UNIVERSIDADE DE SÃO PAULO
FACULDADE DE CIÊNCIAS FARMACÊUTICAS DE RIBEIRÃO PRETO

Simulação computacional distribuída: Aplicação a problemas de folding
de heteropolímeros

Dissertação de Mestrado apresentada ao
Programa de Pós-Graduação em Ciências
Farmacêuticas para obtenção do Título de
Mestre em Ciências

Área de Concentração: Química e Física
Biológica

Orientador: Prof. Dr. Antonio Caliri

**Versão corrigida da Dissertação de Mestrado apresentada ao Programa de
Pós-Graduação em Ciências Farmacêuticas em 17/01/2017. A versão original encontra-se
disponível na Faculdade de Ciências Farmacêuticas de Ribeirão Preto/USP.**

**Ribeirão Preto
2017**

AUTORIZO A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTE TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Silva, Pablo Andrei

Simulação computacional distribuída: Aplicação a problemas de folding de heteropolímeros . Ribeirão Preto, 2017.

107 p.

Dissertação de Mestrado, apresentado à Faculdade de Ciências Farmacêuticas de Ribeirão Preto/USP– Área de concentração: Química e Física Biológica.

1. Computação Distribuída. 2. *Folding*. 3. Proteínas.

FOLHA DE APROVAÇÃO

Pablo Andrei Silva

Simulação computacional distribuída: Aplicação a problemas de folding de heteropolímeros

Dissertação de Mestrado apresentada ao
Programa de Pós-Graduação em Ciências
Farmacêuticas para obtenção do Título de
Mestre em Ciências

Área de Concentração: Química e Física
Biológica

Orientador: Prof. Dr. Antonio Caliri

Aprovado em:

Banca Examinadora

Prof. Dr. _____

Instituição: _____ Assinatura: _____

Prof. Dr. _____

Instituição: _____ Assinatura: _____

Prof. Dr. _____

Instituição: _____ Assinatura: _____

Este trabalho é dedicado para minha família que forneceu suporte e entusiasmo. A minha noiva Thamy Senni que sempre esteve ao meu lado me apoiando, mesmo nos momentos de stress. E ao meu amigo João Paulo Dal Molin que foi fundamental como guia, amigo e muitas vezes revisor. Em especial, aos meus avós, em memória, que me incentivaram a nunca desistir dos meus sonhos.

AGRADECIMENTOS

Os agradecimentos principais são direcionados à HPE (Hewlett Packard Enterprise) em nome do amigo Luiz Carlos Martinez Jr., MSc.. Eles me apoiaram nesta jornada, por conta de reconhecer a importância da Ciência dentro e fora do ambiente empresarial.

Agradecimentos especiais ao meu orientador, o Prof. Dr. Antonio Caliri, que de forma cativante me ajudou durante estes anos com paciência, conhecimento e competência; e ao Dr. João Paulo Dal Molin, por todo suporte prestado durante estes anos.

*“Mesmo desacreditado e ignorado por todos,
não posso desistir, pois para mim, vencer é nunca desistir.”*
Albert Einstein

RESUMO

SILVA, P.A. **Simulação computacional distribuída: Aplicação a problemas de folding de heteropolímeros**. 2017. 107 f. Dissertação (Mestrado em Ciências). Faculdade de Ciências Farmacêuticas de Ribeirão Preto – Universidade de São Paulo, Ribeirão Preto, 2016.

Nesta dissertação apresentamos o desenvolvimento de ferramentas computacionais dedicadas a racionalizar processos que envolvem simulações Monte Carlo e análises de suas aplicações; são evocados conceitos pertinentes às principais áreas envolvidas (computação, tecnologia da informação, matemática e física). São introduzidas e discutidas técnicas de simulação computacional distribuída e o método Monte Carlo, com ênfase à aplicação em heteropolímeros. Exemplos ilustrativos de aplicação da ferramenta também são providos, mediante simulações e análise dos resultados de três tipos de cadeias heteropoliméricas em rede regular: cadeia polar (todos monômeros polares); cadeia hidrofóbica (todos monômeros apolares); cadeia com mescla de monômeros polares e apolares (modelo HP). O propósito motivador deste trabalho é o estudo do problema de *folding* de heteropolímeros, o que inclui proteínas. Contudo, a ferramenta em questão, poderá ser generalizada e aplicada a praticamente todos os tipos de polímeros lineares em rede, pois o usuário poderá definir e implementar o modelo de cadeia que desejar.

Palavras-chave: Computação Distribuída, *Folding*, Proteínas.

ABSTRACT

SILVA, P.A. **Simulação computacional distribuída: Aplicação a problemas de folding de heteropolímeros.** 2017. 107 f. Dissertação (Mestrado em Ciências). Faculdade de Ciências Farmacêuticas de Ribeirão Preto – Universidade de São Paulo, Ribeirão Preto, 2016.

We present the development of computational tools dedicated to streamlining processes involving Monte Carlo simulations and analysis of some applications. Relevant concepts to the main areas involved (computation, information technology, mathematics and physics) are evoked. We introduce and discuss techniques of distributed computer simulation and Monte Carlo method with emphasis on heteropolymers. Illustrative examples of the application of the tool are also provided through simulations and analysis of results from three types of polymeric chains: polar chain (all polar monomers); hydrophobic chain (all nonpolar monomers); and a chain with mixture of polar and nonpolar monomers (HP model). The motivating purpose of this work is the study of the *folding* problem of heteropolymers, which includes proteins. However, the tool in question, can be generalized and applied to virtually all types of linear polymers network, once users can define and implement the chain models they want.

Key-words: Distributed Computing, Folding, Proteins.

LISTA DE ILUSTRAÇÕES

Figura 1 – Este resultado mostra a importância das constantes ($C1$ e $C2$) que caracterizam o gerador. Note que a ordem com que os números são gerados guardam uma correlação linear com os valores gerados, desqualificando o gerador . . .	13
Figura 2 – O pequeno período e a dependência linear entre a ordem com que os números são gerados e os seus valores, embora com alguma oscilação de pequena amplitude, determinam a total insuficiência do presente gerador definido pelas operações: $seed = 89*seed+113$	15
Figura 3 – Distribuição de 10.000 pares de pontos aleatórios no plano X-Y, ($X:[0 - 1]$ e $Y:[0 - 1]$), obtidos pelo gerador definido pelas operações: $seed = 1103515245*seed - 2147483648$. Note que todo o plano é preenchido de forma praticamente uniforme pelos pares gerados	17
Figura 4 – Modelo de Ising em uma dimensão. À esquerda, temos seis spins todos orientados numa mesma direção, a qual corresponde ao estado de menor energia potencial do sistema. A direita, os dois estados possíveis de um mesmo <i>spins</i>	24
Figura 5 – Em 3D são possíveis somente quatorze redes distintas (células que, quando justapostas, preenchem totalmente o espaço que encerram). Estas redes são rotulados como "Redes de Bravais".	27
Figura 6 – Exemplificação do movimento Crankshaft em dois eixos	31
Figura 7 – Ilustração de uma simples cadeia mostrando um contato topológico, entre resíduo "3" e resíduo "6".	35
Figura 8 – Interface de acessibilidade da ferramenta	43
Figura 9 – Elementos do sistema de processamento distribuído; veja descrição no texto que se segue.	44
Figura 10 – Simulação MC de uma cadeia polar para cinco temperaturas distintas. As tabelas ilustram a forma de saída dos dados: a tabela da esquerda mostra a evolução do quadrado do raio de giração (rg2), e a da direita mostra a evolução da energia do sistema (U) . São mostrados 16 instantâneos da parte inicial da simulação.	48

Figura 11 – Resultados da simulação (para cinco diferentes temperaturas) de uma cadeia polar de 27 unidades. São mostrados (à esquerda) a evolução (ao longo da simulação) da raiz quadrada da média do raio de giração ao quadrado $\langle rg^2 \rangle$, e a evolução da média da energia potencial $\langle U \rangle$. As curvas foram construídas usando 10.000 instantâneos igualmente espaçados, cerca de 2% do total de passos MC usados na simulação. Assim, o tempo MC = 10.000 corresponde a 500.000 passos MC	49
Figura 12 – Simulação MC de uma cadeia apolar para cinco temperaturas distintas. As tabelas ilustram a forma de saída dos dados: a tabela da esquerda mostra a evolução do quadrado do raio de giração (rg^2), e a da direita mostra a evolução da energia do sistema (U). São mostrados 16 instantâneos da parte inicial da simulação.	51
Figura 13 – Resultados da simulação de uma cadeia apolar de 27 unidades. São mostrados (à esquerda) a evolução ao longo da simulação da raiz quadrada da média do raio de giração ao quadrado $\langle rg^2 \rangle$, e à direita a evolução da média da energia potencial $\langle U \rangle$, para cinco diferentes temperaturas. As curvas foram construídas usando 10.000 instantâneos igualmente espaçados, cerca de 2% do total de passos MC usados na simulação: tempo MC = 10.000 corresponde a 500.000 passos MC	51
Figura 14 – Simulação MC do modelo HP para cinco temperaturas distintas. As tabelas ilustram a forma de saída dos dados: a tabela da esquerda mostra a evolução do quadrado do raio de giração (rg^2), e a da direita mostra a evolução da energia do sistema (U). São mostrados 16 instantâneos da parte inicial da simulação.	53
Figura 15 – Resultados da simulação de uma cadeia HP de 27 unidades. São mostrados (à esquerda) a evolução ao longo da simulação da raiz quadrada da média do raio de giração ao quadrado $\langle rg^2 \rangle$, e à direita a evolução da média da energia potencial $\langle U \rangle$, para cinco diferentes temperaturas. As curvas foram construídas usando 10.000 instantâneos igualmente espaçados, cerca de 2% do total de passos MC usados na simulação: tempo MC = 10.000 corresponde a 500.000 passos MC	53
Figura 16 – Configuração final da simulação do modelo HP à temperatura $T = 0,3$	54
Figura 17 – Configuração do modelo cliente-servidor	71
Figura 18 – Modelo de comunicação entre cliente-servidor	71
Figura 19 – Imagem ilustrativa de instalação do pré-requisito IIS	74

LISTA DE QUADROS

Quadro 1 – Exemplo da distribuição em intervalos que se espera dos números gerados através de um gerador efetivo, no caso o gerador definido pelas operações: $seed = 1103515245 * seed - 2147483648$. Os números são gerados no intervalo [0 - 10] e distribuídos em 10 intervalos: [0 - 1], [1 - 2]; [2 - 3]; ..., [9 - 10].	19
Quadro 2 – Neste exemplo, com o mesmo gerador $seed = 1103515245 * seed - 2147483648$ do caso anterior, os números foram gerados no intervalo [0-26] e distribuídos em 26 intervalos: [0-1], [1,2], ..., [25-26].	20
Quadro 3 – Quadro com configuração utilizada para obtenção modelo HP 50-50	52

LISTA DE CÓDIGOS-FONTE

Código-fonte 1 – Exemplo do código escrito na linguagem computacional C# para geração de números randômicos	11
Código-fonte 2 – Exemplo do código para o gerador $seed: 1*seed+3$	12
Código-fonte 3 – Exemplo do código para o gerador $seed = 89*seed+113$	14
Código-fonte 4 – Exemplo do código para o gerador efetivo	15
Código-fonte 5 – Exemplo de um programa para testar a distribuição de números gerados aleatoriamente em intervalos preestabelecidos	20
Código-fonte 6 – Exemplo do código fonte em C# do Algoritmo de Metropolis	38
Código-fonte 7 – Script de Liberando as portas do Firewall para acesso NetBIOS . . .	89
Código-fonte 8 – Script de Liberando as portas do Firewall para acesso SQLServer . .	89
Código-fonte 9 – Script de Liberando as portas do Firewall para acesso eFolding Hosting	90
Código-fonte 10 – Configuração padrão para obtenção do LOG no ambiente cliente . . .	106
Código-fonte 11 – Configuração padrão para obtenção do LOG no ambiente servidor . .	106

LISTA DE TABELAS

Tabela 1 – Comparação do percentual de utilizadores por sistema operacional	55
---	----

____LISTA DE ABREVIATURAS, SÍMBOLO E SIGLA

k_B		Constante de Boltzmann
T		Temperatura
CPU		Unidade Central de Processamento
H		Hidrofóbicos
HP		Hidrofóbico- Polar
MC		Monte Carlo
NP		Polinomial não determinístico
P		Polares

SUMÁRIO

1	INTRODUÇÃO	1
1.1	O problema do <i>folding</i> de heteropolímeros lineares	3
1.2	Simulação computacional do processo de <i>folding</i>	3
2	OBJETIVOS E JUSTIFICATIVAS DO TRABALHO	5
2.1	Importância do estudo do problema de <i>folding</i> de proteínas	5
2.2	Importância da Simulação Computacional	6
3	MÉTODO MONTE CARLO	9
3.1	Simulação Monte Carlo.	9
3.2	Geradores de Números Aleatórios	10
3.2.1	<i>O período dos geradores de números pseudo aleatórios</i>	11
3.3	Testes da qualidade de um gerador	12
3.3.1	<i>Gerador não efetivo</i>	12
3.3.2	<i>Gerador efetivo</i>	14
3.3.3	<i>Distribuição em intervalos dos números obtidos pelo gerador</i>	18
4	APLICAÇÕES DO MÉTODO MONTE CARLO	23
4.1	MC passo a passo - Um exemplo: Sistema de <i>spins</i>	23
4.2	Lattices	26
4.3	Método Monte Carlo para polímeros	27
4.3.1	<i>Exemplos da composição de dois modelos simples</i>	29
4.4	Detalhes do programa para simulação de polímeros	29
4.4.1	<i>O processo de amostragem configuracional da cadeia</i>	30
4.4.2	<i>Move set</i>	31
4.4.3	<i>Validação da sequência de configurações geradas</i>	33
4.4.4	<i>Parâmetros que descrevem e controlam a cadeia</i>	33
4.4.5	<i>Cálculo de Energia de uma dada configuração da cadeia</i>	33
4.4.6	<i>Algoritmo para cálculo de energia após um movimento válido</i>	36
4.5	O fator temperatura e a constante de Boltzmann	36
4.5.1	<i>O peso de Boltzmann</i>	37
4.5.2	<i>O algoritmo de Metropolis</i>	37
5	PLATAFORMA DE SIMULAÇÃO DISTRIBUÍDA	41

5.1	Desenvolvimento de uma plataforma de simulação computacional distribuída, intuitiva, versátil e segura	41
5.2	Possibilidade de executar mais de uma simulação em concorrência de máquina	44
6	APLICAÇÕES E RESULTADOS	47
6.1	Cadeia Polar	47
6.2	Cadeia Apolar	50
6.3	Modelo HP	52
6.4	Comentários sobre detalhes técnicos	54
7	COMENTÁRIOS FINAIS	57
7.1	Sobre a plataforma simulacional implementada	57
7.2	Sobre os resultados das simulações obtidos	58
7.2.1	<i>Comparativo de resultados com a versão legada</i>	58
7.3	Perspectivas futuras	59
7.3.1	<i>Política de escalonamento de trabalhos e de distribuição de cargas</i>	60
	REFERÊNCIAS	61
	Glossário	65
ANEXO A	INSTALAÇÃO	67
A.1	Introdução	67
A.2	Licença de utilização do software	68
A.3	Introdução ao Sistema eFolding	68
A.3.1	<i>Características e principais funcionalidades</i>	68
A.4	Arquitetura	69
A.4.1	<i>Computação distribuída</i>	69
A.5	Camada de rede	70
A.5.1	<i>Modelo Cliente-Servidor</i>	70
A.5.2	<i>Comunicação Cliente Servidor</i>	70
A.6	Protocolos e tecnologias utilizados na comunicação eFolding	72
A.6.1	<i>WCF (Windows Communication Foundation)</i>	72
A.6.2	<i>Arquitetura baseada em Web services</i>	72
A.6.3	<i>Funcionamento - Comunicação</i>	72
A.6.4	<i>Componentes de um serviço WCF</i>	72
A.7	Instalação da Aplicação	73
A.7.1	<i>Requisitos gerais para instalação</i>	73
A.7.2	<i>Hardware mínimo necessário</i>	73

A.7.3	<i>Pré-requisitos</i>	73
A.7.4	<i>Passo-a-passo da Instalação no Microsoft SQLServer 2014</i>	74
A.7.5	<i>Passo-a-passo da Instalação no Internet Informatino Services</i>	85
A.7.6	<i>Como configurar o seu site da WEB do IIS</i>	87
A.7.6.1	<i>Configurar um site padrão</i>	87
A.7.6.2	<i>Criar um novo site</i>	88
A.7.7	<i>Habilitando conexões remotas no Servidor - Firewall</i>	89
A.8	<i>Atualização da Aplicação (Se necessário)</i>	90
A.8.1	<i>Atualização do Servidor</i>	90
A.8.2	<i>Atualização do banco de dados</i>	90
ANEXO B	GUIA DO USUÁRIO	91
B.1	<i>Introdução</i>	91
B.1.1	<i>Objetivo</i>	91
B.2	<i>Licença de utilização do software</i>	91
B.3	<i>Requisitos de Sistema</i>	92
B.3.1	<i>Requisitos mínimos de sistema</i>	92
B.3.2	<i>Como instalar o Web Browser</i>	92
B.4	<i>Procedimento de Login</i>	93
B.4.1	<i>Falha no procedimento de Login</i>	94
B.4.2	<i>Procedimento de Logoff</i>	95
B.5	<i>Procedimento para Simulação</i>	95
B.5.1	<i>Encaminhar uma simulação para ser processada</i>	97
B.6	<i>Criando uma nova Simulação.</i>	98
B.7	<i>Como editar e deletar uma Simulação?</i>	100
B.8	<i>Obtendo os resultados de uma Simulação</i>	101
ANEXO C	LOG DE EXCESSÃO E ERRO DA APLICAÇÃO	103
C.1	<i>Introdução</i>	103
C.2	<i>Objetivo</i>	103
C.2.1	<i>Processo de logging de eventos para a plataforma eFolding</i>	103
C.2.2	<i>A trilha de auditoria</i>	104

INTRODUÇÃO

Este trabalho aborda o desenvolvimento de meios computacionais para auxiliar o entendimento do processo de *folding* (dobramento, ou enovelamento) de heteropolímeros, como por exemplo, proteínas. De forma específica, introduziremos ferramentas capazes de agilizar o estudo e pesquisa por meio de processamento computacional distribuído entre muitos computadores (TANENBAUM, 2016). Para sua consecução várias barreiras precisam ser superadas, tanto do ponto de vista do entendimento do problema, quanto dos meios técnicos disponíveis. Do ponto de vista técnico, entre as muitas dificuldades discutidas na literatura (PATTERSON DAVID A.; HENNESSY, 1989), assim como suas possíveis soluções, listamos as seguintes:

1. Não reaproveitamento de resultados de simulações e de dados de pesquisas anteriores. Frente a esta situação a ferramenta proposta neste trabalho contribuirá para inibir a redundância de experimentos simulacionais, por meio da facilitação do acesso a resultados de experimentos já realizados pelos usuários do sistema (item seguinte);
2. Inexistência, em geral, de base de dados de arquivamento dos resultados. A implantação de um arquivo de uso intuitivo e comum, com o registro das "trajetórias" e dos parâmetros dos sistemas já simulados, poderá evitar simulações redundantes;
3. Necessidade da popularização do paradigma (conceito) de *Big Data* (MAGOULAS ROGER; LORICA, 2009). Condicionar, mesmo que de forma incipiente, a utilização dos dados obtidos, de forma a desenvolver e aplicar o conceito na disponibilização e reprocessamento dos resultados obtidos no meio acadêmico;
4. Deficiências na prática do Processo Multidisciplinar. Apesar do termo ser bem conhecido no ambiente acadêmico, na prática sua implementação é ainda tímida em nosso meio, exigindo mudanças de postura. Um exemplo é este trabalho, o qual envolve especialistas de diferentes áreas do conhecimento. Assim, além de induzir o desenvolvimento do comportamento pessoal, por meio de ferramentas facilitadoras numa comunidade multidisciplinar, a formulação dos textos de teor eminentemente técnico, foram elaborados

com a preocupação de torná-los o mais compreensível possível. Para isto, eliminamos ou explicamos os jargões das diferentes áreas envolvidas, afim de possibilitar o entendimento para pessoas de áreas diversificadas do conhecimento. Adicionalmente, o sistema também tem como objetivo incentivar a geração de novos códigos fonte (*hard code*), por qualquer indivíduo envolvido com áreas do problema a que o sistema se destina, ou seja, no estudo do processo de *folding* de heteropolímeros.

5. Dificuldades com programas de computadores: Por exemplo, o código fonte utilizado até então (legado), empregado por anos, possui as seguintes características deficitárias:

- Documentação técnica hermética;
- Técnicas de diagramação de código fonte de difícil entendimento;
- Redundâncias: o programa legado esta altamente acoplado, ou seja, o programa não pode ser modularizado. Existe muita redundância em pontos comuns do programa construído no passado;
- Em muitos pontos, o teor é extremamente técnico ou matemático, e os sumários não são auto-explicativos;
- É necessário ter conhecimento específico de FORTRAN (ECMA, 1965) (a linguagem de programação conhecida como Fortran foi desenvolvida na década de 1950 e é principalmente usada em Ciência da Computação e Análise numérica) para executar, corrigir, alterar, otimizar, etc., o programa;
- Os dados obtidos pela ferramenta atual são gerados em um formato de texto plano, o qual requer intervenção humana para a adequação de sua apresentação;

Estas características dificultam o uso do programa. Assim, segue abaixo a relação de algumas vantagens quanto ao uso do software colaborativo pretendido:

- Interface gráfica intuitiva entre usuário e máquina;
- Minimização de conhecimentos técnicos da área computacional para operação da ferramenta;
- Compartilhamento de dados comuns entre pesquisadores;
- Compartilhamento de recursos de hardware disponíveis entre os pesquisadores e demais colaboradores (pessoas ou instituições que podem e desejam compartilhar recursos de máquina, capacidade de processamento, instalação de softwares);
- Flexibilidade na distribuição de tarefas de processamento, de acordo com os recursos de hardware disponíveis, e não somente na máquina do pesquisador;
- Fila única de processos (simulações) pré “*setadas*” (preparadas) para execução, obtendo-se melhor distribuição de carga de processamento;

Características básicas da ferramenta:

- Compartilhamento de recursos: utilização de recursos de *Input* e *Output* (IO: entrada e saída de dados, etc.) (FILHO GILBERTO FARIAS DE SOUZA; ALEXANDRE, 2014) , memória e processador de máquina conectada à ferramenta de colaboração;

- Escalabilidade: a quantidade de trabalho envolvido no processamento (de qualquer requisição de processamento ou recurso) é compartilhada entre todos os equipamentos envolvidos;
- Tolerância a falhas: as simulações podem ser paralisadas e posteriormente reiniciadas;
- Portabilidade no acesso às informações: pode-se utilizar qualquer dispositivo conectado à Internet, tais como: Notebooks, Tablets, Smartphones e outros;

1.1 O problema do *folding* de heteropolímeros lineares

O processo físico-químico através do qual toda cadeia proteica assume a sua configuração funcional correspondente (estrutura tridimensional específica) é denominado “*folding*” de proteínas. Assim, sob condições fisiológicas naturais, o problema do *folding* trata de um processo espontâneo que conduz a cadeia protéica, a partir de uma configuração aleatória (geralmente aberta), à sua conformação, precisa e única, chamada estrutura nativa. O fenômeno de *folding* de proteínas é complexo, controlado por vários tipos de interações energéticas principalmente (não ligadas) polares e apolares. Durante o processo de *folding*, as ligações covalentes entre os aminoácidos são conservadas; as poucas exceções envolvem as pontes dissulfetos. Os movimentos tridimensionais da cadeia, graças à sua grande flexibilidade são efetuados aminoácido por aminoácido, sem portanto aumento ou diminuição do tamanho da cadeia. Existem situações em que o *folding* não ocorre corretamente no organismo, acarretando doenças como Alzheimer, BSE (doença da vaca louca), Diabetes tipo II, entre outras, podendo até mesmo propiciar o surgimento de alguns tipos de câncer (SELKOE, 2003).

1.2 Simulação computacional do processo de *folding*

O processo de *folding* de peptídeos (pequenas proteínas) *in vivo* pode ser considerado como sendo relativamente rápido, e muitíssimo rápido (em relação tempo de processamento) se comparado as simulações computacionais envolvendo proteínas, atualmente. Assim, apesar de todo o avanço tecnológico na área computacional, ainda é impossível simular e acompanhar rigorosamente o processo de *folding*, mesmo para heteropolímeros pequenos, com apenas dezenas de aminoácidos. *In vivo*, dependendo da proteína, o processo de *folding* pode levar de nano-segundos a horas para se completar, enquanto que nos mais potentes computadores atuais, a simulação completa de tal processo, para uma pequena proteína com algumas dezenas de aminoácidos, quando bem sucedida, pode levar dias. Dessa forma, ainda existe uma barreira computacional a ser ultrapassada.

Uma solução para mitigar essa limitação tem sido a utilização de algoritmos computacionais de forma distribuída, para simular o que seria praticamente impossível em uma única unidade central de processamento (CPU) convencional (TANENBAUM ANDREW; STEEN, 2016). Em outras palavras, estamos falando da divisão de trabalho em vários processadores

que trabalhariam simultaneamente. O intuito específico aqui é distribuir a carga de processamento entre processadores existentes nos computadores que passam boa parte do tempo inativos: é possível criar, dentro de cada processador, *threads* (sequencia de tarefas) de forma que o processamento se dê em paralelo em diversas situações de cálculo computacional.

Um exemplo específico de técnica utilizada para o ganho de performance é a de organizar o processamento de informações entre várias tarefas através de uma linha do tempo. Esta técnica utiliza o seguinte conceito: toda simulação gera um grande número de resultados, mas boa parte destes resultados são dependentes de um resultado fundamental, a partir do qual pode ser obtido posteriormente (e executado em paralelo). Assim, todo esforço computacional se concentra inicialmente neste resultado fundamental, que denominaremos de "trajetória temporal do sistema". No contexto deste trabalho, tal trajetória compõe-se da sequência "temporal" das coordenadas espaciais de cada resíduo da cadeia. Este é o resultado da simulação que deve ser armazenado (ou seja, gravado). A tarefa seguinte, a análise do problema, é feita *a posteriori* e de forma independente. Além disso, todas as outras necessidades, como a de apresentar a informação processada, levamos em conta a criação de arquivos de resultado, por exemplo planilhas Excel, arquivos do estado configuracional da simulação, resultados sumarizados, e outros, serão feitas de forma independente da simulação.

