UNIVERSIDADE DE SÃO PAULO

FFCLRP - DEPARTAMENTO DE FÍSICA E MATEMÁTICA

PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA APLICADA À MEDICINA E BIOLOGIA

Técnicas de Controle da Diversidade de Populações em Algoritmos Genéticos para Determinação de Estruturas de Proteínas

Vinicius Tragante do Ó

Dissertação apresentada à Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto da USP, como parte das exigências para a obtenção do título de Mestre em Ciências, Área: Física Aplicada à Medicina e Biologia.

RIBEIRÃO PRETO – SP

2009

UNIVERSIDADE DE SÃO PAULO

FFCLRP - DEPARTAMENTO DE FÍSICA E MATEMÁTICA

PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA APLICADA À MEDICINA E BIOLOGIA

Técnicas de Controle da Diversidade de Populações em Algoritmos Genéticos para Determinação de Estruturas de Proteínas

Vinicius Tragante do Ó

Orientador: ***Prof. Dr. Renato Tinós***

RIBEIRÃO PRETO - SP

2009

A meus pais, que sempre me deram suporte
para que esta e outras conquistas viessem.

Agradecimentos

Gostaria de agradecer a todos que foram importantes nesta caminhada, que foi de muito crescimento pessoal e acadêmico.

Agradeço ao professor Renato Tinós, sempre solícito e paciente, até nas horas em que meu desempenho era inferior ao esperado; ao professor Antonio Luiz Rodrigues Júnior, por me ajudar a ampliar (ou remover) as fronteiras da Informática Biomédica; ao professor Fernando Luis Barroso da Silva, por ajudar a melhorar este trabalho; e a todos os professores que me passaram um pouco do que sabem em suas disciplinas, e me permitiram ter o pouco conhecimento que tenho hoje.

Agradeço ao programa de Física Aplicada à Medicina e Biologia pela oportunidade, e à CAPES pelo financiamento.

Agradeço também a meus pais, cuja presença, por si só, reconforta e renova a vontade de vencer os desafios; às minhas irmãs, avôs e avós, tios e tias, primos e primas, pelos bons momentos vividos juntos.

Agradeço à Vivian, minha namorada, por estar ao meu lado sempre que possível, e me ajudar a pensar tudo por outro ponto de vista.

Agradeço aos meus amigos em Bauru pelas únicas horas em que foi possível esquecer as obrigações do mestrado! Não posso me esquecer também do pessoal do LIS, que literalmente "suou a camisa" comigo em 2008, e ajudou a diminuir a dificuldade do percurso.

Também tem o pessoal do tênis, do futebol, da academia, que me ajudaram a manter o humor até nas horas mais difíceis...

Enfim, foi uma oportunidade de engrandecimento. Agradeço a todos que me ajudaram a perceber isto e efetivamente crescer.

Resumo

TRAGANTE, V. (2009). *Técnicas de Controle da Diversidade de Populações em Algoritmos Genéticos para Determinação de Estruturas de Proteínas*. Ribeirão Preto, 2009. 97p. Dissertação (Mestrado) – Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto, Universidade de São Paulo.

Recentemente, pesquisadores têm proposto o uso de Algoritmos Genéticos (AGs) para a determinação da estrutura tridimensional de proteínas. No entanto, este é um problema difícil para um AG tradicional, pois na maioria das vezes ocorre a convergência prematura das soluções para ótimos locais. Isto ocorre porque o uso de mecanismos de seleção no AG acarreta uma perda da diversidade das soluções. Assim, neste trabalho, são investigadas estratégias para controlar a diversidade da população do AG e evitar que a solução fique rapidamente presa em ótimos locais. São empregadas bases de dados de ângulos de torção para a cadeia principal, cadeia lateral e técnicas de controle de diversidade em AGs conhecidas como Hipermutação e Imigrantes Aleatórios. Além disso, um novo algoritmo baseado no AG com Imigrantes Aleatórios Auto-Organizáveis é proposto. Os resultados mostram que estas variações são efetivas no objetivo de não manter o conjunto de soluções preso a uma região apenas, além de melhorar o desempenho para o problema de determinação de estruturas terciárias de proteínas.

Palavras-chave: Algoritmos Genéticos, Estruturas de Proteínas, Hipermutação, Imigrantes Aleatórios, Auto-Organização.

Abstract

TRAGANTE, V. (2009). *Control of the Population Diversity in Genetic Algorithms for the Determination of Protein Structures*. Ribeirão Preto, 2009. 97p. Dissertation (Master's Degree) – Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto, Universidade de São Paulo.

Recently, researchers have proposed the use of Genetic Algorithms (GAs) for the determination of the three-dimensional structure of proteins. However, this problem is considered a difficult problem for the standard GA, because most of the cases the convergence occurs early, into local minima instead of the global optimum. This occurs because the use of selection mechanisms in the GA leads to a loss of diversity of solutions. With this in mind, in this work, strategies to control the diversity of the population in the GA are investigated in order to avoid the solution subset to be early caught in local optima. Database sets of torsion angles for the main chain and the side chain are employed, and also modifications in the GAs, known as Hypermutation and Random Immigrants. Besides these approaches, a new algorithm based on the Self-Organizing Random Immigrants is proposed. Results show that these changes are effective in the goal of avoiding the results ensemble to be trapped in a region, and also help improve the performance for the protein structure prediction problem.

Keywords: Genetic Algorithms, Protein Structure Prediction, Hypermutation, Random Immigrants, Self-Organization.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

AG	Algoritmo Genético
DOP	*Dynamic Optimization Problems* (Problemas de Otimização dinâmica)
GA	*Genetic Algorithm* (Algoritmo Genético)
RMSD	*Root Mean Square Deviation* (Desvio de raíz media quadrada)
SORIGA	*Self-Organized Random Immigrants Genetic Algorithm* (Algoritmo Genético com Imigrantes Aleatórios Auto-organizados)

1. INTRODUÇÃO

A busca pela cura de diversas doenças existentes passa em muitos casos pela criação de fármacos que as combatam. Estes fármacos geralmente possuem ação sobre diversos órgãos e causam efeitos colaterais, que em alguns casos requerem novos medicamentos, criando um círculo vicioso de dependência química.

Com maiores recursos de pesquisa, e o advento da genômica e da biologia molecular, torna-se possível desenvolver fármacos com ação direta sobre o problema a ser atacado, por interações entre moléculas de ação direta em tecidos e estruturas microscópicas afetadas, hoje mais conhecidas que no passado [Drews, 2000]. No entanto, os custos de produção de novos fármacos ultrapassam a casa das dezenas de milhões de dólares, pois para cada doença há um conjunto de genes que pode ser relacionado a seus efeitos, e cada um deles se torna um alvo potencial para a ação de um fármaco. Segundo Blundell e Mizuguchi [Blundell & Mizuguchi, 2000], o custo para determinação de uma estrutura protéica por métodos não computacionais gira em torno de US$ 100.000,00. A indústria farmacêutica investigava em 2000 em torno de 500 genes-alvo. Hoje, com o término do processo de sequenciamento do genoma humano, o número de genes-alvo a serem investigados se tornou maior, e por consequência, os custos de produção de novos fármacos aumentam, pois a investigação de estruturas com potencial para agir em genes-alvo se torna mais ampla, demorada e custosa.

Por consequência, o custo final do fármaco aumenta para o consumidor, por problemas como direitos intelectuais de exploração. A indústria farmacêutica é considerada uma indústria de risco, pela alta taxa de experimentos infrutíferos que são normalmente executados; no entanto, um ranking da Revista Forbes indicou que entre as 500 maiores empresas do mundo, as 10 com maiores lucros são farmacêuticas, pois um remédio bem-sucedido torna-se um *blockbuster*, como dito no jargão da área, atingindo faturamento superior a 1 bilhão de dólares/ano [Jannuzzi *et al.*, 2008]. Jannuzzi [Jannuzzi *et al.*, 2008] cita ainda que a política da indústria farmacêutica é a de cobrar o maior valor possível para o medicamento de maneira que o cliente ainda aceite pagar. Sabendo-se disso, métodos que possam

diminuir os custos experimentais de criação, reduzindo o número de alvos e o número de potenciais estruturas a serem investigadas são essenciais para reduzir o custo final de produção, de modo que haja menos motivos para que o medicamento tenha um preço final tão elevado. Neste sentido, ferramentas computacionais podem ser a solução, ao permitir que simulações da interação entre uma molécula e um gene-alvo sejam realizadas com precisão, e moléculas com baixo potencial sejam rapidamente descartadas sem que tantos testes experimentais sejam necessários.

Outras áreas que podem se beneficiar destas simulações são a indústria alimentícia e até mesmo a indústria fotográfica, a partir de técnicas que diminuam o custo final de produção.

As moléculas em questão são as proteínas. Estas moléculas orgânicas (cap. 2), entre outras características, possuem a propriedade de se ligar a uma molécula-alvo, inibindo ou ativando sua ação, o que é de suma importância no tratamento de doenças [Biswas & Roy, 1995].

Atualmente, para determinar com precisão a estrutura tridimensional de uma proteína, utilizam-se os métodos de cristalografia e ressonância magnética nuclear [Han & Kambert, 2001]. No entanto, a cristalografia está sujeita a fatores externos, como má cristalização por efeito da gravidade que atrapalham a determinação da estrutura, e o fato de que a determinação do cristal é custosa em termos de tempo e recursos. No caso da ressonância magnética, restrições de tamanho da estrutura protéica a ser determinada diminuem a sua faixa de aplicabilidade. Sendo assim, métodos adicionais a estes processos são bem-vindos.

Teoricamente, é possível determinar uma estrutura protéica a partir da sequência primária de aminoácidos que a compõe [Ginalski *et al.*, 2005], se utilizada uma simulação refinada de processos físicos, teoria conhecida como Hipótese Termodinâmica de Anfinsen [Anfinsen, 1973], em um processo de dobramento (*folding*) de proteínas. Com a capacidade de processamento atualmente disponível, é possível simular muitas das características presentes nas proteínas em relação a suas características físicas e de ligações químicas, apesar de nem todas as interações serem atualmente passíveis de modelagem computacional. Entretanto, a capacidade computacional ainda não é suficiente para que simulações com mecânica quântica, que seria o método mais oportuno, sejam realizadas [Anile *et al.*, 2006].

É possível classificar os esforços computacionais atualmente empregados para a determinação de estruturas de proteína em quatro grupos [Floudas *et al.*, 2006]:

- Modelagem comparativa ou por homologia: por este método, compara-se a sequência de aminoácidos de cuja estrutura se deseja determinar com outras que possuam sequências similares, baseado na observação de que sequências similares geralmente possuem estruturas similares. No entanto, bons resultados podem ser atingidos apenas quando há similaridade superior a 50% na sequência de aminoácidos [Floudas *et al.*, 2006];
- Reconhecimento de formato: baseia-se no conceito de que estruturas são geralmente mais conservadas que sequências; assim, proteínas análogas podem servir como molde para determinação de estruturas. Em geral, é efetivo para estruturas menores, de até 100 resíduos [Kihara & Skolnick, 2003];
- Primeiros Princípios ou *Ab Initio* sem informações de base de dados: no caso de ausência de proteínas homólogas para serem comparadas, a modelagem por *Ab Initio* se torna a única saída, baseada na Termodinâmica [Santana *et al.*, 2008], a qual define que a estrutura nativa da proteína é aquela na qual a energia potencial atinge o mínimo global. Baseado nisso, vários métodos de busca pela estrutura nativa definem uma aproximação da energia da proteína e usam algoritmos de otimização que procuram a formação que minimiza esta energia, sem utilização de quaisquer informações obtidas de proteínas cujas estruturas já são conhecidas [Floudas *et al.*, 2006];
- Primeiros Princípios ou *Ab Initio* com informações de base de dados: os métodos atuais de avaliação de interações entre átomos dentro da proteína não contemplam todas as características necessárias para modelar corretamente [Lima *et al.*, 2007]. Assim, utilizam-se bases de dados para posições de átomos, ângulos de torção e centróides dos aminoácidos, que possuam valores já descritos como válidos na literatura e em estruturas já determinadas. Esta ideia foi sugerida já em [Branden & Tooze, 1999].

Algoritmos genéticos (AGs) podem ser empregados para a resolução deste problema pela modelagem *ab initio*, uma vez que ele pode ser visto como um algoritmo de otimização, no qual, dada uma sequência de aminoácidos, deve-se encontrar a melhor estrutura dentre várias possíveis. Os AGs podem ser particularmente eficientes nesta área, ao distribuir sua população inicial pelo espaço de busca e rapidamente convergir para um ponto ótimo, por suas características intrínsecas [Goldberg, 1989]. A determinação de estruturas de proteínas é um problema NP-completo [Pierce & Winfree, 2002], ou seja, existe uma explosão combinatória que faz com que a solução ótima seja dificilmente encontrada. Ainda assim, AGs têm sido aplicados com sucesso em diversos destes problemas, quando a otimização é um passo necessário, como, por exemplo, na seleção de atributos relevantes [Yang & Honavar, 1998], logística [Taniguchi *et al.*, 1999], sistemas elétricos [Fukuyama *et al.*, 1996], entre outros. No entanto, para este problema, em sua forma padrão, os AGs já foram aplicados por vários pesquisadores, entre eles os mais citados [Pedersen & Moult, 1996] e [Schulze-Kremer, 1993], sem que resultados satisfatórios tenham sido alcançados, devido à existência de muitos ótimos locais no espaço de busca e, principalmente, à dificuldade de escolha da função de energia a ser minimizada. Como exemplo deste último problema, em [Schulze-Kremer, 1993] foram encontrados resultados com energia potencial menor que a do estado nativo da proteína estudada (Met-Encefalina), porém o estado nativo não foi encontrado. Por este fato, esses trabalhos servem como guia para que alterações sejam implementadas, em busca de melhores resultados. Neste trabalho, o foco está no primeiro problema, ou seja, a existência de muitos ótimos locais no espaço de busca na predição da estrutura de proteínas, o que acarreta na convergência prematura das soluções do AG.

Esta convergência prematura ao longo das gerações ocorre devido à perda de diversidade da população, ou seja, conforme o algoritmo vai sendo executado, os indivíduos se tornam mais e mais parecidos, em torno de um ponto ótimo, mas que na maioria das vezes é um ótimo local, sem que o ótimo global seja encontrado. Desta forma, o objetivo principal é investigar técnicas de manutenção e aumento de diversidade da população em AGs. Salienta-se que o objetivo, neste ponto, não é apresentar um algoritmo que resolva, por si, o problema de determinação de

estruturas de proteínas, pois diversos outros fatores devem ser considerados, entre eles e principalmente a escolha de uma função de energia para avaliação das estruturas as mais fiéis à realidade possível, uma vez que é desta função que se analisam todas as características da estrutura formada. Diversos autores estudam especificamente este problema, como por exemplo [Cornell *et al.*, 1995], [MacKerel Jr. *et al.*, 1998], [Jorgensen & Tirado-Rives, 1988]; este trabalho segue o que a literatura relacionada apresenta como mais utilizado.

A questão principal investigada neste trabalho é se técnicas de controle de diversidade de população em AGs são benéficas para este problema, tomando como base trabalhos que utilizem abordagens semelhantes para a determinação de estruturas, sem se preocupar com a diversidade populacional. Estas técnicas podem ser empregadas, ainda, em outros problemas de otimização, como os Problemas de Otimização Dinâmica (*Dynamic Optimization Problems,* DOP), em que AGs são empregados. Por fim, este trabalho não possui como objetivo comparar seus resultados com outros métodos de otimização nem de determinação de proteínas, por se tratarem de metodologias diferentes e que visualizam o problema de uma forma diferente da abordada aqui.

A abordagem escolhida para este trabalho é a de *Ab Initio* com informações de base de dados. Para isto, foi montada uma base de dados de ângulos de torção ϕ (phi) e ψ (psi) de cada um dos 20 aminoácidos existentes, a partir do projeto CADB [Sheik *et al.*, 2003], disponível online no endereço http://cluster.physics.iisc.ernet.in/cadb/, e outra base de dados para a cadeia lateral dos aminoácidos, que possui ângulos $\chi_{1\text{-}5}$ (ou até mesmo nenhum na cadeia lateral), dependendo do aminoácido em questão. Esta base foi obtida a partir do trabalho de Tuffery [Tuffery, 1991], disponível online, no endereço http://bioserv.rpbs.jussieu.fr/doc/Rotamers.html. Estes dados são colocados como entrada para a formação de indivíduos em um AG [Mitchell, 1996], que foi construído em sua forma padrão, a partir de um sistema explicado em [Linden, 2006], e depois alterado em algumas características para incluir técnicas de controle de diversidade no conjunto de soluções.

As técnicas de aumento e manutenção da diversidade da população investigadas foram Hipermutação [Cobb & Grefenstette, 1993] e Imigrantes

Aleatórios [Cobb & Grefenstette, 1993], [Vavak & Fogarty, 1996], dentro do AG em sua forma padrão. Além disso, é proposto um novo algoritmo baseado no AG com Auto-Organização [Tinós & Yang, 2007], e os resultados são comparados.

Para avaliação das estruturas geradas, foi utilizado o pacote de modelagem molecular Tinker [Ponder *et al.*, 1998], que possui algoritmos para transformação dos ângulos de torção em coordenadas cartesianas e cálculo da energia total (que serve como o *fitness* dos indivíduos neste programa) para diversos campos de força, entre eles o CHARMM27 [MacKerel Jr. *et al.*, 1998], que foi empregado neste trabalho.

O trabalho está dividido da seguinte forma: no Capítulo 2, o conhecimento básico sobre proteínas é apresentado; no Capítulo 3, a base computacional do trabalho está explicada; no Capítulo 4, explica-se a metodologia do trabalho; o Capítulo 5 traz os resultados; e o Capítulo 6 apresenta as conclusões e trabalhos futuros relacionados a este projeto.

2. A PROTEÍNA

Proteínas são as macromoléculas mais abundantes nos seres vivos, por serem os instrumentos moleculares pelos quais a informação genética é expressa, e por assumirem uma enorme diversidade de funções biológicas; sendo assim pode-se dizer que são as moléculas mais importantes para os seres vivos. São formadas por aminoácidos, e podem assumir uma quantidade infindável de conformações tridimensionais, de acordo com a sequência de aminoácidos que a compõe e os ângulos que estes aminoácidos assumem [Lehninger *et al.*, 2005]. A seguir veremos mais a fundo as características de aminoácidos e proteínas.

2.1 Aminoácidos

Aminoácidos são formados pela junção de um grupo carboxila (COO^-) e um grupo amina (NH_3^+), ligados a um mesmo carbono, conhecido como carbono alfa (C_α). Eles se diferenciam pela quarta ligação que ocorre neste C_α (a terceira é um H), os chamados grupos R, que variam em estrutura, tamanho e carga elétrica. Por ser assimétrico, o C_α pode assumir pelo menos duas conformações estereoisométricas (posições cis e trans, por exemplo).

Existem 20 aminoácidos mais comuns na natureza, cujas estruturas estão exemplificadas na Figura 2.1. Existem várias outras estruturas reconhecidas como aminoácidos, porém estas não constituem proteínas.

Os aminoácidos podem ser classificados segundo seu grupo R, sendo divididos em cinco grupos:

- **Apolares alifáticos:** representados pela alanina, valina, leucina e isoleucina, são hidrofóbicos, estabilizando a estrutura protéica por meio de interações hidrofóbicas;
- **Aromáticos:** apresentam cadeias laterais aromáticas, ou seja, com seis carbonos em anel, e participam de interações hidrofóbicas. Pertencem a este grupo a fenilalanina, a tirosina e o triptofano.

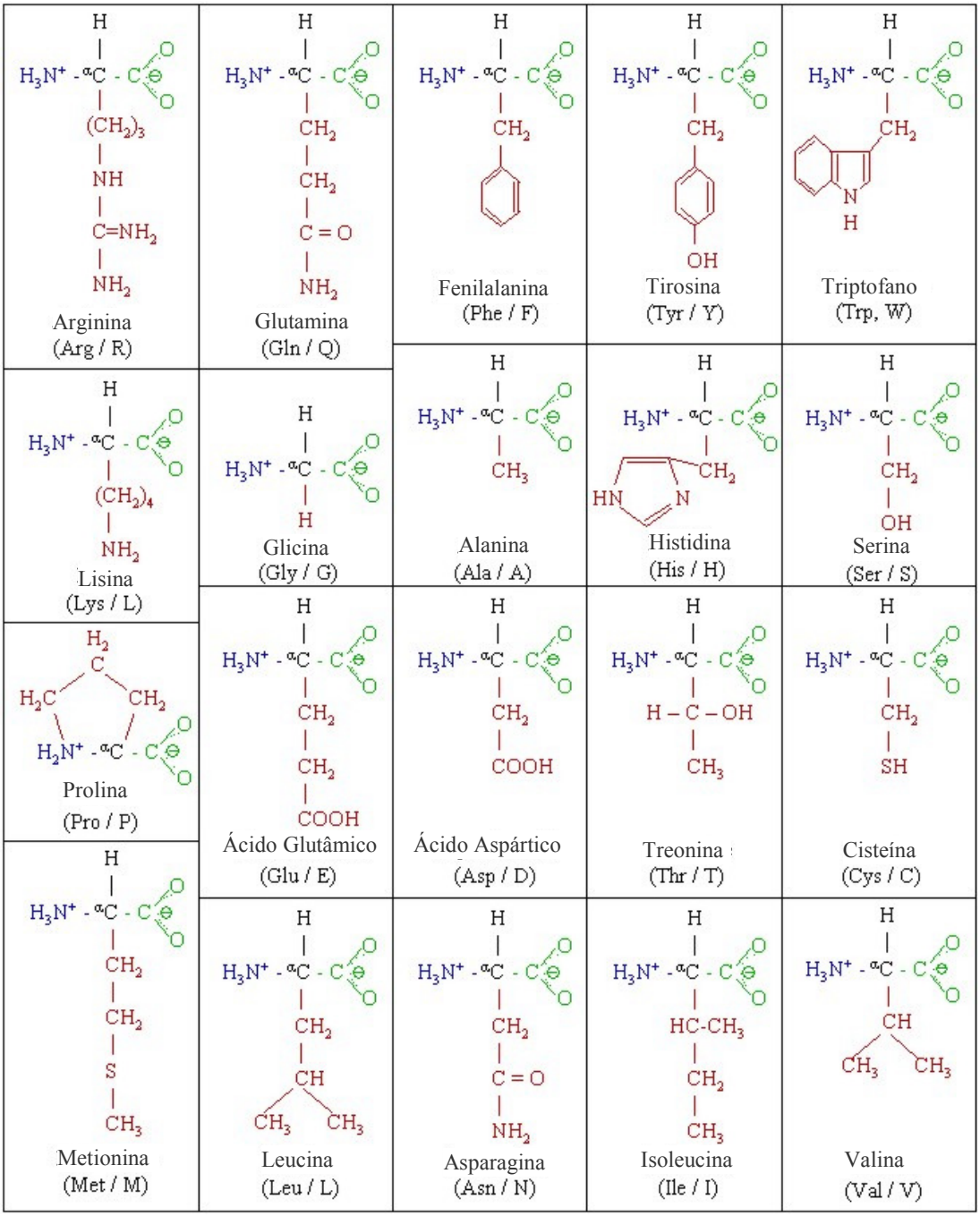

Figura 2.1 – Estruturas planares dos 20 aminoácidos constituintes de proteínas (obtido de http://upload.wikimedia.org/wikipedia/commons/thumb/c/c5/Amino_acids_2.png/600px-Amino_acids_2.png).

