Artur Duque Rossi

Refinamento Manual e Automático de Modelos Tridimensionais de Proteínas para o *Workflow* Científico MHOLline

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional, da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do grau de Mestre em Modelagem Computacional.

Orientadora: Profa. D.Sc. Priscila Vanessa Zabala Capriles Goliatt

Juiz de Fora

2017

Artur Duque Rossi,

Refinamento Manual e Automático de Modelos Tridimensionais de Proteínas para o *Workflow* Científico MHOLline/ Artur Duque Rossi. – Juiz de Fora: UFJF/MMC, 2017.

XIV, 119 p.: il.; 29, 7cm.

Orientadora: Priscila Vanessa Zabala Capriles Goliatt

Dissertação (mestrado) – UFJF/MMC/Programa de Modelagem Computacional, 2017.

Referências Bibliográficas: p. 95 – 111.

1. Modelagem de Proteínas. 2. Fluxo de dados. 3. MHOLline. I. Priscila Vanessa Zabala Capriles Goliatt, . II. Universidade Federal de Juiz de Fora, MMC, Programa de Modelagem Computacional.

Artur Duque Rossi

Refinamento Manual e Automático de Modelos Tridimensionais de Proteínas para o *Workflow* Científico MHOLline

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional, da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do grau de Mestre em Modelagem Computacional.

Aprovada em 24 de Fevereiro de 2017.

BANCA EXAMINADORA

Profa. D.Sc. Priscila Vanessa Zabala Capriles Goliatt - Orientadora
Universidade Federal de Juiz de Fora

Prof. D.Sc. Laurent Emannuel Dardenne
Laboratório Nacional de Computação Científica

Prof. D.Sc Ciro de Barros Barbosa
Universidade Federal de Juiz de Fora

Dedico este trabalho aos meus parentes, presentes ou não, pois sem eles seria impossível chegar até onde cheguei.

AGRADECIMENTOS

Gostaria de agradecer primeiramente aos meus pais, que sempre me apoiaram, me fazendo mover adiante. Obrigado à minha querida Misae, por me dar muito apoio durante os momentos difíceis, sem dúvida você fez toda diferença para mim. Obrigado ao meu irmão Átila, pelo apoio na área científica desde quando éramos pequenos, incrível onde os experimentos nos levaram! Obrigado à Nilza e família por me acolher durante todo o período de Mestrado. Obrigado ao Programa de Pós-Graduação em Modelagem Computacional da Universidade Federal de Juiz de Fora, que me recebeu de braços abertos. Obrigado ao Victor, que refatorou o código do MHOLline e criou muitas interfaces no MHOLline 2.0, além de conseguir debugar meus códigos. Obrigado ao Ruan por ajudar na criação da biblioteca de agrupamento. Obrigado ao Vinicius por ter me ajudado nos códigos, ajudando a *debugar* o meu código, Obrigado a todos os quais eu dividi sala, pois sempre se dispuseram a tirar muitas dúvidas que tive. Obrigado à minha orientadora Priscila, que durante este tempo todo de mestrado me auxiliou nesta jornada complicada, e apesar de todos os contratempos se manteve calma e focada no objetivo, me ensinando não somente conteúdo para o trabalho, mas como ser, caso um dia eu seja, um orientador acadêmico. Gostaria de agradecer à FAPEMIG, que acreditou no meu potencial, financiando toda a minha pesquisa.

"A tarefa não é tanto ver aquilo que ninguém viu, mas pensar o que ninguém ainda pensou sobre aquilo que todo mundo vê."(Arthur Schopenhauer)

RESUMO

O MHOLline é um *workflow* científico voltado para a modelagem e análise de proteínas, atendendo a pesquisadores de diversas áreas, como Bioinformática, Biofísica, Químicos Computacionais e Biólogos Computacionais. Este projeto, iniciado em 2004 como um *software* de uso local, tornou-se um serviço *web* em 2010, através da parceria da *Universidade Federal do Rio de Janeiro* (UFRJ) com o *Laboratório Nacional de Computação Científica* (LNCC), o qual pode ser acessado pelo endereço *web* `http://www.mholline.lncc.br`. Em 2013, uma parceria com a *Universidade Federal de Juiz de Fora* deu início ao projeto do MHOLline 2.0, disponível no endereço *web* `http://www.mholline2.lncc.br`, que conta com adições de *softwares*, uma interface completamente nova e uma área de refinamento de resultados para usuários logados. A área do refinamento de resultados oferece a possibilidade aos usuários de adicionar ou trocar o molde da proteína modelada, criar restrições de estrutura secundária no Modeller, clivar regiões de peptídeo sinal e otimizar *loops* no Modeller, tudo de forma automática, dispensando a necessidade do usuário gerar qualquer *script* manualmente. Caso o usuário deseje é possível refinar a proteína automaticamente, através do uso de ferramentas de inteligência artificial para classificar os resultados gerados com as opções de restrição modeladas, em grupos, visando reduzir o trabalho de analisar os resultados finais do refinamento. Neste trabalho, apresentamos também uma nova proposta para agrupamento de modelos de proteínas baseado em um conjunto de atributos relacionados com a sua qualidade (*e.g.* energia e estrutural). Ao usuário, além dos grupos de estruturas com qualidades similares, também é retornada a estrutura representativa de cada grupo, com o objetivo de auxiliar na tomada de decisão de qual ou quais modelos seguirão para os próximos estudos.

Palavras-chave: Modelagem de Proteínas. Fluxo de dados. MHOLline.

ABSTRACT

MHOLline is a scientific workflow designed to model and analyze proteins, reaching researchers in domains of Bioinformatics, Biophysics, Computational Chemists and Computational Biologists. This project started in 2004 as a local software and became a web service in 2010 (available at `http://www.mholline.lncc.br`), through the partnership between the *Universidade Federal do Rio de Janeiro* (UFRJ) and *Laboratório Nacional de Computação Científica* (LNCC). In 2013, a new partnership with *Universidade Federal de Juiz de Fora* started the development of MHOLline 2.0, now available at `http://www.mholline2.lncc.br`. This version presents a new interface and a refinement area to logged users, offering the possibility to add or modify the template of the protein, remove signal peptides and restrict secondary structures and optimize protein loops on Modeller. All can be done in an automatic way, dispensing the user to manually generate any script. The user can also refine the protein automatically trough the use of artificial intelligence tools classifying the generated results with a set of restrictions in groups, aiming to reduce the effort to analyze the final refinement results. In this work, we also present a new proposal for clustering protein models based on a set of attributes related to their quality (*i.e.*, energy and structural quality). To the user, in addition to the groups of structures with similar qualities, is also returned the representative structure of each group, in order to assist in the decision making of which model or models will follow for the future studies.

Keywords: Protein Modelling. Workflow. MHOLline.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

1 INTRODUÇÃO

No ano 2000, um comitê, organizado pelo *Biomedical Information Science and Technology Initiative Consortium* (BISTIC) definiu as áreas de Bioinformática e Biologia Computacional [1]. As definições foram:

- Bioinformática: Pesquisa, desenvolvimento, ou aplicação de ferramentas computacionais e suas abordagens para expandir o uso de dados biológicos, médicos, comportamentais ou de saúde, incluindo o processo de aquisição, armazenamento, organização, arquivamento, análise, ou visualização de tais dados.

- Biologia Computacional: Desenvolvimento e aplicação de dados analíticos e métodos teóricos, modelagem matemática e técnicas de simulação computacional para estudar sistemas biológicos, comportamentais e sociais.

Apesar de sua categorização tardia, pesquisas nestas áreas já ocorriam em meados da década de 60, modelos para seleção livre de evolução molecular [2], substituição preferencial de aminoácidos em sequências proteicas [3], dentre outros. Na década de 70 foram criados os primeiros algoritmos de alinhamento de sequências [4] e de predição de estrutura secundária de proteínas [5] além do uso de sistemas especializados para modelagem de proteínas [6]. Nesta mesma década, no domínio da dinâmica molecular, foram executados estudos de investigação de estados do potencial energético de superfícies [7], estudos teóricos de reações enzimáticas [8] e estudos de representação simplificada de conformações proteicas para simulações rápidas de enovelamento proteico [9].

Na década de 80, pela primeira vez algoritmos eficientes foram desenvolvidos para processar e analisar um grande volume de dados. Também deu-se início aos estudos de *docking* molecular, um método para predizer a orientação provável que determinada molécula deve se ligar a outra de maneira a formar um complexo estável[10]. Esta técnica é mais comumente aplicada em uma proteína e um ligante (em sua maioria um inibidor), de forma a descobrir quais ligantes apresentam maior afinidade [11, 12] pela proteína. Os avanços na área de predição de estruturas secundárias se deram pela adição de métodos de aprendizado de máquina como máquinas de vetor suporte (SVM) [13] e redes neurais [14].

Na década de 90, com a expansão da internet, a bioinformática e biologia computacional começaram a se difundir mais rapidamente, possibilitando o compartilhamento de dados e informações, bem como o acesso a diversos algoritmos [6]. Nos anos 2000 houve o sequenciamento completo do organismo *Pseudomonas aeruginosa* [15] e o projeto genoma humano [16] que impulsionaram ainda mais os estudos nessas áreas. Adicionalmente, com o avanço dos conhecimentos em métodos de modelagem *in silico*[1], estes se mostraram um meio viável para se gerar modelos de qualidade que possam ser utilizados para estudos mais aprofundados, sendo uma alternativa real para os métodos experimentais.

Neste trabalho, apresentaremos um processo de automação do uso de ferramentas de bioinformática e biologia computacional para o estudo em larga escala de proteínas. Esse processo foi organizado como um *workflow* que integra um conjunto de ferramentas já existentes e novas (desenvolvidas nessa dissertação) para esse tipo de aplicação, sendo disponibilizado para a comunidade científica como um serviço *web* gratuito.

[1] Conduzidos ou produzidos através de modelagem computacional ou simulação computacional.

2 *Workflow*

Os *workflows* nasceram no mundo dos negócios, sendo uma forma de gerenciar e organizar tarefas e processos nas empresas. Os *workflows* voltados para negócios se concentram na segurança e integridade da sequência de ações a serem tomadas [17]. Em 1996, a WfMC (*Worflow Management Coalition*) definiu *workflow* como:

> "A automação de processos de negócios, no todo ou em partes, no qual documentos, informações ou tarefas são passadas de uma para outra ação, de acordo com o conjunto de regras de procedimentos" [18].

Os *workflows* podem ser divididos da seguinte forma [19]:

- *Workflow Ad hoc*: São fluxos que necessitam da intervenção humana para tomada de decisões constantes.
- *Workflow* de Produção: Envolve processos repetitivos e previsíveis com regras complexas de coordenação de tarefas, envolvendo acessos múltiplos sistemas de informação. A automatizaçao deste tipo de *workflow* é possível, porém complexa, visto que a complexidade do fluxo de dados é alta.
- *Workflow* Administrativo: Envolve processos repetitivos e previsíveis com regras simples de coordenação de tarefas. A automatização deste tipo de *workflow* é simples, devido ao fato de que a complexidade do fluxo de dados é baixa.

Apesar de *workflows* científicos terem sua origem nos *workflows* de negócios, eles possuem a característica de serem dinâmicos e adaptáveis, visto que acompanham os avanços científicos, enquanto que os *workflows* voltados para negócios são menos mutáveis, sendo representados por uma série de tarefas ou diretrizes de uma organização [20].

Nesse contexto, aplicando-se as técnicas de fluxo de processos em um ambiente científico, pode-se dizer que o *workflow* científico é a automatização do processamento de dados científicos, através da organização sistemática desses dados e do seus processamentos com o uso de *softwares* [21]. Um dos pontos positivos de um *workflow* científico é a sua flexibilidade, permitindo alterações no fluxo de execução de forma clara. Outro ponto

positivo é a possibilidade de se realizar a divisão do trabalho em uma equipe de maneira simples, visto que cada indivíduo ou grupo de indivíduos, pode criar e gerenciar um conjunto de processos, havendo apenas a preocupação de compatibilidade da entrada e saída dos resultados. Além disso, o *workflow* científico ainda pode ser executado de forma paralela, caso os processos sejam independentes.

Workflows científicos, ao serem aplicados em uma determinada área, fazem uso de um Modelo Computacional, que é um conjunto de regras e passos que são utilizados para se realizar uma tarefa implementados em um ambiente computacional. De acordo com os resultados gerados por um Modelo Computacional é possível obter informações comportamentais do sistema, seja um comportamento passado, presente ou futuro [22]. A escolha do Modelo Científico que vai definir as regras e passos do sistema de estudo, é uma etapa importante que deve ser respondida, antes de se começar a pensar em uma solução, pois é através desta escolha que o sistema será resolvido corretamente ou não. Para esta escolha deve-se levar em conta algumas questões [22]:

- Existe uma dependência do tempo para este *Workflow*? Alguns *workflows* dependem da variável tempo, como por exemplo um que descreva os movimentos de corpos celestes depende intrinsecamente do tempo, pois em momentos diferentes, dois corpos celestes distintos podem ocupar o mesmo espaço, porém não ao mesmo tempo, pois isso violaria o princípio da impenetrabilidade, invalidando o sistema, já que ele não simularia a realidade de uma forma fiel. Já outros não dependem do tempo, como uma compilação de um arquivo de código, o tempo pode ser medido, porém ele não desempenha um papel crucial na execução deste *workflow*.

- Caso o *workflow* não seja dependente do tempo, ele executa uma transformação de dados simples com taxas de produção e consumo constantes? Uma transformação simples de dados não faz uso de lógicas complexas ou execuções em paralelo e a taxa de produção e consumo constante denota que o *workflow* possuirá sempre um número fixo de dados de entrada e saída. Um exemplo de um *workflow* que executa uma transformação simples de dados é a conversão de um arquivo em outro, como por exemplo a conversão de um vídeo no formato .avi para o formato .mp4.

- Caso o *workflow* seja dependente do tempo ele é descrito por uma equação diferencial? Equações diferenciais geralmente são utilizadas para sistemas dinâmicos, variando no

tempo de forma contínua. Um exemplo de *workflow* descrito por equação diferencial é um *workflow* que descreve o processo de endurecimento do aço através do seu aquecimento e resfriamento.

Uma vez que estas perguntas sejam respondidas, é possível determinar Um Modelo Computacional adequado para o *Workflow* de forma que as regras e passos que serão seguidos durante a execução do *workflow* reproduzam o sistema a ser estudado de forma fiel o suficiente para que gerem dados confiáveis e úteis.

Workflows científicos podem ser criados de diversas formas, podendo-se utilizar *softwares* gerenciadores de *workflow*, como Taverna [23] ou Kepler [24]. Estes *softwares* podem auxiliar no processo de criação de *workflows* através de uma interface gráfica, fazendo com que o usuário muitas vezes não precise possuir conhecimento de programação, sendo portanto bem abrangente quanto ao público alvo. Também é possível a criação de um *workflow* científico através da programação, como um serviço *web*, que consiste de um conjunto de *softwares* unidos por um ou mais *scripts* no *backend* de um servidor.

Em sua dissertação de mestrado, Bonifácio [22] definiu o *Workflow* científico em três categorias: o Projeto, a Execução e a Proveniência. O Projeto trata da organização e da montagem do *workflow* científico, a Execução diz respeito ao processamento de dados durante a execução do *workflow* científico e a Proveniência aborda as funcionalidades presentes após a execução de um *wofkflow* científico.

2.1 *Workflows* científicos no auxílio de análise de dados biológicos

Workflows científicos podem ser aplicados em diversas áreas, como no imageamento sísmico [25], astronomia [26], exploração de petróleo [27] e também na área de bioinformática, como por exemplo, no auxílio no processo de descobrimento de novas drogas [28], ou *workflows* mais simples, como os *workflows* criados por DE PARIS em 2008 [29] para realizar tarefas como a busca de GI (*GenInfo Identifier*) *Number*, que são números identificadores de sequências depositadas no NCBI [30], a geração de um arquivo de sequência no formato `.fasta` a partir de um GI *Number* e a obtenção de um arquivo do tipo `.pdb` do banco de dados PDB a partir do identificador da proteína.

Durante a criação do MHOLline 1.0 e 2.0, estudos sobre a utilização de sistemas de gerenciamento de *workflows* como o Taverna e o Kepler para gerenciar o MHOLline foram feitos, porém a complexidade do *workflow* do MHOLline tornou a sua implementação não trivial, optando-se pela criação de um *web service*, para o MHOLline. No serviço *web* do MHOLline utilizam-se as linguagens HTML, CSS e Javascript, como *frontend* e as lingagens PERL, Python, C++, *ShellScript* e PHP como *backend*. Sendo então, um exemplo de *workflow* científico voltado para grandes quantidades de dados biológicos. O MHOLline pode ser acessado em sua versão 1.0 no endereço `www.mholline.lncc.br` e na sua versão 2.0 em `www.mholline2.lncc.br`. Nos capítulos 4 e 6 o *workflow* MHOLline será melhor detalhado.

3 PREDIÇÃO DA ESTRUTURA TRIDIMENSIONAL DE PROTEÍNAS

Visto que a estrutura tridimensional (3D) de uma proteína é um elemento fundamental (ou até determinante) em muitos processos biológicos e que a sua função está diretamente relacionada à sua conformação [31], o estudo e a obtenção da estrutura 3D de proteínas é essencial para a elucidação de diversos problemas biológicos. A seguir serão apresentados os principais métodos para a obtenção da estrutura 3D de proteínas.

3.1 A importância de se conhecer a estrutura tridimensional de uma proteína

Acredita-se que a evolução tende a conservar as funções de proteínas mais pela sua estrutura terciária que pela primária [32]. Com isso, um alinhamento entre uma proteína desconhecida e uma de referência que possuem estruturas tridimensionais (3D) muito semelhantes, mesmo que no sítio ativo existam poucas diferenças, pode ser um meio confiável para se inferir a atividade desempenhada pela proteína desconhecida [33].

A modelagem de proteínas possui diversas aplicações, dentre elas a de fornecer informações importantes para o desenho racional de fármacos. Em 2014, a predição experimental de novos compostos candidatos a fármacos custava aproximadamente US$2,6 bilhões [34]. Nesta temática, as técnicas de modelagem *in silico* ajudam à reduzir o tempo e o custo de pesquisas voltadas ao desenvolvimento de novas terapias através da obtenção de informações sobre a estrutura 3D de uma proteína de interesse farmacológico.

Com relação ao conhecimento de novas proteínas, atualmente, os sequenciadores provêm aos pesquisadores uma quantidade crescente de sequências proteicas de vários organismos. Porém, a obtenção da estrutura 3D destas sequências por métodos experimentais é limitada pelo alto custo financeiro, pelo tempo e por desafios intrínsecos aos métodos aplicados.

Levando em conta estas dificuldades, a modelagem através de métodos teóricos é uma saída válida, visto que, através de um custo financeiro e tempo menores é capaz de se gerar modelos satisfatórios[35].

3.2 Modelagem experimental

Através de análises experimentais de propriedades físico-químicas, determina-se as estruturas 3D de biomoléculas. Essas estruturas resolvidas através de experimentos são depositadas em bancos de dados de proteínas com estruturas resolvidas, como o *Protein Data Bank* (PDB) [36]. Existem diferentes técnicas para a predição experimental da estrutura 3D de proteínas (*e.g.*, difração de Raios-X, drifração de Nêutrons, epectroscopia por ressonância magnética nuclear, crio-microscopia eletrônica e difração de nêutrons), sendo estas brevemente descritas a seguir.

3.2.1 Difração de Raios-X

Neste método, um feixe de Raios-X (RX) incide sobre uma proteína cristalizada que os refrata sobre um filme ou um detector produzindo um padrão de pontos. Os tamanhos e ângulos dos pontos são armazenados e a rotação do cristal para mais feixes de RX é realizada até que se possuam imagens suficiente para que, através de um *software*, seja possível construir um modelo tridimensional da proteína [37].

A técnica de difração de Raios-X pode ser separada em três etapas [38]:

- Obtenção do cristal da proteína. Esta etapa é considerada uma das mais difíceis, pois nem sempre a proteína pode ser cristalizada, e quando pode ser cristalizada, nem sempre o cristal possui um grau de pureza e organização satisfatório. O cristal deve ser suficientemente grande (geralmente maior que 1mm nas três dimensões), puro na composição e regular na forma, sem rachaduras internas ou geminações, como exemplificado na Figura 3.1.

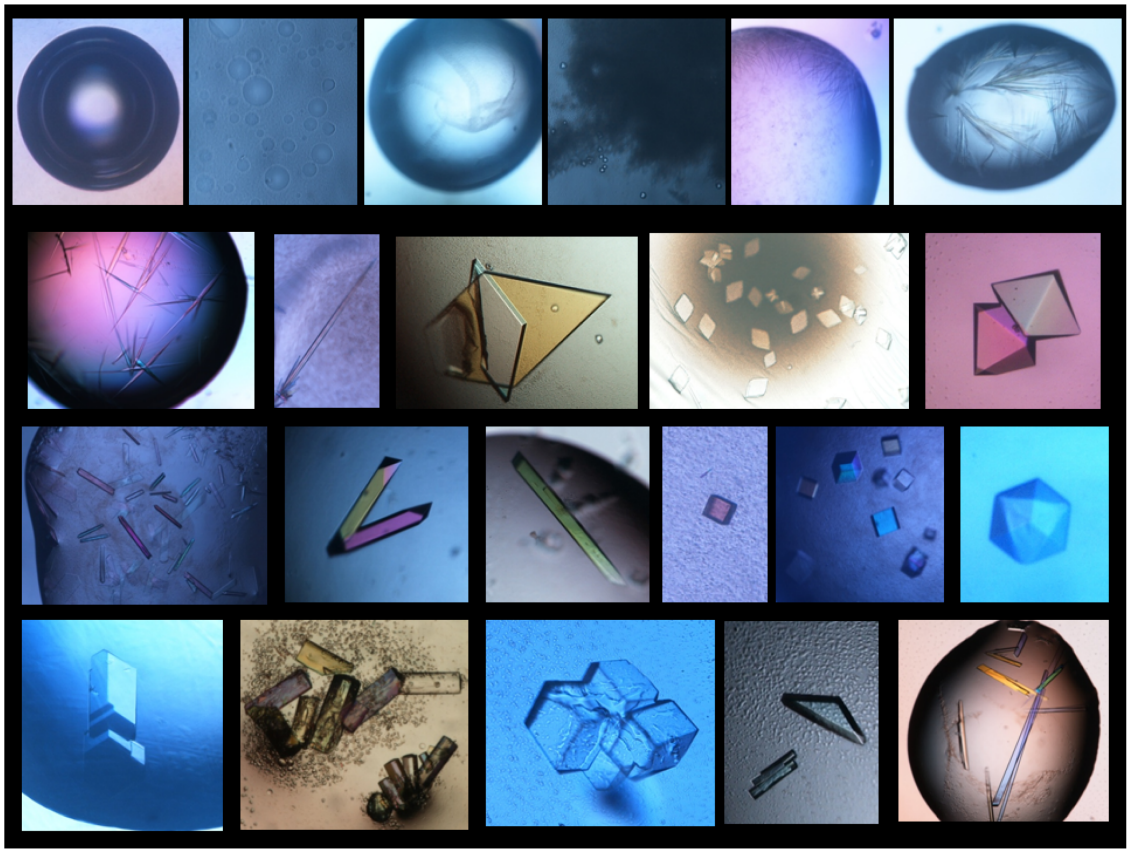


Figura 3.1: Exemplos de gotas de cristalização. Na primeira linha, da esqueda para a direita, gota límpida, óleo, precipitado leve, precipitado pesado, quasi-cristal, agrupamento de cristais agulha. As três últimas linhas são cristais bem formados de diversas conformações. Extraído de Krauss *et al.* (2013) [39].

- O cristal é bombardeado por feixes de Raios-X, que são difratados, incidindo sobre uma superfície detectora, formando o padrão de difração. Então repete-se o bombardeamento de dezenas a milhares de vezes para poder se construir a representação tridimensional da proteína.

- Os dados coletados na etapa anterior são combinados computacionalmente com informações químicas complementares para a produção de um modelo dos átomos no cristal. Este resultado de dados geralmente é alocado em um banco de dados de proteínas público, como o *Protein Data Bank* [36].

Porém, caso a proteína seja muito pequena (cerca de menos de 100 átomos), os átomos se tornam tão bem resolvidos que acabam sendo interpretados como nuvens de regiões separadas de densidade de elétrons. Por outro lado, estruturas muito grandes (cerca de dezenas de milhares de átomos) os resultados das difrações se tornam complexos demais

para o *software* interpretador. por tubos de densidade eletrônica. Ambos problemas resultam em estruturas não precisas [40].

Outro problema desta técnica é a necessidade de se ter proteínas em forma de cristal, uma vez que nem todas as proteínas podem ser cristalizadas, o que reduz a abrangência deste método [37]. Porém, existem esforços para se resolver compostos não cristalinos [41] que podem fazer com que esta última restrição diminua.

3.2.2 Difração de Nêutrons

Este método é muito semelhante ao de difração por RX, a diferença entre eles é que o fóton de RX interage com a nuvem de elétrons dos átomos, já que este possui carga, enquanto que o nêutron, interage com o núcleo dos átomos, pois não possui carga. Isto faz com que átomos com nuvens de elétrons pequenas, como o Hidrogênio, sejam mais facilmente detectados pelos nêutrons, já que os RX têm dificuldade de interagir com sua nuvem [42].

A obtenção dos nêutrons para este método pode vir de duas fontes, fissão nuclear ou uma fonte de espalação (SNS, do inglês, *Spallation Neutron Source*). Consiste de um acelerador de partículas, neste caso prótons, linear ou em combinação com um síncrotron, incidindo um feixe de partículas sobre um alvo (um elemento pesado), geralmente de tantálio ou tungstênio. Esta interação gera nêutrons [43] que podem ou não ser polarizados, dependendo do material a se estudar [42].

Essa técnica é mais comumente aplicada na difração com pó, técnica que requer apenas um pó policristalino. Apesar deste uso, essa técnica também pode ser aplicada a um cristal, porém os cristais devem ser maiores que os usados na técnica de difração de RX, sendo comum encontrar cristais com cerca de $1mm^3$ ou mais [44].

O feixe obtido anteriormente então incide sobre uma amostra que o difrata. As partículas difratadas incidem sobre detectores, que através do padrão gerado criam uma estrutura para o material [44]. A resolução das proteínas geradas com este método depositadas no PDB até a data deste estudo está entre 1.05Å e 2.7Å.

3.2.3 Espectroscopia por ressonância magnética nuclear (RMN)

Esse método envolve as propriedades mecânicas quânticas do núcleo dos átomos, os *spins*. Cada átomo, em seu núcleo possui um ambiente eletrônico próprio, assim sendo possível através de sua leitura a sua identificação. A preparação da proteína para este método consiste em deixá-la altamente purificada em suspensão aquosa. Geralmente as amostras possuem volume entre 300 a 600 $\times 10^{-6}$L e sua concentração varia entre $0,1 \times 10^{-3}$ e $3,0 \times 10^{-3}$ moles [45].

Para se determinar a estrutura, primeiramente deve-se descobrir o seu deslocamento químico, isto é, a frequência ressonante de um núcleo relativa a um padrão em um campo magnético conhecido. Isto pode ser descoberto através de uma atribuição sequencial, com informações oriundas de vários outros experimentos de RMN. A partir dos deslocamentos químicos obtidos, gráficos bidimensionais contendo os deslocamentos químicos em ambos eixos são gerados onde neles as informações de distância, angulação e orientação podem ser captadas. A intensidade dos sinais lidos determinará a resolução final da proteína, visto que uma leitura mais intensa indica uma maior certeza da localização do átomo [46].

Com base nesses resultados é possível a construção de um modelo tridimensional através de serviços especializados como o *webserver* GeNMR [47], o *software* CYANA [48] ou o pacote escrito em C++, Xplor-NIH [49]. Esses *softwares* convertem as restrições de entrada em termos energéticos e tentam minimizá-los, a fim de se obter uma proteína o mais estável possível. Essa etapa produz vários modelos possíveis, que em uma visão mais ampla, possuem estruturas tridimensionais similares. Após a geração do modelo, deve-se então validá-lo estruturalmente para que seja atestada a sua qualidade.

Porém, em moléculas muito grandes os testes conduzidos com RMN produzem muitas sobreposições de leituras, dificultando ou até mesmo impossibilitando a sua prática. Outro problema enfrentado ao se conduzir experimentos de RMN com proteínas grandes é que o tempo disponível para a leitura dos dados durante o experimento é reduzido, reduzindo assim sua precisão [50].

3.2.4 Crio-microscopia eletrônica

Este método criado e desenvolvido nas décadas de 1980 e 1990, era capaz de produzir modelos com resolução de 10 Å, porém no fim de 2013 este método foi capaz de resolver uma estrutura de 3,4 Å e em 2015 gerou um modelo com a resolução de 2,2 Å, mostrando ser capaz de gerar modelos com resoluções ainda maiores. Para a obtenção das estruturas, o etanol líquido congela proteínas suspensas em solução para reduzir a vibração de seus átomos, então um feixe de elétrons incide sobre a amostra, que é amplificado por uma lente e incide sobre um detector de elétrons [51].

Estima-se que futuramente este método pode, e deve, substituir o método de difração de Raios-X, visto que ele dispensa a necessidade de se cristalizar proteínas, abrangendo um número maior de proteínas passíveis de serem modeladas por este método. Apesar de todo poder deste método os custos ainda são altos, o que pode postergar a disseminação do mesmo. Foi estimado que por dia gasta-se cerca de £ 3 mil para manter os equipamentos funcionando e £ 1 mil de luz, totalizando £ 4 mil por dia. Para evitar os custos altos e tornar este método mais disponível, financiadores criaram instalações coletivas onde os pesquisadores podem marcar um horário para o uso do equipamento, dividindo assim os custos [51].

3.3 Predição de estrutura de proteínas em larga escala

Desde o início do sequenciamento de genomas com o método proposto por Sanger na década de 70 [52] até os dias de hoje [53], os sequenciadores mudaram muito na sua capacidade e custo de análise, como mostrado no gráfico da Figura 3.2.

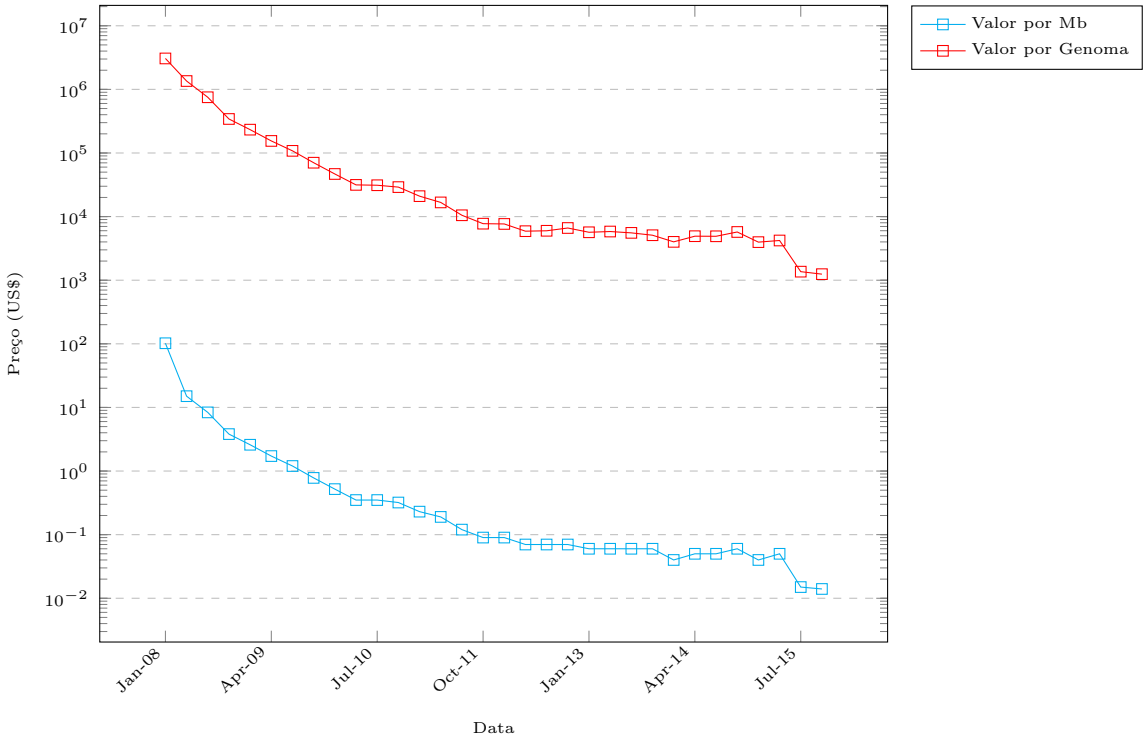


Figura 3.2: Valor (em dólar) de sequenciamento por megabase (Mb, em azul) e para o genoma humano (cerca de 3000Mb, em vermelho) de 2001 até 2015. Retirado de `https://www.genome.gov/sequencingcostsdata/` [54]. Último acesso: 02/02/2017.