OBJETIVOS E JUSTIFICATIVAS DO TRABALHO

O objetivo principal deste trabalho é desenvolver e disponibilizar ferramentas computacionais capazes de agilizar simulações Monte Carlo do processo de *folding* de heteropolímeros. Especificamente, a proposta visa implantar um sistema que: (i) possibilite a computação distribuída em múltiplos processadores conectados remotamente pela Internet; (ii) resultados obtidos, as trajetórias, sejam armazenadas e disponíveis à comunidade; (iii) disponibilize uma interface intuitiva, de fácil uso pelo usuário, para submissão de simulações, recolhimento dos resultados, análise dos resultados, e preparação dos resultados para apresentação visual, por exemplo, gráficos, figuras, filmes, etc. Nos itens seguintes comentamos as principais motivações e justificas para o desenvolvimento da ferramenta em pauta.

2.1 Importância do estudo do problema de *folding* de proteínas

A vida depende basicamente do funcionamento preciso de uma grande quantidade de proteínas. Porém, podem ocorrer diferentes tipos de falhas moleculares e então as proteínas não executam mais suas funções preestabelecidas de forma eficiente. Isso se dá ou porque o organismo não é capaz de produzi-las em quantidades suficiente, ou porque são produzidas com algum defeito. Como a função da proteína depende estritamente da sua estrutura nativa, uma pequena falha em sua estrutura nativa pode fazer com que a proteína deixe de ser capaz de executar eficientemente sua função biológica. Algumas patologias já foram listadas tendo sua origem num processo de *folding* do qual resultam proteínas defeituosas. (SELKOE, 2014)

Esta fina dependência entre a estrutura e função da proteína pode também funcionar contra patologias provocadas por microrganismos que eventualmente invadem o corpo humano, como bactérias, entre outros. Apesar de se encontrar similaridades entre proteínas de organismos

diferentes, as pequenas diferenças entre eles permitem que se analise as proteínas de organismos patológicos com a perspectiva de se conseguir bloquear o funcionamento do patógeno por meio de moléculas cuidadosamente desenhadas, que sejam inócuas para a proteína humana. Portanto, para compreendermos como determinar a estrutura nativa e o funcionamento de uma proteína, a partir de sua sequência de aminoácidos, é necessário elucidar a questão mais geral e mais fundamental que é a do mecanismo de *folding* de proteínas. (ECROYD H; CARVER, 2008)

2.2 Importância da Simulação Computacional

Atualmente em todas as áreas do conhecimento é visível a crescente necessidade/dependência da Tecnologia de Informação (TI) no nosso dia a dia, envolvendo desde *Hardware* superpoderosos utilizados na previsão climática e em equipamentos médicos, até chegarmos ao âmbito doméstico dos *devices* utilizados para Internet e redes sociais. Na grande maioria das atividades humanas, e em particular em vários setores da indústria e nas atividades financeiras, por exemplo, buscam-se incessantemente por inovações tecnológicas. Portanto, estamos numa fase em que podemos esperar por melhorias sistemáticas de resultados e pelo aumento da produtividade. A sigla TI abrange os recursos de *software*, *hardware*, redes e dados. (TURBAN, 2006).

De fato, alguns setores como *Healthcare* e *Finance* já investem sistematicamente na prática multidisciplinar e aplicam volumosas quantias de dinheiro em TI (o que sugere a concorrência e a rivalidade como sendo os principais fatores que justificam esta prática) (LUNARDI G. L.; BECKER, 2003). Mas, especificamente nas nossas Universidades, é ainda reduzido o número de instituições onde já existem o processo (atividade) multidisciplinar, ou seja, interação sinérgica entre as diversas disciplinas clássicas. Aquela visão segregativa e recorrente, do passado, que se motivava pelo mote "minha área de pesquisa é a mais importante porque ..." já não se aplica, pois os problemas importantes dos dias atuais demandam uma interação compromissada entre áreas do conhecimento tradicionalmente compartimentalizadas. Neste cenário, a informática funciona também como uma linguagem universal, facilitando a atividade multidisciplinar. Seguem algumas definições pertinentes.

1. *Contextualização e Fundamentação*. Para Turban, Leidner, McLean e Wetherbe (TURBAN E.; LEIDNER, 2007), um Sistema de Informação (SI) é um sistema capaz de coletar, processar, armazenar, analisar e disseminar informações para atender um propósito específico. Como qualquer sistema, um SI inclui entradas (dados e instruções) e saídas (cálculos, análises e relatórios) (NICKERSON, 2001) e também engloba pessoas, procedimentos e facilidades físicas e opera em um determinado ambiente (TURBAN EFRAIM; RAINER, 2003).
2. *Simulação*. A palavra "simulação" sempre nos chama muito à atenção. Ela carrega consigo o conceito de poder abstrair do mundo real situações nas quais podemos trabalhar em ambi-

ente controlado, sem a necessidade de utilizar recursos materiais. Desta forma, a simulação também pode acelerar o entendimento de diversos tipos de problemas, nas mais diversas situações. Hoje, processos simulacionais são aplicados em quase a totalidade das áreas do conhecimento. Em geral, trata-se de uma maneira de antecipar ou prever acontecimentos; estudar/compreender processos. Simulações são dependentes de "modelos matemáticos", e aqui, mais uma vez, o crescente desenvolvimento de máquinas computacionais poderosas tem permitido uma sistemática melhoria na aproximação entre "modelo" e o "problema" que o mesmo representa.

MÉTODO MONTE CARLO

Nesta seção apresentamos conceitos relativos ao processo de simulação estatístico denominado método Monte Carlo (MC). Este método se baseia em amostragens aleatórias e por isso demanda o uso de números aleatórios (ou mais precisamente, na prática, números pseudo-aleatórios).

3.1 Simulação Monte Carlo.

A técnica de simulação MC é de grande utilidade em áreas básicas diversas como Física, Química, Matemática, e Biologia, assim como em áreas aplicadas como Engenharia, Medicina e Informática. Uma das variantes pioneiras do método foi utilizada para determinar o valor de π , no famoso experimento da agulha de Buffon ([WEISSTEIN, 2017](#)) ([GRIFFITHS, 2007](#)). Na Física, o método MC teve sua origem em problemas envolvendo interpretação probabilística ([KROESE D. P.; BRERETON, 2014](#)), com o estudo de variáveis aleatórias e em problemas que envolvem muitas variáveis, como no cálculo de valores médios de grandezas termodinâmicas extensivas que envolvem interações multi-dimensionais. Pensando na proposta desta dissertação, este método é particularmente apropriado para sistemas discretos, tais como os modelos de “redes de *spins*”, que trataremos de forma ilustrativa do método MC nos tópicos abaixo.

Em geral, MC é utilizado em simulações estocásticas de qualquer natureza, em que se usa distribuições de probabilidades para obtenção de uma aproximação numérica de funções genéricas complexas; é muito popular na comunidade acadêmica e tem sido utilizado sistematicamente em pesquisa. Mesmo que um problema seja originalmente formulado deterministicamente é possível tratá-lo pelo método MC. São objetos corriqueiros do método MC problemas de otimização, integração numérica e determinação da evolução de sistemas governados por certas distribuições de probabilidades ([SOBOL, 1994](#)) ([FRENKEL, 2004](#)).

Uma limitação dos métodos de simulação em geral, incluído MC, consiste no fato de

que somente sistemas finitos podem ser utilizados. Contudo, como para aplicações de nosso presente interesse, que trata de cadeias heteropoliméricas, que sempre são finitas, e relativamente pequenas, torna-se viável a utilização do método para o estudo do processo de *folding*.

3.2 Geradores de Números Aleatórios

Geradores de números aleatórios são fundamentais para métodos de simulação estatísticos, e em particular para o MC (BLIKSTEIN, 1999), pois este depende criticamente da qualidade dos números aleatórios utilizados. Um gerador de números aleatórios é um algoritmo que provê uma sequência de números com suficiente independência uns dos outros. Computacionalmente, existem dois tipos de geradores: geradores que utilizam o *Hardware* do equipamento para gerar a sequência de números aleatórios, e aquele que o faz através de *Software*.

Na geração através do *hardware*, o algoritmo do gerador utiliza fenômenos tais como ruídos térmicos, e necessita de um transdutor (dispositivo que recebe um sinal de um tipo e o retransmite noutra forma) para converter tais ruídos em sinais digitais. Os resultados destes processos são, em teoria, considerados imprevisíveis, isto é randômicos (ou aleatórios).

Por outro lado, através de *software* é possível desenvolver algoritmos capazes de gerar resultados muito próximos de números aleatoriamente perfeitos, chamados "números pseudo-aleatórios", pois na prática é impossível gerar números totalmente aleatórios, ou seja, com distribuição perfeitamente ao acaso. Existem muitas técnicas para geração de aleatoriedade por *software*. Dentre elas, uma das mais simples e utilizadas é a técnica do "truncamento de bits", conhecido como *linear congruential generator* (LCG). Esta técnica consiste em utilizar operações de multiplicação e adição que geram resultados que excedem o número de bits dos registros (isto é, excedem o espaço disponível para armazená-los nos registros) do computador, e assim os bits em excesso são cortados, e os bits restantes no registro (números relativos inteiros) têm comportamento praticamente aleatório.

Utilizaremos em nossos experimentos computacionais, a técnica de "truncamento de bits" em números inteiros, representados em registros de $n = 32$ bits. Neste caso, os valores limites vão de $-(2^{31})$ a $(+2^{31} - 1)$, ou seja, o intervalo de vai de -2147483648 até 2147483647.

Esta sequência de números gerados a partir de uma "*seed*" (semente) é a sequência recursivamente obtida por meio de operações do tipo $seed = seed \times C1 + C2$, onde o par de constantes, $C1$ e $C2$, são valores fixos entre os limites na representação de 32 bits, ajustados de forma a maximizar o período e a qualidade da série de números pseudo-randômicos gerados. Para obter um bom gerador de números pseudo-randômicos é importante estabelecer uma combinação de pares $C1$ e $C2$, de forma que se maximize o seu período (quando então a série de números aleatório volta a se repetir), e reduza ao máximo a correlação de pares, *tripletes*, etc., entre os números gerados. Por exemplo, se houver correlação de pares entre os números aleatórios uniformemente gerados entre zero e um, o gráfico dos pares x, y destes números não

terá distribuição uniforme no quadrado determinado pelos intervalos: $0 \leq X \leq 1$, e $0 \leq Y \leq 1$.

Segue um exemplo do *software* escrito na linguagem computacional C# para nossos experimentos:

Código-fonte 1: Exemplo do código escrito na linguagem computacional C# para geração de números randômicos

```
1 public static void Random()  
2 {  
3     seed = (int)(seed * con1 + con2);  
4 }
```

No exemplo acima a variável *seed* é declarada como sendo de 32 *bits* (31 *bits* reservados para números, e um *bit*, denominado "paridade", utilizado para detectar erros nas transmissões). Vemos que tais operações podem gerar números cujo módulo pode ser maior do que 2147483647. Os bits restantes no registro, tirando aqueles que excedem a capacidade do mesmo, são os que caracterizam o comportamento aleatório dos números gerados.

Em complemento ao tópico, existem inúmeros trabalhos técnicos publicados e disponíveis para validação; dentro destes, podemos citar "Random.org" disponível através endereço <<https://www.random.org/analysis/>>. Este material esta em formato PDF (Analysis2001.pdf) disponível em <<https://www.random.org/analysis/Analysis2001.pdf>>.

Outro trabalho muito visitado é da Silicon Graphics Incorporated Random Number Generator, o trabalho pode ser visitado em <<http://lavarand.sgi.com>>. (A.G., 1971)

Por fim, trabalho muito interessante que serviu como ponto de apoio e de equiparação a este tópico, dentre tantos outros, foi o trabalho com tema "Modelo Computacional da Dinâmica de Transmissão de Tuberculose" que aborda a geração de número randômicos junto a simulação Monte Carlo. Trabalho este provido pelo Departamento de Física e Matemática da USP de Ribeirão Preto em 2000. (PROVIDELO, 2000).

3.2.1 O período dos geradores de números pseudo aleatórios

O período de um gerador de números pseudo aleatórios é estabelecido quando a sequência de números aleatórios gerados começa a se repetir. Quanto maior o período do gerador aleatório for, maior é a qualidade do gerador. Vejamos abaixo observações quanto a qualidade do gerador levando em conta, inicialmente, o seu período:

Primeiro exemplo: **Sbyte** (LAITINEN, 2004) ou **Byte** (LAITINEN, 2004) que utiliza 8 *bites*. Este tipo representa um pequeno número inteiro que pode ser negativo ou positivo. Sendo de 8 *bits* ou 1 *byte*, o registro pode armazenar inteiros entre -128 e 127, portanto existe um período máximo de $2^8 - 1 = 255$ passos. Utilizando-se este registro de 8 *bits* na geração de números randômicos, na melhor das hipóteses, após 255 números randômicos gerados fatalmente

ocorrerá a repetição da sequência do gerador. Portanto, vemos que o período máximo de um gerador depende da capacidade do registro utilizado, ou sejam do número de *bits* utilizado para representar os números.

Segundo exemplo: Short (LAITINEN, 2004) que utiliza 2 bytes ou 16 *bits*. Neste caso, os valores limites (como no caso anterior são 2^{15} e $2^{15} - 1$, ou seja valores (números inteiros) entre -32.768 e 32.767, produzindo um período máximo de $2^{16} - 1 = 65.535$, isto é, na melhor das hipóteses, após gerar 65.535 números aleatórios distintos, ocorre a repetição da sequência do gerador, da mesma forma do caso anterior. No nosso caso, registros de 32 *bits*, produzindo sequências de período máximo de 4294967295 (ou seja, $2^{32} - 1$) números.

Portanto, para a técnica de “truncamento de *bits*”, com estes dois exemplos, fica claro que o período máximo de um gerador de números aleatórios, e sua qualidade é primeiramente determinado pelo número de bits dos registros utilizados.

Contudo, outros fatores, além do período do gerador, influenciam a qualidade dos números gerados. Um importante teste é a verificação de correlações entre dois números gerados consecutivamente, chamado correlação de pares, como já citado acima. Um bom gerador de distribuição uniforme (isto é, qualquer número dentro de um certo intervalo, que pode ser 0 e 1, tendo a mesma chance de ser obtido) deve preencher o plano com pontos de forma uniforme, sem a formação de regiões mais densas, ou regiões menos densas.

3.3 Testes da qualidade de um gerador

3.3.1 Gerador não efetivo

Com o intuito de exemplificação, criamos um pequeno algoritmo para analisar um gerador focando em certos parâmetros que o definem. Nos exemplos a seguir obtemos noções da qualidade do gerador, através da observação do comportamento dos números aleatórios gerados.

Exemplo ilustrativo de um gerador simples: $seed = C1 * seed + C2$

Este gerador, definido pelas constantes $C1=1$ e $C2 = 3$, ilustra um caso de gerador não efetivo (de qualidade extremamente insatisfatória) pois produz números inteiros de período muito pequeno, no intervalo de $[-128, 127]$, além dos números gerados serem completamente correlacionados. Segue o código deste exemplo:

Código-fonte 2: Exemplo do código para o gerador $seed: 1 * seed + 3$

```
1 class Program
2 {
3     static sbyte c1 = (sbyte)1;
4     static sbyte c2 = (sbyte)3;
```



```

5    static sbyte seed = (sbyte)1;
6    static sbyte interval = (sbyte)127;
7    static sbyte[] matrix = new sbyte[interval];
8
9    static void Random()
10   {
11       seed = (sbyte)(c1 * seed + c2);
12
13   }
14   static void Main(string[] args)
15   {
16       // sorteios
17       for (sbyte i = 0; i < interval; i++)
18       {
19           Random();
20           matrix[i] = (sbyte)seed;
21       }
22   }
23 }

```

A Figura 1 mostra o resultado do gerador $seed = 1 * seed + 3$. Os números foram gerados na sequência mostrada, isto é, da esquerda à direita. Portanto, tais números estão extremamente correlacionados o que desqualifica completamente os parâmetros (C1 e C2) do gerador.

Figura 1 – Este resultado mostra a importância das constantes (C1 e C2) que caracterizam o gerador. Note que a ordem com que os números são gerados guardam uma correlação linear com os valores gerados, desqualificando o gerador

Outro exemplo de um gerador ineficaz: $seed = 89 * seed + 113$

Similar ao caso anterior, este gerador testa números inteiros no intervalo de $[-128, 127]$ o qual corresponde ao *data type* (tipo de dado) SByte (LAITINEN, 2004). Constantes fixadas $C1 = 89$ e $C2 = 113$.

Código-fonte 3: Exemplo do código para o gerador $seed = 89 * seed + 113$

```

1      class Program
2      {
3          static sbyte c1 = (sbyte)89;
4          static sbyte c2 = (sbyte)113;
5          static sbyte seed = (sbyte)1;
6          static sbyte interval = (sbyte)127;
7          static sbyte[] matrix = new sbyte[interval];
8
9          static void Random()
10         {
11             seed = (sbyte)(c1 * seed + c2);
12
13         }
14         static void Main(string[] args)
15         {
16             // sorteios
17             for (sbyte i = 0; i < interval; i++)
18             {
19                 Random();
20                 matrix[i] = (sbyte)seed;
21             }
22         }
23     }

```

A Figura 2 mostra o resultado do gerador $seed = 89 * seed + 113$. Note que em relação ao primeiro experimento, o segundo gerador, com constantes alteradas para 89 e 113, não promoveu mudanças significativas, isto é, manteve o mesmo período e a extrema correlação entre os números gerados.

3.3.2 Gerador efetivo

Com o algoritmo a seguir pretendemos validar os testes do gerador através de argumentos estatísticos. Introduzimos noções da qualidade dos números aleatórios por meio de testes gráficos

Figura 2 – O pequeno período e a dependência linear entre a ordem com que os números são gerados e os seus valores, embora com alguma oscilação de pequena amplitude, determinam a total insuficiência do presente gerador definido pelas operações: $seed = 89 * seed + 113$.

e estatísticos, com números aleatórios gerados uniformemente, isto é, números gerados, no intervalo de "0" a "1", de sorte que qualquer um deles ocorra com a mesma chance.

Exemplo de um Programa para um gerador de números aleatórios efetivo.

Este gerador, caracterizado pelas constantes $C1=1103515245$ e $C2 = -2147483648$, ilustra um gerador efetivo pois produz números inteiros no intervalo $[0 - 1]$ uniformemente distribuídos, cujo período é da ordem de 2^{31} e cuja correlação de pares é suficientemente baixa. Segue o código deste exemplo:

Código-fonte 4: Exemplo do código para o gerador efetivo

```

1 class Program
2     {
3         static void Main(string[] args)
4         {
5             Param param = new Param();
6             param.dataToProcess = new DataToProcess() { isem =
1234567 }; // valor da semente inicial
7             AppConfigClient.Param = param;
8
9             int max = 20000;
10            int half = max / 2;
11            double[,] matrix = new double[half, 2];
12

```

```

13         // inicializa a matrix
14         for (int i = 0; i < half; i++)
15             matrix[i, 0] = matrix[i, 1] = 0;
16
17         for (int x = 0; x < half; x++)
18         {
19             for (int y = 0; y < 2; y++)
20             {
21                 Randomic.Random(); // sorteio randomico
22                 matrix[x, y] = Randomic.randu;
23             }
24         }
25
26         using (var dest = File.AppendText(Path.Combine(@"C
:\\", "TesteGerador.txt")))
27         {
28             for (int x = 0; x < half; x++)
29             {
30                 dest.WriteLine(matrix[x, 0] + "|" + matrix[
x, 1]);
31             }
32         }
33     }
34 }

```

A Figura 3 plota os pontos de um conjunto de 10.000 pares x, y (os valores de x e y são obtidos através do gerador acima, sendo o primeiro valor obtido atribuído a x , e o segundo atribuído a y , terceiro atribuído a x , quarto atribuído a y , e assim sucessivamente) de números gerados sequencialmente, nota-se que todo o plano foi coberto de forma uniforme, sugerindo que a correlação de pares pode ser desprezada. Não há praticamente regiões do plano X - Y que não foram visitadas ou com densidades significativamente maiores. Pequenas distorções resultam do tamanho da amostra relativamente pequena (10.000 pares) e podem ser mitigadas com uma amostragem maior.

Determinantes de um gerador efetivo:

(i) o período do gerador, que deve ser o maior possível, depende primeiramente do número de *bits* empregado (32 ou 64 *bits*), mas também dependem das constantes $C1$ e $C2$;

(ii) as constantes $C1$ e $C2$ são obtidas experimentalmente, por meio de tentativa e erro e determinam, além do período do gerador, a correlação entre os números gerados;

Figura 3 – Distribuição de 10.000 pares de pontos aleatórios no plano X-Y, ($X:[0 - 1]$ e $Y:[0 - 1]$), obtidos pelo gerador definido pelas operações: $\text{seed} = 1103515245 * \text{seed} - 2147483648$. Note que todo o plano é preenchido de forma praticamente uniforme pelos pares gerados

(iii) a semente inicial (número inteiro com que gerador é iniciado) deve ser ímpar e também de elevada magnitude, pois o seu valor pode alterar o período do gerador.

3.3.3 Distribuição em intervalos dos números obtidos pelo gerador

Utilizamos agora um teste para ilustrar numericamente como se dá a distribuição em 10 intervalos de valores dos números pseudo-aleatórios gerados entre 0 e 10. Os extremos à direita dos intervalos são excluídos, isto é, considera-se a frequência relativa (em %) dos números que "estão" nos intervalos $[0 - 1]$, $[1 - 2]$, $[3 - 4]$, etc., até $[9 - 10]$.

Observação: $\text{seed} = 1103515245 * \text{seed} - 2147483648$ (extraído do código-fonte tópico anterior).

Quadro 1 – Exemplo da distribuição em intervalos que se espera dos números gerados através de um gerador efetivo, no caso o gerador definido pelas operações: $\text{seed} = 1103515245 * \text{seed} - 2147483648$. Os números são gerados no intervalo $[0 - 10]$ e distribuídos em 10 intervalos: $[0 - 1]$, $[1 - 2]$; $[2 - 3]$; ..., $[9 - 10]$.

Intervalo	frequência	%
[0-1]	42949673	10
[1-2]	42949673	10
[2-3]	42949673	10
[3-4]	42949673	10
[4-5]	42949673	10
[5-6]	42949673	10
[6-7]	42949673	10
[7-8]	42949673	10
[8-9]	42949673	10
[9-10]	42949673	10
total	429496730	100

Note no Quadro-1 que a distribuição ocorre de forma uniforme, não havendo tendências na distribuição dos números favorecendo este ou aquele intervalo. No exemplo a seguir, Quadro-2, utilizando o mesmo gerador acima, os números são agora gerados no intervalo $[0 - 26]$ e distribuídos em 26 intervalos: $[0 - 1]$, $[1 - 2]$, ... , $[24 - 25]$, $[25 - 26]$.

Nos dois exemplos mostrados, utilizando o mesmo gerador de números aleatórios (Quadro-1 e Quadro-2), verificamos uma distribuição números gerados muito uniformemente distribuída em seus respectivos intervalos. Porém, este teste por si só não é suficiente, pois é preciso ainda que não haja a correlação entre pares, como ilustrado pela Figura 3 acima. Outros testes ainda podem ser aplicados, conforme a necessidade de rigor que uma aplicação em particular requer, como testes para se verificar correlação de trincas (pontos no espaço tridimensional); a distribuição de distâncias entre pares consecutivos dos números gerados; etc.

Quadro 2 – Neste exemplo, com o mesmo gerador seed = 1103515245*seed -2147483648 do caso anterior, os números foram gerados no intervalo [0-26] e distribuídos em 26 intervalos: [0-1], [1,2], ..., [25-26].

Intervalo	frequência	%
[0-1]	159072863	3,703703708
[1-2]	159072862	3,703703685
[2-3]	159072864	3,703703731
[3-4]	159072862	3,703703685
[4-5]	159072864	3,703703731
[5-6]	159072862	3,703703685
[6-7]	159072862	3,703703685
[7-8]	159072864	3,703703731
[8-9]	159072862	3,703703685
[9-10]	159072864	3,703703731
[10-11]	159072862	3,703703685
[11-12]	159072862	3,703703685
[12-13]	159072864	3,703703731
[13-14]	159072862	3,703703685
[14-15]	159072864	3,703703731
[15-16]	159072862	3,703703685
[16-17]	159072862	3,703703685
[17-18]	159072864	3,703703731
[18-19]	159072862	3,703703685
[19-20]	159072864	3,703703731
[20-21]	159072862	3,703703685
[21-22]	159072862	3,703703685
[22-23]	159072864	3,703703731
[23-24]	159072862	3,703703685
[24-25]	159072864	3,703703731
[25-26]	159072862	3,703703685
total	41355894438	100

Código-fonte 5: Exemplo de um programa para testar a distribuição de números gerados aleatoriamente em intervalos preestabelecidos

```

1    using GridProteinFolding . ConfigurationHelpers ;
2    using GridProteinFolding . RandomHelpers ;
3    class Program
4    {
5        static void Main(string [] args)
6        {
7            Param param = new Param() ;
8            param.dataToProcess = new DataToProcess() { isem =
1234567 }; // valor da semente inicial
9            AppConfigClient.Param = param;

```

```
10
11     double max = Math.Pow(2, 32);
12     int sorted = -1;
13     int interval = 27;
14
15     double[] matrix = new double[interval];
16
17     // inicializa a matrix
18     for (int i = 0; i < interval; i++)
19         matrix[i] = 0;
20
21
22     for (double i = 0; i < max; i++)
23     {
24         Randomic.Random();          // sorteio randomico
25         sorted = (int)(Randomic.randu * (interval - 1))
26
27         ;
28
29         matrix[sorted]++;
30     }
31 }
```

APLICAÇÕES DO MÉTODO MONTE CARLO

Com o desenvolvimento das máquinas computacionais, os métodos numéricos se tornaram importantes e eficientes ferramentas na solução de problemas em diversas áreas, e em particular o método Monte Carlo (MC) (frequentemente citado juntamente com o algoritmo de Metropolis). Alguns exemplos corriqueiros incluem as aplicações MC em estimativas estatísticas, problemas financeiros, áreas de terapêutica e diagnóstico, engenharia, difusão epidemiológica, em biologia, química e física. Embora possa ser aplicado onde outros métodos também se aplicam, devido a sua generalidade o método MC é uma ferramenta muito importante principalmente onde nada mais funciona. Algumas vantagens do método MC incluem:

- Generalidade: método MC pode ser aplicado em praticamente qualquer problema que tenha uma formulação ou interpretação estocástica;
- Portabilidade: a geração de números aleatórios sendo efetuada pela Unidade Central de Processamento (CPU) do computador, torna o método portátil, isto é, o *software* do gerador pode ser executado na grande maioria dos *hardware* existentes;
- Precisão: em muitas situações a acurácia da simulação pode ser facilmente ampliada através do aumento da extensão (número de passos) da simulação;
- Utilidade: a teoria relativa à simulação Monte Carlo é amplamente reconhecida e validada, o que permite que seus resultados sejam facilmente checados e aceitos;
- Plasticidade: alterações no modelo a ser utilizado podem ser feitas rapidamente, assim os novos resultados podem ser comparados com os anteriores prontamente.

