem.*, 26:1668–1688, 2005.

[11] M. Karplus and A. McCammon. Molecular Dynamics Simulation of Biomolecules. *Nat. Struct. Biol.*, 9:646–652, 2002.

[12] C. Watson. Interview: Yike Guo and Jonathan Sheldon of InforSense Discuss the Impact of Workflow Technology on Drug Discovery. *Drug Discov. Today*, 10:1211–1212, 2005.

[13] I. D. Kunyz. Structure-based Strategies for Drug Design and Discovery. *Science*, 257:1078–1082, 1992.

[14] T. P. Lybrand. Ligand-Protein Docking and Rational Drug Design. *Curr. Opin. Struct. Biol.*, 5:224–228, 1995.

[15] J. Bajorath. Virtual screening in drug discovery: Methods, expectations and reality. *Curr. Drug Discov.*, 3:25–28, 2002.

[16] J. Drews. Drug discovery: A historical perspective computational methods for biomolecular docking. *Curr. Opin. Struct. Biol.*, 6:402–406, 1996.

[17] G. K. Farber. New approaches to rational drug design. *Pharmacol. Ther.*, 84:327–332, 1999.

[18] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P.E. Bourne. PDB - Protein Data Bank. *Nucl. Acids Res.*, 28:235–242, 2000.

[19] J. J. Irwin and B. K. Shoichet. ZINC - A Free Database of Commercially Available Compounds for Virtual Screening. *J. Chem. Inf. Model.*, 45(1):177–182, 2005.

[20] T. Lengauer and M. Rarey. Computational methods for biomolecular docking. *Curr. Opin. Struct. Biol.*, 6:402–406, 1996.

[21] A. Dessen, A. Quemard, J. S. Blanchard, W. R. Jacobs, and J. C. Sacchettini. Crystal Structure and Function of the Isoniazid Target of Mycobacterium tuberculosis. *Science*, 267:1638–1641, 1995.

[22] J. Desmet, I. A. Wilson, M. Joniau, M. DeMaeyer, and I. Lasters. Computation of the binding of fully flexible peptides to proteins with flexible side chains. *FASEB J.*, 11:164–172, 1997.

[23] M. Zacharias and H. Sklenar. Harmonic Modes as variables to approximately account for receptor flexibility in ligand-receptor docking simulations: application to DNA minor groove ligand complexes. *J. Comput. Chem.*, 20:287–300, 1999.

[24] H. A. Carlson. Protein flexibility is an important component of structure-based drug discovery. *Curr. Pharm. Des.*, 8:1571–1578, 2002.

[25] J-H. Lin, A. L. Perryman, J. R. Schames, and J. A. McCammon. Computational drug design accommodating receptor flexibility: the relaxed complex scheme. *J. Am. Chem. Soc.*, 124:5632–5633, 2002.

[26] A. Sali. 100.000 Protein Structures for the Biologist. *Nature Struct. Biol.*, 5:1029–1032, 1998.

[27] W.F. van Gunsteren and H. J. C. Berendsen. Computer Simulation of Molecular Dynamics: Methodology, Applications and Perspectives in Chemistry. *Angew. Chem. Int. Ed.*, 29:992–1023, 1990.

[28] E. K. Schroeder. *Análise Computacional da Enzima 2-trans-Enoil-ACP (CoA)* Reductase *de* Mycobacterium tuberculosis*, Produto do Gene InhA, Como Alvo para o Desenvolvimento de Drogas Anti-tuberculose*. PhD thesis, UFRGS, Porto Alegre, Brazil, 2004. (In Portuguese).

[29] The Workflow Management Coalition. Terminology and Glossary, 1999. Disponível em http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf, acessado em dezembro de 2005.

[30] G. A. Bolcer and R. N. Taylor. Advanced Workflow Managment Technologies. *Software Process: Improvement and Practice*, 4:125–171, 1998.

[31] R. Y. Santos. Ambiente 10+C para Definição e Execução de *Workflows in silico* Através de Serviços Web. Master's thesis, UFRJ, Rio de Janeiro, Brazil, March 2004. (In Portuguese).

[32] L. Meyer, S. Rössle, P. Bisch, and M. Mattoso. Parallelism in Bioinformatics Workflows. In *Proceedings of the International Conference VECPAR*, pages 705–718, Valencia, Spanish, 2004.

[33] M. Weske, G. Vossen, and C. Medeiros. Scientific Workflow Management: WASA Architecture and Applications. In *Proceedings of 6th DEXA Conference*, pages 574–583, London, England, 1995.

