Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

Desenvolvimento de sistema de reuso de dados para o *workflow* MHOLline

Dimitri Guglinski Siqueira

JUIZ DE FORA
NOVEMBRO, 2018

Desenvolvimento de sistema de reuso de dados para o *workflow* MHOLline

Dimitri Guglinski Siqueira

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientadora: Priscila Vanessa Zabala Capriles Goliatt

JUIZ DE FORA

NOVEMBRO, 2018

Desenvolvimento de sistema de reuso de dados para o *workflow* MHOLline

Dimitri Guglinski Siqueira

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Priscila Vanessa Zabala Capriles Goliatt
Doutora em Modelagem Computacional (LNCC)

Jairo Francisco de Souza
Doutor em Informática (PUC-RJ)

Luiz Felipe Carvalho Mendes
Mestre em Modelagem Computacional (UFJF)

JUIZ DE FORA

20 DE NOVEMBRO, 2018

Resumo

Apesar de todos os avanços científicos e descobertas feitas no âmbito biológico e medicinal até hoje, a humanidade ainda se depara com alguns problemas quanto ao tratamento de doenças transmissíveis em geral. Tais problemas se devem ao fato de biomédicos, bioquímicos, biólogos, farmacêuticos, dentre outros cientistas da área, não conseguirem um progresso efetivo em suas pesquisas, por limitações físicas, falta de investimento, falta de conhecimento ou por não possuírem ferramentas adequadas para o progresso científico. Com o auxílio da computação, os pesquisadores vêm aumentando o índice de sucesso em suas pesquisas. O MHOLline, a ferramenta abordada no título, é um *workflow* científico desenvolvido para modelar, tridimensionalmente, por comparação, estruturas moleculares de proteínas, de forma rápida e com certo grau de precisão, muitas vezes, satisfatório. Esse tipo de predição auxilia os cientistas da área da saúde no desenvolvimento de novos remédios, para que sejam mais eficientes no combate das doenças em geral. Este trabalho apresenta uma solução para o problema de armazenamento do *workflow* MHOLline, mostrando o passo a passo na criação de um sistema de reuso de dados, capaz de ressubmeter tarefas salvas pelo usuário na etapa final de processamento, permitindo que essas tarefas sejam reenviadas à plataforma após possuírem suas validades expiradas, e, consequentemente, auxiliando pesquisas a atingirem seus propósitos.

Palavras-chave: *Workflow* científico, MHOLline, ferramenta de reuso de dados.

Agradecimentos

Agradeço a meu pai Márcio, a minha irmã Julita, pelo encorajamento e apoio nos estudos; À professora Priscila Capriles pela orientação, amizade, paciência e confiança, sem os quais este trabalho não se realizaria; Aos meus amigos do projeto, Artur Rossi e Pedro Eveling, por todo auxílio e esclarecimentos que necessitei a respeito do MHOLline; Aos meus amigos de vida que me deram apoio e coragem para caminhar até aqui; À Universidade Federal de Juiz de Fora, por me fornecer, com qualidade, a estrutura necessária para que minha ciência fosse criada; Aos demais professores do Departamento de Ciência da Computação pelos seus ensinamentos, ideias, e todo conhecimento disseminado; E, sobretudo, por quem arquiteta essa obra de arte chamada vida.

"Those who can imagine anything can create the impossible".

Alan Turing

Conteúdo

Lista de Abreviações

DCC	Departamento de Ciência da Computação
UFJF	Universidade Federal de Juiz de Fora
TCC	Trabalho de Conclusão de Curso
3D	Tridimensional
RX	Raios X
RMN	Ressonância Magnética Nuclear
PDB	do inglês *Protein Data Bank*
BLAST	do inglês *Basic Alignment Search Tool*
BATS	do inglês *BLAST Automatic Targeting for Structures*
EC	do inglês *Enzyme Commission*
PCR	Reação em cadeia da Polimerase, do inglês *Polimerase Chain Reaction*
SGBD	Sistema de gerenciamento de banco de dados
BD	Banco de dados
TrEMBL	do inglês *translation of EMBL nucleotide sequence database*
EMBL	do inglês *European Molecular Biology Laboratory*
HMMTOP	do inglês *Hidden Markov Model Topology*
SQL	do inglês *Structured query language*
CSV	do inglês *Comma-separated values*
SHA	do inglês *Secure Hash Algorithm*
NSA	do inglês *National Security Agency*

1 Introdução

Com o intuito de facilitar o avanço nos estudos da biologia, a computação tomou frente, há algumas décadas, na elaboração de *softwares* auxiliadores, instrutivos, simuladores e decodificadores para o ramo da biologia. Assim, com o passar dos anos, foi surgindo o conceito de Bioinformática. A Bioinformática se refere ao emprego de ferramentas computacionais no estudo de problemas e questões biológicas, abrangendo também as aplicações relacionadas à saúde humana, como, por exemplo, no planejamento de novos medicamentos e fármacos. (Verli 2014)

No entanto, no começo dos anos 70, época que o conceito de bioinformática ainda estava sendo enraizado, detectou-se um importante problema: a falta de comunicação e entendimento entre as partes distintas dos projetos em desenvolvimento. O informata não compreendia totalmente as necessidades do biólogo e vice-versa, como descrito abaixo:

"Os primeiros projetos na área eram compostos por profissionais de diferentes áreas da biologia e informática e percebia-se uma certa dificuldade de comunicação: enquanto o biólogo procurava uma solução que levasse em consideração as incertezas e erros que ocorrem na prática, o cientista da computação procurava uma solução eficiente para um problema bem definido. Assim, surgiu a necessidade de um novo profissional, que entendesse bem ambas as áreas e fizesse a ponte entre elas: o Bioinformata. Esse profissional deveria ter o conhecimento suficiente para saber quais eram os problemas biológicos reais e quais seriam as opções viáveis de desenvolvimento e abordagem computacional dos problemas em questão. " (Prosdocimi et al. 2002)

Portanto, nos estudos em bioinformática é de suma importância a união dos conhecimentos nas áreas da biologia e da saúde com os conhecimentos computacionais. Por exemplo, conhecer a estrutura ou modelo estrutural de uma determinada proteína é necessário para que possamos compreender alguma simples função dessa proteína. Esse tipo de estudo tem sido visado para ajudar bioquímicos e farmacêuticos a sintetizarem medicamentos cujas substâncias agem diretamente no funcionamento dessas proteínas, impedindo que elas exerçam funções metabólicas ou reprodutoras.

Capriles et al. 2010, falam sobre a importância de se utilizar programas para a predição da estrutura tridimensional de proteínas, citando o caso do *Trypanosoma cruzi*, protozoário causador da doença de Chagas, devido a um certo grau de ineficiência dos atuais métodos de farmacoterapia desta doença. Como resultado da pesquisa, após aplicarem técnicas de modelagem comparativa em 3286 proteínas presentes no protozoário, foram identificados 397 subsequências de enzimas, sendo 356 homologas, 3 análogas e 38 específicas ao parasita. A partir desses resultados, novas estratégias de combate ao *T. cruzi* têm sido analisadas ou desenvolvidas. Esse exemplo apresenta alguns benefícios do uso da computação em geral aplicada nas áreas da biologia e da saúde.

Ao longo dos anos, a computação vem ajudando a comunidade científica na área de ciências biológicas e da saúde, química e farmácia a desenvolverem suas pesquisas de forma mais rápida e objetiva. Considerando que as melhorias realizadas em um *software*, de tempos em tempos, é de suma importância para seu bom funcionamento, por questões de adequação às novas versões de sistemas, de bancos de dados, linguagens utilizadas em geral e por questões de segurança, o presente trabalho busca fornecer uma contribuição para a área da bioinformática, mais especificamente para o *workflow* MHOLline (Capriles et al. 2010), a ser descrito na Seção 1.2 e no Capítulo 3.

1.1 Apresentação do Tema

O tema abordado nesse trabalho é um composto entre reuso de dados (*data reuse*) previamente processados, inseridos em um contexto ao qual a proveniência desses dados (*data provenance*) é necessária. Há uma crescente preocupação em tomar conhecimento das origens de uma informação específica, para que ela não seja inserida de qualquer maneira e em qualquer meio de comunicação sem sua devida referência e/ou autorização, como descrito por Buneman, Khanna e Tan 2000.

De acordo com os autores, não bastando ser difícil identificar a origem de uma informação pela falta de referência ao publicar um artigo, se torna difícil provar de onde veio pelo menos alguma parte dessa informação, não necessariamente ela por inteiro. Sob tal análise, é importante que o reuso de dados tenha como garantia a unicidade de uma informação recuperada, e os autores propõem métodos de rastreabilidade da origem dessas

informações. Ademais, eles também levantam a questão de se estabelecer um novo modelo de armazenamento e trocas de informações, onde as informações possam ser rastreadas, contendo um histórico completo de como elas foram manejadas durante a sua existência, mantendo a linhagem dos dados.

O tema foi definido baseando-se na necessidade de garantir a origem de um arquivo manipulado pelo sistema de reuso de dados, principalmente por questões de segurança e, especificamente, levando em consideração o problema abordado e os três objetivos que serão definidos e explicados adiante.