4.1 MC passo a passo - Um exemplo: Sistema de *spins*

Aqui apresentamos um exemplo para introduzir o método MC; trata-se do modelo de *spins* em uma dimensão (modelo de Ising 1-D). Este modelo representa um arranjo regular de átomos, dispostos linearmente, sendo que cada átomo possui um momento magnético espontâneo

(*spin*) que interage com seus vizinhos mais próximos. Cada momento magnético é representado por uma seta, (Figura 4), e no presente caso, há apenas dois estados possíveis para cada *spin*, indicados pelas duas orientações antagônicas da seta; veja ilustração à direita da Figura 4.

Figura 4 – Modelo de Ising em uma dimensão. À esquerda, temos seis spins todos orientados numa mesma direção, a qual corresponde ao estado de menor energia potencial do sistema. À direita, os dois estados possíveis de um mesmo spins.

A configuração de menor energia potencial do sistema é aquela com todos os *spins* orientados para cima, ou equivalentemente, todos orientados para baixo. E, para cada par de *spins*, a orientação reversa dos *spins* corresponde à configuração do par com a maior energia potencial (como as orientações dispostas na ilustração mais à direita da Figura 4).

Este modelo, é denominado "Modelo de Ising"; para um sistema com N spins, há um total de 2^N possíveis configurações. A cada configuração em particular é associado uma energia potencial E dada pela seguinte expressão:

$$E = -J \sum_{\{i,j\}} S_i S_j, \quad (4.1)$$

onde $\{i, j\}$ representa o conjunto de todas as 2^N configurações sobre as quais a soma Σ incide; S_i representa o i -ésimo *spin*, o qual pode assumir os valores numéricos +1, ou -1 (por exemplo +1 representa *spin up*, isto é, a seta apontando para cima, e -1 representando o oposto (*spin down*)). J é uma constante positiva ($J > 0$) chamada constante de acoplamento entre dois *spins*. Assim se dois *spins* vizinhos apontam na mesma direção, o produto $S_i S_{i+1} = +1$ (positivo) e este par de *spins* contribui com $-J$ para a energia do sistema, e o oposto ocorre se os *spins* apontam em direções contrárias: o produto neste caso, $S_i S_{i+1} = -1$, é negativo e este par de *spins* contribui com $+J$ para a energia do sistema. No presente caso, em que a soma na Equação 4.1 considera somente as interações entre os primeiros vizinhos, a referida equação pode ser mais explicitamente definida como

$$E = -J \sum_{i=1}^{N-1} S_i S_{i+1}, \quad (4.2)$$

para um sistema com N spins dispostos linearmente. A soma é efetuada desde o *spin* $i = 1$ até o *spin* $(N - 1)$, pois há $N - 1$ interações entre os N spins.

Uma das grandezas termodinâmicas de interesse para este sistema, a energia potencial média, representada aqui por $\langle E \rangle$, é uma função da temperatura. $\langle E \rangle$ pode ser estimada como a média da energia de um grande número de configurações selecionadas com o sistema já em equilíbrio termodinâmico. Tal equilíbrio térmico é alcançado depois de um certo tempo em

que o sistema foi colocado em contato com um reservatório térmico à temperatura T (imaginado como uma grande massa de forma que a variação de sua temperatura devido a entrada ou saída de energia possa ser desprezível), ou termostato. Assim, cumprindo certo requerimento na amostragem das configurações para o cálculo (como discutido logo abaixo), depois que o equilíbrio térmico foi estabelecido, a energia média $\langle E \rangle$ pode ser estimada diretamente como

$$\langle E \rangle = (E_1 + E_2 + E_3 + \dots + E_m)/m \quad (4.3)$$

onde E_i é a energia da configuração i , e m é o número de configurações (descorrelacionadas) consideradas no cálculo de $\langle E \rangle$.

Para realizar a estimativa de $\langle E \rangle$ pelo método MC, geramos uma série de configurações alternando aleatoriamente cada um dos *spins*; a cada mudança do estado de cada *spin* uma nova configuração é gerada. Assim, depois de alcançado o equilíbrio termodinâmico, coletamos m configurações (estatisticamente independentes) para calcular $\langle E \rangle$. Para cumprir os requisitos necessários para o cálculo referente à Equação 4.3, procedemos com os seguintes passos:

1. Dados iniciais: estabelecer a Temperatura (T) do sistema em unidades de J/k_B , onde Constante de Boltzmann (k_B) (cuja unidade é J/K – Joule/Kelvin). T pode variar a partir de zero até altas temperaturas, ou seja, $T \rightarrow \infty$. É considerado que a condição de equilíbrio termodinâmico foi alcançada quando a energia E do sistema atinge o regime estacionário, isto é, o mesmo não evolui mais para qualquer direção: seu valor oscila ao redor da energia de equilíbrio $\langle E \rangle$. Depois desse estado ter sido alcançado, inicia-se a coleta de configurações estatisticamente independentes para calcular a média acima indicada, Equação 4.3.
2. Configuração inicial
 - a) escolha aleatória do estado do primeiro *spin* da cadeia, com 50% de chance de ser *up* e 50% de chance de ser *down*;
 - b) repita o passo acima para o *spin* seguinte até varrer os N *spins*.
Como pode ser verificado mais abaixo, este procedimento gera uma configuração inicial que corresponde a temperaturas elevadas ($T \rightarrow \infty$).
3. Obtenção do equilíbrio termodinâmico
 - a) Calcule a energia E_a da configuração atual;
 - b) Sorteie ao acaso um *spin* e mude seu estado: se for *up* (*down*) mude para *down* (*up*);
 - c) Calcule a energia E_n da nova configuração:
 - i. Se $E_n \leq E_a$, a nova configuração é aceita, a energia do sistema é atualizada e vai para o item (d) abaixo. Do contrário:
 - ii. Se $E_n > E_a$, calcule $\Delta E = E_n - E_a$, e gere um número randômico R entre 0 e 1, com distribuição uniforme:
 - A. Se $R \leq \exp(-\Delta E/k_B T)$, a nova configuração é aceita, a energia do sistema é atualizada e vai para o item (d) abaixo;

- B. Do contrário, se $R > \exp(-E/Tk_B)$, a nova configuração é rejeitada;
- d) Volte sistematicamente ao passo (3) até que a condição de equilíbrio termodinâmico seja satisfeita: a energia E do sistema estabiliza-se, oscilando ao redor de um certo valor $\langle E \rangle$, ΔE é da ordem de J (Equação 4.1). Em condições normais (temperatura ambiente) $\Delta E/k_B T \approx 1$.
4. Cálculo da energia média. O passo (3) é repetidos sistematicamente para a coleta de dados no regime de equilíbrio térmico. Um certo número de passos de simulação é estabelecido *a priori* a fim de obter estimativas estatisticamente significativas. No presente exemplo, uma vez no regime de equilíbrio, cada nova configuração está devidamente ponderada e suas correspondentes energias flutuam sistematicamente ao redor da energia média do sistema. Logo, para o cálculo da energia média $\langle E \rangle$ do sistema, basta utilizar um número m suficientemente grande de configurações, separadas entre si para que os valores $\{E_i\}$ usados na Equação 4.3 sejam descorrelacionados, assegurando que a média obtida tenha significância estatística.

As quatro etapas acima, de forma resumida, representam muito bem o método MC para uma grande variedade de sistemas físicos de interesse. Na parte "1" se estabelece os parâmetros gerais do sistema: temperatura, tamanho do sistema, critérios de estratificação (descartes, controle) e de finalização da simulação; na parte "2" é introduzida a configuração inicial do sistema, a qual em certas situações é muito importante, como no problema central deste trabalho, conforme veremos adiante; a parte "3", para muitos propósitos, refere-se à etapa da simulação que boa parte será descartada de forma criteriosa (obtenção do equilíbrio termodinâmico). Porém, em alguns problemas, como é o caso do problema do *folding*, esta etapa é de central importância. Finalmente, na parte "4" é que se obtém o resultado das estimativas desejadas, como é o caso de $\langle E \rangle$ no presente exemplo.

O item (c) da etapa "3" acima (em que é decidido se a nova configuração gerada será aceita ou não) é conhecido como Algoritmo de Metropolis.

4.2 Lattices

No tópico anterior o sistema físico considerado é descrito por um arranjo regular unidimensional de *spins*. Mas sistemas em duas (2D) e em três dimensões (3D) são mais importantes e mais frequentemente considerados. Para os casos em que o arranjo espacial das partes destes sistemas físicos seguem um padrão de regularidade, é interessante introduzir a noção de *lattice*. Em termos gerais uma *lattice* (rede regular), em 3D, é uma distribuição regular de pontos ou qualquer outro objeto, que preenche completamente o espaço. Na Química, por exemplo, uma *lattice* pode ser uma matriz 3D de pontos representando moléculas ou átomos em um padrão periódico, formando um cristal. Neste caso, são possíveis 14 tipos de *lattices*, conhecidas como

Figura 5 – Em 3D são possíveis somente quatorze redes distintas (células que, quando justapostas, preenchem totalmente o espaço que encerram). Estas redes são rotulados como "Redes de Bravais".

Redes de Bravais, que incluem a rede cúbica, hexagonal, triclínica, entre outras, como ilustra a figura 5. (KITTEL, 1996)

4.3 Método Monte Carlo para polímeros

O conceito de *Lattices* também pode ser aplicado em modelos computacionais de polímeros e em particular de heteropolímeros; por exemplo, as proteínas. Neste caso, os modelos são simplificações que visam representar os atributos essenciais de um heteropolímero real e, em particular aqui, com o intuito de investigação dos mecanismos de *folding* (LAU K. F.; DILL, 1989). Atualmente existem limites computacionais que impedem simular o comportamento configuracional das proteínas reais de forma completa; em particular, ainda não é possível estudar o processo de *folding* da maioria das proteínas em tempo razoável por meio de simulação. Dessa forma, modelos simplificados, mas que mantenham os aspectos fundamentais do problema de interesse tem sido frequentemente empregados. E aqui também, na presente dissertação, fazemos uso de redes regulares para também contribuir com o avanço no entendimento do problema do *folding*.

Um modelo simplificado de heteropolímero, como o adotado aqui, consiste basicamente em reduzir cada elemento (resíduo) da cadeia (polipeptídica, para o caso de uma proteína) a pontos conectados de uma *lattice* cúbica. Para garantir a ligação correta entre pares consecutivos

de elementos da cadeia, estes são dispostos em pontos adjacentes da *lattice* (pontos de menor distância na *lattice*). Restrições estéricas de sobreposição são obtidas impondo, por exemplo, que num mesmo instante cada unidade da cadeia ocupa um ponto exclusivo da *lattice*. Simplificações deste tipo reduzem significativamente o esforço computacional no manuseamento do modelo, embora mesmo neste cenário, a complexidade do problema do *folding* se mantém como um problema Polinomial não determinístico (NP) (LEIGHTON, 1998) Em Ciência da Computação, mais precisamente na teoria da complexidade computacional, NP é a mais fundamental classe de complexidade; trata-se de uma sigla em Inglês que refere-se ao "tempo polinomial não determinístico" (*Non deterministic Polynomial time*). Se um problema é dito ser NP, então sua solução é obtida em tempo computacional que cresce com o tamanho do sistema de forma polinomial.

Neste trabalho utilizamos apenas redes em três dimensões. As interações entre os elementos da cadeia e entre estes com o meio solvente são simplificadas pela introdução de potenciais energéticos de pares: condições que especificam a energia de interação entre pares de elementos da cadeia (e entre estes elementos e o meio) em contato, isto é, quando se encontram a uma distância relativamente pequena. Estes potenciais podem incluir características estereoquímicas, como tamanho e forma dos resíduos, interações energéticas entre os resíduos, além das peculiaridades do efeito hidrofóbico. Para modelos em redes regulares (*lattices*), quando o solvente pode ser aproximado por um meio uniforme e isotrópico com densidade constante, por exemplo, as interações entre os elementos da cadeia e o meio podem ser representadas equivalentemente por interações inter-resíduos por meio de um potencial aditivo. Esta situação é descrita, por exemplo, pelo modelo estereoquímico. (ROCHA L.F.O.; SILVA, 2009)

Um dos modelos considerados neste trabalho, o Modelo Hidrofóbico- Polar (HP) (T., 1998) é um dos mais simples para representar uma cadeia peptídica. Ele foi concebido para enfatizar os atributos Hidrofóbicos (H) e Polares (P) dos aminoácidos que compõem uma proteína. Para cada resíduo da cadeia é atribuído um rótulo: H se for um resíduo hidrofóbico, e P se for polar, formando uma sequência, por exemplo, PHHPHHP ... PHH. Contatos topológicos H-H (isto é, entre dois Hs não consecutivos) são favoráveis energeticamente e portanto correspondem a uma parcela da energia do sistema de, digamos $e = -1$ (unidade de energia; arbitrária); por outro lado, contatos P-P são desfavoráveis e portanto correspondem a uma componente energética $e = +1$; no caso de um contato H-P, é atribuído energia $e = 0$, ou seja, indiferente. Note que se representamos a sequência [PHHP ... PHH] que define o heteropolímero, por uma sequência de $+1$ e -1 , ou seja resíduo polar e hidrofóbico, isto é $[+1, -1, -1, +1 \dots +1, -1, -1]$ então a função

$$e_{m,n} = \frac{R_m + R_n}{2} \quad (4.4)$$

reproduz a energia $e_{m,n}$ de interação entre dois resíduos contatantes, onde $R_m = -1$ se o resíduo m for hidrofóbico, e $R_m = +1$ se for polar. Logo, se ambos forem hidrofóbicos a contribuição para a energia do sistema será $e = -1$; se ambos forem polares $e = +1$; mas se forem diferentes $e = 0$.

4.3.1 Exemplos da composição de dois modelos simples

Implementamos aqui dois modelos com a finalidade específica de validação dos programas, visto que suas soluções são intuitivamente fáceis de se prever. Assim, como descrito acima, atribuímos o valor -1 ao resíduo hidrofóbico e $+1$ ao resíduo polar, e definimos os modelos de cadeia como segue:

(i)- *Modelo que descreve um polímero em um mau solvente.* Neste caso esperamos obter o que se denomina "colapso hidrofóbico", isto é, situação que promove a compactação total da cadeia. Assim, para cada resíduo R_i , $i = 1, 2, 3, \dots, N$ é atribuído o mesmo valor -1 , para qualquer i . Neste modelo, todo contato entre pares de resíduos são favoráveis, isto é cada contato contribui (Equação 4.4) com energia $e = -1$: $e = (R_m + R_n)/2 = -1$, para qualquer par (m, n) de resíduos.

(ii)- *Modelo que descreve polímero em um bom solvente.* Neste caso como todas as interações entre pares de resíduos são desfavoráveis, isto é com energia $e = +1$ (R_i , $i = 1, 2, 3, \dots, N$ é atribuído o mesmo valor $+1$, para qualquer i), esperamos que a resposta com o sistema em equilíbrio térmico seja correspondente a uma cadeia distendida, ou seja grandes valores para o raio de giração são esperados. Neste modelo, ao contrário do anterior, todo e qualquer contato entre pares de resíduos são desfavoráveis porque contribui com energia $e = +1$ para o sistema: $e = (R_m + R_n)/2 = +1$, para qualquer par (m, n) .

4.4 Detalhes do programa para simulação de polímeros

O método MC aplicado em polímeros é apresentado aqui por meio de uma visão mais prática, em que mostramos detalhes da programação, na forma de pseudo-código (descrição literal das operações computacionais).

Nosso algoritmo é estruturado através de três *loops* (laços) de execução:

- O primeiro *loop* envolve um número de iterações P fornecido pelo usuário que executa o experimento; este número pode ser entendido como uma supra-unidade de tempo Monte Carlo, isto é, P corresponde a um múltiplo da unidade básica de tempo MC de simulação, usualmente denominado passo MC elementar, o que corresponde a um movimento (ou tentativa de movimento) da cadeia; usualmente é usado como unidade básica, N passos MC elementares, ao qual se regere simplesmente, passo MC, onde N é o tamanho da cadeia;
- O segundo *loop* envolve $D = 5$ (cinco) iterações, que serve de estratificação entre estados do sistema que, quando em equilíbrio, garantem a decorrelação entre as configurações da cadeia que serão utilizadas para cálculos de médias de grandezas extensivas, como energia $\langle E \rangle$, por exemplo;

- O terceiro *loop* utiliza com parâmetro o tamanho N da cadeia a ser utilizada, isto é, o *loop* fará N iterações, com N unidades da cadeia escolhidas ao acaso, que corresponderá, em nosso caso, à unidade, ou seja 1 (um) passo MC. Desta forma, o "tempo MC" não dependerá do tamanho da cadeia utilizada.

Como informado acima, a cada tentativa de movimento de cada unidade da cadeia, denominamos "passo MC elementar". Assim, o número total de passos MC elementares é determinado pelo produto $P \times D \times N$. E como unidade de tempo MC de trabalho podemos escolher o número de passos MC elementares que for mais convenientes; por exemplo, N ou $N \times D$.

Assim, o tamanho dos *loops*, por exemplo, seriam:

- 1º *loop*, $P = 1000$ (mil);
- 2º *loop*, $D = 5$ (cinco);
- 3º *loop*, $N = 27$ (vinte e sete), teremos: $1000 \times 5 \times 27 = 135.000$.

4.4.1 O processo de amostragem configuracional da cadeia

Para cada passo elementar MC usamos uma técnica de amostragem denominada *importance sampling* (amostragem por importância), a qual retornaremos adiante com mais detalhes, no tópico 4.5. Neste processo, os seguintes passos são executados:

1. Copiar (guardar na memória) a presente configuração da cadeia, isto é, as coordenadas x, y, z de todos os monômeros, assim como a E_a desta configuração; sobre esta configuração C_a (configuração anterior) é que serão feitas as tentativas de geração da nova configuração C_n ;
2. Sortear ao acaso (utilizando um algoritmo de sorteio randômico) o resíduo sobre qual se procederá a tentativa de movimento configuracional;
3. Tentar efetuar um dos seguintes movimentos: *CrankShaft*, *End*, ou *Kink* (veja definição dos movimentos abaixo), obedecendo a ordem:
 - a) Tentativa de movimento de *CrankShaft* (manivela);
 - b) Tentativa de movimento de *End* (extremos da cadeia);
 - c) Tentativa de movimento de *Kink* (prega);
 - d) Se nenhum movimento pode ser efetuado devido a restrições estéricas (superposição de resíduos por exemplo), a presente configuração (C_a) não é alterada, e volta-se para o passo (2) acima.
4. Vencida a etapa (3), isto é, a etapa da verificação da possibilidade física da nova configuração, a variação entre a energia da nova configuração e a anterior é calculada, $\Delta E = E_n - E_a$:
 - Se $\Delta E \leq 0$, a nova configuração é aceita e o programa segue a partir do passo (5);

- Se $\Delta E > 0$, gera-se um número randômico $0 \leq R \leq 1$ (distribuição uniforme):
 - se $R \leq \exp(-\Delta E/k_B T)$, a nova configuração C_n é aceita e o programa continua a partir do item (5);
 - se $R > \exp(-\Delta E/k_B T)$, a tentativa de mudança configuracional é rejeitada, a presente configuração (C_a) não é alterada e volta-se para o passo "2" acima.

5. Atualização

- Atualizar a energia da nova configuração E ;
- Salvar *snapshot* do cálculo de ΔE e salvar o valor da energia U da nova configuração.
- Registrar para efeito estatístico o tipo de movimento executado (cadeia);

O processo acima é repetido do item (2) ao item (5) até que o número total de passos Monte Carlo seja exaurido, ou quando por algum outro critério o mesmo seja interrompido.

4.4.2 Move set

Os movimento da cadeia citados acima garantem que praticamente qualquer uma das possíveis configurações da cadeia podem ser alcançadas a partir de qualquer configuração inicial (ergodicidade). Este conjunto de movimentos de setores da cadeia, no intuito de geração de novas e sucessivas configurações, são denominados de *Move Set* (conjunto de movimentos). Como já mencionados acima, os movimentos adotados aqui apresentam a propriedade de completeza configuracional, sendo três: movimento de *Ends*, *Kinks* e *Crankshaft*.

(i)- *crankshaft*: o movimento *Crankshaft*, ou movimento de manivela no plano, pode ocorrer em qualquer direção X , Y ou Z do espaço tridimensional. Para tal movimento existir, as coordenadas de quatro resíduos consecutivos, $R_i; R_{i+1}; R_{i+2}; R_{i+3}$ de uma configuração válida (R_i refere-se ao resíduo sorteado), definem uma manivela planar se a distância entre o resíduo R_i e R_{i+3} elevada ao quadrado se igualar à unidade, isto é, $(\vec{R}_i - \vec{R}_{i+3})^2 = 1$;

Figura 6 – Exemplificação do movimento Crankshaft em dois eixos

Assim, "manivelas" na direção X , Y ou Z são definidas em função do resíduo sorteado R_i . Portanto, uma vez determinada a planaridade dos quatro pontos $R_i; R_{i+1}; R_{i+2}; R_{i+3}$ consecutivos, passamos à tentativa de realizar um dos possíveis movimentos do tipo manivela. Para isso definimos inicialmente os vetores $\vec{P} = \vec{R}_{i+3} - \vec{R}_i$ e $\vec{Q} = \vec{R}_{i+1} - \vec{R}_i$. Agora, pela propriedade

do produto vetorial, temos que o vetor $\vec{V} = \vec{P} \times \vec{Q}$ é perpendicular a ambos vetores, $\vec{P}$ e $\vec{Q}$, e que as componentes de $\vec{V}$ nas coordenadas locais são: $V_x = (P_y Q_z - P_z Q_y)$, $V_y = (P_z Q_x - P_x Q_z)$, e $V_z = (P_x Q_y - P_y Q_x)$.

O movimento ocorre a partir do sorteio, ao acaso, de três possibilidades de ocorrência de movimento simultâneo dos resíduos R_{i+1} e R_{i+2} (descartando a sobreposição):

- Primeira possibilidade, rotação de 90° : $\vec{R}'_{i+1} = \vec{R}_i + \vec{V}$; and $\vec{R}'_{i+2} = \vec{R}_2 + \vec{V}$;
- Segunda possibilidade, rotação de -90° : $\vec{R}'_{i+1} = \vec{R}_i - \vec{V}$; and $\vec{R}'_{i+2} = \vec{R}_2 - \vec{V}$;
- Terceira possibilidade, rotação de 180° : $\vec{R}'_{i+1} = \vec{R}_i - \vec{Q}$; and $\vec{R}'_{i+2} = \vec{R}_2 - \vec{Q}$;

(ii)- *kink*: o movimento de *kink*, movimento de "prega" ou "quina" é executado quando o resíduo R_i sorteado e seus vizinhos R_{i-1} e R_{i+1} satisfazem a seguinte condição $(\vec{R}_{i+1} - \vec{R}_{i-1})^2 = 2$, isto é: distância ao quadrado entre os resíduos R_{i-1} e R_{i+1} é igual a 2. Isto satisfeito, a tentativa de movimento de *kink* é então processada sobre o resíduo R_i por meio da seguinte transformação: $\vec{R}'_i = \vec{R}_{i-1} + (\vec{R}_{i+1} - \vec{R}_i)$

(iii)- *end*: o movimento de *end* ou movimento dos "terminais" da cadeia, ocorrem quando o primeiro monômero, R_1 , ou o último, R_N , é sorteado. Assim um vetor $\vec{P}$ é construído como sendo $\vec{P} = \vec{R}_2 - \vec{R}_1$, no caso de ser sorteado o vetor R_1 , e $P = \vec{R}_N - \vec{R}_{N-1}$, no caso de ser sorteado o vetor R_N . Indiferentemente do extremo sorteado, o vetor $\vec{P}$ terá um dos seis possíveis resultados: $\vec{V}_1 = (1, 0, 0)$; $\vec{V}_2 = (-1, 0, 0)$; $\vec{V}_3 = (0, 1, 0)$; $\vec{V}_4 = (0, -1, 0)$; $\vec{V}_5 = (0, 0, 1)$, ou $\vec{V}_6 = (0, 0, -1)$. Logo, a nova posição de um dos extremos da cadeia será determinada pelos seguintes passos:

- Se o movimento se refere ao resíduo R_1 :
 - Escolha aleatoriamente um dos seis vetores acima listados, obtendo, digamos, $\vec{V}_j$;
 - Se o vetor escolhido $\vec{V}_j = \vec{P}$, volte ao passo anterior; do contrário a nova posição de R_1 será $\vec{R}'_1 = \vec{R}_2 + \vec{V}_j$
- Se o movimento se refere ao resíduo R_N :
 - Escolha aleatoriamente um dos seis vetores acima listados, digamos $\vec{V}_j$;
 - Se o vetor escolhido $\vec{V}_j = \vec{P}$, volte ao passo anterior; do contrário a nova posição de R_N será $\vec{R}'_N = \vec{R}_{N-1} + \vec{V}_j$

Note que para os casos de redes reais, em que não são permitidas superposições de resíduos, um teste será feito e a nova configuração somente será aceita para o passo seguinte se nenhuma superposição ocorrer. Se o teste falhar, a antiga configuração é mantida e todo o processo de tentativa de gerar uma nova configuração deve ser repetido.

4.4.3 Validação da sequência de configurações geradas

Alguns testes podem ser feitos para verificar se as configurações geradas são fisicamente válidas. Para conferir a ocorrência de algum erro no processo de geração configuracional, listamos a seguir três testes que, se satisfeitos, o processo de geração é validado. Este teste será feito somente uma vez, no período de checagem do programa.

- $1 \leq D_{pp} \leq N - 1$: distância ponta-a-ponta da cadeia, D_{pp} , deve pertencer ao intervalo $[1 - N - 1]$
- $D_{i,i+1}^2 = (\vec{R}_{i+1} - \vec{R}_i)^2 = 1$: quadrado da distância entre monômeros consecutivos, $D_{i,i+1}^2$, deve ser igual a 1: Qualquer outro valor para $D_{i,i+1}^2$ resultante é indicativo de erro.
- $D_{i,i+2}^2 = (\vec{R}_{i+2} - \vec{R}_i)^2 = 2$, ou 4: quadrado da distância entre os monômeros R_i e R_{i+2} deve ser igual a 2 ou 4. Qualquer outro valor para $D_{i,i+2}^2$ é indicativo de erro.