Com menores preços, mais instituições puderam fazer o uso do sequenciamento, gerando assim um volume maior de dados. Esses dados precisam ser analisados para que sejam úteis. Para tal tarefa, ferramentas voltadas para a bioinformática, visualização molecular e modelagem comparativa vêm sendo desenvolvidas para auxiliar os pesquisadores a examinar, cada vez mais, grandes volumes de dados [55] [56]. Uma das dificuldades de se lidar com grandes quantidades de dados reside na confiabilidade dos mesmos, de forma que seja aproveitado o máximo de informação de maneira correta, a fim de se gerar resultados confiáveis e úteis.

As teorias utilizadas nesta categoria vêm de resultados obtidos de experimentos e observações realizados em proteínas que possuem uma estrutura tridimensional já determinada e de conhecimentos sobre as interações físico-químicas dos seus resíduos. Foi criada para ser uma alternativa aos métodos experimentais, reduzindo o tempo e custo financeiro final para se obter uma estrutura 3D de uma proteína [48]. As principais técnicas serão apresentadas a seguir.

3.4 Modelagem teórica

Enquanto que os métodos de modelagem experimentais são caros, pois demandam equipamentos e reagentes, e o acesso a esses equipamentos é limitado, os métodos teóricos necessitam apenas de computadores, tornando-se acessíveis a todos e com custo reduzido. Os modelos gerados por estes métodos se baseiam em observações de estruturas de proteínas já modeladas através dos métodos experimentais.

3.4.1 Template based

3.4.1.1 Modelagem comparativa

Também conhecida como modelagem por homologia, é uma técnica que envolve predizer a estrutura 3D de uma proteína baseada na semelhança entre as sequências de aminoácidos das proteínas molde e modelo. Sendo o molde uma proteína com estrutura tridimensional já conhecida, obtida através de métodos experimentais (*e.g.*, RMN, difração de Raios-X), tendo seus dados depositados em bancos de dados como o PDB, e o modelo é a proteína que se deseja obter sua estrutura tridimensional [55]. Essa técnica é possível pois pequenas mudanças na sequência de aminoácidos da proteína geralmente resultam em poucas mudanças na sua estrutura 3D final [57].

A modelagem comparativa é constituída por quatro etapas (Figura 3.3): (i) seleção do(s) molde(s), (ii) alinhamento entre molde e modelo, (iii) construção do modelo e (iv) avaliação do modelo. Caso o modelo não esteja satisfatório, retorna-se à etapa de identificação de molde, seleção do molde ou alinhamento da sequência com os moldes para que uma conformação aceitável, de acordo com as necessidades, seja atingida.

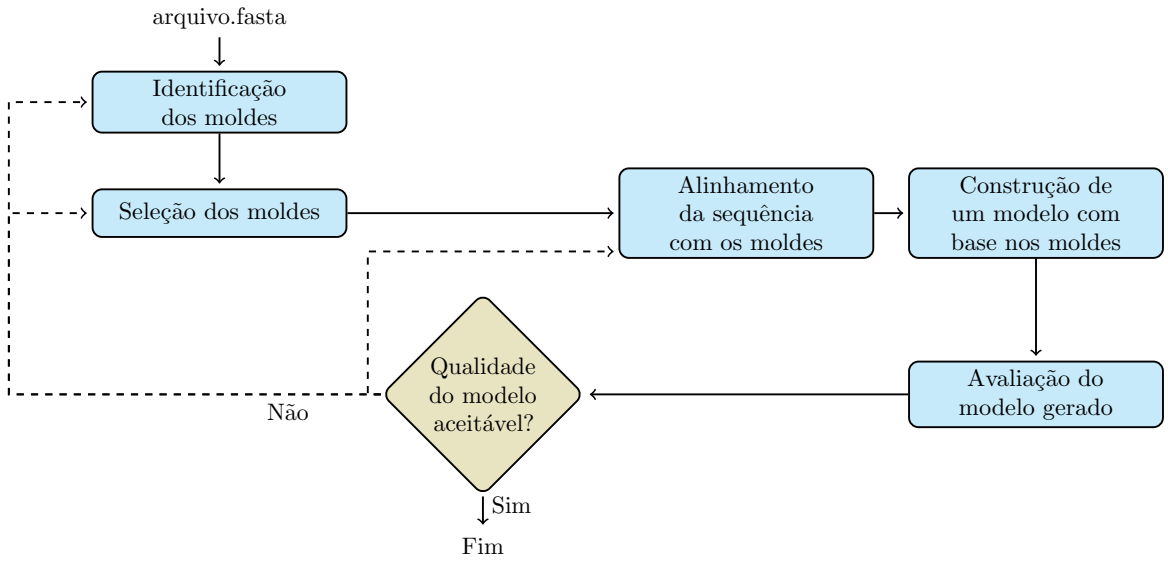


Figura 3.3: Esquema do processo de modelagem comparativa. Baseado em Martí-Renom *et al.* (2000) [58].

A primeira etapa, a identificação dos moldes, consiste em encontrar proteínas com modelos tridimensionais já determinados. Atualmente, tem-se bancos de dados como o PDB e o *Protein Data Bank in Europe* (PDBe) [59] como repositórios de dados relativos à proteínas com seus modelos tridimensionais resolvidos. Até a presente data o PDB conta com mais de 124 mil modelos depositados como mostra o gráfico na Figura 3.4.

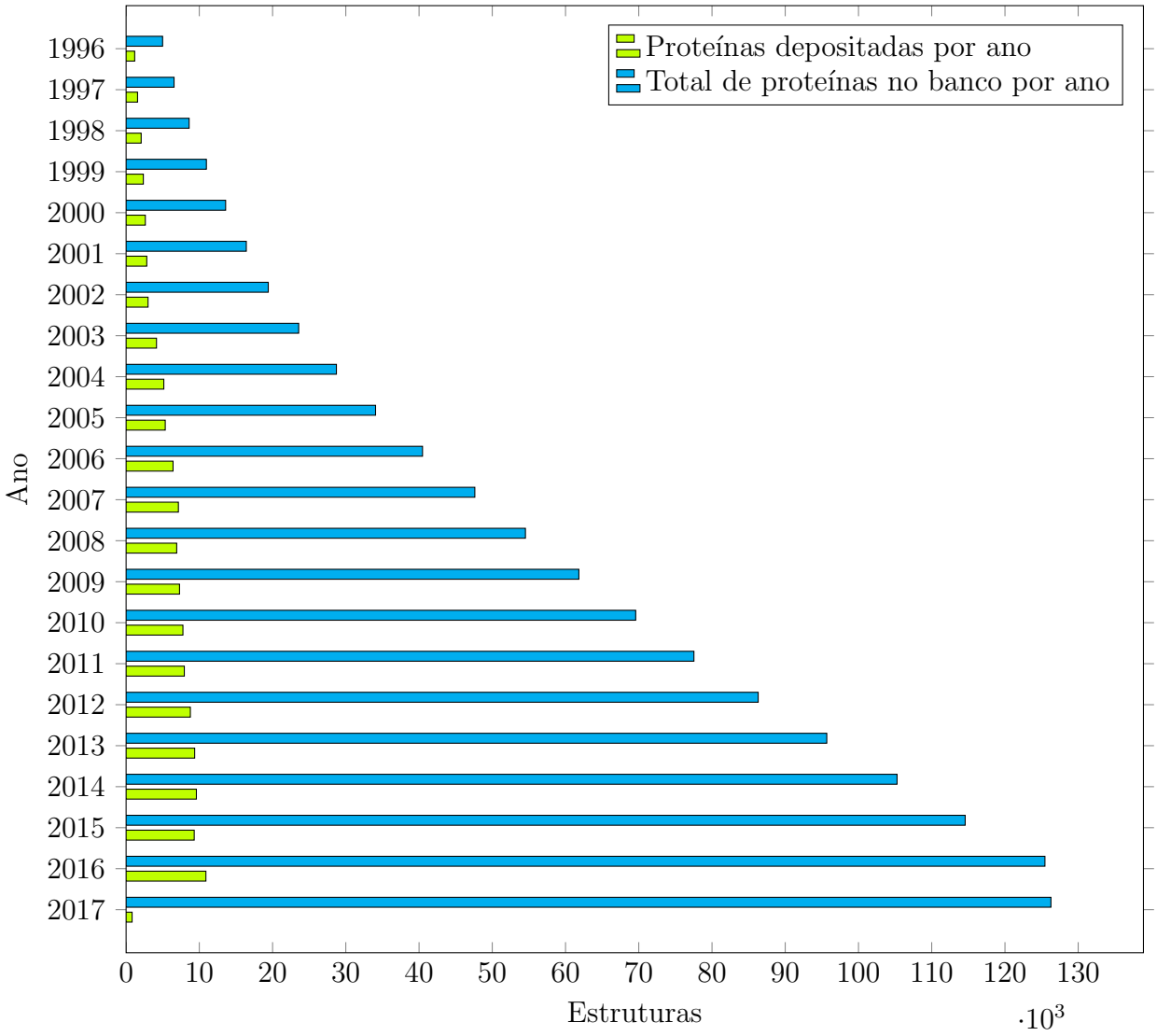


Figura 3.4: Número de proteínas depositadas no PDB nos últimos 20 anos. Dados obtidos em `http://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=total&seqid=100` [60]. Último acesso: 31/01/2017.

Existem diferentes algoritmos para a predição 3D de proteínas por modelagem comparativa. Alguns dos *softwares* de modelagem comparativa estão listados na Tabela 3.1[61].

Tabela 3.1: *Softwares* de modelagem comparativa, seu distribuidor ou desenvolvedor e suas características

Programa	Desenvolvido/ Distribuído	Metodologia
Prime [62]	Schrödinger, LLC	Constrói um modelo usando as posições dos átomos dos moldes. Faz uso do campo de força OPLS.
MOE [63]	Chemical Computing Group,Inc.	Uma combinação dos métodos de *segment-matching* e uma abordagem de adição/ deleção de regiões. Utiliza método de busca de *loop* baseado em conhecimentos prévios e seleção de rotâmeros de cadeias laterais.
SWISS-MODEL [64]	ExPASy	Servidor automatizado para modelar as proteínas com o uso do *software* ProModeII. O SWISS-MODEL utiliza o campo de força Gromos96 para realizar a minimização da energia do modelo gerado.
MODELLER [65]	Sali Lab	Criação de um modelo baseado na satisfação de restrições espaciais, baseado e uma função densidade de probabilidade. Faz uso de bibliotecas internas que podem ser importadas em qualquer *script* e Python, dando ao usuário a possibilidade de integrar a execução do Modeller em seus *scripts*.

3.4.1.2 Método por conhecimento de enovelamento ou *Threading*

O método de *Threading* possui como base a suposição da existência de um número finitos de enovelamentos possíveis e na preservação da estrutura 3D como mais importante que a preservação de sua sequência na evolução. O objetivo desta técnica, é estabelecer uma relação entre a sequência de aminoácidos e um modelo estrutural. Durante este processo os aminoácidos são alocados seguindo sua ordem sequencial em uma posição estrutural de um modelo tridimensional proteico de uma maneira ótima [66]. A modelagem por *Threading* é constituída de três etapas:

- Construção de uma biblioteca de possíveis dobramentos de modelos estruturais: Esta biblioteca é construída a partir de estruturas de proteínas conhecidas, geralmente derivadas de uma base de dados como o PDB[36]. Geralmente as coordenadas tridimensionais de uma estrutura de proteína são reduzidas a representações mais abstratas. Os elementos estruturais principais são definidos pelos elementos de estrutura secundárias, tais quais, fitas, folhas, hélices e alças. Com frequência, as informações das cadeias laterais são removidas restando apenas um molde da cadeia principal [67].

- Criação de uma tabela de pontuação para avaliar toda inserção de uma sequência em um dobramento em particular: as funções de pontuação são geralmente uma lista de referências estatísticas de cada resíduo de aminoácido para cada estrutura ou ambiente de enovelamento [68] descrevendo o quão favorável a substituição de uma sequência alvo por um molde de estrutura são [69]. Grande parte das técnicas de *Threading* não fazem uso das funções de energia livre dos átomos, como é utilizado nos métodos *Template Free*. A maior parte das funções de energia são determinadas pela análise estatística das estruturas tridimensionais de proteínas do banco de dados. Estas funções são referenciadas geralmente como funções com base em conhecimentos prévios.

- Então, um método para varrer a biblioteca em busca de substituições válidas, através de um algoritmo que identifica a relação sequência-estrutura ótima. A tarefa principal é a identificação da melhor pontuação e substituição sequência-estrutura.

3.4.2 Template Free

3.4.2.1 *Ab initio*

Baseia-se nos princípios da termodinâmica e de que a estrutura nativa de uma proteína corresponde ao valor mínimo global de energia livre [70] [71]. Em métodos puramente *ab initio* não se usa nenhum tipo de informação provida por estruturas já conhecidas, como se faz nos métodos baseados em *template*. As informações estruturais de proteínas previamente determinadas são utilizadas somente para a parametrização dos potenciais empíricos dos átomos utilizados em campos de força (funções de potenciais energéticos) como AMBER [72], GROMOS[73], OPLS [74] e outros mais.

O enovelamento proteico no método *ab initio* é considerado um problema de otimização global, sendo o objetivo a identificação dos valores de um conjunto de variáveis como ângulo de torsão e posição de alguns ou de todos os átomos [66]. O método *ab initio* requer três elementos [75][76]:

- A representação geométrica da cadeia proteica: Uma representação que corresponde ao modo como a proteína será representada computacionalmente;
- Uma função de potencial: São funções utilizadas em simulações de mecânica molecular [77], *design* de proteína [78] e predição de estrutura proteica [79];
- Uma técnica de busca de superfície de energia: Estes métodos incluem abordagens como simulações de Monte Carlo que não utilizam forças porém compara energias através da distribuição de Boltzmann [80], simulações de dinâmica molecular de proteínas [81], entre outros.

3.4.2.2 *De novo*

Diferentemente do método *ab initio*, o método *de novo* faz uso de informações obtidas através de preditores de estruturas secundárias, fragmentos de proteínas e modificação da função de energia. A predição de estruturas secundárias engloba as técnicas que visam sugerir quais estruturas secundárias estão presentes na sequência e onde elas se localizam. Através de abordagens como levar em consideração a propensão de cada resíduo ser uma determinada estrutura secundária e os efeitos trazidos pelos resíduos adjacentes, foi possível alcançar uma acurácia superior à 60% nas predições, atualmente os preditores de estrutura

secundária possuem desempenho na precisão em torno de 80% [35]. Em 2017, Trevizani *et al.* [82] conduziu um estudo que corroborou que o uso de preditores de estruturas secundárias é capaz de melhorar a qualidade dos modelos gerados.

Com o aumento dos bancos de dados de proteínas, mais informação sobre estruturas conhecidas ficaram disponíveis, então, através de ferramentas computacionais, as informações de enovelamento provenientes destas estruturas puderam ser inclusas nas predições. Esta abordagem consiste em, a partir de uma biblioteca de fragmentos, buscar a semelhança entre as sequências dos fragmentos e a sequência da proteína a qual deseja-se modelar. Então, um conjunto de fragmentos que possivelmente gerarão a estrutura são selecionados, pois uma mesma sequência pode gerar estruturas diferentes do mesmo modo que sequências diferentes podem gerar uma mesma estrutura. Esta seleção dos fragmentos se dá com base em uma pontuação, proveniente do grau da similaridade entre a sequência e cada fragmento, bem como o resultado da predição do *software* preditor utilizado. Ao se classificar por essa pontuação, os melhores resultados serão os fragmentos escolhidos para a montagem da proteína [35].

As principais limitações deste método são a aplicação restrita à sequências curtas (<150 resíduos) e a dificuldade em se parametrizar as funções estatísticas utilizadas, visto que uma má configuração destas funções acarretará em modelos com qualidade reduzida [35].

3.5 Validação dos resultados

Todas as estruturas 3D geradas, tanto por métodos experimentais durante a fase computacional quanto os métodos *in silico* devem passar pelo processo de validação, para se assegurar a qualidade dos modelos. Alguns *softwares* comumente utilizados para validação de proteínas (principalmente de modelos 3D) são o Molprobity [83], Procheck [84] e Procheck-NMR [85].

3.6 Auxílio de *workflows* na predição de estrutura tridimensional de proteína em larga escala

Em um cenário onde se possui um volume de proteínas modeladas cerca de 3 vezes menor que o número de sequências curadas manualmente (3.5), algo precisa ser feito para

reduzir esta diferença. Uma das possíveis abordagens é a automatização do processo de obtenção das estruturas, que pode ser feita através de um *workflow*.

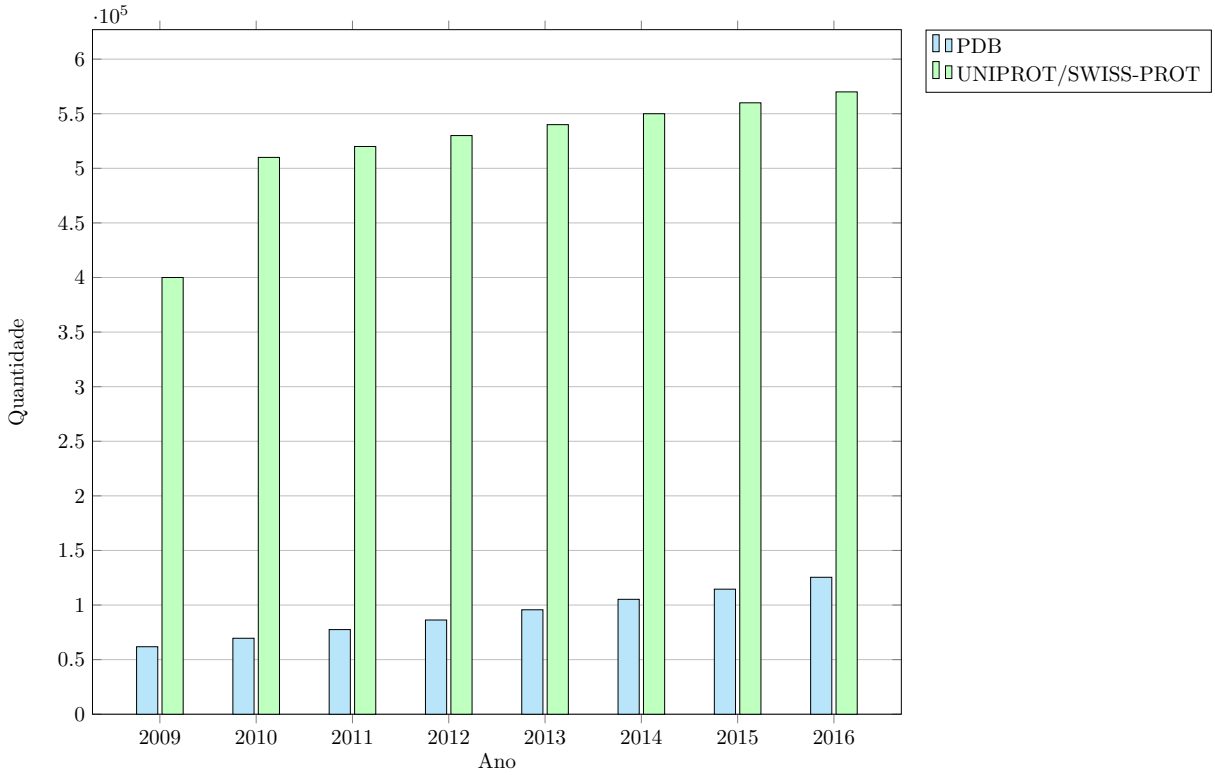


Figura 3.5: Comparação dos tamanhos dos bancos de dados de estruturas e sequências de proteínas. Quantidade de sequências de proteínas obtida de: `http://web.expasy.org/docs/relnotes/relstat.html`, último acesso em 20-04-2017. Quantidade de estrutura de proteínas obtida de: `http://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=total&seqid=100`, último acesso em 20-04-2017

Por suportar a reprodutibilidade de tarefas de maneira organizada, através de entradas e saídas de arquivos e dados [21], os *workflows* podem ser utilizados para modelar e analisar estruturas tridimensionais de proteínas, reduzindo a necessidade de intervenção humana em algumas etapas. Um *workflow* pode auxiliar na organização e definição de ordem de execução de *softwares*, bem como manter a clareza dos processos que já foram e que serão cumpridos. Estas medidas permitem aos usuários gerar uma quantidade maior de dados em um tempo reduzido, em comparação com o tempo necessário para se modelar manualmente um conjunto de proteínas.

4 MHOLline

O MHOLline é um *workflow* científico desenvolvido para automatizar a modelagem tridimensional e análise de proteínas. Sua origem data do ano de 2004, sendo apresentado por Shaila C. S. Rössle em sua tese de Doutorado na Universiadade Federal do Rio de Janeiro (UFRJ) [86].

4.1 MHOLline 1.0

Em 2007, uma parceria da UFRJ com o Laboratório Nacional de Computação Científica (LNCC) deu início ao desenvolvimento da primeira versão *web* do MHOLline [87, 88], sendo lançada em 2010 no endereço eletrônico `www.mholline.lncc.br`. A partir de um arquivo texto de entrada no formato FASTA 4.1, os dados podem percorrer até sete módulos do *workflow*, como apresentado na Figura 4.2.

```
>1ME4:A|PDBID|CHAIN|SEQUENCE
APAAVDWRARGAVTAVKDQGQCGSCWAFSAIGNVECQWFLAGHPLTNLSEQMLVSCDKTDSGCSGGLMNNAFEWIVQENN
GAVYTEDSYPYASGEGISPPCTTSGHTVGATITGHVELPQDEAQIAAWLAVNGPVAVAVDASSWMTYTGGVMTSCVSEQL
DHGVLLVGYNDSAAVPYWIIKNSWTTQWGEEGYIRIAKGSNQCLVKEEASSAVVG
```

Figura 4.1: Exemplificação de uma sequência no formato .fasta.

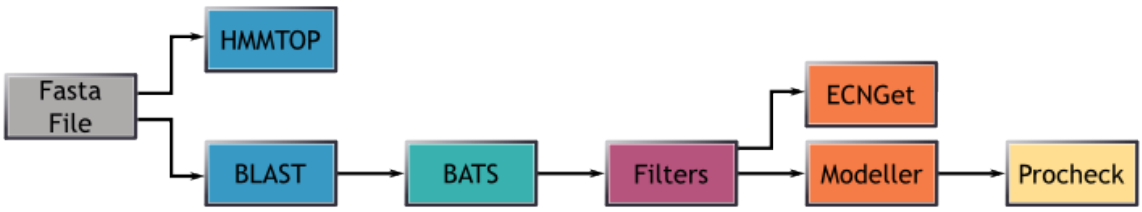


Figura 4.2: Representação esquemática do fluxo de processos do MHOLline 1.0 [87, 88]. Obtido em `http://www.mholline.lncc.br/images/workflow_scheme.png`. Último acesso: 01/02/2017.

O MHOLline 1.0 é composto por três partições como pode ser visto na Figura 4.4:

- MHOL*core* - Núcleo do MHOLline. Nesta partição encontram-se todos os *scripts* de processamento de dados do MHOLline. As linguagens utilizadas nesta partição são *PERL*, *C* e *ShellScript*;

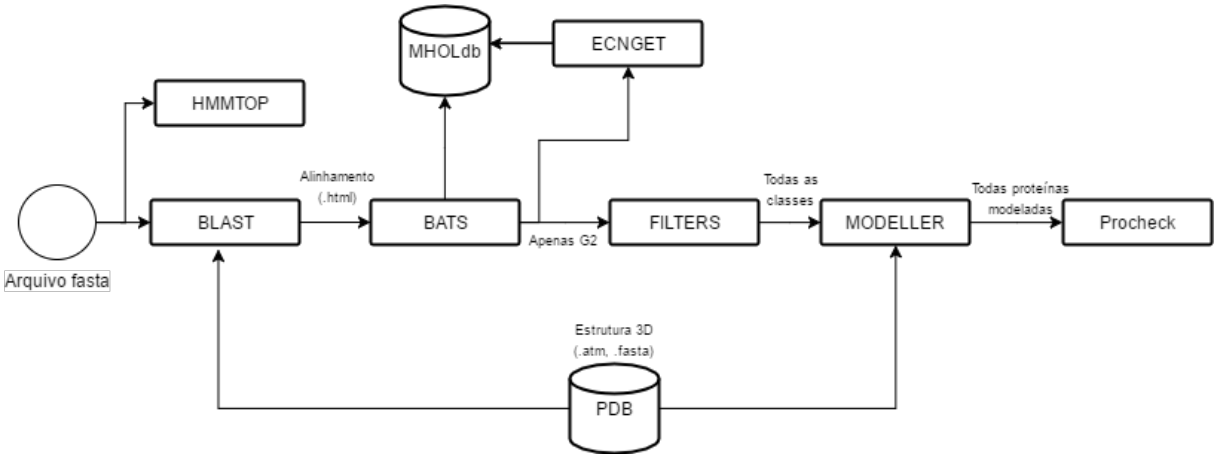


Figura 4.3: *Workflow* completo do MHOLline 1.0. *textWorkflow* criado através da ferramenta draw.io [89].

- MHOL*web* - Interface existente entre os *jobs* submetidos e os usuários. Através desta é possível fazer submissão, análise do progresso dos dados no *workflow* do MHOLline e *download* dos resultados. Esta partição do MHOlline utiliza linguagens como *PHP*, *JavaScript* e *CSS* além dos protocolos *HTTPS* e *FTP*;

- MHOL*db* - Base de dados. Utilizada para armazenamento de informações do MHOLline, como usuários, *jobs* e dados relativos ao sistema. O sistema de gerenciamento de banco de dados utilizado nesta partição é o MySQL, com a interface PHPMyAdmin.

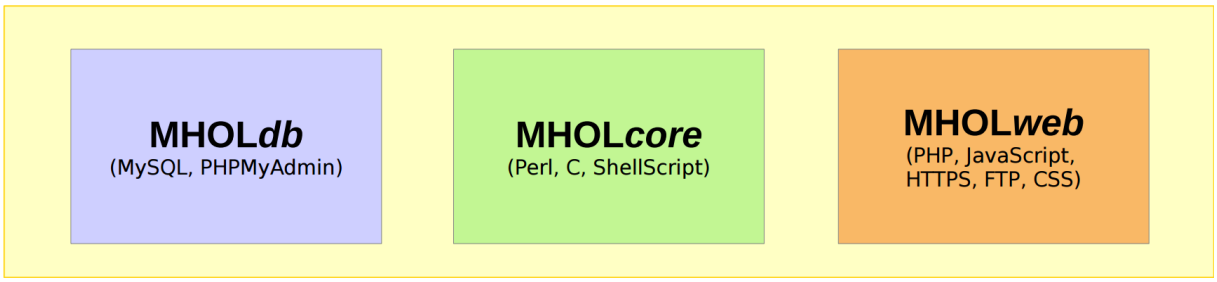


Figura 4.4: Estrutura do MHOLline 1.0. O MHOLline é dividido em MHOL*db* (azul), MHOL*core* (verde) e MHOL*web* (laranja). Obtido de Reis *et al.* (2014) [90].

No acesso ao MHOLline existem três tipos de permissão de usuário:

- Usuário Não Logado: Qualquer usuário que se conecte ao *site* do MHOLline [91] e submeta um *job* sem logar em uma conta. Usuários deste grupo estão restritos à

submissão de *jobs* com até no máximo 50 sequências e seus resultados serão recebidos por *e-mail* (caso informado) e através de um *link* com uma chave SHA-512;

- Usuário Logado: Para se tornar um usuário logado, deve-se submeter um pedido de cadastro, onde o solicitante deve informar dados pessoais como nome, *e-mail*, instituição e dados sobre o projeto no qual o MHOLline será utilizado. O usuário logado pode submeter arquivos sem limite de número de sequências, os términos de seus *jobs* são notificados via *e-mail* e o usuário pode acessar os resultados através de uma interface própria de gerenciamento de *jobs*, sem a necessidade de se guardar um *link*.;

- Administrador: O administrador é uma conta de usuário especial criada para gerenciamento: (i) dos dados de outros usuários, dentre suas funções destacam-se o aceite ou rejeição de novas aplicações de usuários no MHOLline, (ii) de atualização dos repositórios e *backup* e restauração do sistema, e (iii) dos relatórios de estatísticas de uso do sistema. Apesar de ser uma conta administrativa, o administrador pode submeter *jobs* e analisá-los como qualquer outro usuário, seu modo de recebimento de notificações de término de *jobs* é idêntica a de um usuário logado.

Atualmente o MHOLline 1.0 conta com 73 usuários e já processou 656.315 sequências, totalizando 13.396.745 segundos (mais de 37.213 horas) de processamento [91]. Estas estatísticas englobam todas as atividades realizadas no MHOLline 1.0 de 2009 até março de 2017. A função de cada um dos sete módulos do MHOLline será descrita a seguir.

4.1.1 HMMTOP

O HMMTOP é um preditor de hélices transmembranares, o qual utiliza o método de cadeia oculta de Markov para identificá-las através da divergência máxima da composição de aminoácidos de segmentos de sequências [92, 93]. A saída resultante é dada por um arquivo no formato texto.

4.1.2 BLAST

O BLAST (do inglês, *Basic Local Alignment Search Tool*) é um *software* de busca de similaridade entre sequências contra bancos de dados. O BLAST é uma das ferramentas

mais usadas em bioinformática. O processo de busca no BLAST pode ser subdividido em três módulos: *Setup*, *Scanning* e *Traceback* [94].

O módulo de *Setup* consiste na leitura da sequência, aplicação de filtros e construção de uma tabela de pesquisa (*e.g.*, tabela *hash*) com as sequências. O segundo módulo, de *scanning*, engloba a busca na base de dados, escaneando cada sequência dada como entrada em busca de encontros, chamados *hits*, correspondentes às palavras contidas na tabela de pesquisa. A partir dos *hits* são gerados alinhamentos sem *gaps*[2], que se obtiverem uma pontuação superior a um determinado limite serão reprocessados com a adição de *gaps*, e se novamente excederem outro limite serão guardados para processamento futuro sendo classificados como alinhamentos preliminares. Esta etapa também compreende otimizações como retornar apenas resultados significativos como a pontuação e o tamanho do alinhamento, reduzindo o uso de memória e CPU. O terceiro módulo, de *traceback*, compreende a localização das inserções e deleções nos alinhamentos que obtiveram *hits* [94].

Após o processamento, uma lista é retornada com todos os *hits* ordenados em ordem decrescente das pontuações dos resultados. Para o MHOLline o BLAST é utilizado para buscas no banco de dados PDB, sendo toda consulta executada localmente no próprio servidor do MHOLline (em cópia local do repositório do PDB).

4.1.3 BATS

O BATS (do inglês, *BLAST Automatic Targeting for Structures*) é capaz de identificar as sequências que as técnicas de modelagem são capazes de atuar, separando o resultado do BLAST em 4 grupos de acordo com os critérios descritos na Tabela 4.1. O BATS usa um índice para denotar a cobertura, chamado LVI (*Length Variation Index*), que pode ser calculado de acordo com a equação 4.1.

$$LVI = \frac{Q_L - N_{Ali}}{Q_L} \tag{4.1}$$

Onde Q_L é o tamanho da sequência de entrada e N_{Ali} é o número de aminoácidos alinhados entre a sequência de entrada e o *hit* em questão.

[2]Espaçamentos inseridos em uma sequência de um alinhamento para que os caracteres sucessivos se alinhem segundo algum critério definido pelo algoritmo de alinhamento. No ponto de vista biológico, um gap representa uma deleção na sequência na qual ele foi inserido ou uma inserção na(s) sequência(s) na(s) qual(is) ele não foi inserido.

Tabela 4.1: Classificação dos grupos do MHOLline para os resultados do BLAST

Classe	**Critérios**
G0	Sequência não alinhada
G1	E-value $>$ 10e-5 ou identidade $<$ 15%
G2	E-Value $\leqslant$ 10e-5 e Identidade $\geqslant$ 25% e LVI $\leqslant$ 0,7
G3	E-Value $\leqslant$ 10e-5 e (Identidade entre 15% e 25% ou LVI $>$ 0,7)

Para os grupos G0, G1 e G3, apesar de na literatura não terem sido descritas formas eficientes de se modelar as proteínas destes grupos, estima-se que as técnicas eficazes sejam descritas na Tabela 4.1 ou uma combinação entre elas.

As proteínas classificadas como G2, passíveis de serem utilizadas na modelagem por homologia, o único método utilizado pelo MHOLline atualmente, são ordenadas por uma pontuação, dada pela equação 4.2, seguidamente como segundo critério de ordenação a resolução do molde. A proteína que ficar na primeira posição desta lista será utilizada como molde para a modelagem [88]. O MHOLline se restringiu a selecionar apenas um molde visto que as melhorias causadas por múltiplos moldes não são estatisticamente relevantes [95].

$$S_f = W_i P_i + W_s P_s + W_n P_n + W_l Pl + W_g P_g \tag{4.2}$$

Sendo W_i o peso e P_i o valor da identidade resultante do BLAST, W_s o peso e P_s a pontuação da similaridade resultante do BLAST, W_n o peso e P_n o número de aminoácidos alinhados, W_l é o peso e Pl o valor do índice LVI (descrito na Equação 4.1) e W_g é o peso e P_g o valor do índice GRSI (*Gap Relative Strength Index*) denotado por

$$GRSI = \frac{wG_a + G_g}{Q_L} \tag{4.3}$$

onde w é a penalidade para a abertura de um *gap* novo, G_a é o número de *gaps* novos abertos, G_g e p número de *gaps* adicionados em um *gap* já existente e Q_l é o tamanho da sequência de *input*.