- **Polares sem carga:** possuem grupos mais solúveis em água, por conterem grupos funcionais que formam pontes de hidrogênio. Grupo formado pela serina, cisteína, asparagina, glutamina e treonina.
- **Positivamente carregados:** representam o grupo mais hidrofílico carregado positivamente. Representam este grupo a lisina, a arginina e a histidina.

- **Negativamente carregados:** também fortemente hidrofílico, é composto pelo aspartato e pelo glutamato, sendo que cada um possui um segundo grupo carboxila.

O conhecimento a respeito de hidrofobia e hidrofilia é extremamente importante no desenvolvimento de estruturas, pois a partir destas regiões hidrofílicas e hidrofóbicas é possível estudar formas de interação com moléculas-alvo.

Os aminoácidos variam também em relação a curvas características de titulação, o que os torna reconhecíveis e úteis para finalidades específicas [Lehninger *et al.*, 2005].

2.2 Peptídeos

Aminoácidos podem ser combinados entre si para formar estruturas maiores, conhecidas como peptídios, por meio do processo de polimerização. Esta ligação pode ser considerada uma reação de condensação, comum em seres vivos [Petsko & Ringe, 2003].

Quando poucos aminoácidos se juntam, chamamos a estrutura final de oligopeptídio. Conforme a cadeia vai crescendo, forma-se um polipeptídio, que pode ser considerado sinônimo de proteína; no entanto, se utiliza mais proteína para estruturas maiores (a partir de 25 aminoácidos), enquanto estruturas menores são chamadas de polipeptídios [Lehninger *et al.*, 2005].

Três ligações separam um carbono-alfa do carbono-alfa do próximo aminoácido. As ligações N-C_α e C_α-C podem "girar", assumindo ângulos entre -180º e 180º. Estes ângulos de ligação são os ângulos ϕ e ψ (demonstrados na Figura 2.2), que são essenciais para este trabalho, por serem usados pelo AG para determinar a estrutura tridimensional da cadeia principal das proteínas. Existe ainda o ângulo ω (Omega), que define a ligação C-N', que geralmente assume o ângulo 0º (posição cis) ou 180º (posição trans).

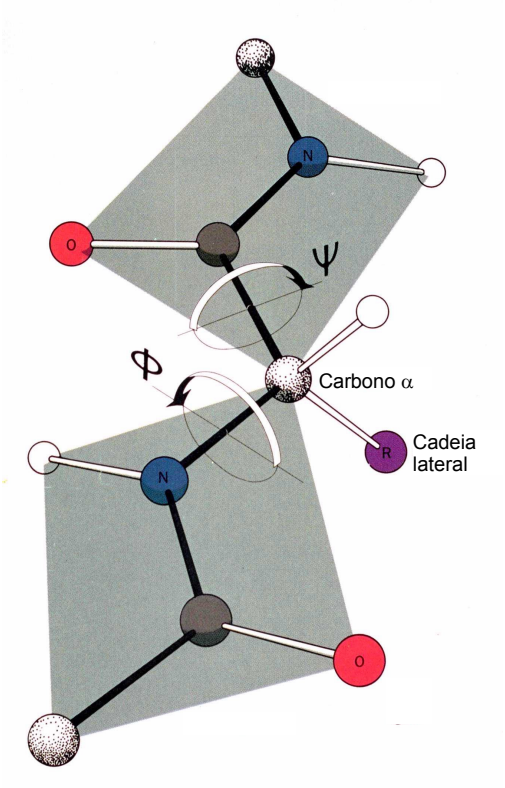

Figura 2.2 – Ilustração gráfica dos ângulos ϕ e ψ de uma ligação peptídica (imagem obtida em http://courses.cm.utexas.edu/jrobertus/ch339k/overheads-1/ch6_phi-psi.jpg).

De forma semelhante, o grupo R, ou cadeia lateral, apresenta seus próprios ângulos de ligação, conhecidos como ângulos χ [Lesk & Lrdk, 2001]. Dependendo do aminoácido empregado, estes ângulos podem variar entre nenhum e cinco, dependendo do tamanho da cadeia lateral. Os aminoácidos glicina e alanina, por exemplo, não possuem ângulos de rotação na cadeia lateral [Falcão *et al.*, 2002]. A Figura 2.3 apresenta um exemplo de ângulos χ para o aminoácido lisina.

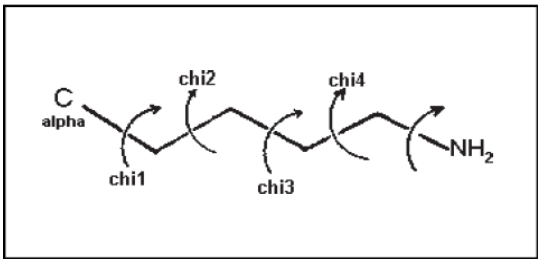

Figura 2.3 – Ângulos χ para o aminoácido lisina [Falcão *et al.*, 2002].

2.3 Níveis estruturais

Com o objetivo de aprofundar o estudo e facilitar a compreensão das estruturas protéicas em seus diferentes níveis de formação, pode-se definir as proteínas em quatro níveis estruturais:

Estrutura primária

Apresenta a sequência de aminoácidos que compõe a proteína, sem informações de volume ou ligações químicas; para definição da sequência são empregados métodos como cromatografia e eletroforese [Lehninger *et al.*, 2005];

Estrutura secundária

Define características locais da proteína, como padrões de dobramento da cadeia principal, e pode ser definida por espectroscopia, por exemplo. As estruturas principais conhecidas são a α-hélice, a folha β e a volta β [Petsko & Ringe, 2003], de acordo com o formato que elas representam. O mapa de Ramachandran [Ramachandran & Sasisekharan, 1968] demonstra as combinações de ângulos ϕ e ψ válidas e que tipos de estruturas secundárias estes podem formar, como pode ser visto na Figura 2.4. A seguir são explicitadas as estruturas secundárias mais comuns:

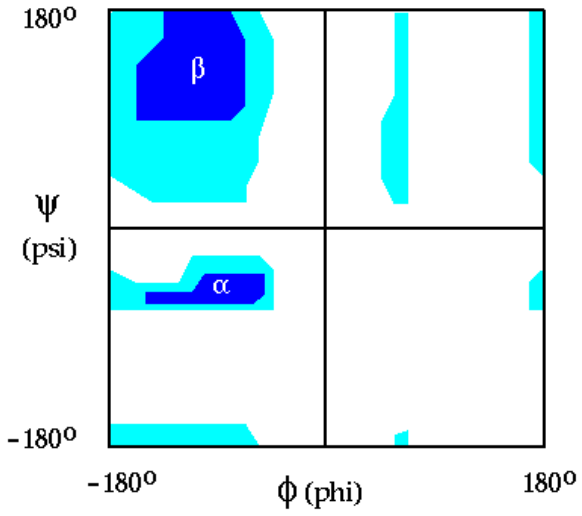

Figura 2.4 – Mapa de Ramachandran. As áreas em azul escuro identificam as combinações de ângulos ϕ e ψ que formam α-hélices e folhas β (imagem obtida em http://www.cgl.ucsf.edu/home/glasfeld/tutorial/AAA/plot.gif).

- **α-hélice:** é o arranjo estrutural mais simples que os aminoácidos assumem, dando voltas em torno de um eixo imaginário distribuído longitudinalmente no meio da hélice (por conta da repetição seguida de ângulos φ em torno de –60º e ψ em torno de –50º), e com os grupos R se apresentando na região externa da hélice [Lehninger *et al.*, 2005]. A Figura 2.5 ilustra graficamente uma α-hélice.

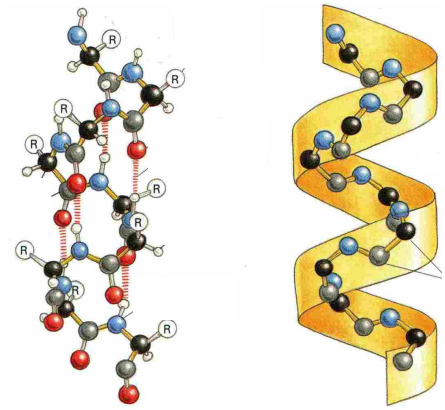

Figura 2.5 – Exemplo de estrutura secundária α-hélice [Alberts *et al.*, 2003].

- **Folha β:** na folha β, ao invés de os aminoácidos se distribuírem em torno de um eixo, eles formam um "ziguezague", com ligações de hidrogênio entre segmentos adjacentes da cadeia polipeptídica. Podem se formar na forma paralela (no mesmo sentido da orientação da proteína, grupo amina->grupo carboxila) ou antiparalela (sentido contrário da orientação da proteína) [Lehninger *et al.*, 2005]. A Figura 2.6 ilustra uma folha β.

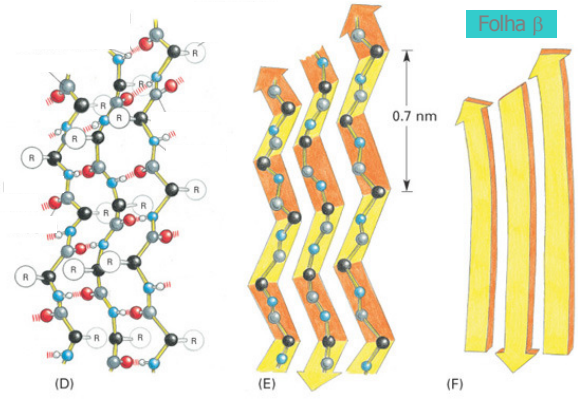

Figura 2.6 – Esquematização gráfica de uma folha β [Alberts *et al.*, 2003].

- **Volta β:** em proteínas globulares, ocorrem muitas voltas, por serem estruturas compactas. As voltas são elementos de ligação entre α-hélices e folhas β. A estrutura costuma ser de quatro aminoácidos, dando uma volta de 180º na estrutura. Em geral são encontradas nas áreas mais externas da proteína, em contato com o ambiente aquoso, e seus aminoácidos centrais não formam pontes de hidrogênio. A Figura 2.7 esquematiza uma volta β, o tipo mais comum.

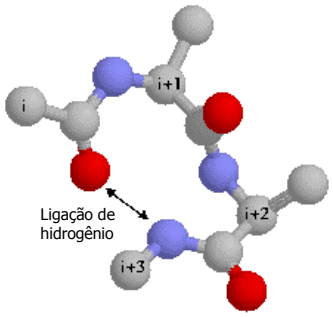

Figura 2.7 – Exemplo gráfico de uma volta β (átomos *i* são os carbonos-alfa) (obtido em http://www.nku.edu/~russellk/tutorial/peptide/b-turn2.jpg).

Estrutura terciária

Se passarmos a observar a proteína como um todo, veremos sua estrutura terciária, que se compõe das estruturas secundárias explicadas anteriormente. A esta altura, é possível enxergar interações entre aminoácidos que não eram possíveis em fases anteriores, como interações entre estruturas secundárias. Sua conformação final é alcançada após uma série de reações, que auxiliam, por exemplo, na formação de um centro hidrofóbico e uma superfície hidrofílica, com a menor energia livre disponível quando em condições favoráveis [Copeland, 1994], segundo a hipótese termodinâmica de Anfinsen [Anfinsen, 1973]. A Figura 2.8 exemplifica a estrutura terciária da proteína Met-Encefalina (código PDB 1PLW), um neurotransmissor formado de 5 aminoácidos, enquanto a Figura 2.9 ilustra a estrutura terciária da proteína Crambina (código PDB 1CRN). A imagem foi criada com o auxílio do software PyMOL [Delano, 2002], um pacote de modelagem molecular de código aberto, porém pago para versão final.

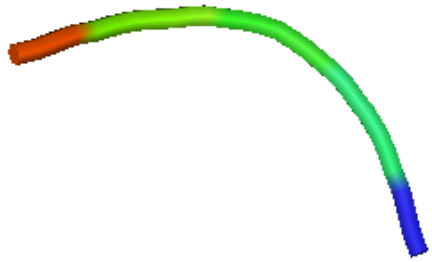

Figura 2.8 – Estrutura terciária da proteína Met-Encefalina (código PDB 1PLW). Como se nota, ela não possui estruturas secundárias, por se tratar de uma proteína pequena (oligopeptídeo); assim esta é a forma possível de visualização macro-estrutural.

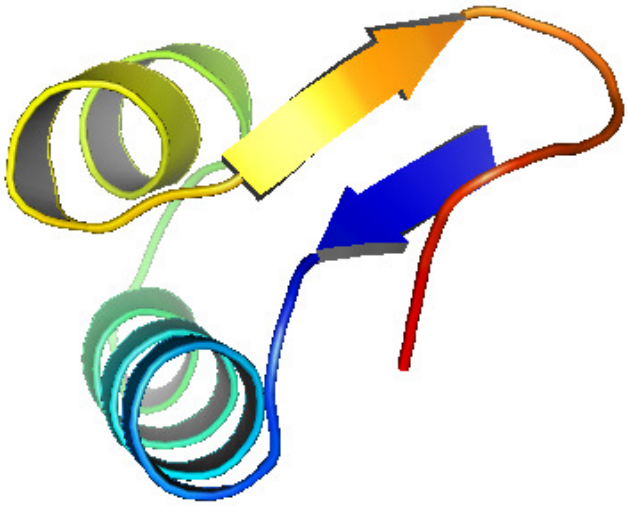

Figura 2.9 – Estrutura terciária da proteína Crambina (código PDB 1CRN). É possível visualizar duas alfas-hélice e duas folhas-beta compondo sua estrutura terciária (apresentada na forma macro-estrutural de visualização).

É possível classificar as estruturas terciárias em dois grupos principais: proteínas fibrosas, que são geralmente constituídas de apenas um tipo de estrutura secundária e dão formato, suporte e proteção externa a vertebrados, por exemplo; e proteínas globulares, de estrutura mais complexa, contendo mais de um tipo de

estrutura secundária e um volume mais compacto, com possibilidades estruturais mais amplas, que garantem uma série de capacidades biológicas, como regulação, transporte, catalisadores, digestão, entre outras [Lehninger *et al.*, 2005].

Estrutura quaternária

Muitas proteínas possuem uma estrutura final que consiste da junção de várias subunidades, ou estruturas terciárias, cada uma com uma subfunção, que pode ser afetada por pequenas estruturas que interajam com estas subunidades. A Figura 2.10 esquematiza as quatro estruturas existentes para a proteína Hemoglobina (código PDB 2HHB), desde a sequência protéica, passando pela α-hélice até a estrutura quaternária [Lehninger *et al.*, 2005].

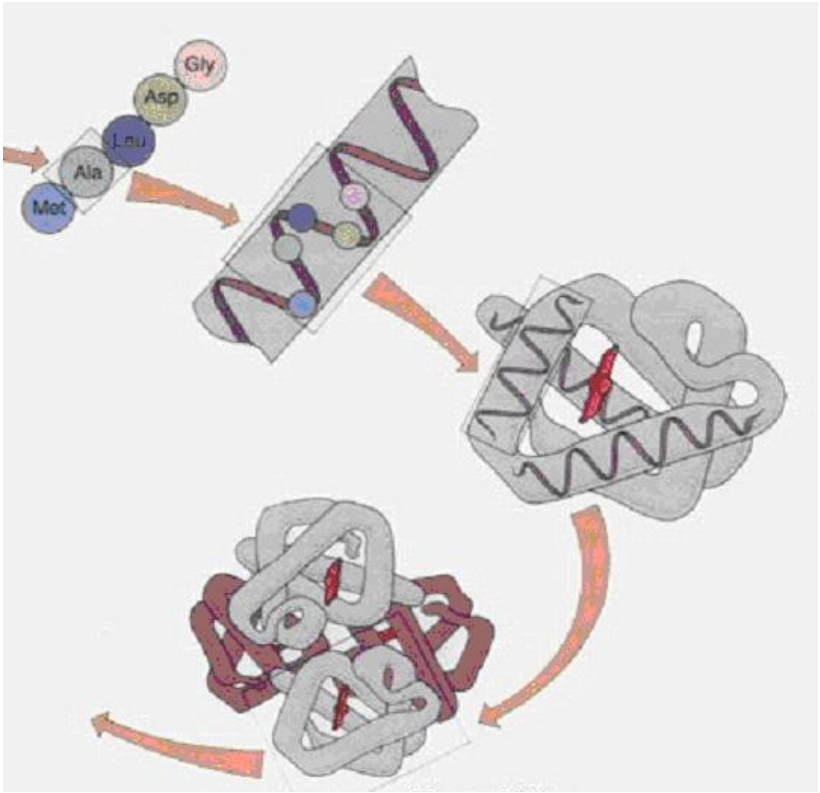

Figura 2.10 – As quatro estruturas existentes para a proteína Hemoglobina (imagem obtida em http://hemoglobinas.files.wordpress.com/2008/04/hemoglobina.jpg).

2.5 Domínios Protéicos

Proteínas grandes, de mais de 100 resíduos geralmente se dobram em subunidades globulares, como explicado anteriormente. Estas unidades são conhecidas como domínios, e estes podem apresentar alto nível de interações, de forma que em alguns casos é difícil distinguir os domínios. Cada domínio é

responsável por uma função, entre ligar-se a pequenas moléculas ou interação com outras proteínas [Lehninger *et al.*, 2005].

Para manter a estabilidade de cada domínio, centros hidrofóbicos são importantes, por minimizarem as interações com a água e aumentarem as interações Van der Waals entre domínios hidrofóbicos.

Os domínios são formados por combinações de estruturas secundárias e *motifs* (que são, por si sós, combinações de estruturas secundárias estáveis), que podem ser combinados entre si para formar *motifs* mais complexos [Lehninger *et al.*, 2005]. Estas estruturas são visualmente interessantes para se efetuar comparações visuais entre proteínas em seu estado nativo e as determinadas por meio de simulação. Os domínios mais comuns são:

- **Domínios α:** geralmente constituídos de um conjunto de α-hélices ligadas por aminoácidos sem estrutura definida [Petsko & Ringe, 2003]. A parte interna das hélices, que se dispõem pareadas, forma um centro hidrofóbico, enquanto a parte externa apresenta um padrão hidrofílico. Uma estrutura conhecida como pacote de quatro hélices, que possui um ângulo de 20º entre cada hélice, apresenta como funções transporte de oxigênio e ligação com ácidos nucléicos [Branden & Tooze, 1999].
- **Domínios β:** estes domínios apresentam conjuntos de folhas β, voltas curtas e irregulares [Branden & Tooze, 1999]. Também interagem entre si, fazendo formações antiparalelas, que originam vários motivos, como os sanduíches β, barris β e a chave-grega, com funções específicas [Petsko & Ringe, 2003].
- **Domínios α/β:** existem em maior quantidade nas proteínas, participam em reações de catálise e ligação [Petsko & Ringe, 2003]. Existem dois motivos mais comuns, o barril TIM (devido à proteína onde foi descoberto) e a dobra de ligação nucleotídica [Branden & Tooze, 1999].

2.6 Considerações Finais

Neste capítulo foi apresentado o essencial sobre proteínas, desde sua formação a partir dos aminoácidos até suas macroestruturas, com algumas formas

de classificação abordando vários aspectos. É interessante notar que o assunto possui ampla literatura, e pode ser estudado sobre diferentes aspectos.

No próximo capítulo serão abordados tópicos sobre biologia evolutiva e computação evolutiva.

3. EVOLUÇÃO E COMPUTAÇÃO EVOLUTIVA

Neste capítulo serão apresentados brevemente os conceitos biológicos sobre evolução e sua ligação com a técnica dos AGs, que se aproveita desses conceitos para a formulação de novas abordagens computacionais para a resolução de problemas de busca e otimização. Serão abordados também os conceitos de AGs necessários para a compreensão da metodologia deste trabalho.

Três técnicas para aumentar ou manter a diversidade das populações dos AGs utilizadas neste trabalho são apresentadas: Hipermutação, Imigrantes Aleatórios e Imigrantes Aleatórios com Auto-Organização Simplificado, proposta neste trabalho.

3.1 Darwin e a Seleção Natural

Charles Robert Darwin (1809-1882), membro da Igreja Anglicana, começou a desenvolver a teoria da Seleção Natural a partir de suas viagens pelo navio *Beagle* (ocorrida entre 27 de dezembro de 1831 e 2 de outubro de 1836), mas não durante a viagem, pois apresentou grande dificuldade em concatenar este pensamento com a imutabilidade das espécies propagada (indiretamente) pela Igreja [Futuyma *et al.*, 2002]. Somente em 1837, quando o ornitólogo John Gould lhe mostrou diferenças significativas entre espécimes de tordos-dos-remédios observados em diferentes ilhas do arquipélago de Galápagos (de forma que seria possível até classificá-los em espécies diferentes [Sulloway, 1982]), que Darwin começou a agrupar evidências sobre a evolução. Auxiliou-lhe o ensaio de Malthus [Malthus, 1809], que o permitiu inferir que variações favoráveis tendem a ser preservadas, no caso de facilidade de adaptação à alimentação, enquanto as desfavoráveis tendem a ser destruídas, já que o crescimento descontrolado da população levaria a uma situação em que não haveria alimentos suficientes para todos.

Darwin discorreu, em seu livro *A Origem das Espécies* [Darwin, 2004], sobre duas teses: a primeira, de que todos os organismos vivos descendem de um ancestral comum e sofreram modificações (utilizando para comprovação desta tese registros fósseis, anatomia e embriologia comparadas, entre outros); e que o

principal agente de modificação é a seleção natural sobre a variação natural [Futuyma *et al.*, 2002]. No entanto, faltava-lhe o conhecimento sobre genética para comprovar esta afirmação. Por este motivo, a seleção natural foi desacreditada até os anos vinte do século XX, quando, a partir de descobertas feitas por Mendel [Mendel, 1865], pesquisadores como Theodosius Dobzhansky [Dobzhansky, 1982] conseguiram ligar as duas áreas e explicar a seleção natural, criando conceitos que hoje são amplamente reconhecidos, como por exemplo: as populações contêm variação genética que surge após mutações ocorridas ao acaso e recombinação; por conta disso, as populações alteram suas frequências gênicas e evoluem, de modo que mudanças fenotípicas são graduais, ou seja, não se percebem grandes mudanças de uma vez na conformação de seres vivos; a diversificação surge após o isolamento reprodutivo, e este é o processo definitivo para a especiação.

Esse conjunto de evidências e raciocínios influenciou diversas áreas de pesquisa, que passaram a ver na natureza possibilidades de adaptação de muitos conceitos para aplicação na solução de problemas; particularmente a computação, que viu nos meios evolutivos uma inspiração para a produção dos Algoritmos Genéticos, uma das técnicas de Computação Evolutiva. AGs foram inspirados em fatores genéticos e na seleção natural e são utilizados para evoluir conjuntos de soluções e atingir resultados melhores em problemas de otimização. A seguir os AGs são explicados.