4.4.4 Parâmetros que descrevem e controlam a cadeia

A cadeia polimérica é descrita e controlada ao longo de sua evolução por alguns parâmetros, vetores, tabelas e matrizes, como descrito a seguir

- Tamanho da cadeia é fixada pelo número N de resíduos;
- A distância entre resíduos consecutivos da cadeia é fixa e igual ao parâmetro de rede λ (lattice), estabelecido aqui como $\lambda = 1$;
- A menor distância entre dois resíduos em contato topológico (não conectados) também é λ ;
- Cada resíduo R_i é localizado no espaço por suas coordenadas, memorizadas na forma de um matriz $N \times 3$: $R(i, 1)$; $R(i, 2)$; e $R(i, 3)$ memorizam as coordenadas $x(i)$; $y(i)$ e $z(i)$ do resíduo R_i ;
- O *status* dos vizinhos de cada resíduo são guardados na forma de uma matriz $N \times 6$.
 $M(i, 1)$ = status do sítio $x(i) + 1, y(i), z(i)$; $M(i, 2)$ = status do sítio $x(i) - 1, y(i), z(i)$; $M(i, 3)$ = status do sítio $x(i), y(i) + 1, z(i)$; $M(i, 4)$ = status do sítio $x(i), y(i) - 1, z(i)$; $M(i, 5)$ = status do sítio $x(i), y(i), z(i) + 1$; $M(i, 6)$ = status do sítio $x(i), y(i), z(i) - 1$. O status de um sítio pode ser:
 - ocupado por um resíduo vizinho da cadeia: $M(i, j) = -1$
 - ocupado por um vizinho topológico: $M(i, j) = +1$;
 - sítio vazio: $M(i, j) = 0$;

4.4.5 Cálculo de Energia de uma dada configuração da cadeia

Após cada nova configuração gerada, a energia da mesma precisa ser calculada. Portanto, após um movimento válido de um ou de mais monômeros, são formados novos contatos topológicos com seus vizinhos da cadeia, assim como outros contatos podem ser desfeitos.

Como as interações são em geral, no presente caso, de curto alcance, envolvendo somente os primeiros vizinhos da rede, o cálculo da energia da nova configuração somente precisa considerar os monômeros envolvidos na mudança conformacional. Generalizações, envolvendo segundos vizinhos, por exemplo, podem ser feitas introduzindo matrizes energéticas específicas adicionais.

Se pensamos na cadeia de monômeros teremos o seguinte padrão para o armazenamento de energia (seguindo a abstração para modelo computacional) e das informações necessárias para o cálculo:

- O número da semente a qual gerou a cadeia de monômeros (valor numérico inicial utilizado, chamada de Isem);
- O tamanho da cadeia de monômeros ou Sítios [tamanho deve ser maior que 1];
- O conjunto de pontos cartesianos da configuração ao longa da cadeia de monômeros. (obedecendo o tamanho de Sítios);
- O tipo de cadeia utilizada;
- Estrutura de classificação dos vizinhos topológicos;
- Valores contidos residualmente dentro da matriz energética;
- Valor energético da configuração da cadeia de monômeros;

Portanto, cada monômero é representado por uma estrutura chamada de “*Point*”. Por sua vez, cada ponto possui uma variável dimensionável chamada “*e*”, a qual representa o valor energético devido aos contatos topológicos. Em qualquer configuração da cadeia, cada monômero possui até seis outros monômeros vizinhos (a coordenação da rede cúbica é seis). Assim, até cinco contatos topológicos são possíveis para um monômero dos extremo da cadeia, enquanto quatro contatos topológicos são possíveis para os demais monômeros.

A Figura 7 mostra a classificação geral de resíduos, onde: (i) os pontos enumerados por 1 e 3 são vizinhos **não-topológicos** em relação ao ponto enumerado por 2 (as duas ligações entre eles representam ligações covalentes que unem os monômeros da cadeia); (ii) os pontos enumerados por 3 e 6 respectivamente formam contato **topológico** (representam interações energéticas mais fracas).

As energias decorrentes de cada contato topológico que cada monômero (resíduo) faz com cada um de seus vizinhos são armazenadas individualmente numa matriz $N \times 6$, conforme o exemplo a seguir:

- Cadeia de monômeros, 11^o posição, contato topológico na posição 1, valor energético igual 0.056;
- Cadeia de monômeros, 10^o posição, contato topológico na posição 2, valor energético igual -1.76;

Figura 7 – Ilustração de uma simples cadeia mostrando um contato topológico, entre resíduo "3" e resíduo "6".

Portanto teremos, o monômero rotulado por 10, que ocupa a 11^o posição na cadeia (lembrando que a contagem inicializa-se em zero), possui dois contatos topológicos de primeiros vizinhos; o primeiro contato é feito com o primeiro monômero da cadeia (rotulado por "0") cuja energia de interação é "0.056", e o segundo, fazendo contato com o resíduo "1", tem energia "-1.76".

Para efetuarmos o cálculo da energia, ou da variação de energia de cada novo contato, fazemos uso de uma matriz rotulada como matriz energética. Trata-se de uma matriz quadrada $N \times N$, na qual constam as energias de todos os possíveis contatos entre todos os $N \times (N - 1)/2$ pares de contatos entre resíduos possíveis. Estas energias são previamente calculadas e devidamente gravadas. Assim, a energia de interação entre os resíduos "i" e "j" é armazenada na posição (i,j) na forma $m(i, j) = -3, 14$, por exemplo.

Os valores contidos na matriz energética dependem do modelo considerado, isto é, do potencial de interação entre os resíduos. Por exemplo, os elementos $m(i, j)$ podem ser os mesmos, indiferentemente do par (i, j), que representa o caso em que todos os resíduos são idênticos. Se for o caso do modelo cuja energia de interação entre os pares de resíduos é fixa, porém randômica, a matriz conterá tais valores randômicos. No caso de um heteropolímero, podemos empregar o potencial aditivo, ou seja, $m(i, j) = h_i + h_j$, onde h_i é o nível de hidrofobicidade específica do resíduo "i".

Para a conveniência de testes, a matriz energética pode conter, por exemplo, o valor para qualquer que seja o par de monômeros, que corresponde a uma cadeia em que todos os resíduos

são idênticos. Assim, para o caso de atração entre pares de tal resíduo ($m(i, j) = c$ para todos os pares (i, j)) o resultado da simulação pode ser usado como teste de validação do programa, pois espera-se que haja um colapso absoluto da cadeia para temperaturas relativamente baixas. Outro caso similar é aquele em que todos os resíduos se repelem, quando esperamos que o resultado da simulação revele uma cadeia essencialmente aberta ao longo da janela de simulação, sem contatos topológicos, praticamente, a baixas temperaturas..

4.4.6 Algoritmo para cálculo de energia após um movimento válido

O algoritmo para cálculo de energia após movimento ocorre da seguinte forma:

- Recorre-se à classificação de todos os monômeros da cadeia, em relação aos seus vizinhos através da matriz de vizinhos, digamos, M , que são de três categorias: (a) quando o vizinho j do monômero i é ocupado por um monômero pertencente à cadeia: $M(i, j) = -1$; (b) quando seu vizinho j trata-se de um vizinho topológico: $M(i, j) = +1$; e (c) quando o vizinho j é um sítio vazio: $M(i, j) = 0$. Somente os casos (b) acima, isto é, quando ocorrer um vizinho topológico, é que será feito cálculo da variação da energia do sistema.
- Cálculo da variação da energia: recorre-se aos valores correspondente às interações pertinentes na matriz energética, adicionando os valores respectivos aos novos contatos topológicos e subtraindo aqueles dos contatos topológicos desfeitos, pelo movimento ocorrido.

4.5 O fator temperatura e a constante de Boltzmann

Ludwig Eduard Boltzmann foi um físico austríaco conhecido pelo seu trabalho no campo da termodinâmica estatística. Em conjunto com Josiah Willard Gibbs e James Clerk Maxwell é considerado o fundador da Mecânica Estatística. Um ponto interessante aqui é que Boltzmann foi defensor da teoria atômica numa época em que esta se tratava de uma hipótese bem controversa.

Sua grande contribuição foi conseguir visualizar um método probabilístico para estimar a entropia do gás ideal (isto em meados de 1877). Em sua abordagem, ele definiu entropia como proporcional ao logaritmo natural do número de microestados que um gás pode ocupar:

$$S = -k_B \ln \Omega, \quad (4.5)$$

onde S é a entropia, Ω o número de microestados possíveis para o sistema acessar, e k_B a constante que leva seu nome, a constante de Boltzmann; tal constante, em muitos contextos, é também representada por k simplesmente. Trata-se da constante física que relaciona temperatura e energia cinética das moléculas; $k_B T$ tem dimensão de energia e é proporcional a energia cinética média de um átomo ou qualquer molécula constituinte de um meio à temperatura T . (MOHR PETER J.; TAYLOR, 2008)

4.5.1 O peso de Boltzmann

O produto $k_B T$ indica a ordem de grandeza da energia cinética média das moléculas de qualquer corpo à temperatura T . transições entre microestados de um sistema físico podem ser ponderados quanto a sua ocorrência ou não, por meio do fator $k_B T$. Para o problema deste trabalho, por exemplo, quando uma nova configuração é gerada, pode ocorrer que a mesma seja energeticamente inviável. Pelo princípio da "Mínima Energia Potencial", que estabelece a tendência de qualquer sistema físico evoluir para configurações de energia potencial cada vez mais baixas (trata-se do mesmo princípio que explica a razão dos corpos materiais "caírem" em direção ao centro da terra: redução da energia potencial gravitacional). Ocorrida uma nova configuração, em nosso sistema, a energia da mesma deverá ser determinada e submetida ao princípio mencionado. Se sua energia for menor do que a da configuração que a precedeu, a nova configuração é prontamente aceita. Porém se sua energia for maior, ela poderá ainda ser aceita, contudo sob uma condição probabilística (o fator entrópico entra aqui em ação), e é neste momento que entra em jogo o "peso P_c de Boltzmann". Seu significado é o de "Probabilidade", isto é, probabilidade de ocorrência de uma dada configuração c com energia E_c , e é expressa matematicamente como (KITTEL, 1996)

$$P_c = A \exp(-E_c/k_B T), \quad (4.6)$$

onde E_c é a energia da configuração c , e A é uma constante que normaliza a probabilidade P_c no intervalo $[0,1]$,

$$A = 1 / \sum_{\{c\}} \exp(-E_c/k_B T) \quad (4.7)$$

onde $\{c\}$ indica que a soma é sobre o conjunto de todas as possíveis configurações da cadeia. Assim, o peso de Boltzmann é função somente da temperatura e da energia E_c da configuração, isto é, $P_c = P_c(T, E_c)$

4.5.2 O algoritmo de Metropolis

Este algoritmo, para o caso de simulações de cadeias heteropoliméricas, foi já introduzido antes, mas aqui vamos repeti-lo frisando que o mesmo é geral, isto é, funciona para qualquer sistema físico. Sua essência está na decisão de como as configurações do sistema se sucedem. Assim, para sistemas termodinâmicos, governados pelo princípio da "mínima energia potencial", quando uma nova configuração com energia E_n é gerada a partir de uma configuração anterior com energia E_a , a decisão sobre sua aceitação, ou não, segue sempre os seguintes passos:

- Sempre que $E_n \leq E_a$, a nova configuração é aceita e a energia do sistema, assim como todas eventuais variáveis do problema são atualizadas. Do contrário, um número randômico R uniformemente distribuído entre 0 e 1 é gerado e ao mesmo tempo uma probabilidade de transição $T_{a \rightarrow n} = \exp(-\Delta E/k_B T)$ é calculada:

- Se $R \leq T_{a \rightarrow n}$, a nova configuração é aceita e a energia do sistema é atualizada, assim como todas eventuais variáveis do problema;
- Se $R > T_{a \rightarrow n}$, a configuração anterior é mantida
- Um nova configuração é gerada, e volta-se ao primeiro passo acima.

Esta sequência de condições para o aceite/rejeite de uma nova configuração constitui-se numa amostragem ponderada energeticamente, envolvendo o princípio de mínima energia (*importance sampling*). A condição probabilística estabelecida pelo peso de Boltzmann é conhecida como Algoritmo de Metropolis; sua elaboração envolve o conceito de balanço detalhado ([METROPOLIS ARIANNA W. ROSENBLUTH; TELLER, 2016](#)).

Segue abaixo a implementação ilustrativa do Algoritmo de Metropolis, na linguagem C#, aqui utilizado:

Código-fonte 6: Exemplo do código fonte em C# do Algoritmo de Metropolis

```

1 public class Temperature
2     {
3         public static bool Do()
4         {
5             double kt = Convert.ToDouble( Structs.Environment.k *
AppConfigClient.Param.dataToProcess.temperature );
6             double En = GCPS.tempChain.U;
7             double Ea = GCPS.chain.U;
8             double deltaE = En - Ea;
9
10            if ( deltaE <= 0 )
11            {
12                return true;
13            }
14            else if ( deltaE > 0 )
15            {
16                Randomic.Random();
17                double r = Randomic.randu;
18
19                if ( r <= MathExtesion.Pow(-deltaE / kt , 2) )
20                    return true;
21                else
22                    return false;
23            }
24

```

```
25         return false ;  
26     }  
27 }
```

PLATAFORMA DE SIMULAÇÃO DISTRIBUÍDA

Os exemplos de aplicação da plataforma desenvolvida constituem aqui dois grupos de experimentos simulacionais: (i) testes de validação do programa Monte Carlo desenvolvido, e (ii) aplicação em alguns casos ilustrativos de interesse físico, com o foco em heteropolímeros. Em relação à versão legada (programa de simulação que inspirou este trabalho), esses exemplos também visam ilustrar como podemos modernizar e agilizar o procedimento simulacional, através de uma plataforma que suporta computação distribuída.

5.1 Desenvolvimento de uma plataforma de simulação computacional distribuída, intuitiva, versátil e segura

O processo de simulação envolve etapas repetitivas, as quais tomam tempo significativo de máquina que pode ser reduzido. Com a conquista deste ponto, ou seja, a redução do tempo de processamento, surge uma outra demanda: o tratamento massivo de dados, que gera necessidade computacional extra e de nova natureza. E, no presente caso, também envolve a transmissão de dados pela Internet. Assim, com o intuito de otimizar todos estes processos, a plataforma simulacional apresentada aqui contém uma série de atributos distintos:

- Possibilidade de executar múltiplas simulações simultâneas através da concorrência entre máquinas;
- Entrega dos resultados já pré-formatados;
- Memorização das "trajetórias" das simulações efetuadas para a posterior utilização para análises (feitas por programas extras que constituem o sistema), e por outros eventuais usuários da plataforma.
- Interface amigável e intuitiva para utilização da aplicação;

- Criptografia dos resultados obtidos;

Interface amigável e intuitiva para utilização da aplicação

A aplicação legada em uso foi escrita na linguagem FORTRAN. Trata-se de uma linguagem tradicionalmente usada por cientistas e engenheiros durante muitas décadas. Com foco em aplicações científica e técnica, tem sido adotada especialmente devido a sua eficiência para cálculos matemáticos avançados. Entretanto, para Cientistas da Computação esta linguagem mostra-se deficitária devido a suas limitações de generalidades, como por exemplo, a não disponibilidade de uma plataforma distribuída nativa, e por não ter um *front-end* (terminal) interativo com o usuário. O FORTRAN é a mais antiga das linguagens de alto nível; foi desenvolvida pelo grupo IBM no final da década de 1950. Ela é a abreviação de FORMula TRANslation (ou originalmente, IBM Mathematical FORMula Translation System).

Analisando o cenário nos laboratórios de simulação, se percebe que os pesquisadores que dependem da utilização da aplicação de programas computacionais, em geral conhecem o básico em relação a Tecnologia da Informação. Portanto, são pessoas com algum conhecimento de informática, mas sem possibilidades técnicas mais abrangentes para efetuar eficientemente grandes alterações, ou mesmo para automatizar processos manuais, como é necessário no tratamento massivo de dados.

Neste sentido, a proposta deste trabalho é também proporcionar um exemplo das possibilidades tecnológicas presentes, e também auxiliar naqueles problemas fora do foco central do interesse científico do pesquisador. Por exemplo, na aplicação legada, o pesquisador executa sequencialmente o seu programa de simulação, caso a caso, através do maquinário local: os *inputs* dos dados ocorrem manualmente e em etapas para o início da simulação, assim como no recolhimento e tratamento dos resultados para visualização. São processos que consomem muito tempo e podem gerar dificuldades extras, tendo em vista a necessidade de conhecimento específico da particular linguagem em uso. Contudo, nesta nova Interface estes e outros problemas são mitigados por meio da utilização do conceito de *WebSite*.

Um *Website* (DICIONÁRIO..., 2008) é um conjunto de páginas (isto é, conteúdos em mídia eletrônica) acessadas através da Internet. Este *Website* promove e mantém um processo interativo com o usuário que não necessita de conhecimento mais profundo sobre a Tecnologia da Informação envolvida. A proposta básica deste conceito é fornecer ferramentas de automação, produtividade e compartilhamento, substituindo aplicações criadas para rodar em *desktop* disponíveis (*stand-alone*).

Diferenças e melhorias para o usuário. Com a evolução da Internet nas últimas duas décadas, aplicações criadas para a *web* passaram de simples “páginas estáticas” para grandes aplicações corporativas. Não demorou muito para se tornar um padrão de mercado. As aplicações escritas para *Desktop* (famosos *Client/Server*) foram substituídas pelo modelo *Web*. Entre várias razões para esta mudança, podemos listar:

- Acessibilidade – A partir de qualquer lugar no mundo, que possua conexão com a Internet, disponível vinte quatro horas por dia e sete dias por semana;
- Flexibilidade para Clientes – Não precisam de capacidade de processamento e nem instalação local para executar a aplicação;
- Execução distribuída – Diferentes partes da aplicação são executadas em diferentes máquinas;
- Escalabilidade – Infra-estrutura escalável para suportar números crescentes de acessos simultâneos;
- Manutenção e atualização – Novas versões de sistemas são atualizados nos servidores, propagando-se para o mundo, sem a necessidade de atualização em máquinas de usuários;
- Multiplataforma – Comunicação e apresentação baseadas em padrão de conteúdo da web, independente de plataforma, navegador ou do sistema operacional utilizado;

A Figura 8 mostra uma imagem extraída do *front-end* da aplicação já desenvolvida. Este será a "porta de entrada" (*login*, cadastro) para a utilização da ferramenta proposta neste trabalho. Embora o processamento das simulações ocorram em *background* (em segundo plano), o usuário contará com esta interface de acessibilidade, a qual não limita o usuário com exigências de conhecimentos específicos em qualquer linguagem computacional.

The image shows a web browser window with the title 'Log In'. The address bar shows the URL 'efolding.fcfrp.usp.br/Account/Login.aspx?ReturnUrl=%2f'. The browser's navigation bar includes 'Home', 'Simulations', and a '[Log In]' button. Below this is a dark blue header with the text 'Grid Protein Folding'. The main content area has a white background with the heading 'Log In'. Below the heading is a line of text: 'Please enter your username and password. [Register](#) if you don't have an account.' This is followed by a section titled 'Account Information' with a horizontal line separator. Under this section, there are two input fields: 'Username:' and 'Password:'. Below the password field is a checkbox labeled 'Keep me logged in'. At the bottom of the form is a 'Log In' button.

Figura 8 – Interface de acessibilidade da ferramenta

Resumindo, esta ferramenta não exige conhecimentos específicos em Tecnologia da Informação para ser utilizada, tendo em vista que a Interface de acesso está em conformidade com o padrão utilizado pelo milhões de páginas da Internet.

5.2 Possibilidade de executar mais de uma simulação em concorrência de máquina

A computação paralela faz uso de múltiplos núcleos de processamento simultaneamente para resolver um problema. Por exemplo, no problema do *folding* de proteínas, temos várias situações que possibilitam o processamento em paralelo: variáveis parametrizáveis que são informados para as simulações, como, por exemplo, a temperatura, mutações introduzidas no modelo, e o tamanho da cadeia polimérica. Podemos efetuar a mesma simulação com 27, 36, 80 ou 100 monômeros, em diferentes temperaturas, resultando num grande espectro de resultados. Na aplicação legada, deveríamos executá-la e tratar cada uma das várias simulações decorrentes individual e sequencialmente.

Figura 9 – Elementos do sistema de processamento distribuído; veja descrição no texto que se segue.

De uma forma geral, o que se faz é subdividir o problema em partes independentes de modo que cada núcleo de processamento pode executar sua parte do algoritmo simultaneamente com outros núcleos, conforme ilustração que se segue (Figura 9):

- **Layer (camada) de Servidor** - Correspondem ao servidor de Banco de Dados (*Database Server*), Servidor de Aplicação (*Application Server*) e Servidor de Web Sites (ou *Web Server*) os quais hospedam toda a aplicação;
- **Camada de Software colaborativo (ou groupware)** (MISTRÍK IVAN; GRUDY, 2010) - São as máquinas as quais possuem instalação da aplicação cliente, a qual efetua processamentos e cálculos pertinentes a uma simulação. Note que a topologia de rede está aberta, pois podemos contar com máquinas dentro de uma sub-rede proprietária (por exemplo:

USP), assim como com qualquer uma de outras máquinas que possuem o software de simulação instalado e conectada através da Internet. Note que existe uma *layer* chamada "*Application*"; esta *layer* é implícita para os usuários e para os resultados simulacionais, pois, além da possibilidade de executarmos simulações simultâneas em *groupware*, internamente a aplicação trabalha utilizando o conceito de *Multi-Threads*, que é a capacidade de uma unidade de processamento central (CPU) compartilhar os recursos de um único, ou vários núcleos. Em outras palavras, *Multi-Threads* é a capacidade de executar várias tarefas simultaneamente. Exemplo: Processamentos aritméticos, acessos ao disco e outros. (NEMIROVSKY MARIO; TULLSEN, 2013).

- **Camada de Usuário** - A camada de usuário, para esta aplicação, descreve a utilização de *softwares* para simulação, onde o usuário/pesquisador através de seu computador pessoal ou corporativo utiliza a Interface *Web* (*Web Site*) para inserir parâmetros em sua futura simulação.

APLICAÇÕES E RESULTADOS

Para nossas aplicações Unitárias e Funcionais, utilizamos seis valores distintos para a temperatura do banho térmico; respectivamente $T = 0,3; 0,9; 1; 2; 3$ e 5 . Essas temperaturas são expressas na métrica em que $\Delta E/k_B T \approx 1$. É ainda importante frisar que comumente, em simulações MC, trabalhamos com um sistema de variáveis reduzidas, o que possibilita simplificações nas notações. Assim, para muitos propósitos, como neste presente contexto, consideramos o valor $T = 1$ como sendo 300 Kelvin (aproximadamente 27°C). Logo, cada variação na temperatura de $0,1$ equivale a 30 Kelvin, ou 30°C . Seguem resultados obtidos para três modelos de cadeias aqui utilizadas: cadeia **polar**, **apolar** e modelo **HP**.

6.1 Cadeia Polar

Como o nome sugere, uma cadeia polar é constituída por moléculas polarizadas (presença de dipolos elétricos permanentes, ou seja, moléculas com distribuição assimétrica de cargas elétricas). A água é o exemplo mais usual de molécula polar ([DILL KEN A.; BROMBERG, 2003](#)). No presente caso, assumimos uma cadeia com N unidades idênticas, todas polares. Desta forma as interações entre pares de resíduos forma uma matriz, denominada aqui por 'matriz energética' cujos elementos $m_{i,j}$ são todos positivos; especificamente, $m_{i,j} = +1$ para qualquer par (i,j) , isto é: qualquer contato entre elementos da cadeia é sempre repulsiva (aumenta a energia do sistema de $+1$). As simulações realizadas aqui utilizam cadeias com 27 resíduos, portanto todas as interações de pares possíveis são representadas por uma matriz de 27×27 elementos, cuja diagonal principal $m_{i,i} = 0$ (com energia de auto-interação nula).

A Figura 10 apresenta o formato de saída (planilha Excel) dos resultados de uma simulação que empregou um total de 13.500.000 passos MC elementares para cinco diferentes temperaturas. Podemos usar outra unidade de tempo MC, que chamaremos simplesmente de "passos MC" quando normalizados pelo tamanho da cadeia. No caso, com cadeia de 27 monômeros, 13.500.000 passos MC elementares corresponde a 500.000 passos MC.

A tabela da esquerda mostra numericamente a evolução (ao longo da simulação) do valor médio do **quadrado do raio de giração** (raio de giração elevado ao quadrado), rotulado por **rg2**; e a tabela da direita mostra a evolução da energia média do sistema, representada por **<U>**. A coluna 'idx' (das duas tabelas) indexa o números de amostra ao longo da simulação. São mostrados somente os 15 primeiros dos 10.000 "instantâneos" da simulação registrados.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
	idx	T 0,3	T 0,9	T 1,0	T 2,0	T 3,0	T 5,0		idx	T 0,3	T 0,9	T 1,0	T 2,0	T 3,0	T 5,0	
	0	5,504801097	5,274348422	5,274348422	5,159122085	5,159122085	5,159122085		0	1,000000	2,000000	2,000000	4,000000	4,000000	4,000000	
	1	7,137174211	7,839506173	7,847736626	8,658436214	8,869684499	5,315500686		1	0,500000	1,500000	1,000000	3,000000	2,500000	3,500000	
	2	7,480109739	8,123914037	8,08001829	7,74256973	10,71284865	6,411979881		2	0,666667	1,333333	0,666667	2,666667	2,333333	3,000000	
	3	7,9739369	8,032921811	9,134430727	7,744855967	10,56584362	6,083676269		3	0,500000	1,250000	1,000000	2,250000	2,000000	2,750000	
	4	7,951989026	7,573388203	9,02085048	7,645816187	9,786282579	6,565980796		4	0,400000	1,200000	1,000000	3,400000	3,200000	2,400000	
	5	8,790580704	7,722450846	8,927754915	8,051668953	9,551897577	6,77046182		5	0,333333	1,000000	1,166667	2,833333	2,833333	2,166667	
	6	8,411326671	7,960611405	8,944405252	8,177738585	10,68214776	6,641387419		6	0,285714	1,142857	1,142857	2,714286	2,857143	2,428571	
	7	8,447187929	7,520919067	8,718106996	8,255144033	11,23388203	6,586419753		7	0,250000	1,125000	1,375000	2,500000	2,500000	2,250000	
	8	8,142813595	7,690748362	8,237921049	8,810394757	10,71041	6,489102271		8	0,222222	1,000000	1,444444	2,444444	2,777778	3,000000	
	9	8,018106996	7,639780521	7,891906722	8,640329218	10,47297668	6,427434842		9	0,200000	0,900000	1,500000	2,500000	2,700000	3,400000	
	10	8,01334331	7,869185684	7,85147774	8,255518144	10,97555805	6,580995137		10	0,181818	0,818182	1,454545	2,272727	2,636364	3,181818	
	11	8,158664838	7,708962048	7,653177859	8,15386374	11,23616827	6,839734797		11	0,166667	0,916667	1,333333	2,250000	2,583333	3,166667	
	12	8,114909782	8,01994302	7,59280363	8,166191833	10,7501319	6,931201857		12	0,153846	0,846154	1,307692	2,153846	2,692308	3,153846	
	13	8,090339016	8,100921027	7,533803645	7,838526357	10,53497942	7,47148736		13	0,142857	0,928571	1,428571	2,285714	2,785714	3,071429	
	14	8,217192501	8,17219936	7,826886145	8,092272519	10,55518976	7,486694102		14	0,133333	0,933333	1,400000	2,133333	2,666667	3,133333	
	15	8,409979424	8,090534979	8,448388203	8,051783265	10,4686214	7,282750343		15	0,125000	0,937500	1,312500	2,000000	2,750000	3,312500	

Figura 10 – Simulação MC de uma cadeia **polar** para cinco temperaturas distintas. As tabelas ilustram a forma de saída dos dados: a tabela da esquerda mostra a evolução do **quadrado** do raio de giração (**rg2**), e a da direita mostra a evolução da energia do sistema (**U**) . São mostrados 16 instantâneos da parte inicial da simulação.