4.1.4 Filters

O Filters é uma ferramenta desenvolvida pela equipe do MHOLline para classificar as sequências selecionadas como G2 pelo BATS, denotando a qualidade final do modelo

gerado com base nos valores de identidade e LVI [88]. A classificação pode ser conferida na Tabela 4.2.

Tabela 4.2: Classificação do FILTERS para os resultados do BATS para o grupo G2

Qualidade	**Identidade**	**Cobertura**
Very High	$\geqslant$ 75%	$\geqslant$ 90%
High	$\geqslant$ 50% e <75%	$\geqslant$ 90%
Good	$\geqslant$ 50%	$\geqslant$ 70% e <90%
Medium to Good	$\geqslant$ 35% e <50 %	$\geqslant$ 70%
Medium to Low	$\geqslant$ 25% e <35%	$\geqslant$ 70%
Low	$\geqslant$ 25%	$\geqslant$ 50% e <70%
Very Low	$\geqslant$ 25%	$\geqslant$ 30% e <50%

4.1.5 ECNGet

Ferramenta criada pela equipe do MHOLline para capturar o número EC (do inglês, *Enzyme Comission*) para cada sequência no grupo G2. Através de base de dados obtidas no Uniprot (*Universal Protein Resource*) e no ExPASy (*Expert Protein Analysis System*) é possível estabelecer uma relação entre o código de identificação da proteína (textitPDB ID) e seu número EC, sendo capaz de identificar enzimas com mais de um número EC [88].

4.1.6 Modeller

Software capaz de modelar proteínas com base em moldes já existentes, partindo de um molde com estrutura tridimensional já determinada, valendo-se da similaridade estrutural e funcional do modelo e molde para sugerir a possível estrutura tridimensional do modelo desejado [58]. As restrições espaciais para se determinar a conformação do modelo se baseiam em análises estatísticas envolvendo algumas características do modelo e seus moldes, como por exemplo as distâncias entre os C_α da proteína e valores de ângulos diedrais de cadeias principais de proteínas equivalentes. Essas características são expressas por uma função densidade de probabilidade e podem ser utilizadas diretamente como restrições espaciais. As restrições espaciais e os termos de energia CHARMM [96] são combinados em uma função objetivo. Finalmente, o modelo é obtido pela otimização da função objetivo em um espaço cartesiano. Esta otimização é feita através do método

de função alvo variável [97] empregando métodos de gradientes conjugados e dinâmica molecular com recozimento simulado. [98] A conformação 3D final é dada como um arquivo texto no formato PDB.

4.1.7 Procheck

O Procheck é um avaliador de estruturas tridimensionais proteicas composto de cinco ferramentas chamadas de *clean.f*, *secstr.f*, *nb.c*, *anglen.f* e *pplot.f*, sendo executadas na ordem citada. A entrada do Procheck é um arquivo de estrutura tridimensional de proteína (formato PDB) e opcionalmente um arquivo de texto contendo informações para plotagem e resultados de saída.

A primeira ferramenta, o *clean.f*, executa a "limpeza" do arquivo de entrada, assegurando-se que os átomos estejam devidamente classificados de acordo com as regras definidas pela IUPAC (do inglês International Union of Pure and Applied Chemistry) [99]. A segunda ferramenta, o *secstr.f*, é responsável por criar a montagem da estrutura secundária, resíduo a resíduo de acordo com o método modificado de Kabsch e Sander [100]. A terceira, o *nb.c*, identifica todas interações não ligadas entre pares de resíduos. Estas interações são definidas como o contato entre átomos mais próximo menor que 4 Å e os átomos se encontram separados por quatro ou mais ligações. A quarta ferramenta, o *anglen.f*, calcula o tamanho e os ângulos formados pelas ligações da cadeia principal. A quinta e última, o *pplot.f*, gera os gráficos de saída com os dados obtidos nos outros *softwares* do Procheck. A saída do Procheck consiste em vários gráficos no formato *postscript* em conjunto com uma avaliação, em arquivo texto, do gráfico de Ramachandran 4.5 dos resíduos da proteína fornecida [84].

Nessa dissertação, o desenvolvimento da segunda versão do MHOLline será o alvo principal do trabalho considerando-se as atuais necessidades dos usuários. O MHOLline 2.0 será apresentado no Capítulo 6.

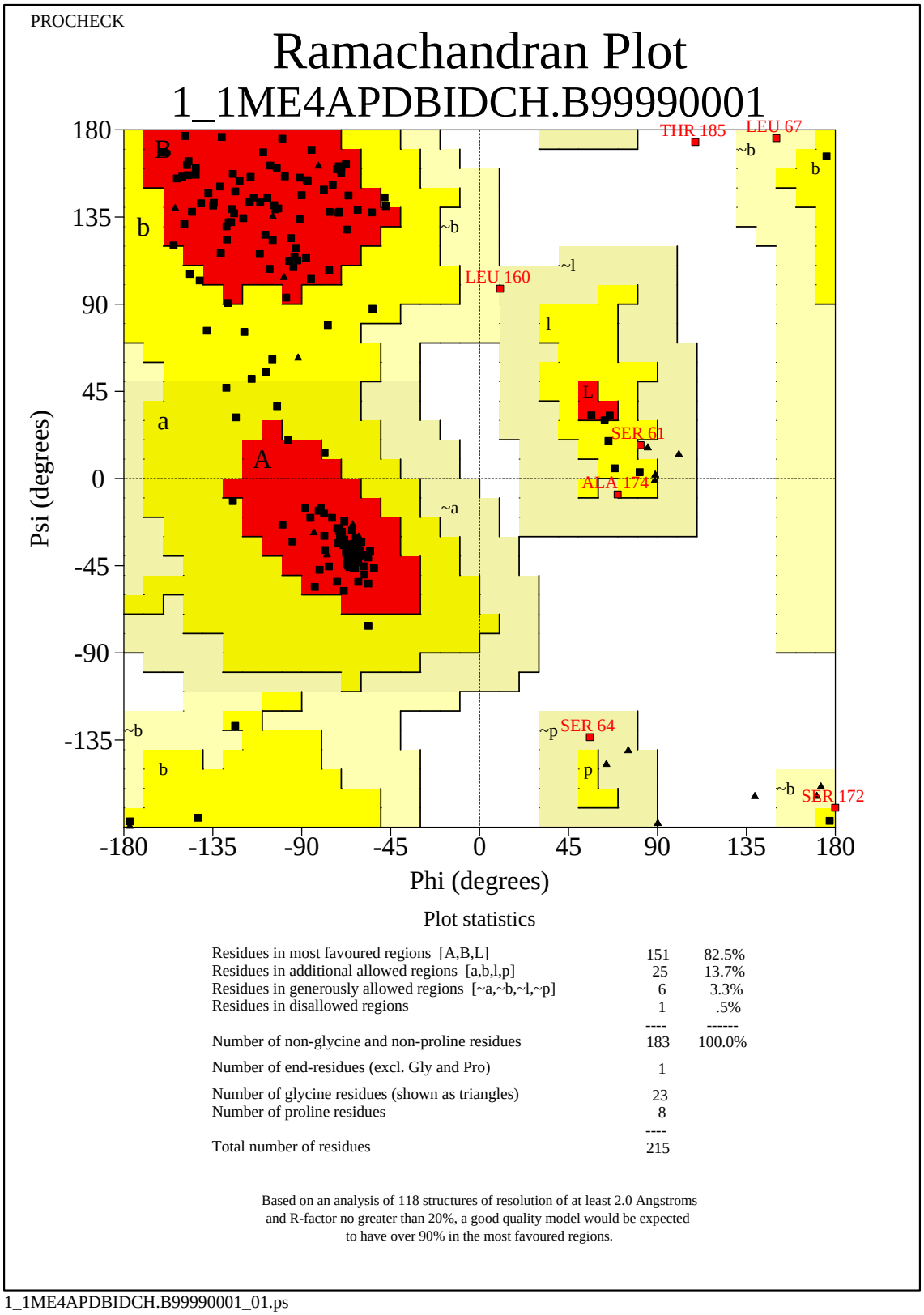


Figura 4.5: Gráfico de Ramachandran da proteína 1ME4[101] gerado pelo *software* Procheck

5 OBJETIVOS

5.1 Objetivo Geral

Desenvolver a ferramenta MHOLline 2.0 com a adição de novas ferramentas e com foco na área de resubmissão de proteínas para remodelagem e refinamento de modelos.

5.2 Objetivos Específicos

- Adição de novos softwares no *workflow* do MHOLline.
- Otimizar a execução do *software* Modeller visando economizar tempo no processamento do *workflow*.
- Criar uma ferramenta para refinamento automático e manual de modelos 3D de proteínas gerados via MHOLline.
- Criar um método para agrupamento de modelos 3D baseado em qualidade estrutural.

6 MHOLline 2.0

A partir de 2013, foi estabelecida uma nova parceria agora com a Universidade Federal de Juiz de Fora (UFJF) para o desenvolvimento da versão 2.0 do MHOLline. Essa versão conta com a adição de novos módulos, bem como uma área de refinamento de proteína para que os usuários sejam capazes de analisar e otimizar os modelos já gerados no *workflow* principal do MHOLline. O MHOLline 2.0 conta com doze *softwares* em seu *workflow* principal e mais três na área de ressubmissão, como apresentado no esquema de processos mostrado na Figura 6.1.

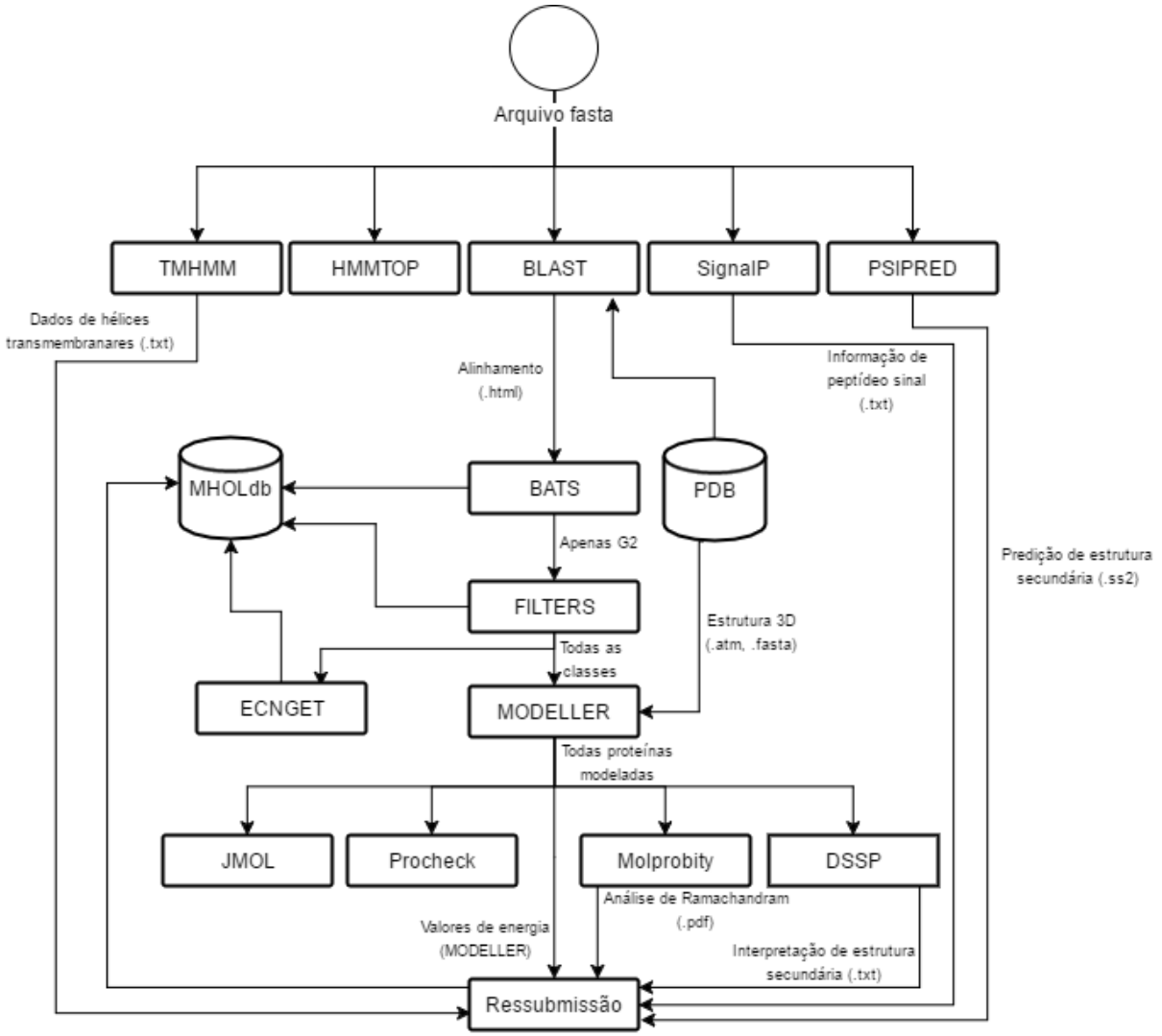


Figura 6.1: Workflow completo do MHOLline 2.0. *Workflow criado com a ferramenta [?].*

Apesar das adições feitas no MHOLline 2.0, a estrutura das partições não necessitou ser

alterada, mas houveram alterações dentro de cada partição. No MHOL*core* houve a adição da linguagem Python em sua lista de linguagens utilizadas. No MHOL*db* houve otimização dos tipos de dados e nos formatos das tabelas, visando reduzir o tempo de consulta sem aumentar o tamanho da base de dados, além de evitar *overload*. No MHOL*web* houve a mudança de toda a interface, para um modelo mais moderno e intuitivo. A seguir, será apresentada uma breve descrição dos *softwares* acrescentados no MHOLline 2.0 em comparação com sua versão 1.0.

6.1 SignalP

O *SignalP* é um identificador de regiões de peptídeo sinal [3] e de sítios de clivagem, e foi incorporado ao MHOLline 2.0 com o objetivo de detectar esses tipos de segmentos em regiões não cobertas por moldes. Atua com base em redes neuronais [103], treinadas para conjunto de dados de procariotos e eucariotos, sendo que procariotos foram subdivididos em dois grupos, gram-positivos e gram-negativos (sendo que vírus, fagos, proteínas organelares, *Archaebacteria* e *Mycoplasma* não estão inclusos nos *datasets*).

Posteriormente, na versão 2.0 do SignalP, o método de modelo oculto de Markov foi adicionado visando aumentar a precisão das predições [104]. A versão 3.0 do programa não apresentou grandes modificações. A partir da versão 4.0, o SignalP se tornou capaz de distinguir entre regiões de peptídeo sinal e regiões transmembranares, e também ocorreu a substituição do método de modelo oculto de Markov por apenas redes neuronais novamente [105]. A versão do SignalP utilizada no MHOLline2.0 é a versão 4.1

Como arquivo de entrada, o *SignalP* recebe um arquivo de sequência de aminoácidos no formato `.fasta`. Como resultado, o programa gera um arquivo de texto (Figura 6.2) mostrando se existe ou não uma região de peptídeo sinal, caso positivo, apresenta também a informação de qual é o ponto de clivagem do peptídeo.

[3] Sequência entre 15 e 30 resíduos de aminoácidos presente na região N-terminal de proteínas com o papel de sinalizar para a proteína o local no qual ela deve ser ativada. [102]

```
# Measure   Position   Value     Cutoff    signal peptide?
max. C      37         0.538
max. Y      37         0.272
max. S      41         0.296
mean S      1-36       0.133
D           1-36       0.197     0.450     NO
Name=1ME4:A|PDBID|CHAIN|SEQUENCE
SP='NO' D=0.197 D-cutoff=0.450 Networks=SignalP-noT
```

Figura 6.2: Arquivo de saída do SignalP para a proteína 1ME4 [101].

6.2 PSIPRED

O *PSIPRED* é um preditor de estrutura secundária em proteínas que tem como base uma rede neuronal de duas camadas baseada nos resultados das matrizes geradas pelo *PSI-BLAST*. Inicialmente, um perfil é obtido através de uma consulta com o *PSI-BLAST* [106], visto que ele gera estes arquivos durante a sua execução em busca de homólogos distantes [107].

Os perfis são utilizados como arquivos de entrada para a predição de estrutura secundária, eliminando-se assim a etapa computacionalmente mais custosa, a de alinhamento múltiplo, reduzindo significativamente o tempo necessário para se obter a predição da estrutura secundária. A fim de se produzir perfis altamente sensíveis, um banco de dados de proteínas não redundantes é compilado extraindo-se sequências não idênticas de alguns bancos de dados públicos, então o *software SEG* [108] é utilizado para se remover regiões com pouca informação e outro *software* é utilizado para se remover as regiões de transmembranas.

O PSI-BLAST produz uma matriz de pontuação específica de dimensão $20 \times M$ elementos, onde M é o tamanho da sequência de entrada. Esta matriz é utilizada como dado de entrada da rede neuronal, que é treinada através do método de *backpropagation* com *cross-validation*. Os valores utilizados no treinamento foram: *Momentum*: 0.9, *Learning rate*: 0.005 e para evitar o excesso de treinamento, 10% da amostra de treinamento não foi utilizada para treinar, mas sim, utilizada para avaliar a performance durante o treinamento. Esta parcela de 10% não foi utilizada para calcular os ajustes dos pesos da rede. O critério de parada foi quando houve uma piora na performance avaliada.

A partir de testes realizados com o *PSIPRED*, de um conjunto de 23 predições, pelo menos 17 estavam corretas, resultando em pelo menos 70% de acurácia [109], o que faz com que, ainda hoje, seja um dos *softwares* mais utilizados para esta finalidade.

Como arquivo de entrada o *PSIPRED* recebe um arquivo de sequência no formato `.fasta`. Como arquivo de saída, gera um arquivo de texto (Figura 6.2) contendo qual estrutura secundária foi prevista para cada resíduo, assim como um valor de confiança para essa predição.

```
# PSIPRED HFORMAT (PSIPRED V3.5)

Conf: 9964344547862001346876533566666666444565359963353221122456 89
Pred: CCCCCCCCCCCCCCCCCCCCCCCCCCHHHHHHHHHHHHHHHHHHCCCCCCCCCCCCEEECCCCC
AA: APAAVDWRARGAVTAVKDQGQCGSCWAFSAIGNVECQWFLAGHPLTNLSEQMLVSCDKTD
10        20        30        40        50        60

Conf: 9999988316899999717886444588886678998999999998630599412328 94
Pred: CCCCCCCCCHHHHHHHHHHCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCEEEEEEEECCC
AA: SGCSGGLMNNAFEWIVQENNGAVYTEDSYPYASGEGISPPCTTSGHTVGATITGHVELPQ
70        80        90        100       110       120

Conf: 1999999998519979999702474104742644899757817999982279995249 99
Pred: HHHHHHHHHHHHCCCEEEEEEECCCCCCCCCCEEECCCCCCCCCCEEEEEEECCCCCCCEEEE
AA: DEAQIAAWLAVNGPVAVAVDASSWMTYTGGVMTSCVSEQLDHGVLLVGYNDSAAVPYWII
130       140       150       160       170       180

Conf: 5357899765615999707995320023322346 9
Pred: ECCCCCCCCCCCCEEEEEECCCCCCCCCCCCCCCCC
AA: KNSWTTQWGEEGYIRIAKGSNQCLVKEEASSAVVG
190       200       210
```

Figura 6.3: Arquivo de saída do PSIPRED para a proteína 1ME4 [101].

6.3 TMHMM

O *TMHMM* é um *software* para predição de hélices transmembranares através do método de cadeia oculta de Markov. Apesar de métodos experimentais serem capazes de mostrar a existência de uma estrutura de transmembrana [110], não é possível de se determinar a posição com total exatidão de seus componentes [111]. Trabalhando com estes preceitos, a fim de se minimizar os erros, no conjunto de dados de teste não foram incluídas proteínas que obtiveram topologias divergentes em diversos experimentos e não estava claro qual a topologia seria a correta.

Com um conjunto de dados de 645 proteínas com estruturas secundárias já conhecidas, foi mostrado em estudo que o *TMHMM* foi capaz de separar com sucesso as proteínas integralmente membranares e solúveis em água. Menos de 1% das proteínas não membranares analisadas obtiveram uma classificação errada e apenas uma, de 160 proteínas de membrana foi classificada erroneamente. Porém, o conjunto negativo utilizado não continha proteínas com peptídeo sinal.

O conjunto de peptídeo sinal utilizado para o treino do TMHMM, foi o mesmo conjunto utilizado para treinar o *software SignalP*, afim de se avaliar a capacidade de diferenciação entre hélices de membrana e peptídeos sinais do TMHMM. Tal conjunto, composto por 1011 sequência eucarióticas, 141 Gram-positivas e 266 Gram-negativas, continha o peptídeos sinais e 30 aminoácidos após o sítio de clivagem [112].

Para hélices transmembranares o *TMHMM* obteve sucesso em cerca de 77-78% das topologias. A determinação dos resultados da predição de transmembranas se dá em três etapas [112]:

Etapa 1: Consiste em corrigir as incertezas nas fronteiras dos resultados (*e.g.*, Mudança de uma região de transmembrana para uma região interna ou externa da proteína). Para tal, uma área de doze resíduos (6 em cada parte) é transformada em um caractere curinga, estes caracteres denotam qualquer uma das três classes (Interna, Transmembrana, Externa), caso uma das regiões seja menor do que seis resíduos os dois resíduos centrais são mantidos enquanto que os demais são transformados em caracteres curinga. As áreas com caracteres curinga serão estimadas com a adição de ruído e posteriormente otimizadas conforme a probabilidade de suas ocorrências.

Etapa 2: Consiste em refazer a primeira etapa, porém com a criação de áreas de caracteres

curinga com dez resíduos ao invés de doze e sem a adição de ruído na etapa de estimativa.

Etapa 3: Nesta etapa o modelo gerado é otimizado novamente para então ser retornado ao usuário.

Como arquivo de entrada o *TMHMM* recebe um arquivo de sequência no formato `.fasta`. Como arquivo de saída, o *TMHMM* gera arquivos de texto com as porcentagens de suposição para as três classes, bem como um gráfico sobre esses dados para facilitar a visualização pelo usuário.

```
# 1ME4:A|PDBID|CHAIN|SEQUENCE
# AA     inside   membr    outside
1 A      0.16317  0.00000  0.83683
2 P      0.16317  0.00000  0.83683
3 A      0.16317  0.00000  0.83683
4 A      0.16317  0.00000  0.83683
5 V      0.16317  0.00000  0.83683
6 D      0.16317  0.00000  0.83683
7 W      0.16317  0.00000  0.83683
8 R      0.16317  0.00000  0.83683
9 A      0.16317  0.00000  0.83683
10 R     0.16317  0.00000  0.83683
11 G     0.16317  0.00000  0.83683
12 A     0.16317  0.00000  0.83683
13 V     0.16316  0.00001  0.83683
```

Figura 6.4: Fragmento do arquivo de saída do TMHMM da proteína 1ME4 [101] mostrando as porcentagens dos resíduos estarem em cada uma das três classes.

6.4 DSSP

O *DSSP* é um interpretador de estruturas secundárias do tipo α-hélice, resíduo isolado em uma ponte do tipo β, fita β estendida (com participação em uma folha β), 3-hélice, 5-hélice, volta por ligação de hidrogênio e dobras [100].

Como arquivo de entrada o *DSSP* recebe um arquivo de estrutura tridimensional de proteína no formato `.pdb`. Como resultado o *DSSP* retorna um arquivo de texto (Figura **??**) contendo as informações da estrutura interpretada.

#	RESIDUE	AA	STRUCTURE
1	1	A	
2	2	P	
3	3	A	S
4	4	A	
5	5	V	E
6	6	D	E
7	7	W	H
8	8	R	H
9	9	A	H
10	10	R	H
11	11	G	T
12	12	A	
13	13	V	
14	14	T	
15	15	A	
16	16	V	
17	17	K	
18	18	D	B
19	19	Q	
20	20	G	T

Figura 6.5: Fragmento do arquivo de saída do DSSP da proteína 1ME4 [101].

Neste arquivo de saída, a primeira coluna é um contador, a segunda coluna é o número do resíduo, na terceira coluna qual aminoácido que este resíduo é e na quarta coluna é o tipo de estrutura secundária prevista, a correspondência de cada letra com a sua estrutura secundária utilizada pelo MHOLline se encontra na Tabela 6.1.

Tabela 6.1: Relação da codificação da coluna "STRUCTURE"do arquivo de saída do DSSP e suas estruturas secundárias correspondentes.

Caractere	Estrutura correspondente
H, G, I	α-Hélice
B, E	Fita-β
I, T, S, " "	Alça

6.5 Molprobity

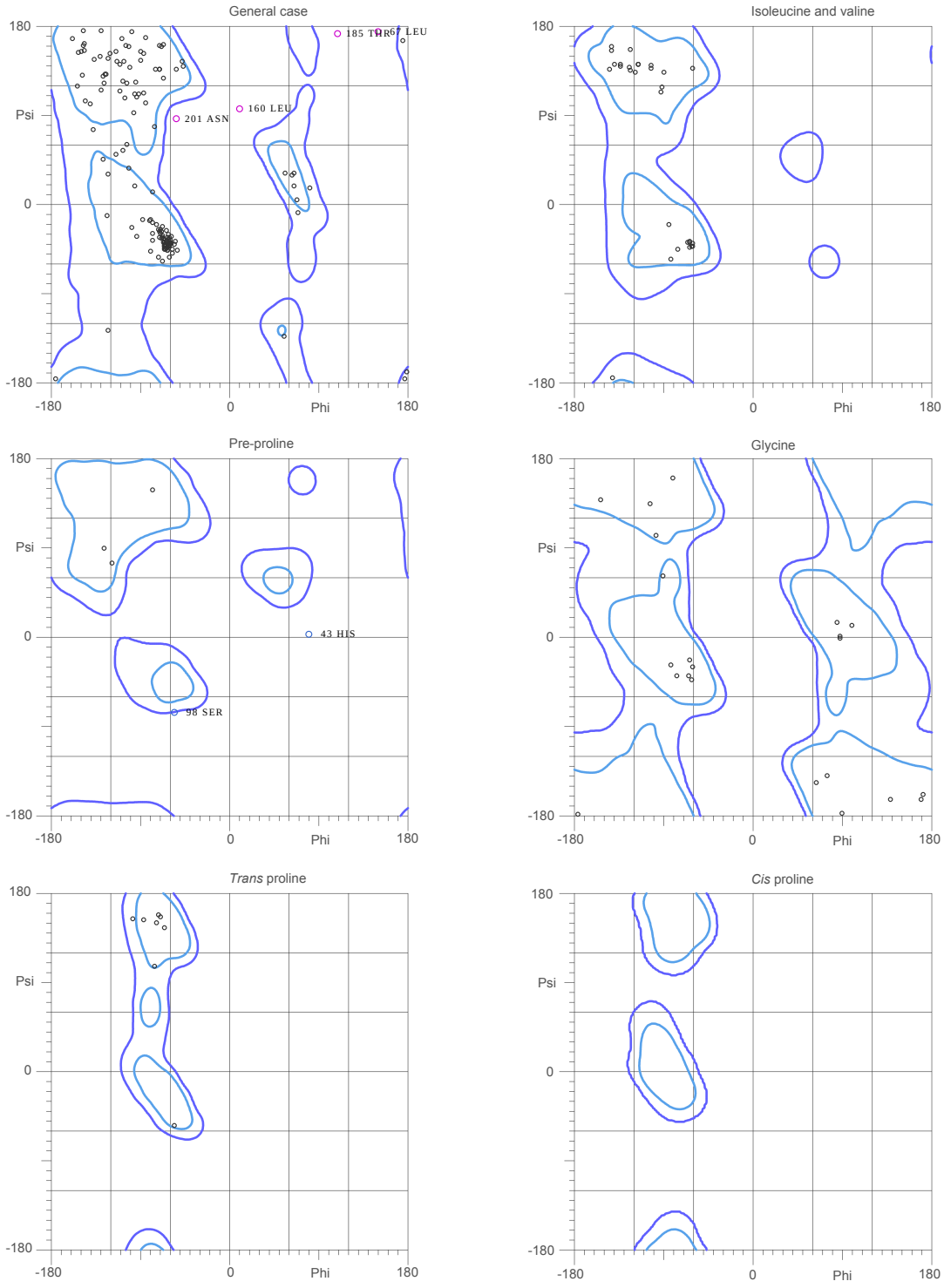
O *Molprobity* [83] é um avaliador de proteínas e estruturas de ácidos nucléicos nos níveis global e local. Primeiramente, o *Molprobity* detecta se o arquivo possui uma quantidade de hidrogênios satisfatória, visto que a presença destes é um pré-requisito exigido para a análise de contato entre átomos, caso a quantidade seja insuficiente a ferramenta REDUCE [113] é utilizada para inserir e otimizar os hidrogênios.

Após a etapa de verificação/adição de hidrogênios, a análise de contato entre átomos é feita a partir da ferramenta PROBE [114], que cria esferas com raio de $0.5\mathring{A}$ em volta da superfície de van der Waals dos átomos visando medir as sobreposições ocorrentes entre estes campos. Quando átomos que não possuem ligações coordenadas entre si se sobrepõem em mais que $0.4\mathring{A}$ é sinalizado um conflito forte. Esse tipo de sinalização indica que pelo menos um dos dois átomos foram modelados de forma incorreta, visto que conflitos como este não ocorrem na natureza. Como resultado desta etapa, uma pontuação chamada de '*clashscore*' é computada a partir do número de conflitos fortes/ 1000 átomos.

A próxima etapa consiste em analisar ângulos de torção, que compreende na verificação da distribuição dos resíduos da proteína em um espaço tridimensional de acordo com os valores φ e ψ no gráfico de Ramachandran para cadeia principal e do rotâmetro χ para as cadeias laterais. As distribuições de referência, utilizadas para construir os resultados desta etapa têm como base 10^5 resíduos em 500 arquivos. Como resultado desta etapa, 4 plotagens dos gráficos de Ramachandran são feitos, sendo uma para Glicinas (Gly), uma para Prolinas (Pro), uma para resíduos próximos à Prolinas (pre-Pro) e demais resíduos (Figura 6.6).

MolProbity Ramachandran analysis

1_1ME4APDBIDCH.B99990001.pdb, model 1



89.7% (191/213) of all residues were in favored (98%) regions.
97.2% (207/213) of all residues were in allowed (>99.8%) regions.

There were 6 outliers (phi, psi):
43 HIS (81.0, 3.3)
67 LEU (150.1, 175.7)
98 SER (-56.3, -76.0)
160 LEU (10.3, 98.0)
185 THR (109.1, 173.7)
201 ASN (-54.2, 87.6)

http://kinemage.biochem.duke.edu

Lovell, Davis, et al. Proteins 50:437 (2003)

Figura 6.6: Gráfico de Ramachandran da proteína 1ME4 gerado pelo *software* Molprobity.

A quarta e última análise executada é a de geometria covalente. Nesta, é examinado o tamanho e ângulos das ligações entre os átomos da cadeia principal, retornando os resíduos considerados como tendo geometria proibitiva (denominados de *outliers*).

Como arquivo de entrada, o *Molprobity* recebe um arquivo de estrutura tridimensional de proteína no formato `.pdb`. Como arquivo de saída o *Molprobity* retorna um arquivo no formato `.pdf` com o gráfico de Ramachandran, o número de *outliers* e a porcentagem de resíduos em regiões favoráveis e permitidas.

6.6 Jmol e Jsmol

O Jmol [115] e o JSmol [116] são *softwares* para visualizar e interagir com proteínas em um ambiente tridimensional, sendo o Jmol feito em java, podendo ser executado localmente, e o JSmol construído em Javascript, sendo adicionado à páginas *web* e podendo ser executado em uma quantidade maior de dispositivos. Durante a execução do MHOLline, o Jmol é utilizado para a geração de *thumbnails* das estruturas modeladas para complementar a página de resultados. O JSmol é utilizado para o usuário poder visualizar a proteína gerada pelo *workflow* principal do MHOLline eliminando a necessidade de baixar o arquivo de estrutura para visualização local. Seu arquivo de entrada é um arquivo de estrutura tridimensional de proteína no formato `.pdb`.

6.7 Refinamento

Por ser o objetivo principal desta dissertação, o desenvolvimento deste módulo bem como os testes aplicados serão descritos no capítulo 7.