1.2 Problema abordado

O MHOLline é um *workflow* de processamento de dados científicos no ramo da bioinformática. A plataforma cria predições estruturais de proteínas a partir de uma sequência de proteínas codificadas em um arquivo texto no formato FASTA, exemplificado na figura a seguir:

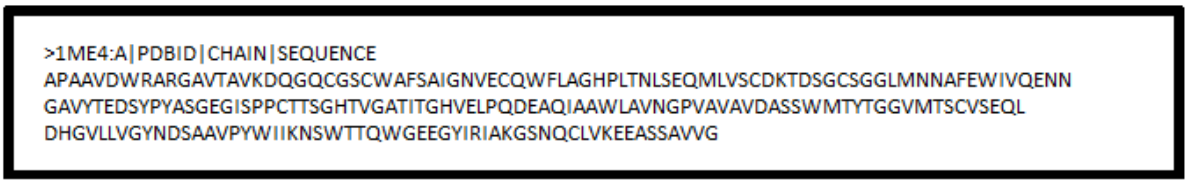

Figura 1.1: Conteúdo de um arquivo do formato .FASTA (estrutura 1ME4)

O arquivo FASTA pode ser interpretado por programas de alinhamento de proteínas, processo pelo qual se identifica, através do *Protein Data Bank* (PDB), uma estrutura molecular, dado uma sequência específica de aminoácidos. Esse alinhamento serve de base para o propósito principal do *workflow* MHOLline, que é a transformação dessas sequências em um compacto .TAR.GZ de arquivos provenientes dos resultados dos módulos executados no *workflow*. Denominamos o arquivo .TAR.GZ resultante de *job*.

No entanto, como qualquer programa, o MHOLline também possui as suas limitações. A plataforma garante a possibilidade de se executar o arquivo .FASTA em módulos distintos, alguns deles em sequência (gerando um fluxo de dados síncrono) e outros independentes, gerando resultados que não irão servir de entrada para outro módulo.

Inserir um novo programa, ou modificar o existente de acordo com as necessidades do trabalho, poderia acarretar em possíveis falhas na geração de um resultado, comprometendo o resultado final do *workflow* ou a sua robustez.

Além disso, o *workflow* funciona em um servidor com um espaço de armazenamento específico, e manter os arquivos processados dos pesquisadores nesse servidor por um tempo indeterminado acarretaria no preenchimento total de suas mídias de armazenamento, impossibilitando novos usuários de se beneficiarem do MHOLline, ou até mesmo podendo causar alguma falha no sistema. É custoso manter armazenado no servidor todo o conjunto de arquivos de proteínas modeladas, de análises realizadas, de refinamentos e dos dados descritivos do banco de dados. Por esse motivo, o sistema exclui periodicamente, para cada usuário, os resultados processados que se encontram no servidor e no banco dados com data superior a 15 dias. Se o usuário necessita realizar um estudo que exceda os 15 dias de validade do *job*, o próprio propósito do *job*, de contribuir com a ciência, foi em vão.

O problema constatado pode ser resolvido, tecnicamente, de diferentes maneiras. A informação perdida deve ser recuperada, isso pode ser feito importando/exportando arquivos .CSV (*Comma-separated values*) ou SQL (*Structured query language*) do banco de dados do sistema. A maneira adotada para solucionar o problema foi manipular, de forma semelhante ao .CSV, arquivos metadados, capazes de recuperar a informação perdida de um *job*.

1.3 Motivação

Devido a importância do *workflow* MHOLline no seu meio científico, é de grande relevância para o projeto atender as demandas de seus usuários por aperfeiçoamentos no sistema. Neste trabalho visamos atender como principal demanda o reaproveitamento de resultados previamente obtidos para novas análises e refinamentos sem a necessidade de ressubmeter o processo desde o início. Caso esse problema não fosse solucionado, o sistema do MHOLline teria de processar novamente um arquivo .FASTA, acarretando em custo computacional redundante e a perda de tempo pelo usuário na espera de que os resultados sejam novamente gerados. Contribuir com o aperfeiçoamento de um sistema

possibilita que os usuários não apenas desfrutem de novas tecnologias, mas que façam isso de forma segura e privada, considerando suas necessidades individuais.

1.4 Objetivos

Este trabalho tem como objetivo geral aplicar os conhecimentos adquiridos, através do curso de Ciência da Computação e da revisão bibliográfica realizada, no desenvolvimento da ferramenta de reuso de dados, garantindo a unicidade dos dados e a segurança do sistema. Para que o objetivo em questão seja alcançado, todos os objetivos específicos, que serão listados a seguir, devem ser alcançados também.

1.4.1 Objetivos Específicos

1. Desenvolver uma ferramenta de reuso de dados previamente processados no *workflow* MHOLline;

2. Validar os dados durante o processamento de *upload* do *job*, garantindo segurança ao sistema, integridade dos dados e a unicidade do conteúdo ressubmetido;

3. Integração básica-funcional entre a ferramenta desenvolvida e a plataforma já existente.

2 Revisão bibliográfica

Para realização desse trabalho, foi feito um estudo com foco nos conhecimentos computacionais a respeito dos termos "Reuso de dados *(Data Reuse)*", "Dados abertos *(Open data)*" , "Workflows *(Fluxos de trabalho)*" , "Proveniência de dados (*Data provenance*)" e "Bioinformática".

De acordo com o estudo feito sobre o artigo de Barns e Bollinger 1991, reutilizar toda a ideia criada e desenvolvida por trás de códigos, *layouts*, documentações, modelos de aplicação, estatísticas, dentre outros, simplifica o trabalho de remodelar/adaptar o que foi feito e permite que o projeto em questão seja elaborado desde o início com um norteamento antecipado dos resultados e do trabalho em si. No caso do artigo de Barns e Bollinger 1991, os autores abordam uma estratégia de reuso de informações para otimizar o custo financeiro ou o tempo de processamento em um fluxo de negócio através de otimizações lineares e estratégias de negócio. Podemos notar que o termo "reuso de dados" possui vertentes externas ao meio computacional, uma vez que a informação, por si só, não é algo inerente apenas ao meio computacional. O Reuso de dados, quando praticado de forma coordenada e planejada, pode garantir a melhoria de desempenho de um trabalho ou contenção de gastos envolvidos nele.

Fienberg et al. 1985 nos apresentam alguns motivos pertinentes acerca do compartilhamento de dados. Sob a hipótese de que os dados, contendo seus mais diversos formatos, dado certo interesse e valor ao seu conteúdo, ao serem compartilhados com entidades governamentais e pessoas de interesse, estarão sujeitos a receberem investimentos direta ou indiretamente, nesse suposto tipo de pesquisa científica. Para facilitar o acesso do seu trabalho à terceiros, as informações contidas na mídia compartilhada devem ser claras, bem documentadas e específicas quanto ao seus objetivos. Eles explicam que o cientista, ao questionar a fonte de uma informação científica, a maneira como ela foi elaborada, desenvolvida e descoberta, busca realizar experimentos e moldar qualquer empirismo sob a forma de fórmulas ou meta-heurísticas a fim de concluir sua tese ou determinar a solução ideal para suas supostas hipóteses.

Fica mais claro, portanto, que elaborar ciência considerando o teor e a veracidade do conteúdo-base de sua pesquisa, pode acarretar em uma maior credibilidade e aceitação por parte do meio acadêmico acerca de seu estudo. Do contrário, seria "questão de tempo" até ela ser desconsiderada por se tornar inconsistente, pelo simples fato dela contradizer, hipoteticamente, algo recém-comprovado que a contradiz. Sob o ponto de vista do Comitê Nacional de Estatísticas (CNSTAT), o ato de compartilhar dados científicos é essencial para reforçar a prática do inquérito científico (situação em que um artigo ou uma tese é submetida a sugestões, correções e adaptações).

Os autores White et al. 2013 atentam para a padronização da informação a ser reutilizada. Sob uma abordagem computacional, arquivos gravados em formatos padronizados, reconhecidos a nível mundial, são mais fáceis de serem acessados ou executados, pois a quantidade de *softwares* diversos que os reconhecem é vasta. Ao organizar tuplas, por exemplo, em um programa desenvolvido pela empresa *Microsoft*, o arquivo gerado pode não ser reconhecido por algum outro programa nativo de um outro sistema operacional diferente do *Windows*. Em decorrência disso, o usuário precisa utilizar um programa genérico para exibir o arquivo, ou então realizar uma conversão.

Além disso, os autores também atentam para o fato de que converter arquivos é comprometedor quanto à sua confiabilidade e ao seu conteúdo, pois pode haver perda de dados, modificações ou demais problemas nesse processo. Portanto, deve-se atentar à legitimidade e integridade daquela informação e de seus atributos inerentes. Vale lembrar que a padronização do nome do arquivo, utilizando os conceitos de linguagens formais, também é importante, pois assim conseguimos ordená-los por nome de acordo com suas características, data, tamanho, entre outros, caso opte pela não utilização de metadados para isso.