3.2 Algoritmos Genéticos

Os primeiros passos da Computação Evolutiva começam com a criação do ramo da Inteligência Artificial, nos anos 40, com pesquisas sobre processos de raciocínio e aprendizado. Os métodos de seleção e mutação foram aplicados pela primeira vez por Box [Box, 1957], para alterar algumas variáveis em um problema de controle. A codificação em genes, binários, inteiros e reais, foi trabalhada primeiro por Bledsoe [Bledsoe, 1961] e Bremmermann [Bremmermann, 1962]. Quem primeiro reuniu todos estes conceitos e recebeu o crédito pela criação dos AGs, foi John Holland [Holland, 1975]. Este propôs um modelo computacional baseado na evolução das espécies, que poderia oferecer boas soluções para problemas difíceis de resolver pelas técnicas existentes na época. Um detalhe interessante é que Holland não tinha

a intenção inicial de criar novos algoritmos de otimização, e sim uma metáfora para os processos evolutivos, com o objetivo de estudar a adaptação e a evolução no mundo real, usando computadores. Sua proposta inicial de codificação das soluções foi binária [Linden, 2006].

O AG, conforme vislumbrado por Holland, consiste de uma população (conjunto de indivíduos, no qual cada indivíduo representa uma solução em potencial). Os indivíduos são compostos por um cromossomo que possui valores que podem representar a solução procurada, e estes cromossomos estão sujeitos a recombinação gênica e mutação. Um processo de seleção inspirado na seleção natural se encarrega de eliminar os indivíduos pior adaptados ao problema e permitir a sobrevivência daqueles que se adaptam melhor às condições oferecidas.

Este comportamento é muito importante para aplicação em problemas de otimização, nos quais diversos parâmetros devem ser combinados para gerar a melhor solução. A inicialização aleatória dos indivíduos permite que, possivelmente, exista a melhor solução para o problema, com trechos desta solução espalhados em diversos indivíduos. Com a seleção e recombinação gênica, novos conjuntos de soluções, que combinam partes das soluções anteriores, se formam, levando os indivíduos para uma mesma região do espaço de soluções, que possivelmente representa onde melhores resultados podem ser alcançados.

Um AG possui a estrutura básica mostrada no Algoritmo 3.1.

Algoritmo 3.1 - Estrutura básica de um AG

```
procedimento ag( )
inicio
    geracao = 0
    inicialização (pop_velha)          // procedimento para inicialização
                                       // das variáveis e da população
    faça
        geracao = geracao + 1      // número de gerações
        pop_nova = selecao_individuos(pop_velha)//pop_velha = pop. da geração atual
        crossover(pop_nova)              //pop_nova = pop. da geração seguinte
        mutacao(pop_nova)
        estatistica(pop_nova)
        popvelha = popnova
    enquanto (geracao ≤ max_geracoes )
fim.
```

A seguir, os principais mecanismos existentes em um Algoritmo Genético são explicados.

3.2.1 Codificação do Cromossomo

O cromossomo é a representação de cada indivíduo do AG. É constituído pelo conjunto de valores que representam uma das possíveis soluções.

A codificação original sugerida por Holland [Holland, 1975] era composta por números binários, mais simples e rápidos de calcular, servindo melhor para o poder computacional da época. No entanto, este tipo de codificação pode gerar complicações adicionais, como a dificuldade de se alterar o valor de alguns números inteiros codificados por números binários em apenas uma unidade [Deb, 2001] (por exemplo, de 0111 para 1000, já que seriam necessárias 4 mutações para atingir um valor vizinho).

Assim, as codificações inteira e real podem ser utilizadas diretamente. Linden [Linden, 2006] afirma que a codificação deve se adaptar ao problema estudado, e não o contrário, para facilitar a implementação e os cálculos.

3.2.2 Inicialização dos Indivíduos

A inicialização dos indivíduos no AG é a forma de distribuir a população inicial pelo espaço de busca. Normalmente, efetua-se uma inicialização aleatória, ou seja, valores aleatórios obtidos a partir de uma distribuição uniforme são utilizados para gerar os indivíduos iniciais. Assim, enquanto não tiverem sido formados todos os indivíduos na primeira geração, sorteiam-se números aleatórios, que serão armazenados como alelos de cada indivíduo.

3.2.3 Seleção de indivíduos

De forma similar ao processo de seleção natural, costuma-se privilegiar os indivíduos que possuem maior adaptação ao meio ou problema sem, no entanto, proibir os menos adaptados de se reproduzirem também.

No AG padrão, vislumbrado por Holland, foi criado o método da roleta para seleção de indivíduos, no qual é atribuída a cada indivíduo uma probabilidade de ser escolhido para ser um dos pais, de forma proporcional ao *fitness* deste indivíduo.

Assim, não se garante a presença do melhor indivíduo na geração seguinte. Para contornar este problema, selecionam-se os melhores indivíduos de uma geração para passá-los automaticamente para a geração seguinte, processo conhecido como elitismo [Fogel, 1994].

No entanto, há problemas em que a discrepância de *fitness* entre os indivíduos pode fazer com que a roleta sorteie quase sempre o mesmo indivíduo, invalidando o processo. Por este motivo, outros métodos de seleção foram desenvolvidos.

Neste trabalho, é empregado o método de seleção por torneio [Goldberg, 1989]. Este método, em sua forma mais simples, seleciona aleatoriamente dois indivíduos na população e define uma probabilidade de escolha maior para o indivíduo de melhor *fitness* (para este trabalho definiram-se 75% de chances de o melhor indivíduo ser escolhido, contra 25% para o pior indivíduo).

O Algoritmo 3.2 descreve este processo de seleção por torneio para um indivíduo em um processo de minimização. O procedimento *gera_aleatorio* cria um número aleatório entre 0 e 1 com distribuição uniforme.

Algoritmo 3.2 – Seleção por Torneio

```
procedimento torneio( )
inicio
    individuo1 = gera_aleatorio * (tamanho_populacao-1)    // sorteia um indivíduo
    indivíduo2 = gera_aleatorio * (tamanho_populacao-1)    // sorteia outro indivíduo
     sorteio = gera_aleatorio                                  // valor do sorteado
    se sorteio < 0,75 entao                          // chance de o melhor ser escolhido
            se fitness(individuo1) < fitness(individuo2) então
                    pai=individuo1
            senao
                    pai=individuo2
            fim_se
    senao
            se fitness(individuo1) < fitness(individuo2) entao
                    pai=individuo2
            senao
                    pai=individuo1
            fim_se
    fim_se
fim.
```

Existem ainda outros métodos de seleção, como por exemplo a seleção aleatória (sem pesos para os melhores *fitnesses*) e a seleção por diversidade, quando

são selecionados os indivíduos mais diversos para formar a próxima geração. No entanto, julgou-se mais apropriado o método de seleção por torneio para o problema estudado, por se tratar de um problema de otimização e minimização, no qual o menor *fitness* é o elemento principal a ser alcançado, e assim menores *fitness*es devem ser preservados.

3.2.4 Crossover

O operador de *crossover* é uma simulação da fusão dos gametas dos seres vivos para a formação de um novo ser. Por este método, uma parte do cromossomo de um dos pais é replicada para o filho, enquanto o restante do cromossomo é obtido por replicação do outro pai, tendo sido selecionados ambos os pais anteriormente por torneio, por exemplo. Desta forma, é possível recombinar partes da solução de cada indivíduo em um indivíduo novo, com potencial para este ser ainda melhor que os indivíduos que o geraram. No AG padrão define-se uma probabilidade de que ocorra o *crossover*, sendo que se um sorteio aleatório definir que não deve ser efetuado *crossover*, os pais são automaticamente transferidos para a geração seguinte, sem recombinação. Para este trabalho, definiu-se uma probabilidade de 80% de ocorrência de *crossover*.

O operador de *crossover* mais simples e utilizado neste trabalho é o *crossover* de um ponto. Por este método, seleciona-se um ponto aleatório que dividirá qual parte do cromossomo será formada a partir de qual pai (o chamado ponto de corte). Deste ponto para a esquerda são retiradas as informações do pai 1 para o filho 1, enquanto deste ponto para a direita são replicadas as informações do pai 2 para o mesmo filho. A Figura 3.1 exemplifica este processo. Notar que o filho 2 é gerado de forma análoga, com as informações dos pais que não foram usadas para o filho 1.

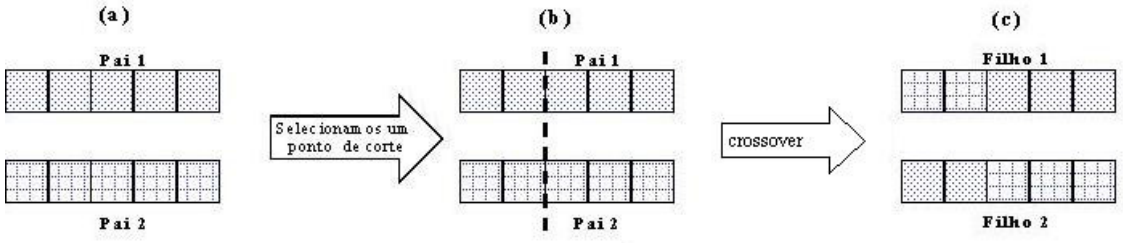

Figura 3.1 – Esquema gráfico do *crossover* de um ponto. Em (a), são selecionados os dois pais. Em (b), é selecionado aleatoriamente um ponto de corte, quando ocorre o *crossover*, e são gerados dois filhos que são a recombinação de uma parte de cada um dos pais (c). Adaptado de [Linden, 2006].

De forma análoga, o *crossover* de dois pontos define dois pontos aleatórios de corte, entre os quais será retirado material de um pai, e externamente a estes pontos é obtido material do outro pai.

Outros operadores mais complexos podem ser aplicados, como o *crossover* uniforme [Michalevicz & Fogel, 2002], no qual cada bit pode ser alterado, e não blocos de bits, aumentando as possibilidades de recombinação. No *crossover* aritmético [Michalevicz & Fogel, 2002], operações matemáticas relacionadas à recombinação dos valores dos pais são realizadas, gerando, por exemplo, a média destes valores como o valor do alelo do filho, entre outros métodos. Este método não é particularmente apropriado para este problema, uma vez que os ângulos gerados podem estar fora do conjunto de ângulos válidos para o aminoácido correspondente. Assim, a abordagem mais apropriada para este problema parece ser a de *crossover* simples.

3.2.5 Mutação

O operador de mutação é a forma como o AG gera novos valores para os alelos de seus indivíduos, tendo o efeito contrário em relação ao *crossover*: enquanto o *crossover* faz com que a população apresente cromossomos cada vez mais semelhantes, a mutação pode retirar esta igualdade e levar o conjunto de soluções a regiões do espaço de busca que não poderiam ser alcançadas pelo *crossover* [Linden, 2006]. A definição de uma boa taxa de mutação é essencial para o bom desenvolvimento do algoritmo: em caso de uma taxa baixa demais, os indivíduos ficarão todos praticamente iguais rapidamente, e a mutação não será suficiente para tirar os indivíduos das melhores soluções locais; por outro lado, uma taxa de mutação muito alta pode tornar o desempenho do algoritmo próximo a um passeio aleatório, pois as boas características que tenham sido acumuladas podem ser trocadas com uma frequência muito grande, perdendo a capacidade de evolução, melhor característica dos AGs.

Em uma codificação binária, uma mutação significa uma substituição de um valor 0 por um valor 1 ou vice-versa. Para codificações inteiras ou reais, as mutações devem ser substituições por valores dentro da faixa de valores válida para o

problema, restrição que deve ser verificada durante a mutação. A Figura 3.2 ilustra um exemplo de mutação numa codificação binária.

Figura 3.2 – Exemplo gráfico de uma mutação. O gene envolto pela circunferência pontilhada sofreu mutação, para o valor inverso (codificação binária). De forma análoga, podem ser feitas mutações para os outros genes do cromossomo (adaptado de [Linden, 2006]).

Para o problema descrito neste trabalho, o operador de mutação substitui o valor do índice da base de dados de ângulos por seu índice vizinho, com probabilidade igual para o vizinho de cima e o de baixo do índice atual. Assim, ao efetuar uma mutação, o que está sendo alterado é a linha da base de dados que possui os ângulos que serão usados para a estrutura protéica, e, por consequência, os ângulos de torção correspondentes àquela posição serão alterados, por cada linha da base de dados possuir um par ordenado de ângulos diferente.

Outros modelos de mutação podem ser a mutação gaussiana, na qual todos os alelos do cromossomo são modificados por um vetor de variáveis aleatórias com distribuição gaussiana; mutação não-uniforme, que efetua o chamado ajuste-fino em algum trecho do cromossomo, ao incrementar as taxas de mutação para que um processo de *hill climbing* seja efetuado; entre outras possibilidades [Lima, 2006].

3.3 Variações no Algoritmo Genético

Apesar de já existirem estes diversos mecanismos no AG, que permitem que esta tecnologia seja empregada com sucesso em diversas aplicações, existem problemas considerados "difíceis", seja por possuírem um espaço de busca de soluções amplo demais ou pela existência de diversas soluções que possam ser consideradas "boas", mas apenas um conjunto menor de soluções "ótimas"; e nestes, o desempenho do AG deixa a desejar, justamente por sua característica de rapidamente convergir para soluções ótimas, deixando de explorar outras possibilidades no espaço de busca. Entre estes problemas "difíceis", está justamente

o objeto de pesquisa deste trabalho, a predição de estruturas de proteínas [Tragante & Tinós, 2007].

Assim, pesquisadores vêm propondo variações no AG para evitar a convergência prematura da população para uma solução, que pode não ser a melhor; a seguir, veremos algumas das abordagens criadas, que são empregadas neste trabalho.

É importante ressaltar que muitos destes AGs foram inicialmente sugeridos para DOPs, ou seja, nos quais a função de *fitness* muda com o tempo, devido a características ambientais que variam conforme situações ocorrem, exigindo também a alteração das soluções, com pequenas variações. Estas técnicas buscam diminuir a convergência prematura com técnicas de aumento ou manutenção da diversidade das populações.

3.3.1 Hipermutação

Descrito pela primeira vez em [Cobb & Grefenstette, 1993], a Hipermutação é uma estratégia que aumenta as taxas de mutação periodicamente, de acordo com critérios pré-estabelecidos: por exemplo, ao se analisar o *fitness* médio da população e este tiver um valor próximo ao valor do melhor indivíduo da geração, o que é um indicativo de que todos os indivíduos possuem uma conformação semelhante, a taxa de mutação pode ser incrementada, para voltar a haver diversidade na população. Outra forma, que é a empregada neste trabalho, é incrementar as taxas de mutação desde o começo da execução, de forma intermitente: durante 5 gerações, a taxa de mutação está em seu valor normal, e nas 5 gerações seguintes, esta taxa é aumentada, e o processo continua durante toda a execução do algoritmo. Este processo pode ser eficiente no aumento de diversidade da população, uma vez que periodicamente inserem-se novas características nos indivíduos, com grande probabilidade de que estas alterações permaneçam nas gerações seguintes se estas forem benéficas para a melhoria do *fitness* dos indivíduos.

O Algoritmo 3.3 descreve este procedimento.

Algoritmo 3.3 – Hipermutação

```
procedimento geracao ( )
        inicio
        para(contador=0;contador<tamanho_populacao;contador+2)     //2 filhos
            pai_1 = torneio( )                      //seleção do pai 1
            pai_2 = torneio( )                      //seleção do pai 2
            filho = pai_1.crossover(pai_2)            //crossover
            se (flag == 0)
               taxa_mutacao = taxa_normal  //taxa_normal=(1/2m), onde 2m é o
                                           //tamanho do cromossomo
               contador++                  // conta gerações com hipermutação
               se (contador = 5)
                  flag = 1
                  contador = 0
                  pare
               fim_se
            senao
               taxa_mutacao = taxa_alta     //taxa_alta=80% de probabilidade
               contador++
               se (contador = 5)            //5 gerações com mutação alta
                  flag = 0
                  contador =0
               fim_se
            fim_se
            filho[0].mutacao(taxa_mutacao)      //envia o filho 1 para ser mutado
            filho[1].mutacao(taxa_mutacao)      //envia o filho 2 para ser mutado
        fim_para
        fim.
```

3.3.2 Imigrantes Aleatórios

Proposto inicialmente em [Cobb & Grefenstette, 1993], sugere a substituição de uma porcentagem do número de indivíduos da população por novos indivíduos criados aleatoriamente, a cada geração. Os indivíduos a serem substituídos podem ser escolhidos de forma aleatória, ou critérios como piores indivíduos podem ser empregados para substituição.

Neste trabalho, são gerados os novos indivíduos e estes são automaticamente inseridos na geração seguinte, sem realizar *crossover* na geração atual e sem serem avaliados. A avaliação destes indivíduos só será efetuada na geração seguinte, e os indivíduos atuais, assim, possuirão um número menor de *crossovers* para poderem "disseminar" suas características.

Além disso, testou-se a possibilidade de começar a inserir novos indivíduos apenas após passadas algumas gerações, de modo que o procedimento de *crossover* permitisse a convergência mais rápida da população, e só então novas características sejam inseridas. Este procedimento é particularmente proveitoso para manter a diversidade da população, uma vez que em todas as gerações há a inserção de novos indivíduos, que possivelmente carregam em si ângulos que nunca foram usados antes na execução do algoritmo, e combinados a indivíduos já existentes e de bom *fitness* podem levar a combinações ainda melhores de indivíduos, que podem atingir a solução ideal do problema. Além disso, não permite que todos os indivíduos fiquem muito parecidos com *crossovers* sucessivos, e assim mais do espaço de busca seja explorado. A seguir, o Algoritmo 3.4 descreve o AG com Imigrantes Aleatórios.

Algoritmo 3.4 – Imigrantes Aleatórios

```
procedimento geracao ( )
   inicio
      total_imigrantes = 0
      enquanto (total_imigrantes < taxa_subst) //taxa_subst=numero de novos
         filho = novo_individuo( )              //cria novo individuo aleatório
         nova_pop.adicionar(filho)              //inclui novo à nova população
         total_imigrantes++
      fim_enquanto
      para(contador=taxa_subst;contador<tamanho_populacao;contador+2)//2 filhos
           pai_1 = torneio( )                   //seleção do pai 1
           pai_2 = torneio( )                   //seleção do pai 2
           filho=pai_1.crossover(pai_2)          //crossover
           filho[0].mutacao(taxa_mutacao)       //envia o filho 1 para ser mutado
           filho[1].mutacao(taxa_mutacao)       //envia o filho 2 para ser mutado
        fim_para
    fim.
```

3.3.3 Imigrantes Aleatórios com Auto-Organização Simplificado

A definição da taxa de substituição de indivíduos pelo AG com Imigrantes Aleatórios é um problema importante para esta abordagem. Uma taxa de substituição pequena demais pode não atingir o objetivo desejado de manter a diversidade de soluções da população; por outro lado, uma taxa de substituição alta demais impede que características boas sejam propagadas ao longo das gerações, pois estas características são rapidamente substituídas nas gerações subsequentes.

Assim, pensou-se na possibilidade de tornar a taxa de inserção de novos indivíduos dinâmica, de maneira que o algoritmo analise as condições durante a

execução e decida se o número de indivíduos aleatórios a serem inseridos deve aumentar ou diminuir em relação à geração anterior. Em [Tinós & Yang, 2007], propôs-se um algoritmo com estas características, chamado *Self-Organizing Random Immigrants Genetic Algorithm* (SORIGA - "Algoritmo Genético com Imigrantes Aleatórios Auto-Organizados"). No trabalho supra citado, é criada uma sub-população de imigrantes aleatórios, de forma que as características recém-inseridas no contexto não sejam perdidas logo nas gerações seguintes. Esta subpopulação possui tamanho variável. Ao longo das gerações, são feitas substituições entre indivíduos das populações, assim que há uma troca de características entre as duas subpopulações, que evoluem em conjunto e melhores resultados podem ser alcançados.

No entanto, este método aumenta a complexidade do algoritmo, pois o número de avaliações de indivíduos aumenta, já que agora são duas populações que estão sendo trabalhadas; além disso, há a dificuldade em lidar com uma população cujo tamanho não é fixo. Assim, pensou-se em uma estratégia para simplificar este problema, eliminando a subpopulação. Esta estratégia é proposta neste trabalho, e a ela foi dado o nome de "Imigrantes Aleatórios com Auto-Organização Simplificado".

Por este método, o algoritmo procura o indivíduo de pior *fitness* da geração. Se este indivíduo for um dos imigrantes aleatórios que foram criados na mesma geração, o número de novos imigrantes na geração seguinte será incrementado em dois, o que significa um processo de *crossover* a menos; por outro lado, se o pior indivíduo estiver fora do rol de imigrantes criados naquela geração, o processo é reiniciado, ou seja, na geração seguinte haverá apenas 2 novos imigrantes, sendo todos os outros indivíduos gerados pelo processo padrão de *crossover*. Caso o número de imigrantes aleatórios a serem criados em uma dada geração atinja 70%, o processo também é reiniciado, e apenas 2 novos indivíduos serão gerados aleatoriamente, enquanto os outros serão criados por meio de *crossover* entre os indivíduos existentes.

O Algoritmo 3.5 mostra as alterações a serem feitas em relação ao Algoritmo 3.4 para que a estratégia de auto-organização seja utilizada.

Algoritmo 3.5 – AG com Imigrantes Aleatórios Auto-Organizados Simplif.

```
procedimento geracao ( )
   inicio
      pior=achar_pior_individuo(pop_velha)        //definir qual o pior indivíduo
      se (limite_inferior<indice(pior)<limite_superior)
         qtde_aleatorio= qtde_aleatorio + 2     //imigrantes na geração seguinte
         se (qtde_aleatorio >= 0.7*tam_pop)  //se passar limite de aleatórios
            qtde_aleatorio = 2                 //reinicia imigrantes na geração seguinte
         fim_se
      senao
         qtde_aleatorio = 2                    //reinicia imigrantes na geração seguinte
      fim_se
      total_imigrantes = 0
      enquanto (total_imigrantes < taxa_subst)    //taxa_subst=porcentagem
                                                  //de novos indivíduos
          filho=novo_individuo( )                    //cria novo individuo aleatório
          nova_pop.adicionar(filho)                  //inclui novo à nova população
          total_imigrantes++
      fim_enquanto
      para(contador=taxa_subst;contador<tamanho_populacao;contador+2)//2 filhos
            pai_1 = torneio( )                  //seleção do pai 1
            pai_2 = torneio( )                  //seleção do pai 2
            filho=pai_1.crossover(pai_2)        //crossover
            filho[0].mutacao(taxa_mutacao)      //envia o filho 1 para ser mutado
            filho[1].mutacao(taxa_mutacao)      //envia o filho 2 para ser mutado
         fim_para
   fim.
```

Este procedimento faz o algoritmo ser executado mais rapidamente, se a média de indivíduos gerados por este método for menor que a porcentagem definida previamente, uma vez que a geração de novos indivíduos é uma tarefa computacionalmente custosa, pois há a necessidade de se abrir a base de dados de cada aminoácido para buscar os ângulos sorteados, salvar estes valores em vetores auxiliares, e o valor do índice no cromossomo deste novo indivíduo, tarefas computacionalmente mais custosas que o procedimento de *crossover*.

No AG com Imigrantes Aleatórios Auto-Organizados Simplificado, assim como no SORIGA [Tinós & Yang, 2007], o número de imigrantes aleatórios é controlado por auto-organização. No início da execução, quando em geral todos os indivíduos da população tem valores de *fitness* similares e altos, a probabilidade de o pior indivíduo ser um dos imigrantes inseridos na geração anterior é baixa, fazendo com que o número de imigrantes aleatórios criado na geração seguinte seja baixo.