[34] Enhydra Shark Homepage. Disponível em http://forge.objectweb.org/projects/shark/, acessado em junho de 2006.

[35] N. Guex and M.C. Peitsch. SWISS-MODEL and the Swiss-PdbViewer: An Environment for Comparative Protein Modeling. *Electrophoresis*, 18:2714–2723, 1997.

[36] S. A. Barretto, J. Warren, and A. Goodchild. Designing Guideline-based Workflow-enabled Electronic Health Records. In *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 6*, page 60135.2, Hawaii, USA, 2004. IEEE.

[37] J. Shin, J. Hammer, and W. J. O'Brien. Distributed Process Integration: Experiences and Opportunities for Future Research. In *Proceedings of the IEEE International Workshop on Web and Mobile Information Systems (WAMIS 2006)*, pages 247–251, Vienna, Austria, 2006. IEEE.

[38] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao. Scientific Workflow Management and the Kepler System. *Concurrency and Computation: Practice and Experience*, 18(10):1039–1065, 2005.

[39] D. Georgakopoulos, M. Hornick, and A. Sheth. An Overview of Workflow Management. Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, 3:119–353, 1995.

[40] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and D. L. Wheeler. NAR Molecular Biology Database. GenBank. *Nucleic Acids Res.*, 33:D34–8, 2005.

[41] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: a New Generation of Protein Database Search Programs. *Nucleic Acids Res.*, 25:3389–3402, 1997.

[42] Chemical Computing Group, Inc. Montreal, Quebec, Canada. Molecular Operating Environment (MOE 2004.03). Disponível em http://www.chemcomp.com, acessado em setembro de 2006.

[43] AutoDock User's Guide - AutoDock: Automated Docking of Flexible Ligands and Receptors. Version 3.0.5., 2001. Disponível em http://autodock.scripps.edu/help-center/manual/autodock-3-user-s-guide/AutoDock3.0.5_UserGuide.pdf, acessado em julho de 2006.

[44] E.K. Schroeder, L.A. Basso, D.S. Santos, and O. Norberto de Souza. Molecular Dynamics Simulation Studies of the Wild-Type, I21V, and I16T Mutants of Isoniazid-Resistant Mycobacterium tuberculosis Enoyl Reductase (InhA) in Complex with NADH: Toward the Understanding of NADH-InhA Different Affinities. *Biophys. J.*, 89:876–884, 2005.

[45] J. L. Henning. SPEC CPU2000: Measuring CPU Performance in the New Millennium. *Computer*, 33(7):28–35, 2000.

[46] L. A. Basso, R. Zheng, J. M. Musser, W. R. Jacobs Jr., and J. S. Blanchard. Mechanisms of isoniazid resistance in Mycobacterium tuberculosis: enzymatic characterization of enoyl reductase mutants identified in isoniazid-resistant clinical isolates. *J. Infect.Dis.*, 178:769–775, 1998.

[47] W. Humphrey, A. Dalke, and K. Schulten. VMD - Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[48] M. R. Kuo, H. R. Morbidoni, D. Alland, S. F. Sneddon, B. B. Gourlie, M. M. Staveski, M. Leonard, J. S. Gregory, A. D. Janjigian, C. Yee, J. M. Musser, B. Kreiswirth, H. Iwamoto, R. Perozzo, W. R. Jacobs, J. C. Sacchettini, and D. A. Fodock. Targeting Tuberculosis and Malaria through Inhibition of Enoyl Reductase: Compound Activity and Structural Data. *J. Biol. Chem.*, 278(23):20851–20859, 2003.

[49] D. A. Rozwarski, G. A. Grant, D. H. Barton, W. R. Jacobs Jr., and J. C. Sacchettini. Modification of the NADH of the Isoniazid Target (InhA) from Mycobacterium tuberculosis. *Science*, 279:98–102, 1998.

[50] J. S. Oliveira, E. H. S. Sousa, L. A. Basso, M. Palaci, R. Dietze, D. S. Santos, and I. Moreira. An Inorganic Iron Complex that Inhibits Wild-type and an Isoniazid-resistant Mutant 2-trans-enoyl-ACP (CoA) Reductase from Mycobacterium tuberculosis. *Chem. Commun.*, 15:312–313, 2004.

[51] J. Singh, Z. Deng, G. Narale, and C. Chuaqui. Structural Interaction Fingerprints: A New Approach to Organizing, Mining, Analyzing, and Designing Protein-Small Molecule Complexes. *Chem. Biol. Drug. Des.*, 67:5–12, 2006.