Sob o ponto de vista analítico de Piwowar e Vision 2013, os autores apresentam um estudo detalhado de sua pesquisa, comparando com pesquisas anteriores feitas pelo próprio grupo. Eles inferem estatisticamente que variáveis como a data de publicação (artigos recentes são mais citados em um certo período de tempo do que os antigos) e os jornais onde esses artigos foram publicados (os mais visados, reconhecidos ou com boa influência) influenciam na quantidade e na frequência que eles sejam citados. A

quantidade de citações aumenta de acordo com a disponibilidade dos artigos: a tendência é que as citações aumentam gradativamente ao publicar pesquisas em novos repositórios de dados científicos. Além disso, eles concluem que houve um crescimento nas citações referentes aos artigos mais acessíveis, recebendo cerca de 9% a mais de citações do que aqueles que não foram disponibilizados (95% no Intervalo de Confiança da estimativa, variando entre 5% e 13%). Acerca do trabalho realizado, é importante ressaltar que os benefícios gerados pelo reuso de dados, podem ser comprovados estatisticamente. No caso do que foi proposto e confirmado pelo trabalho acima, utilizando uma abordagem não-computacional ou não-algorítmica para o termo, o simples fato de citar artigos (reutilizar os dados do artigo) garante uma maior visibilidade do artigo citado.

No ambiente científico, é comum que os resultados de uma pesquisa sejam obtidos através de um fluxo virtual vasto de processamentos, dependentes ou não entre si. Esse conjunto de processamentos submetidos a uma determinada disposição, com suas regras de funcionamento e seu fluxo de dados bem definidos é denominado *Workflow*. Hollingsworth e Hampshire 1995 definem *workflow* como a "Automatização ou facilitação computadorizada de um processo de negócios, inteiramente ou em partes". As tarefas executadas no processo, com suas regras, possuem a função de alcançar realizações ou contribuir para que objetivos de negócios, de uma forma geral, sejam alcançados. Os autores, em suma, apresentam um "Modelo referencial" para sistemas de gerenciamento de *workflows*.

Workflows podem ser definidos por tipos, e cada tipo possui uma conjunto de regras e finalidades distintas. Como exemplo, existe uma diferença entre *business workflow* e *scientific workflow* sob a perspectiva de Ludäscher et al. 2006, pois as características e requisitos de um *workflow* científico são diferentes em relação ao de negócios. Enquanto os *workflows* de negócios visam um resultado baseado em controle de fluxo de produção, os científicos visam um resultado baseado no funcionamento, definido e regrado, de um fluxo de dados.

Existem empresas, como exemplo, a Prodigious brand logistics, que oferecem o serviço de elaboração completa de um modelo de *workflow* empresarial, que são vendidos para outras empresas que desejam melhorar seus respectivos processos de produção. De

fato, cada tipo de *workflow* possui, em sua essência, a finalidade de economizar tempo, custo e elaborar um trabalho de qualidade.

Um modelo científico de *workflow* assemelha-se a uma rede de processos de *data-flow*, ou está diretamente implementado como tal. As abordagens da orientação a fluxo de dados são aplicáveis em diferentes níveis de granularidade, da mais simples operação de baixo nível de uma CPU até paradigmas de programação em alto nível, como a de programação baseada em fluxo. Em suma, os autores Ludäscher et al. 2006 apresentam o sistema KEPLER, capaz de auxiliar o usuário na construção de um *workflow* científico sem que ele necessite ter um vasto conhecimento em programação ou esteja acostumado a programar, tudo através de facilidades que um sistema de gerenciamento de *workflow* oferece, como uma interface gráfica bem elaborada ou suporte via *web service*, dentre outros.

A comunidade científica, de forma geral, faz uso de *workflows* científicos para automatizarem e organizarem seus métodos de produção de informação. Assim, o usuário fica isento de certas preocupações, como a forma que as informações são processadas ou sob qual metodologia o fluxo de trabalho foi desenvolvido, caso seja considerado um cenário em que o *workflow* utilizado seja confiável e completamente funcional, sem erros. (Deelman et al. 2009).

Utilizando um exemplo deste trabalho, o MHOLline é um *workflow* que auxilia no processo de predição de estruturas proteicas, gerando resultados prontos para serem refinados e, por fim, escolhidos e, em sua mais recente versão, resultados também em forma de plotagem tridimensional (Reis et al. 2016).

Não só o MHOLline, mas outros *workflows*, como exemplo o SWISS-Prot, também atuam na área biológica. *Workflows* nessa área consistem, basicamente, em simular eventos biológicos, químicos e físicos para reduzir custos provenientes dessa pesquisa e acelerar o processo de obtenção de resultados. Para que esse processo inteiro apresente um resultado confiável, diversas análises devem ser feitas e decisões tomadas durante todo o fluxo. Até o próprio usuário influencia diretamente no resultado final, uma vez que ele necessita realizar o refinamento dos resultados obtidos após seguir algumas etapas do fluxo. O SWISS-Prot Protein Knowledgebase, *workflow* base do artigo de Boeckmann et al. 2003,

é constituído de um *Data bank* inteligente de informações sobre proteínas, aminoácidos e outros atributos que podem ser discretos de forma computacional. O trabalho citado acima resume um processo de integralização do TrEMBL (translation of EMBL nucleotide sequence database) ao SWISS-Prot, para gerar novas traduções de sequências de nucleotídeos e inseri-las no banco de dados daquele sistema.

Digiampietri et al. 2005 tentam provar e desenvolver um modelo de *framework* padrão para resolver os problemas de aplicações de bioinformática, e espera alcançar um *framework* que seja capaz de especificar *workflows* através de *web services* e que seja capaz de armazenar essa especificação; de descobrir, de maneira semântica, serviços e *workflows* de interesse específico; de gerenciar a execução do *workflow* via orquestra de serviços; e que possa auditar a execução do fluxo.

Os autores identificaram dois principais tipos de *frameworks* para projetos de bioinformática. O primeiro consiste em um *framework* adaptativo, ou seja, para cada novo projeto, o *framework* deve ser alterado de acordo com as especificações do projeto. O segundo é formado por componentes básicos e, para cada novo projeto, deve se utilizar uma série de componentes específicos, facilitando a configuração. A arquitetura do modelo possui vários módulos: Interface do usuário, mecânica do *workflow*, design do *workflow*, aplicações que solicitam serviço, serviço de descoberta, dentre outros. O reuso de componentes e softwares pôde ter seu uso e implementação observados através da estruturação dos *workflows* e *frameworks* gerenciadores de *workflows*.

Por fim, foi estudado o artigo de Ogasawara et al. 2011. O artigo fala a respeito de um modelo algébrico de execução de atividades paralelas em um *workflow* científico através de um escalonador personalizado, capaz de gerar uma estratégia de execução de acordo com os parâmetros definidos. Quatro dessa estratégias foram estudadas e analisadas, e não se pôde definir uma estratégia específica como a melhor entre as outras, pois o comportamento de cada execução varia em *workflows* distintos, ilustrando o quão difícil é, para os cientistas, determinar uma estratégia eficiente de execução paralela. As estratégias analisadas para operações de dados são divididas entre dinâmicas e estáticas, e as dinâmicas se sobressaem em performance e redução de tempo em relação às estáticas. O fragmento do *workflow* é alocado previamente no modelo estático, e sob demanda, no

dinâmico, o que interfere diretamente na eficiência de ambos. Analisando os casos de estudos realizados, a diferença na eficiência pode divergir de acordo com a informação processada no workflow: o tempo de execução pode até ser estimado, mas é imensurável na prática.

A relação entre reuso de dados e *workflows* é que o processo de funcionamento de um módulo (*software*), imerso em um *workflow*, pode requisitar, como entrada, um arquivo simples ou gerado por um processamento prévio. O reuso de dados consiste em reutilizar informação durante um fluxo de processamentos, por necessidade do próprio funcionamento do fluxo, ou por questões diferentes, como melhoria no desempenho, melhoria na segurança, etc. A variação ocorre de acordo com necessidade que se tem de reutilizar dados. No caso do MHOLline, o reuso é necessário para garantir a recuperação daquilo que, por padrão, era deletado por invalidez (deleção necessária, de acordo com o problema abordado). Em outro tipo de situação, por exemplo, o reuso de dados é garantia de economia de recursos ou de tempo em um processo de produção.

Como um módulo pertencente a um *workflow* pode necessitar um arquivo de entrada, ele também pode necessitar que seja reutilizado dados processados em etapas anteriores, frisando a relação entre os termos estudados e a importância da proveniência dos dados, considerando um *workflow* científico da área da bioinformática a ser estudado, no caso, o MHOLline.

3 MHOLline

O MHOLline é uma plataforma criada por vários desenvolvedores da área de bioinformática cuja finalidade é a obtenção rápida e precisa de modelos tridimensionais (3D) de proteínas através de métodos computacionais, auxiliando todos os pesquisadores e biólogos dessa área. A plataforma se mostra necessária uma vez que a escolha da determinação experimental (método não computacional) como método de predição de estrutura proteica, ainda que seja a melhor maneira de se obter esse resultado, não é suficiente, devido à sua lentidão e, em alguns casos, por não conseguir alcançar resultado algum (Capriles et al. 2010).