6.8 Fluxo de dados

Nesta seção mostra-se o fluxo de dados no *workflow* do MHOLline2.0 com a ferramenta de ressubmissão. A Figura 6.1, no início deste capítulo, ilustra o fluxo de dados do MHOLline 2.0, com *input* e *output* de arquivos. O fluxo se inicia quando o usuário submete um arquivo de sequências do formato `.fasta`. Este arquivo serve como *input* dos *softwares* *TMHMM*, *HMMTOP*, *BLAST*, *SignalP* e *PSIPRED*. Os *outputs* dos *softwares* *TMHMM* (`.txt`) e *PSIPRED* (`.ss2`) são diretamente utilizados na ferramento de Ressubmissão.

O *output* do *software BLAST* (`.html`) é utilizado pelo *BATS*. O *BATS* é um *script* escrito em *PERL* usado para classificar os resultados do *BLAST*. Os resultados da classificação são armazenados no banco de dados MHOL*db*.

As proteínas que forem classificadas pelo *BATS* como grupo *G2* são então encaminhadas para a ferramenta *FILTERS*, um *script* em *PERL* criado para classificar as proteínas em 7 categorias (*Very High*, *High*, *Good*, *Medium to Good*, *Medium to Low*, *Low* e *Very Low*). Os resultados do *FILTERS* são também armazenados no banco de dados MHOL*db*. Todos os grupos classificatórios do *FILTERS* serão modelados pelo *Modeller*. Para isso, a sequência de aminoácidos de cada proteína é alinhada com a sequência de melhor resultado selecionado pelo *BATS*, lido do banco de dados de estrutura 3D (*PDB*). Esse alinhamento junto com um conjunto de *scripts* são encaminhados para a execução do *Modeller* para a geração dos modelos 3D.

O *ECNGET* é uma ferramenta que utiliza o *ID* do *PDB* que obteve o melhor resultado no *BATS* para cada sequência de entrada a fim de buscar o seu *Enzyme Comission Number*, já processado e armazenado no MHOL*db*. A partir do resultado do *Modeller* (`.pdb`), o *JMOL* é usado para gerar um *thumbnail* (`.png`) da proteína modelada e para visualizar na interface do MHOL*web* em tempo real o modelo gerado. As análises de Ramachandran são executadas pelo *Procheck* gerando arquivos *Post Script* (`.ps`) e texto e pelo *Molprobity* gerando arquivos no formato `.pdf`. Os resultados de análise de energia do *Modeller* e as informações sobre a qualidade estereoquímica gerada pelo *Molprobity* (*outliers* e porcentagem de resíduos em regiões permitidas) são usados na ferramenta de Ressubmissão.

Na Ressubmissão, os dados coletados dos *softwares TMHMM*, *PSIPRED*, *SignalP*, *Molprobity* e *BLAST* são usados para prover os usuário com informações relevantes que possam auxiliá-lo no refinamento manual, bem como tornar possível o refinamento automático.

6.9 Banco de dados MHOLdb no MHOLline 2.0

O banco de dados do MHOLline 2.0 atualmente possui 16 tabelas e 13 chaves estrangeiras, como pode ser verificado na Figura 6.7. É possível notar que as tabelas “ecnget”, “chain”, “resolution” e “sistema” não possuem chaves secundárias, porém elas

se relacionam através do registro “pdb_id”, também presente nas tabelas “refinement”, “summary”, “bats_filter” e “champ”. O motivo pelo qual não se colocou uma chave estrangeira para estas tabelas se deu pela intenção de se reduzir o cruzamento de tabelas, que aumentam o tempo de consulta ao banco de dados.

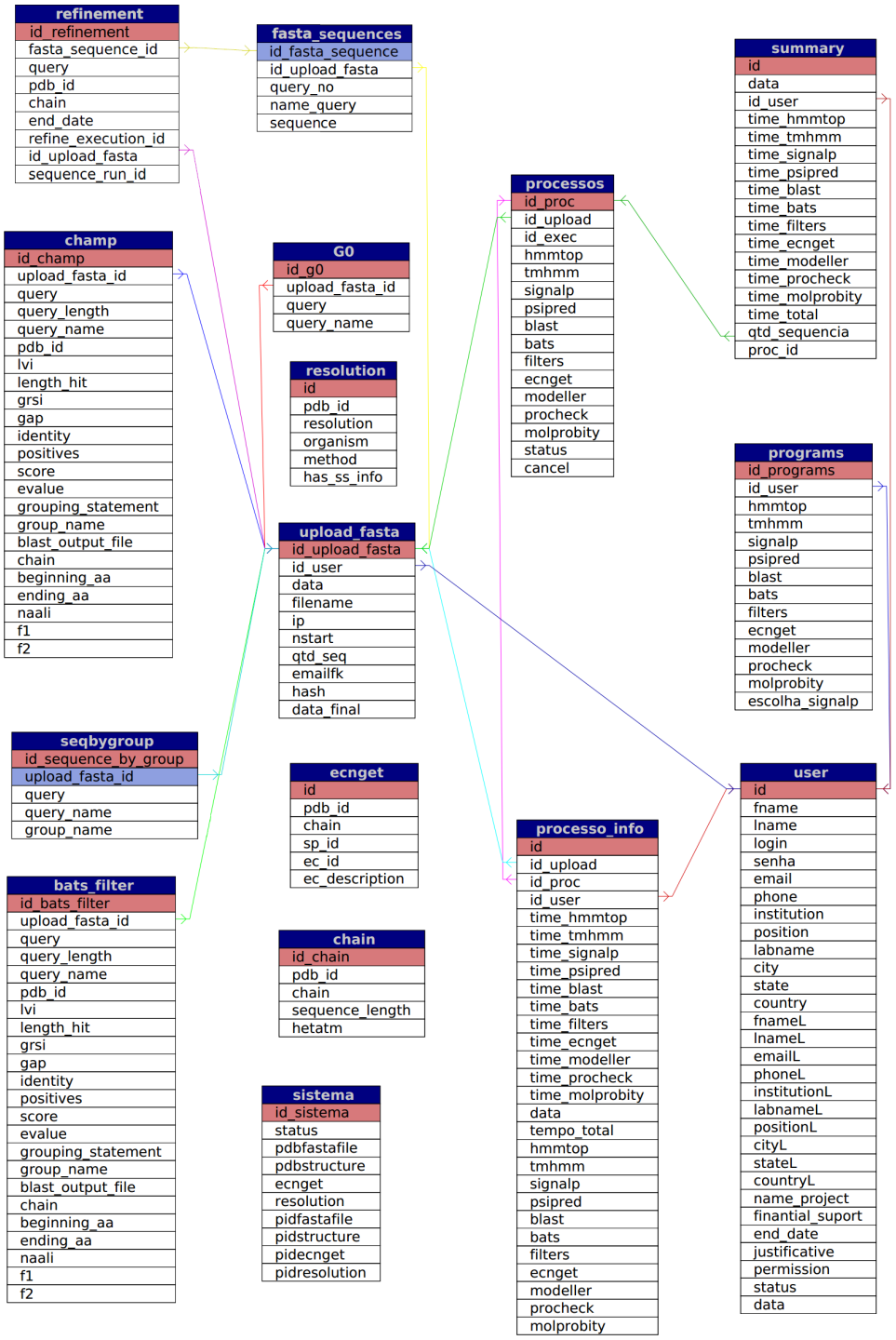


Figura 6.7: Modelo entidade-relacionamento do banco de dados MHOLdb do MHOLline 2.0. Modelo gerado através do *software* phpMyAdmin [117].

6.10 Controle de versão do MHOLline 2.0

O controle de versão do MHOLline 2.0 é feito através de *backups* periódicos. Para tal, um *script* escrito na linguagem *shellscript* foi desenvolvido onde todas as pastas e o banco de dados são compactados separadamente e alocados em um diretório próprio em um *hard disk* (HD) separado, possuindo como nome o identificador de origem (MHOLcore, MHOLweb ou MHOLdb), acrescidos da data do *backup* de forma a manter o histórico de mudanças no MHOLline. Atualmente este *script* roda manualmente, porém é possível criar uma rotina de *backup* periódico através do uso da ferramenta *crontab* do sistema UNIX.

6.11 Validação do MHOLline 2.0

Em agosto de 2016 foi feito um teste de carga no MHOLline 2.0, visto que neste período ocorreu a VIII EMMSB (Escola de Modelagem Molecular de Sisteas Biológicos) [118] no LNCC. Durante a escola os usuários presentes fizeram o *feedback* sobre as interfaces e as funcionalidades do MHOLline 2.0.

Os *feedbacks* fornecidos sobre os *softwares* incorporados foram todos positivos, já que o MHOLline passou a prover uma quantidade maior de informações sobre as proteínas submetidas. Quanto à área de refinamento, os *feedbacks* sobre a interface também foram todos positivos, com destaque para a apresentação simples e direta de informações úteis aos pesquisadores.

Também foram recebidos *feedbacks* a respeito de funcionalidades desejadas, como por exemplo, a adição do nome do organismo a qual cada modelo da lista de possíveis *templates* pertence. As sugestões apresentadas pelos usuários foram implementadas e já estão em funcionamento do *worflow*. Algumas sugestões foram implementadas ainda durante a EMMSB e os usuários também as elogiaram.

7 Desenvolvimento da Ferramenta de Refinamento de Proteínas

A área de refinamento consiste em uma sessão do MHOLline voltada especificamente para usuários cadastrados (com *login*). Nela é possível analisar os resultados da saída do *workflow* original do MHOLline com algumas informações a mais em uma interface única, e fazer algumas alterações tais como a troca do molde ou a adição de até no máximo três moldes (para alinhamentos múltiplos), para serem utilizados em uma nova modelagem comparativa executada pelo *software* Modeller [58]. Existem dois tipos de refinamento que o MHOLline é capaz de realizar, o automático e o manual, ambos serão descritos a seguir.

7.1 Refinamento Automático

Caso o usuário opte pelo refinamento automático, as regiões de *gap* no alinhamento são mapeadas, ou seja, cada sequência molde tem os caracteres de seu alinhamento classificados como: (i) alinhamento com *gap* ou (ii) alinhamento sem *gap*. Em um alinhamento, o *gap* representa uma inserção ou uma deleção de um resíduo e a sua representação é feita pelo sinal "-", como pode ser observado no alinhamento da Figura 7.1.

```
query      MGKEKKKATNKDGGNDGGGGGKAAKDTSGGSGYTKVKVRHILCEKHGRAMEALKKINEGSSFADVAREYSEDKAR
template   ------------------------------NA---VKVRHILCEKHGKIMEAMEKLKSGMRFNEVAAQYSEDKAR

query      SGGDLGWKRRGEMVGPFQEAAFALPKGGM-----TPEPVKTSFGYHIILVEDKQ
template   QGGDLGWMTRGSMVGPFQEAAFALPVSGMDKPVFTDPPVKTKFGYHIIMVEGRK
```

Figura 7.1: Exemplo de alinhamento com *gaps*.

Um consenso de *gaps* a partir das sequências do molde é gerado da seguinte forma:

(Passo 1) Se houver somente uma sequência molde, o consenso será a própria sequência, caso contrário, uma operação lógica *AND* será realizada entre as sequências dos moldes. Tal operação é codificada da seguinte maneira: 1 - Posições com *gap*; 0 - Posições sem *gap*. O resultado desta operação é uma sequência que onde houver *gap*, significa que existe um *gap* naquela posição em todas sequências moldes.

(Passo 2) No passo seguinte, essa sequência codificada fará uma operação lógica *OR*, desta vez, com o resultado da predição do PSIPRED (Figura7.2-a), sendo agora recodificada da seguinte maneira: 1 - Existe nessa posição uma predição de estrutura secundária do tipo α-Hélice ou Fita-β; 0 - Caso contrário.

(Passo 3) O resultado dessas operações é combinado (Figura7.2-b), gerando-se uma sequência final onde 1 é o resultado da predição do PSIPRED para regiões com *gap* na sequência consenso do Passo 1, e 0 é um *gap* (-).

(Passo 4) Este resultado será então variado em tamanho e posição, respeitando os limites nos quais as regiões com restrição de estrutura secundária (predita pelo PSIPRED) foi delimitada (Figura7.2-c).

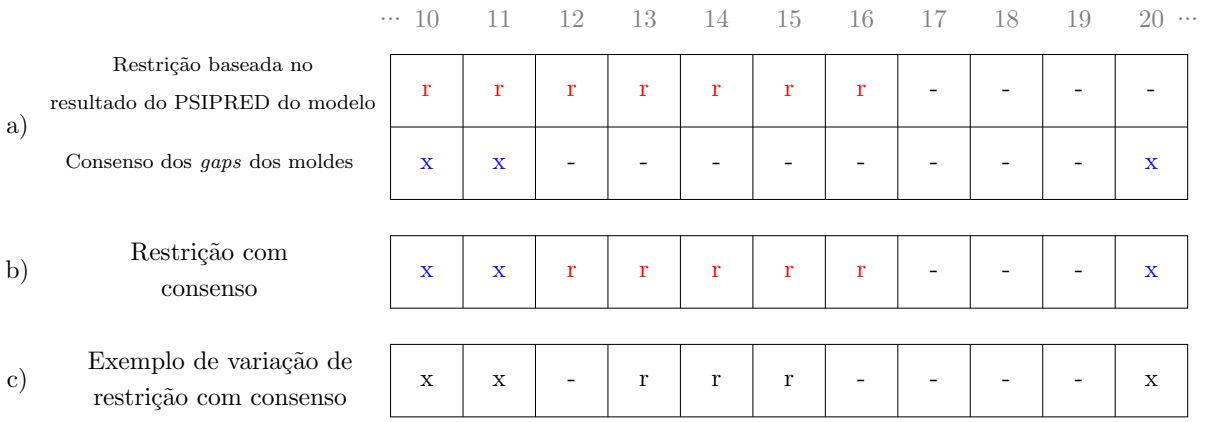


Figura 7.2: Operação de aplicação de restrição no modo de refinmento automático. Sendo "r": Restrição, "-": *Gap* e "x": Não *gap*. a) Checagem se algum fragmento de resultado do PSIPRED se localiza em uma região de *gap*. b) Resultado da checagem executada na etapa anterior. c) Variação do resultado da etapa anterior para gerar várias combinações de restrição.

Um pseudocódigo ilustrando o processo de criação das restrições pode se conferido no Algoritmo 1 contido no Apêndice deste trabalho.

Durante a etapa de criação das restrições, a quantidade de combinações possíveis, levando em consideração as restrições existentes, é calculada para determinar o modo como o refinamento automático irá tratar os resultados. O cálculo da probabilidade é feito da seguinte maneira: (i) Cada tipo de restrição é calculada separadamente como uma combinação simples de n elementos dois a dois, onde n é a diferença entre o número do resíduo final e o inicial da restrição; (ii) Após calcular cada tipo de restrição separadamente, novamente uma combinação simples entre os resultados das combinações individuais é

feita a fim de se descobrir a quantidade m total de combinações possíveis para as restrições dadas.

No MHOLline2 foi implementada uma tomada de decisão automática baseada no valor de m. Este número serve como ponto de corte para determinar como os resultados da modelagem serão dispostos para o usuário. Existem três abordagens atualmente implementadas, descritas também no fluxograma da figura 7.3:

- $m \geq 50$: Existem cinquenta ou mais combinações de restrição. Neste caso uma seleção aleatória de 50 combinações é feita para serem modeladas e agrupadas (de acordo com o método apresentado no seção 7.5) antes de serem retornadas para o usuário;

- $20 < m < 50$: Existem mais que vinte e menos de cinquenta combinações. Todas as combinações possíveis são modeladas e agrupadas (de acordo com o método apresentado no seção 7.5) antes de serem retornadas para o usuário;

- $m < 20$: Existem vinte ou menos combinações. Nesse último caso, todas as combinações são modeladas e retornadas para o usuário.

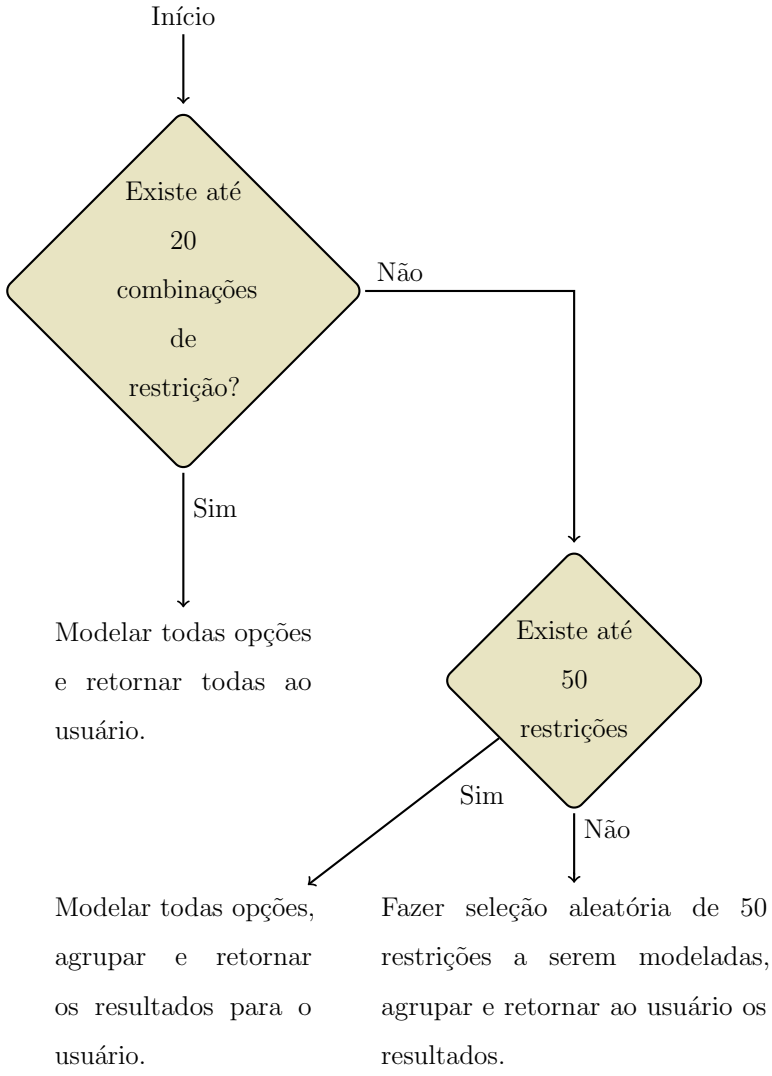


Figura 7.3: Esquema das tomadas de decisões sobre como os resultados serão retornados para o usuário no refinamento automático.

Na ausência de regiões com estrutura secundária predita pelo PSIPRED situadas em regiões de *gap*, as regiões de *loop* (ou voltas) nessas regiões são otimizadas e os resultados agrupados. Por fim, caso não haja *gap* no alinhamento, 20 proteínas serão geradas sem qualquer restrição ou otimização e os resultados serão retornados para o usuário.

7.2 Refinamento Manual

Caso o usuário opte pelo refinamento manual é possível três opções, como descritas abaixo e mostrado na Figura 7.6 :

- Fazer a clivagem de peptídeo sinal, caso exista e tenha sido detectada pelo

SignalP através de um botão, eliminando a necessidade do usuário fazer a deleção manualmente e preencher a região com *gaps*, ou até mesmo refazer o alinhamento;

- Adicionar restrições de estrutura secundária no Modeller, como hélices, fitas, folhas e distância entre átomos, fazendo com que o próprio MHOLline gere o *script* em *Python* (Figura 7.4) para ser executado no Modeller com o uso da biblioteca paralela;

```
from modeller import *
from modeller.parallel import *
from modeller.automodel import *
from modeller.scripts import complete_pdb

class MyModel(automodel):
        def special_restraints(self, aln):
                rsr = self.restraints
                at = self.atoms
#====== Alpha helix:=============================================
                rsr.add(secondary_structure.alpha(
                                self.residue_range('48:','57:')))
#====== Strand:==================================================
                rsr.add(secondary_structure.strand(
                                self.residue_range('60:','66:')))
#====== Sheet:===================================================
                rsr.add(secondary_structure.strand(
                                self.residue_range('25:','32:')))
                rsr.add(secondary_structure.strand(
                                self.residue_range('40:','44:')))
                rsr.add(secondary_structure.sheet(at['N:25'],at['O:44'],
                                sheet_h_bonds=-5))
#================================================================

a = MyModel(env, alnfile='1_1ME4APDBIDCH.ali', knowns=('1F2A_A'),
sequence='1_1ME4APDBIDCH', assess_methods=(assess.DOPE, assess.DOPEHR,
assess.normalized_dope, assess.GA341))

a.starting_model = 1
a.ending_model = 1

a.library_schedule = autosched.slow
a.max_var_iterations = 300
a.md_level = refine.slow
a.repeat_optimization = 2
a.max_molpdf = 1e6

a.make()
```

Figura 7.4: Código em Python de restrição de estrutura secundária no Modeller.

- Otimizar *loops* com o Modeller, gerando o *script* em *Python* (Figura 7.5) do Modeller com o uso da biblioteca paralela, com todas otimizações solicitadas pelo usuário.

```
from modeller import *
from modeller.automodel import *

# Create a new class based on 'loopmodel' so that we can redefine
# select_loop_atoms (necessary)
class MyLoop(loopmodel):

# This routine picks the residues to be refined by loop modeling
        def select_loop_atoms(self):

# One or more loops from residue X to Y inclusive
                return selection(
                        self.residue_range('28:', '31:'),
                        self.residue_range('45:', '47:'),
                        self.residue_range('55:', '60:')
                        )
```

Figura 7.5: Código em Python de uma classe contendo parâmetros de otimização de *loops* no Modeller. A necessidade de uma classe em um arquivo separada foi necessária pois ao se usar a bibloteca paralela, cada processo precisa ter acesso aos parâmetros de restrição. A solução encontrada foi separar a classe em um arquivo próprio. No outro arquivo, os comandos contidos são os mesmos da Figura 7.4, excluindo-se a classe "MyModel"e a classe instanciada será a classe criada "MyLoop"e não a deletada "MyModel".

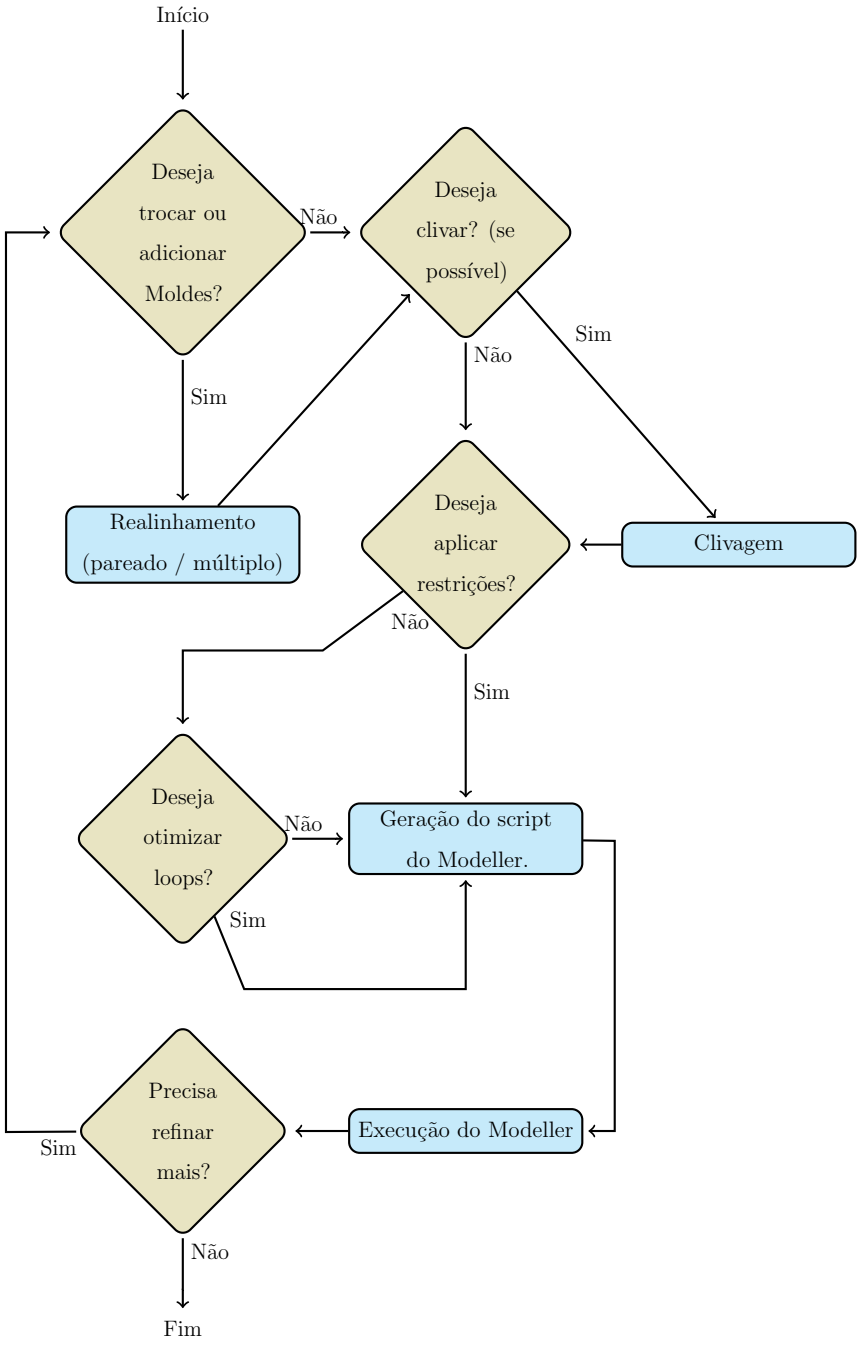


Figura 7.6: Esquema da etapa de refinamento manual no MHOLline2.0.

As funcionalidades de realinhamento, clivagem, restrição e otimização foram criadas para reduzir o trabalho do usuário em realizar estas etapas manualmente, através de uma interface amigável e intuitiva. Ao se escolher um único molde novo, o *script* do Modeller *salign*, para alinhamentos pareados é usado. Caso o usuário escolha mais de um molde, o *software* ClustalΩ é utilizado. O motivo da escolha de *softwares* diferentes para alinhamentos múltiplos e pareados se deu pois o ClustalΩ produziu um resultado de qualidade superior que o *salign*, porém em um alinhamento pareado, o *salign* gerou um

resultado melhor que o ClustalΩ. O processo de seleção do algoritmo de alinhamento se encontra na Seção 7.4.

7.3 Redução no tempo de execução do Modeller

Visando escolher as melhores condições para uma melhor experiência do usuário (*e.g.*, menor tempo de processamento) e um resultado de qualidade, algumas otimizações foram feitas na execução do Modeller. O Modeller possui uma biblioteca paralela, introduzida na versão 9.1 lançada em 2007, que é capaz de paralelizar seus cálculos em múltiplos processadores. Quando se constrói dois modelos ou mais em um mesmo *script*, o módulo paralelo executa uma instância do Modeller para cada modelo, por exemplo, em um *script* onde se constrói 10 modelos deseja-se distribuir os trabalhos para dois núcleos, esperando-se que cada núcleo irá modelar 5 proteínas cada [65].

Por ainda ser uma biblioteca experimental, realizamos testes de tempo e qualidade em um computador com a seguinte configuração: Processador Intel® $Core^{TM}2$ *Quad* CPU $Q8200$ com clock de $2.33GHz \times 3$, placa de vídeo $NVIDIA$ $GTX470$ com *driver* *Gallium* 0.4 *on* $NVC0$ visto que apresentou maior estabilidade no sistema que o *driver* da $NVIDIA$, $4GB$ de memória RAM ($2 \times 2GB$) com frequência de $800MHz$ cada, Sistema Operacional *Linux Ubuntu* 14.04 $Trusty$ $Tahr$ ($x86\_x64$). Esta configuração foi escolhida pois se assemelha muito à configuração atual do servidor que hospeda atualmente a versão *web*.

A sequência fasta da cadeia A da proteína 5FPY [119], contendo 666 aminoácidos, foi submetida ao MHOLline2.0 que sugeriu como molde a cadeia B da proteína 1CU1 [120], contendo 645 aminoácidos. Porém, na etapa de refinamento o seu molde foi propositalmente modificado para a cadeia A da proteína 1HEI [121], contento 451 aminoácidos. O motivo desta mudança, apesar de 5FPY e 1CU1 serem hidrolases e a 1HEI ser uma helicase, deu-se pela intenção de avaliar o desempenho de tempo de modelagem do Modeller com e sem a biblioteca paralela, e não o de gerar um modelo de alta qualidade. O objetivo deste ensaio era ter um teste controlado para comparação de tempo de uma proteína em condições ideais e de restrição. Para tal, precisava-se de uma proteína com uma cobertura no alinhamento menor, que possibilitasse a função automática de refinamento do MHOLline2.0 gerar restrições a serem aplicadas pelo Modeller.

As restrições aplicadas neste teste foram:

- α-hélices: do resíduo 48 ao 57 e do resíduo 207 ao 215;
- Folha-β com fitas antiparalelas: do resíduo 25 ao 32 e do 40 ao 44 iniciando no nitrogênio (N) do resíduo 25 e finalizando no oxigênio (O) do resíduo 44 com 5 ligações de hidrogênio (HB), do resíduo 25 ao 32 e do 68 ao 72 iniciando no resíduo 25(N) e finalizando no resíduo 72(O) com 5 HB e do resíduo 25 ao 32 e do 77 ao 83 iniciando no resíduo 25(N) e finalizando no resíduo 83(O) com 5 HB;
- Folha-β com fitas antiparalelas: do resíduo 116 ao 121 e do 86 ao 90 iniciando no resíduo 86(N) e finalizando no resíduo 121(O) com 5 HB e do resíduo 116 ao 121 e do 109 ao 112 iniciando no resíduo 109(N) e finalizando no resíduo 116(O) com 4 HB;
- Folha-β com fitas antiparalelas: do resíduo 138 ao 143 e do 146 ao 153 iniciando no resíduo 138(N) e finalizando no resíduo 153(O) com 7 HB e do resíduo 138 ao 143 e do 146 ao 153 iniciando no 138(N) e finalizando no resíduo 153(O) com 6 HB;
- Folha-β com fitas antiparalelas: do resíduo 183 ao 195 e do 198 ao 206 iniciando no resíduo 183(N) e finalizando no resíduo 206(O) com 9 HB, do resíduo 183 ao 195 e do 662 ao 665 iniciando no resíduo 183(N) e finalizando no resíduo 665(O) com 4 HB e do resíduo 183 ao 195 e do 159 ao 161 iniciando no resíduo 159(N) e finalizando no resíduo 161(O) com 3 HB.

Adicionalmente, outras quatro proteínas classificadas como *Very High* foram remodeladas sem qualquer restrição ou mudança do molde, a fim de se verificar apenas se o desempenho no processamento e qualidade dos modelos gerados seriam semelhantes aos do modelo originalmente gerado pelo *workflow*. Detalhes do conjunto teste pode ser observada na Tabela 7.1

Tabela 7.1: Informações do conjunto de proteínas usadas nos testes de tempo e de comparação entre os modos automático e manual.

Modelo						Molde						Refinamento	
PDB ID	Cadeia	Nome	NR	SCOP[a]	CATH[b]	PDB ID	Cadeia	Nome	NR	SCOP[a]	CATH[b]	Região sem cobertura	Restrição
2W4J	A	*Death-associated protein kinase 1*	277	NC	α/β	1JKS	A	*Death-associated protein kinase*	294	α/β	α/β	1/ 278-281	NC
1JG8	A	*L-allo-threonine aldolase*	347	α/β	α/β	1JG8	D	*L-allo-threonine aldolase*	347	α/β	α/β	1-3/ 203	NC
5M5U	A	*Clathrin heavy chain 1*	365	NC	NC	1UTC	B	*Clathrin heavy chain*	363	β	β	1-4/ 358-365	NC
1HI8	A	*P2 protein*	664	α/β	NC	1UVJ	B	*P2 protein*	664	α/β	NC	NC	NC
5FPY[c]	A	*Serine protease NS3*	666	NC	NC	1HEI	A	*HCV helicase*	451	α/β	NC	1-221/ 663-666	α-hélices: 48-57; 207-215. Pares de fitas-β antiparalelas: 25-32/ 40-44; 25-32/ 68-72; 25-32/ 77-83; 116-121/ 86-90; 116-121/ 109-112; 138-143/ 146-153; 138-143/ 146-153; 183-195/ 198-206; 183-195/ 662-665; 183-195/ 159-161.

NR: número de resíduos. NC: não contém. [a] Classificação de acordo com o SCOP [122]. [b] Classificação de acordo com o CATH [123]. [c]Esta proteína teve seu molde alterado para simular de modo controlado uma condição de uma proteína com cobertura reduzida.

Para os testes de tempo três modelagens de cada proteína foram executadas gerando 20 modelos em cada execução, no caso paralelo foram utilizados duas threads, divididas em um núcleo cada, para não comprometer o processamento do servidor, caso múltiplos refinamentos sejam disparados simultaneamente no servidor, sendo medidos com a função *time* do Linux. As modelagens foram executadas uma por vez e o computador não foi utilizado para quaisquer tarefas adicionais. Então a partir dos três resultados de tempo real, a média aritmética simples dos tempos medidos foi calculada para se obter o tempo médio de execução do Modeller com e sem a biblioteca paralela.