De acordo com o "Algoritmo de Metropolis", quando uma nova configuração com energia E_n é gerada a partir de uma configuração anterior com energia E_a , a decisão sobre sua aceitação, ou não, quando $E_n > E_a$, depende do peso estatístico de Boltzmann, isto é: a transição ocorrerá com probabilidade $\exp(-(E_a - E_n)/k_B T)$. Como no presente caso, todo contato é repulsivo (isto é, qualquer contato entre monômeros aumenta a energia do sistema), a chance de haver um contato, mesmo aumentando a energia do sistema, é função somente da temperatura e decresce com a redução da mesma. Logo, para baixas temperaturas ($T = 0,3$) espera-se que $\langle rg2 \rangle$ seja maior do que para temperaturas altas ($T = 5$, por exemplo), quando as chances de haver contatos entre os resíduos é muito maior. De fato, como mostra os resultados da simulação, Figura 11, para as cinco temperaturas, o valor de equilíbrio de $\langle rg2 \rangle$ cresce sistematicamente com a redução da temperatura. Na temperatura $T = 5$, na escala usada aqui, que se aproxima do limite $T \rightarrow \infty$, a ocorrência de pares com e sem contatos ocorrem com praticamente a mesma probabilidade. O resultado obtido no regime de equilíbrio para o raio de giração $R_G = \sqrt{\langle rg2 \rangle} \approx 8,2$, Figura 11, se aproxima do resultado para o modelo *freely rotating chain*, que prevê o resultado exato $R_G = \sqrt{27} \approx 5,2$ para ângulo de 90° entre as ligações que unem 3 unidades consecutivas da cadeia. Este modelo, formando configurações somente com *kinks*, se aproxima do modelo em rede simulado aqui, mas com R_G funcionando como um *lower bound* para o presente modelo em rede.

O comportamento de $\langle U \rangle$ ao longo da simulação, por sua vez, mostra que a energia do sistema aumenta com o aumento da temperatura. De fato, sem praticamente nenhum contato, para temperatura baixa, $T = 0,3$, a energia fica perto de zero, e para $T = 5$, com chances muito maior de contatos, a energia será maior, pois cada contato aumenta a energia do sistema; ver

gráfico à esquerda da figura 11.

Figura 11 – Resultados da simulação (para cinco diferentes temperaturas) de uma cadeia **polar** de 27 unidades. São mostrados (à esquerda) a evolução (ao longo da simulação) da raiz quadrada da média do raio de giração ao quadrado $\langle rg^2 \rangle$, e a evolução da média da energia potencial $\langle U \rangle$. As curvas foram construídas usando 10.000 instantâneos igualmente espaçados, cerca de 2% do total de passos MC usados na simulação. Assim, o tempo MC = 10.000 corresponde a 500.000 passos MC

6.2 Cadeia Apolar

Substâncias apolares são também denominadas "hidrofóbicas", porque não se misturam com a água (que é polar). Material apolar, como o próprio nome indica, não apresenta moléculas polarizadas, ou seja, em tais materiais, as moléculas não apresentam distribuição assimétrica de cargas elétricas. Exemplos corriqueiros incluem, óleos vegetais, parafina, tetracloreto de carbono e gasolina. (DILL KEN A.; BROMBERG, 2003)

Neste caso, diferentemente do caso anterior, a matriz energética é constituída por elementos todos negativos, ou seja, $m_{i,j} = -1$ para qualquer par (i,j) . Aqui novamente a probabilidade da ocorrência de um contato entre os resíduos depende somente da temperatura, pois eles são energeticamente equivalentes, embora, ao contrário do caso anterior, os contatos agora diminuem a energia do sistema, o que favorece a compactação da cadeia. De fato, a energia do sistema diminui proporcionalmente ao número de contatos, pois para cada contato a energia do sistema, conforme a Equação 4.4, $e_{m,n} = (R_m + R_n)/2$, é reduzida por -1, para qualquer par (m,n) .

Assim, esperamos que tanto a energia quanto o raio de giração do sistema diminuam com a diminuição da temperatura. A Figura 12 e 13 mostram os resultados obtidos após uma simulação envolvendo 13.500.000 passos MC elementares, ou 500.000 passos MC. A compactação da cadeia é inicialmente rápida para temperaturas $T \leq 1$.

Particularmente, notamos que para $T = 0,3$, depois de um rápido colapso, a cadeia lentamente vai se rearranjando num processo lento que leva a cadeia muito próximo de sua compactação máxima, quando o raio de giração atinge o limite $R_G = \sqrt{2}$. Contudo este valor, como a Figura 13 (lado esquerdo mostra) mostra, não é atingido, o que indica a ocorrência de flutuações conformacionais, mesmo para temperaturas muito baixas.

Outro caso de destaque ocorre para a temperatura $T = 2$. Note que inicialmente a cadeia fica presa numa conformação que determina um raio de giração extremo, contrariando inclusive os valores correspondentes às temperaturas mais altas, $T = 3$ e $T = 5$. Contudo, no tempo MC ao redor de 6.000, o raio de giração começa a se reduzir lenta mas sistematicamente. Neste instante inicia-se uma *transição de fase conformacional*, que vai levar a cadeia a uma maior compactação como indica a tendência da curva. Note que 500.000 passos MC (ou tempo MC = 10.000) não foram suficientes, neste caso, para atingir o regime de equilíbrio termodinâmico e assim, a rigor, a simulação deve ser continuada até que a condição de equilíbrio seja alcançada, com o valor final do raio de giração devidamente se acomodando entre as curvas para a temperatura $T = 1$ e $T = 3$.

As curvas da direita da Figura 13 mostram a evolução da energia potencial média ao longo da simulação. O comportamento de cada uma das duas temperaturas discutidas no parágrafo acima, a saber $T = 2$ e $T = 0,3$, está fortemente correlacionado com os seus respectivos eventos conformacionais relativos ao raio de giração. No caso para $T = 2$, no instante em que a cadeia começa a se compactar, por volta do tempo MC = 6.000, consistentemente a energia potencial

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
	idk	T 0,3	T 0,9	T 1,0	T 2,0	T 3,0	T 5,0		idk	T 0,3	T 0,9	T 1,0	T 2,0	T 3,0	T 5,0	
	0	5,159122085	4,906721536	4,906721536	5,406035665	5,570644719	5,641975309		0	-14,00000000	-13,00000000	-13,00000000	-7,00000000	-8,00000000	-7,00000000	
	1	5,444444444	4,960219479	6,862825789	5,310013717	6,342935528	4,666666667		1	-15,00000000	-11,00000000	-11,00000000	-5,50000000	-5,50000000	-7,00000000	
	2	5,543208977	4,603109282	7,428898034	5,534064929	6,073616827	6,870141747		2	-15,66666667	-11,66666667	-9,66666667	-5,66666667	-7,00000000	-6,33333333	
	3	5,618555693	4,641975309	6,499314129	5,407407407	5,29039781	6,303840878		3	-16,25000000	-12,00000000	-11,75000000	-6,75000000	-9,50000000	-6,25000000	
	4	5,576680384	4,620301783	6,225788752	5,355555556	5,148148148	7,203017833		4	-16,60000000	-12,40000000	-10,80000000	-6,00000000	-10,00000000	-6,40000000	
	5	5,555098308	4,457247371	5,598079561	5,272519433	5,291266575	7,78600823		5	-16,83333333	-12,00000000	-12,50000000	-5,66666667	-9,50000000	-5,50000000	
	6	5,54281795	4,174211248	5,216343327	5,683519498	5,043503821	8,015873016		6	-16,85714286	-12,85714286	-13,00000000	-5,57142857	-9,285714286	-5,71428571	
	7	5,525720165	4,024348422	4,836076818	5,840877915	4,982510288	7,799382716		7	-17,00000000	-13,12500000	-13,50000000	-5,25000000	-9,25000000	-5,37500000	
	8	5,449016918	3,84865112	4,544276787	6,17268709	5,068282274	7,339277549		8	-17,11111111	-13,88888889	-14,33333333	-5,00000000	-9,22222222	-5,33333333	
	9	5,34266118	3,67297668	4,519890261	6,141015089	5,111659808	7,26776406		9	-17,30000000	-14,80000000	-14,10000000	-4,80000000	-9,30000000	-5,30000000	
	10	5,321985285	3,536725277	4,450056117	6,055742611	5,000748223	7,402793366		10	-17,36363636	-15,27272727	-13,90909090	-4,90909090	-9,36363636	-5,09090909	
	11	5,277777778	3,464791952	4,379286694	6,152949246	4,913808871	7,596479195		11	-17,58333333	-15,33333333	-14,00000000	-4,91666667	-9,41666667	-5,00000000	
	12	5,230241638	3,438113327	4,233301678	6,18444655	5,014983645	7,545320249		12	-17,76923077	-15,307692308	-14,230769231	-5,153846154	-9,230769231	-5,00000000	
	13	5,231040564	3,380560455	4,10542818	6,475406624	5,106016069	7,299235744		13	-17,92857143	-15,785714286	-14,428571429	-5,50000000	-9,142857143	-5,214285714	
	14	5,19295839	3,325011431	3,992409694	6,385093736	5,313671696	7,068952904		14	-18,06666667	-16,13333333	-14,86666667	-5,33333333	-8,93333333	-5,40000000	
	15	5,159636488	3,30366941	3,887688615	6,291323731	5,699074074	6,909122085		15	-18,18750000	-15,87500000	-15,18750000	-5,37500000	-8,50000000	-5,56250000	

Figura 12 – Simulação MC de uma cadeia **apolar** para cinco temperaturas distintas. As tabelas ilustram a forma de saída dos dados: a tabela da esquerda mostra a evolução do **quadrado** do raio de giração (rg_2), e a da direita mostra a evolução da energia do sistema (U). São mostrados 16 instantâneos da parte inicial da simulação.

começa a se reduzir de forma lenta mas sistematicamente mas sem atingir o equilíbrio térmico no tempo MC = 10.000.

No outro caso notável, para a temperatura $T = 0.3$, após um rápido colapso da cadeia, ocorre um mínimo da energia potencial ao redor do tempo MC = 100, mas com a cadeia ainda numa conformação *não maximamente compacta*, o que irá ocorrer com um raio de giração $R_G = \sqrt{2}$. A partir daí, a energia volta a subir atingindo um patamar a partir do qual inicia-se um novo processo de lenta redução. Note que o referido mínimo da energia, $U_m = -26$, é maior do que o mínimo global $U_0 = -28$, o que corresponde à energia da configuração maximamente compacta que contém um total de 28 contatos topológicos. Trata-se de um estado metaestável que se desfaz de forma relativamente rápida.

Como se pode ver nestes dois exemplos, o comportamento termodinâmico de uma cadeia apolar (hidrofóbica) em temperaturas próximas a $\Delta E/k_B$, e temperaturas mais baixas, mostra-se mais complexos do que para a cadeia polar, com manifestação de diferentes fases conformacionais ao longo da simulação.

Figura 13 – Resultados da simulação de uma cadeia **apolar** de 27 unidades. São mostrados (à esquerda) a evolução ao longo da simulação da raiz quadrada da média do raio de giração ao quadrado $\langle rg_2 \rangle$, e à direita a evolução da média da energia potencial $\langle U \rangle$, para cinco diferentes temperaturas. As curvas foram construídas usando 10.000 instantâneos igualmente espaçados, cerca de 2% do total de passos MC usados na simulação: tempo MC = 10.000 corresponde a 500.000 passos MC

6.3 Modelo HP

O Modelo **HP** (Hidrofóbico-Polar) é um "mix" dos dois modelos anteriores: a resíduos polares e apolares serão atribuídos valores +1 e -1, respectivamente. Assim, neste caso, a matriz energética possui elementos positivos, negativos e nulos, isto é a interação entre cada par de resíduos pode variar de +1; -1 ou 0. O modelo é estabelecido previamente, atribuindo a cada unidade da cadeia a sua carga hidrofóbica. Para isso, o sistema provê uma rotina que permite o montagem *a priori* da cadeia. Após o *design* da cadeia, o sistema gera a matriz energética (27×27), com a energia $e_{m,n}$ referente a cada par (m,n) de monômero, que pode ser: +1, -1 ou 0 (e com a diagonal principal nula).

No exemplo trabalhado aqui, escolhemos um modelo que nos facilitará observar o resultado de forma visual: À sequência de unidades desde 1 a 12, estabelecemos como inteiramente apolar (H), recebendo então -1 como carga hidrofóbica. As demais unidades, de 13 a 27, serão todas polares (P), isto é hidrofílicas e recebem valor +1. O Quadro 3 mostra a cadeia resultante.

Quadro 3 – Quadro com configuração utilizada para obtenção modelo HP 50-50

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Para este modelo também foram feitas simulações para cinco temperaturas distintas, com 13.500.000 passos MC elementares, ou 500.000 passos MC. Como a cadeia está dividada em um primeiro segmento **apolar** e outro restante **polar**, o que se espera é que a primeira metade da cadeia de monômeros tende a se compactar, e segunda metade, por sua vez, formar uma espécie de "rabicho" distendido.

As Figuras 14 e 15 mostram, no mesmo estilo das anteriores, os resultados da simulação do modelo **HP**. A parte esquerda da Figura 15 mostra que o raio de giração aumenta com o aumento da temperatura, e fica limitado ao intervalo [5,5 - 7,5], enquanto que para a cadeia **apolar** (Figura 13) este intervalo é de [2 - 7,5] e para a cadeia **polar** (Figura 11) é de [8-11]. O limite superior é o mesmo para a cadeia **apolar** e o modelo **HP**, mas é menor do que o limite inferior do R_G para a cadeia **polar**, o que indica que mesmo a temperaturas altas, como $T = 5$, os contatos entre monômeros apolares são favorecidos de forma significativa.

A parte da direita da Figura 15 mostra a evolução da energia $\langle U \rangle$ do sistema, a qual fica restrita ao intervalo de aproximadamente $[(-10) - (-1)]$, contrastando com o intervalo energético referente à cadeia **apolar**, a saber $[(-27) - (-7)]$, pois esta possui mais contatos com energia -1. Para a cadeia **polar**, consistentemente, as energias para todas as temperaturas satisfazem a relação $\langle U \rangle \geq 0$

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
	idx	T 0,3	T 0,9	T 1,0	T 2,0	T 3,0	T 5,0		idx	T 0,3	T 0,9	T 1,0	T 2,0	T 3,0	T 5,0	
	0	4,994513032	4,994513032	4,994513032	5,400548697	5,049382716	5,049382716		0	-10,000000	-8,000000	-8,000000	-6,000000	-8,000000	-8,000000	
	1	6,185185185	4,942386831	5,609053498	5,371742112	7,75308642	5,781893004		1	-11,000000	-8,500000	-5,500000	-4,000000	-4,000000	-4,000000	
	2	6,053497942	4,903978052	6,869227252	4,867398262	9,172839506	6,580246914		2	-11,333333	-8,000000	-5,333333	-3,333333	-3,333333	-2,666667	
	3	6,170096022	5,543895748	7,388888889	5,189986283	9,072016461	6,263374486		3	-11,500000	-7,250000	-6,500000	-2,500000	-3,250000	-1,750000	
	4	6,229080933	6,698216735	7,30617284	5,748971193	10,02222222	6,399176955		4	-11,600000	-6,400000	-6,600000	-2,000000	-2,400000	-1,600000	
	5	6,256515775	7,496113397	7,185185185	5,923639689	9,721993599	7,020118884		5	-11,666667	-5,833333	-6,333333	-1,833333	-2,333333	-1,166667	
	6	5,815990594	8,763276504	7,596511856	6,069958848	9,581226729	7,433078581		6	-11,714286	-5,428571	-6,142857	-1,857143	-2,714286	-1,000000	
	7	5,812757202	9,17558299	7,712277092	6,569958848	9,199588477	7,032921811		7	-11,750000	-5,250000	-6,125000	-2,000000	-3,125000	-1,000000	
	8	5,890413047	8,956409084	7,904130468	6,971650663	8,575064777	6,800335315		8	-11,777778	-5,222222	-6,000000	-1,888889	-3,111111	-0,666667	
	9	5,799725652	8,632098765	7,876817558	7,094650206	8,264197531	6,870233196		9	-11,800000	-5,300000	-5,800000	-2,300000	-3,100000	-0,900000	
	10	6,011348048	8,229829156	7,877665544	7,008729268	8,022321985	6,896246415		10	-11,818182	-5,272727	-6,000000	-2,272727	-2,636364	-0,909091	
	11	6,189071788	7,804298125	7,925468679	6,889117513	7,810928212	7,012117055		11	-11,833333	-5,250000	-6,000000	-2,416667	-2,416667	-1,000000	
	12	6,303788119	7,682283423	7,898385565	6,817663818	7,886356442	6,946396539		12	-11,846154	-5,153846	-6,230769	-2,076923	-2,153846	-0,846154	
	13	6,387419165	7,413874192	7,806976288	6,825788752	7,950813247	6,682735646		13	-11,857143	-5,214286	-6,142857	-2,357143	-1,928571	-0,928571	
	14	6,458984911	7,192226795	7,732876086	6,665843621	7,872062186	6,678646548		14	-11,866667	-5,333333	-6,066667	-2,333333	-1,933333	-0,933333	
	15	6,401748971	7,083333333	7,615397805	6,72085048	8,04063786	6,604938272		15	-11,875000	-5,375000	-5,937500	-2,437500	-1,937500	-1,125000	

Figura 14 – Simulação MC do modelo **HP** para cinco temperaturas distintas. As tabelas ilustram a forma de saída dos dados: a tabela da esquerda mostra a evolução do **quadrado** do raio de giração (rg_2), e a da direita mostra a evolução da energia do sistema (U). São mostrados 16 instantâneos da parte inicial da simulação.

Figura 15 – Resultados da simulação de uma cadeia **HP** de 27 unidades. São mostrados (à esquerda) a evolução ao longo da simulação da raiz quadrada da média do raio de giração ao quadrado $\langle rg_2 \rangle$, e à direita a evolução da média da energia potencial $\langle U \rangle$, para cinco diferentes temperaturas. As curvas foram construídas usando 10.000 instantâneos igualmente espaçados, cerca de 2% do total de passos MC usados na simulação: tempo MC = 10.000 corresponde a 500.000 passos MC

A Figura 16 mostra uma configuração da etapa final da simulação com $T = 0,3$ para o modelo **HP** utilizado aqui. Note a formação de um "rabicho" numa das extremidades da cadeia, correspondente aos monômeros da posição 14 à 27, que são polares, e o restante da cadeia numa configuração bem mais compactada, como se espera, por se tratar de monômeros apolares.

Figura 16 – Configuração final da simulação do modelo **HP** à temperatura $T = 0,3$.

6.4 Comentários sobre detalhes técnicos

- **Código fonte da plataforma.** O código fonte de toda a plataforma citada neste trabalho de mestrado está disponível no *Version Control* (KEYES, 2004) através da URL <<https://gridproteinfolding.codeplex.com/>>.
- **C# (C Sharp)** - "é uma linguagem de programação interpretada multi-paradigma fortemente tipada, e, possuindo paradigmas de programação imperativa, funcional, declarativa, orientada a objetos e generica. C# foi desenvolvida pela Microsoft como parte da plataforma .NET. A sua sintaxe orientada a objetos foi baseada no C++ mas inclui muitas influências de outras linguagens de programação, como Object Pascal e Java. O código fonte é compilado para *Common Intermediate Language* (CIL) que é interpretado pela máquina virtual *Common Language Runtime* (CLR)." (BALAGURUSAMY, 2010a)
- **C# é multiplataforma?** C# é uma linguagem que difere da plataforma. Por padrão, a mesma é executável baseada em Sistemas Operacionais Windows, entretanto, existe a possibilidade de compilação para multiplataforma através do MONO (SOLIN, 2013).
- **MONO** - é uma implementação open source do .NET Framework da Microsoft com base nos padrões ECMA para C# e Common Language Runtime (BALAGURUSAMY, 2010b). Uma solução para desenvolvimento de aplicações em multi plataforma.

Tabela 1 – Comparação do percentual de utilizadores por sistema operacional

Nome	Fundação/Empresa	Versão mais recente	Ano versão mais recente	Percentagem	Utilizadores
Windows	Microsoft	Windows 10	2015	88.90%	400 milhões
macOS	Apple Inc.	macOS v10.12 "Sierra"	2016	5.54%	22,5 milhões
Sistemas baseados em Linux	Linus Benedict Torvalds	Kernel Linux 3.18.1	2015	2,13%	8,5 milhões

Fonte: <https://pt.wikipedia.org/wiki/Sistema_operativo> A tabela acima demonstrar o comparativo de utilização dos sistema operacionais utilizados hoje em dia, por este motivo, a primeira versão desta plataforma está disponível para o sistema operacional Windows, posteriormente, ocorrerá distribuição de uma versão para os demais sistemas operacionais.

- **Qual é o formato dos dados processados na plataforma?** Existe dois momentos: Primeiramente durante o processamento em tempo Monte Carlo, os dados são armazenados no formato de Fat File, ou seja, arquivos texto no padrão ANSI (INSTITUTE, 2016) legível no formato UTF-8 (W3SCHOOLS.COM, 2016). Pós processamento, para a devolução dos resultados de forma "amigável", os dados provindos dos arquivos Fat File são extraídos, manipulados e convertidos para as devidas saídas, tais como formato Excel, ou o formato Origin.
- **Existe outra plataforma que efetua de forma similar o processamento computacional distribuído para simulação de *Folding* de proteínas?** O Folding@home <<http://folding.stanford.edu>> é um projeto de computação distribuída que realiza simulações do *folding* de proteínas. O projeto é mantido pelo Departamento de Química da Universidade de Stanford.
- **A aplicação em questão, é acessível através de multiplataforma?** O *frontend* (ou interface de acesso do usuário) é uma Web App (ou aplicativo acessível através de Internet) o qual utiliza os padrões descritos pela órgão regulamentador W3C (DAHL, 2017) e é legível através de qualquer navegador de mercado, como por exemplo: Chrome, Internet Explorer, Safari entre outros.
- **Backend** (ou aplicativo que efetua a simulação computacional) é o aplicativo responsável pelo processamento das informações (no qual o usuário pode optar por instalar ou não, ou seja, contribuir com a sua CPU para o problema em questão) em sua atual versão, está disponível nesta primeira versão para o sistema operacional Windows.

COMENTÁRIOS FINAIS

Esta dissertação descreve uma aplicação criada com intuito de efetuar simulações relacionadas com o problema do *folding* de heteropolímeros mediante o método MC.

7.1 Sobre a plataforma simulacional implementada

A presente aplicação, concebida na forma de uma plataforma simulacional, foi inspirada em um programa computacional escrito em linguagem FORTRAN (referido aqui como "legado"), há mais de uma década. O referido legado não faz uso dos novos recursos que a tecnologia da informação oferece, e que este trabalho procura implementar. Logo, para várias de suas deficiências, foram aplicadas soluções específicas: *i*- Documentação: deficiências foram sanadas pela codificação de um programa plenamente comentado; *ii*- Modularidade: o programa original, altamente acoplado, impedia medir os impactos de qualquer ajuste ou alteração na aplicação legada de modo eficiente; o novo programa foi reescrito, ainda que parcialmente, de forma completamente modular; *iii*- Computação distribuída: o funcionamento do programa original utiliza um único núcleo computacional; aqui foi introduzido o conceito de computação distribuída, que não pode ser feito em FORTRAN, migrando assim para a linguagem C#, com a aplicação do conceito de *Software* colaborativo, o *groupware*, isto é, um programa de computador que apoia o trabalho desenvolvido coletivamente.

Para a implementação do novo sistema, durante as sessões de análises iniciais, com levantamento de requisitos, *workshops* e entrevistas dos usuários atuais, também foi analisado o código fonte da aplicação legada. Assim, foi tomada a decisão de não se tentar efetuar a engenharia reversa da codificação já existente, para então se obter prontamente um sistema mais geral em que o usuário final poderá desenhar o seu próprio modelo de cadeia heteropolimérica, com ênfase no paradigma de um programa criado para ser executado em *Stand-alone* (independente) em uma plataforma computacional colaborativa.

A presente versão do sistema, antes de tudo, trata-se de um exemplo das possibilidades que o atual estágio da TI pode oferecer ao cenário acadêmico, como também mostra a disponibilidade na comunidade de profissionais com conhecimento para contribuir na importante transição da tradicional tarefa de *processar dados* localizados, para o grande universo do *Big data*, no qual se lida com grande velocidade de processamento, grandes volumes e variedades de dados, autenticidade e valores. Para isso, os domínios de diversas esferas do saber precisam se juntar num processo multidisciplinar.

A nova versão do Software construído durante este trabalho de mestrado está disponível e funcionando em versão Alpha. O software se encontra instalado no laboratório de Física Biológica da FCFRP. Após a finalização do Mestrado, disponibilizaremos esta plataforma de colaboração através da Internet pelo endereço: <http://efolding.fcfrp.usp.br>. Todos os exemplos, resultados e análises mostradas aqui são provenientes da versão atual do Software desenvolvido.

7.2 Sobre os resultados das simulações obtidos

Os resultados aqui mostrados representam aplicações em simples de cadeias poliméricas em rede regular (rede cúbica no caso), cujos elementos podem interagir entre si de acordo com o modelo que o usuário definir. Tais interações são memorizadas em uma matriz energética $N \times N$, contendo todas as $\binom{N}{2}$ combinações possíveis de pares de contatos de uma cadeia linear de tamanho N .

Três modelos foram escolhidos, cadeia **polar**, cadeia **apolar** e o modelo **HP**, com dois propósitos principais. Primeiramente, para validar o programa. Isso foi feito com os dois primeiros casos (cadeia polar e apolar), cujos resultados das simulações comparamos com resultados conhecidos através de solução analítica (exata) ou resultado lógico esperado. O segundo propósito, foi mostrar a facilidade com que (em potencial ao menos) se poderá criar qualquer modelo (em rede) através do *design* da cadeia heteropolimérica. Isto foi feito com o modelo *HP*, cujo resultado da simulação, embora não comparados diretamente com outros resultados (de simulação ou não), fomos capazes de verificar visivelmente por meio de uma estratégia peculiar: desenhamos uma cadeia com cerca da metade sendo hidrofóbica e o restante hidrofílica. Desta forma fomos capazes de prever e verificar o resultado final de equilíbrio: uma conformação com metade da cadeia compactada e parte restante distendida. Desta forma, o aplicativo pode ser validado e ao mesmo tempo, suas versatilidades foram ilustradas sob diferentes perspectivas.