3.1 Bases teóricas

Uma proteína pode ser modelada utilizando alguns métodos, que dependem ou não de estruturas moldes. Exemplos de como a modelagem comparativa e predição de enovelamento (*threading*), que dependem de estrutura molde, ou predição por primeiros princípios (*ab initio*), também chamado de "livre de molde", são métodos computacionais para predição de estruturas 3D de proteínas. Outros métodos, como difração de Raios X (RX) e Ressonância Magnética Nuclear (RMN), são métodos experimentais usados para determinar a estrutura 3D de proteínas que podem ser usadas como moldes nos métodos computacionais.(CAPRILES et al. 2014)

O MHOLline utiliza a modelagem comparativa para a obtenção de modelos 3D de proteínas. Essa técnica consiste em construir um modelo de uma sequência de proteínas a partir de um molde "estruturalmente semelhante", dito homólogo. Uma identificação de proteína de referência é feita após um alinhamento entre a estrutura alvo e as proteínas resolvidas experimentalmente e depositadas em bancos de dados, e essas serão referenciadas não apenas pela similaridade, mas pelo seu grau de identidade, o tamanho da cobertura da sequência e a quantidade de *gaps*. Após esse processo, a decisão de quais moldes serão usados leva em consideração fatores como:

(i) A família de proteínas à qual o molde e a proteína alvo pertencem;

(ii) Quantidade ou semelhanças nas funções correlacionadas entre elas;

(iii) Se as estruturas das moléculas candidatas a molde possuem alto grau de identidade (Capriles et al. 2010).

Uma vez decidido quais moldes serão utilizados, um alinhamento global é feito entre toda a sequência da estrutura alvo e dos moldes, gerando modelos 3D da proteína escolhida. Em alguns casos, não é possível encontrar um molde satisfatório para a proteína alvo como um todo, portanto, para esta parte da proteína outras abordagens devem ser feitas, como o uso de outras técnicas de modelagem de proteínas. Ao final da montagem da proteína, é necessário que se faça a validação do modelo, através de *softwares* avaliadores como o Procheck, Molprobity, Verify3D, dentre outros. Após esta etapa de validação do modelo, caso os resultados estejam insatisfatórios, deve-se retornar ao passo de seleção e alinhamento dos moldes com o modelo para que haja uma melhoria no modelo final.(CAPRILES et al. 2014)

No momento da busca por moldes, se não for encontrado alguma proteína que possua um grau de identidade maior do que 25% com a estrutura alvo, o sistema não modelará essas proteínas, apontando no máximo possíveis moldes para serem usados em técnicas como busca de enovelamento.

A escolha do modelo final pode ser feita através de funções filtros ou por inspeção manual. O modelo será analisado quanto a sua qualidade energética e conformacional, analisando a estrutura quanto as violações espaciais que um resíduo apresenta. Caso o modelo não obtenha um nível satisfatório de qualidade, opta-se por realizar o refinamento do modelo, por exemplo, impondo restrições ao algoritmo de construção do modelo.(CAPRILES et al. 2014)

3.2 Implementação

De acordo com o Manual Técnico do MHOLline 2.0 (Rossi et al. 2015), a plataforma MHOLline pode ser subdividida em três partes:

- MHOLweb: parte responsável pelo gerenciamento da plataforma através de interface

web, feito a partir das linguagens PHP e Javascript, além de HTML e CSS;

- MHOLcore: parte responsável pelo processamento de dados, gerando resultados e os armazenando, tudo feito em um servidor MySQL. Utiliza-se Perl, ShellScript, C, Python e JavaScript;

- MHOLdb: parte responsável pelo controle do banco de dados usado no sistema. Utiliza o SGBD MySQL.

3.3 O *workflow*

A plataforma MHOLline segue um fluxo de trabalho (ver figura 3.3). O programa recebe um arquivo .FASTA como entrada, descrito na seção 1.2 e realiza uma sequência de processamentos em módulos distintos, de tal forma que a saída gerada em um módulo pode servir de entrada para o outro (exemplo do BLAST em diante). Para identificar as estruturas através de *templates*, o MHOLline utiliza o algoritmo **BLAST**, que procura estruturas 3D no Protein Data Bank (PDB). (Capriles et al. 2010)

O Protein Data Bank (http://www.wwpdb.org) é um repositório específico da discretização feita das informações acerca de uma proteína através de modelagens experimentais, que servem de base para alinhadores identificarem estruturas. Atualmente, O PDB conta com mais de 140 mil estruturas homologadas (ver figura 3.5). Seu formato de arquivo (.PDB) é padronizado desde 2012, e consiste em um texto onde cada linha representa um *record*, estruturadas de forma organizada para conseguir descrever uma proteína. Os tipos de *records* (gravações) são:

- ATOM: *record* de coordenada atômica (X,Y,Z ortogonal Å) para átomos em resíduos padronizados (aminoácidos e ácidos nucleicos);

- HETATM: *record* de coordenada atômica (X,Y,Z ortogonal Å) para átomos em resíduos não padronizados (inibidores, cofatores, íons e solventes). A diferença funcional entre ele e o ATOM é que os resíduos do HETATM, por padrão, não são conectados a outros resíduos;

- TER: delimitador do término de uma corrente de resíduos, prevenindo que o restante do arquivo seja ligado nela;

- HELIX: indica a localização e o tipo da hélice da estrutura;

- SHEET: indica a localização e o sentido de uma vertente da estrutura (em relação à vertente anterior estabelecida);

- SSBOND: define as ligações covalente disulfidicas entre os resíduos de cysteína.

A seguir, um exemplo de arquivo .PDB. A linha CONECT realiza semelhante função ao SSBOND, e tal imprevisto se deve às diversas padronizações existentes.

```
COMPND    UNNAMED
AUTHOR    GENERATED BY EXAMPLE
ATOM      1  N   ALA A   1      0.000   0.000   0.000  1.00  0.00           N
ATOM      2  CA  ALA A   1      1.456   0.000   0.000  1.00  0.00           C
ATOM      3  C   ALA A   1      1.930   0.000   1.463  1.00  0.00           C
ATOM      4  O   ALA A   1      1.160   0.000   2.421  1.00  0.00           O
ATOM      5  HN  ALA A   1     -0.495   0.091   0.883  1.00  0.00           H
ATOM      6  HA  ALA A   1      1.799  -0.926  -0.476  1.00  0.00           H
ATOM      7  CB  ALA A   1      2.010   1.208  -0.746  1.00  0.00           C
ATOM      8  1HB ALA A   1      1.657   1.230  -1.782  1.00  0.00           H
ATOM      9  2HB ALA A   1      1.695   2.143  -0.268  1.00  0.00           H
CONECT    1   2   3   4   5
CONECT    2   6   7   8   9
TER
```

Figura 3.1: Conteúdo de um arquivo de formato *.PDB

Após seleção das estruturas no PDB, o algoritmo **BATS** (*Blast Automatic Targeting for Structures*) é utilizado para dividir as sequências em grupos (G0, G1, G2 e G3). As estruturas de G0 são as que não puderam ser alinhadas, e as de G1 são as que possuem baixo valor de identidade e não podem ser modeladas, mas podem ser enviadas para plataformas de predição *ab intio*, por exemplo. Entretanto, as estruturas de G3 possuem um baixo valor de identidade, mas por serem estruturas conhecidas, podem ser resolvidas em alguma outra plataforma que realiza métodos não comparativos (*threading ou ab initio*). Por fim, as estruturas de G2 são as que possuem alta taxa de identidade, e são as escolhidas para continuarem no processo de modelagem comparativa. (Capriles et al. 2010)

Após essa seleção, são aplicados filtros nas estruturas selecionadas para que se-

jam reavaliadas por nível qualitativo, determinados pela ferramenta **Filters**. Após essa avaliação, o **ECNGet** possui a função de classificar as enzimas pelo número EC (Enzyme Commission). Essa classificação é importante para entendermos o funcionamento de uma proteína com função de enzima. Paralelamente, a ferramenta **Modeller** realiza a criação da estrutura tridimensional da proteína alvo com os melhores moldes selecionados pelo BATS (e Filters). Uma vez construídos os modelos, o programa **Procheck** os avalia quanto suas qualidades estereoquímicas. Adicionalmente, a plataforma conta com o programa **HMMTOP**, que identifica regiões de transmembranas. Seu uso é complementar, não interfere nos resultados do *workflow*. (Capriles et al. 2010) Há outros módulos complementares que, apesar de diferentes, realizam funções semelhantes, como exemplo, a relação entre HMMTOP e o TMHMM.

A figura 3.2 representa a exibição do resultado de um *job* finalizado, denominado JOBNOVO3. Os ícones coloridos são os módulos executados e finalizados durante o fluxo, e cada resultado individual pode ser acessado separadamente. A legenda dos ícones está descrita na imagem, bem como na página *web* da plataforma.

Figura 3.2: Exibição de um job na página de resultados.

Referente ao MHOLline, esse trabalho propõe a criação de uma ferramenta capaz de reinserir, no banco de dados do *workflow*, a copia exata de um *job* (em formato .TAR.GZ), que hipoteticamente já deixou de existir por ter seu prazo de validade expirado. Assim, o usuário poderá salvar seu trabalho e, a qualquer momento, inseri-lo novamente no MHOLline para realizar os refinamentos necessários ou usufruir de qualquer ferramenta que venha a ser implantada ao *workflow*, de uso pós-geração do *job*.