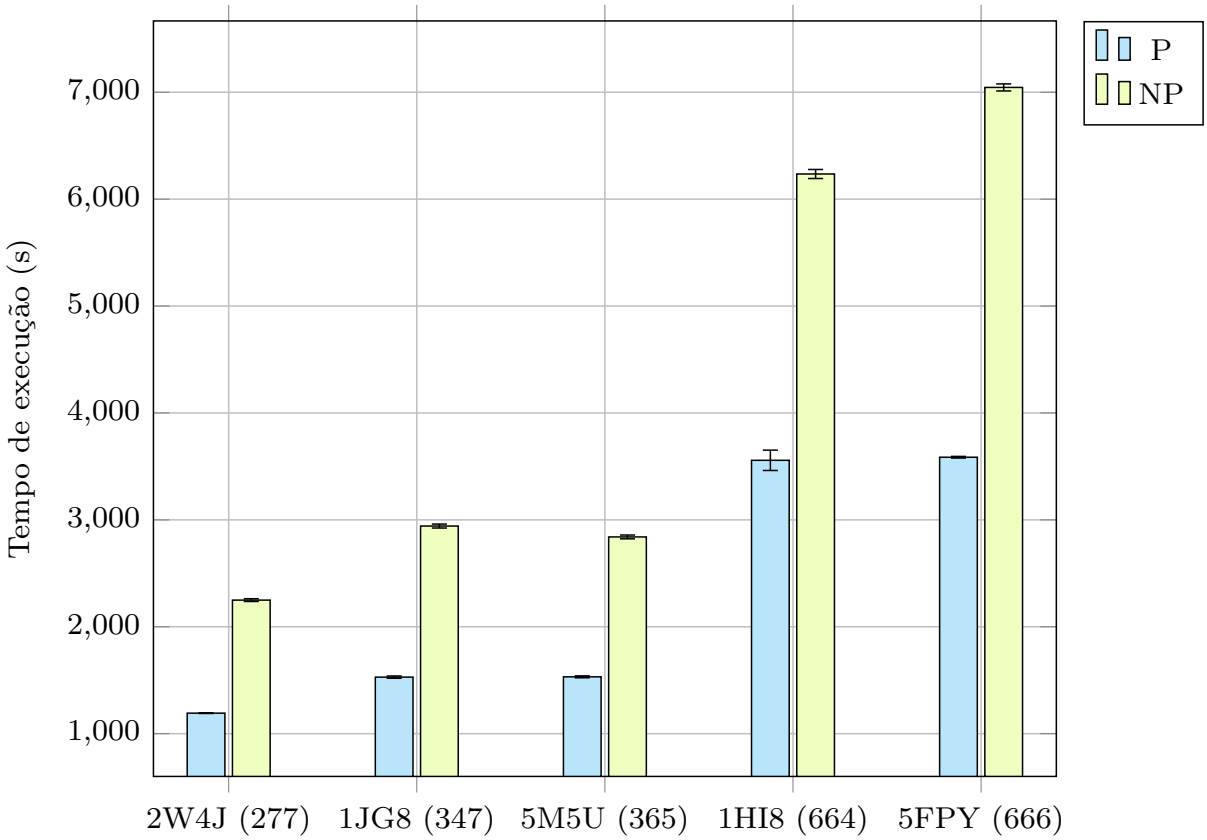


Figura 7.7: Média do tempo real (em segundos) de três execuções de modelagem do conjunto teste. Modelos gerados com (P, em azul) e sem (NP, em verde) a biblioteca paralela do Modeller. Entre parênteses está descriminado o número de resíduos de aminoácidos de cada proteína.

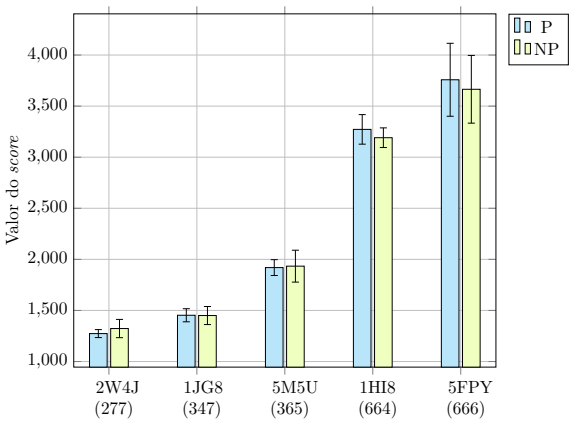
A partir da comparação dos resultados, exibidos na Figura7.7 conclui-se que, a respeito do tempo, a modelagem paralela se mostrou muito superior, obtendo uma melhora de cerca de 200% em todos os casos, como pode ser conferida na tabela 7.2

Tabela 7.2: Comparação entre o tempo real de execução do conjunto teste usando ou não o pacote paralelo do Modeller.

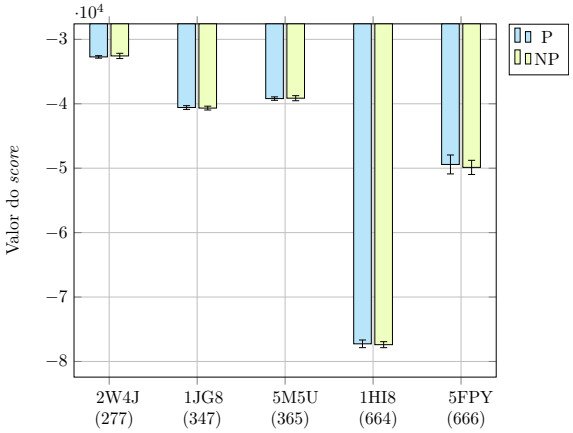
ID	NR[a]	Paralelo		Não Paralelo	
		Média	RMSD	Média	RMSD
2W4J	277	1192,157	3,039	2249,483	13,047
1JG8	347	1529,130	10,674	2840,477	17,875
5M5U	365	1531,863	8,943	2942,307	18,600
1HI8	664	3557,173	95,115	6235,250	42,400
5FPY [b]	666	3585,933	7,840	7044,951	32,909
5FPY [c]	666	4238,287	152,829	8137,087	107,560

[a]NR: número de resíduos de aminoácidos. [b] Proteína modelada com mudança de molde (de 1CU1 para 1HEI) e com restrições. [c] Proteína modelada sem mudança de molde e sem restrições.

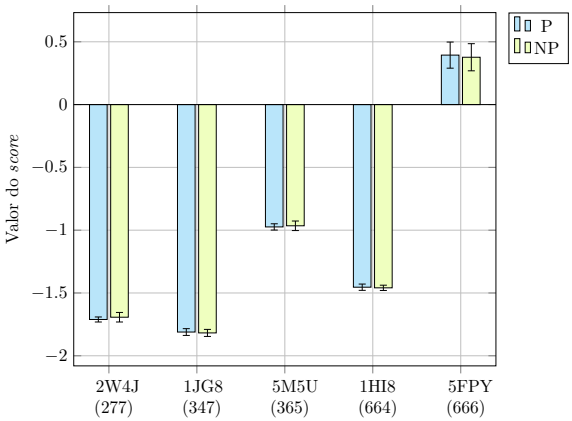
Em relação à qualidade, os 20 modelos gerados para cada proteína tiveram os valores de pontuações de energia calculadas pelo Modeller: molpdf (Figura 7.8a), nDOPE (Figura 7.8c) e DOPE-HR (Figura 7.8b). O desvio médio quadrático (RMSD) (Figura 7.8d) dos 20 modelos foi calculado com a ferramenta Pymol [124] e os valores de *outlier* (Figura 7.8e) e resíduos alocados em regiões favoráveis (Figura 7.8f) computados com o MolProbity. O resultado de cada método é a média aritmética simples dos valores obtidos. A partir destes resultados, conclui-se que o uso da biblioteca paralela não interfere negativamente na geração dos modelos do Modeller.



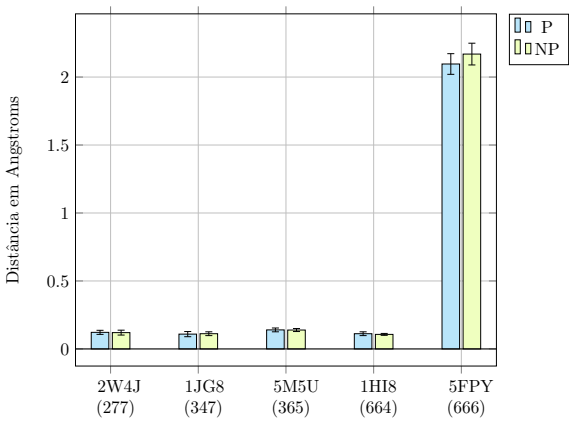
(a) Molpdf *score*.



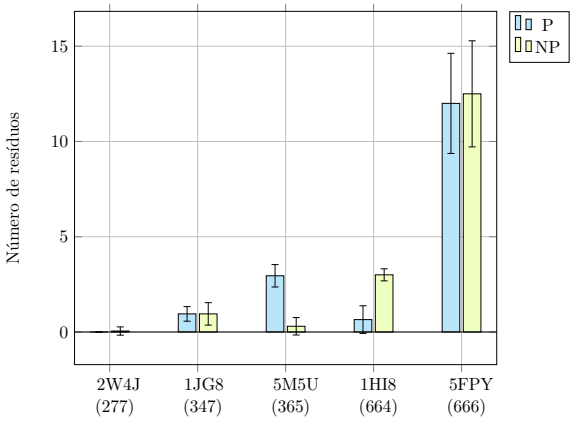
(b) DOPE-HR *score*.



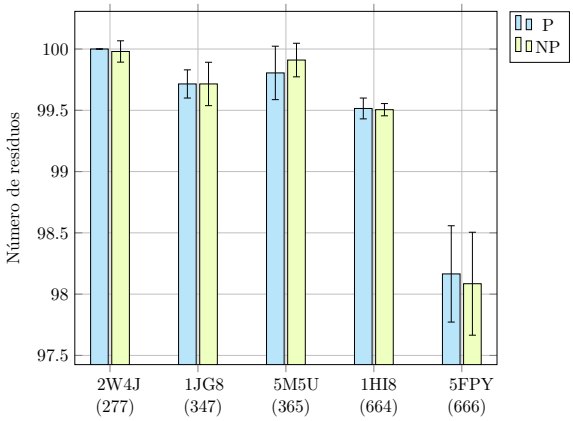
(c) nDOPE *score*.



(d) RMSD.



(e) *Outliers*



(f) Resíduos favoráveis.

Figura 7.8: Comparação dos resultados obtidos para o teste da biblioteca paralela do Modeller. Os valores de *Molpdf*, *DOPE-HR* e *nDope scores* foram calculados com o *software Modeller* e o desvio padrão foi calculado com a ferramenta *Pymol* para 20 modelos gerados com (P, em azul) e sem (NP, em verde) a biblioteca paralela do *Modeller*. Entre parênteses está descriminado o número de resíduos de aminoácidos de cada proteína.

A partir destes resultados, levando em consideração a diferença não significativa da qualidade dos modelos e da diferença significativa na redução do tempo de execução, concluiu-se que o uso da biblioteca paralela do Modeller pode ser usada para agilizar o refinamento. Esta opção será inclusa quando mais de uma proteína for modelada pelo mesmo *script* do Modeller, caso contrário, o uso desta biblioteca não oferece vantagens no tempo de processamento.

7.4 Escolha do *software* de alinhamento de sequências

Outro problema a ser resolvido na ferramenta de refinamento e remodelagem de proteínas é a escolha do *software* de alinhamento a ser usado. Os testes foram conduzidos para se selecionar qual seria o melhor *software* para alinhamento múltiplo e para alinhamento pareado.

7.4.1 Testes de alinhamento múltiplo

Os *softwares* analisados para alinhamento global múltiplo foram o ClustalΩ [125], MUSCLE [126] e T-COFFEE [127].

As análises foram conduzidas utilizando a proteína cruzipaína (PDB ID 3IUT) [128], um monômero de 221 aminoácidos com resolução de 1.2Å pertencente ao organismo *Trypanossoma cruzi*. Essa proteína foi alinhada contra as três proteínas que obtiveram os menores *e-value* em alinhamento com BLAST:

- 3HD3 [129], homodímero de 215 aminoácidos com $e\text{-}value = 1e - 157$;
- 1EWP [130] monômero de 215 aminoácidos com $e\text{-}value = 3e - 156$;
- 4XUI [131] homotrímero de 216 aminoácidos com $e\text{-}value = 3e - 156$.

Foi gerado um arquivo único com todas as sequências e então utilizado como entrada para todos os *softwares* de teste. Somente o tempo foi levado em consideração nesta seção pois a qualidade do alinhamento gerado foi muito semelhante ou igual entre os três programas. Para tal análise, o tempo de execução de cada *software* foi medido com a função *time* do linux utilizando o tempo real. Cada *software* foi executado dez vezes para

reduzir qualquer interferência que processos secundários do sistema operacional tenham causado, e a média e desvio padrão dos tempos foram calculados. Os resultados desta análise estão dispostos na Figura 7.9.

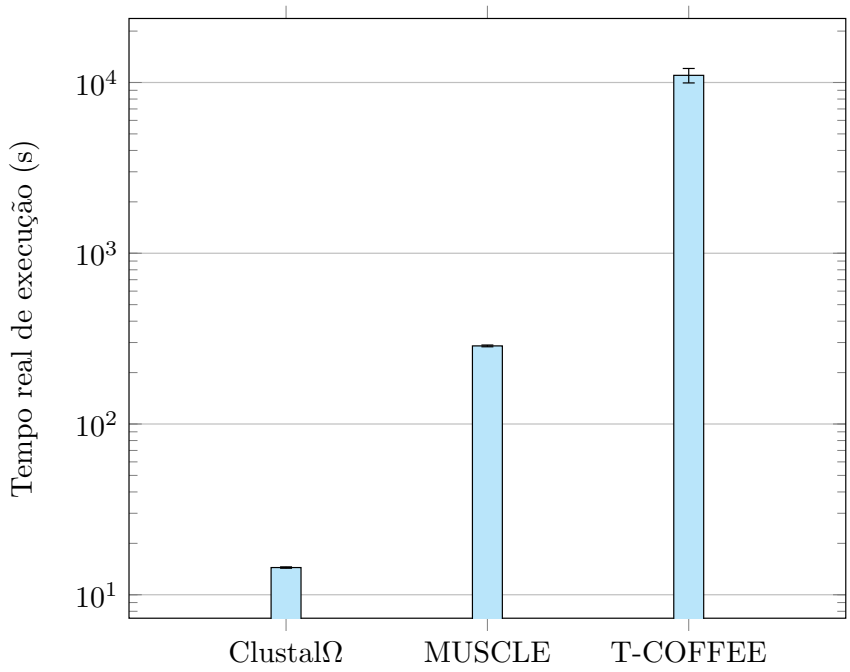


Figura 7.9: Tempo médio real de execução dos *softwares* de alinhamento global múltiplo ClustalΩ, MUSCLE e T-COFFEE. Os valores médios foram obtidos a partir de 10 execuções de cada *software*.

A partir destes resultados, podemos observar que o clustalΩ foi mais eficiente quanto ao tempo de execução, sendo assim selecionado para uso na etapa de alinhamento múltiplo e realização de testes de alinhamento pareado.

7.4.2 Testes de alinhamento pareado

Os testes de alinhamento global pareado foram executados com a ferramenta clustalΩ e a função *salign* do Modeller. Cada *software* foi executado dez vezes para reduzir qualquer interferência que processos secundários do sistema operacional tenham causado, e a média e desvio padrão dos tempos foram calculados.

Para este teste, a proteína 5FPY de 666 resíduos foi alinhada com a proteína 1HEI de 451 resíduos e o seu tempo real de execução foi medido com a função *time* do Linux (Figura 7.10). A razão da escolha de um molde com tamanho menor que o modelo se deve à intenção de forçar o algoritmo de alinhamento a inserir *gaps* nas sequências, a fim de se

testar o desempenho dessas duas ferramentas.

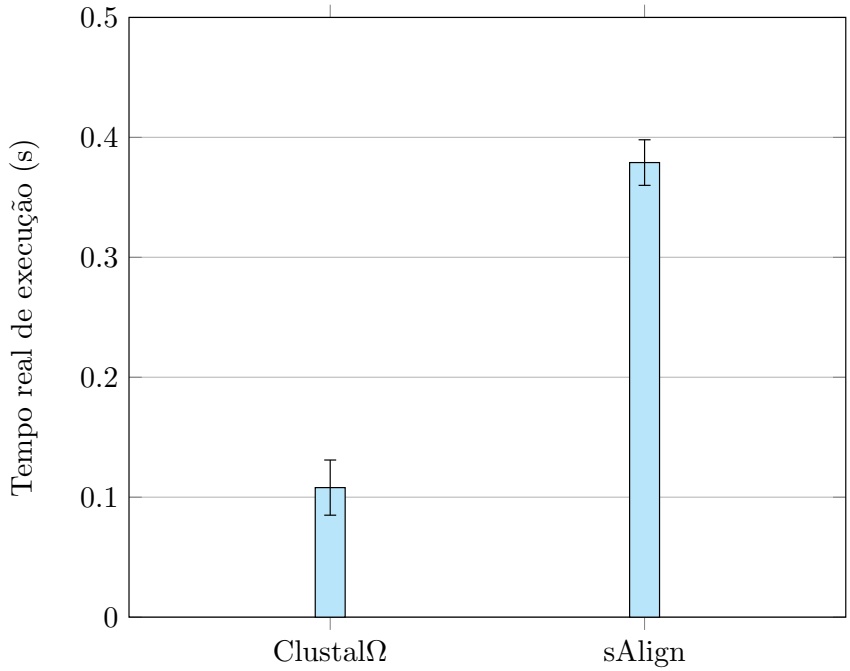


Figura 7.10: Tempo médio real de execução dos *softwares* de alinhamento global pareado *salign* e ClustalΩ. Os valores médios foram obtidos a partir de 10 execuções de cada *software*.

A qualidade do alinhamento gerado por ambas abordagens foi idêntica, permitindo comparar diretamente o tempo de execução. A diferença de tempo do ClustalΩ em relação ao *salign* pode ser compensada pela necessidade de reprocessamento dos dados de saída do ClustalΩ e a falta dela para os dados do *salign*, visto que este último já produz resultados no formato de entrada do Modeller, evitando assim reprocessamentos. Outro motivo pelo qual o *salign* obteve uma pequena desvantagem é pelo fato de ele ser utilizado para gerar dois arquivos, enquanto o ClustalΩ apenas um. O motivo de um arquivo extra é somente para facilitar a visualização do alinhamento pelo usuário e prover o arquivo de entrada do Modeller simultaneamente.

Pelas facilidades oferecidas pelo algoritmo *salign* e por obter uma diferença de tempo não significativa em relação à outra abordagem proposta (ClustalΩ), este foi escolhido como ferramenta de alinhamento pareado a ser usada na ferramenta de refinamento do MHOLline 2.0.

7.5 Ferramenta de Agrupamento

Como citado anteriormente, em alguns casos é necessário fazer o agrupamento dos resultados para facilitar a análise pelo usuário. Apesar de já existirem ferramentas de agrupamento de proteínas, como o cd-hit [132] e o MaxCluster [133], elas não atendem as necessidades deste estudo pois neste trabalho deseja-se fazer uma comparação não da semelhança estrutural 3D entre os modelos, mas sim da qualidade dos modelos gerados. Para isso, um *script* em *Python3* foi criado e para esta tarefa, optou-se pelo algoritmo *Mean Shift* [134] devido à sua simplicidade de parâmetros de entrada, como por exemplo não requerer abordagens para se descobrir a quantidade de agrupamentos que deve ser gerada, aliado com um resultado de agrupamento semelhante à triagem humana feita por especialistas (observado durante essa pesquisa).

O *Mean Shift* é um método não paramétrico que busca o máximo de uma função de densidade dado um conjunto de dados discretos como entrada [135]. O algoritmo executa o cálculo da seguinte equação:

$$M(x) = \frac{\sum_{x_i \in N(x)} K(x_i - x) x_i}{\sum_{x_i \in N(x)} K(x_i - x)} \tag{7.1}$$

sendo $N(x)$ a vizinhança de x onde $K(x) \neq 0$, $K(x_i - x)$ uma função kernel, esta determina os pesos dos pontos próximos para o cálculo da média. A diferença entre $M(x)$ e x é o desvio da média (*Mean Shift*). Então o algoritmo executa a operação $x \leftarrow M(x)$ até que haja uma convergência de $M(x)$.

O *Mean Shift* é considerado um algoritmo de subida de encosta (*Hill Climbing*), visto que ele se inicia com uma conformação qualquer e tenta otimizá-la até não haver mais convergência dos resultados. Cada desvio é definido por um vetor de desvio da média, este apontando sempre para a direção do maior aumento da densidade. A cada iteração, o kernel é desviado para o centróide do agrupamento. O cálculo da média no algoritmo pode variar dependendo do kernel usado. Caso seja um kernel Gaussiano, a média será ponderada, atribuindo um peso para cada elemento [134].

Nessa dissertação, utilizou-se o Python 3.4 com as seguintes bibliotecas: *scikit-learn* versão 0.18.1 [136], *pandas* versão 0.19.2 [137] e *numpy* versão 1.11.2 [138]. Os dados de entrada são os valores de energia gerados pelo *Modeller* (*Molpdf*, *DOPE-HR* e *nDope*) e os resultados da análise do gráfico de Ramanchandran gerado pelo *Molprobity*, sendo

considerado somente os valores de número de *Outliers* e a quantidade de resíduos alocados em regiões favoráveis. Apesar de não serem utilizados durante o agrupamento os valores das pontuações *GA341* e *DOPE* geradas pelo Modeller também são inclusas nos resultados para dispor ao usuário um conjunto de informações completo.

Uma vez que todos os dados necessários estão disponíveis, o *script* lê o arquivo de entrada e carrega os valores, normaliza cada dado de entrada pelo seu máximo, realiza uma análise de componentes principais (PCA) [139, 140] visando reduzir as dimensões de dados de cinco para dois, restando apenas os dois dados mais significativos. O *MeanShift* é então executado, seus resultados são salvos em um arquivo do tipo `.csv` onde constam cada modelo gerado e o seu grupo designado pelo *MeanShift*.

Para determinar se a ferramenta de agrupamento criada é eficaz, a proteína 5FPY foi modelada com e sem restrições, gerando 20 modelos cada uma. Os dois testes de modelagem foram agrupados separadamente e seus resultados foram dispostos em gráficos de dispersão. Por fim, foram realizadas comparações entre as classificações realizadas pelo *script* e um especialista da área.

8 Resultados e Discussão

8.1 Interface de Submissão

A interface de submissão foi refatorada (Figura 8.1), sendo adicionados os novos *softwares* disponíveis no MHOLline 2.0. O portal passou a dar suporte também à submissão de arquivos através de todos os navegadores *web*.

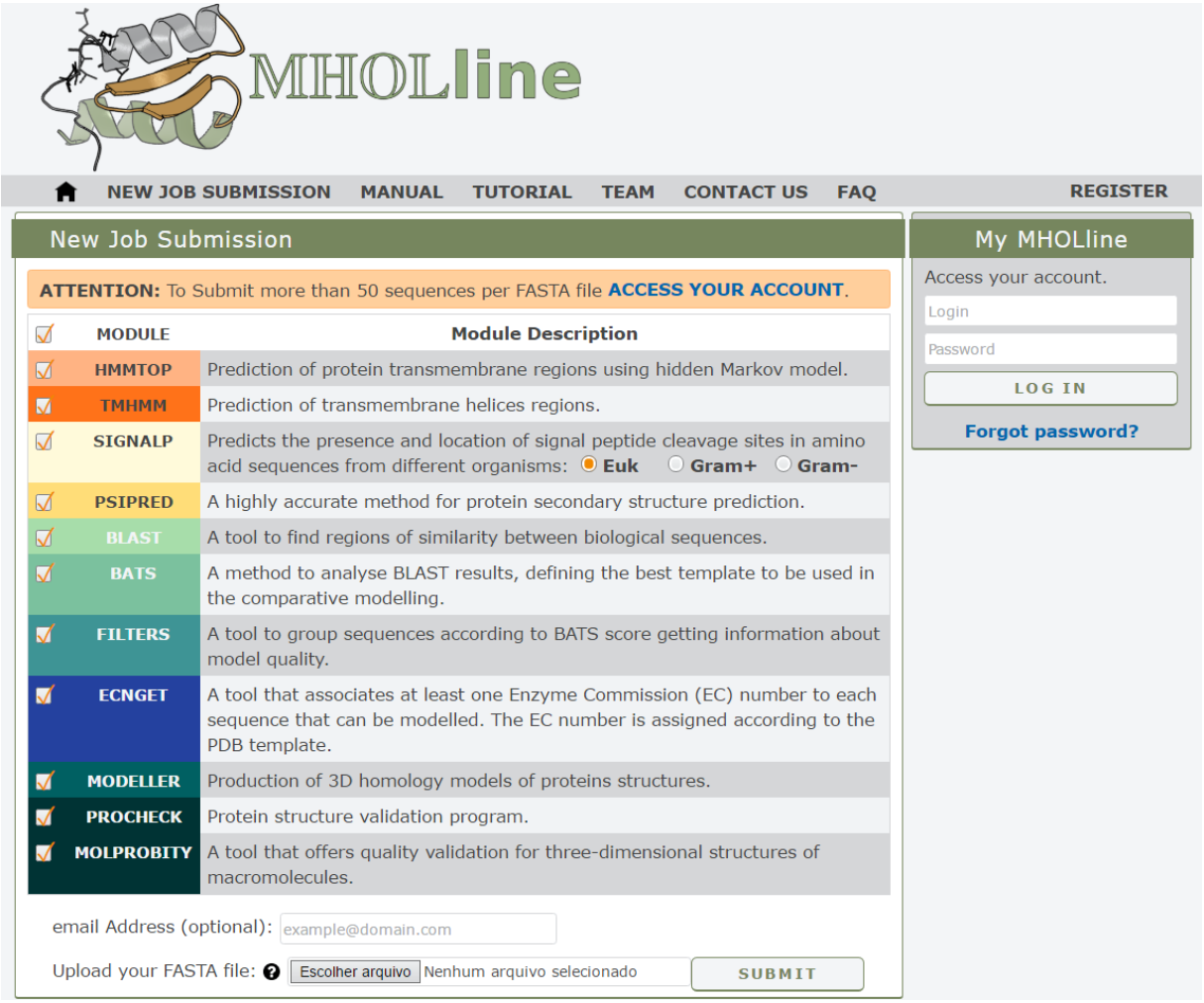


Figura 8.1: Interface de submissao de arquivos para um usuário não logado.

8.2 Interface de Resultados

A interface de resultados (Figura 8.2) do MHOLline foi refatorada de forma a dar suporte aos resultados dos novos *softwares* e permitir ao usuário acessar a área de resultados

em detalhe. Os resultados em detalhe são referentes a cada *job* separado, oferecendo ao usuário detalhes de cada sequência submetida no respectivo *job*.

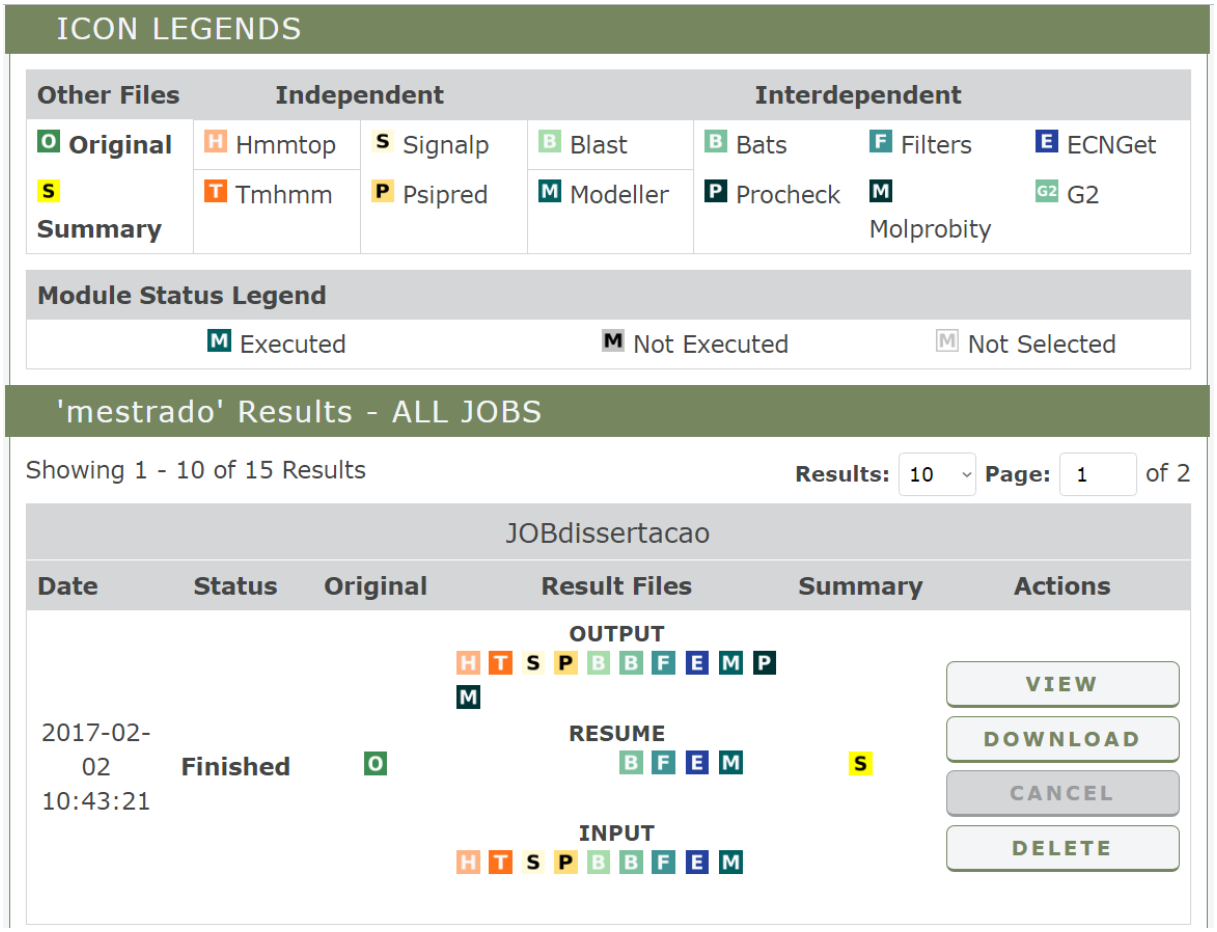


Figura 8.2: Interface de resultados do MHOLline.

Foi criada uma interface de visualização detalhada dos resultados (Figura 8.3) na qual é possível o usuário visualizar informações das proteínas dos grupos (*G0*, *G1*, *G2* ou *G3*) separadamente, e no grupo *G2* é possível selecionar quais grupos do *Filters* (*Very High*, *High*, *Good*, *Medium to Good*, *Midim to Low*, *Low* e *Very Low*) deseja-se visualizar. Para estas proteínas no grupo *G2* é possível realizar de forma individual o refinamento de cada proteína.