7.2.1 Comparativo de resultados com a versão legada

Sobre a comparação entre as linguagens de programação utilizadas na versão do software apresentado neste trabalho, em relação a versão legada, escrito em Fortran, elas possuem escopo e estruturas diferenciadas. Cada qual com suas características e vantagens. O ponto

importante não é equipararmos tais características, mas sim, a mudança de contexto da estrutura de processamento unitário para uma estrutura distribuída. Portanto, o ponto a ser observado é a mudança de contexto, onde o software escrito em Fortran (legado) apresenta as seguintes características em relação a solução apresentada:

- A plataforma proposta não usa o potencial estereoquímico (a matriz de interação) utilizada no código legado. Para efeito de comparação maior a implementação deste algoritmo deveria ser feita com o referido potencial. Isto ficará para uma próxima versão da plataforma. Para efetuar uma comparação justa será necessário realizar milhares simulações Monte Carlo com condições iniciais distintas;
- Arquivos de resultados simulacionais armazenam dados brutos, sem formatação, integração ou mesmo sem conversão em um formato comum de uso e leitura, por exemplo, como planilhas do Excel;
- As simulações são efetuadas de forma unitária, ou seja, uma a uma, e quando são executadas na mesma máquina acabam concorrendo o tempo de processamento da máquina de forma compartilhada;
- Cada simulação deve ser submetida de forma manual nas máquinas disponíveis;
- Não é possível efetuar pause/resume de um processamento;
- Entre outras características, o equiparativo fica restrito a mudança do contexto descrito acima;

7.3 Perspectivas futuras

Por se tratar de uma versão Alpha, admite-se *a priori* a existência desde pequenos *bugs*, até dificuldades maiores, como também alguma falha. Assim, durante algum tempo, o aplicativo deverá ser submetido a diversos testes de robustez, consistência e generalidade, os quais finalmente deverão possibilitar o estabelecimento claro de seus atributos e limites. Mas, alguns futuros desenvolvimentos mais significativos são desejados e esperados:

Um desenvolvimento importante, que poderá ser concluído num futuro breve, é a introdução da possibilidade de simulação de heteropolímeros em solução. Uma vez concluída, essa implementação possibilitará a escolha entre duas classes de modelos: sistemas com solvente implícito e com solvente explícito. Então, os resultados de modelos com e sem solvente explícitos poderão ser comparados e em que condições suas equivalências poderiam ser admitidas.

O presente sistema considera somente simulações em rede cúbica. Porém, a possibilidade de fazer simulações em outras redes regulares é desejável em muitas situações. Dentre elas, a

rede de diamante, por exemplo, é um objetivo que deve despertar o interesse da comunidade, devido ao fato de reproduzirem as configurações de cadeias alifáticas lineares.

Enfim, dada a natureza do atual aplicativo, que possibilita contribuições vindas também da comunidade, espera-se que estas e outras generalizações e implementações, poderão compor, finalmente, um plataforma simulacional para heteropolímeros com características gerais e de utilidade para a comunidade que trabalha na área pertinente.

7.3.1 Política de escalonamento de trabalhos e de distribuição de cargas

A política de escalonamento utilizado neste trabalho atua inicialmente como um agendador de tarefas (em inglês *scheduling*), esta atividade possibilita executar simulações em concorrência. O algoritmo utilizado foi FIFO (First in, first out) em português "primeiro que entra, primeiro que sai", onde a primeira requisição de processamento que chega será o primeiro a ser executado, não-preemptivo, ou seja, executa o processo como um todo do início ao fim não interrompendo o processo executado antes dele ser finalizado. Contextualizando, a plataforma aguarda a disponibilidade de tempo de máquina para o processamento de uma simulação; quando algum nó está disponível, o escalonamento ocorre então encaminhando as simulações utilizando a ordem de criação desta fila. Quanto maior for o número de nós (máquinas) disponíveis, maior o fluxo do FIFO e por consequência ocorrerá maior concorrência de resultados simulacionais.

A distribuição de carga do processamento por sua vez, é efetuada seguindo o contexto apresentado acima, onde, todos os nós de máquinas disponíveis receberam somente uma tarefa a ser executada (uma simulação), uma segunda tarefa será encaminhada somente quando um dos nós estiverem livres para o recebimento.

Nesta versão da plataforma, não foi realizado o estudo que relacionaria qual o melhor nó de máquina disponível por tipo de simulação enviado. Tendo em vista o tempo despendido nesta dissertação deixaremos este estudo para futuras versões da plataforma.

REFERÊNCIAS

A.G., B. J. P. **Random Data: Analysis and Measurement Procedures**. [S.l.]: Wiley, 1971. Citado na página 11.

BALAGURUSAMY, E. **Programming In C#**. [S.l.]: McGraw-Hill, 2010. 1 p. <<https://books.google.com/books?id=JUNqMPIZ4yMC&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false>>. Citado na página 54.

_____. _____. [S.l.]: McGraw-Hill, 2010. 13 p. <<https://books.google.com/books?id=JUNqMPIZ4yMC&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false>>. Citado na página 54.

BLIKSTEIN. P. **Monte Carlo ou Sentido da Vida**. [S.l.]: Jornal da USP, 1999. pg. 20 p. Citado na página 10.

DAHL, D. A. **Multimodal Interaction with W3C Standards: Toward Natural User Interfaces to Everything**. [S.l.]: Springer, 2017. xxi p. <<https://books.google.com/books?id=VNaGDQAAQBAJ&pg=PA34&dq=W3C&hl=pt-BR&sa=X&ved=0ahUKEwinjY6h0P7RAhWM7SYKHSXEDNkQ6AEIJTAB#v=onepage&q=W3C&f=false>>. Citado na página 55.

DICIONÁRIO escolar da língua portuguesa/Academia Brasileira de Letras. [S.l.]: Companhia Editora Nacional, 2008. p. 1 304 p. Citado na página 42.

DILL KEN A.; BROMBERG, S. **Molecular Driving Forces: Statistical Thermodynamics in Biology, Chemistry, Physics, and Nanoscience**. 2003. 643-657 p. Citado 2 vezes nas páginas 47 e 50.

DONY CHRISTOPHE; KNUDSEN, J. L. R. A. T. A. **Advanced Topics in Exception Handling Techniques**. [S.l.]: Springer, 2006. Citado na página 103.

ECMA. **Standard ECMA (European Computer Manufactures Association) Fortan**. 1965. <<http://www.ecma-international.org/publications/files/ECMA-ST-WITHDRAWN/ECMA-9,%201st%20Edition%20April%201965.pdf>>. Citado na página 2.

ECROYD H; CARVER, J. **Unraveling the mysteries of protein folding and misfolding**. 2008. <<https://www.ncbi.nlm.nih.gov/pubmed/18767168>>. Citado na página 6.

FILHO GILBERTO FARIAS DE SOUZA; ALEXANDRE, E. d. S. M. **Introdução a Computação**. [S.l.]: Editora da UFPB / João Pessoa, 2014. 56 p. <<http://pt.wikipedia.org/wiki/Entrada/saida>>. Citado na página 2.

FRENKEL, D. **Introduction to Monte Carlo Methods**. [S.l.]: John von Neumann Institute for Computing, 2004. 29-60 p. <<http://www.phys.ubbcluj.ro/~zneda/edu/mc/mc1.pdf>>. Citado na página 9.

GRIFFITHS, D. J. **Introduction to Quantum Mechanics - Second Edition**. [S.l.]: Cambridge University Press, 2007. 2-12 p. Citado na página 9.

- GUNDERLOY, M. **Coder to Developer: Tools and Strategies for Delivering Your Software**. [S.l.]: SYBEX, 2004. 189,190,191,192,193,194 p. <https://books.google.co.cr/books?id=-2wqcKAq4tIC&pg=PA191&dq=log4net&hl=pt-BR&sa=X&redir_esc=y#v=onepage&q=log4net&f=false>. Citado na página 104.
- INSTITUTE, A. N. S. **ANSI - American National Standards Institute**. 2016. <<https://www.ansi.org/>>. Citado na página 55.
- JR., H. O. **Pirataria de Software (Licença de software)**. [S.l.]: MM Livros Editora e Distribuidora, 2004. 81,82 p. <https://books.google.co.cr/books?id=X9G_05fbVd0C&pg=PA81&dq=licen%C3%A7a+de+software&hl=pt-BR&sa=X&redir_esc=y#v=onepage&q=licen%C3%A7a%20de%20software&f=false>. Citado na página 91.
- KEYES, J. **Software Configuration Management**. [S.l.]: CRC Press, 2004. 1,2,3 p. <<https://books.google.co.cr/books?id=0vjMIBz4nC0C&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false>>. Citado na página 54.
- KITTEL, C. **"Chapter 1". Introduction to Solid State Physics (Seventh ed.)**. 1996. Citado 2 vezes nas páginas 27 e 37.
- KROESE D. P.; BRERETON, T. T. T. B. Z. I. **Why the Monte Carlo method is so important today**. 2014. <<http://onlinelibrary.wiley.com/doi/10.1002/wics.1314/abstract;jsessionid=8C09B74976DCBB981817EC1D15BBAD2.f03t01?systemMessage=WOL+Usage+report+download+page+will+be+unavailable+on+Friday+27th+January+2017+at+23%3A00+GMT%2F+18%3A00+EST%2F+07%3A00+SGT+%28Saturday+28th+Jan+for+SGT%29++for+up+to+2+hours+due+to+essential+server+maintenance.+Apologies+for+the+inconvenience.>> Citado na página 9.
- LAITINEN, K. **A Natural Introduction to Computer Programming with C#**. [S.l.]: Trafford Publishing (UK), 2004. 105 p. <<https://books.google.com/books?id=wVIm5xwMp7YC&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false>>. Citado 3 vezes nas páginas 11, 12 e 14.
- LAU K. F.; DILL, K. A. **A lattice statistical mechanics model of the conformational and sequence spaces of proteins**. 1989. pp 3986–3997 p. Citado na página 27.
- LEIGHTON, B. B. T. **Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete**. [S.l.]: Journal of Computational Biology 5, 1998. 27–40 p. Citado na página 28.
- LUNARDI G. L.; BECKER, J. L. M. A. C. G. **Relacionamento entre Investimentos em Tecnologia de Informação (TI) e Desempenho Organizacional: um Estudo Cross-counrtry envolvendo os Bancos Brasileiros**. 2003. Citado na página 6.
- MAGOULAS ROGER; LORICA, B. **Introduction to Big Data**. 2009. <<http://www.oreilly.com/data/free/release-2-issue-11.csp>>. Citado na página 1.
- METROPOLIS ARIANNA W. ROSENBLUTH, M. N. R. A. H. T. N.; TELLER, E. **Equation of State Calculations by Fast Computing Machines**. 2016. Disponível em: <<http://scitation.aip.org/content/aip/journal/jcp/21/6/10.1063/1.1699114>>. Citado na página 38.
- MISTRÍK IVAN; GRUDY, J. H. A. v. d. W. J. **Collaborative Software Engineering**. [S.l.]: Springer, 2010. 394 p. <<https://books.google.com/books?id=NOprLdi25qUC&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false>>. Citado na página 44.

MOHR PETER J.; TAYLOR, B. N. N. D. B. **Recommended Values of the Fundamental Physical Constants**. 2008. 633–730 p. Citado na página 36.

NEMIROVSKY MARIO; TULLSEN, D. **Multithreading Architecture**. [S.l.]: Mark D. Hill, 2013. 1,2,3 p. <<https://books.google.com/books?id=mZ5eAQAAQBAJ&printsec=frontcover&hl=pt-BR#v=onepage&q&f=false>>. Citado na página 45.

NICKERSON, R. C. **Business and information systems**. [S.l.]: Prentice Hall, 2001. Citado na página 6.

PATTERSON DAVID A.; HENNESSY, J. L. **Computer Organization and Design**. [S.l.]: Morgan Kaufmann Publishers, 1989. p. 715 p. Citado na página 1.

PROVIDELO, F. D. **Modelo Computacional da Dinâmica de Transmissão da Tuberculose**. 2000. 9-23 p. Citado na página 11.

ROCHA L.F.O.; SILVA, I. C. A. **Distinct conformational properties determined by implicit and explicit representation of protein–solvent interactions. An analytical and computer simulation study**. [S.l.]: Elsevier - Physica A, 2009. 4097–4104 p. Citado na página 28.

SELKOE, D. J. **Folding proteins in fatal ways**. [S.l.]: Nature, 2003. 900-904 p. <<https://www.ncbi.nlm.nih.gov/pubmed/14685251>>. Citado na página 3.

_____. **Cell biology of protein misfolding: The examples of Alzheimer’s and Parkinson’s diseases**. [S.l.]: Nature, 2014. 1054-1061 p. <<http://www.nature.com/ncb/journal/v6/n11/full/ncb1104-1054.html>>. Citado na página 5.

SOBOL, I. M. **A Primer for the Monte Carlo Method**. [S.l.]: CRC Press, 1994. 126 p. <<https://archive.org/details/APrimerForTheMonteCarloMethod>>. Citado na página 9.

SOLIN, D. **Developing on Linux With C# and .Net: The Mono Project**. [S.l.]: Apress L. P, 2013. Citado na página 54.

T., B. B. L. **Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete**. [S.l.]: Journal of Computational Biology, 1998. 27–40 p. Citado na página 28.

TANENBAUM, A. S. **Distributed Systems: Principles and Paradigms**. 2016. 2 p. <https://pt.wikipedia.org/wiki/Sistema_de_processamento_distribuido>. Citado na página 1.

TANENBAUM ANDREW; STEEN, M. V. **Distributed Systems: Principles and Paradigms**. [S.l.]: CreateSpace Independent Publishing Platform, 2016. 2 p. Citado na página 3.

TURBAN, E. **Information Technology for Management - transforming organizations in the digital economy**. 2006. Citado na página 6.

TURBAN E.; LEIDNER, D. M. E. W. J. **Information technology for management: transforming organizations in the digital economy**. 2007. Citado na página 6.

TURBAN EFRAIM; RAINER, R. K. P. R. E. **Administração de tecnologia da informação. RIO DE JANEIRO: Campus**. 2003. 598 p. Citado na página 6.

W3SCHOOLS.COM. **UTF-8 @ONLINE**. 2016. <http://www.w3schools.com/charsets/ref_html_utf8.asp>. Citado na página 55.

WEISSTEIN, E. W. "**Buffon's Needle Problem**"@ONLINE. 2017. <<http://mathworld.wolfram.com/BufonsNeedleProblem.html>>. Citado na página 9.

WIKIPEDIA. **Log de dados** @ONLINE. 2016. <https://pt.wikipedia.org/wiki/Log_de_dados>. Citado na página 103.

GLOSSÁRIO

CC: Ciência da Computação é a ciência que estuda algoritmos computacionais, lógicas binárias, matemática, redes de computadores entre outros aspectos para aplicar a sua implementação, na forma de software, para execução em computadores eletrônicos.

Framework: É uma abstração que une códigos comuns entre vários projetos de *software* provendo uma funcionalidade genérica. *Frameworks* são projetados com a intenção de facilitar o desenvolvimento de *software*, habilitando designers e programadores a gastarem mais tempo determinando as exigências do *software* do que com detalhes de baixo nível do sistema.

GNU: General Public License (Licença Pública Geral), GNU GPL ou simplesmente GPL, é a designação da licença para software livre idealizada por Richard Matthew Stallman em 1989, no âmbito do projeto GNU da Free Software Foundation (FSF). <https://pt.wikipedia.org/wiki/GNU_General_Public_License>.

HTTP: O Hypertext Transfer Protocol (HTTP), em português Protocolo de Transferência de Hipertexto, é um protocolo de comunicação (na camada de aplicação segundo o Modelo OSI) utilizado para sistemas de informação de hipermídia, distribuídos e colaborativos.¹ Ele é a base para a comunicação de dados da World Wide Web.

Informática: É um termo usado para descrever o conjunto das ciências relacionadas ao armazenamento, transmissão e processamento de informações em meios digitais, estando incluídas neste grupo: a ciência da computação, a teoria da informação, o processo de cálculo, a análise numérica e os métodos teóricos da representação dos conhecimentos e da modelagem dos problemas. Mas também a informática pode ser entendida como ciência que estuda o conjunto de informações e conhecimentos por meios digitais.

IP: Cada computador ou device conectado na internet possui um IP (Internet Protocol ou Protocolo de internet) único, que é o meio em que as máquinas usam para se comunicarem na Internet.

OSI: OSI ou Open Systems Interconnection, serve de base para a implementação de qualquer tipo de rede, seja de curta, média ou longa distância.

Padrões de projeto: ou *Design Pattern*, descreve uma solução geral reutilizável para um problema recorrente no desenvolvimento de sistemas de *software* orientados a objetos. Não é um código final, é uma descrição ou modelo de como resolver o problema do qual trata, que pode ser usada em muitas situações diferentes.

SOAP: SOAP (Simple Object Access Protocol) é um protocolo para troca de informações estruturadas em uma plataforma descentralizada e distribuída.

SQLServer: O MS SQL Server é um SGBD - sistema gerenciador de Banco de dados relacional desenvolvido pela Microsoft.

Tanenbaum: Andrew Stuart "Andy" Tanenbaum (White Plains, 16 de março de 1944) é o chefe do Departamento de sistemas de computação, na Universidade Vrije, Amsterdã nos Países Baixos. Ele é bastante conhecido por seus livros sobre ciência da computação.

TCP: O acrônimo para o inglês Transmission Control Protocol é "Protocolo de controle de transmissão", o qual é um dos protocolos sob os quais assenta o núcleo da Internet.

Template: É um documento sem conteúdo, com apenas a apresentação visual (apenas cabeçalhos por exemplo) e instruções sobre onde e qual tipo de conteúdo deve entrar a cada parcela da apresentação.

TI: Tecnologia da Informação definido como o conjunto de todas as atividades e soluções providas por recursos de computação que visam a produção, o armazenamento, a transmissão, o acesso, a segurança e o uso das informações.

Web: Sinônimo mais conhecido de *World Wide Web* (WWW). É a interface gráfica da Internet que torna os serviços disponíveis totalmente transparentes para o usuário e ainda possibilita a manipulação multimídia da informação.

WSDL: A Web Services Description Language (WSDL) é uma linguagem baseada em XML utilizada para descrever Web Services funcionando como um contrato do serviço. Trata-se de um documento escrito em XML que além de descrever o serviço, especifica como acessá-lo e quais as operações ou métodos disponíveis.

XML: XML (eXtensible Markup Language) é uma recomendação da W3C para gerar linguagens de marcação para necessidades especiais.

INSTALAÇÃO

A.1 Introdução

Objetivo deste texto é prover suporte técnico para as ações de instalar, manter e configurar a aplicação eFolding, desde o Servidor até ao aplicativo de acesso distribuído.

Os textos e pontos abordados são de referência técnica. Temos em mente as seguintes premissas, estamos falando de uma aplicação distribuída, portanto a sua configuração é efetuada em pontos distintos.

Destacamos três principais itens a ser configurados:

- Configuração do Banco de Dados;
- Configuração do servidor WEB (páginas acessível via Internet que permite interação com a ferramenta);
- Serviço que provê distribuição e transmissão de simulações;

Antes de pontuarmos os itens destacados acima, apresentamos alguns pontos importantes sobre a ferramenta, código fonte entre outros detalhes.

Observação importante: Não estamos abordando neste texto, a instalação do software cliente, o qual segundo este trabalho de mestrado é distribuído em máquinas chamadas de colaboradores. A distribuição é efetuada através de download no web site da Aplicação. As ações de instalação do aplicativo cliente, bem como "utilizá-lo", serão descritas no ANEXO II de Referência técnica usuário do aplicativo.

A.2 Licença de utilização do software

Para a utilização deste documento é necessário seguir as regras da licença GNU disponíveis em [GNU - General Public License](#).

Em termos gerais, a GPL baseia-se em 4 liberdades:

1. A liberdade de executar o programa, para qualquer propósito (liberdade nº 0)
2. A liberdade de estudar como o programa funciona e adaptá-lo às suas necessidades (liberdade nº 1). O acesso ao código-fonte é um pré-requisito para esta liberdade.
3. A liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo (liberdade nº 2).
4. A liberdade de aperfeiçoar o programa e liberar os seus aperfeiçoamentos, de modo que toda a comunidade se beneficie deles (liberdade nº 3). O acesso ao código-fonte é um pré-requisito para esta liberdade.

Com a garantia destas liberdades, a GPL permite que os programas sejam distribuídos e reaproveitados, mantendo, porém, os direitos do autor por forma a não permitir que essa informação seja usada de uma maneira que limite as liberdades originais. A licença não permite, por exemplo, que o código seja apoderado por outra pessoa, ou que sejam impostos sobre ele restrições que impeçam que seja distribuído da mesma maneira que foi adquirido. A GPL está redigida em inglês e atualmente nenhuma tradução é aceita como válida pela Free Software Foundation, com o argumento *GNU General Public License* de que há o risco de introdução de erros de tradução que poderiam deturpar o sentido da licença. Deste modo, qualquer tradução da GPL é não-oficial e meramente informativa, mantendo-se a obrigatoriedade de distribuir o texto oficial em inglês com os programas.

A.3 Introdução ao Sistema eFolding

O eFolding é um conjunto de ferramentas computacionais dedicadas a racionalizar processos que envolvem simulações Monte Carlo e análises de seus resultados através de técnicas de simulação computacional distribuída e o método Monte Carlo - com ênfase à aplicação em heteropolímeros. O propósito motivador deste software é o estudo do problema de *folding* de heteropolímeros, o que inclui proteínas. Contudo, a ferramenta em questão, em sua primeira versão, poderá ser aplicada a praticamente todos os tipos de polímeros lineares em rede, pois o usuário poderá definir e implementar também o modelo de cadeia que desejar.

A.3.1 Características e principais funcionalidades

O código fonte (que é livre e aberto) está disponível para ser baixado livremente no endereço <https://gridproteinfolding.codeplex.com/SourceControl/latest> e sua implementação

foi escrita em C# (C sharp) tendo como camada de armazenamento o sistema gerenciador de banco de dados MSSQLServer *MicrosoftSqlServer2014*.

O aplicativo possui variadas funcionalidades entre as quais destacam-se:

- Simulação computacional distribuída;
- Armazenamento de informações pós simulacionais em Banco de Dados;
- Extração de dados para formato Office;

A.4 Arquitetura

Este tópico é dedicado a arquitetura utilizada para a aplicação em questão.

A.4.1 Computação distribuída

Um sistema distribuído segundo a definição de Andrew Tanenbaum (*Tanenbaum*) é uma "coleção de computadores independentes que se apresenta ao usuário como um sistema único e consistente". Assim, a computação distribuída consiste em adicionar o poder computacional de diversos computadores interligados por uma rede de computadores. A união desses diversos computadores com o objetivo de compartilhar a execução de tarefas, é conhecida como sistema distribuído.

A intenção para este projeto é o uso do processamento distribuído o qual visa interligar vários nós de processamento (computadores individuais, não necessariamente homogêneos) de maneira que um processo de grande consumo seja executado no nó "mais disponível", ou mesmo subdividido por vários nós.

A ideia é ter ganho real nas simulações do *folding* para o qual algumas máquinas desse sistema distribuído terão um papel único, e importante para o ganho do processamento.

O intuito é dividir as simulações para que possam ser realizadas por computadores pessoais comuns. Este conceito utiliza a participação voluntária dos internautas, que "emprestam" o tempo de processamento ocioso de seus computadores para processamento de simulações que emulam o processo de *folding*. Portanto, quando conectado a internet, com o software pré-instalado em sua máquina, durante o tempo livre do processador o software utilizará a máquina disponível para análise do *folding*.

Entre alguns aspectos vantajosos podemos citar:

- Centralização da distribuição das simulações em um único ponto, que chamaremos de Servidor, sua função é a realização distribuída das simulações entre os clientes disponíveis.
- Distribuição de simulações entre vários computadores não homogêneos disponíveis na rede mundial de computadores (Internet), os quais chamamos de máquinas cliente;

- Na máquina cliente, ocorrerá a divisão de tarefas de processamento, quando divisível em várias sub-tarefas, elas podem ser realizadas em paralelo (utilização de threads);

A.5 Camada de rede

O modelo de camada de rede *OSI*, é responsável por controlar a operação da rede de um modo geral. Dessa forma não precisamos nos preocupar com a entrega de mensagens entre servidor e cliente, pois entre suas principais funções estão o roteamento dos pacotes entre fonte e destino (servidor e cliente), mesmo que estes tenham que passar por diversos nós intermediários durante o percurso. Podemos citar como principais responsáveis pela flexibilidade da camada:

- IP, Internet Protocol o qual é um protocolo de comunicação usado entre duas ou mais máquinas em rede para encaminhamento dos dados.
- TCP, Transmission Control Protocol é um dos protocolos sobre os quais assenta o núcleo da Internet. Entre outras várias camadas de protocolos, citamos uma outra camada importante para o nosso projeto, a camada de Aplicação; nessa camada que ocorre a interação micro-usuário. Utilizaremos dois protocolos o HTTP e SMTP.
- HTTP, HyperText Transfer Protocol Hypertext Transfer Protocol é um protocolo de comunicação utilizado para a obtenção de recursos interligados a World Wide Web, serviços de WebSite.
- SMTP, Simple Mail Transfer Protocol Simple Mail Transfer Protocol (SMTP) é o protocolo padrão para envio de e-mails através da Internet.

A.5.1 Modelo Cliente-Servidor

O modelo cliente-servidor (na Ciência da Computação) *CC* define uma estrutura de aplicação distribuída fisicamente em locais distintos, seu propósito é distribuir tarefas a ser processadas entre os fornecedores de um recurso ou serviço, designados como servidores, e os requerentes dos serviços, designados como clientes. O servidor é um host que está executando um ou mais serviços ou programas que compartilham recursos com os clientes. O cliente não compartilhará qualquer um de seus recursos, mas solicita um conteúdo ou função do servidor. Os clientes iniciam sessões de comunicação com os servidores que aguardam requisições de entrada. Do ponto de vista de um diagrama de rede tanto os clientes, quanto o servidor são os nós (pontos de comunicação) na rede, esta organização dos nós de uma rede designa-se por topologia de rede.