Para cada módulo de execução do *workflow*, um resultado é gerado e pode servir de entrada para a execução de outro módulo, dado a necessidade determinada pelo usuário e os módulos de execução selecionados para processamento. Os resultados de um módulo específico são salvos no banco de dados, em uma tabela referente à esse módulo, e também em forma de arquivo, a ser interpretado por algum programa ou legível pelo usuário. A figura 3.3 representa o *Workflow* em sua versão atual, exemplificando os módulos e o fluxo ao qual estão inseridos. (Rossi 2017)

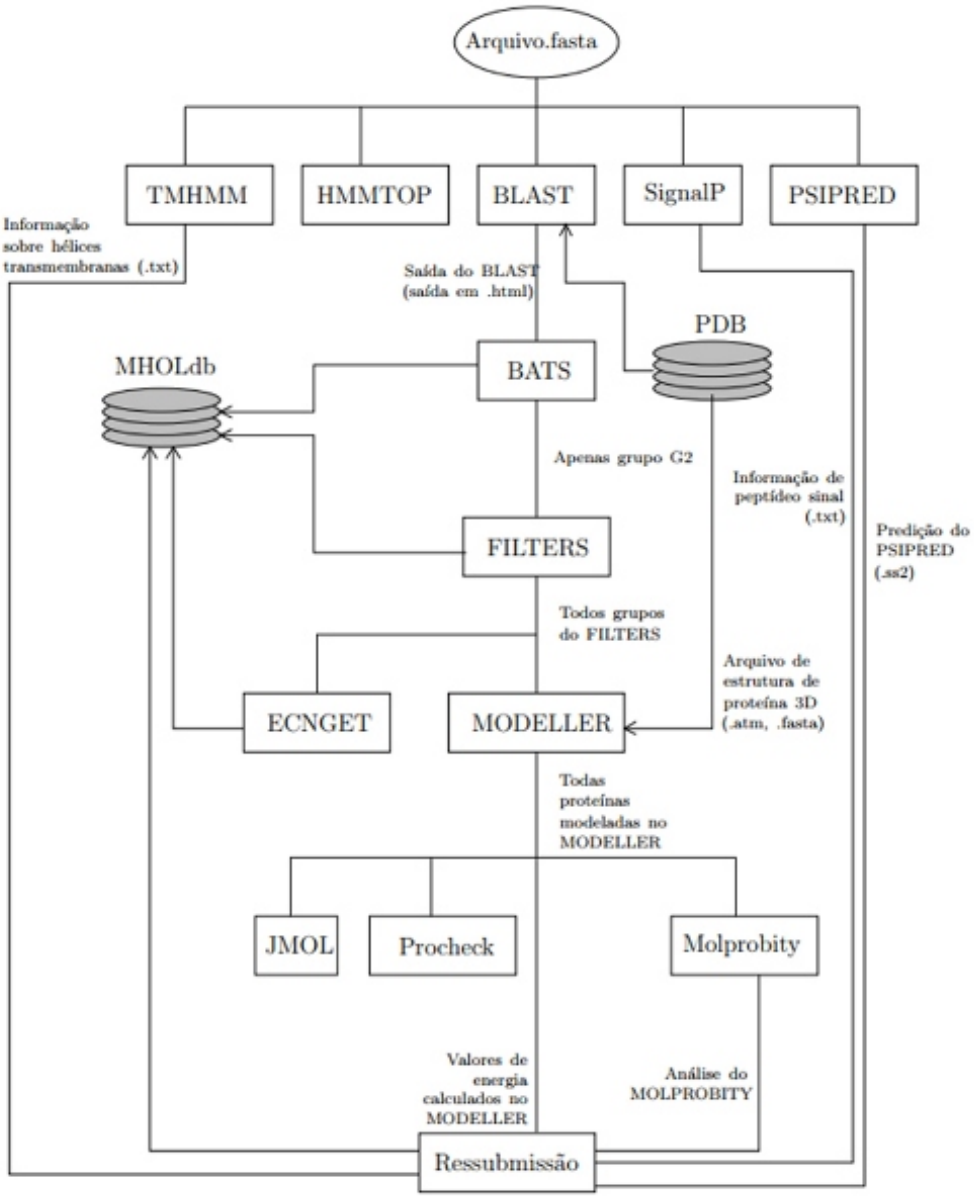

Figura 3.3: Fluxo de dados do MHOLline V2.0

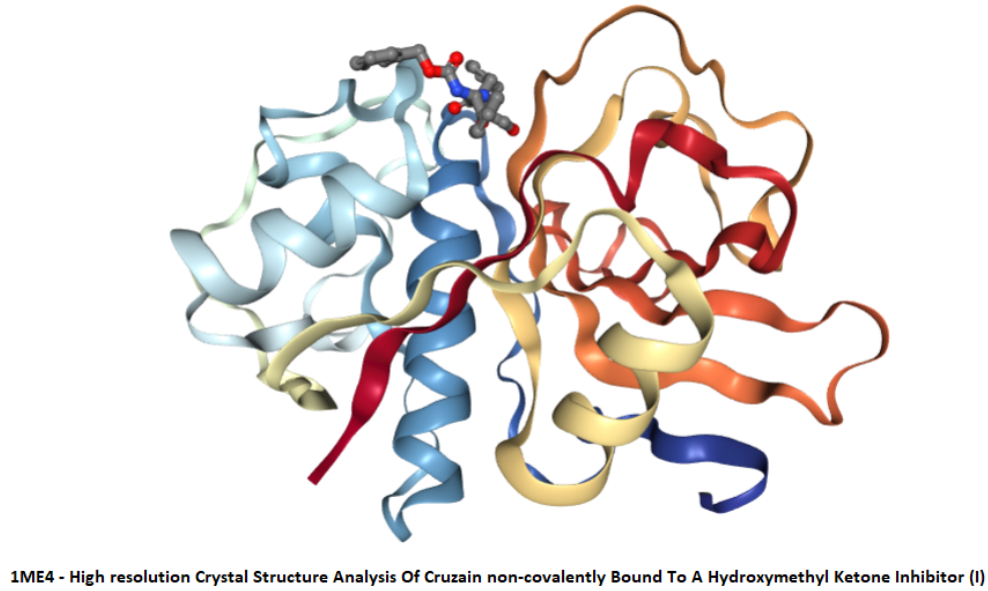

Figura 3.4: Exibição do modelo 3D da proteína 1ME4.

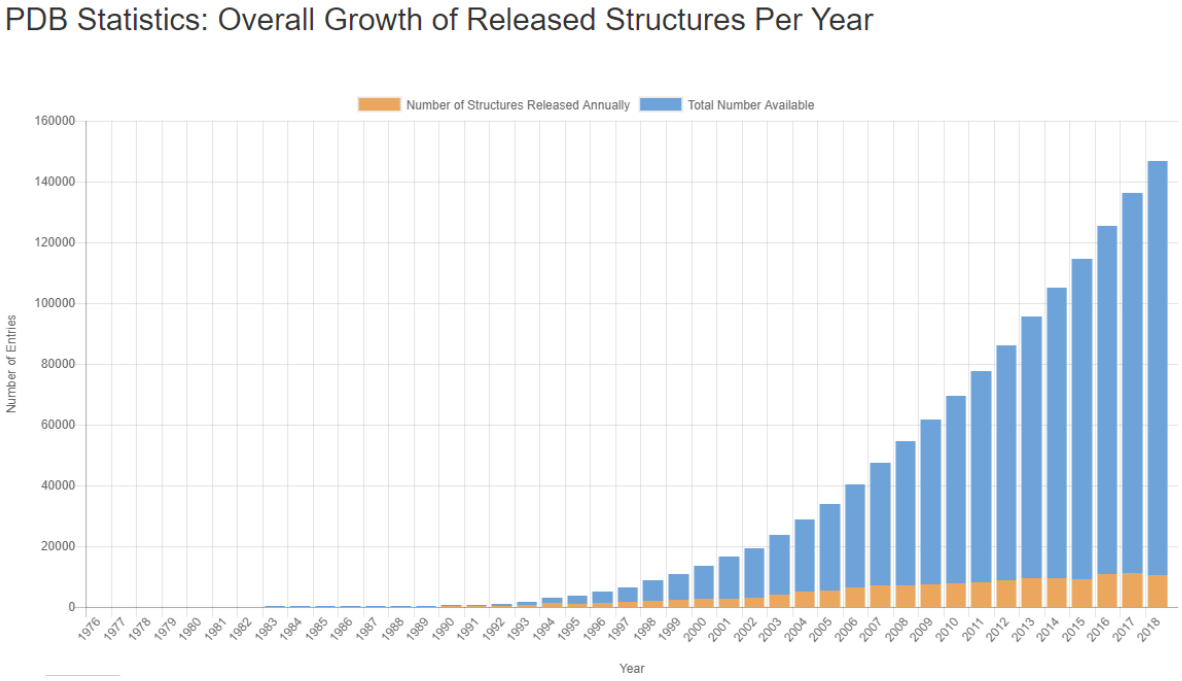

Figura 3.5: Relação Entradas x Anos no PDB.

4 Material e Métodos

4.1 Desenvolvimento

O desenvolvimento do trabalho foi realizado em um servidor local de testes (Ubuntu 14.04), configurado de forma semelhante ao servidor de produção do MHOLline no LNCC. Durante esse período, o conteúdo do *workflow* foi analisado, tanto os códigos programados quanto as configurações predefinidas nos arquivos do SGBD e do sistema operacional. Foi necessário a execução fiel do Manual Técnico do MHOLline (Rossi et al. 2015), principalmente na instalação de bibliotecas e pacotes, necessários para execução dos módulos do MHOLline.