No entanto, ao decorrer das gerações, os indivíduos criados por *crossover* e mutação passam a ser cada vez mais parecidos (baixa diversidade), e com *fitness*, em média, melhor que os correspondentes aos imigrantes aleatórios. Desta forma, o número de indivíduos aleatórios criados é aumentado [Tinós & Yang, 2007]. Assim, quanto menor a diversidade e maior a diferença de *fitness* entre os indivíduos da população e os aleatórios, maior é a probabilidade de o número de indivíduos substituídos aumentar. O inverso ocorre quando a diversidade é alta, gerando assim um controle auto-organizado do número de indivíduos aleatórios.

3.4 Considerações Finais

Os AGs têm sido empregados, com sucesso, em diversas tarefas computacionais, atingindo até mesmo resultados nunca atingidos anteriormente, como descrito há quase 20 anos na literatura [Davis, 1991]. No entanto, em problemas mais complexos, há uma grande possibilidade de o AG ficar "preso" em uma única região do espaço de soluções, devido à sua rápida convergência, que é uma grande vantagem para alguns problemas, mas uma grande desvantagem quando há um grande número de soluções ótimas, mas poucas soluções ótimas globais.

Por este fato, são necessárias alterações no AG original para contemplar a maior inserção de diversidade na população, de modo que mais regiões do espaço de busca possam ser varridas, e melhores resultados sejam atingidos.

Assim, este capítulo descreveu o conhecimento biológico por trás da Computação Evolutiva, os mecanismos dos AGs, e as estratégias empregadas neste trabalho para tentar escapar deste problema de convergência prematura.

O próximo capítulo tratará da metodologia empregada neste trabalho, relacionando-a ao que já foi publicado sobre estruturas de proteínas, de forma a contextualizar o atual nível de conhecimento na área e justificar as escolhas feitas na elaboração deste.

4. METODOLOGIA

A seguir será apresentada a metodologia deste trabalho, relacionando-a a outros trabalhos encontrados na literatura. Alguns conceitos necessários para a compreensão deste trabalho, como AGs e Proteínas, estão explicados nos capítulos anteriores; outros, como campos de força e os programas utilizados conjuntamente ao trabalho, serão descritos neste capítulo.

4.1 AGs para o Problema de Determinação de Estruturas de Proteínas

Alguns trabalhos envolvendo a aplicação de AGs no problema de determinação de estruturas terciárias de proteínas são agora destacados.

Em [Schulze-Kremer, 1993] utilizou-se uma codificação real para os cromossomos de seu AG, representando seus aminoácidos por coordenadas internas e utilizando como função de energia o campo CHARMM. Na época, devido a restrições de processamento, foram empregadas menos opções de ângulos de torção, tornando o trabalho mais limitado, além de não efetuar alterações no AG padrão.

Unger e Moult [Unger & Moult, 1993] fizeram um modelo 2D de interações e o compararam contra simulações Monte Carlo, também empregando coordenadas internas. Posteriormente, o modelo foi ampliado para 3D [Pedersen & Moult, 1997], atingindo resultados satisfatórios para proteínas pequenas. No entanto, um conjunto de ângulos possíveis pouco abrangente foi empregado, além de várias simulações Monte Carlo e *crossovers* de dois pontos.

Dandekar e Argos [Dandekar & Argos, 1994] utilizaram um AG padrão com uma função heurística e altamente especializada para *fitness* e uma representação por coordenadas internas, atingindo muito bons resultados. No entanto, por estar altamente vinculado às proteínas estudadas e muito especializado, é pouco provável que o método seja útil para outras proteínas sem adaptações significativas.

Herrmannn e Suhai [Herrmann & Suhai, 1995] utilizaram um AG padrão com representação por coordenadas internas junto com uma busca local e um modelo detalhado de campo de força, que atingiu bons resultados, mas para proteínas de tamanho muito reduzido, pelo problema do custo computacional para estruturas maiores.

Lima [Lima, 2006] apresenta um AG multi-objetivo para predição de estruturas, utilizando algumas funções presentes no campo CHARMM e dividindo a proteína em trechos de até 20 aminoácidos, evoluindo cada trecho separadamente, e utilizando uma função de *crossover* diferente (BLX-α). Os resultados mostraram uma favorável taxa de acerto para α-hélices, mas não para folhas β.

AGs Multi-objetivo procuram otimizar várias funções de *fitness* ao mesmo tempo, utilizando mais indivíduos por geração e trabalhando com fronteiras de otimização.

4.2 O Algoritmo

4.2.1 Implementação

Neste trabalho, o algoritmo é implementado em Java, para uma integração mais fácil entre os programas utilizados (apesar da redução de desempenho), e o código contém 3 arquivos diferentes, atendendo a requisitos de reuso e orientação a objetos. O arquivo *cromossomoReal.java* possui a estrutura e os métodos relativos a operações em cromossomos; o arquivo *GA.java* implementa os processos relativos à construção de arquivos, gerações e *rankings* de indivíduos; e o arquivo *callTinker.java* possui apenas o construtor das classes e as chamadas aos métodos. Esta abordagem é baseada no algoritmo descrito em [Linden, 2006].

O algoritmo é iniciado fazendo a leitura do arquivo *sequencia.txt*, que contém a sequência de aminoácidos da proteína que se pretende minimizar a energia. O arquivo deve estar escrito com a sequência de aminoácidos no formato do código de 1 ou 3 letras, e cada aminoácido deve estar separado por espaço, sem passagens de linha. Em seguida, devem ser gerados os ângulos de torção para cada aminoácido.

Inicialmente, o cromossomo criado para o problema consistia de valores para os ângulos ϕ, ψ e χ gerados aleatoriamente entre –180º e 180º. No entanto, esta estratégia não respeita as restrições de Ramachandran [Ramachandran & Sasisekharan, 1968], e foi inicialmente prevista para validar a codificação do AG implementado e, como veremos nos resultados, é insuficiente para se atingir resultados satisfatórios em relação à redução da energia mínima do sistema, devido à quantidade muito grande de combinações possíveis entre os ângulos de todos os aminoácidos da cada proteína.

Esta estratégia é referida nos experimentos como CompRand (ver Tabela 4.1), sendo que o cromossomo é formado por números reais representando os ângulos ϕ, ψ e χ de cada aminoácido. Neste método, a mutação é realizada gerando um novo valor aleatório dentro do intervalo de –180º a 180º, substituindo o valor anterior.

A solução encontrada foi fazer uso de bases de dados de ângulos de torção. Estas bases de dados possuem combinações de ângulos válidas, pois foram retiradas de proteínas cujas estruturas já foram determinadas por ressonância magnética ou cristalografia.

4.2.1.1 Bases de Ângulos

Para os ângulos de torção da cadeia principal, fez-se uso do projeto CADB 2.0 [Sheik *et al.*, 2005], que foi desenvolvido usando dois conjuntos de dados com proteínas com identidade de 25% e 90%, e armazena cerca de 2,28 milhões de combinações de ângulos de torção da cadeia principal, de mais de 7.000 proteínas. Possui funcionalidades como a exibição da cadeia principal e lateral para um aminoácido específico e um estudo de inter-relação entre a cadeia principal e a cadeia lateral. Possui limitações, conforme discutido em [Dayalan *et al.*, 2005], no entanto estas limitações não se referem ao que é preciso para este trabalho. Todas as combinações encontradas de cada aminoácido foram inseridas em arquivos-texto (.txt), no qual cada arquivo é relativo a um aminoácido, sendo nomeado com o código de três letras dos aminoácidos (por exemplo "ala.txt"). No total, portanto, existem 20 arquivos-texto para a cadeia principal.

Para os ângulos de torção da cadeia lateral, empregou-se o banco de dados de Tuffery [Tuffery, 1991]. Este projeto analisou cadeias laterais de proteínas cujas

estruturas já são conhecidas e efetuou a distribuição de frequências de cada sequência conforme estas foram encontradas, gerando duas bases de dados: a base dependente da cadeia principal e a base independente da cadeia principal, a qual foi utilizada para este trabalho. Outros trabalhos fazem uso da mesma abordagem [Koehl & Delarue, 1994] [Holm & Sander, 1992]. Todos os valores estão mantidos como um vetor dentro do algoritmo, por ser uma quantidade menor de ângulos, assim diminuindo o tempo de acesso a estes valores.

Duas abordagens para as bases de dados foram testadas: com os ângulos distribuídos de forma aleatória nas bases (ver Apêndice E), e com os ângulos ordenados de –180º a 180º (ver Apêndice D). Esta última estratégia, que é proposta por este trabalho, se justifica pelo fato de que pelo operador de mutação, uma pequena mudança no índice dos ângulos pode significar uma grande mudança nos valores dos ângulos, quando a base não está ordenada, pois os valores não possuem relação entre si, fazendo com que uma mudança de índice mude os ângulos para valores completamente diferentes, mudando de forma dramática a estrutura protéica, e consequentemente a energia potencial da mesma.

A ordenação é efetuada pelo ângulo ϕ, de forma crescente, ou seja, primeiro vem os ângulos mais próximos de –180º, até os ângulos mais próximos de 180º. Em caso de ângulos ϕ iguais, a ordenação segue para o ângulo ψ, nos mesmos moldes do anterior. Vale lembrar que nem todos os ângulos ϕ podem formar pares com os ângulos ψ existentes, e sim cada entrada na base de dados representa uma combinação única e válida, sem que estes valores possam ser misturados.

4.2.2 Cromossomo

Em seguida, definiu-se um cromossomo no qual cada alelo representa o índice no banco de dados de ângulos relativos ao aminoácido daquela posição, sempre aos pares. O primeiro valor representa o índice da base de dados dos ângulos da cadeia principal, e o segundo é relativo ao índice da base de dados dos ângulos da cadeia lateral. A codificação é inteira.

As figuras 4.1 e 4.2 ilustram um cromossomo típico deste problema, para uma proteína composta por 5 aminoácidos (Figura 4.1) e a ligação entre o cromossomo e as bases de dados (Figura 4.2). Notar que cada par de alelos representa um

aminoácido (por exemplo, $I_{\phi\psi_1}$ e I_{χ_1} são os valores dos índices para o primeiro aminoácido), e todos os aminoácidos devem estar representados no cromossomo, com um vetor auxiliar que armazena os valores relativos àquele índice no banco de dados, para não precisar acessá-lo constantemente.

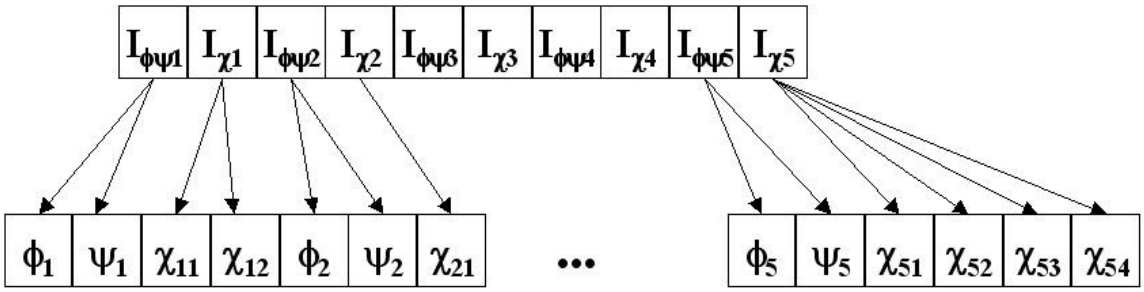

Figura 4.1 – Representação gráfica de um cromossomo típico deste trabalho. Cada I representa o índice da base de dados de ângulos de cada aminoácido constituinte da proteína em estudo, seja da cadeia principal ou da cadeia lateral. Os índices da cadeia principal estão ligados a dois valores, os ângulos ϕ e ψ, que estão armazenados em outro vetor, como um "cromossomo auxiliar"; os índices da cadeia lateral estão ligados desde a nenhum valor (alguns aminoácidos não possuem cadeia lateral) até cinco valores de ângulos χ, também armazenados no vetor auxiliar.

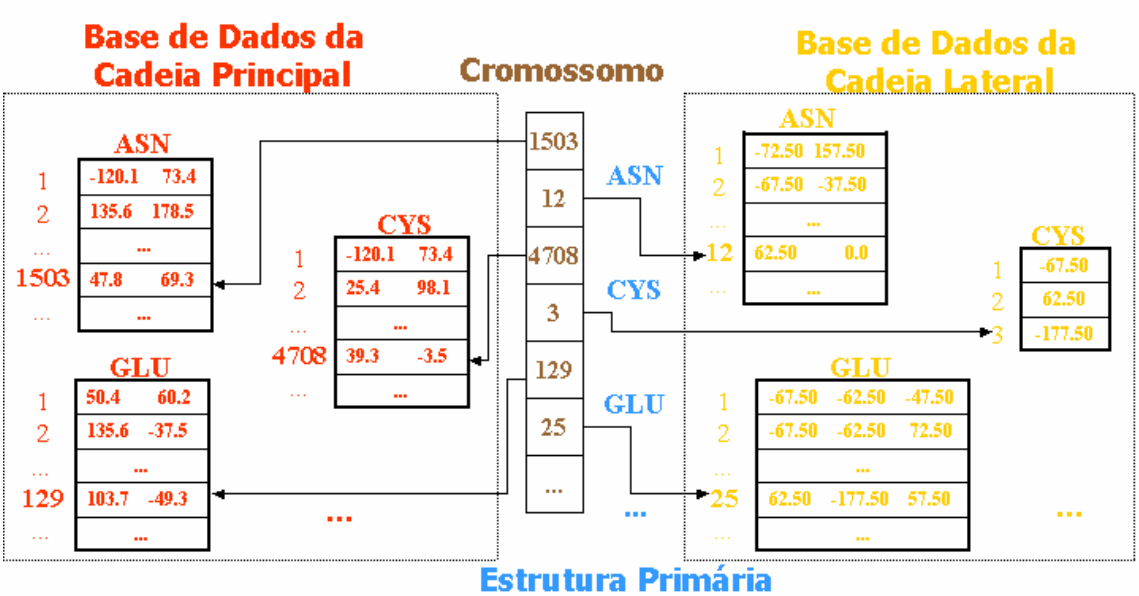

Figura 4.2 – Representação gráfica da relação entre um cromossomo exemplo deste trabalho e as bases de dados a que cada índice se liga.

Assim, o algoritmo efetua a inicialização aleatória de todos os indivíduos (possíveis soluções), montando os cromossomos a partir das bases de dados descritas acima. Uma vez obtidos estes ângulos, cria-se um arquivo de extensão *.dat* (ver Apêndice B) que concatena as informações do aminoácido com seus respectivos

ângulos. Este arquivo é a entrada do algoritmo *protein* (ver seção 4.2.3.1), do pacote de modelagem molecular Tinker [Ponder *et al.*, 1998], que possui implementadas diversas funções relacionadas ao estudo de estruturas químicas, como cálculos de energia, frequências vibracionais, geometria de distâncias, entre outras funções que auxiliam o estudo de proteínas. O software Tinker será melhor explicado na subseção 4.2.3.1.

4.2.3 Fitness

A função de *fitness* escolhida foi a energia potencial total da estrutura. Para efetuar este cálculo, fez-se uso de duas ferramentas constantes do pacote Tinker, *protein* e *analyze*, que serão explicadas a seguir. É importante notar que todos os cálculos efetuados são dependentes da escolha de campo de força que envolve a proteína. Este assunto será discutido na próxima subseção.

4.2.3.1 Ferramentas do Tinker

Protein

É um programa que efetua a construção de peptídeos e proteínas. A partir da entrada de uma sequência de aminoácidos, e opcionalmente dos ângulos de torção (como neste trabalho), o programa retorna as coordenadas internas e cartesianas, utilizando comprimentos e ângulos de ligação padronizados, e definições de átomos a partir do campo de força escolhido para a simulação. A saída gerada utilizada é um arquivo de extensão *.xyz* (ver Apêndice C), que representa a posição no plano cartesiano de cada um dos átomos da estrutura protéica.

Analyze

A seguir, o arquivo de saída do algoritmo *protein* é enviado para o algoritmo *analyze.* Este algoritmo fornece informações sobre uma estrutura protéica específica, que deve estar no formato *.xyz*. As informações disponíveis são: (1) a energia potencial total do sistema, que é a informação necessária para este trabalho; (2) energia específica sobre um átomo; (3) estudo do momento de dipolo total e seus componentes, momentos de inércia e raio de rotação; (4) listagem dos termos de energia usados para computar as energias de interação; e (5) energias associadas a interações individuais específicas.

O software TINKER retorna como saída do algoritmo *analyze* a energia por componente energético e total. Esta soma é utilizada como *fitness* de cada indivíduo, e o objetivo é minimizar a energia ao longo das gerações. Todos os valores são armazenados em relação aos indivíduos aos quais pertencem.

O cálculo da energia potencial total é dependente da função de energia escolhida, pois os parâmetros calculados são dependentes das funções implementadas em cada campo de força. Esta abordagem é usada da mesma forma por outros trabalhos encontrados na literatura [Snow *et al.*, 2002], [Cutello *et al.*, 2005], [Faccioli, 2007], [Brasileiro Filho, 2007].

4.2.3.2 Campo de Força

O campo de força, no sentido computacional, é um conjunto de parâmetros e funções de energia, utilizado para efetuar as simulações de energia da proteína, e representa um papel principal no processo, uma vez que ainda não é possível modelar todas as interações existentes entre átomos e o ambiente que os cerca, devido ao alto custo computacional e alguns mecanismos existentes ainda desconhecidos. Assim, um campo de força que modele as principais e mais importantes interações entre átomos de proteínas deve ser empregado, de modo a aproximar ao máximo a simulação da realidade. Como exemplos, existem modelados no software Tinker os campos de força Amber [Pearlman *et al.*, 1995], composto por quatro termos de energia; OPLS [Jorgensen & Tirado-Rives, 1988], composto por seis termos de energia; e CHARMM [Brooks *et al.*, 1983], composto por sete termos de energia, para ficar entre os mais conhecidos. Uma discussão extensa e útil a este respeito se encontra em [Lazaridis & Karplus, 2000]. Apesar de alguns testes terem sido realizados utilizando o campo OPLS, adotou-se como padrão o campo de força CHARMM27, implementado no pacote Tinker, que é mais completo e utilizado com frequência na literatura [Merkle *et al.*, 1996] [Day *et al.*, 2002] [Cutello *et al.*, 2005] [Anile *et al.*, 2006] [Lima *et al.*, 2007], desde os primeiros trabalhos na área, de Stephen Schulze-Kremer [Schulze-Kremer, 1993] e John Moult [Moult, 1997].

O campo de força CHARMM (ver apêndice A) consiste dos seguintes componentes (Equação 4.1):

$$E_{tot} = E_{bond} + E_{angle} + E_{tors} + E_{urey} + E_{improper} + E_{VdW} + E_{charge} \quad (4.1)$$

sendo:

E_{tot} = Energia potencial total (usado aqui como *fitness*);

E_{bond} = Energia do comprimento de ligação (bond stretching);

E_{angle} = Energia de ângulo de ligação (angle bending);

E_{tors} = Energia de ângulo de torção (torsion angle);

E_{urey} = Energia Urey-Bradley;

$E_{improper}$ = Energia imprópria (improper torsion);

E_{VdW} = Energia Van der Waals;

E_{charge} = Energia Eletrostática (charge-charge).

A seguir o campo de força CHARMM será decomposto, e seus componentes explicados.

Energia de Comprimento de Ligação

A energia de comprimento de ligação, também conhecida como comprimento de ligação de equilíbrio, é dependente da distância entre as partículas que estão sendo analisadas. Se a ligação é comprimida, a nuvem de elétrons dos dois átomos será gradualmente sobreposta. Conforme a ligação é afastada do equilíbrio, a energia começa a aumentar, até um limite onde a ligação se desfaz.

A expansão de Taylor é aplicada em $(r-r_0)$, na qual r_0 é a distância de referência e r é a distância real. A Equação 4.2 apresenta a expansão de Taylor utilizada para o cálculo da energia potencial de ligação.

$$E(r) = E(r_0) + \left.\frac{dE}{dr}\right|_{r=r_0}(r-r_0) + \frac{1}{2}\left.\frac{d^2E}{dr^2}\right|_{r=r_0}(r-r_0)^2 + \frac{1}{6}\left.\frac{d^3E}{dr^3}\right|_{r=r_0}(r-r_0)^3 + \ldots \quad (4.2)$$

Em sua forma simplificada, a Equação 4.2 é concluída no termo $(r-r_0)^2$, sendo conhecida como aproximação harmônica. Considerando $E(r_0)=0$ e que em $r=r_0$ a energia é nula, assim a primeira derivada da energia é zero, e assumindo $k_r = \left.\frac{d^2E}{dr^2}\right|_{r=r_0}$, temos que (Equação 4.3):

$$E_{bond}(r) = \frac{1}{2}k_r(r - r_0)^2 \tag{4.3}$$

Energia de Ângulo de Ligação

O ângulo de ligação θ é obtido a partir da interação entre três átomos (A, B e C). Como os ângulos de ligação variam (experimental e teoricamente) em torno de um valor é suficiente em muitas aplicações utilizar uma representação harmônica, similar à energia de comprimento de ligação (Equação 4.4).

$$E_{angle}(\theta) = \frac{1}{2}k_\theta(\theta - \theta_0)^2 \tag{4.4}$$

na qual θ_0 é o ângulo, k_θ é a constante de força de ângulo de ligação e θ é o ângulo de ligação atual. A energia necessária para alterar o caminho de um ângulo do equilíbrio é muito menor do que a necessária para distorcer o comprimento de ligação, assim as constantes de força de ângulo de ligação são proporcionalmente menores do que as constantes de força de comprimento de ligação. Assim como no caso da energia de comprimento de ligação, quando mais termos são adicionados à equação 4.4, por meio da expansão de Taylor, mais exatidão se obtém no resultado [Faccioli, 2007]. Os parâmetros k_θ e θ_0 utilizados são definidos no próprio campo CHARMM.

Energia de Ângulo de Torção

Argumenta-se que os ângulos de torção de rotação são os mais importantes dos termos intramoleculares em um campo de força; no entanto, alguns campos de força não efetuam cálculos de ângulo de torção, modelando barreiras rotacionais por uma combinação de interações não ligadas, sem obter o mesmo resultado.

Interações de ângulo de torção diferem das interações de comprimento de ligação e ângulo de ligação por dois fatores. O primeiro é que as barreiras de rotação internas são baixas em relação às outras interações, significando que mudanças nos ângulos diedrais podem ser grandes; e segundo, o potencial de torção, E_{tors} é periódico a cada 360º. Assim, seria inapropriado aproximar E_{tors} por uma série de Taylor. Além disso, E_{tors} pode ser utilizada em muitas diferentes maneiras dependendo dos átomos que a compõem.

Costuma-se modelar as interações de torção por uma série de Fourier (Equação 4.5), onde *n* é o número de fases utilizadas, V_n são as constantes de força de rotação de torção e ϕ é o ângulo de torção atual. Soma-se um para mover o zero do potencial e um fator de fase é incluído (γ_n), assim termos com V_n positivo tenham energia mínima em 180º.

$$E_{tors}(\phi) = \sum_n \frac{1}{2} V_n (1 + \cos(n\phi - \gamma_n)) \tag{4.5}$$

Novamente, neste componente os parâmetros empregados são os encontrados no campo de força CHARMM, sem alterações.