A.5.2 Comunicação Cliente Servidor

Abaixo ilustramos a arquitetura de comunicação do ambiente distribuído do projeto eFolding:

Figura 17 – Configuração do modelo cliente-servidor

Figura 18 – Modelo de comunicação entre cliente-servidor

- DataBase: Servidor banco de dados SqlServer *SQLServer* do projeto eFolding
- Server eFolding: Fisicamente é o servidor presente nas instalações da Universidade. O mesmo abriga o software criado por este projeto denominado de “Host Distributed”. A função dessa aplicação é distribuir para as máquinas clientes as quais possuem o software pertinente ao processamento de artefatos do eFolding.
- Firewal: Firewall existente na intranet da Universidade. Este projeto utiliza portas “ports”

específicas para comunicação entre Servidor e os Clientes; tais portas necessitam estar devidamente configuradas para protocolar TCP/IP + SOAP em in e outgoing.

- **Contribuidor (Cliente):** Possuem esta denominação, pois contribuem com processamento de re-quisições providas do Server eFolding. O único recurso compartilhado é o poder de processamento dessas máquinas. O Server do eFolding encaminha a solicitação de processamento com informações necessárias para o mesmo. O Contribuidor, cliente, devolve as informações para o Server eFolding no final do processamento, o qual persiste essas informações no DataBase local e efetua outros processamentos com intuito de melhorar a apresentação dos dados para os pesquisadores (usuários).

A.6 Protocolos e tecnologias utilizados na comunicação eFolding

A.6.1 WCF (*Windows Communication Foundation*)

WCF é conjunto de tecnologias lançada com o Framework .NET da Microsoft para comunicação entre processos em múltiplas máquinas em uma rede. A API utiliza o modelo de programação orientado a serviços construído sobre o Framework .NET, e a base de sua arquitetura é baseada na arquitetura dos Web services. O objetivo principal do WCF é facilitar e agilizar a programação de Web services para comunicação entre processos.

A.6.2 Arquitetura baseada em Web services

WCF utiliza a arquitetura dos Web services como base para sua arquitetura. O uso de *XML* para definição dos dados, *SOAP* para troca de mensagens e *WSDL* para definição dos serviços é fruto desta associação.

A.6.3 Funcionamento - Comunicação

Para comunicação, o WCF utiliza o protocolo SOAP, o que torna suas aplicações compatíveis com as diversas outras que usam este padrão. Quando uma aplicação desenvolvida com WCF se comunica com uma aplicação não WCF, é utilizado o protocolo SOAP tradicional, codificado com o uso de XML. Porém, quando duas aplicações desenvolvidas com WCF se comunicam, elas codificam as mensagens SOAP utilizando um formato binário otimizado (entre outras opções), com intenção de reduzir a quantidade de informação desnecessária a ser transmitida pela rede.

A.6.4 Componentes de um serviço WCF

Um serviço WCF é composto de 3 partes: a classe que implementa o serviço, um ambiente (em inglês chamado de host environment) que abriga e mantém o serviço e um ou

mais endpoints (chamados Service Endpoint), componentes nos quais os clientes irão se conectar para realizar a comunicação. Os endpoints definem quais métodos estarão acessíveis e como os clientes devem se comunicar com os serviços (através de componentes chamados bindings). Um binding especifica quais protocolos serão usados para se conectar ao endpoint (WCF suporta alguns como *SOAP/HTTP* e *SOAP/TCP*), qual a codificação será utilizada, questões de segurança, entre outros. Eles também possuem um endereço (uma URL) e propriedades para especificar como os dados serão transmitidos.

A.7 Instalação da Aplicação

A.7.1 Requisitos gerais para instalação

Abaixo seguem os requisitos de plataforma, softwares, servidores para instalação do sistema.

1. Um servidor com a sua distribuição Windows 10 (recomendado). Aqui vamos usar a versão do Windows 7 onde será instalado o Serviço de Distribuição, Banco de Dados e a Hospedagem do Site.
2. As máquinas contribuidoras (computadores) são clientes que recebem uma requisição e devolve após processamento em rede, este problema se chama eFolding Cliente e o download e distribuição do aplicativo é através Web Site Hospedado no Servidor (item acima). As máquinas de contribuição nesta versão devem ser Windows.

A.7.2 Hardware mínimo necessário

- Servidor:
 - Computador com processador Intel ou Amd.
 - Memória RAM: 4GB RAM (8GB RAM recomendável);
 - HD: 250GB para instalar o Windows, e 1TB para utilização das aplicações;
- Cliente:
 - Computador com processador Intel ou Amd.
 - Memória RAM: 2GB RAM (8GB RAM recomendável);
 - HD: 250GB para instalação e uso da Aplicação;

A.7.3 Pré-requisitos

- Cliente & Servidor
 - Framework - .Net Framework 4.5: <<https://www.microsoft.com/pt-br/download/details.aspx?id=40779>>

- Servidor

- Microsoft SQL Server 2014: <<https://www.microsoft.com/pt-br/download/details.aspx?id=42299>>

- * Requisito mínimo: Versão Express Edition

- IIS (Internet Information Services): Pacote de distribuição do Windows contém o instalador.

- * Caminho: Control Panel, Programs, Turn Windows features on or off

Figura 19 – Imagem ilustrativa de instalação do pré-requisito IIS

A.7.4 Passo-a-passo da Instalação no Microsoft SQL Server 2014

Esta descrito abaixo o roteiro de instalação do SQL Server.

1. Insira a mídia contendo a instalação do SQL Server ou descompacte a imagem ISO em um diretório.
2. Abra o diretório raiz dos arquivos de instalação do SQL Server e execute o arquivo “setup.exe” SQL Server 2014 - Tela inicial.
3. Clique na opção “Installation” no menu lateral esquerdo e depois em “New SQL Server stand-alone installation or add features to an existing installation”. Essa opção serve para instalar o SQL Server em apenas uma máquina e é a que utilizo nesse tutorial.
4. Caso você deseje criar um cluster de servidores SQL Server (o banco de dados será executado em mais de 1 máquina simultaneamente, fazendo o balanceamento de carga e aumentando a disponibilidade do serviço, uma vez que se um dos servidores falha, os outros assumem as atividades do que falhou), selecione a segunda opção, “New SQL Server failover cluster installation”.
5. A terceira opção de instalação “Add node to a SQL Server failover cluster” serve para você adicionar uma máquina em um cluster já existente.

6. A última opção na tela de instalação, serve para quem já possui uma versão anterior do SQL server instalada e deseja atualizar a sua versão para a 2014.

7. Entre com a chave de ativação do SQL Server 2014 ou escolha a primeira opção (specify a free edition) para usar no modo de avaliação do produto (Trial).

8. Aceite o termo de uso e clique em “Next”.

9. Aqui você define se irá utilizar as atualizações automáticas do SQL Server ou não. Eu recomendo habilitar essa opção, pois as atualizações muitas vezes trazem correções para funcionalidades e para falhas de segurança, melhorias de desempenho e adição de novos recursos. Clique em “Next”.

10. Nesta tela, o SQL Server irá listar os pré-requisitos para a instalação. Para mim, ele gerou um Warning alertando que devo configurar as portas no Firewall para permitir conexões remotas (Saiba mais aqui). Se estiver tudo certo, clique em “Next”.

11. Aqui vamos definir o tipo de instalação do SQL Server.

- Na opção “SQL Server Feature Installation”, a instalação irá te permitir selecionar cada um dos componentes, que incluem o Banco de dados, Analysis Services (Cubos / BI), Reporting Services (Ferramenta de extração de dados e relatórios) e Integration Services (Ferramenta de programação usando T-SQL e outros recursos como manipulação de arquivos, geração de Excel, etc.). Ao selecionar essa opção, iremos configurar cada um dos serviços nas próximas telas e não vem com nada pré-selecionado.
- Na opção “SQL Server PowerPivot for SharePoint”, você irá instalar o recurso PowerPivot no servidor Sharepoint e opcionalmente pode optar por instalar o banco de dados no servidor.
- Na opção “All Features with Defaults”, a instalação irá selecionar todos os recursos (Banco de Dados, Analysis Services, Reporting Services e Integration Services) com suas configurações padrão na tela seguinte. É o modo mais fácil de instalar o SQL Server e o que irei utilizar no tutorial.

12. Selecione os componentes que deseja instalar, o caminho de instalação e clique em “Next”.

13. Nessa tela, a instalação irá verificar mais alguns pré-requisitos para avançar com a instalação.

14. Caso falhe na etapa da do .NET Framework 3.5 Service Pack 1, como no meu caso, onde eu havia acabado de instalar o Windows Server, Baixe o arquivo de instalação ou ative os recursos do .NET se estiver instalando no Windows Server (Gerenciamento do Servidor > Adição de Funções e Recursos logado como ADMINISTRADOR). Instale, reinicie o

servidor para a instalação surtir efeito e inicie novamente a instalação do SQL Server repetindo os passos anteriores.

15. Nesta tela podemos definir qual será o nome da instância para acessar o SQL Server. Configurando a instância como padrão "Default instance", não será necessário informar o nome da instância para se conectar ao servidor SQL Server. Caso contrário, para se conectar ao banco de dados, você terá que informar o nome do servidor e instância. Ex: servidor \ instanciaTeste. Uma instância, seja padrão ou nomeada, tem seu próprio conjunto de arquivos de programas e de dados, bem como um conjunto de arquivos comuns compartilhados entre todas as instâncias do SQL Server no computador. Por este motivo, mesmo que você instale o SQL Server como a instância padrão, você DEVE informar o ID da instância.

16. Nesta tela, configuramos a forma de inicialização dos serviços do SQL Server (Automático, Desabilitado ou Manual). Para quem for utilizar apenas o banco de dados, verifique principalmente os serviços SQL Server Agent, que é responsável pelos Jobs (rotinas automatizadas) que são executados no servidor e que geralmente vem configurado por

padrão como inicialização Manual e o SQL Server Database Engine, que é o motor do banco de dados. Esses dois serviços devem estar configurados como inicialização automática, caso você deseja que ao ligar o servidor/computador, o banco de dados já fique disponível automaticamente.

17. Nesta tela, definimos o Collation do servidor, que é o idioma de entrada e saída dos textos. É aqui que definimos se o banco terá suporte a acentuação, se será case-sensitive (letras minúsculas são diferente de letras maiúsculas. Essa é a configuração padrão) ou case-insensitive (Letras maiúsculas e minúsculas são interpretadas pelo banco como a mesma coisa).

Obs: O Collation das tabelas pode ser diferente do collation de uma coluna específica, que também pode ser diferente do Collation do database. O Collation do database serve para definir o collation padrão das tabelas e colunas quando não é especificado.

18. Ao clicar no botão “Customize” (opcional), podemos personalizar o collation do banco de dados de acordo com a necessidade. Particularmente, eu não gosto muito de utilizar o banco Case-Sensitive, apesar de ser uma boa prática, mas é uma questão de gosto pessoal.

19. Na tela seguinte, vamos configurar o motor do banco de dados. Aqui definimos o modo de autenticação do banco de dados e os usuários do AD que serão administradores. Para adicionar o usuário logado que está instalando o SQL Server, basta clicar no botão “Add Current User”. Os modos de autenticação do SQL Server são:

- **Windows authentication:** Utiliza o Active Directory para autenticar os usuários e permite apenas usuários que estão cadastrados no AD. Não é necessária a digitação de usuário e senha.
- **Mixed Mode:** Permite tanto a autenticação usando o Active Directory quanto autenticação do SQL Server, onde o login é criado no banco de dados e é acessado utilizando usuário e senha. É a opção mais comum, pois na maioria das vezes, as aplicações e sistemas não utilizam usuários do AD, e a configuração dentro da aplicação para utilizar usuário SQL Server é muito mais simples.

20. Na aba “Data Directories” podemos configurar os diretórios onde serão armazenados fisicamente no disco, os databases de sistema, o diretório padrão de Backup e os databases TempDB (dados e log) e User (dados e log).

21. Na aba “FILESTREAM”, é possível definir se esse recurso será habilitado no banco de dados. O FILESTREAM integra o SQL SERVER com o sistema de arquivos NTFS armazenando objetos do tipo (BLOB) com o tipo varbinary(max) utilizando-se da do sistema de arquivos NTFS para este armazenamento. É possível através de T-SQL realizar a inserção, exclusão, atualização e consulta e backup destes dados nativamente.

22. Na tela de configuração do Analysis Services, você pode definir o tipo de modelagem dos dados (Multidimensional – Opção Padrão) e o novo modo Tabular, presente a partir do SQL Server 2012 e especificar os usuários Administradores do Analysis Services.

23. Mais uma vez, o SQL Server nos permite definir os diretórios onde serão armazenados fisicamente no disco, os dados dos Cubos, os arquivos de log, o diretório temporário e o diretório de backup padrão.

24. Na tela de configuração do Reporting Services, podemos selecionar a primeira opção para Instalar e já configurar a ferramenta, de relatórios ou marcar a segunda opção (a qual eu fiz), onde será apenas instalado e configurado depois.

25. Aqui podemos definir os usuários Administradores do Distributed Replay Controller, a ferramenta de testes e avaliação e simulação de impacto do SQL Server, para complementar o SQL Server Profiler.

26. Nesta tela, podemos definir o nome do Controller criado, o caminho de instalação do Distributed Display e o caminho padrão onde serão armazenados os resultados das avaliações.

27. Nesta última tela, apenas revisamos tudo o que será instalado, o que já está instalado no servidor e as configurações que realizamos na instalação. Se estiver tudo OK, clique em “Install” para iniciar a instalação do Microsoft SQL Server 2014.
28. Aguarde, a instalação demora vários minutos. . .

SQL Server 2014 instalado!

A.7.5 Passo-a-passo da Instalação no Internet Informatino Services

Para instalar o IIS em seu computador siga os seguintes passos:

1. Acesse o painel de controle, Programas e recursos.

2. Clique em Ativar, Desativar recursos do Windows

3. Marque a opção Serviços de Informações da Internet.

4. Certifique-se de que as funcionalidades desejadas estão marcadas, como ASP, ASP.NET... etc...

5. Clique em Ok e aguarde a finalização da instalação.
6. Abra o browser e acesse <http://localhost>, ou <http://127.0.0.1>, deverá aparecer o logo do iis.

A.7.6 Como configurar o seu site da WEB do IIS

Este tópico descreve como configurar o seu primeiro site no IIS (Serviços de Informações da Internet) usando um número mínimo de etapas necessárias para você preparar o ambiente.

A.7.6.1 Configurar um site padrão

Quando o IIS é instalado, ele é pré-configurado para servir como um site padrão. No entanto, você pode querer alterar algumas das configurações. Para alterar as configurações

básicas do site e para emular as etapas necessárias para configurar o Apache pela primeira vez usando o arquivo de configuração:

1. Faça login no computador do servidor Web como administrador.
2. Clique em Iniciar, aponte para Configurações e clique em Painel de Controle.
3. Clique duas vezes em Ferramentas Administrativas e depois em Gerenciador de Serviços da Internet.
4. Clique com o botão direito do mouse no site que você deseja configurar no painel esquerdo e depois clique em Propriedades.
5. Clique na guia Site.
6. Digite uma descrição para o site na caixa Descrição.
7. Digite o endereço IP a ser usado para o site ou deixe a configuração padrão Todos (Não Atribuídos).
8. Modificar a porta de protocolo de controle de transmissão (TCP) conforme apropriado.
9. Clique na guia Diretório Base.
10. Para usar uma pasta no computador local, clique em um diretório neste computador e, em seguida, clique em Procurar para localizar a pasta que você deseja usar.
11. Para usar uma pasta que foi compartilhada a partir de outro computador na rede, clique em um compartilhamento localizado em outro computador e digite o caminho da rede ou clique em Procurar para selecionar a pasta compartilhada.
12. Clique em Leitura para conceder acesso de leitura à pasta (necessário).
13. Clique em OK para aceitar as propriedades do site.

A.7.6.2 Criar um novo site

Para criar um novo site, você deve configurar um host virtual e definir as configurações individuais desse host.

Para criar um novo site no IIS, execute estas etapas:

1. Faça login no computador do servidor Web como administrador.
2. Clique em Iniciar, aponte para Configurações e clique em Painel de Controle.
3. Clique duas vezes em Ferramentas Administrativas e depois em Gerenciador de Serviços da Internet.
4. Clique em Ação, aponte para Novo e clique em Site.
5. Após o início do Assistente para Criação de Site, clique em Avançar.
6. Digite uma descrição para o site. Essa descrição é usada internamente para identificar o site da Web só no Gerenciador de serviços de Internet.
7. Selecione o endereço IP a ser usado para o site. Se você selecionar Todos (Não Atribuídos), o site ficará acessível em todas as interfaces e endereços IP configurados.
8. Digite o número da porta TCP para publicar o site no.

9. Digite o nome de cabeçalho de Host (o nome real que é usado para acessar este site).
10. Clique em Próximo.
11. Digite o caminho para a pasta que contém os documentos do site ou clique em Procurar para selecionar a pasta e clique em Avançar.
12. Selecione as permissões de acesso para o site e clique em Avançar.
13. Clique em Concluir.

A.7.7 Habilitando conexões remotas no Servidor - Firewall

Dependendo da configuração do seu Windows, precisamos habilitar as conexões externas ao SQL Server, IIS e do aplicativo eFolding Hosting, uma vez que na maioria dos casos, os acessos são realizados de outras máquinas que não o servidor, como no nosso caso, onde o servidor fica na máquina conhecida e o acesso aos recursos da máquina são efetuados fisicamente. Caso a conectividade com os serviços não estejam ocorrendo, o Firewall do nosso servidor provavelmente está bloqueando as portas necessárias para que o eFolding Hosting e IIS sejam acessíveis por outras máquinas a partir da Internet. Abaixo segue o script de como liberar essas portas no Firewall.

Liberando as portas do Firewall para acesso NetBIOS:

Código-fonte 7: Script de Liberando as portas do Firewall para acesso NetBIOS

```
1 netsh advfirewall firewall add rule name="NetBIOS UDP Porta 137
  " dir=in action=allow protocol=UDP localport=137
2 netsh advfirewall firewall add rule name="NetBIOS UDP Porta 137
  " dir=out action=allow protocol=UDP localport=137
3
4 netsh advfirewall firewall add rule name="NetBIOS UDP Porta 138
  " dir=in action=allow protocol=UDP localport=138
5 netsh advfirewall firewall add rule name="NetBIOS UDP Porta 138
  " dir=out action=allow protocol=UDP localport=138
6
7 netsh advfirewall firewall add rule name="NetBIOS TCP Porta 139
  " dir=in action=allow protocol=TCP localport=139
8 netsh advfirewall firewall add rule name="NetBIOS TCP Porta 139
  " dir=out action=allow protocol=TCP localport=139
```

Liberando as portas do Firewall para acesso SQLServer:

Código-fonte 8: Script de Liberando as portas do Firewall para acesso SQLServer

```
1 netsh advfirewall firewall add rule name="SQL Server Porta 1433
  " dir=in action=allow protocol=TCP localport=1433
```

```
2 netsh advfirewall firewall add rule name="SQL Server Porta 1433"  
  " dir=out action=allow protocol=TCP localport=1433
```

Liberando as portas do Firewall para acesso eFolding Hosting:

Código-fonte 9: Script de Liberando as portas do Firewall para acesso eFolding Hosting

```
1 netsh advfirewall firewall add rule name="eFolding Hosting 1111"  
  " dir=in action=allow protocol=TCP localport=1111  
2 netsh advfirewall firewall add rule name="eFolding Hosting 1111"  
  " dir=out action=allow protocol=TCP localport=1111
```

A.8 Atualização da Aplicação (Se necessário)

As instruções que se seguem levam em conta as versões anteriores do eFolding na forma padrão de disponibilização, isto é, a versão não foi alterada internamente. A atualização para uma nova versão tem três pontos básicos:

- Aplicação de Servidor (servidor)
 - eFolding Hosting
 - Web Site
- Banco de dados
- Aplicações de Distribuição (clientes)

Observação: Antes de qualquer passo realize cópia de segurança (backup) de sua instalação atual.

A.8.1 Atualização do Servidor

Na existência de uma versão anterior (prévia) a uma nova instalação de versão, se faz necessário sobrescrever os diretórios e arquivos existentes com a nova versão disponibilizada. Nenhuma outra ação é necessária, após o sobreposição do aplicativo basta iniciá-lo novamente.

A.8.2 Atualização do banco de dados

A atualização de banco de dados (MSSQLServer) de versões anteriores deverão utilizar os scripts criados especificamente para esse fim. Para cada versão disponibilizada oficialmente há um scripts (em formato sql) específico de atualização. Esses scripts estão disponíveis Source Control do Projeto. A execução do script em ambiente SQL deve ser commitado (fazer permanentes um conjunto de mudanças experimentais) no banco de dados.

GUIA DO USUÁRIO

B.1 Introdução

Hoje em qualquer local do globo que possua conectividade com a Internet, nos conectamos com pessoas e aplicativos de todas as partes do mundo, simultaneamente, através de redes sociais, smartphones, TVs ou mesmo em computadores domésticos, em ambiente acadêmico ou corporativo. A Tecnologia da Informação (TI), quanto aos aplicativos criados e disponíveis pode ser descrita como "ponto de encontro de informações". Algumas dessas informações são geradas e transitadas neste mundo digital, o qual necessita ser seguro. Neste ambiente, o acesso às informações é permitido somente a algumas pessoas.

Com a utilização de TI o tempo é algo a ser medido, pois há interesse na economia de tempo utilizando, por exemplo, nas compras via internet, em cursos on-line, faculdades virtuais, dentre outras inúmeras opções, a TI traz ganhos para toda a sociedade. Neste contexto o Efolding também necessita assegurar o acesso às informações pertinentes em seu domínio, e condicionar a possibilidade de simulações realizadas à distância.

B.1.1 Objetivo

O foco deste material é providenciar um guia prático de como utilizar a plataforma de simulação, através da Internet. A plataforma visual de acesso foi desenvolvida utilizando os padrões de aplicações Web comumente existentes hoje.

B.2 Licença de utilização do software

"Uma licença de software é uma definição de ações autorizadas (ou proibidas), no âmbito do direito de autor de um programador de software de computador concedidas (ou impostas) ao usuário deste software."(JR., 2004)

Entende-se por usuário qualquer entidade legal, empresas ou um "usuário final (doméstico)", a origem do termo vem do inglês "end user license agreement (EULA)".

O acesso à ferramenta eFolding não necessita de licença; entretanto é necessário o cadastro prévio de usuário/pesquisador, que pretende utiliza-lo.

B.3 Requisitos de Sistema

Abaixo segue a lista dos requisitos mínimos e recomendados para utilização do aplicativo Web eFolding.

Web Browser	Chrome ou similar compatível
Conectividade	Acesso a internet
Equipamento	Qualquer Sistema Operacional compatível com Chrome

B.3.1 Requisitos mínimos de sistema

- **Windows:** Para utilizar o Chrome no Windows, é necessário o seguinte:
 1. Windows 7, Windows 8, Windows 8.1, Windows 10, ou posterior;
 2. Processador Intel Pentium 4 ou posterior compatível com SSE2;
- **Mac:** Para utilizar o Chrome no Mac, é necessário o seguinte:
 1. OS X Mavericks 10.9 ou posterior;
- **Linux:** Para utilizar o Chrome no Linux, é necessário o seguinte:
 1. Ubuntu de 64 bits 14.04 ou superior, Debian 8 ou superior, openSUSE 13.1 ou superior, ou Fedora Linux 21 ou superior;
 2. Processador Intel Pentium 4 ou posterior compatível com SSE2;

B.3.2 Como instalar o Web Browser

- **Windows**
 1. Baixar o Chrome.
 2. Clicar em Executar ou Salvar.
 3. Se escolher Salvar, clique duas vezes no arquivo de instalação para iniciar o processo de instalação. As definições da página inicial e o histórico do navegador são importadas automaticamente.
 4. Inicie o Chrome:

- a) Windows 7: uma vez concluída a configuração, é apresentada uma janela do Chrome.
- b) Windows 8 e 8.1: é apresentada uma caixa de diálogo de boas-vindas. Clique em Seguinte para selecionar o navegador pré definido.
- c) Windows 10: uma vez concluída a configuração, é apresentada uma janela do Chrome

- **Mac**

1. Transfira a imagem do disco.
2. Abra o arquivo com o nome "Google Chrome.dmg".
3. Na janela apresentada, procure o ícone do Chrome . Arraste-o para a pasta Aplicações. Se não possuir direitos de administrador, será solicitada a autenticação. Se não conseguir efetuar a autenticação, arraste o ícone para uma localização no computador onde tenha direitos, como o ambiente de trabalho.
4. Abra o Chrome

- **Linux**

Utilize o mesmo software que instala programas no seu computador para instalar o Chrome. Certifique-se primeiro de que tem acesso administrativo.

1. Transfira o arquivo de instalação.
2. Clique em OK para abrir o pacote.
3. Clique em Instalar pacote.

B.4 Procedimento de Login

O acesso à ferramenta é através da Internet no endereço <<http://efolding.fcfrp.usp.br>>, (através de qualquer navegador compatível com Chrome, em seu computador, tablet ou smartphone). Para utilizar os recursos disponíveis é necessário efetuar o Login. Esta ação requer a solicitação do *username* e *password* para autenticação. Abaixo segue o passo a passo da **ação Login**:

- Endereço Aplicação Web (eFolding): <<http://efolding.fcfrp.usp.br>>
- Procedimento: segue descrito abaixo passo a passo o procedimento para “Login”
 1. Acessar o endereço fornecido acima;
 2. Informar *Username* e *Password*;
 3. Clicar no botão “Log In”;
- Tempo aproximado de conclusão: O procedimento é realizado pelo próprio interessado, e leva poucos minutos para ser concluído.

B.4.1 Falha no procedimento de Login

Em caso de falha ao efetuar Login, uma mensagem de orientação é gerada na tela da aplicação, com a orientação ao o usuário. (Conforme imagem abaixo)

B.4.2 Procedimento de Logoff

O procedimento de Logoff, é a ação contrária do Login. Neste caso, a sessão junto ao aplicativo é finalizada. Caso for efetuado com sucesso, a ferramenta irá ser redirecionada para a tela abaixo:

B.5 Procedimento para Simulação

Através do menu é acessível a opção de “Simulations”, a qual permite o acesso à ferramenta de simulação. Observe que a opção é acessível (de forma similar ao “Log out”) somente depois de efetuado com sucesso o procedimento de Login.

Na opção de “Simulations”. É possível visualizar ou pesquisar simulações já existentes, ou, encaminhar uma nova simulação.

Bastante similar a muitos aplicativos Web, a ferramenta permite a visualização das últimas simulações ordenadas por data de criação. É possível navegar entre as páginas que contém as simulações, bem como visualizar número total de simulações já executadas. Nesta versão, a pesquisa pode ser efetuada através do ID da Simulação, palavras chave, nome do autor.