Para a criação do sistema de reuso de dados (objetivo principal deste trabalho), foi necessário uma readaptação na forma que o *job* é gerado inserindo nele, anteriormente à etapa de geração do TAR.GZ, metadados a fim de auxiliar a ferramenta em seu processamento. Sem esses metadados, o servidor não consegue recuperar as informações necessárias no momento de exibição dos resultados do *job*, restando apenas os arquivos isolados no diretório ao qual ele é armazenado por padrão. Isso se faz necessário porque as tuplas referentes aos IDs dos *jobs* são deletadas, juntamente com os arquivos. Assim, foi desenvolvido em PERL (no arquivo mholline.pl), durante o processo de criação do *job*, todo o procedimento para realizar as instruções de SELECT no MHOLdb de todos os dados necessários para recuperação e registro, gerando o arquivo *metadata* referente à tabela contendo os valores das tuplas selecionadas.

O padrão adotado para os arquivos metadados foi estabelecido de forma semelhante ao CSV, ou seja:

- A tabela no BD a ser manipulada está descrita no nome do arquivo metadado;
- Vírgulas são usadas como delimitadores;
- Primeira delimitação: id da tabela;

- Segunda até N-ésima delimitação: valor1, valor2, [...], valor n;
- Quebra de linha com conteúdo seguinte representa uma nova coluna a ser inserida.
- Quebra de linha sem conteúdo seguinte representa o fim do arquivo.

Para cada tabela cujas informações necessitam ser recuperadas, foi criado um arquivo metadado diferente. Como há uma ordenação necessária dos valores a serem inseridos e extraídos nas tabelas do MHOLdb, os metadados também foram desenvolvidos de forma a garantir essa ordem. Durante a criação do metadado referente a um *job*, as tuplas selecionadas são discretas e inseridas de forma ordenada no arquivo, e durante a sua replicação no MHOLdb, as informações também são reinseridas em ordem.

Além disso, durante a exibição dos resultados dos *jobs*, uma função desenvolvida (SHAVerify) entra em funcionamento sempre que o *job* estiver pronto para *download*. Tal função é responsável por inserir, em uma nova tabela criada (upload_targz), o *hash* (SHA512) gerado pelo arquivo .TAR.GZ, garantindo que apenas esse arquivo, exatamente da forma como ele está, sem qualquer modificação, possa ser inserido novamente no sistema pela ferramenta de upload, para que o *job* seja revalidado. O SHA512 (*Secure Hash Algorithm*512) é uma vertente do algoritmo SHA-2, desenvolvido pela NSA (*National Security Agency*), e pode ser utilizado para, por exemplo e não somente a isso, garantir a autenticidade de um dado. A versão 512 foi escolhida devido ao seu grau de segurança e sua menor taxa de colisões, em comparação com as demais versões.

A figura 4.1 foi retirada do artigo "Throughput and Efficiency Analysis of Unrolled Hardware Architectures for the SHA-512 Hash Algorithm" (Algredo-Badillo et al. 2012) e retrata as características do SHA-1 e da família de algoritmos do SHA-2 (256, 384 e 512), o que irá determinar a eficiência de cada vertente desse algoritmo.

Algorithm	SHA-1	SHA-256	SHA-384	SHA-512
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block size	512	512	1024	1024
Word size	32	32	64	64
Message digest size	160	256	384	512
Security	80	128	192	256

Figura 4.1: Tabela de características específicas de cada algoritmo SHA

A ferramenta de reuso de dados foi desenvolvida para que o o *job* possa ser revalidado por mais 15 dias no sistema, sendo possível utilizá-lo em alguma ferramenta na etapa posterior à execução dos módulos principais do MHOLline. O arquivo é submetido a um *upload* na página de reuso de dados, que está acessível apenas para usuários cadastrados, e deve estar em formáto válido (.TAR.GZ). Tal verificação é desnecessária, uma vez que o *hash* garante a unicidade do arquivo, no entanto, possui a finalidade de evitar consultas desnecessárias ao BD.

Na sequência, é comparado o *hash* salvo durante a formação do *job* no processamento do arquivo .FASTA com o código que foi gerado pelo *upload* do *job*, e quando os códigos são iguais, significa que o *job*, salvo em arquivo .TAR.GZ, inserido na ferramenta de reuso de dados, é o mesmo arquivo que, dali, foi copiado previamente (garantindo a unicidade). Esse procedimento, denominado *checksum*, evita inserções fraudulentas e maliciosas, contribuindo de forma a minimizar as vulnerabilidades de se permitir o envio de um arquivo do usuário para o servidor. O sistema também trata o caso do usuário submeter um arquivo idêntico ao que já está nele e ainda não foi deletado, reportando um erro.

Após o processo de validação do arquivo inserido, os metadados são executados por um analisador de arquivos desenvolvido também para essa finalidade, que irá tratar de guardar as informações novamente no banco de dados e modificar a data de validade do *job* para mais 15 dias, a partir do momento que foi inserido. Esse analisador é responsável por realizar, no MHOLdb, as inserções corretas e ordenadas das informações contidas nos metadados (que são os resultados ordenados de *SELECTS* referentes às tabelas a serem

recuperadas).

Vale ressaltar que configurações do próprio PHP.ini no Ubuntu foram alteradas do seu padrão, garantindo uma banda maior para o envio do arquivo a ser ressubmetido, e que novos alertas de erro também foram estabelecidos. A representação da ferramenta de reuso de dados pode ser visualizada através do modelo criado (figura 4.2).

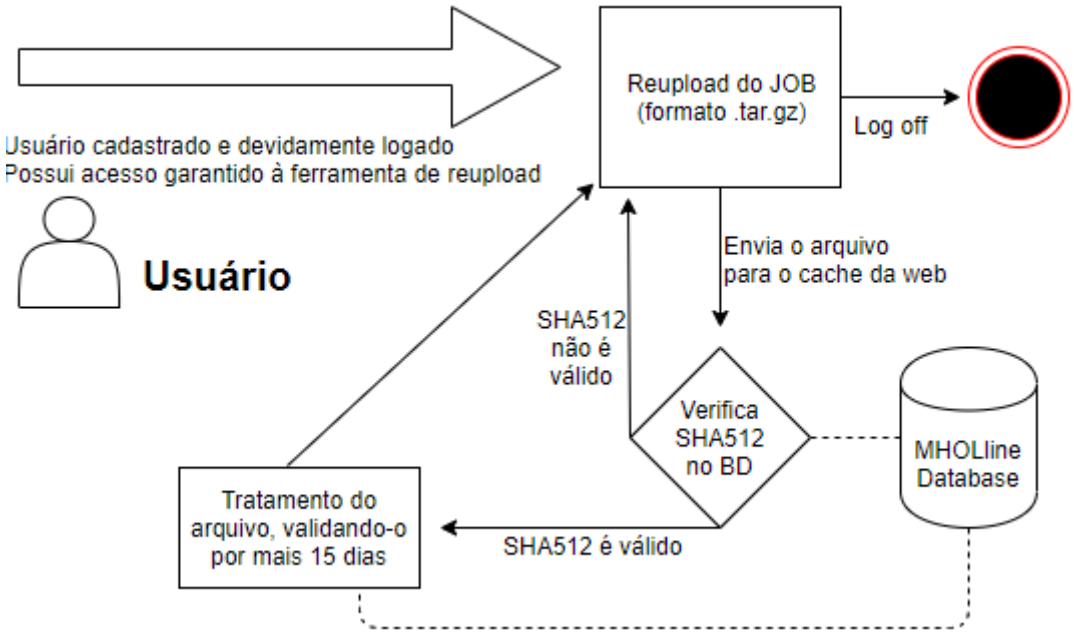

Figura 4.2: Modelo representativo da ferramenta de reuso de dados

4.2 Algoritmos

Durante a fase de desenvolvimento, novas páginas e funções foram criadas e outras modificadas. A seguir, serão listadas essas páginas e o que foi realizado de modificação em cada uma delas, de forma abstrata.

4.2.1 Módulo de execução do MHOLline

1. Acrescentado no código (etapa de formação de arquivos e diretórios):

- Definição dos metadados para cada tabela a ser recuperada;

- Definição da permissão para garantir que os metadados possam ser sobrescritos pela função.

2. Acrescentado no código (etapa de finalização do *job*):

- Acessa o banco de dados;

- Define uma query selecionando as colunas de uma tabela X, que contém informações necessárias para descrever algum módulo de execução;

- Escreve os resultados obtidos pela query no metadado referente à tabela X.

Define-se X como as colunas deletadas de uma tabela do MHOLdb durante o processo de deleção de um *job* (as informações a serem recuperadas).

4.2.2 Página de exibição de resultados dos *jobs*

1. Modificado no código (etapa de liberação do download):

- Verifica se o download dos *jobs* de um usuário está disponível;

- Se estiver, executar a função "SHAVerify".