Energia Urey-Bradley

O campo de força CHARMM, diferentemente da maioria dos campos de força, inclui o termo de energia Urey-Bradley, que diz respeito às interações entre pares de átomos separados por duas ligações atômicas, conhecida como interação 1:3 átomos. Estas interações são calculadas por uma aproximação harmônica da distância entre os átomos *i* e *j*, como o utilizado para energia de comprimento de ligação e energia de ângulo de ligação [Lima, 2006]. A expressão utilizada para a energia de interação Urey-Bradley é dada pela Equação 4.6, na qual k_{urey} é a constante de força da interação Urey-Bradley e s_0 é a distância entre os átomos *i* e *j*.

$$E_{urey}(s) = \frac{1}{2} k_{urey} (s - s_0)^2 \tag{4.6}$$

Os parâmetros possuem valores padrão no campo CHARMM e não foram alterados neste trabalho.

Energia Imprópria

Energia Imprópria está associada com deformações dos ângulos de torção impróprios. Estes ângulos de torção referem-se a átomos com hibridização sp^2, que geram deformações fora do plano. Este termo está presente em campos de força mais elaborados, assim como a energia Urey-Bradley [Lima, 2006].

Para o cálculo da energia referente às interações de ângulos de torção impróprios, é utilizada uma aproximação harmônica dada pela Equação 4.7, onde

$k_{improper}$ é a constante de força imprópria, ω é o ângulo real e ω_0 é o ângulo de torção impróprio ideal (parâmetros definidos pelo campo CHARMM).

$$E_{improper}(\omega) = \frac{1}{2} k_{improper}(\omega - \omega_0)^2 \tag{4.7}$$

Energia Van der Waals

Van der Waals é uma força elétrica relativamente fraca e inespecífica, de atração de moléculas neutras em gases e na maioria dos líquidos e sólidos orgânicos [Lodish *et al.*, 2004]. Entende-se como uma interação inespecífica o caso em que dois átomos ligados de forma não-covalente (não compartilham um par de elétrons) estiverem suficientemente próximos a ponto dos elétrons de um dos átomos perturbar os elétrons do outro, sendo que esta perturbação gera um dipolo temporário no segundo átomo e atrair-se-ão fracamente. A interação de Van der Waals entre dois átomos faz o balanceamento entre forças de atração e repulsão.

A interação de Van der Waals é frequentemente modelada utilizando o potencial de Lennard-Jones 6-12 que expressa a energia de interação utilizando constantes *A* e *C*, dependentes do tipo do átomo. Os valores de *A* e *C* podem ser determinados por uma variedade de métodos, como distância dos átomos não ligados em cristais e medidas de dispersão na fase gasosa [Lima, 2006]. A Equação 4.8 é a forma geral do potencial de Lennard-Jones

$$E_{vdw} = \sum_{i,j} \frac{A_{i,j}}{r^{12}} - \frac{C_{i,j}}{r^6} \tag{4.8}$$

, na qual $r = \frac{r_{i,j}}{R_i + R_j}$.

Energia Eletrostática ou Carga-Carga

A interação eletrostática entre um par de átomos é representada pelo potencial de Coulomb, sendo *D* a função dielétrica efetiva para a média e *r* é a distância entre dois átomos tendo cargas q_i e q_j . Para a constante dielétrica, o valor foi alterado de 1 (vácuo) para 78,7 a 25º C, o que simula a presença de moléculas de água no sistema. Este fator se justifica pelo aumento desproporcional de tempo computacional necessário para se calcular a interação de uma proteína envolta em

um solvente como a água; no entanto, é possível simular na constante dielétrica o que a água faria caso estivesse presente. Esta energia é dada pela Equação 4.9.

$$E_{charge} = \sum_{i,j} \frac{q_i q_j}{Dr_{i,j}} \tag{4.9}$$

Considerando que as cargas (q_i e q_j) dos átomos não variam, tem-se que a energia eletrostática varia de acordo com a distância entre os átomos. É possível observar que a energia tende a infinito conforme a distância entre os átomos diminui; e quando a distância aumenta, a energia tende a zero [Faccioli, 2007].

4.2.3.3 Avaliação das Estruturas

Apesar de a função de energia ser favorável à execução do algoritmo em termos de ganho computacional, nem sempre menores valores de energia correspondem a estruturas mais próximas da estrutura nativa. Isto posto, é necessário outro método para estudo da eficiência da predição em relação à estrutura original. O método escolhido é o da raiz quadrada média do desvio (RMSD), representado pela Equação 4.10:

$$RMSD = \sqrt{\frac{\sum_i d_i^2}{n}} \tag{4.10}$$

na qual n é o número de átomos e d_i é a distância entre dois átomos i correspondentes das duas estruturas, predita e real [Verli, 2008]. Este cálculo é realizado pelo software VMD [Humphrey *et al.*, 1996], desenvolvido pela Universidade de Illinois e disponível para *download* gratuito em http://www.ks.uiuc.edu/Development/Download/download.cgi?PackageName=VMD.

Este método não é empregado diretamente como *fitness* dos indivíduos porque o objetivo final do algoritmo (não no estágio atual, mas no futuro) é ser capaz de prever estruturas de proteínas ainda não conhecidas, quando não haveria comparação a ser efetuada; no entanto, como os testes são feitos com proteínas já conhecidas, este cálculo pode ser empregado ao final da execução do AG, de maneira a definir a proximidade da estrutura predita em relação à estrutura original.

4.2.4 Seleção

Ao fim de todas as avaliações da geração, uma comparação é feita em busca do par de indivíduos com melhores avaliações. Estes são automaticamente colocados na geração seguinte, sem alterações, processo conhecido como elitismo. Para as posições restantes, é feito o processo de recombinação e mutação para se criar a geração seguinte. No algoritmo genético padrão, o processo para a geração de um novo par de indivíduos começa com a seleção por torneio (ver seção 3.3.3).

4.2.5 Crossover

Selecionados os dois indivíduos, estes são enviados para que seja feito um *crossover* simples entre eles (ver seção 3.3.4), enquanto a população não estiver completa.

4.2.6 Mutação

Por fim, um processo de mutação (ver seção 3.3.5), com probabilidade $1/(2m)$, no qual m é o número de aminoácidos da proteína, dá as características finais de cada indivíduo. Esta taxa é empregada para que em média haja uma mutação por indivíduo por geração. O processo se repete até que o mesmo número de indivíduos da geração anterior seja formado. Tem início a geração seguinte, com avaliação, classificação, *crossover* e mutação até que o número de gerações definido inicialmente seja atingido.

4.2.7 Outras Estratégias

Com as estratégias implementadas neste trabalho, há alterações em alguns destes processos do AG.

Na estratégia de Imigrantes Aleatórios (seção 3.4), além de os dois melhores indivíduos da geração serem automaticamente enviados para a geração seguinte, uma porcentagem da população é formada por indivíduos totalmente novos, gerados aleatoriamente como no processo de inicialização dos indivíduos. Testaram-se taxas de introdução de imigrantes de 2%, 6%, 10% e 30%, mantendo todas as outras características em relação ao AG padrão. Estas taxas foram escolhidas por representar taxas de substituição pequenas, médias e altas de indivíduos. Também se testou a possibilidade de começar a inserir novos indivíduos apenas com 10% das

gerações já executadas, de modo que já houvesse uma convergência da população inicial para depois serem inseridos indivíduos.

Estendendo a estratégia de Imigrantes Aleatórios para uma taxa de introdução de imigrantes variável (seção 3.4), temos o algoritmo de Imigrantes Aleatórios Auto-Organizados Simplificado, que também mantém os outros parâmetros do AG tradicional.

Por fim, testou-se também a estratégia de Hipermutação (seção 3.4), que aumenta a taxa de mutação por 5 gerações, voltando à taxa normal nas 5 gerações seguintes, durante toda a execução do AG. Sendo assim, temos 250 gerações com a taxa normal de mutação e 250 gerações com a taxa de mutação aumentada, para um exemplo de execução com 500 gerações.

Assim, temos os nove algoritmos diferentes, empregados com a base de dados ordenada e desordenada, totalizando assim 15 configurações diferentes para cada proteína estudada. A Tabela 4.1 nomeia os algoritmos e os descreve, com o intuito de facilitar sua identificação no próximo capítulo.

Tabela 4.1 – Algoritmos desenvolvidos neste trabalho.

Estratégia	Descrição
CompRand	AG padrão sem uso de bases de ângulos
AgPad	AG padrão com uso de bases da ângulos
RandIm2	AG com bases de ângulos e Imigrantes Aleatórios, com taxa de substituição de 2%
RandIm6	AG com bases de ângulos e Imigrantes Aleatórios, com taxa de substituição de 6%
RandIm10	AG com bases de ângulos e Imigrantes Aleatórios, com taxa de substituição de 10%
RandIm30	AG com bases de ângulos e Imigrantes Aleatórios, com taxa de substituição de 30%
RandImAp	AG com bases de ângulos e Imigrantes Aleatórios, com taxa de substituição de 10% a partir de 10% das gerações concluídas
AutoRandIm	AG com bases de ângulos, Imigrantes Aleatórios com taxa de substituição dinâmica
Hipermut	AG com bases de ângulos, e taxa de mutação variável

4.3 Considerações Finais

Este capítulo apresentou a metodologia empregada neste trabalho. Muitas outras abordagens podem ser (ou já foram) utilizadas neste problema para a determinação da estrutura tridimensional de proteínas.

Piccolboni [Piccolboni & Mauri, 1998] argumenta que três técnicas de representação de estruturas protéicas foram propostas para algoritmos evolutivos:

- coordenadas cartesianas, que são inviáveis para algoritmos baseados em população, uma vez que estruturas basicamente iguais podem possuir coordenadas completamente diferentes;

- coordenadas internas, que definem a posição dos aminoácidos em relação a seus vizinhos, especificando distâncias e ângulos, a escolha da maioria das abordagens genéticas para enovelamento de proteínas;

- geometria de distâncias, que descrevem uma estrutura por meio de uma matriz de todas as distâncias entre cada par de pontos e foi proposta para minimização de energia desde [Nemethy & Scheraga, 1977].

De acordo com Piccolboni, até o momento de seu trabalho, todas as abordagens evolutivas para predição de estruturas de proteínas eram feitas utilizando coordenadas internas; desta forma, algumas características estruturais relevantes não podem ser descritas como hiperplanos, enquanto a geometria de distâncias seria capaz de calcular as distâncias entre pares de resíduos por meio de fórmulas complexas; no entanto, este processo aumenta o custo computacional. Assim, a representação por coordenadas internas foi escolhida para a execução deste trabalho.

Da mesma forma, outros campos de força podem ser aplicados, e outras estratégias são empregadas para a solução deste problema, em diversas áreas, como Física, Química, Farmácia e Engenharia: modelagem por homologia [Bower *et al.*, 1997], que consiste na modelagem de novas moléculas a partir do conhecimento de moléculas cuja estrutura já foi determinada, por famílias protéicas; *de novo design* [Floudas *et al.*, 2006], no qual o foco é arranjar os aminoácidos para que uma estrutura particular seja formada (com conhecimento prévio de qual estrutura é

necessária para um fármaco, por exemplo); Monte Carlo [Da Silva *et al.*, 2004], [da Silva *et al.*, 2001], [Alves *et al.*, 1990], cujo objetivo é calcular as propriedades de equilíbrio e de transporte de um sistema ao longo de um tempo, por meio de características físicas e simulações (por exemplo, Monte Carlo); entre vários outros métodos, para ficar nos mais comuns.

Assim sendo, o objetivo deste trabalho, num primeiro momento, não é o de atingir o estado da arte na determinação de estruturas de proteínas; é, sim, mostrar que estratégias de otimização atingem bons resultados e podem ser técnicas promissoras nesta área.

O próximo capítulo apresenta os resultados obtidos pelas técnicas descritas neste capítulo, para as proteínas testadas Crambina (código PDB 1CRN), Met-Encefalina (1PLW) e DNA-Ligante (1ENH), retiradas do PDB (Protein Data Bank).

5. RESULTADOS

Este capítulo apresenta os resultados obtidos para os métodos de manutenção e aumento da diversidade de populações em AGs para o problema de determinação de estruturas de proteínas.

Três proteínas foram escolhidas como casos de teste, de acordo com suas características e por terem sido amplamente utilizadas na literatura: Crambina (código PDB 1CRN), Met-Encefalina (código PDB 1PLW) e um DNA/RNA ligante (código PDB 1ENH). Elas serão melhor explicadas a seguir. Estas estruturas foram escolhidas por apresentarem as estruturas secundárias existentes e serem computacionalmente tratáveis, por não serem muito grandes.

O PDB é o maior repositório existente de proteínas de estruturas decifradas (atualmente com mais de 50.000 estruturas), contendo arquivos que descrevem cada proteína de acordo com as coordenadas centrais de cada átomo que faz parte de uma dada proteína, bem como informações estruturais, referências de artigos que publicaram inicialmente a estrutura proposta e observações sobre o processo de obtenção daquela estrutura. Pode ser encontrado no endereço http://www.rcsb.org/pdb/home/home.do .

5.1 Proteínas de Estudo

5.1.1 Crambina (1CRN)

Proteínas possuem, em média, cerca de 350 resíduos. A crambina, no entanto, possui apenas 46 aminoácidos. Ela é encontrada nas sementes do repolho abissínio, e sua função biológica é desconhecida, apesar de se saber que ela não está relacionada a nenhuma doença humana. Possui duas alfa-hélices e duas lâminas-beta formando uma folha antiparalela. Possui seis resíduos de cisteína (cerca de 13% da estrutura), o que é incomum quando comparado a outras proteínas. É muito utilizada tanto teoricamente quanto experimentalmente, pois os cristais de crambina possuem uma difração muito boa, tanto que a estrutura de melhor resolução já determinada até hoje é dela, a 0,54 Å [PDBJ, 2008]. Devido a este fato,

é uma proteína útil para efetuar testes e *benchmarking*, tendo sido utilizada por diversos trabalhos, por exemplo [Schulze-Kremer & Tiedemann, 1994], [Pedersen & Moult, 1996], [Lima, 2006].

Sua energia potencial total, quando analisada pelo pacote Tinker utilizando o campo CHARMM, é de 465,538 kcal/mol. Sua estrutura é demonstrada na Figura 5.1.

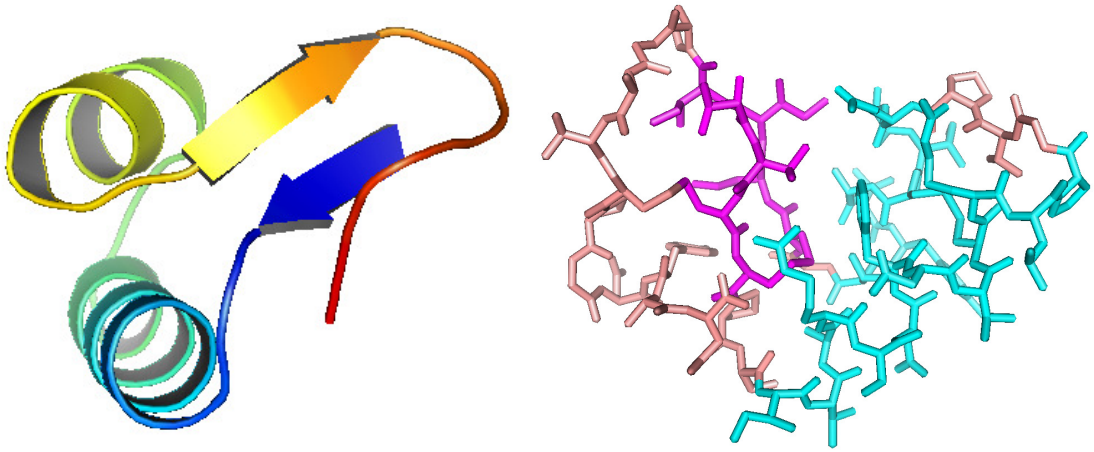

Figura 5.1 – Estrutura da proteína Crambina. À esquerda, é possível ver suas alfa-hélices e suas lâminas-beta. À direita, as ligações são demonstradas.

5.1.2 Met-Encefalina (1PLW)

A Met-encefalina é um neurotransmissor narcótico, dotado de atividade analgésica semelhante à da morfina. Ela se fixa nos receptores de certas células nervosas pela extremidade da sua cadeia tirosina N-terminal, cuja conformação é semelhante à dos opiáceos [MDP, 2008]. Pode terminar sua cadeia com uma Metionina ou uma Leucina. Por sua reduzida estrutura, de apenas 5 aminoácidos, é muito útil como prova de funcionamento de algoritmos, tendo sido empregada em muitos trabalhos, entre eles [Kaiser *et al.*, 1997], [Bindewald *et al.*, 1998], [Nicosia & Stracquadanio, 2008].

Esta estrutura apresenta uma energia potencial total de 345,978 kcal/mol, segundo o pacote Tinker e empregado o campo CHARMM. Sua estrutura é demonstrada pela Figura 5.2.

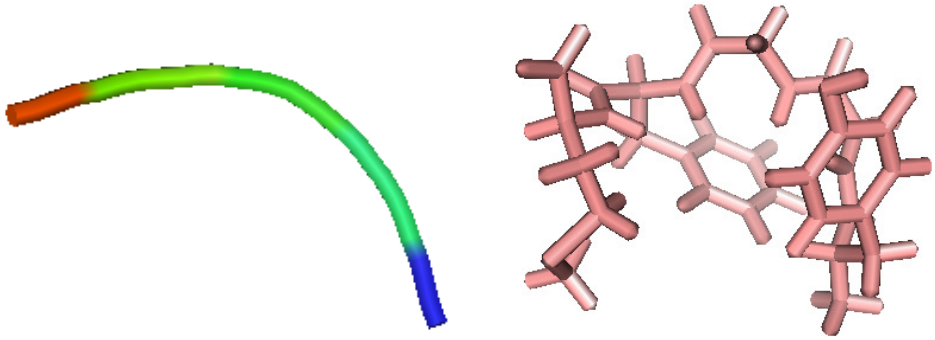

Figura 5.2 – Estrutura da proteína Met-Encefalina, estrutural (esq.), e ligações (dir.).

5.1.3 DNA Ligante (1ENH)

Esta proteína representa o homeodomínio granulado da *Drosophila*, e representa uma importante família de proteínas ligantes ao DNA [Clarke et al, 1994]. Sua principal característica é ser formada por 3 alfa-hélices e 55 aminoácidos, sendo um bom representante do domínio α e um bom estudo de caso, empregado também em [Lima, 2006].

O pacote Tinker, sob o campo de força CHARMM, apresentou uma energia potencial total de 427,305 kcal/mol para esta proteína. A Figura 5.3 exibe sua estrutura tridimensional.

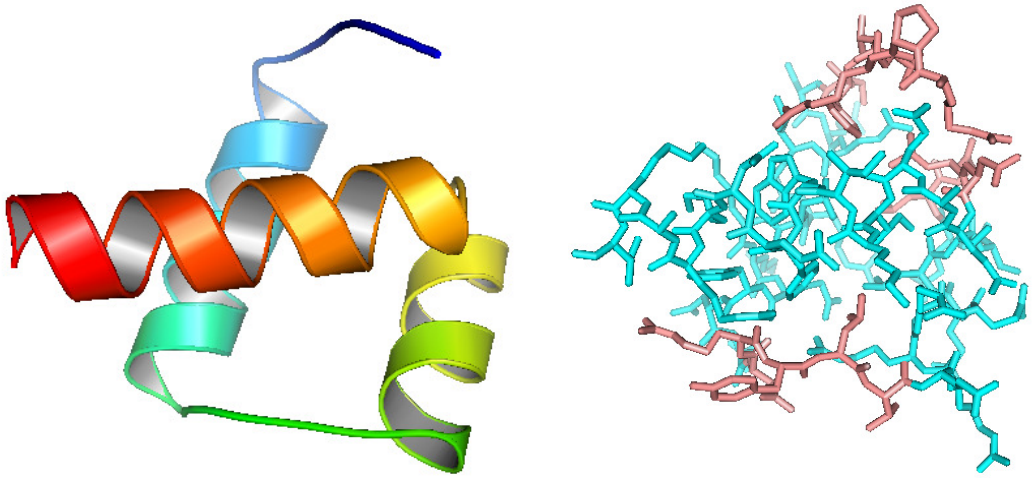

Figura 5.3 – Estrutura tridimensional da proteína DNA-Ligante, com suas alfa-hélices (esq.) e suas ligações (dir.).

5.2 Resultados dos Algoritmos

Todos os algoritmos testados abaixo foram configurados para apresentar os mesmos parâmetros em relação ao AG padrão, ou seja, todos foram executados com dez sementes aleatórias diferentes (sendo que as dez sementes são sempre as mesmas para todas as estratégias), e fazendo uso de 100 indivíduos por geração. O número de gerações utilizado foi de 500 para 1CRN e 1ENH, e 50 gerações para 1PLW, devido a seu reduzido tamanho, que torna a busca mais fácil e sem necessidade do emprego de tantas gerações. As taxas e mutação e *crossover* são as explicadas na metodologia: 80% de probabilidade de *crossover* e 1/(2*m*) de mutação.

Para que comparações estatísticas pudessem ser feitas, testes de Lilliefors [Lilliefors, 1967] foram executados, para certificar que o comportamento dos resultados se assemelha a uma distribuição normal. Para a proteína 1PLW, todos os resultados obtidos apresentam um comportamento normal, para uma taxa α de 5%. Já para a proteína 1CRN, apenas o algoritmo RandIm30 não apresentou comportamento semelhante à curva normal, enquanto a proteína 1ENH apenas não apresentou comportamento normal para Hipermut desordenado e RandIm10. Quando considerados semelhantes à normal, testes T de Student foram executados, com 18 graus de liberdade, pelo software Microsoft Excel, e os p-valores são fornecidos nos resultados; quando não, foram utilizados testes *Wilcoxon rank sum* [Wilcoxon *et al.*, 1963] para comparar os valores obtidos entre os algoritmos testados e o AG padrão. Estas funções estão implementadas no software MATLAB [Mathworks, 1992], que foi empregado para estes cálculos.

5.2.1 CompRand

Este algoritmo não utiliza as bases de dados de ângulos de torção. Foi o primeiro algoritmo implementado, e é uma espécie de validação do método. Não há nenhuma restrição quanto a ângulos inválidos, o que torna o resultado muito aquém dos outros métodos. A Tabela 5.1 mostra os resultados obtidos pelo método, para as três proteínas estudadas. Para esta tabela e todas as seguintes, "*Fitness* do melhor indivíduo" significa o melhor *fitness* obtido ao final de todas as 10 execuções de determinado algoritmo, considerando que com o elitismo, o melhor indivíduo gerado

na última geração é também o melhor de toda a execução do algoritmo; "*fitness* médio" significa a média dos melhores indivíduos para as 10 sementes executadas; "desvio padrão" é o desvio padrão desta média e "Energia Real" é a energia obtida para a estrutura retirada do PDB.

Tabela 5.1 – Resultados do melhor *fitness* nas 10 execuções do algoritmo CompRand, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	49698,864	6833,685	22396,925	22910,434	12913,046	465,538
1PLW	50,410	46,308	48,941	49,267	1,390	345,978
1ENH	376490,552	24069,796	81035,288	41674,952	106900,939	427,305

Vistos os resultados acima, nota-se que destoam muito do valor verdadeiro das proteínas. Isto se dá pelo número muito grande de combinações que podem ser feitas ao se deixar a faixa de valores livre para qualquer possibilidade entre –180º e 180º, contando até mesmo com ângulos inválidos por causarem choques com outros átomos. No entanto, para a proteína 1PLW, resultados melhores que os originais foram obtidos, por conta de relativamente baixa quantidade de combinações que podem ser feitas por estes poucos aminoácidos [Tragante & Tinós, 2008].