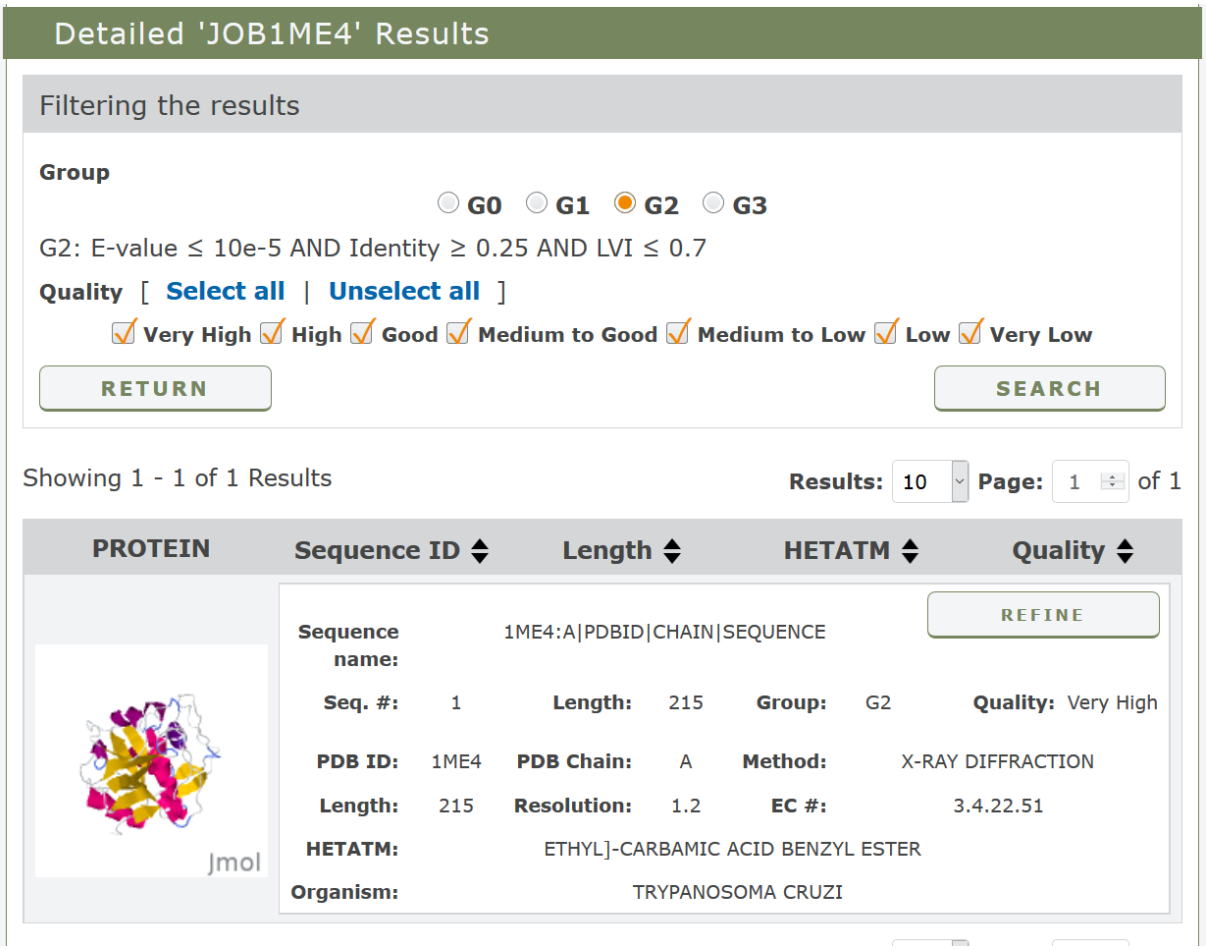


Figura 8.3: Interface de resultados detalhados no MHOLline 2.0. Interface criada para prover mais detalhes de cada proteína e ser o acesso para o refinamento de proteínas.

8.3 Interface de Refinamento

Na área de refinamento, os usuários logados podem analisar os resultados individualmente de cada modelo gerado pelo MHOLline como mostrado na Figura 8.4. Nesta interface única, o usuário pode optar por realizar uma remodelagem da sua proteína de interesse de forma automática ou manual.

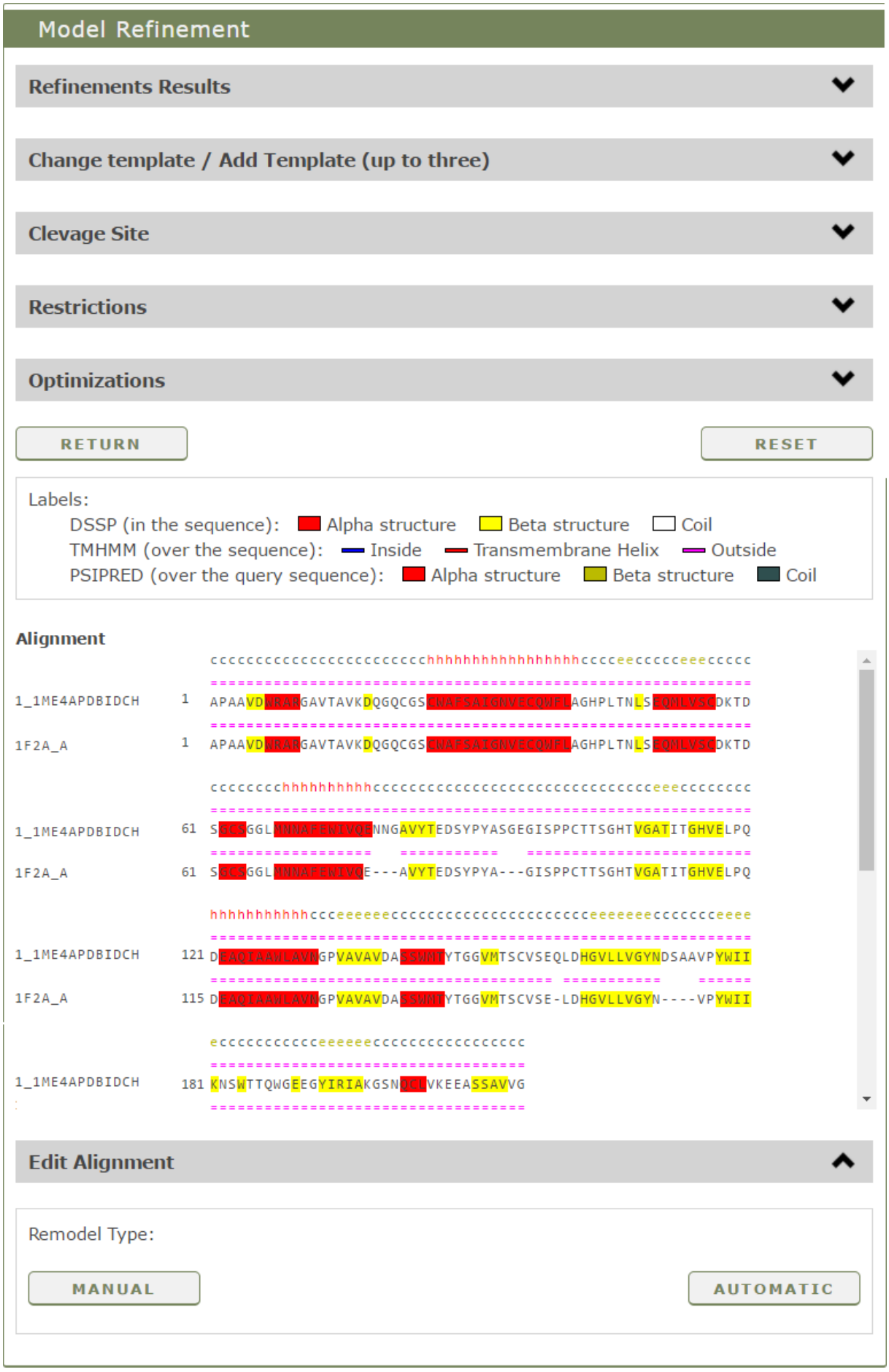


Figura 8.4: Interface de refinamento no MHOLline 2.0.

Uma das possíveis alterações no modelo original é a troca de molde ou a adição de até no máximo três moldes para a geração de um novo modelo através do *software* Modeller [58]. Essa interface pode ser vista na Figura 8.5.

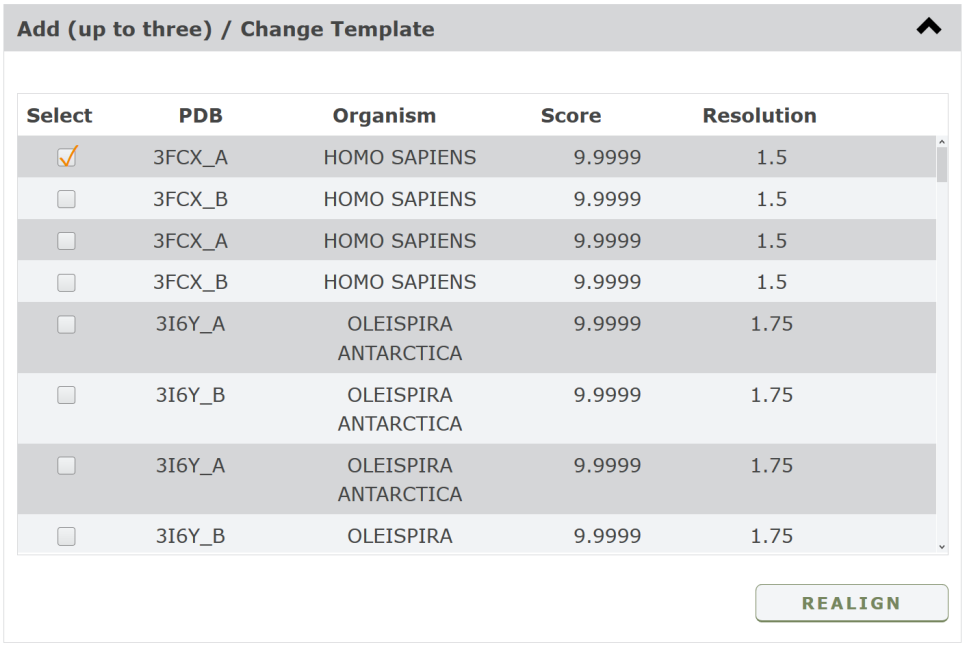


Figura 8.5: Interface para troca ou adição de moldes na sessão de refinamento.

O usuário ainda pode fazer o *download* do alinhamento (formato .aln) para refiná-lo localmente em sua máquina, fazendo o *upload* do arquivo editado, no mesmo formato (.aln), que então será processado pelo servidor para ser remodelado (Figura 8.6).

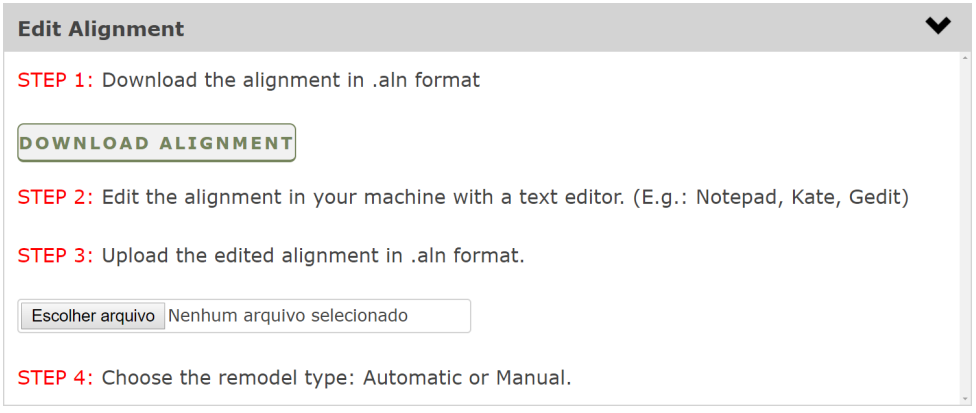


Figura 8.6: Interface com instruções dos passos necessários para se utilizar um alinhamento modificado pelo usuário.

Caso o usuário opte pelo refinamento manual é possível:

- Fazer a clivagem de regiões detectadas pelo SignalP como mostrado na Figura 8.7;

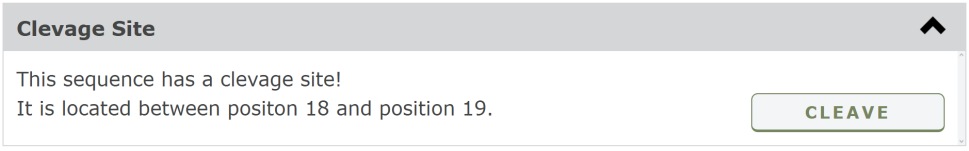


Figura 8.7: Interface para clivagem de peptídeo sinal.

- Adicionar restrições de estruturas secundárias no Modeller, como hélices, fitas, folhas e distância entre átomos, como mostrado na Figura 8.8;

Figura 8.8: Interface para adição de restrições para estruturas secundárias no Modeller.

- Otimizar *loops* com o Modeller como mostrado na Figura 8.9.

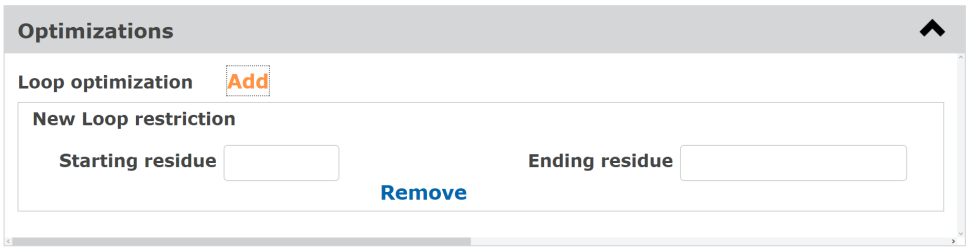
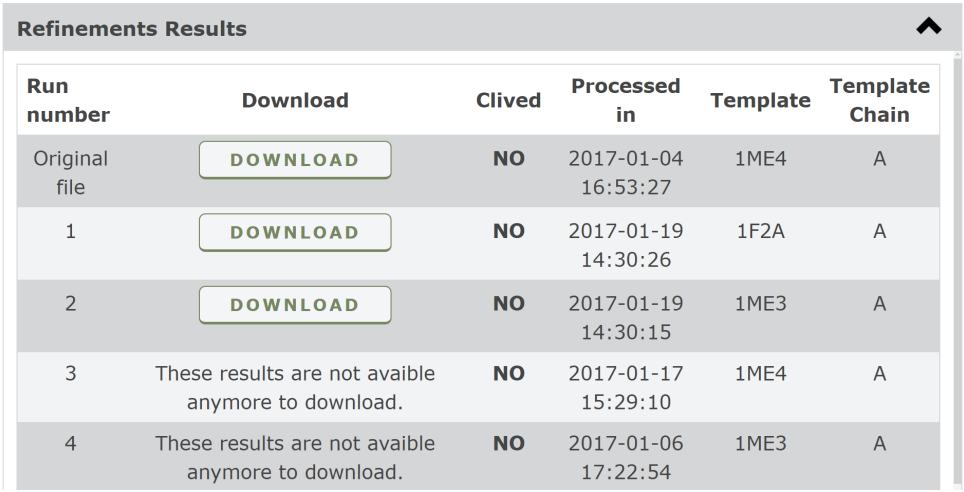


Figura 8.9: Interface para otimização de loops no Modeller.

Caso o usuário opte pelo refinamento automático, não havendo *gaps*, 20 modelos são gerados sem restrição alguma, se houver predições de estrutura secundária em região de *gaps*, estas são aplicadas, caso haja *gaps* porém sem predições de estrutura secundária em toda extensão do alinhamento, a otimização de *loop* é aplicada nesta região.

Após os refinamentos (manual e/ ou automático), o usuário pode acompanhar seus novos resultados como mostrado na Figura 8.10. O usuário tem a sua disposição para *download* o modelo original gerado pelo *workflow* e os dois últimos refinamentos testados.



Refinements Results

Run number	Download	Clived	Processed in	Template	Template Chain
Original file	DOWNLOAD	NO	2017-01-04 16:53:27	1ME4	A
1	DOWNLOAD	NO	2017-01-19 14:30:26	1F2A	A
2	DOWNLOAD	NO	2017-01-19 14:30:15	1ME3	A
3	These results are not avaible anymore to download.	NO	2017-01-17 15:29:10	1ME4	A
4	These results are not avaible anymore to download.	NO	2017-01-06 17:22:54	1ME3	A

Figura 8.10: Interface de escolha de moldes.

8.4 Teste da ferramenta de Refinamento

No intuito de se fazer uma simulação controlada da modelagem de uma proteína com uma cobertura reduzida, modificou-se, o molde da proteína 5FPY, que era 1CU1 para 1HEI, causando assim uma região de *gap* de cerca de 33% na proteína alvo. O teste de refinamento foi submetido aos módulos Manual e Automático, com restrições apresentadas na Tabela 7.1. A qualidade dos resultados obtidos pode ser observada através do alinhamento estrutural entre o modelo original e os novos gerados, apresentado na Figura 8.11.

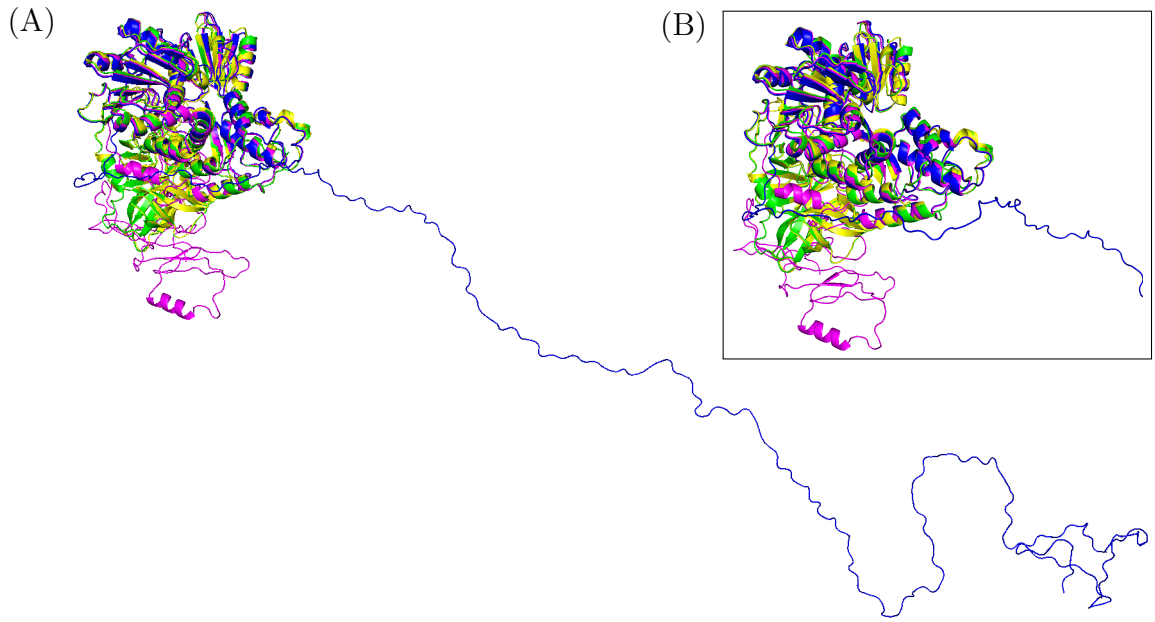


Figura 8.11: Alinhamento 3D da proteína 5FPY, usando a ferramenta de refinamento do MHOLline, com o molde 1CU1. Proteína refinada no modo automático (azul) e modo manual (magenta). O modelo original da 5FPY está representado em amarelo e seu molde 1CU1 em verde. (A) Visualização completa da proteína. (B) Visualização em detalhe da proteína com sua parte enovelada. Alinhamento estrutural obtido via *software* Pymol [124]

Visualmente, nota-se que a proteína gerada através do refinamento automático (Figura 8.11 em cor azul) não gerou estruturas secundárias nem um enovelamento de parte da proteína, no refinamento manual (Figura 8.11 em cor magenta) houve um enovelamento, porém não semelhante ao modelo original gerado no *workflow* principal (Figura 8.11 em cor amarela) nem ao molde 1HEI (Figura 8.11 em cor verde). O melhor desempenho relativo do modo manual deveu-se ao uso de restrições de contato entre os resíduos. Sendo assim, é sugerido que proteínas com uma cobertura pequena sejam refinadas com mais moldes e com o uso de restrições de contanto usando preferencialmente o modo manual.

Testes no modo automático da ferramenta de refinamento do MHOLline 2.0 foram conduzidos para se inferir até qual classificação do BATS a qualidade do modelo pode ser melhorada quantitativamente e qualitativamente. Neste teste, foi escolhida uma proteína do organismo *Trypanossoma cruzi* [141] em cada grupo classificatório da ferramenta *Filters* do MHOLline (*Very High, High, Good, Medium to Good, Medium to Low, Low e Very Low*), a fim de se atestar a qualidade da melhora dos resultados com o refinamento automático. As informações das proteínas desse novo conjunto teste podem ser obtidas na Tabela 8.1.

Tabela 8.1: Informações do conjunto de proteínas usadas nos testes de modelagem para todos os grupos classificatórios do *Filters*.

Modelo			Molde						Refinamento	
Nome	Qualidade	NR	PDB ID	Cadeia	Nome	NR	SCOP	CATH	Região sem cobertura	Restrição
Tc00.1047053506925.104	Very High	90	1CMI [142]	B	protein inhibitor of neuronal nitric oxide synthase	85	$\alpha+\beta$	$\alpha+\beta$	1-5	N
Tc00.1047053506925.319	High	413	1LWD [143]	A	isocitrate dehydrogenase	413	α/β	α/β	1/ 392/ 415-416	N
Tc00.1047053508153.490	Good	124	1FJD [144]	A	peptidyl prolyl cis/trans isomerase (PPIASE)	104	$\alpha+\beta$	$\alpha+\beta$	1-25/ 105-110	Variações de α-hélice 4:7
Tc00.1047053508153.820	Medium to Good	276	1E9G [145]	A	inorganic pyrophosphatase	286	β	α/β	1-29/ 56-59/ 191-194/ 273-279/ 288-318	Variações de fita-β 5:8
Tc00.1047053511667.30	Medium to Low	112	1MS9 [146]	A	trans-sialidase	622	β	β	1-30/ 41-341/ 394-401/ 448-588/ 593-622	N
Tc00.1047053506435.200	Low	428	1VHN [147]	A	putative flavin oxidoreductase	318	α/β	α/β	1-76/ 114-116/ 203-206/ 259-260/ 307-310/ 367-370/ 393-429	Sorteio entre restrições variadas
Tc00.1047053508153.210	Very Low	371	1TA0 [148]	A	carboxy-terminal domain RNA polymerase II polypeptide A small phosphatase 1	197	α/β	α/β	1-115/ 157-167/ 179-183/ 299-322/ 329-333/ 342-371	Sorteio entre restrições variadas

NR: número de resíduos. NC: não contém. [a] Classificação de acordo com o SCOP [122]. [b] Classificação de acordo com o CATH [123].

Cada proteína desse novo conjunto teste foi modelada de acordo com o esquema de classificação do modo automático do refinamento do MHOLline2.0, portanto tendo uma quantidade variada de modelos, dependendo então muito do seu alinhamento. Os resultados dos testes estão dispostos a seguir e maiores detalhes podem ser observados nas Tabelas 10.2, 10.3, 10.4, 10.5, 10.6 e 10.7 do Apêndice dessa dissertação:

- *Very High*: Ao gerar 20 modelos com o refinamento automático obteve-se uma melhora não significativa dos resultados de energia do Modeller e no Molprobity não houve alterações. Uma imagem do modelo que obteve a melhor combinação de resultados alinhada com o seu molde pode ser conferido na Figura 8.12;

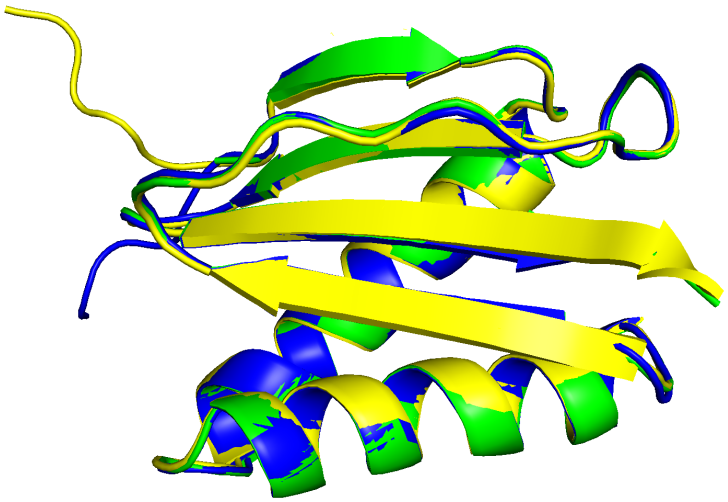


Figura 8.12: Teste de refinamento de proteína classificada como *Very High*. Modelo refinado automaticamente (em azul) alinhado com seu molde, 1CMI [143] cadeia B (em verde), e proteína modelada no *workflow* principal do MHOLline (em amarelo). Alinhamento estrutural obtido via *software* Pymol [124].

- *High*: Ao gerar 20 modelos com o refinamento automático obteve-se uma melhora não significativa na avaliação de energia do Modeller, as estruturas mantiveram-se estáveis e o Molprobity não obteve mudanças significativas. Uma imagem do modelo que obteve a melhor combinação de resultados alinhada com o seu molde pode ser conferido na Figura 8.13;

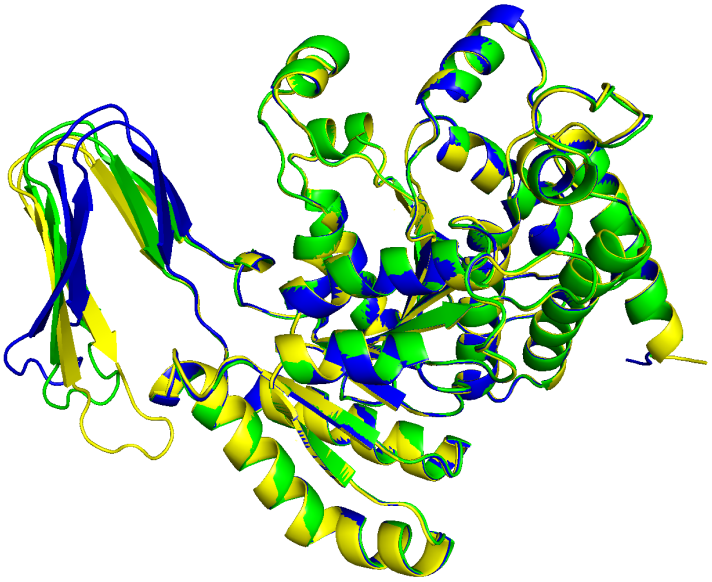


Figura 8.13: Teste de refinamento de proteína classificada como *High*. Modelo refinado automaticamente (em azul) alinhada com seu molde, 1LWD [142] cadeia A, (em verde) e proteína modelada no *workflow* principal do MHOLline (em amarelo). Alinhamento estrutural obtido via *software* Pymol [124].

- *Good*: Ao gerar 6 modelos com o refinamento automático não se obteve uma melhora significativa nos resultados de energia do Modeller, porém houve uma melhora nos resultados do Molprobity, sendo uma redução na quantidade de *Outliers* de 3 para 1, e um aumento na porcentagem da quantidade dos resíduos em regiões favoráveis e permitidas. Uma imagem do modelo que obteve a melhor combinação de resultados alinhada com o seu molde pode ser conferido na Figura 8.14;

Figura 8.14: Teste de refinamento de proteína classificada como *Good*. Modelo refinado automaticamente (em azul) alinhada com seu molde, 1FJD [144] cadeia A, (em verde) e proteína modelada no *workflow* principal do MHOLline (em amarelo) Alinhamento estrutural obtido via *software* Pymol [124].

- *Medium to Good*: Ao se gerar 10 modelos não se obteve melhora significativa nos resultados de energia do Modeller, porém obteve-se uma melhora na estrutura, de acordo com os resultados do Molprobity, sendo a redução, dois resíduos que se encontravam como *outliers* na parte interna da estrutura foram otimizados. Uma imagem do modelo que obteve a melhor combinação de resultados alinhada com o seu molde pode ser conferido na Figura 8.15;

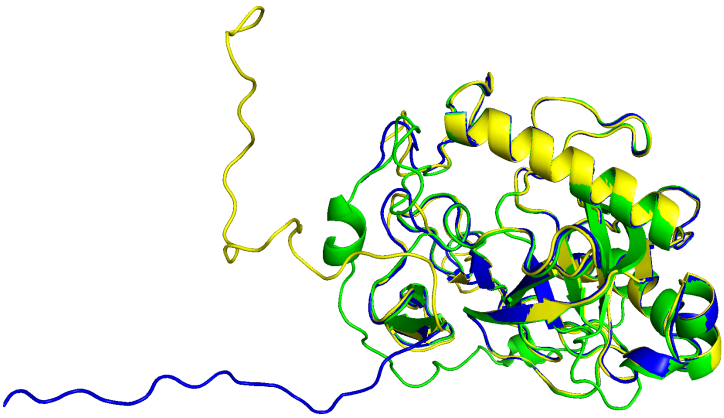


Figura 8.15: Teste de refinamento de proteína classificada como *Medium to Good*. Modelo refinado automaticamente (em azul) alinhada com seu molde, 1E9G [145] cadeia A, (em verde) e proteína modelada no *workflow* principal do MHOLline (em amarelo). Alinhamento estrutural obtido via *software* Pymol [124].

- *Medium to Low*: Ao se gerar 10 modelos não se obteve uma mudança significativa nos resultados de energia do Modeller, porém obteve-se uma redução no número de *Outliers* e um aumento na porcentagem de estruturas em regiões favoráveis. Uma imagem do modelo que obteve a melhor combinação de resultados alinhada com o seu molde pode ser conferido na Figura 8.16;

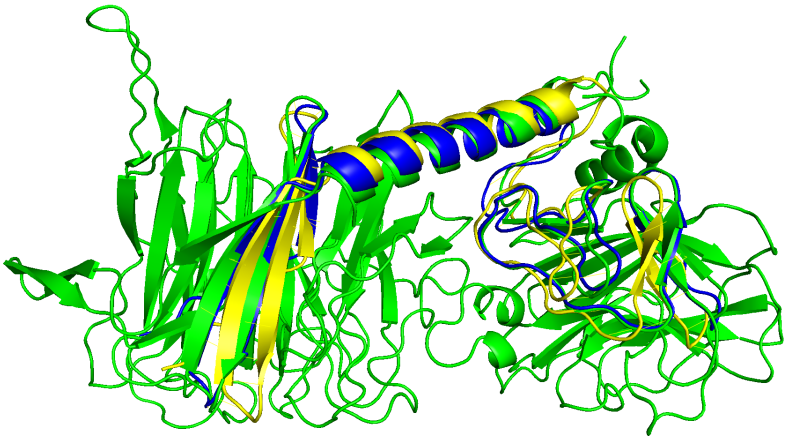


Figura 8.16: Teste de refinamento de proteína classificada como *Medium to Low*. Modelo refinado automaticamente (em azul) alinhada com seu molde, 1MS9 [146] cadeia A, (em verde) e proteína modelada no *workflow* principal do MHOLline (em amarelo). Alinhamento estrutural obtido via *software* Pymol [124].

- *Low*: Ao se gerar 50 modelos obteve-se uma melhora significativa nos resultados de energia do Modeller, porém não se obteve melhoras significativas nos resultados do Molprobity. Uma imagem do modelo que obteve a melhor combinação de resultados alinhada com o seu molde pode ser conferido na Figura 8.17;

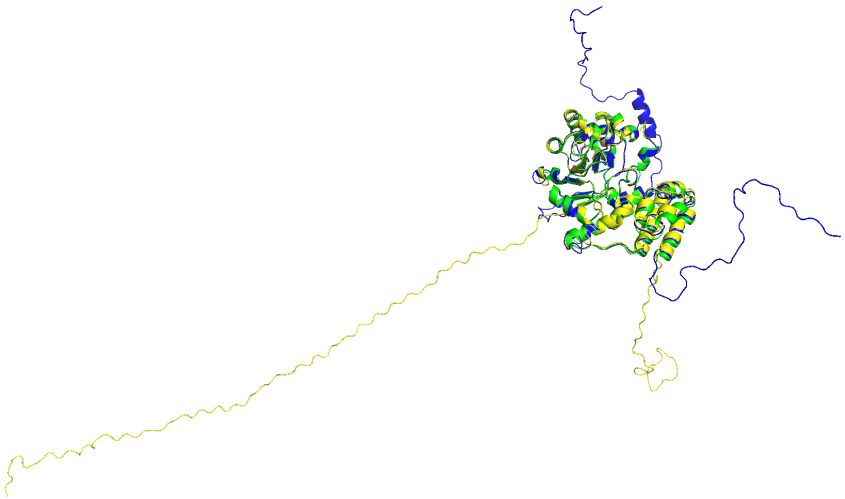


Figura 8.17: Teste de refinamento de proteína classificada como *Low*. Modelo refinado automaticamente (em azul) alinhada com seu molde, 1VHN [147] cadeia A, (em verde) e proteína modelada no *workflow* principal do MHOLline (em amarelo). Alinhamento estrutural obtido via *software* Pymol [124].

- *Very Low*: Ao se gerar 50 modelos não se obteve uma melhora significativa nos resultados de energia do Modeller, porém obteve uma melhora significativa nos

resultados do Molprobity, contudo não houve uma melhora na qualidade da estrutura tridimensional final da proteína. Uma imagem do modelo que obteve a melhor combinação de resultados alinhada com o seu molde pode ser conferido na Figura 8.18.