2. Acrescentada a função "SHAVerify":

- Gera uma *hash* do arquivo .TAR.GZ em questão;

- Verifica a existência desse *hash* na tabela criada "upload_targz, e caso o hash de um mesmo *job* for modificado, acrescenta o novo *hash* na tabela (pois o *job* foi modificado em alguma etapa posterior à finalização da execução dos módulos iniciais).

4.2.3 Script da ferramenta de reuso de dados

1. Na página criada e denominada "Reupload", o script é executado pela action do botão de envio do arquivo .TAR.GZ:

- Verifica se o *inputbox* está preenchido, ou reporta erro e encerra o script;

- Verifica se o arquivo está no formato correto (TAR.GZ), ou reporta erro e encerra o script;

- Verifica se ainda há um *job* ativo de mesmo nome do arquivo enviado, encerrando o algoritmo e alertando o usuário que o envio desse *job* é desnecessário, caso já exista um igual no sistema;

- Verifica se o arquivo gera algum *hash* contido na tabela upload_targz;

- Se não encontrar *hash* igual, reportar erro e encerra o script;

- Se encontrar, ele busca os metadados no arquivo recém enviado, os insere em suas respectivas tabelas e atualiza a validade do arquivo para mais 15 dias, revalidando o

job.

5 Resultados e Discussão

5.1 Apresentação do resultado

A página de reuso de dados foi criada de forma a atender os objetivos do trabalho (ver figura 5.1) e inserida no *workflow*. É garantido acesso à ferramenta de *reupload* apenas usuários cadastrados. O algoritmo se mostrou eficiente nos testes realizados em servidor local, e o sistema foi forçado a realizar vários *uploads* diferentes para testar a funcionalidade das ferramentas de verificação, que se mostraram consistentes em seus resultados: foram submetidos arquivos de formatos diferentes para garantir que a ferramenta valide apenas arquivos com a extensão .TAR.GZ. e garanta a unicidade pelo *hash*.

Figura 5.1: Página de reupload executada no servidor de testes

O arquivo é inserido corretamente no seu diretório específico, e a funcionalidade dos algoritmos de criptografia SHA512 também foi garantida, uma vez que o mesmo

arquivo (ou cópias exatas dele, sem qualquer modificação) gera(m) o mesmo *hash*. Bastou modificar o nome do arquivo compactado, ou qualquer conteúdo dentro dele, que o *hash* gerado modificou, garantindo a unicidade do conteúdo inserido e excluindo possibilidades de inserções mal-intencionadas, com um arquivo (.TAR.GZ) modificado. Como o arquivo somente é manipulado após a etapa de verificação do *hash*, garante-se que, pela ferramenta de reuso de dados, nenhum tipo de arquivo (.TAR.GZ) que já não tenha sido criado pelo MHOLline será ressubmetido.

Os metadados foram analisado e as informações pertinentes foram corretamente inseridas no banco de dados do MHOLline, revalidando o *job* por mais 15 dias. Dessa forma, qualquer problema proveniente de limitação de espaço em disco rígido pode ser parcialmente solucionada com essa ferramenta, garantindo o reuso dos dados cujas origens foram provadas. O usuário ativo no sistema poderá reutilizar o seu projeto, desde que o SHA512 seja validado, ou seja, que os arquivos baixados e re-enviados sejam os mesmos. No entanto, caso o usuário perca a sua conta, o *job* não pode ser reaproveitado em outra conta, pois a análise e o processamento de um arquivo ficam atrelados ao ID do usuário, e uma mudança na forma como a informação é processada e armazenada necessita de ser estudada para tratar esse problema, a fim de causar o mínimo de efeito negativo no o sistema. Tomemos como exemplo a seguinte inserção de dados no arquivo metadado, referente a tabela “upload_fasta”:

- 198,897,“Nomearquivo”,0,1,1,“ACTGTGCTTATATCTA”
- 199,897,“Nomearquivo”,0,1,1,“ACTACTCTAGGAATCA”
- 200,897,“Nomearquivo”,0,1,1,“TCATTGCTATAGCTC”
- 201,897,“Nomearquivo”,0,1,1,“ACTGTGCTTATATCTA”
- 202,897,“Nomearquivo”,0,1,1,“ACTGTGCTACTGGACA”
- 203,897,“Nomearquivo2”,0,1,1,“ACTGGTTCCATATCTA”
- 204,897,“Nomearquivo2”,0,1,1,“AGTCTGCTTCTAGCTA”

O campo 897, constantemente presente na segunda coluna, representa o identificador do usuário, exemplificando a forte dependência entre a visualização dos *jobs* e o usuário ao qual pertence.

A ferramenta de reuso de dados foi desenvolvida, os dados foram validados durante o processamento do upload, garantindo sua unicidade, e a ferramenta foi integrada à versão atual de forma a minimizar possíveis efeitos negativos provenientes dessa integração, atingindo, respectivamente, os três objetivos especificados no trabalho.

5.2 Validação do resultado

Durante o processo de desenvolvimento, testes foram necessários para validar a robustez do *workflow*, ou seja, garantir o pleno funcionamento do processo de execução do arquivo .FASTA e da exibição de resultados dos *jobs* gerados. Adaptações para esse trabalho foram feitas tanto no arquivo responsável pela execução dos módulos, quanto na estrutura web, responsável por fazer a mediação entre o usuário e o sistema.

Primeiramente, foi testado se a ferramenta de reuso de dados realizava o *upload* de forma correta (transferir o arquivo para o seu diretório específico do sistema), verificando as condições necessárias para isso. A condição de verificar o *hash* foi a última a ser testada, pois necessitava de uma estrutura maior de verificação (uma nova tabela, funções responsáveis por manipular dados entre o BD e os arquivos do *job*).

O sistema só reconhece um *job* (e os seus respectivos resultados e status) quando as informações referentes a ele puderem ser descritas pelas tabelas do MHOLdb, ou seja, foi verificado que os arquivos resultados do *job* de nada valem se estiverem soltos no sistema, sem que possam ser descritos por essas tabelas. A partir desse teste, constatou-se a necessidade dos arquivos resubmetidos conterem as informações necessárias a serem recuperadas pelo MHOLdb. Isso foi solucionado com um manipulador de arquivo, também desenvolvido e testado, que recuperasse as informações contidas nas tabelas do MHOLdb, referentes ao *job* do usuário, e salvasse em um metadado, no próprio arquivo final (.TAR.GZ).

Os manipuladores de arquivos foram validados analisando se o que era escrito no metadado (gerado para esse propósito na etapa de definição de pastas e arquivos-base)

era exatamente as informações contidas em uma tabela específica a ser recuperada futuramente. A outra parte da manipulação (responsável por inserir de volta, no MHOLdb, a informação recuperada) foi validada por inserir de volta, com sucesso, as tuplas que haviam sido deletadas pelo sistema, tornando o *job* visível na página "results.php" (figura 3.2).

A validação das funções criadas para tratar a situação dos *hashs* foi realizada, tanto modificando o nome do arquivo, quanto modificando algum conteúdo dentro do arquivo (especificamente, uma letra foi removida de um texto resultado de um alinhamento), e em ambos os casos, o *hash* gerado era diferente. A validação da identidade entre *hahs* (*checksum*) acaba por ser o próprio funcionamento correto da ferramenta, permitindo a visualização de um *job* que havia sido deletado e atendendo os objetivos do trabalho. Ela foi realizada com a ferramenta em seus estágios finais, submetendo um arquivo .FASTA e selecionando módulos arbitrários para serem executados.

Após aguardar o término do *workflow*, um download do *job* é realizado (cópia do arquivo, do *server side* para o *client side*). A deleção do arquivo no sistema é feita e esse mesmo arquivo, salvo em qualquer outro diretório, é submetido de volta ao MHOLline pela ferramenta de reuso de dados com sucesso (*client side* para *server side*) e, em contrapartida, um outro arquivo, de conteúdo modificado (caracter removido de um dos arquivos-texto), é enviado e rejeitado por não conseguir encontrar um *hash* compatível. Isso se deve ao efeito Avalanche (Moldovyan e Moldovyan 2007), onde uma simples modificação em um BIT de um arquivo desencadeia um novo *hash* para esse arquivo, completamente distinto do antigo. Nos casos em que o *job* não era deletado, a tentativa de upload (independente da cópia ser ou não a exata) também fracassava, de acordo com o que foi programado.

5.3 Discussão

O *workflow* MHOLline possui uma forte dependência entre o MHOLcore, o MHOLweb e o MHOLdB. Uma hipotética falha ocasionada da interação do Web ou do Core com o DB (insuficiência das informações salvas no BD referente a um *job*) pode gerar um *job* falho que, ao ser explorado em outras ferramentas de uso posterior à conclusão dos módulos de execução do FASTA, procederá com mais erros e de forma inconsistente. Foi necessário rever, para cada simples modificação na estrutura do código do MHOLweb ou do MHOLline, as consequências geradas na execução do fluxo e da amostragem dos resultados.

De acordo com os materiais de estudo desse trabalho, as informações são mais bem aproveitadas se estiverem dispostas para uso de forma organizada e interpretável, então, no caso de um *workflow*, é extremamente recomendado possuir um modelo descritivo de suas funcionalidades e limitações, não só para o uso rotineiro da ferramenta, mas para nortear-se durante o desenvolvimento de novos módulos. A ferramenta de reuso de dados exerce o papel de recuperar tudo aquilo que é necessário para exibição e posterior validação do *job*, mas sem uma descrição clara do *workflow* ou sem garantia do entendimento da execução dos módulos, bem como suas respectivas entradas e saídas, o desenvolvedor pode encontrar dificuldades para recuperar todas as informações que forem necessárias.