Label	Date	Guid of execution	Temperature	User	Status			
1233	17/06/2016 15:58:02	a66de3f7-f4d8-40a3-a6db-6777c9d34d47	0,02	admin	Waiting	Change status	Edit	
11111	16/06/2016 14:47:48	f800ea10-b4fb-439d-9a92-8d7e9f78fa99	1	admin	Created	Change status	Edit	Delete
1teste	14/05/2016 13:24:37	d7134f14-793f-4513-b58f-2aae6ab158f1	2	admin	ResultsProcessed	Change status	Edit	Download Results
1teste	14/05/2016 13:24:32	8faac73c-2ce1-4e34-8bcf-20d4cc5e56ca	1	admin	ResultsProcessed	Change status	Edit	Download Results
5	07/05/2016 14:16:12	938073e9-b2a9-48ad-939f-5fec730fb439	5	admin	ResultsProcessed	Change status	Edit	Download Results
1	07/05/2016 13:55:21	ba70cd6e-ba9c-4c18-ae19-2028db53e5dc	1	admin	ResultsProcessed	Change status	Edit	Download Results
1234567	07/05/2016 13:52:12	3314d7da-a03b-4065-b8d2-1d2663de64ff	0	admin	Error	Change status	Edit	
123	09/04/2016 18:04:48	73c1bb51-7f11-4582-a39b-fb55006d4b4a	3	admin	ResultsProcessed	Change status	Edit	Download Results
123	09/04/2016 18:04:47	283c122e-b3b2-47d3-a4b6-f42937ad61ef	2	admin	ResultsProcessed	Change status	Edit	Download Results
123	09/04/2016 18:04:46	0cf0a61c-7685-4d2b-bd6-897cad5f7c39	1	admin	ResultsProcessed	Change status	Edit	Download Results

First Previous Next Last

Total: 24 Actual page: 1/3

A ferramenta de pesquisa efetua a busca por aproximação, por exemplo: a palavra “teste”.

Simulations

Q teste Pesquisar New Simulation

List

Label	Date	Guid of execution	Temperature	User	Status			
1teste	14/05/2016 13:24:37	d7134f14-793f-4513-b58f-2aae6ab158f1	2	admin	ResultsProcessed	Change status	Edit	Download Results
1teste	14/05/2016 13:24:32	8faac73c-2ce1-4e34-8bcf-20d4cc5e56ca	1	admin	ResultsProcessed	Change status	Edit	Download Results
teste	09/04/2016 18:03:28	3de15ae7-38a2-4889-9a84-2f7bd5d4d6e	1	admin	ResultsProcessed	Change status	Edit	Download Results
teste	09/04/2016 17:59:20	2c382a8f-eb74-4017-945c-7f4044fb1448	2	admin	ResultsProcessed	Change status	Edit	Download Results
teste	09/04/2016 17:58:40	afd51e03-6e9e-4232-94c6-d6bbb9f416e3	1	admin	ResultsProcessed	Change status	Edit	Download Results

Ação: Efetuar pesquisa de simulações

- Endereço Aplicação Web (eFolding): <http://efolding.fcfrp.usp.br>
- Procedimento: Abaixo segue descrito passo a passo para procedimento pesquisa de Simulações. Primeiramente, o usuário deve estar logado na aplicação.
 1. Acessar ferramenta de pesquisa clicando no item de menu “Simulations”;
 2. Uma tela será apresentada com listagem de todas as simulações existentes, de forma paginada;
 3. Fornecer a keyword de busca no campo de pesquisa;
 4. Clicar no botão “Search”;
- Tempo aproximado de conclusão O procedimento é realizado pelo próprio interessado, e leva poucos segundos para ser concluído.

B.5.1 Encaminhar uma simulação para ser processada

Como encaminhar uma nova simulação para ser processada? Após a criação de uma nova simulação, o item deve ser liberado para processamento, mudando o seu status para “Sent to pipeline”. A opção “Decline” cancela o item a ser simulado.

The screenshot shows the 'Simulations' page of the Grid Protein Folding application. At the top, there is a search bar with the placeholder 'Pesquisar' and a 'New Simulation' button. Below the search bar is a table titled 'Simulations' with columns: Label, Date, Guid of execution, Temperature, User, Status, and actions. The table lists several simulations. The simulation with label 'tteste' (ID 14/05/2016 13:24:37) is highlighted, and its dropdown menu is open, showing options: 'Change status', 'Send to pipeline', and 'Decline'. A red arrow points to the 'Send to pipeline' option.

Label	Date	Guid of execution	Temperature	User	Status	Actions
1233	17/06/2016 15:58:02	a66de3f7-f4d9-40a3-a6db-6777c9d34d47	0,02	admin	Waiting	Change status Edit
11111	16/06/2016 14:47:48	f800ea10-b4fb-439d-9a92-8d7e9f78fa99	1	admin	Created	Change status Edit Delete
tteste	14/05/2016 13:24:37	d7134f14-793f-4513-b58f-2aae6ab158f1	2	admin	ResultsProcessed	Change status Send to pipeline Decline Edit Download Results
tteste	14/05/2016 13:24:32	8faac73c-2ce1-4e34-8bcf-20d4cc5e56ca	1	admin	ResultsProcessed	Change status Edit Download Results
5	07/05/2016 14:16:12	938073e9-b2a9-48ad-939f-5fec730fb439	5	admin	ResultsProcessed	Change status Edit Download Results
1	07/05/2016 13:55:21	ba70cd6e-ba9c-4c18-ae19-2028db53e5dc	1	admin	ResultsProcessed	Change status Edit Download Results
1234567	07/05/2016 13:52:12	3314d7da-a03b-4065-b8d2-1d2663de64ff	0	admin	Error	Change status Edit
123	09/04/2016 18:04:48	73c1bb651-7f11-4582-a39b-fb55006d4b4a	3	admin	ResultsProcessed	Change status Edit Download Results
123	09/04/2016 18:04:47	283c122e-b3b2-47d3-a4b6-f42937ad61ef	2	admin	ResultsProcessed	Change status Edit Download Results
123	09/04/2016	0cf0a61c-7685-4d2b-bbd6-	1	admin	ResultsProcessed	Change status Edit Download

Ação: Encaminhando uma simulação para processamento

- Endereço: Aplicação Web (eFolding) <<http://efolding.fcfrp.usp.br>>
- Procedimento: Abaixo segue descrito passo a passo o procedimento para encaminhar uma simulação para o processamento. Primeiramente, o usuário deve estar logado na aplicação.
 1. Acessar ferramenta de pesquisa clicando no item de menu “Simulations”;
 2. Efetuar a pesquisa condicionada a sua necessidade, e após;
 3. Alterar o status da sua simulação, através do dropdown disponibilizado chamado de “change status”. O status de “Send to pipeline” encaminha a simulação para a fila de processamento;
- Tempo aproximado de conclusão O procedimento é realizado pelo próprio interessado, e leva poucos minutos para ser concluído.

B.6 Criando uma nova Simulação.

A ação de criação de uma nova simulação é iniciada após acionamento da opção “New Simulation”.

Simulations

Search:

List

Label	Date	Guid of execution	Temperature	User	Status			
1233	17/06/2016 15:58:02	a66de3f7-f4d8-40a3-a6db-6777c9d34d47	0,02	admin	Waiting	Change status	Edit	
11111	16/06/2016 14:47:48	f800ea10-b4fb-439d-9a92-8d7e9f78fa99	1	admin	Created	Change status	Edit	Delete
1teste	14/05/2016 13:24:37	d7134f14-793f-4513-b58f-2aae6ab158f1	2	admin	ResultsProcessed	Change status	Edit	Download Results
1teste	14/05/2016 13:24:32	8faac73c-2ce1-4e34-8bcf-20d4cc5e56ca	1	admin	ResultsProcessed	Change status	Edit	Download Results
5	07/05/2016 14:16:12	938073e9-b2a9-48ad-939f-5fec730fb439	5	admin	ResultsProcessed	Change status	Edit	Download Results
1	07/05/2016 13:55:21	ba70cd6e-ba9c-4c18-ae19-2028db53e5dc	1	admin	ResultsProcessed	Change status	Edit	Download Results
1234567	07/05/2016 13:52:12	3314d7da-a03b-4065-b8d2-1d2663de64ff	0	admin	Error	Change status	Edit	
123	09/04/2016 18:04:48	73c1bb51-7f11-4582-a39b-f05500b04b4a	3	admin	ResultsProcessed	Change status	Edit	Download Results
123	09/04/2016 18:04:47	283c122e-b3b2-47d3-a4b6-f42937ad61ef	2	admin	ResultsProcessed	Change status	Edit	Download Results
123	09/04/2016	0cf0a61c-7685-4d2b-bbd6-	1	admin	ResultsProcessed	Change status	Edit	Download

Após, ocorre redirecionamento para a página de criação/configuração de uma nova simulação.

Grid Protein Folding

localhost:50950/Pages/SimulationAddEdit.aspx

Home Simulations User: admin [Log Out]

Welcome!

This is Website for Computer Simulation Distributed: Application of Folding Heteropolymers problems

Simulations

Add experiment

Informations:

Label of identification:

Isem:

Number of iterations: McSteps: 1350000

Temperature:

Total of sitio:

Type:

Select the Model:

Note:

Configuration of e-mail:

Hints:

Informations (Informações para a simulação)

- Label of identification** - Rotulo de identificação da simulação.
- Isem** - Número inicial da semente. Deve ser um número inteiro impar.
- Number of iterations** - Número de interações para geração Monte Carlo Step.
- Temperature** - Temperatura utilizada na simulação. (1=27 Celsius)
- Total of sitio** - Número de sitios da cadeia.
- Type** - Define o tipo da cadeia a ser utilizada na simulação. O tipo REAL não ocorre sobreposição, entretanto existe vizinhos de

Para finalizar a criação basta efetuar click no botão “Finish” disponível no final do formulário. As críticas de preenchimento e mensagens de *warning* serão exibidas na região de sumário, localizada acima do botão de “Finish”.

arquivo. O split de arquivo é necessário em cenários os quais os passo de Monte Carlo são muito longos. Em algumas máquinas poderá ocorrer problemas de I/O. Indicado é utilizar o valor 1000.

Log:

Ação: Criando um novo item a ser simulado

- Endereço: Aplicação Web (eFolding) <<http://efolding.fcfrp.usp.br>>
- Procedimento: Abaixo segue descrito passo a passo o procedimento para encaminhar uma simulação para o processamento. Primeiramente, o usuário deve estar logado na aplicação.
 1. Acessar ferramenta de pesquisa clicando no item de menu “Simulations”;
 2. Acessar ferramenta de “New Simualtion” ao clicar no botão “New Simulations”;
 3. Será apresentado um formulário para preenchimento:

- a) Ao lado direito do formulário existe uma tabela de dicas para o preenchimento do formulário;
- b) Após preencher os campos, deve-se clicar no botão “Finish”
- Tempo aproximado de conclusão O procedimento é realizado pelo próprio interessado, e leva poucos minutos para ser concluído.

B.7 Como editar e deletar uma Simulação?

Por regra, após uma simulação ter sido concluída, a mesma não poderá ser excluída. A sua edição permite a alteração de alguns atributos. Dessa forma é possível somente editar/deletar a simulação quando ela atende algumas premissas, conforme figura abaixo:

The screenshot shows a web browser window with the URL `localhost:50950/Pages/Simulation.aspx`. The page title is "Grid Protein Folding". The navigation bar includes "Home", "Simulations", and "User: admin [Log Out]". The main content area is titled "Simulations" and contains a search bar with the text "Pesquisar" and a "New Simulation" button. Below the search bar is a "List" section containing a table of simulations.

Label	Date	Guid of execution	Temperature	User	Status	Actions
1233	17/06/2016 15:58:02	a66de3f7-f4d8-40a3-a6db-6777c9d34d47	0,02	admin	Waiting	Change status Edit
11111	16/06/2016 14:47:48	f800ea10-b4fb-439d-9a92-8d7e9f78fa99	1	admin	Created	Change status Edit Delete
1teste	14/05/2016 13:24:37	d7134f14-793f-4513-b58f-2aae6ab158f1	2	admin	ResultsProcessed	Change status Edit Download Results
1teste	14/05/2016 13:24:32	8faac73c-2ce1-4e34-8bcf-20d4cc5e56ca	1	admin	ResultsProcessed	Change status Edit Download Results
5	07/05/2016 14:16:12	938073e9-b2a9-48ad-939f-5fec730fb439	5	admin	ResultsProcessed	Change status Edit Download Results
1	07/05/2016 13:55:21	ba70cd6e-ba9c-4c18-ae19-2028db53e5dc	1	admin	ResultsProcessed	Change status Edit Download Results
1234567	07/05/2016 13:52:12	3314d7da-a03b-4065-b8d2-1d2663de64ff	0	admin	Error	Change status Edit
123	09/04/2016 18:04:48	73c1bb51-7f11-4582-a39b-fb55006d4b4a	3	admin	ResultsProcessed	Change status Edit Download Results
123	09/04/2016 18:04:47	283c122e-b3b2-47d3-a4b6-f42937ad61ef	2	admin	ResultsProcessed	Change status Edit Download Results
				admin	ResultsProcessed	Change status Edit Download

Ação: Editando uma simulado

- Endereço: Aplicação Web (eFolding) <http://efolding.fcfrp.usp.br>
- Procedimento: Abaixo segue descrito passo a passo o procedimento para encaminhar uma simulação para o processamento. Primeiramente, o usuário deve estar logado na aplicação.
 1. Acessar feature de pesquisa clicando no item de menu “Simulations”;
 2. Efetuar a pesquisa condicional para encontrar a sua Simulação
 3. Clicar no botão de “Edit”

- Ocorrerá o redirecionamento para a página de edição, similar a página de criação de uma nova simulação;
 - Ao lado direito do formulário existe uma tabela de dicas para o preenchimento do formulário;
 - Após preencher os campos, deve-se clicar no botão “Finish”
- Tempo aproximado de conclusão O procedimento é realizado pelo próprio interessado, e leva poucos minutos para ser concluído.

B.8 Obtendo os resultados de uma Simulação

Após conclusão de uma simulação, é possível efetuar o download dos seus resultados. O procedimento é muito simples, basta acionar o link button disponível na tela de simulações; quando disponível, o link “Download Results” estará apto. Você receberá o arquivo no formato ZIP com o conteúdo incluso.

The screenshot shows the 'Simulations' page of the Grid Protein Folding application. At the top, there is a search bar and a 'New Simulation' button. Below is a table listing various simulations. A red arrow points to the 'Download Results' link for the simulation with Label '1teste'.

Label	Date	Guid of execution	Temperature	User	Status	Actions
1293	17/06/2016 15:58:02	a66de3f7-f4d8-40a3-a6db-6777c9d34d47	0,02	admin	Waiting	Change status ▼ Edit
11111	16/06/2016 14:47:48	f800ea10-b4fb-439d-9a92-8d7e9f78fa99	1	admin	Created	Change status ▼ Edit Delete
1teste	14/05/2016 13:24:37	d7134f14-793f-4513-b58f-2aae6ab158f1	2	admin	ResultsProcessed	Change status ▼ Edit Download Results
1teste	14/05/2016 13:24:32	8faac73c-2ce1-4e34-8bcf-20d4cc5e56ca	1	admin	ResultsProcessed	Change status ▼ Edit Download Results
5	07/05/2016 14:16:12	938073e9-b2a9-48ad-939f-5fec730fb439	5	admin	ResultsProcessed	Change status ▼ Edit Download Results
1	07/05/2016 13:55:21	ba70cd6e-ba9c-4c18-ae19-2028db53e5dc	1	admin	ResultsProcessed	Change status ▼ Edit Download Results
1234567	07/05/2016 13:52:12	3314d7da-a03b-4065-b8d2-1d2663de64ff	0	admin	Error	Change status ▼ Edit
123	09/04/2016 18:04:48	73c1bb51-7f11-4582-a39b-fb55006d4b4a	3	admin	ResultsProcessed	Change status ▼ Edit Download Results
123	09/04/2016 18:04:47	283c122e-b3b2-47d3-a4b6-f42937ad61ef	2	admin	ResultsProcessed	Change status ▼ Edit Download Results
		d7134f14-793f-4513-b58f-2aae6ab158f1		admin	ResultsProcessed	Change status ▼ Edit Download

O nome do arquivo será o mesmo descrito na coluna “Guid of execution”.

Grid Protein Folding x

localhost:50950/Pages/Simulation.aspx

Home Simulations User: admin [Log Out]

Simulations

Q Pesquisar New Simulation

List

Label	Date	Guid of execution	Temperature	User	Status				
1233	17/06/2016 15:58:02	a66de3f7-f4d8-40a3-a6db-6777c9d34d47	0,02	admin	Waiting	Change status	Edit		
11111	16/06/2016 14:47:48	f800ea10-b4fb-439d-9a92-8d7e9f78fa99	1	admin	Created	Change status	Edit	Delete	
1teste	14/05/2016 13:24:37	d7134f14-793f-4513-b58f-2aae6ab158f1	2	admin	ResultsProcessed	Change status	Edit	Download Results	
1teste	14/05/2016 13:24:32	8faac73c-2ce1-4e34-8bcf-20d4cc5e56ca	1	admin	ResultsProcessed	Change status	Edit	Download Results	
5	07/05/2016 14:16:12	938073e9-b2a9-48ad-939f-5fec730fb439	5	admin	ResultsProcessed	Change status	Edit	Download Results	
1	07/05/2016 13:55:21	ba70cd6e-ba9c-4c18-ae19-2028db53e5dc	1	admin	ResultsProcessed	Change status	Edit	Download Results	
1234567	07/05/2016 13:52:12	3314d7da-a03b-4065-b8d2-1d2663de64ff	0	admin	Error	Change status	Edit		
123	09/04/2016 18:04:48	73c1bb51-7f11-4582-a39b-fb55006d4b4a	3	admin	ResultsProcessed	Change status	Edit	Download Results	
123	09/04/2016	283c122e-b3b2-47d3-a4b6-	2	admin	ResultsProcessed	Change status	Edit	Download	

d7134f14-793f-...zip Mostrar todos os downloads...

Ação: Obtendo dados pós Simulação

- Endereço: Aplicação Web (eFolding) <<http://efolding.fcfrp.usp.br>>
- Procedimento: Abaixo segue descrito passo a passo o procedimento para encaminhar uma simulação para o processamento. Primeiramente, o usuário deve estar logado na aplicação.
 1. Acessar feature de pesquisa clicando no item de menu “Simulations”;
 2. Efetuar a pesquisa condicional para encontrar a sua Simulação
 3. Clicar no botão de “Download Results”
 - a) Ocorrerá download de um arquivo no formato ZIP. Este arquivo contém resultados da simulação;
- Tempo aproximado de conclusão O procedimento é realizado pelo próprio interessado, e leva poucos minutos para ser concluído.

LOG DE EXCESSÃO E ERRO DA APLICAÇÃO

C.1 Introdução

"Em computação, log de dados é uma expressão utilizada para descrever o processo de registro de eventos relevantes num sistema computacional. Esse registro pode ser utilizado para restabelecer o estado original de um sistema ou para que um administrador conheça o seu comportamento no passado. Um arquivo de log pode ser utilizado para auditoria e diagnóstico de problemas em sistemas computacionais."(WIKIPEDIA, 2016) (DONY CHRISTOPHE; KNUDSEN, 2006)

C.2 Objetivo

O objetivo desta sessão é descrever como tratar, ler e acessar o log da aplicação, em detrimento a solução adotada para eFolding, tanto para aplicação cliente, quanto para a aplicação servidor.

C.2.1 *Processo de logging de eventos para a plataforma eFolding*

O procedimento de logging (logar) de eventos (erros, avisos e etc) está presente na solução do eFolding, tanto para o servidor (HostDistributed eFolding) tanto para a aplicação cliente e (ProteinFoldingService eFolding). Para ambos os cenários, ter uma implementação efetiva de logging em nossas aplicações pode acelerar consideravelmente processos de depuração e/ou apoio ao cliente na investigação de um possível problema ou excessão não prevista.

Para o projeto eFolding foi utilizado a referência do consolidado Log4Net, pacote o qual é amplamente conhecido e utilizado pela comunidade TI (Tecnologia de Informação) no amago acadêmico e empresarial. O código fonte, licença de uso e do procedimento de Logging junto as

suas referência técnicas estão disponíveis em <<https://logging.apache.org/log4net/>> e fonte em <<https://github.com/ElemarJR/Learning.Log4Net>>.

Mas, o que seria log4Net? Segundo o site oficial *"The Apache log4net library is a tool to help the programmer output log statements to a variety of output targets. log4net is a port of the excellent Apache log4jTM framework to the Microsoft® .NET runtime. We have kept the framework similar in spirit to the original log4j while taking advantage of new features in the .NET runtime."* (GUNDERLOY, 2004)

Em resumo é uma ferramenta para ajudar registrar a saída de eventos a um destino de saída (arquivo de log). Para o nosso aplicativo cliente ProteinFoldingService eFolding, a configuração de localização do arquivos de log está configurado como padrão no caminho "C:\GridProteinFolding\Client\Application.log".

De forma similar para o nosso aplicativo servidor HostDistributed eFolding, a configuração de localização do arquivos de log está configurado como padrão no caminho:

"C:\GridProteinFolding\Server\HostDistributed.log".

A configuração esta definida no arquivo log4net.config existente na pasta raiz de ambas aplicações.

C.2.2 A trilha de auditoria

A trilha de auditoria é uma coleção de arquivos de log de texto que contêm informações sobre ocorrência de log de erros, falhar e avisos. Com a Trilha de Auditoria, você pode gerar relatórios, entender e validar causa raiz de eventuais problemas que decorram com o passos do tempo junto a utilização do aplicativo.

A trilha de auditoria (a formatação das informações no arquivo de LOG) é composto pelo seguinte layout:

```
%date [%thread] %-5level %logger [%ndc] - %message%newline
```

Onde:

- %date = Usado para a saída da data do evento de registro no fuso horário local.
Resumo Técnico: Usado para a saída da data do evento de registro no fuso horário local. Para emitir a data com a hora universal usar o padrão %utcddate. O especificador de conversão de data pode ser seguido por um especificador de formato de data fechado entre chaves. Por exemplo, a %date HH: mm: ss, fff ou %date dd MMM aaaa HH: mm: ss, fff. Se nenhum formato de data especificador é dado em seguida, formato ISO8601 é assumido (Iso8601DateFormatter).

- [%thread] = Usadas para transmitir o nome do segmento que gerou o evento de log. Usa o número do segmento se nenhum nome está disponível.
- %-5level = Usado para a saída do nível do evento de log .
Resumo técnico: Quando,
Off - O nível Off designa um nível mais alto do que todo o resto.
Fatal - O nível Fatal designa eventos de erro muito graves que será presumivelmente levar a aplicação para abortar.
Erro - O nível de erro designa eventos de erro que ainda possam permitir a aplicação para continuar funcionando.
Avisar - O nível alertar designa situações potencialmente nocivas.
Debug - O nível de depuração designa eventos informativos refinadas que são mais úteis para depurar um aplicativo.
Todos - O nível Todos designa o nível mais baixo possível .
- %logger = Em complemento ao item anterior. Resumo técnico: Usado para a saída do logger do evento de log. O especificador de conversão logger pode ser opcionalmente seguido por especificador de precisão, que é uma constante decimal entre parênteses.
Se um especificador de precisão é dada, então, apenas o número correspondente de certas maioria dos componentes do nome logger serão impressos. Por padrão o nome logger é impresso na íntegra. Por exemplo, para o nome do logger "a.b.c" o padrão verb |
- [%ndc] = Usado para a saída do NDC (contexto de diagnóstico aninhada) associado com o segmento que gerou o evento de log .
- %message% = Usadas para transmitir o nome do método em que o pedido de registro foi emitido. Resumo técnico: Utilizado para transmitir o nome do método em que o pedido de registro foi emitido.
AVISO Gerar informações de localização da chamada é extremamente lento. Seu uso deve ser evitado a menos que a velocidade de execução não seja um problema.
- newline = Saídas da linha dependente caractere separador plataforma ou caracteres.
Resumo técnico: Saídas do log separado por linha, dependente do uso do caractere separador da plataforma (Sistema Operacional). Este padrão de conversão oferece o mesmo desempenho como a utilização de "corta linha" de separação não-portáteis, como verb | " n ou verb | " r n. Assim, esta é utilizada como forma preferida de especificar uma linha separadora entre desenvolvedores de softwares.

Abaixo seguem a configuração padrão para o cliente e servidor.

Client eFolding

Código-fonte 10: Configuração padrão para obtenção do LOG no ambiente cliente

```

1      <appender name="FileAppender" type="log4net.Appender.
      FileAppender">
2          <file value="C:\GridProteinFolding\Client\Application.log
      " />
3          <appendToFile value="true" />
4          <staticLogFileName value="true" />
5          <rollingStyle value="Size" />
6          <maxSizeRollBackups value="5" />
7          <maximumFileSize value="1MB" />
8          <lockingModel type="log4net.Appender.FileAppender+
      MinimalLock" />
9          <layout type="log4net.Layout.PatternLayout">
10             <conversionPattern value="%date [%thread] %-5level %
      logger [%ndc] - %message%newline" />
11         </layout>
12     </appender>
13
14     <root>
15         <level value="INFO" />
16         <appender-ref ref="ConsoleAppender" />
17         <appender-ref ref="FileAppender" />
18     </root>

```

Host eFolding

Código-fonte 11: Configuração padrão para obtenção do LOG no ambiente servidor

```

1 <appender name="FileAppender" type="log4net.Appender.
      FileAppender">
2     <file value="C:\GridProteinFolding\Server\HostDistributed
      .log" />
3     <appendToFile value="true" />
4     <staticLogFileName value="true" />
5     <rollingStyle value="Size" />
6     <maxSizeRollBackups value="5" />
7     <maximumFileSize value="1MB" />

```

```
8      <lockingModel type="log4net.Appender.FileAppender+
MinimalLock" />
9      <layout type="log4net.Layout.PatternLayout">
10         <conversionPattern value="%date [%thread] %-5level %
logger [%ndc] - %message%newline" />
11     </layout>
12 </appender>
13
14 <root>
15     <level value="INFO" />
16     <appender-ref ref="ConsoleAppender" />
17     <appender-ref ref="FileAppender" />
18 </root>
19 </log4net>
```

Dicas: As dicas abaixo são referências adicionais ao arquivo de configuração para saída dos eventos de excessão (Log).

- `<appendToFile value="true"/>` = arquivo de log sempre será incremental
- `<staticLogFileName value="true"/>` = true if always should be logged to the same file, otherwise false.
- `<rollingStyle value="Size"/>` = can be either Date, Size or Composite. the default setting Composite, uses a combination of Size and Date settings. Thus if you have the datePattern set to “.yyyy-MM-dd” and maxSizeRollBackups set to 10, themn it will maintain 10 log backups for each day.
- `<maxSizeRollBackups value="5"/>` = The maximum number of backup files that are kept before the oldest is erased.
- `<maximumFileSize value="1MB"/>` = The maximum size that the output file is allowed to reach before being rolled over to backup files.
- `<appender-ref ref="ConsoleAppender"/>` = ConsoleAppender appends log events to System.out or System.err using a layout specified by the user. The default target is System.out.
- `<appender-ref ref="FileAppender"/>` = FileAppender appends log events to a file.