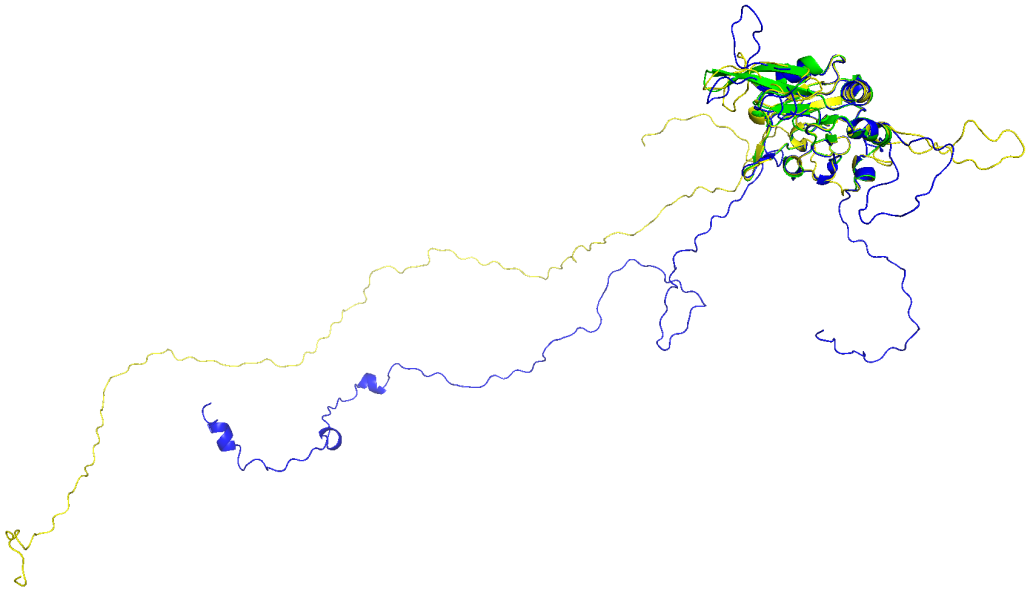


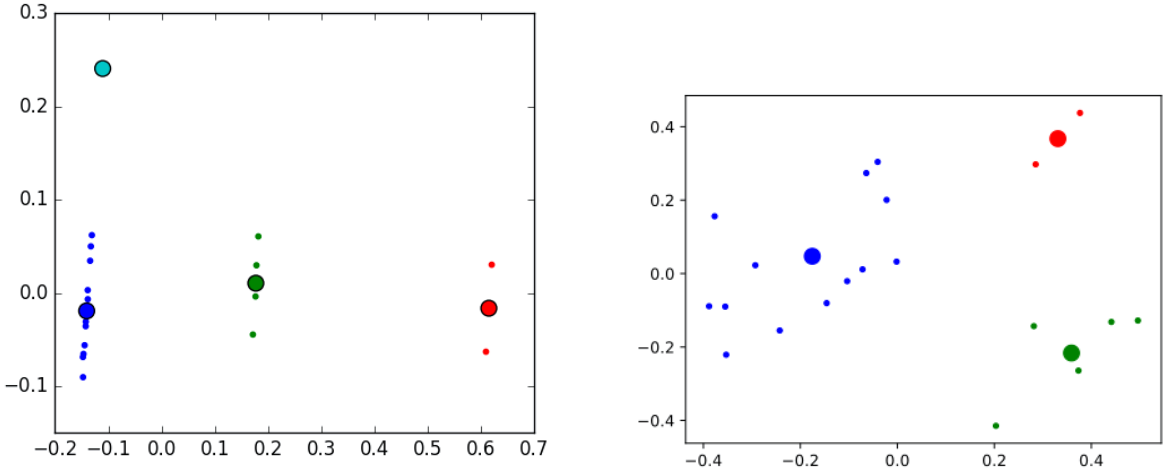
Figura 8.18: Teste de refinamento de proteína classificada como *Very Low*. Modelo refinado automaticamente (em azul) alinhada com seu molde, 1TA0 [148] cadeia A, (em verde) e proteína modelada no *workflow* principal do MHOLline (em amarelo). Alinhamento estrutural obtido via *software* Pymol [124].

Com base na análise dos dados acima, conclui-se que apesar de uma melhora, mesmo que não signficativa em todos os grupos do *Filters*, pôde-se notar visualmente que o que determina a possibilidade do refinamento ser efetivo é a cobertura da sequência pelo(s) molde(s), visto que a proteína do grupo *Medium to Low*, apresentou-se superior às proteínas dos grupos *Medium to Good*, *Low* e *Very Low*.

Através desta observação, sugere-se que caso haja regiões de *gaps* extensas na sequência do modelo, seja feito um alinhamento múltiplo a fim de se reduzir o impacto de um alinhamento com uma cobertura baixa. Acredita-se que o valor mínimo seja de 70% devido ao fato de que as proteínas dos grupos *Low* e *Very Low* possuírem cobertura inferior a 70% e ambas não terem obtido resultados finais significativos no refinamento. Portanto, caso a proteína não tenha sido classificada pelo *Filters* como pelo menos *Medium to Good*, é recomendado que seja feito um alinhamento múltiplo para se tentar aumentar a cobertura final.

8.5 Agrupamento de dados

Os gráficos de dispersão gerados pelos testes executados sem restrições 8.19a e com restrições 8.19b foram inspecionados manualmente por especialista e foi notado que o agrupamento fornecido pelo *MeanShift* se assemelhou ao resultado de separação manual.



(a) Proteína modelada sem restrições.

(b) Proteína modelada com restrições.

Figura 8.19: Gráficos de dispersão da proteína 5FPY gerados com o *script* de agrupamento. Os círculos maiores representam a conformação medóide de cada grupo.

Também notou-se que proteínas com características semelhantes tenderam a se agrupar no mesmo conjunto. Tal afirmação se baseia nos resultados de agrupamento dispostos no arquivo `.csv` e pode ser conferido na tabela de dados de entrada do *script* de agrupamento na Tabela 8.2 e na tabela de grupos resultante da execução do *script* de agrupamento (Tabela 10.1).

Tabela 8.2: Dados de entrada do *script* de agrupamento da proteína 5FPY com restrições.

Modelo	DOPE	DOPEHR	GA341	NDOPE	MOLPDF	Allowed	Outlier
1_5FPY.B99990001	-6.293.348.438	-5.012.426.172	1.0	0.28671	34.996.521	98.3	11
1_5FPY.B99990002	-6.148.621.094	-4.796.979.688	1.0	0.44061	459.655.518	97.4	17
1_5FPY.B99990003	-6.205.260.547	-5.098.928.516	1.0	0.38038	34.262.832	98.2	12
1_5FPY.B99990004	-628.666.875	-5.155.321.875	1.0	0.29381	350.668.555	98.6	9
1_5FPY.B99990005	-6.256.751.172	-5.068.316.797	1.0	0.32562	353.274.146	97.7	15
1_5FPY.B99990006	-6.229.152.344	-5.022.223.438	1.0	0.35497	351.500.244	98.6	9
1_5FPY.B99990007	-6.029.044.141	-4.699.273.047	1.0	0.56776	379.673.706	97.9	14
1_5FPY.B99990008	-6.229.187.891	-5.015.666.016	1.0	0.35493	444.366.797	98.0	13
1_5FPY.B99990009	-6.077.190.234	-4.780.326.562	1.0	0.51657	351.867.358	98.0	13
1_5FPY.B99990010	-6.211.791.406	-4.772.741.406	1.0	0.37343	420.516.748	98.3	11
1_5FPY.B99990011	-6.055.985.156	-4.828.841.797	1.0	0.53912	357.781.665	98.6	9
1_5FPY.B99990012	-6.148.073.047	-4.885.971.875	1.0	0.44119	422.902.002	97.6	16
1_5FPY.B99990013	-6.257.944.922	-4.975.738.672	1.0	0.32435	341.404.053	98.9	7
1_5FPY.B99990014	-6.037.747.266	-4.759.833.594	1.0	0.55851	369.534.521	98.2	12
1_5FPY.B99990015	-6.377.746.484	-5.126.914.844	1.0	0.19696	349.062.476	98.2	12
1_5FPY.B99990016	-6.042.120.703	-4.714.204.297	1.0	0.55386	415.898.291	98.0	13
1_5FPY.B99990017	-6.259.373.828	-4.937.498.828	1.0	0.32283	3.794.302	97.7	15
1_5FPY.B99990018	-6.210.810.938	-4.974.873.828	1.0	0.37448	340.938.379	98.3	11
1_5FPY.B99990019	-6.304.007.812	-5.147.684.375	1.0	0.27537	378.002.637	98.8	8
1_5FPY.B99990020	-6.182.810.547	-5.061.916.406	1.0	0.40425	356.748.975	98.0	13

9 Conclusões e Perspectivas Futuras

Dos desenvolvimentos e implementações de funcionalidades no MHOLline 2.0 podemos concluir que:

- Com as adições dos *softwares SignalP*, *PSIPRED*, *TMHMM*, *DSSP*, *Molprobity*, *Jmol e Jsmol* a capacidade do MHOLline de fornecer informações relevantes para o estudo de proteínas aumentou, completando as análises dos pesquisadores e atendendo parte das requisições dos usuários do MHOLline 1.0, vindas do *feedback*.

- A interface de usuário criada para o refinamento e remodelagem das proteínas se mostrou simples, com funcionalidades que ajudam o pesquisador a fazer a análise da proteína reduzindo o trabalho do pesquisador sem comprometer a qualidade do modelo.

- Os testes de qualidade efetuados com a biblioteca paralela do Modeller se mostraram eficazes com uma modificação não significativa da qualidade dos modelos gerados e reduzindo o tempo de geração dos Modelos em cerca de 50%.

- A partir dos testes realizados na ferramenta de agrupamento desenvolvida neste trabalho, notou-se a alta semelhança entre os agrupamentos manuais e os realizados pela ferramenta, além de alocar proteínas com resultados semelhantes no mesmo grupo, provando que a ferramenta de agrupamento criada neste estudo atingiu o seu objetivo, que era separar os modelos gerados em grupos com membros semelhantes, facilitando assim a análise do usuário sem a perda de qualidade.

- Baseado nos resultados de qualidade dos modelos refinados através da ferramenta de refinamento do MHOLline, conclui-se que apesar de não ser eficaz em proteínas com baixa cobertura, se mostrou eficaz em reduzir os *Outliers*, aumentar os valores de resíduos em regiões favoráveis e melhorar as pontuações de energia do Modeller.

- Com todas essas modificações, conclui-se que toda a interface de usuário avançado provou ser útil, reduzindo o trabalho dos pesquisadores e gerando modelos de qualidade.

Como perspectivas futuras podemos citar:

- Adicionar mais ferramentas ao escopo do refinamento do MHOLline 2.0, como as de predição de estrutura secundária (*e.g.*, Jpred [149], RaptorX Property Prediction [150], Predict Protein [151]) no intuito de gerar análises de consenso e poder oferecer ao usuário uma predição mais acurada.

- Também é sugerida a análise mais a fundo para a validação do algoritmo de agrupamento, visto que atualmente a escolha deste deu-se principalmente pela inspeção visual, havendo a necessidade de uma validação através de um estudo mais aprofundado.

- Ainda pode-se sugerir a filtragem de combinações provenientes do refinamento automático a fim de se evitar algumas combinações que não gerem estruturas secundárias biologicamente estáveis, como hélices com menos do que três resíduos.

- Sugere-se fortemente a adição de um preditor de contato para se obter informações de distância 3D entre resíduos sequencialmente afastados.

- Pode-se adicionar também outros métodos de modelagem de proteínas para os grupos *G0*, *G1* e *G3*, aumentando a capacidade do MHOLline de modelar mais proteínas.

- Devido à quantidade de combinações possíveis no refinamento automático, mesmo quando algumas combinações são excluídas através de um filtro, sugere-se a implantação de uma heurística (*e.g.*, algoritmo genético) para tratar de uma maneira mais seletiva os resultados, retornando ao usuário resultados mais signficativos.

- Sugere-se implementar uma rotina de reuso de dados para que o usuário possa fazer o *download* e armazenamento das informações de execução de um *job* para, caso deseje futuramente, poder reutilizar os dados desse *job* na ferramenta de refinamento, fazendo o *upload* desses dados.

- Por fim, recomenda-se mais testes relacionados à área de refinamento do MHOLline 2.0 para assegurar a robustez dos métodos à exceções.

REFERÊNCIAS

[1] HUETRA M., HASELTINE F., LIU Y., DOWNING G., and SETO B. Nih working definition of bioinformatics and computational biology. Biomedical Information Science and Technology Initiative Consortium, Julho 2010.

[2] KING J.L. and JUKES T.H. Non-darwinian evolution. *Science*, 164(3881):788–798, 1969.

[3] EPSTEIN C.J. Non-randomness of ammo-acid changes in the evolution of homologous proteins. *Nature*, 215:355–359, 1967.

[4] NEEDLEMAN S.B. and WUNSCH C.D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48:443–453, 1970.

[5] CHOU P.Y. and FASMAN G.D. Prediction of protein conformation. *Biochemistry*, 13(2):222?245, 1974.

[6] OUZONIS C.A. and VALENCIA A. Early bioinformatics: the birth of a discipline? a personal view. *Bioninformatics*, 19(17):2176–2190, 2003.

[7] WARSHEL A. and KARPLUS M. Calculation of ground and excited state potential surfaces of conjugated molecules. i. formulation and parametrization. *Molecular Physics*, 94(16):5612?5625, 1972.

[8] WARSHELL A. and M. LEVITT M. Theoretical studies of enzymic reactions: Dielectric, electrostatic and steric stabilization of the carbonium ion in the reaction of lysozyme. *J. Mol. Biol.*, 103:227–249, 1976.

[9] LEVITT M. A simplified representation of protein conformations for rapid simulation of protein folding. *J. Mol. Biol.*, 104:59?107, 1976.

[10] Thomas LENGAUER T. and RAREY M. Computational methods for biomolecular docking. *Current Opinion in Structural Biology*, 6(3):402–406, 1996.

[11] FRIESNER R.A, MURPHY R.B, REPASKY M.P., FRYE L.L, GREENWOOD J.R., HALGREN T.A., SANCHAGRIN P.C., and MAINZ D.T. Extra precision

glide:? docking and scoring incorporating a model of hydrophobic enclosure for protein?ligand complexes. *J. Med. Chem*, 49(21):6177–6196, 2006.

[12] VERDONK M.L., COLE J.C, HARSHORN M.J., MURRAY C.W., and TAYLOR R.D. Improved protein?ligand docking using gold. *Proteins: Structure, Function, and Bioinformatics*, 52(4):609–623, 2003.

[13] ZIMMERMANN O. and HANSMANN U.H.E. Support vector machines for prediction of dihedral angle regions. *Bioinformatics*, 22(24):3009, 2006.

[14] KOLA I. and LANDIS J. Can the pharmaceutical industry reduce attrition rates? *Nature Reviews*, 3:711–715, 2004.

[15] STOVER C.K., PHAM X.Q., ERWIN A.L., MIZOGUCHI S.D., WARRENER P., HICKEY M.J., BRINKMAN F.S.L., HUFANGLE W.O., KOWALIK D.J., LAGROU M., GARBER R.L., GOLTRY L., TOLENTINO E., WESTBROCK-WADMAN S., YUAN Y., BRODY L.L., COULTER S.N., FOLGER K.R., KAS A., LARBIG K., LIM R., SMITH K., SPENCER D., WONG G.K.-S., WU Z., PAULSEND I.T., REIZER J. SAIER M.H. HANCOCK R.E.W., LORY, and OLSON M.V. Complete genome sequence of pseudomonas aeruginosa pao1, an opportunistic pathogen. *Nature*, 406:959–964, 2000.

[16] International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, 2001.

[17] TAYLOR I.J., DEELMAN E., GANNON D.B., and SHIELDS M. *Workflows for e-Science: Scientific Workflows for Grids.*

[18] YU J. and BUYYA R. A taxonomy of scientific workflow systems for grid computing. *SIGMOD Rec.*, 34(4):44–49, 2005.

[19] GEORGAKOPOULOS D., HORNICK M., and SHETH A. An overview of work?ow management: From process modeling to work?ow automation infrastructure. *Distributed and Parallel Databases*, 3:119–153, 1995.

[20] BRAGA R. and GANNON D. *Workflows for e-Science - Scientific versus Buisness Workflows.*

[21] DEELMAN E., GANNONB D., SHIELDSC M., and TAYLOR I. Workflows and e-science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, 25:528–540, 2009.

[22] BONIFÁCIO A.L. Análise de ferramentas computadorizadas para suporte à modelagem computacional - estudo de caso no domínio de dinâmica dos corpos deformáveis, 2008.

[23] WOLSTENCROFT K., HAINES R., FELLOWS D., WILLIAMS A., WITHERS D., OWEN S., SOILAND-REYES S., DUNLOP I., NENADIC A, FISHER P., BHAGAT J., BELHAJJAME K., BACALL F., HARDISTY A., HIDALGA A.N. de la H., VARGAS M.P.B., SUFI S., and GOBLE C. The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. *Nucleic Acids Research*, 41(W1):W557–W561, 2013.

[24] B. LUDÄSCHER B., ALTINTAS I., BERKLEYL C., HIGGINS D., JAEGER-FRANK E., JONES M., LEE E., TAO J., and ZHAO Y. Scientific workflow management and the kepler system. *Concurrency and Computation: Practice and Experience*, 18(10):1039–1065, 2006.

[25] LEITE L.S. Sistema de gerência de *workflow* científico aplicado ao imageamento sísmico, 2015.

[26] JACOB J.C., KATZ D.S., BERRIMAN G.B., GOOD J., LAITY A.C., DEELMAN E., KESSELMAN C., SINGH G., SU M., PRINCE T.A., and WILLIAMS R. Montage: a grid portal and software toolkit for science-grade astronomical image mosaicking. *Int. J. Computational Science and Engineering*, 4(2):1–16, 2009.

[27] da SILVA F.A.B and SENGER H. Improving scalability of bag-of-tasks applications running on master?slave platforms. *Parallel Computing*, 35:57–71, 2009.

[28] STEVENS R., MCENTIRE R., GOBLE C., GREENWOOD M., ZHAO J., WIPAT A., and LI P. mygrid and the drug discovery process. *DDT: BIOSILICO*, 2(4):140–148, 2004.

[29] de PARIS R. Desenvolvimento de *Workflow* científico para bioinformática., 2008.

[30] Sequence ids: Accessions and gi numbers. `https://www.ncbi.nlm.nih.gov/genbank/sequenceids/`. Último acesso em: 2017-05-17.

[31] SÁNCHEZ R., PIEPER U., MELO F., ESWAR N., MARTÍ-RENOM M.A., MADHUSUDHAN M.S., MIRKOVIć N., and ŠALI A. Protein structure modeling for structural genomics. *Nature*, 7:986–990, 2000.

[32] LEE D., REDFERN O., and ORENGO C. Predicting protein function from sequence and structure. *Nature Reviews*, 8:995–1005, 2007.

[33] MARSDEN R.L., MAIBAUM D.L.M., YEATS C., and ORENGOE C.A. Comprehensive genome analasys of 204 genomes provides structural genomics with new insights unto protein family space. *Nucleic Acids Research*, 34(3):1066–1080, 2006.

[34] MULLARD A. New drugs cost us$2.6 billion to develop. *Nature Reviews Drug Discovery*, 13:877, 2014.

[35] FELTES B.C, de MAGALHAES C. S., STAATS C.C, JUNQUEIRA D.M., BONATTO D., YATES E.A., CUSTODIO F.L., da SILVA E.R., MALUF F.V., OLIVA G., ROCHA G.K., de MORAIS G.L., NADER H.B, VERLI H., GUEDES I.A., TERSARIOL I.L.S., MUNIZ J.R.C., POLONI J. de F., DARDENNE L.E., LIMA L.M.T.R., LIMA M.A., ALMEIDA M. da S., CAPRILES P.V.S.Z, TREVIZANI R., GUIDO R.V.C., BRAUN R.L., MARGIS R., and CORDEIRO. *Bioinformática: da Biologia à flexibilidade Molecular*. Sociedade Brasileira de Bioquímica e Biologia Molecular - SBBq, São Paulo, Brasil, 1st edition, 2014.

[36] BERMAN H.M., WESTBROOK J., FENG Z., GILLILAND G., BHAT T.N., WEISSIG H., SHINDYALOV I.N., and BOURNE P.E. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.

[37] PARKER M.W. Protein structure from x-ray diffraction. *Journal of Biological Physics*, 29:341–362, 2003.

[38] MCPHERSON A. *Crystallization of Biological Macromolecules*. Cold Spring Harbor Laboratory Press, New York, USA, 1st edition, 1999.

[39] KRAUSS I.R., MERLINO A., VERGARA A., and SICA F. An overview of biological macromolecule crystallization. *J. Mol. Sci.*, 14(6):11643–11691, 2013.

[40] MIZRACHI I. K. and KEITH J.M. *Bioinformatics: Data, Sequence Analysis and Evolution*. Humana Press, Brisbane Queensland Australia, 1st edition, 2008.

[41] MIAO J., CHARALAMBOUS P., KIRZ J., and SAYRE D. Extending the methodology of x-ray crystallography to allow imaging of micrometre-sized non-crystalline specimens. *Nature*, 292:342–344, 1999.

[42] TEIXEIRA J., CERETTI M., MIKULA P., VRÁNA M., LUKÁŠ P., YOUTSOS A.G., VENTER A., SCHINDER R.P., POESTE T., FREYDANK H., HOFMANN M., GYULA T., and GNÄUPEL-HEROLD T. *Measurement of residual stress in materials using neutrons*. International Atomic Energy Agency. IAEA-TECDOC-1457.

[43] CARPENTER J.M. *Neutron Production, Moderation, and Characterization of Sources*. Argonne National Laboratory. IAEA-TECDOC-1457.

[44] PICCOLI P.M.B., KOETZLE T.F., and SCHULTZ A.J. Single crystal neutron diffraction for the inorganic chemis - a pracrical guide. *A Journal of Critical Discussion of the Current Literature*, 28:3–38, 2007.

[45] Nmr - resources - protocols — campus chemical instrument center. `http://www.ccic.ohio-state.edu/nmr-protocols`. Último acesso em: 2017-02-03.

[46] BALCI M. *Basic 1H- and 13C-NMR Spectroscopy*. Elsevier Science, Amsterdam, Netherland, 1st edition, 2005.

[47] BERINJASKII M., TANG P., LIANG J., CRUZ J.A., ZHOU J., ZHOU Y., BASSETT E., MACDONELL C., LU P., LIN G., and WISHART D.S. Genmr: a web server for rapid nmr-based protein structure determination. *Nucleic Acids Res.*, 37:Web Server issue, 2009.

[48] GÜNTERT P. Automated nmr structure calculation with cyana. *Methods Mol Biol.*, 278:375–378, 2004.

[49] SCHWIETERS C.D., KUSEWSKI J.J., and CLORE M. Using xplor-nih for nmr molecular structure determination. *Progress in Nuclear Magnetic Resonance Spectroscopy*, 48(1):47–62, 2006.

[50] CHIZHIK V.I., CHERNYSHEV Y.S., DONETS A.V., FROLOV V.V., KMOLKIN A.V., and SHELYAPINA M.G. *Magnetic Resonance and Its Applications*. Springer, New York, USA, 1st edition, 2014.

[51] CALLAWAY E. The revolution will not be crystallized. *Nature*, 525:172–174, 2015.

[52] SANGER F., NICKLEN S., and COULSON A.R. Dna sequencing with chain-terminating inhibitors. *National Acad Sciences*, 74(12):5463–5467, 1977.

[53] HEATHER J.M. and CHAIN B. The sequence of sequencers: The history of sequencing dna. *Genomics*, 107(1):1–8, 2015.

[54] Dna sequencing costs: Data - national human genome research institute (nhgri). `https://www.genome.gov/sequencingcostsdata/`. Último acesso em: 2017-02-02.

[55] GOLDSMITH-FISCHMAN S. and HONIG G. Structural genomics: Computational methods for structure analysis. *Protein Science*, 12:1813–1821, 2003.

[56] USSERY D. W. Bioinformatics 2000. *Genome Biology*, 1(3):reports4014.1–4014.2, 2000.

[57] CHOTIA C. and LESK A.M. The relation between the divergence of sequence and structure in proteins. *The EMBO Journal*, 5(4):823–826, 1987.

[58] MARTÍ-RENOM M.A., STUART A.C., FISER A., SÁNCHEZ R., MELO F., and ŠALI A. Comparative protein structure modeling of genes and genomes. *Annu. Rev. Biophys. Biomol. Struct.*, 29:291–325, 2000.

[59] VELANKAR S., BEST C., BEUTH B., BOUTSELAKIS C.H., SOUSA DA SILVA A.W. COBLEY N., DIMITROPOULOS D., GOLOVIN A., HIRSHBERG M., JOHN M., KRISSINEL E.B., NEWMAN R., OLDFIELD T., PAJON A., PENKETT C.J., PINEDA-CASTILLO J., SAHNI G., SEN S., SLOWLEY R., SUAREZ-URUENA A., SWAMINATHAN J., van GINKEL G., VRANKEN

W.F., HENRICK K., and KLEYWEGT G.J. PDBe: Protein data bank in europe. *Nucleic Acids Research*, 38:308–317, 2010.

[60] RSCB PDB - content growth report. `http://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=total&seqid=100`. Último acesso em: 2017-01-30.

[61] DOLAN M.A., NOAH J. W., and HUR D. *Homology Modeling: Methods and Protocols - Comparison of Common Homology Modeling Algorithms: Application of User-Defined Alignments*. Molsoft L.L.C., San Diego, CA, USA, 1st edition, 2012.

[62] JACOBSON M.P., PINCUS D.L., RAPP C.S., DAY T.J.F., HONIG B., SHAW D.E., and FRIESNER R.A. A hierarchical approach to all-atom protein loop prediction. *Proteins: Structure, Function and Bioinformatics*, 55:351–367, 2004.

[63] Chemical Computing Group ULC. Molecular operating environment (moe). 2017.

[64] BIASINI M., BIENERT S., WATERHOUSE A., ARNOLD K., STUDER F., SCHMIDT T., KIEFER F., CASSARINO T.G., BERTONI M., BORDOLI L., and SCHWEDE T. Swiss-model: modelling protein tertiary and quaternary structure using evolutionary information. *Nucleic Acids Research*, 42(W1):W252–W258, 2014.

[65] ŠALIALI A., WEBB B., MADHUSUDHAN M.S., SHEN M., DONG H., MARTI-RENOM M.A., ESWAR N., ALBER F., TOPF M., OLIVA B., FISER A., SÁNCHEZ R., YERKOVICH B., BADRETDINOV A., MELO F., OVERINGTON J.P, and FEYFANT E. *MODELLER A Program for Protein Structure Modeling*. Andrej Šali Lab.

[66] DORN M., SILVA M.B., BURIOL L.S., and LAMB L.C. Three-dimensional protein structure prediction: Methods and computational strategies. *Computational Biology and Chemistry*, 53:251–276, 2014.

[67] SMITH T. The art of matchmaking: sequence alignment methods and their structural implications. *Nature*, 7(1):R7, 1999.

[68] SMITH T., Lo CONTE L., BIENKOWSKA J., GAITATZES C., ROGERS R.J., and LATHROP R. Current limitations to protein threading approaches. *J. Comput. Biol*, 4(3):217, 1997.

[69] XU Y., LIU Z., CAI L., and XU D. *Protein Structure Prediction by Protein Threading*. Springer, Springer-Verlag New York, 1st edition, 2007.

[70] TRAMONTANO A. and LESK A.M. *Protein Structure Prediction: Concepts and Applications*. John Wiley and Sons, Weinheim, Germany, 1st edition, 2006.

[71] ANFINSEN C.B. Principles that govern the folding of protein chains. *Science*, 181(4096):223–230, 1973.

[72] CORNELL W.D., CIEPLAK P., BAYLY C.I., GOULD I.R., MERZ K.M. Jr., FERGUSON D.M., SPELLMEYER D.C., FOX T., CALDWELL J.W., and KOLLMAN P.A. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *J. Am. Chem. SOC*, 117:5179–5197, 1995.

[73] CHRISTEN M., HÜNENBERGER P.H., BAKOWIES D., BARON R., BÜRGI R. GEERKE D.P., HEINZ T.N., KASTENHOLZ M.A., KRÄUTLER V., OOSTENBRINK C., PETER C. andTRZENIAK D., and van GUNSTEREN W.F. The gromos software for biomolecular simulation: Gromos05. *J Comput Chem*, 26(16):1719–1751, 2005.

[74] JORGENSEN W.L., MAXWELL D.S., and TIRADO-RIVES J. Development and testing of the opls all-atom force field on conformational energetics and properties of organic liquids. *J. Am. Chem. Soc.*, 118(45):11225–11236, 1996.

[75] CHIVIAN D., ROBERTSON T., BONNEAU R., and BAKER D. *AB INITIO METHODS*, volume 44. John Wiley and Sons, Inc, Hoboken, NewJersey, 2005.

[76] OSGUTHORPE D.J. Ab initio protein folding. *Current Opinion in Structural Biology*, 10:146?152, 2000.

[77] JORGENSEN E.L. and TIRADO-RIVES J. Potential energy functions for atomic-level simulations of water and organic and biomolecular systems. 102(19):6665–6670, 2005.

[78] LI Z., YANG Y., ZHAN J., DAI L., and ZHOU Y. Energy functions in de novo protein design: Current challenges and future prospects. *Annu Rev Biophys.*, 42:315–335, 2013.

[79] LAZARIDIS T. and KARPLUS M. Effective energy functions for protein structure prediction. *Curr Opin Struct Biol.*, 10(2):139–145, 2000.

[80] SIMONS K.T., KOOPERBERG C., HUANG E., and BAKER D. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and bayesian scoring functions. *J. Mol. Biol.*, 268:209–225, 1997.

[81] van GUNSTEREN W.E and BERENDSEN H.J.C. Computer simulation of molecular dynamics: Methodology, applications, and perspectives in chemistry. *Angew. Chem. Int. Ed. Engl.*, 5(4):992–1023, 1990.

[82] TREVIZANI R., CUSTÓDIO F.L., dos SANTOS K.B., and DARDENNE L.E. Critical features of fragment libraries for protein structure prediction. *PLOS ONE*, 12(1):e0170131, 2017.

[83] CHEN V.B., ARENDALL W.B., HEADD J.J., KEEDY D.A., IMMORMINO R.M., KAPRAL G.J., MURRAY L.W., RICHARDSON J.S., and RICHARDSON D.C. Molprobity: all-atom structure validation for macromolecular crystallograph. *Acta Cryst*, 66:12–21, 2010.

[84] LASKOWSKI R.A., MACARTHUR M.W., MOSS D.S., and THORNTON J.M. Procheck: a program to check the stereochemicai quality of protein structures. *J. App. Cryst.*, 26:283–291, 1993.

[85] LASKOWSKI R.A., ANTOON J., RULLMANN C., MACARTHUR M.W., KAPTEIN R., and THORTON J.M. Aqua and procheck-nmr: Programs for checking the quality of protein structures solved by nmr. *Journal of Biomolecular NMR*, 8:477–486, 1996.

[86] RÖSSLE S.C.S. *Desenvolvimento de um sistema computacional para modelagem comparativa em genômica estrutural: Análise de sequências do genoma da Gluconabacter diazotrophicus.* PhD thesis, Universidade Federal do Rio de Janeiro, Instituto de Biofísica Carlos Chagas Filho - Brasil, 2004.

[87] GOLIATT P.V.Z.C. Técnicas de bioinformática e modelagem computacional aplicadas ao estudo do genoma de *Trypanosoma cruzi* e de enzimas consideradas de interesse no tratamento da doença de chagas: Estudo particular das cruzipaínas 1 e 2. Master's thesis, Laboratório Nacional de Computação Científica, Brasil, 2007.

[88] CAPRILES P.V.S.Z., GUIMARÃES A.C.R., OTTO T.D., MIRANDA A.B., DARDENNE L.E., and DEGRAVE W.M. Structural modelling and comparative analysis of homologous, analogous and specific proteins from *Trypanosoma cruzi* versus *Homo sapiens*: putative drug targets for Chagas disease treatment. *BMC Genomics*, 11:610, 2010.

[89] Flowchart maker & online diagram software. `https://www.draw.io`. Último acesso em: 2017-03-03.

[90] REIS V.C.C., GOLIATT P.V.Z.C., and DARDENNE L.E. The evolution of a scientific workflow to problems in bioinformatics and structural biology. Apresentação de pôster em congresso: ISCB-LA X-Meeting SolBio - Belo Horizonte, Minas Gerais, Brasil, October 2014.

[91] Welcome to mholline ;-). `http://www.mholline.lncc.br`. Último acesso em: 2017-02-11.

[92] TUSNÁDY G.E. and SIMON I. Principles governing amino acid composition of integral membrane proteins: Application to topology prediction. *J. Mol. Biol.*, 283:489–506, 1998.

[93] TUSNÁDY G.E. and SIMON I. The hmmtop transmembrane topology prediction server. *Bioninformatics Applications Note*, 17(9):849–850, 2001.

[94] CAMACHO C., COULOURIS G., AVAGYAN V., MA N., PAPADOPOULOS J., BEALER K., and MADDEN L.T. Blast+: architecture and applications. *BMC Bioinformatics*, 10:421, 2009.

[95] PEITSCH M.C. About the use of protein models. *Bioinformatics*, 18(7):934–938, 2002.

[96] MACKERELL A.D.Jr. BASHFORD D., BELLOT M., DUNBRACK R.L.Jr., EVANSECK J.D., FIELD M.J., FISCHER S., GAO J., GUI H., HA S., JOSEPH-MCCARTHY D., KUCHNIR L., KUCZERA K., LAU F.T.K., MATTOS C., MICHNICK S., NGO T., NGUYEN D.T. annd PRODHOM B., REIHER W.E.III, ROUX B., SCHLENKIRCH M., SMITH J.C., STOTE R., STRAUB J., WATANABE M., WIÓRKIEWICZ-KUCZERA J., YIN D., and KARPLUS M. All-atom empirical potential for molecular modeling and dynamics studies of proteins. *J. Phys. Chem. B*, 102:3586–3616, 1998.