O cálculo do RMSD comprova esta teoria. A Tabela 5.2 apresenta os valores obtidos por esta técnica para as proteínas de estudo.

Tabela 5.2 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo CompRand.

	RMSD do melhor	RMSD Médio
1CRN	25,177	40,223
1PLW	8,820	11,059
1ENH	44,798	63,943

Como previsto, todos os valores se apresentam com pouca precisão. É importante frisar, no entanto, que para este método, as sementes aleatórias que geraram os menores *fitnesses* também atingiram os menores RMSDs entre as sementes testadas, o que credencia a função de energia a ser uma avaliação de *fitness* dos indivíduos. Assim, passaremos aos algoritmos seguintes, nos quais restrições começaram a ser implementadas.

5.2.2 AgPad

Em relação ao algoritmo anterior, a mudança apresentada é a inserção das bases de ângulos para a cadeia lateral e para a cadeia principal. Testou-se este método com a base de dados ordenada e desordenada. A Tabela 5.3 apresenta os resultados obtidos para a base ordenada, enquanto a Tabela 5.4 mostra os resultados obtidos para a base desordenada.

Tabela 5.3 – Resultados do melhor *fitness* nas 10 execuções do algoritmo AgPad ordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	1042,387	695,754	831,733	817,469	110,018	465,538
1CRN (1000ger)	1012,933	685,122	812,896	804,914	102,015	465,538
1PLW	48,852	45,599	47,107	46,689	1,092	345,978
1ENH	4722,807	1446,176	3721,321	3391,329	2794,986	427,305

Tabela 5.4 – Resultados do melhor *fitness* nas 10 execuções do algoritmo AgPad desordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	1474,926	626,9084	816,237	763,0615	247,777	465,538
1CRN (1000ger)	1280,081	610,076	765,273	700,381	203,007	465,538
1PLW	49,641	46,203	47,598	47,152	1,223	345,978
1ENH	17467,372	1077,668	4290,645	2727,655	5047,524	427,305

Estão incluídos também nas tabelas anteriores os resultados obtidos para os algoritmos quando executados por 1000 gerações, pois a hipótese a ser testada era se o comportamento seria alterado ao longo de mais gerações. Como se vê pelos resultados, não há alteração significativa que justifique tamanho aumento no custo computacional (dobrando-se o número de gerações, dobra o tempo de execução).

Os resultados comprovam a melhora no desempenho ao se utilizar as bases de ângulos de torção. Por meio de um teste T, verificou-se que para todas as proteínas a probabilidade de estes resultados serem fruto de erro amostral é inferior a 0,5%, tanto ao se utilizar a base ordenada quanto a desordenada, abaixo do p-valor de 5%. Já entre as bases de dados, os melhores resultados individuais foram obtidos pela base desordenada, enquanto a base ordenada atinge melhores resultados na média. Testes T demonstraram que a probabilidade de os resultados

serem os mesmos para a Crambina é de 85,95%, para a Met-Encefalina de 35,57%, e para o DNA-Ligante, de 77,21%, o que não permite concluir a superioridade de um método sobre outro.

Por outro lado, efetuou-se um estudo sobre o número de melhoras dos indivíduos ao longo da execução dos algoritmos, ou seja, quantas vezes o melhor indivíduo se torna ainda melhor ao longo das gerações. Por este estudo, viu-se que, com a base ordenada, um número muito maior de substituições ocorre: para a proteína 1CRN, usando a base ordenada obteve-se uma média de 73,6 melhoras de *fitness* do melhor indivíduo ao longo das 500 gerações, contra 57 da base desordenada, com probabilidade de apenas 0,3% de esta diferença se dar por erro amostral, de acordo com um teste T; para a proteína 1PLW, a média de melhoras foi de 13,7 para a base ordenada, contra 12,6 da base desordenada, atingindo uma probabilidade de erro amostral de 56%. O maior número de melhoras no melhor indivíduo para a base ordenada se deve ao fato de que as mutações podem efetuar um "ajuste-fino" na proteína, pois uma substituição de mais ou menos 1 no índice da base ordenada leva a uma pequena mudança nos valores dos ângulos de torção correspondentes, enquanto para a base desordenada essa mudança leva a um par de ângulos sem nenhuma relação com os ângulos anteriores, podendo ser melhor (pouco provável) ou pior (muito mais provável, pois um processo de evolução já foi executado até aquele ponto e um ângulo favorável já foi escolhido). Assim, há vantagens em se aplicar a base de dados ordenada.

A Tabela 5.5 mostra os resultados obtidos do cálculo do RMSD para as proteínas estudadas utilizando a base ordenada, enquanto a Tabela 5.6 apresenta os mesmos resultados, para a base desordenada.

Tabela 5.5 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo AgPad ordenado.

	RMSD do melhor	RMSD Médio
1CRN	24,527	36,244
1PLW	7,935	11,316
1ENH	37,843	70,705

Tabela 5.6 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo AgPad desordenado.

	RMSD do melhor	RMSD Médio
1CRN	19,593	40,292
1PLW	7,031	10,887
1ENH	37,287	56,375

Neste ponto, comprova-se o fato de que nem sempre o menor valor de energia significa uma estrutura mais próxima da estrutura real, ao menos a partir das interações modeladas neste trabalho. No entanto, pelos motivos explicados no capítulo anterior, o método mais aplicável para cálculo de *fitness* é realmente a energia potencial total. Por outro lado, vê-se que o RMSD obtido utilizando-se as bases de dados é menor que os obtidos sem seu uso, o que mostra que é melhor utilizar as bases de dados.

5.2.3 Hipermut

Em relação ao algoritmo anterior, este altera as taxas de mutação de maneira intermitente, a cada 5 gerações. Este método foi aplicado com as bases de dados ordenada e desordenada. A Tabela 5.7 apresenta os resultados obtidos para este método com a base ordenada, enquanto a Tabela 5.8 mostra os resultados usando a base desordenada. Nela foram incluídos também resultados para 1000 gerações com a base desordenada, de melhor resultado até as primeiras 500 gerações. Porém, como no caso do AG padrão, o maior número de gerações não levou a melhora significativa, então a execução com 1000 gerações foi abandonada.

Tabela 5.7 – Resultados do melhor *fitness* nas 10 execuções do algoritmo Hipermut ordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	904,791	586,178	716,465	697,460	88,462	465,538
1PLW	49,188	43,736	46,237	45,927	1,50	345,978
1ENH	11098,015	1018,911	4920,488	2950,800	4226,027	427,305

Tabela 5.8 – Resultados do melhor *fitness* nas 10 execuções do algoritmo Hipermut desordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	866,729	581,893	672,237	648,535	87,112	465,538
1CRN (1000ger)	776,419	577,128	652,684	625,905	68,302	465,538
1PLW	48,505	44,9416	46,797	46,902	1,078	345,978
1ENH	3780,822	1053,500	2073,168	1641,622	986,010	427,305

É possível notar uma melhora no desempenho do algoritmo em relação ao AG padrão. Esta informação é comprovada por um teste T, que afirma haver probabilidade de erro amostral de apenas 1,9% para a base ordenada e predizendo a Crambina, enquanto há uma probabilidade de 11% para a Crambina usando a base desordenada; para a Met-Encefalina, há uma probabilidade menor que 15% de erro amostral pelo teste T, usando a base ordenada, e de 13,8% para a base desordenada; para o DNA-Ligante, pelo teste *Wilcoxon sum rank*, há uma probabilidade associada de 7,89% de as amostras serem iguais, mas o desempenho para e a base ordenada foi pior que o AG padrão, resultado que destoa dos outros obtidos, por ser o único caso entre todos os algoritmos testados cujo desempenho foi inferior ao AG padrão.

Já em relação ao cálculo do RMSD, os resultados são apresentados nas Tabelas 5.9 e 5.10, respectivamente para a base ordenada e desordenada.

Tabela 5.9 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo Hipermut ordenado.

	RMSD do melhor	RMSD Médio
1CRN	20,335	33,910
1PLW	8,253	11,107
1ENH	30,672	51,806

Tabela 5.10 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo Hipermut desordenado.

	RMSD do melhor	RMSD Médio
1CRN	17,697	29,492
1PLW	6,457	10,592
1ENH	44,621	69,965

5.2.4 RandIm

O AG com imigrantes aleatórios foi testado com diferentes taxas de substituição: 2%, 6%, 10%, e 30%.

5.2.4.1 RandIm2

Os resultados apresentados a seguir referem-se à taxa de inserção de novos indivíduos de 2%. As Tabelas 5.11 e 5.12 mostram os resultados obtidos para este algoritmo usando as bases de dados ordenada e desordenada, respectivamente.

Tabela 5.11 – Resultados do melhor *fitness* nas 10 execuções do algoritmo RandIm2 ordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	594,184	561,596	574,746	531,137	11,978	465,538
1PLW	48,399	43,420	46,577	46,781	1,284	345,978
1ENH	1408,256	795,085	1047,238	1013,552	183,626	427,305

Pelo teste T efetuado comparando-se os resultados obtidos utilizando RandIm2 ordenado contra o AG padrão, há uma chance de 10^{-3}% de os resultados serem os mesmos para a Crambina, 33% para Encefalina e 31% para o DNA-Ligante, ou seja, o algoritmo RandIm2 possui desempenho superior ao AG padrão. É notável que uma significativa melhora seja obtida com a inserção de apenas 2 novos indivíduos por geração, pois a possibilidade de estes genes serem inseridos na geração seguinte é pouca, devido à pequena quantidade de novos "genes".

Tabela 5.12 – Resultados do melhor *fitness* nas 10 execuções do algoritmo RandIm2 desordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	637,158	559,022	590,557	589,029	30,941	465,538
1PLW	47,821	44,602	46,618	46,706	0,899	345,978
1ENH	1662,950	704,036	1196,289	1056,478	458,129	427,305

Pelo teste T, também com a base desordenada o desempenho é melhor que o AG padrão, com probabilidade de erro amostral de 1,8% para a crambina, 5,7% para a Met-Encefalina e 2,7% para o DNA-Ligante.

Da mesma forma que para os algoritmos anteriores, o cálculo de RMSD foi efetuado para estes resultados, e o que e obteve está descrito nas Tabelas 5.13, para a base ordenada, e 5.14 para a base ordenada.

Tabela 5.13 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo RandIm2 ordenado.

	RMSD do melhor	RMSD Médio
1CRN	22,512	33,318
1PLW	8,025	10,842
1ENH	46,948	69,481

Tabela 5.14 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo RandIm2 desordenado.

	RMSD do melhor	RMSD Médio
1CRN	19,416	37,898
1PLW	7,298	10,286
1ENH	30,410	59,863

5.2.4.2 RandIm6

Com a taxa de substituição de 6%, foram obtidos os resultados apresentados pelas Tabelas 5.15 e 5.16.

Tabela 5.15 – Resultados do melhor *fitness* nas 10 execuções do algoritmo RandIm6 ordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	559,859	506,252	525,767	525,340	16,054	465,538
1PLW	47,844	44,86	46,439	46,365	0,979	345,978
1ENH	739,464	691,593	713,582	713,836	12,922	427,305

Tabela 5.16 – Resultados do melhor *fitness* nas 10 execuções do algoritmo RandIm6 desordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	555,950	517,040	538,819	540,134	11,936	465,538
1PLW	47,160	43,404	45,737	45,786	1,246	345,978
1ENH	839,261	673,558	728,646	722,505	60,821	427,305

Novamente, por meio de um teste T, comprovou-se a eficácia deste método sobre o AG padrão. Para 1ENH e a base desordenada, probabilidade de as amostras serem iguais é de apenas 6,7%, para 1CRN é de 0,6% e para 1PLW é de 0,3%; para a base ordenada, para 1PLW a probabilidade é de 16,7%, para 1CRN é de 10^{-6}%, e para 1ENH é de 1,2%. Assim, este método é estatisticamente melhor que o AG padrão. Este resultado, no entanto, não se refletiu no RMSD, como mostram as Tabelas 5.17 e 5.18.

Tabela 5.17 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo RandIm6 ordenado.

	RMSD do melhor	RMSD Médio
1CRN	23,897	34,570
1PLW	8,748	10,777
1ENH	59,990	85,854

Tabela 5.18 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo RandIm6 desordenado.

	RMSD do melhor	RMSD Médio
1CRN	17,512	27,331
1PLW	9,824	11,599
1ENH	47,269	74,117

5.2.4.3 RandIm10

A seguir são apresentados os resultados obtidos ao empregar uma taxa de substituição de indivíduos de 10%, utilizando a base ordenada (Tabela 5.19). Não se

empregou a base desordenada por opção, uma vez que a base ordenada representa uma contribuição deste trabalho.

Tabela 5.19 – Resultados do melhor *fitness* nas 10 execuções do algoritmo RandIm10 ordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	569,479	519,987	538,219	533,379	18,476	465,538
1PLW	47,862	44,847	46,160	46,131	0,848	345,978
1ENH	1085,296	746,979	868,154	848,293	101,005	427,305

Por meio de um teste T, verifica-se que esta taxa de substituição é estatisticamente superior ao AG padrão, possuindo apenas 10^{-3}% de chances de esta diferença se dar por causa de erro amostral, para a Crambina e de 4% para a Met-Encefalina. Os resultados para o DNA-Ligante não apresentaram comportamento semelhante à curva normal (p=0,035 pelo teste de Lilliefors), então um teste Wilcoxon *rank sum* foi executado, e provou-se a superioridade do método RandIm10, com p-valor de $2{,}16 \times 10^{-5}$.

Na sequência, os resultados obtidos pelo cálculo do RMSD são exibidos (Tabela 5.20).

Tabela 5.20 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo RandIm10 ordenado.

	RMSD do melhor	RMSD Médio
1CRN	19,728	34,233
1PLW	6,008	10,779
1ENH	34,261	68,310

5.2.4.4 RandIm30

Com uma taxa de substituição de 30% ocorre uma dificuldade de convergência do AG, maior que a ideal, pois menos indivíduos são gerados explorando as melhores soluções correntes, que evoluíram para chegar àquele ponto, pois muitos indivíduos que poderiam ser resultados de *crossover* entram como imigrantes aleatórios de baixo *fitness*, tornando o algoritmo ineficiente, como pode ser visto nos resultados. Assim, não foi testado o uso da base desordenada nem se testou a proteína 1ENH, por ser a mais computacionalmente custosa. Os resultados obtidos para as outras proteínas, usando a base ordenada, estão demonstrados na Tabela 5.21. Os resultados de RMSD estão na Tabela 5.22. Estes também apresentam performance inferior, mostrando que uma taxa alta de substituição de

indivíduos é prejudicial ao AG, e justificando o estudo de técnicas que efetuem a substituição de indivíduos a taxas dinâmicas.

Tabela 5.21 – Resultados do melhor *fitness* nas 10 execuções do algoritmo RandIm30 ordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	774,967	661,803	735,586	747,037	43,491	465,538
1PLW	48,819	46,426	47,907	47,930	0,670	345,978

Tabela 5.22 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo RandIm30 ordenado.

	RMSD do melhor	RMSD Médio
1CRN	25,392	43,956
1PLW	9,568	12,124

Ainda assim, por meio de um teste *Wilcoxon sum rank* é possível, para a proteína Crambina, provar que os resultados obtidos por RandIm30 são melhores que o AG padrão, com p=0,452.

5.2.4.5 RandImAp

A principal contribuição deste algoritmo é que ele não começa a substituição de indivíduos logo na primeira geração, passando a inserir diversidade após 10% do algoritmo já ter sido executado. Adotou-se uma taxa de substituição de 10% como teste para este algoritmo, e a base de dados ordenada. Os resultados estão dispostos na Tabela 5.23.

Tabela 5.23 – Resultados do melhor *fitness* nas 10 execuções do algoritmo RandImAp ordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	578,671	507,320	552,867	560,724	23,347	465,538
1PLW	47,764	43,746	46,252	46,395	1,230	345,978
1ENH	977,007	749,063	839,056	815,778	81,739	427,305

Ao ser executado um teste T, descobre-se que este algoritmo é estatisticamente superior ao AG padrão, pois a probabilidade de que as duas amostras sejam iguais é de apenas 11,8% para a Met-Encefalina, enquanto para a Crambina é de 10^{-3}% e para o DNA-Ligante, de 1,5%. Porém, contra o AG com imigrantes aleatórios com substituição desde a primeira geração, não se conclui a superioridade de um método sobre outro, ao menos para os parâmetros definidos por estes testes. A convergência do RandImAp é mais rápida que o RandIm10,

portanto este algoritmo é mais apropriado quando o tempo é um fator crítico. Os resultados do cálculo de RMSD estão na Tabela 5.24.

Tabela 5.24 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo RandImAp ordenado.

	RMSD do melhor	RMSD Médio
1CRN	25,667	37,166
1PLW	7,165	7,905
1ENH	52,238	67,214

5.2.5 AutoRandIm

Principal contribuição deste trabalho, o algoritmo AutoRandIm realiza a inserção de novos indivíduos com taxas diferentes a cada geração, dependendo do que ocorreu na geração anterior, conforme explicado na Seção 3.3.3. Os resultados obtidos por este método estão demonstrados nas Tabelas 5.25 e 5.26, para a base ordenada e desordenada, respectivamente.

Tabela 5.25 – Resultados do melhor *fitness* nas 10 execuções do algoritmo AutoRandIm ordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	550,101	527,095	538,387	539,519	8,359	465,538
1PLW	48,306	42,819	46,229	46,706	1,640	345,978
1ENH	1113,551	732,863	909,881	893,529	156,665	427,305

Tabela 5.26 – Resultados do melhor *fitness* nas 10 execuções do algoritmo AutoRandIm desordenado, em kcal/mol.

	Pior Fitness	Melhor Fitness	Fitness Médio	Mediana	Desvio Padrão	Energia Real
1CRN	559,338	503,559	535,086	533,946	20,984	465,538
1PLW	47,864	43,876	46,077	45,898	1,267	345,978
1ENH	934,531	639,502	759,303	754,978	89,934	427,305

Testes T executados comparando os resultados obtidos por este algoritmo e os métodos de substituição fixa de indivíduos mostram um desempenho estatisticamente melhor do AG auto-organizável contra as taxas de substituição de 2%, 10% e 30%, e uma performance equivalente ao algoritmo com 6% de reposição, atingindo 98% e 63,2% para a Crambina, com a base ordenada e desordenada, respectivamente; e 86,6% e 90,8%, para a Met-Encefalina, também usando a base ordenada e desordenada, respectivamente; e para o DNA-Ligante, o resultado obtido foi pior que RandIm6, tanto para a base ordenada quanto para a base desordenada, mas melhor que o AG padrão, com apenas 1,7% de probabilidade

de erro amostral, de acordo com o teste T, para a base ordenada, e 6,9% para a base desordenada.

Se os resultados atingem valores semelhantes, o foco do estudo passa ao número de substituições efetuadas por geração para o algoritmo. Para a proteína Crambina, a taxa média de reposição de indivíduos foi de 4,45% para a base ordenada, contra 10% de substituição pra o melhor resultado obtido usando taxas de substituição fixas, enquanto que com a base desordenada a taxa média foi de 4,55% de novos indivíduos por geração, contra 6% obtidos pelo melhor resultado alcançado usando taxas fixas. É importante que um menor número de novos indivíduos seja gerado, pois o processo mais custoso computacionalmente é a geração de novos indivíduos, pelo acesso aos arquivos das bases de ângulos.

Outro ponto importante é que o máximo de indivíduos substituídos em uma geração foi 32% para ambas as bases de dados, sendo que até 70% poderiam ser substituídos de uma vez, mostrando que tão alta taxa de reposição não é necessária.

A Figura 5.4 mostra o comportamento da curva de substituição de indivíduos para a Crambina, usando a base ordenada; a Figura 5.5 faz o mesmo, mas para a base desordenada.

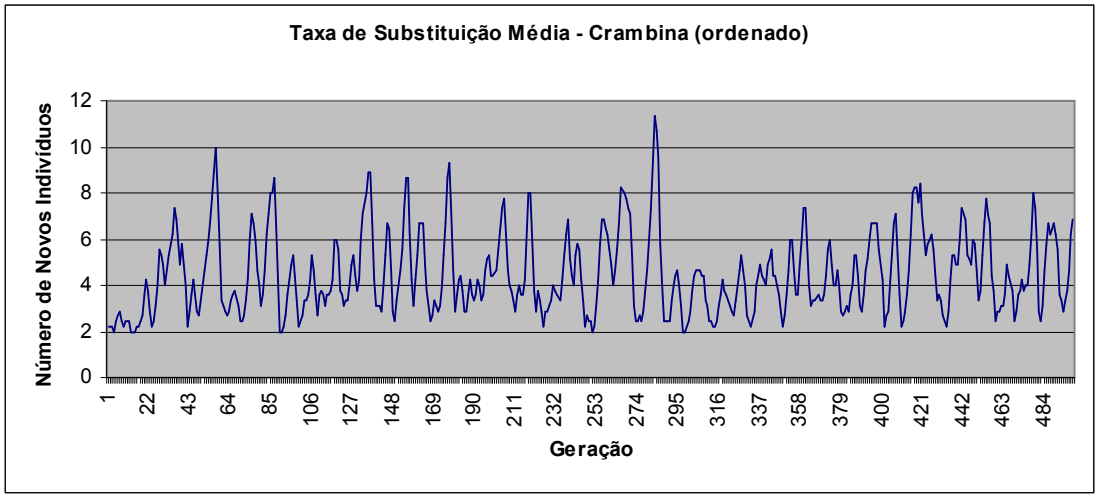

Figura 5.4 – Curva média de substituição de indivíduos pelo algoritmo AutoRandIm ordenado, geração a geração.

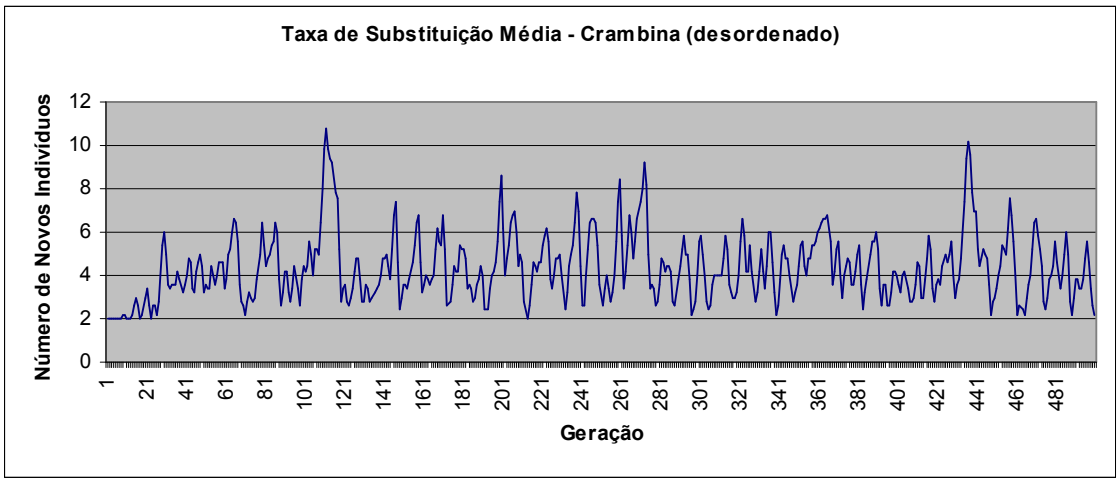

Figura 5.5 – Curva média de substituição de indivíduos pelo algoritmo AutoRandIm desordenado, geração a geração, para a proteína 1CRN.