Além de haver apenas essa interação entre os módulos, o MHOLline possui uma forte dependência do sistema operacional ao qual ele está funcionando, no caso, o Ubuntu 14.04. Durante o processamento de um *job*, diversos comandos em *ShellScript* são executados para definição de diretórios, arquivos e permissões. A execução correta das aplicações do MHOLline também dependem que uma lista vasta de bibliotecas esteja instalada e que essas bibliotecas estejam atuantes e acessíveis no sistema. Vale salientar que a falta de alguma dessas bibliotecas pode acarretar em falhas nos módulos e o *workflow* não funcionará corretamente, ou não funcionará.

6 Conclusões e Perspectivas Futuras

A computação contribui de forma a facilitar o processo de discretização das informações obtidas por modelagens experimentais, armazená-las em bancos de dados e, não menos importante, interpretá-las e processa-las da maneira que mais convir ao usuário, sob a perspectiva analítica de quem compreende a biologia profundamente.

Na situação do *workflow* MHOLline, foi necessário gerar uma informação no processo de formação do *job* que servia de base para retro-alimentar o banco de dados, caso o arquivo seja submetido à ferramenta de reuso de dados e revalidado por um novo período. Portanto, durante esse processo, não há definição ou criação de nenhuma nova informação relevante aos resultados do *job* em si, apenas para uso e manutenção do *workflow*, garantindo que nenhum resultado seja modificado nesse processo.

Assegurando essa integridade do arquivo, e considerando um caso hipotético de um novo módulo ser adicionado durante o processo de execução do arquivo .FASTA, bastaria recuperar toda a informação que fosse deletada durante o processo de deleção do *job*, e uma adaptação no código seria necessária para assegurar que todas as novas informações sejam inseridas no banco de dados.

Cada módulo age de forma independente (com exceção dos filtros, que necessitam como entrada, a saída da execução de outro módulo), mas é exigido uma sincronia entre o que esta sendo desenvolvido e o que já é funcional no sistema, e caso o fluxo não seja seguido à risca, qualquer arquivo resultado de um suposto módulo novo não será incluído na elaboração do .TAR.GZ, permanecendo somente no sistema. Se essas informações não puderem ser descritas, recuperadas ou se não estiverem atualizadas na pasta antes do processo de criação do .TAR.GZ e da geração do *hash* de validade, haverá divergência entre o arquivo do *job* original que havia no sistema e do disponível para download. Resumidamente, o usuário teria acesso ao resultado final, mas só iria ter a oportunidade de recuperar o arquivo do resultado gerado precipitadamente, o que é uma falha.

Para garantir o pleno funcionamento do verificador de *hash*, sempre que o *job* concluído for modificado, um novo *hash* é gerado, permitindo ao usuário restaurar o seu

job a partir de qualquer ponto durante as etapas de refinamento.

A ferramenta de reuso de dados trabalha com as seleções e inserções corretas, dado a atual estrutura do *workflow*. Qualquer mudança na forma como o *job* é gerado, lido e armazenado, deve ser levada em consideração. Se novas informações necessitam de ser recuperadas, a ferramenta deve ser adaptada a tratar esse novo tipo de situação. Caso contrário, a visualização do *job* pode ocorrer de forma divergente do arquivo original, ou pode nem mesmo ocorrer.

Com a conclusão desse trabalho, é estimado que em breve, a ferramenta de reuso de dados seja capaz de criptografar o arquivo metadado gerado para que oculte toda a informação explícita, desnecessária ao conhecimento do usuário comum, e consiga interpretá-la no momento de revalidação do *job*. Além disso, levantou-se a questão da possibilidade de usuários distintos poderem revalidar um mesmo *job*, se a modificação necessária para que isso aconteça é viável ou não, e se compromete a segurança e a robustez do sistema em sua atual versão.

Existem outros complementos para o *workflow* que ainda estão sendo desenvolvidos na segunda versão do MHOLline. Além desses complementos, os módulos estão sendo atualizados, a segurança do sistema está sendo melhorada e uma nova interface está sendo criada, melhorando a relação entre a área da bioinformática e o usuário ainda não familiarizado com essa área (REIS et al. 2016).

Bibliografia

Algredo-Badillo et al. 2012 ALGREDO-BADILLO, I. et al. Throughput and efficiency analysis of unrolled hardware architectures for the sha-512 hash algorithm. In: IEEE. *VLSI (ISVLSI), 2012 IEEE Computer Society Annual Symposium on*. [S.l.], 2012. p. 63–68.

Barns e Bollinger 1991 BARNS, B. H.; BOLLINGER, T. B. Making reuse cost–effective. *IEEE software*, v. 8, n. 1, p. 13–24, 1991.

Boeckmann et al. 2003 BOECKMANN, B. et al. The swiss-prot protein knowledgebase and its supplement trembl in 2003. *Nucleic acids research*, Oxford University Press, v. 31, n. 1, p. 365–370, 2003.

Buneman, Khanna e Tan 2000 BUNEMAN, P.; KHANNA, S.; TAN, W.-C. Fst tcs 2000: Foundations of software technology and theoretical computer science. Springer Berlin Heidelberg, Berlin, Germany, 2000.

Capriles et al. 2010 CAPRILES, P. V. et al. 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*, BioMed Central, v. 11, n. 1, p. 610, 2010.

Capriles et al. 2010 CAPRILES, P. V. S. Z. et al. Structural modelling and comparative analysis of homologous, analogous and specific proteins from trypanosoma cruzi versus homo sapiens: putative drug targets for chagasd́isease treatment. *BMC Genomics*, v. 11, n. 1, p. 610, 2010. ISSN 1471–2164. Disponível em: ⟨http://dx.doi.org/10.1186/1471-2164-11-610⟩.

CAPRILES et al. 2014 CAPRILES, P. V. S. Z. et al. Bioinformática: Da biologia à flexibilidade molecular. SBBq, São Paulo - Brasil, 2014. Pages 148 - 171.

Deelman et al. 2009 DEELMAN, E. et al. Workflows and e-science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, Elsevier, v. 25, n. 5, p. 528–540, 2009.

Digiampietri et al. 2005 DIGIAMPIETRI, L. A. et al. A framework based on web service orchestration for bioinformatics workflow management. *Genetics and molecular research: GMR*, 2005.

Fienberg et al. 1985 FIENBERG, S. E. et al. *Sharing research data*. [S.l.]: National Academies, 1985.

Hollingsworth e Hampshire 1995 HOLLINGSWORTH, D.; HAMPSHIRE, U. Workflow management coalition: The workflow reference model. *Document Number TC00-1003*, Document Status, v. 19, 1995.

Ludäscher et al. 2006 LUDÄSCHER, B. et al. Scientific workflow management and the kepler system. *Concurrency and Computation: Practice and Experience*, Wiley Online Library, v. 18, n. 10, p. 1039–1065, 2006.

Moldovyan e Moldovyan 2007 MOLDOVYAN, N.; MOLDOVYAN, A. Innovative cryptography. Charles River Media, Thomson learning Inc, Boston - Massachusetts, 2007.

Ogasawara et al. 2011 OGASAWARA, E. et al. An algebraic approach for data-centric scientific workflows. *Proc. of VLDB Endowment*, v. 4, n. 12, p. 1328–1339, 2011.

Piwowar e Vision 2013 PIWOWAR, H. A.; VISION, T. J. Data reuse and the open data citation advantage. *PeerJ*, PeerJ Inc., v. 1, p. e175, 2013.

Prodigious brand logistics PRODIGIOUS brand logistics. Web page: ⟨http://prodigious.com/our_services/prodigiouscloud/⟩ - Accessed on 22/08/2018.

Prosdocimi et al. 2002 PROSDOCIMI, F. et al. Bioinformática: manual do usuário. *Biotecnologia Ciência e Desenvolvimento*, v. 29, p. 12–25, 2002.

Reis et al. 2016 REIS, V. et al. Xi jornada de iniciação científica e tecnológica - livro de resumos. Laboratório Nacional de Computação Científica, Petrópolis - Brasil, 2016.

REIS et al. 2016 REIS, V. et al. Xi jornada de iniciação científica e tecnológica - livro de resumos. Laboratório Nacional de Computação Científica, Petrópolis - Brasil, 2016. Page 47.

Rossi 2017 ROSSI, A. D. Refinamento manual e automático de modelos tridimensionais de proteínas para o workflow científico mholline. *Universidade Federal de Juiz de Fora*, 2017.

Rossi et al. 2015 ROSSI, A. D. et al. *Manual Técnico MHOLline 2.0*. [S.l.], 2015. Page 2 - Manual não publicado.

Verli 2014 VERLI, H. Bioinformática: Da biologia à flexibilidade molecular. SBBq, São Paulo - Brasil, 2014. Pages 2-3.

White et al. 2013 WHITE, E. P. et al. Nine simple ways to make it easier to (re) use your data. *Ideas in Ecology and Evolution*, v. 6, n. 2, 2013.