[97] BRAUNT W. amd GÕ N. Calculation of protein conformations by proton-proton distance constraints a new efficient algorithm. *J. Mol. Biol.*, 186:611–626, 1985.

[98] SALI A. and BLUNDELL T.L. Comparative protein modelling by satisfaction of spatial restraints. *J Mol Biol*, 234:779?8153, 1993.

[99] International Union of Pure and Applied Chemistry. Iupac-iub commission on biochemical nomenclature. *Archives of biochemisrty and biophysics*, 145:405–421, 1971.

[100] KABSCH W. and SANDER C. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22:2577–2637, 1983.

[101] HUANG L., BRINEN L.S., and ELLMAN J.A. Crystal structures of reversible ketone-based inhibitors of the cysteine protease cruzain. *Bioorg.Med.Chem*, 11:21–29, 2003. PDB ID: 1ME4.

[102] ALBERTS B., JOHNSON A., LEWIS J., RAFF M., ROBERTS K., and WALTER P. *Molecular Biology of the Cell.* NewYork: Garland science, New York, USA, 5th edition, 2008.

[103] NIELSEN H., ENGELBRECHT J., BRUNAK S., and von HEIJNE G. Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *Protein Engineering*, 10(1):1–6, 1997.

[104] NIELSEN H. and KROGH A. Prediction of signal peptides and signal anchors by a hidden markov model. *Proc Int Conf Intell Syst Mol Biol.*, 6:122–130, 1998.

[105] PETERSEN T.N., BRUNAK S., von HEIJNE G., and NIELSEN H. Signalp 4.0: discriminating signal peptides from transmembrane regions., 2011.

[106] ALTSCHUL S.F., MADDEN T.L., SCHÄFFER A.A., ZHANG J., ZHANG Z., MILLER W., and LIPMAN D.J. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.*, 25:4436–4440, 1994.

[107] SMITH T.F. and WATERMAN M.S. Identification of common molecular subsequences. *J. Mol. Biol.*, 147:195–197, 1981.

[108] WOOTON J.C. Non-globular domains in protein sequences: automated segmentation using complexity measures. *Comput Chem.*, 18(3):269–285, 1994.

[109] JONES D.T. Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.*, 292:195–202, 1999.

[110] CARPENTER E.P., BEIS K., CAMERON A.D., and IWATA S. Overcoming the challenges of membrane protein crystallography. *Curr Opin Struct Biol.*, 18(5):581–586, 2008.

[111] WALLIN E., TSUKIHARA T., YOSHIKAWA S., von HEIJNE G., and ELOFSSON A. Architecture of helix bundle membrane proteins: An analysis of cytochrome c oxidase from bovine mitochondria. *Protein Science*, 3:808–815, 1997.

[112] KROGH A., LARSSON B, von HEIJNE G., and SONNHAMMER E.L.L. Predicting transmembrane protein topology with a hidden markov model: Application to complete genomes. *J. Mol Biol.*, 305:567–580, 2001.

[113] WORD J.M., LOVELL S.C., RICHARDSON J.S., and RICHARDSON D.C. Asparagine and glutamine: using hydrogen atom contacts in the choice of side-chain amide orientation. *J. Mol. Biol.*, 285(4):1735–1747, 1999.

[114] WORD J.M., LOVELL S.C., LABEAN T.H., TAYLOR H.C., ZALIS M.E., PRESLEYA B.K., RICHARDSON J.S., and RICHARDSON D.C. Visualizing and quantifying molecular goodness-of-fit: small-probe contact dots with explicit hydrogen atoms. *J. Mol. Biol.*, 285(4):1711–1733, 1999.

[115] HANSON R.M. Jmol - a paradigm shift in crystallographic visualization. *Journal of Applied Crystallography*, 43:1250–1260, Outubro 2010.

[116] HANSON R.M., PRILUSKY J., RENJIAN Z., NAKANE T., and SUSSMAN J.L. Jsmol and the next-generation web-based representation of 3d molecular structure as applied to proteopedia. *Israel Journal of Chemistry*, 53:207–216, 2013.

[117] RATSCHILLER T., MÜLLER O., DELISLE M., and CHAPEAUX L. phpmyadmin. 2017.

[118] Vii emmsb 2016. `http://www.emmsb.lncc.br/`. Último acesso em: 2017-03-21.

[119] LUDLOW R.F., VERDONK M.L., SAINI H.K., TICKLE I.J., and JHOTI H. Detection of secondary binding sites in proteins using fragment screening. *Proc.Natl.Acad.Sci.USA*, 112:15910, 2015. PDB ID: 5FPY.

[120] YAO N., REICHERT P., TAREMI S.S., PROSISE W.W., and WEBER P.C. Molecular views of viral polyprotein processing revealed by the crystal structure of the hepatitis c virus bifunctional protease-helicase. *Structure Fold.Des.*, 7:1353–1363, 1999. PDB ID: 1CU1.

[121] YAO N., HESSON T., CABLE M., HONG Z., KAWONG A.D., LE H.V., and WEBER P.C. Structure of the hepatitis c virus rna helicase domain. *Nat.Struct.Biol*, 4:463–467, 1997. PDB ID: 1HEI.

[122] MURZIN A.G., BRENNER S.E., HUBBARD T., and CHOTIA C. Scop: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, 247:536–540, 1995.

[123] SILLITOE I., LEWIS T.E., CUFF A.L., DAS S., ASHFORD P., DAWSON N.L., FURNHAM N., LASKOWSKI R.A., LEE D., LEES J., LEHTINEN S., STUDER R., THORNTON J.M., and ORENGO C.A. Cath: comprehensive structural and functional annotations for genome sequences. *Nucleic Acids Res.*, 43(D1):D376–D381, 2015.

[124] SCHÖDINGER L.L.C. The PyMOL molecular graphics system, version 1.8. November 2015.

[125] SIEVERS F., WILM A., DINEEN D., GIBSON T.J., KARPLUS K., LI W., LIPEZ R., MCWILLIAM H., REMMERT M., SÖDING J., THOMPSON J.D., and HIGGINS D.G. Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega. *Molecular Systems Biology*, 7:539, 2011.

[126] EDGAR R.C. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797, 2004.

[127] NOTREDAME C., DESMOND G.H., and HERINGA J. T-coffee: A novel method for multiple sequence alignments. *Journal of Molecular Biology*, 302:205–217, 2000.

[128] BRAK K., KERR I.D., BARRETT K.T., FUCHI N., DEBNATH M., ANG K., ENGEL J.C., MCKERROW J.H., DOYLE P.S., BRINEN L.S., and ELLMAN J.A. Nonpeptidic tetrafluorophenoxymethyl ketone cruzain inhibitors as promising new leads for chagas disease chemotherapy. *J.Med.Chem.*, 53:1763–1773, 2010. PDB ID: 3IUT.

[129] BRYANT C., KERR I.D., DEBNATH M., ANG K.K. RATNAM J., FERREIRA R.S., JAISHANKAR P., ZHAO D., ARKIN M.R., MCKERROW J.H., BRINEN L.S., and RENSLO A.R. Novel non-peptidic vinylsulfones targeting the s2 and s3 subsites of parasite cysteine proteases. *Bioorg.Med.Chem.Lett.*, 19:6218–6221, 2009. PDB ID: 3HD3.

[130] GILLMOR S.A. Chapter 3: X-ray structures of complexes of cruzain with designed covalent inhibitors. *Enzyme-ligand Interactions, Inhibition and Specificity*, pages 50–80, 1998. PDB ID: 1EWP.

[131] JONES B.D., TOCHOWICZ A., TANG Y., CAMERON M.D., MCCALL L.I., HIRATA K., SIQUEIRA-NETO J.L., REED S.L., MCKERROW J.H., and ROUSH W.R. Synthesis and evaluation of oxyguanidine analogues of the cysteine protease inhibitor wrr-483 against cruzain. *Acs Med.Chem.Lett.*, 7:77–82, 2016. PDB ID: 4XUI.

[132] LI W. and GODZIK A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics*, 22(13):1658–1659, 2006.

[133] Alex Herbert. Maxcluster - a tool for protein structure comparison and clustering. `http://www.sbg.bio.ic.ac.uk/~maxcluster/index.html`.

[134] CHENG Y. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, Aug 1995.

[135] FUKUNAGA K. and HOSTETLER L. The estimation of the gradient of a density function, with applications in pattern - recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.

[136] PEDREGOSA F., VAROQUAUX G., GRAMFORT A., MICHEL V., THIRION B., GRISEL O., M. BLONDEL, PETTERNHOFFER P., WEISS R., DUBOURG V., VANDERPLAS J., PASSOS A., D. COURNAPEAU, BRUCHER M., PERROT M., and DUCHESNAY E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[137] MCKINNEY W. Data structures for statistical computing in python. *Proceedings of the 9th Python in Science Conference*, pages 51–56, 2011.

[138] van der WALT S. and COLBERT S.C. VARQUAUX G. The numpy array: A structure for efficient numerical computation. *Computing in Science and Engineering*, 13:22–30, 2011.

[139] PEARSON K. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559?572, 1901.

[140] HOTELING H. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(11):417?441 and 498?520, 1933.

[141] EL-SAYED N.M., MYLER P.J., BARTHOLOMEU D.C., NILSSON D., AGGARWAL G., TRAN A.N., GHEDIN E., WORTHEY E.A., DELCHER A.L., BLANDIN G., WESTENBERGER S.J., CALER E., CERQUEIRA G.C., BRANCHE C., HAAS B., ANUPAMA A., ARNER E., ASLUND L., ATTIPOE P., BONTEMPI E., BRINGAUD F., BURTON P., CADAG E., CAMPBELL D.A., CARRINGTON M., CRABTREE J., DARBAN H., DA SILVEIRA J.F., DE JONG P., EDWARDS K., ENGLUND P.T., FAZELINA G., FELDBLYUM

T., FERELLA M., FRASCH A.C., GULL K., HORN D., HOU L., HUANG Y., KINDLUND E., KLINGBEIL M., KLUGE S., KOO H., LACERDA D., LEVIN M.J., LORENZI H., LOUIE T., MACHADO C.R., MCCULLOCH R., MCKENNA A., MIZUNO Y., MOTTRAM J.C., NELSON S., OCHAYA S., OSOEGAWA K., PAI G., PARSONS M., PENTONY M., PETTERSSON U., POP M., RAMIREZ J.L., RINTA J., ROBERTSON L., SALZBERG S.L., SANCHEZ D.O., SEYLER A., SHARMA R., SHETTY J., SIMPSON A.J., SISK E., TAMMI M.T., TARLETON R., TEIXEIRA S., VAN AKEN S., VOGT C., WARD P.N., WICKSTEAD B., WORTMAN J., WHITE O., FRASER C.M., STUART K.D., and ANDERSSON B. The genome sequence of trypanosoma cruzi, etiologic agent of chagas disease. *Science*, 309(5733):409–415, 2005.

[142] CECCARELLI C., GRODSKY N.B., ARIYARATNE N., COLMAN R.F., and BAHNSON B.J. Crystal structure of porcine mitochondrial nadp+-dependent isocitrate dehydrogenase complexed with mn2+ and isocitrate. *J.Biol.Chem.*, 277:43454–43462, 2002. PDB ID: 1LWD.

[143] LIANG J., JAFFREY S.R., GUO W., SNYDER S.H., and CLARDY J. Structure of the pin/lc8 dimer with a bound peptide. *Nat.Struct.Biol.*, 6:735–740, 1999. PDB ID: 1CMI.

[144] TERADA T., SHIROUZU M., FUKUMORI Y., FUJIMORI F., ITO Y., KIGAWA T., YOKOYAMA S., and UCHIDA T. Solution structure of the human parvulin-like peptidyl prolyl cis/trans isomerase, hpar14. *J.Mol.Biol.*, 305:917–926, 2001. PDB ID: 1FJD.

[145] HEIKINHEIMO P., TUOMINEN V., AHONEN A.-K., TEPLYAKOV A., COOPERMAN B.S., BAYKOV A.A., LAHTI R., and GOLDMAN A. Toward a quantum-mechanical description of metal-assisted phosphoryl transfer in pyrophosphatase. *Proc.Natl.Acad.Sci.USA*, 98:3121–3126, 2001. PDB ID: 1E9G.

[146] BUSCHIAZZO A., AMAYA M.F., CREMONA M.L., FRASCH A.C., and ALZARI P.M. The crystal structure and mode of action of trans-sialidase, a key enzyme of trypanosoma cruzi pathogenesis. *Mol.Cell*, 10:757–768, 2002. PDB ID: 1MS9.

[147] PARK F., GAJIWALA K., NOLAND B., WU L., HE D., MOLINARI J., LOOMIS K., PAGARIGAN B., KEARINS P., CHRISTOPHER J., PEAT T., BADGER J., HENDLE J., LIN J., and BUCHANAN S. The 1.59 a resolution crystal structure of tm0096, a flavin mononucleotide binding protein from thermotoga maritima. *Proteins*, 55:772–774, 2004. PDB ID: 1VHN.

[148] KAMENSKI T., HEILMEIER S., MEINHART T., and CRAMER P. Structure and mechanism of rna polymerase ii ctd phosphatases. *Mol.Cell*, 15:399–407, 2004. PDB ID: 1TA0.

[149] DROZDETSKIC A., COLE C., PROCTER J., and BARTON G.J. Jpred4: a protein secondary structure prediction server. *Nucl Acids Res*, 43(W1):W389–W394, 2015.

[150] WANG Z., ZHAO F., PENG J., and XU J. Protein 8-class secondary structure prediction using conditional neural fields. *Proteomics*, 11(19):3786–3792, 2011.

[151] YACHDAV G., KAJAN L. KLOPPMANN E. and, HECHT M., GOLDBERG T., HAMP T., HÖNIGSCHMID P., SCHAFFERHANS A., ROOS M., BERNHOFER M3., RICHTER L., ASHKENAZY H., PUNTA M., SCHLESSINGER A., BROMBERG Y., SCHNEIDER R., VRIEND G., SANDER C., BEN-TAL N., and ROST B. Predictprotein – an open resource for online prediction of protein structural and functional features. *Nucleic acids research*, pages W337–W343, 2014.

10 APÊNDICE

Algorithm 1: Pseudocódigo da geração de restrição.

```
ali = [‘‘DNSS-----PPAVPQ’’, ‘‘-NSS-----PPAVPQ’’]          // Lista de sequências
rest = ‘‘CCCSSSSSSCCCCCS’’          // Predição de estrutura secundária do modelo
consenso = pop(ali)                        // Remoção do primeiro elemento de ali
foreach element IN ali                                      // Criação do consenso
 do
    i = 0
    foreach c IN element                              // Para cada caractere em element
     do
        if c == '-' AND consenso[i] == '-' then
            consenso[i] = '-'
        end
        else
            consenso[i] = 'x'
        end
        i++
    end
end
pop(ali)                                  // Descarta o primeiro elemento da lista
i = 0
foreach c IN consenso                             // Para cada caractere em consenso
 do
    if c != '-'                                    // Se o caractere não for um gap
     then
        rest[i] = '-'                // A restrição na mesma posição será ignorada
    end
    i++
end
i = 0
foreach c IN consenso                  // Gerando variação da esquerda para direita
 do
    if (i == 0 OR rest[i-1] == '-') AND rest[i] != '-' AND c != '-' then
        sortear se rest[i] será '-'
    end
    i++
end
i = tamanho de rest
while i > 0                           // Gerando variação da direita para a esquerda
 do
    if c != '-'                                    // Se o caractere não for um gap
     then
        sorteio se rest[i] será '-'
    end
    i--
end
```

Tabela 10.1: Tabela de resultados do *script* de agrupamento.

	Filename	Cluster
CLUSTER MEDOID:	1_5FPY.B99990003.pdb	0
	1_5FPY.B99990001.pdb	0
	1_5FPY.B99990003.pdb	0
	1_5FPY.B99990004.pdb	0
	1_5FPY.B99990005.pdb	0
	1_5FPY.B99990006.pdb	0
	1_5FPY.B99990008.pdb	0
	1_5FPY.B99990010.pdb	0
	1_5FPY.B99990013.pdb	0
	1_5FPY.B99990015.pdb	0
	1_5FPY.B99990017.pdb	0
	1_5FPY.B99990018.pdb	0
	1_5FPY.B99990019.pdb	0
	1_5FPY.B99990020.pdb	0
CLUSTER MEDOID:	1_5FPY.B99990014.pdb	1
	1_5FPY.B99990007.pdb	1
	1_5FPY.B99990009.pdb	1
	1_5FPY.B99990011.pdb	1
	1_5FPY.B99990014.pdb	1
	1_5FPY.B99990016.pdb	1
CLUSTER MEDOID:	1_5FPY.B99990012.pdb	2
	1_5FPY.B99990002.pdb	2
	1_5FPY.B99990012.pdb	2

Tabela 10.2: Comparação dos resultados das pontuações de molpdf para o conjunto teste usando ou não o pacote paralelo do Modeller.

Modelo	2W4J		1JG8		5M5U		1HI8		5FPY	
	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]
1	1291,862	1292,042	1497,138	1554,195	1885,499	1816,347	3419,449	3043,687	3499,652	3378,920
2	1230,992	1335,969	1522,270	1541,183	1849,046	1868,046	3120,031	3143,829	4596,555	3543,628
3	1286,322	1239,787	1383,656	1452,543	1821,750	1802,095	3472,731	3397,085	3426,283	3441,791
4	1220,480	1524,174	1537,191	1363,813	1862,326	1976,850	3143,016	3324,770	3506,686	4924,947
5	1343,982	1233,973	1346,802	1402,041	1993,304	1808,000	3651,821	3189,006	3532,741	3653,455
6	1227,092	1373,261	1445,188	1370,944	1898,407	2534,065	3118,890	3283,486	3515,002	3696,805
7	1269,792	1279,443	1467,023	1499,100	2109,930	1864,313	3280,219	3112,931	3796,737	3481,125
8	1264,360	1345,186	1516,453	1327,724	2037,907	1917,860	3235,384	3239,922	4443,668	3707,963
9	1239,586	1223,436	1465,796	1388,335	1862,518	1884,467	3221,232	3107,719	3518,674	3355,143
10	1266,232	1252,789	1413,981	1357,788	1840,520	1881,599	3198,126	3159,369	4205,167	3551,123
11	1249,980	1419,761	1567,877	1657,995	2009,079	2025,823	3284,022	3172,320	3577,817	4054,370
12	1273,823	1270,840	1423,124	1534,259	1982,210	1807,513	3142,733	3142,534	4229,020	3719,227
13	1238,200	1436,803	1553,228	1397,887	1942,711	1927,877	3119,415	3348,714	3414,041	3617,918
14	1243,515	1208,928	1386,310	1416,664	1907,187	1905,924	3257,975	3161,957	3695,345	3465,323
15	1270,298	1432,583	1400,682	1611,888	1784,933	1879,904	3286,296	3017,075	3490,625	3437,016
16	1307,211	1244,543	1439,862	1380,722	1933,736	2074,365	3150,489	3264,424	4158,983	3670,874
17	1319,471	1285,921	1503,533	1398,385	1966,947	1846,794	3303,102	3169,907	3794,302	3698,602
18	1355,935	1308,766	1405,507	1460,949	1926,607	1880,804	3505,503	3264,167	3409,384	3426,108
19	1239,729	1475,618	1423,429	1402,980	1886,355	1995,190	3387,329	3128,952	3780,026	3764,761
20	1329,202	1271,818	1358,318	1482,480	1888,290	1972,164	3148,085	3144,952	3567,490	3706,225
Média	1273,403	1322,782	1452,868	1450,094	1919,463	1933,500	3272,292	3190,840	3757,910	3664,766
Desvio Padrão	39,076	89,416	64,044	88,337	77,439	156,079	144,631	96,202	357,089	331,522

Rodadas considerando ([a]P) e não considerando ([b]NP) o uso da biblioteca paralela do Modeller.

Tabela 10.3: Comparação dos resultados das pontuações de DOPE-HR para o conjunto teste, obtido com o *software* Modeller.

Modelo	2W4J		1JG8		5M5U		1HI8		5FPY	
	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]
1	-32866,359	-33141,137	-40767,082	-40815,684	-38651,734	-39379,496	-76544,711	-78077,227	-50124,262	-50190,559
2	-32679,445	-32223,010	-40104,438	-40708,328	-39400,875	-39188,605	-78103,297	-77425,031	-47969,797	-50100,992
3	-32646,053	-32700,414	-40352,805	-40723,508	-39673,109	-39136,273	-77429,000	-76682,211	-50989,285	-51127,496
4	-32834,754	-31915,963	-40015,969	-40822,387	-39372,211	-38659,922	-77768,164	-77189,984	-51553,219	-48432,238
5	-32921,785	-33001,633	-40985,598	-40515,723	-39107,344	-39358,574	-76910,391	-77093,234	-50683,168	-49398,473
6	-32998,855	-32561,104	-40418,168	-40630,977	-39277,828	-38678,305	-77930,477	-76350,734	-50222,234	-47527,137
7	-32657,338	-32462,076	-40754,586	-39888,770	-38944,207	-39094,406	-77571,008	-77966,984	-46992,730	-49220,137
8	-32517,107	-32593,705	-40252,801	-40830,684	-38880,141	-39206,867	-77213,414	-77174,938	-50156,660	-51520,254
9	-33164,887	-32681,629	-40185,188	-40526,027	-39037,891	-39030,848	-77239,750	-78017,195	-47803,266	-48891,813
10	-32648,223	-32942,543	-40721,934	-41394,117	-39280,719	-39526,898	-77189,117	-77160,633	-47727,414	-49511,516
11	-32786,805	-32484,336	-40341,781	-40221,168	-38866,539	-39036,547	-77019,445	-77633,172	-48288,418	-48820,156
12	-32602,199	-32645,732	-40977,586	-40516,918	-39379,316	-39592,598	-77072,125	-77996,719	-48859,719	-50993,156
13	-33010,707	-32072,514	-40697,805	-40916,016	-39146,070	-39005,469	-78742,422	-76991,773	-49757,387	-48204,820
14	-32371,703	-32886,184	-40972,469	-40308,930	-39148,441	-39039,324	-77221,148	-77417,953	-47598,336	-49382,316
15	-32643,023	-31623,023	-40870,539	-40821,730	-39669,965	-39221,828	-77484,148	-77746,758	-51269,148	-50905,289
16	-32561,463	-33207,660	-40557,254	-40860,848	-38961,199	-38069,938	-77393,719	-76967,625	-47142,043	-50250,277
17	-32606,803	-32758,791	-40675,418	-40674,129	-39172,676	-39598,938	-76112,766	-77952,039	-49374,988	-51058,777
18	-32835,195	-32527,334	-40672,211	-40850,844	-39051,043	-39672,137	-76295,406	-77192,234	-49748,738	-51138,574
19	-32626,434	-32068,426	-40460,023	-40729,969	-39265,055	-39577,176	-76742,344	-77192,836	-51476,844	-50737,230
20	-32519,766	-32961,098	-40855,465	-40611,934	-39592,590	-38589,902	-77236,266	-77633,594	-50619,164	-50073,004
Média	-32724,945	-32572,916	-40581,956	-40668,434	-39193,948	-39133,203	-77260,956	-77393,144	-49417,841	-49874,211
Desvio Padrão	192,095	409,082	292,458	297,505	266,175	393,790	594,186	466,533	1469,324	1098,934

Rodadas considerando ([a]P) e não considerando ([b]NP) o uso da biblioteca paralela do Modeller.

Tabela 10.4: Comparação dos resultados das pontuações de nDOPE para o conjunto teste, obtido com o *software* Modeller.

Modelo	2W4J		1JG8		5M5U		1HI8		5FPY	
	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]
1	-1,708	-1,737	-1,818	-1,816	-0,936	-0,986	-1,413	-1,482	0,287	0,363
2	-1,712	-1,678	-1,762	-1,814	-0,977	-0,934	-1,485	-1,476	0,441	0,432
3	-1,690	-1,688	-1,774	-1,842	-1,006	-0,985	-1,462	-1,412	0,380	0,197
4	-1,735	-1,637	-1,766	-1,832	-0,992	-0,916	-1,464	-1,454	0,294	0,425
5	-1,762	-1,707	-1,840	-1,809	-0,986	-0,992	-1,448	-1,452	0,326	0,428
6	-1,713	-1,677	-1,787	-1,820	-0,977	-0,910	-1,484	-1,402	0,355	0,518
7	-1,706	-1,688	-1,812	-1,758	-0,943	-0,965	-1,476	-1,485	0,568	0,385
8	-1,696	-1,716	-1,783	-1,833	-0,978	-0,962	-1,451	-1,462	0,355	0,218
9	-1,738	-1,727	-1,794	-1,820	-0,973	-0,966	-1,441	-1,462	0,517	0,505
10	-1,712	-1,739	-1,830	-1,874	-0,974	-0,993	-1,459	-1,470	0,373	0,413
11	-1,735	-1,670	-1,818	-1,757	-0,942	-0,972	-1,443	-1,470	0,539	0,501
12	-1,705	-1,705	-1,861	-1,800	-1,032	-0,991	-1,452	-1,467	0,441	0,380
13	-1,728	-1,652	-1,822	-1,817	-0,945	-0,955	-1,517	-1,445	0,324	0,516
14	-1,689	-1,723	-1,842	-1,806	-0,995	-0,963	-1,434	-1,466	0,559	0,476
15	-1,704	-1,596	-1,828	-1,855	-0,993	-0,983	-1,453	-1,477	0,197	0,316
16	-1,690	-1,746	-1,813	-1,831	-0,957	-0,871	-1,459	-1,439	0,554	0,291
17	-1,703	-1,719	-1,822	-1,808	-0,969	-1,016	-1,419	-1,472	0,323	0,161
18	-1,708	-1,698	-1,789	-1,861	-0,936	-1,019	-1,414	-1,455	0,374	0,374
19	-1,686	-1,645	-1,814	-1,811	-0,963	-1,009	-1,432	-1,461	0,275	0,428
20	-1,691	-1,717	-1,845	-1,797	-1,009	-0,914	-1,467	-1,475	0,404	0,217
Média	-1,711	-1,693	-1,811	-1,818	-0,974	-0,965	-1,454	-1,459	0,394	0,377
Desvio Padrão	0,020	0,038	0,027	0,028	0,025	0,038	0,025	0,021	0,104	0,108

Rodadas considerando ([a]P) e não considerando ([b]NP) o uso da biblioteca paralela do Modeller.

Tabela 10.5: Comparação do resultados dos valores de RMSD para o conjunto teste, obtido com o *software* Pymol.

Modelo	2W4J		1JG8		5M5U		1HI8		5FPY[1]	
	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]
1	0,138	0,109	0,089	0,121	0,120	0,146	0,118	0,118	2,125	2,125
2	0,097	0,152	0,120	0,107	0,120	0,122	0,102	0,104	2,108	2,128
3	0,129	0,114	0,099	0,100	0,138	0,138	0,122	0,115	2,069	2,271
4	0,146	0,127	0,104	0,100	0,134	0,130	0,102	0,104	2,042	2,206
5	0,107	0,119	0,094	0,096	0,146	0,142	0,120	0,103	2,018	2,136
6	0,119	0,101	0,096	0,108	0,136	0,122	0,104	0,108	2,104	2,304
7	0,098	0,105	0,102	0,108	0,139	0,160	0,116	0,116	2,102	2,258
8	0,146	0,128	0,114	0,100	0,169	0,140	0,103	0,115	2,059	2,076
9	0,119	0,108	0,120	0,108	0,155	0,128	0,110	0,109	2,211	2,176
10	0,135	0,108	0,089	0,120	0,115	0,130	0,125	0,116	2,128	2,056
11	0,130	0,120	0,111	0,100	0,136	0,127	0,113	0,109	2,119	2,165
12	0,111	0,122	0,165	0,095	0,142	0,156	0,109	0,107	2,066	2,120
13	0,103	0,165	0,103	0,147	0,153	0,126	0,096	0,121	1,987	2,285
14	0,126	0,100	0,120	0,105	0,147	0,144	0,117	0,098	2,063	2,093
15	0,138	0,160	0,088	0,116	0,166	0,126	0,104	0,099	2,069	2,089
16	0,126	0,114	0,094	0,110	0,127	0,142	0,096	0,106	2,058	2,166
17	0,121	0,109	0,119	0,137	0,154	0,154	0,121	0,099	2,231	2,134
18	0,123	0,128	0,117	0,132	0,133	0,141	0,155	0,102	2,163	2,070
19	0,108	0,117	0,096	0,125	0,134	0,149	0,118	0,104	1,939	2,186
20	0,111	0,103	0,149	0,103	0,131	0,147	0,088	0,095	2,251	2,327
Média	0,122	0,120	0,109	0,112	0,140	0,139	0,112	0,107	2,096	2,169
Desvio Padrão	0,015	0,018	0,019	0,014	0,014	0,011	0,014	0,007	0,076	0,080

[1]A proteína 5FPY teve seu RMSD calculado com a proteína 1CU1 ao invés de seu molde original 1HEI, no intuito de simular uma condição de proteína com cobertura reduzida. Rodadas considerando ([a]P) e não considerando ([b]NP) o uso da biblioteca paralela do Modeller.

Tabela 10.6: Comparação da quantidade de *outliers* no gráfico de Ramanchandran, obtido para o conjunto teste com o *software* Molprobity.

Modelo	2W4J		1JG8		5M5U		1HI8		5FPY	
	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]
1	0	0	1	0	2	0	4	3	11	15
2	0	0	1	1	0	0	3	3	17	7
3	0	0	1	2	1	0	3	3	12	10
4	0	0	1	2	0	1	3	3	9	17
5	0	0	0	1	2	0	3	3	15	13
6	0	0	1	0	1	1	3	3	9	9
7	0	0	2	1	0	0	4	3	14	19
8	0	0	1	1	0	0	3	2	13	15
9	0	0	1	1	0	0	3	3	13	11
10	0	0	1	0	0	1	1	3	11	11
11	0	1	1	1	1	0	3	3	9	12
12	0	0	1	1	0	0	3	3	16	14
13	0	0	1	2	0	0	3	3	7	9
14	0	0	1	1	1	1	3	3	12	13
15	0	0	1	1	0	0	2	3	12	13
16	0	0	1	1	1	1	3	4	13	11
17	0	0	1	1	2	0	3	3	15	15
18	0	0	0	0	1	0	3	3	11	12
19	0	0	1	1	1	0	3	3	8	13
20	0	0	1	1	0	1	3	3	13	11
Média	0,000	0,050	0,950	0,950	0,650	0,300	2,950	3,000	12,000	12,500
Desvio Padrão	0,000	0,218	0,384	0,589	0,726	0,458	0,589	0,316	2,627	2,784

Rodadas considerando ([a]P) e não considerando ([b]NP) o uso da biblioteca paralela do Modeller.

Tabela 10.7: Comparação da porcentagem do número de resíduos alocados em regiões favoráveis no gráfico de Ramanchandran, obtido para o conjunto teste com o *software* Molprobity.

Modelo	2W4J		1JG8		5M5U		1HI8		5FPY	
	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]	P[a]	NP[b]
1	100	100	99,7	100	99,4	100	99,4	99,5	98,3	97,7
2	100	100	99,7	99,7	100	100	99,5	99,5	97,400	98,900
3	100	100	99,7	99,4	99,7	100	99,5	99,5	98,200	98,500
4	100	100	99,7	99,4	100	99,7	99,5	99,5	98,600	97,400
5	100	100	100	99,7	99,4	100	99,5	99,5	97,700	98,000
6	100	100	99,7	100	99,7	99,7	99,5	99,5	98,600	98,600
7	100	100	99,4	99,7	100	100	99,4	99,5	97,900	97,100
8	100	100	99,7	99,7	100	100	99,5	99,7	98,000	97,700
9	100	100	99,7	99,7	100	100	99,5	99,5	98,000	98,300
10	100	100	99,7	100	100	99,7	99,8	99,5	98,300	98,300
11	100	99,6	99,7	99,7	99,7	100	99,5	99,5	98,600	98,200
12	100	100	99,7	99,7	100	100	99,5	99,5	97,600	97,900
13	100	100	99,7	99,4	100	100	99,5	99,5	98,900	98,600
14	100	100	99,7	99,7	99,7	99,7	99,5	99,5	98,200	98,000
15	100	100	99,7	99,7	100	100	99,7	99,5	98,200	98,000
16	100	100	99,7	99,7	99,7	99,7	99,5	99,4	98,000	98,300
17	100	100	99,7	99,7	99,4	100	99,5	99,5	97,700	97,700
18	100	100	100	100	99,7	100	99,5	99,5	98,300	98,200
19	100	100	99,7	99,7	99,7	100	99,5	99,5	98,800	98,000
20	100	100	99,7	99,7	100	99,7	99,5	99,5	98,000	98,300
Média	100,000	99,980	99,715	99,715	99,805	99,910	99,515	99,505	98,165	98,085
Desvio Padrão	0,000	0,087	0,115	0,177	0,218	0,137	0,085	0,050	0,393	0,420

Rodadas considerando ([a]P) e não considerando ([b]NP) o uso da biblioteca paralela do Modeller.