Nota-se, na Figura 5.4, um maior número de pontos fora da base (o mínimo de indivíduos substituídos é 2, esta é a base) a partir da metade da execução do algoritmo, ou seja, quando a diversidade é menor, o número de novos indivíduos aumenta. Já na figura 5.5, notam-se picos periódicos (aproximadamente 160 gerações) nos quais um grande número de indivíduos é substituído, com consequente redução dramática nas gerações seguintes. Esta curva mostra que o algoritmo é eficiente em inserir mais diversidade justamente quando é mais importante: quando todos os indivíduos estão muito semelhantes.

Da mesma forma, para a proteína Met-Encefalina, houve um menor número de inserção de indivíduos do que as taxas fixas. Para a base ordenada, a reposição média foi de 3,26% por geração, muito menor que quaisquer taxas de reposição fixas usadas de resultado semelhante (a taxa de reposição de 2% não obteve os melhores resultados); para a base desordenada, 3,46% de indivíduos foram substituídos em média a cada geração, resultado também melhor que as taxas fixas testadas com resultados próximos. O número máximo de substituições de indivíduos em uma única geração foi de 18 indivíduos para ambas as bases de ângulos, longe dos 70 possíveis se necessário.

As Figuras 5.6 e 5.7 mostram a curva de reposição média de indivíduos para a proteína Met-Encefalina, usando a base ordenada (5.6) e desordenada (5.7).

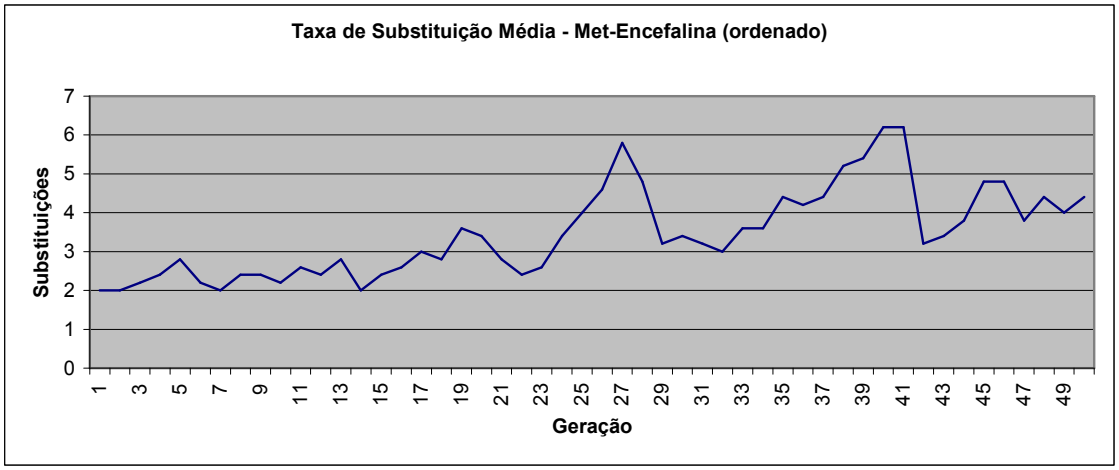

Figura 5.6 - Curva média de substituição de indivíduos pelo algoritmo AutoRandIm ordenado, geração a geração, para a proteína 1PLW.

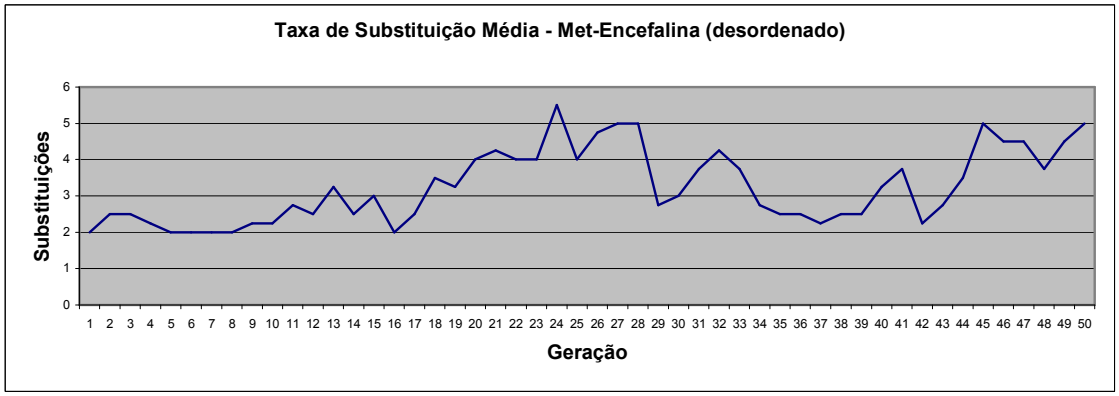

Figura 5.7 - Curva média de substituição de indivíduos pelo algoritmo AutoRandIm desordenado, geração a geração, para a proteína 1PLW.

Para ambas as curvas, é possível traçar uma reta suporte ascendente, que mostra que conforme a diversidade vai diminuindo devido a sucessivos *crossovers*, o número de novos indivíduos inseridos aumenta, para manter esta diversidade. Este fator é extremamente importante para que o espaço de busca seja bem explorado, e o algoritmo é capaz de efetuar este processo.

De forma semelhante, para o DNA-Ligante o comportamento da curva é ligeiramente ascendente, com alguns picos de substituição de indivíduos, mas em geral em valores baixos. Utilizando a base ordenada, a média de substituições em todas as gerações foi de 3,79%, enquanto para a base desordenada foi de 3,90%. O máximo de indivíduos novos em uma geração foi de 22 para a base desordenada.

A seguir vê-se o comportamento das curvas de substituição de indivíduos para a base ordenada (Figura 5.8) e para a base desordenada (Figura 5.9).

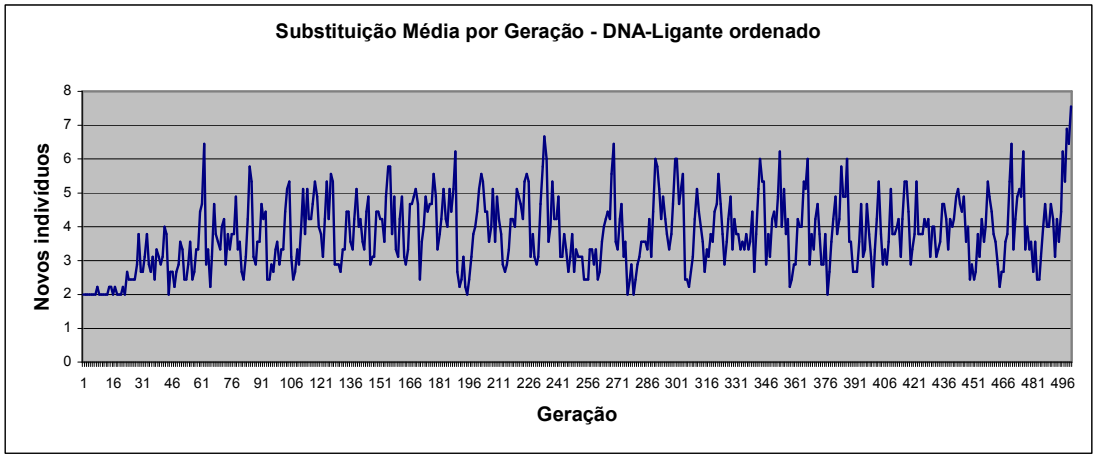

Figura 5.8 - Curva média de substituição de indivíduos pelo algoritmo AutoRandIm ordenado, geração a geração, para a proteína 1ENH.

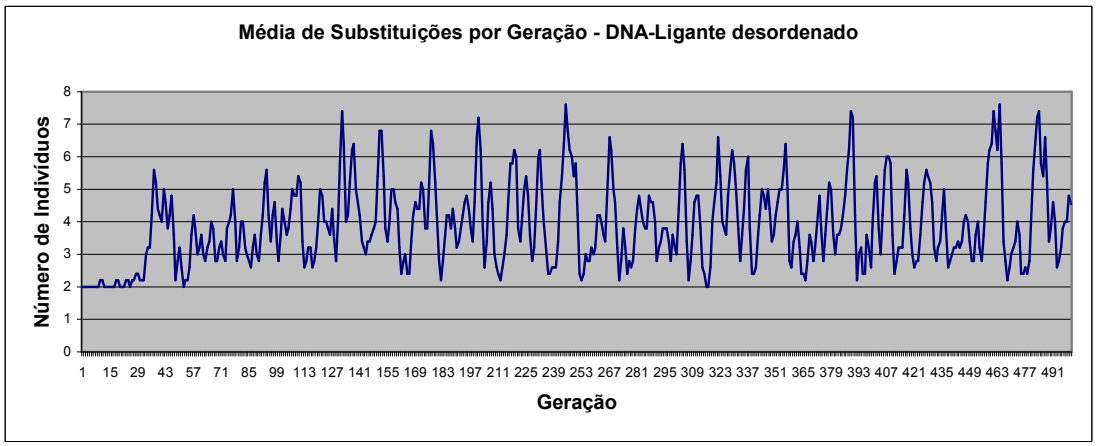

Figura 5.9 - Curva média de substituição de indivíduos pelo algoritmo AutoRandIm desordenado, geração a geração, para a proteína 1ENH.

A título de exemplificação, a Figura 5.10 mostra o comportamento da substituição de indivíduos para uma única semente aleatória, para a proteína Met-Encefalina e usando a base de dados ordenada, enquanto a Figura 5.11 mostra curva para uma semente para o DNA-Ligante, usando a base desordenada.

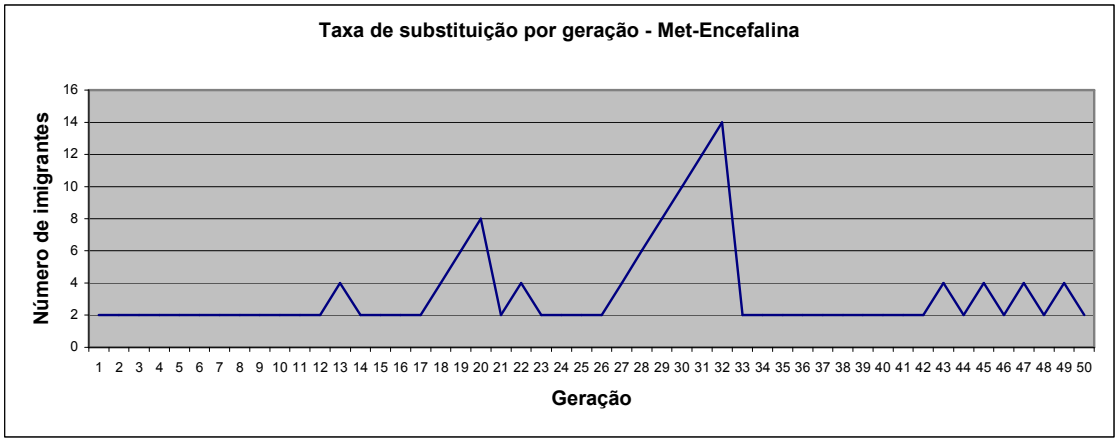

Figura 5.10 – Inserção de novos indivíduos, geração a geração, para a proteína Met-Encefalina e uma semente aleatória apenas (semente 5).

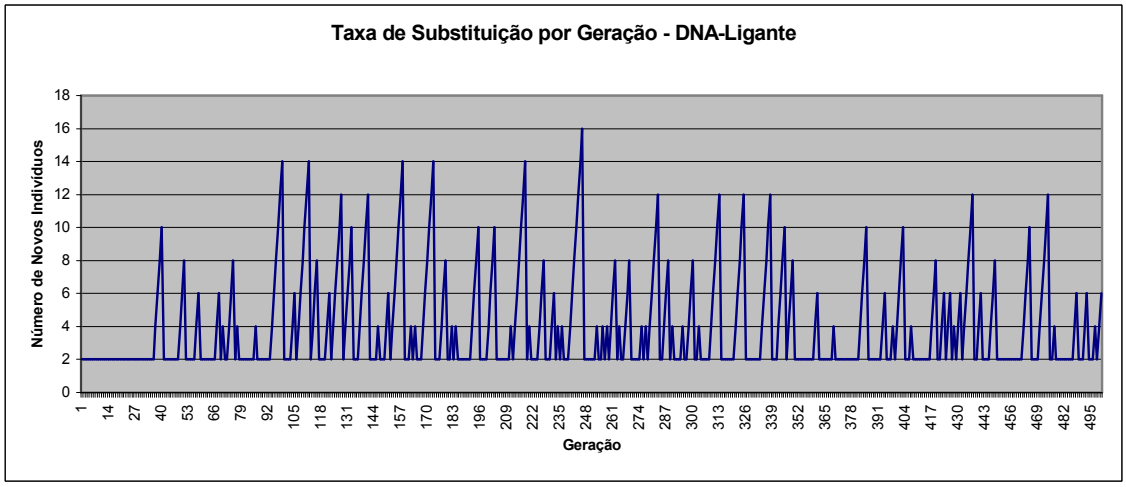

Figura 5.11 – Inserção de novos indivíduos, geração a geração, para a proteína DNA-Ligante e uma semente aleatória apenas (semente 6).

A seguir, as Tabelas 5.27 e 5.28 mostram os RMSDs resultantes da utilização do algoritmo AutoRandIm.

Tabela 5.27 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo AutoRandIm ordenado.

	RMSD do melhor	RMSD Médio
1CRN	20,607	29,500
1PLW	6,719	10,151
1ENH	32,732	48,757

Tabela 5.28 – RMSD do melhor indivíduo e RMSD médio do melhor indivíduo para as 10 sementes aleatórias em Å, para o algoritmo AutoRandIm desordenado.

	RMSD do melhor	RMSD Médio
1CRN	22,865	31,655
1PLW	9,528	10,769
1ENH	53,544	71,118

5.3 Análise Visual

Assim como os valores de RMSD mostram, as estruturas preditas não apresentam semelhança em geral com a proteína real. Alguns trechos possuem orientação semelhante, porém a estrutura secundária folha-β não foi predita com sucesso, e poucas vezes alfa-hélices foram formadas. Isto ocorre porque este tipo de informação não foi dado ao algoritmo, e apenas uma pequena faixa dos valores possíveis de ângulos gera estas estruturas secundárias.

As próximas páginas apresentam a melhor estrutura obtida para cada um dos algoritmos, de acordo com o cálculo de RMSD. Na próxima página, a Figura 5.12 mostra todas as melhores estruturas obtidas para a proteína Crambina; na página seguinte, a Figura 5.13 mostra as estruturas obtidas para a Met-Encefalina; e a Figura 5.14 mostra as estruturas obtidas para o DNA-Ligante.

Deve-se notar que na maioria das vezes a estrutura de menor energia potencial não possui nenhuma relação com a estrutura de menor RMSD. Este resultado coincidiu apenas para os algoritmos CompRand e AutoRandIm, para a proteína Crambina; RandImAp para a Met-Encefalina, e para o DNA-Ligante, nenhuma proteína de menor RMSD coincidiu com a de menor energia potencial, pelos algoritmos executados.

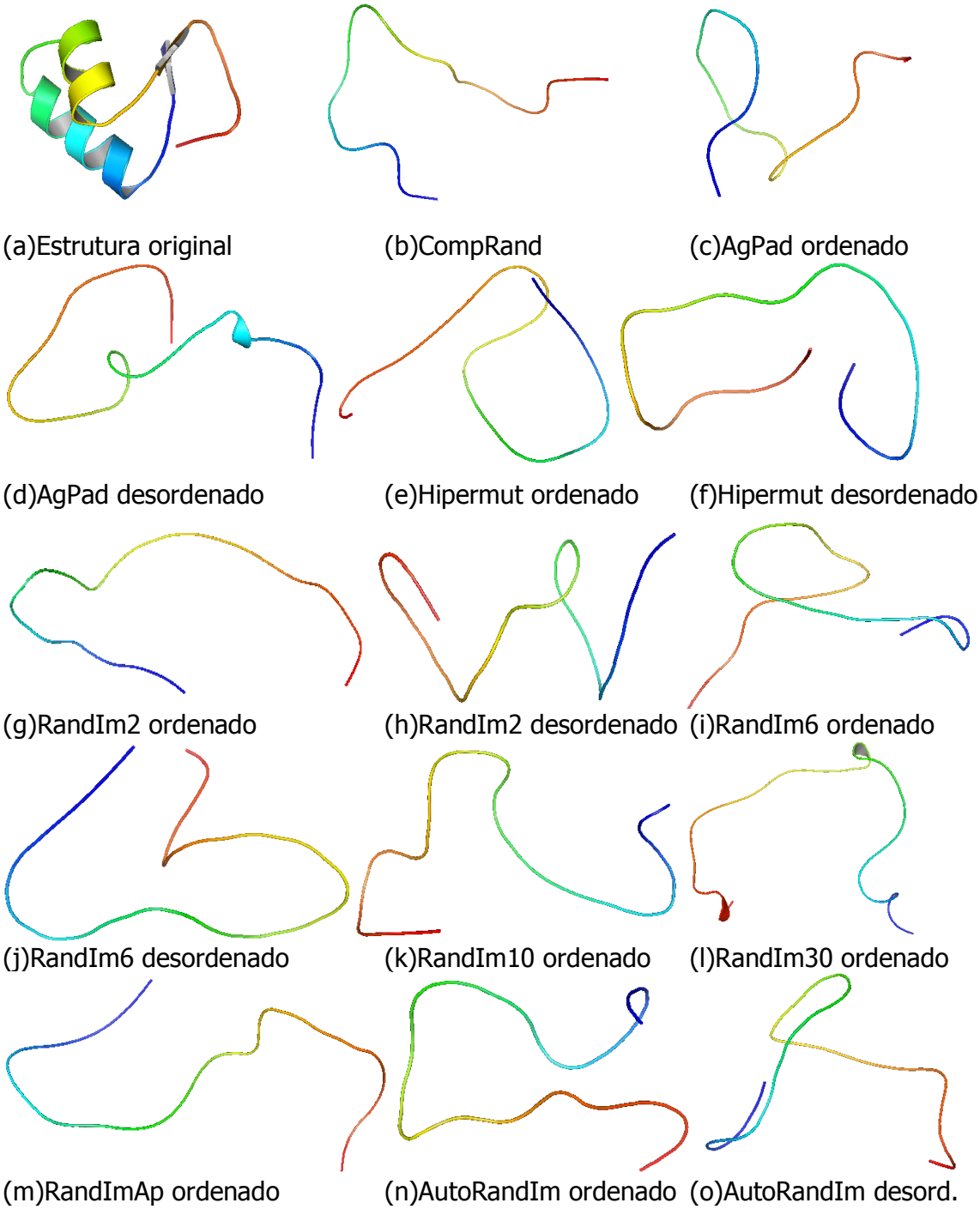

Figura 5.12 – Visualização estrutural dos melhores resultados obtidos por cada um dos algoritmos trabalhados nesta dissertação, para a proteína Crambina, de acordo com o cálculo de RMSD. A estrutura de menor RMSD está representada em (j), obtendo 17,512 Å, enquanto a estrutura de menor energia potencial está representada em (o), 503,5586 kcal/mol.

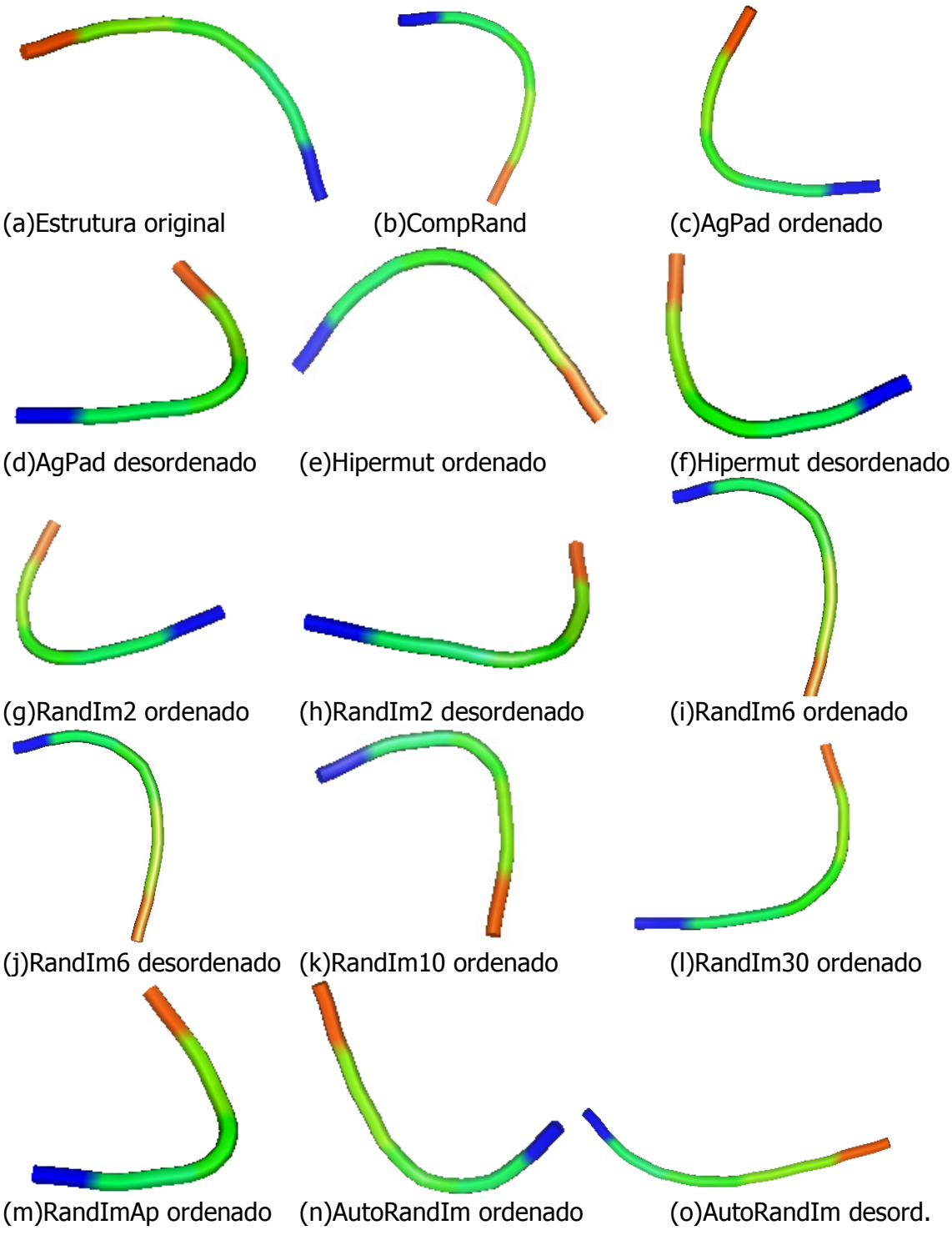

Figura 5.13 – Visualização estrutural dos melhores resultados obtidos por cada um dos algoritmos trabalhados nesta dissertação, para a proteína Met-Encefalina, de acordo com o cálculo de RMSD. A estrutura de menor RMSD está representada em (k), obtendo 6,008 Å, enquanto a estrutura de menor energia potencial está representada em (m), 42,1026 kcal/mol.

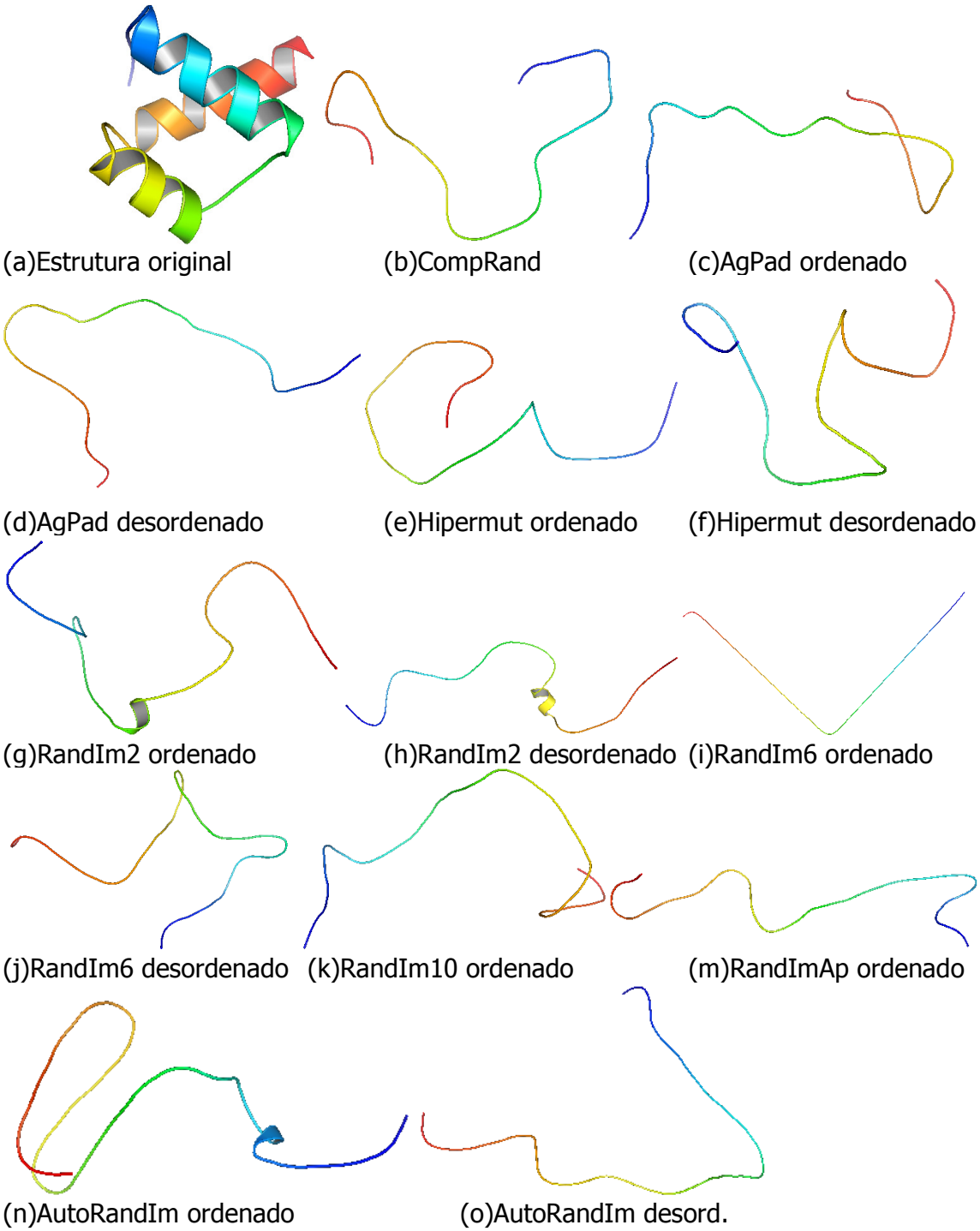

Figura 5.14 – Visualização estrutural dos melhores resultados obtidos por cada um dos algoritmos trabalhados nesta dissertação, para a proteína DNA-Ligante, de acordo com o cálculo de RMSD. A estrutura de menor RMSD está representada em (h), obtendo 30,410 Å, enquanto a estrutura de menor energia potencial está representada em (o), 639,502 kcal/mol.

5.4 Discussão

Todos as variações do AG testadas por este trabalho foram capazes de obter desempenho superior ao AG padrão na tarefa de minimização de energia potencial total, desempenho este comprovado por meio de testes estatísticos, como mostram a seguir as tabelas 5.29 a 5.31, uma para cada proteína empregada neste trabalho.

Tabela 5.29 – Resultados para a proteína Crambina.

Algoritmo	Melhor Fitness	Fitness Médio	Desvio Padrão
CompRand	6833,685	22396,925	12913,046
AGPad (ord)	695,754	831,733	110,018
AGPad (ord) 1000 gerações	685,122	812,896	102,015
AGPad (desord)	626,908	816,237	247,777
AGPad (desord) 1000 ger.	610,076	765,273	203,007
Hipermut (ord)	586,178	716,465	88,462
Hipermut (desord)	581,893	672,237	87,112
Hipermut (desord) 1000 ger.	577,128	652,684	68,302
RandIm2 (ord)	561,596	574,746	11,978
RandIm2 (desord)	559,022	590,557	30,941
RandIm6 (ord)	506,252	525,767	16,054
RandIm6 (desord)	517,040	538,819	11,936
RandIm10 (ord)	519,987	538,219	18,476
RandIm30 (ord)	661,803	735,586	43,491
AutoRandIm (ord)	527,095	538,387	8,359
AutoRandIm (desord)	503,559	535,086	20,984
Energia Real (PDB)	465,538		-

Tabela 5.30 – Resultados para a proteína Met-Encefalina.

Algoritmo	Melhor Fitness	Fitness Médio	Desvio Padrão
CompRand	46,308	48,941	1,390
AGPad (ord)	45,599	47,107	1,092
AGPad (desord)	46,203	47,598	1,223
Hipermut (ord)	43,736	46,237	1,50
Hipermut (desord)	44,492	46,797	1,078
RandIm2 (ord)	43,420	46,577	1,284
RandIm2 (desord)	44,602	46,618	0,899
RandIm6 (ord)	44,86	46,439	0,979
RandIm6 (desord)	43,404	45,737	1,246
RandIm10 (ord)	44,847	46,160	0,848
RandIm30 (ord)	46,426	47,907	0,670
AutoRandIm (ord)	42,819	46,229	1,640
AutoRandIm (desord)	43,876	46,077	1,267
Energia Real (PDB)	345,978		-

Tabela 5.31 – Resultados para a proteína DNA-Ligante.

Algoritmo	Melhor Fitness	Fitness Médio	Desvio Padrão
CompRand	24069,796	81035,288	106900,939
AGPad (ord)	1446,176	3721,321	2794,986
AGPad (desord)	1077,668	4290,645	5047,524
Hipermut (ord)	1018,911	4920,488	4226,027
Hipermut (desord)	1053,500	2073,168	986,010
RandIm2 (ord)	795,085	1047,238	183,626
RandIm2 (desord)	704,036	1196,289	458,129
RandIm6 (ord)	691,593	713,582	12,922
RandIm6 (desord)	673,558	728,646	60,821
RandIm10 (ord)	746,979	868,154	101,005
AutoRandIm (ord)	732,863	909,881	156,665
AutoRandIm (desord)	639,502	759,303	89,934
Energia Real (PDB)	427,305		-

O uso de bases de ângulos permite ao algoritmo uma busca muito mais direcionada, e com resultados muito melhores, como apresentado neste capítulo. De fato, permitir ao algoritmo que gere aleatoriamente valores sem nenhum critério de exclusão deixa o problema com uma quantidade dramaticamente alta de possibilidades, de modo que soluções razoáveis sejam muito pouco prováveis de serem encontradas. Já a comparação entre bases ordenadas e desordenadas é útil para mostrar as possibilidades de um processo de *hill-climbing*, dado pelo uso de bases ordenadas, que mostrou que as mutações permitiram um maior número de melhoras do indivíduo ao trocar os valores dos ângulos de torção por valores semelhantes.

Além disso, vê-se pelas médias dos *fitnesses* dos indivíduos a cada geração, para todos os algoritmos testados, que o objetivo de manter a diversidade ou aumentá-la periodicamente é eficaz, pois, quando o AG padrão é executado, a partir de uma certa geração, em todos os casos, a média da população é muito próxima ao *fitness* do melhor indivíduo, o que significa que todos os indivíduos são muito semelhantes; por outro lado, quando as técnicas trabalhadas nesta dissertação foram aplicadas, a média nunca ficou próxima ao *fitness* do melhor indivíduo. Como exemplo, as Figuras 5.15 e 5.16 mostram a média do *fitness* da população geração a geração, para o AG padrão (Figura 5.15) e para AutoRandIm (figura 5.15), para uma única semente.

Para o AG padrão, na maioria das gerações a linha da média não aparece, por estar muito próxima ao *fitness* do melhor indivíduo (o melhor indivíduo sempre

possui *fitness* muito menor que os indivíduos menos evoluídos, o que deixa a média sempre em valores altos). Para AutoRandIm, a linha da média nunca desaparece, ou seja, a diversidade está sendo mantida. Esta diversidade é extremamente importante para que mais regiões do espaço de busca possam ser varridas, aumentando a confiabilidade do algoritmo e a expectativa de que resultados melhores sejam alcançados por conta desta maior capacidade de busca no espaço de soluções.

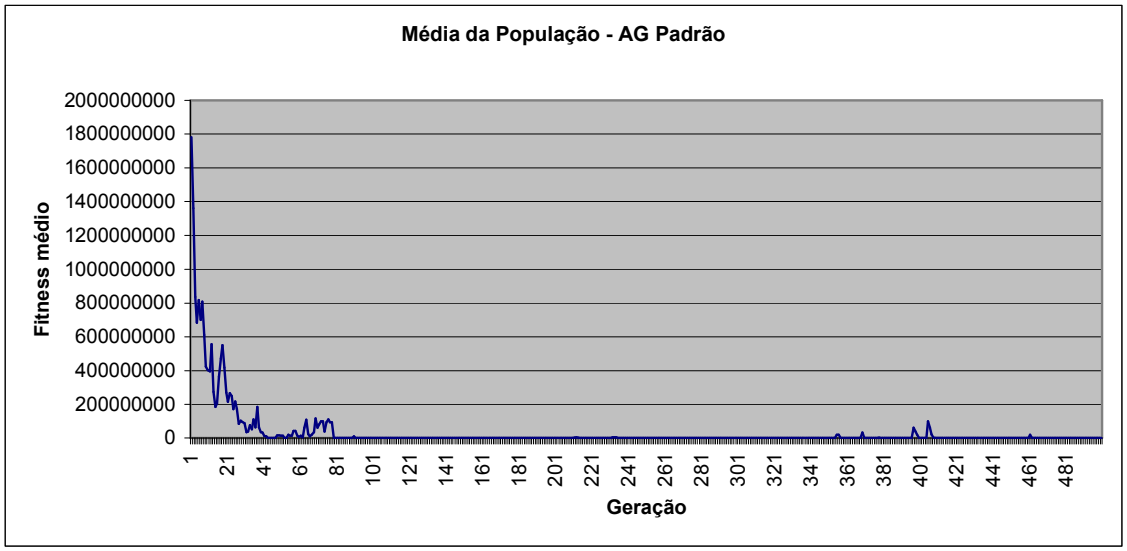

Figura 5.15 – *Fitness* médio da população ao longo das gerações para o AG padrão (para uma semente aleatória). Onde a linha não aparece o valor é muito próximo ao *fitness* do melhor indivíduo.

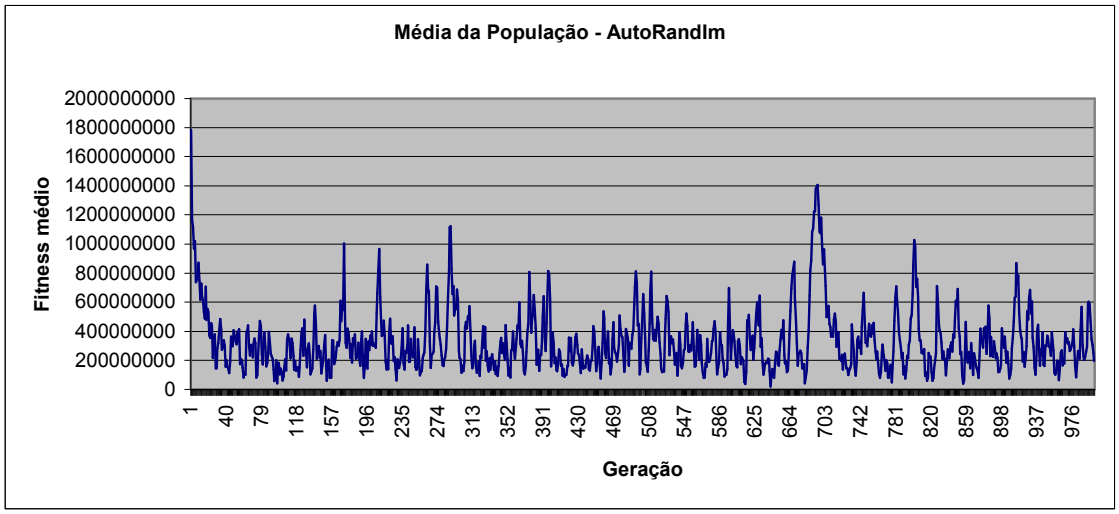

Figura 5.16 – *Fitness* médio da população ao longo das gerações para AutoRandIm. Notar que em nenhum momento a linha desaparece, o que indica que a média não está próxima do *fitness* do melhor indivíduo.

No entanto, para a tarefa de determinação da estrutura terciária de proteínas, os resultados ficaram a desejar, pois as estruturas obtidas pela predição são diferentes da estrutura nativa das proteínas testadas, seja pela análise visual ou pelo cálculo do RMSD.

Ainda assim, sem nenhum conhecimento prévio, alguns esboços de estruturas α-hélice foram preditos, o que mostra que há possibilidade que com maior direcionamento, estas estruturas sejam alcançadas.

Cabe ressaltar, portanto, que a tarefa de otimização proposta ao AG é a de minimização da energia potencial, de acordo com os parâmetros existentes no campo de força CHARMM, e neste ponto os AGs propostos foram eficientes. O que se mostra com isso é que esta função de otimização não é a mais adequada para a determinação de estruturas protéicas, porém o RMSD não deve ser considerado como função de *fitness,* pois o objetivo final é que estruturas absolutamente desconhecidas possam ser descobertas. Faz-se necessário que um campo de força que modele com maior precisão as complexas interações existentes entre cada um dos átomos de uma molécula.

Além disso, pode ser inserido um pouco mais de conhecimento prévio, de maneira que o algoritmo tenha um ponto de partida menos aleatório e seja mais direcionado, como, por exemplo, com o uso de estruturas homólogas para imigrantes aleatórios, obtidas de alinhamentos realizados pelo BLAST, por exemplo.

Assim, conclui-se que, com uma modelagem mais apropriada do problema, é possível aplicar AGs para o problema de determinação de estruturas de proteínas.

6. CONCLUSÕES

Este trabalho investigou o uso de técnicas de manutenção e aumento da diversidade de populações em Algoritmos Genéticos para o problema de Predição de Estruturas de Proteínas.

Estas técnicas são advindas de *DOPs*, mas se mostraram eficientes para este problema também. De fato, técnicas que permitam que um maior espaço do conjunto de soluções seja explorado são bem-vindas, principalmente em problemas como o de predição de estruturas protéicas, que possui um conjunto NP-completo de soluções, mas apenas uma parte deste conjunto destas soluções é válido, e facilmente a população do AG fica presa em um dos muitos ótimos locais existentes, característicos deste problema.

Primeiramente, investigou-se o uso de bases de dados de ângulos de torção, que se mostraram eficientes em diminuir a energia total da proteína, por considerar apenas combinações de ângulos válidas para cada aminoácido, já que são valores retirados de proteínas cujas estruturas já são conhecidas. Desta base de dados surgiu a primeira contribuição inédita deste trabalho, que é o estudo da efetividade da ordenação destas bases de dados para a melhoria dos indivíduos ao longo das gerações graças à mutação. Viu-se que, em muitos casos, esta ordenação faz diferença, e as mutações ajudam a atingir melhores resultados do que aquelas realizadas na base de dados não ordenada, devido ao fato de que sem a ordenação cada mutação pode modificar completamente os valores dos ângulos de torção, dependendo dos valores dos vizinhos, enquanto uma mutação em uma base ordenada altera os valores para ângulos próximos, possivelmente até sem alterações em um dos ângulos, e esta alteração pode refinar o resultado atingido.

Estas bases de dados foram empregadas em suas duas formas em todos os algoritmos testados, que são o AG padrão, o AG com Hipermutação, o AG com Imigrantes Aleatórios e o AG com Imigrantes Aleatórios Auto-Organizáveis Simplificado. Todos os métodos empregados foram capazes de superar o desempenho apresentado pelo AG padrão, tanto utilizando a base desordenada

quanto a ordenada, o que credita estes métodos para esta aplicação. Infelizmente, a função de *fitness* empregada não se mostrou a mais adequada para que as estruturas sejam preditas com eficiência, no entanto, dentro do que foi proposto, as alterações efetuadas no AG padrão cumpriram com o objetivos propostos. É necessário que uma função de *fitness* mais condizente com as interações atômicas seja implementada, não disponível ainda. Por outro lado, as conclusões atingidas neste trabalho serão úteis quando melhores tecnologias de modelagem de interações protéicas estiverem disponíveis, pois a necessidade de manter uma diversidade na população continuará sendo importante.

Outra contribuição inédita deste trabalho é o AG com Imigrantes Aleatórios Auto-Organizáveis Simplificado, que aproveita características do algoritmo original SORIGA [Tinós & Yang, 2007], mas efetua menos avaliações de indivíduos e não necessita de uma subpopulação, simplificando o algoritmo e aumentado a velocidade de execução do mesmo. Este algoritmo deve ser efetivo também em outros problemas nos quais AGs são úteis para sua resolução, o que deve ser testado no futuro.

Para os testes foram empregadas proteínas já largamente utilizadas na literatura científica, por possuírem características que proporcionam desafios diversos aos algoritmos. Em geral, os resultados são satisfatórios em relação à minimização da energia. No entanto, a estrutura final das proteínas ainda não se apresenta em um patamar satisfatório, o que indica que o conhecimento apresentado para o AG não parece suficiente para que a estrutura completa seja determinada. Assim, propõe-se a investigação de técnicas que adicionem conhecimento ao AG, para que ele possa atingir melhores resultados partindo de alguma informação prévia.

Além disso, há o problema relacionado à escolha da função de *fitness* dos indivíduos. Por um lado o campo de força CHARMM parece adequado, no entanto interações de energia livre não são consideradas por este campo, o que limita o alcance deste método. Campos de força mais completos devem ser considerados, porém o desempenho computacional é reduzido, pelo maior número de operações matemáticas necessárias. Por outro lado, a distância entre os átomos da estrutura real e da estrutura predita podem ser comparados, porém, além de ser um método

computacionalmente custoso, não é aplicável para proteínas cuja estrutura ainda não é conhecida, e o objetivo final da área de pesquisa é obter um algoritmo capaz de realizar predições sem que se saiba previamente a estrutura original.

Outras sugestões de avanço nesta pesquisa são o emprego de outras técnicas de *crossover*, mutação e geração de imigrantes, com o intuito de que estes não sejam totalmente aleatórios, e sim direcionados de acordo com características já sabidas corretas.

Da mesma forma, os AGs apresentados podem ser estudados em suas características principais, e alterações podem ser efetuadas para aumentar ainda mais sua eficiência.

Por fim, nota-se um grande esforço da comunidade científica em resolver este problema, que já vem sendo estudado há muitos anos sem que uma solução definitiva seja alcançada. Espera-se que este trabalho represente um pequeno avanço na área de computação evolutiva, e sua aplicação em predição de estruturas de proteínas, para os quais se demonstrou aplicabilidade, porém ainda com avanços a serem alcançados.

REFERÊNCIAS

[Alberts *et al.*, 2003] Alberts, B.; Bray, D.; Johnson, A.; Lewis, J.; Raff, M.; Roberts, K. & Walter, P., 2003. *Essential Cell Biology* 2nd Edition. Taylor and Francis, New York.

[Alves *et al.*, 1990] Alves, N.A.; Berg, B.A. & Villanova, R., 1990. Ising-model Monte Carlo simulations: Density of states and mass gap. *Physical Review B,* 41, 1:383-394, American Physical Society.

[Anfinsen, 1973] Anfinsen, C.B., 1973. Principles that govern the folding of protein chains. *Science,* 181, 223-230.

[Anile *et al.*, 2006]. Anile, A.M.; Cutello, V.; Narzisi, G.; Nicosia, G. & Spinella, S., 2006. Lipschitzian Pattern Search and Immunological Algorithm with Quasi-Newton Method for the Protein Folding Problem: An Innovative Multistage Approach. *Lecture Notes in Computer Science,* 3931:307-323.

[Bindewald *et al.*, 1998] Bindewald, E.; Hesser, J. & Manner, R., 1998. Implementing genetic algorithms with sterical constrains for protein structure prediction. *Proceedings of International Conference on Parallel Problem Solving from Nature* (PPSN V), pages 959–967, Amsterdam, Netherlands.

[Biswas & Roy, 1995] Biswas, B.B. & Roy, S., 1995. *Proteins: structure, function, and engineering*. Plenum, New York.

[Bledsoe, 1961] Bledsoe, W.W., 1961. The use of biological concepts in the analytical study of systems. *Proceedings of the ORSA-TIMS National Meeting*.

[Blundell & Mizuguchi, 2000] Blundell, T. & Mizuguchi, K., 2000. Structural Genomics: an overview. *Progress Biophysics and Molecular Biology*, 73:289-295.

[Bower *et al.*, 1997] Bower, M.J.; Cohen, F.E. & Dunbrack, R.L., 1997. Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: a new homology-modelling tool. *Journal of Molecular Biology* 267, 5:1268-1282, Elsevier.

[Box, 1957] Box, G.E.P., 1957. Evolutionary operation: a method of increasing industrial productivity. *Applied Statistics,* 6, 81-101.

[Branden & Tooze, 1999] Branden, C.I. & Tooze, J., 1999. *Introduction to Protein Structure*. Garland Pub.

[Brasileiro Filho, 2007] Brasileiro Filho, V.P., 2007. *Algoritmo para Predição de Estruturas Moleculares Protéicas*. Monografia (Bacharelado em Ciências da Computação), Instituto de Computação, Universidade Federal de Alagoas, Maceió, Brasil.

[Bremermann, 1962] Bremermann, H.J., 1962. Optimization through evolution and recombination. *Self-Organizing Systems*, 93-106.

[Brooks *et al.*, 1983] Brooks, B.R.; Bruccoleri, R.E.; Olafson, B.D.; States, D.J.; Swaminathan, S. & Karplus, M., 1983. CHARMM: A Program for Macromolecular Energy, Minimization, and Dynamics Calculations. *Journal of Computational Chemistry,* 4, 187-217.

[Clarke et al, 1994] Clarke, N.D.; Kissinger, C.R.; Desjarlais, J.; Gilliland, G.L. & Pabo, C.O., 1994. Structural studies of the engrailed homeodomain. *Protein Science* 3:1779-1787.

[Cobb & Grefenstette, 1993] Cobb, H. G. & Grefenstette, J. J., 1993. Genetic algorithms for tracking changing environments, *S. Forrest (ed.), 5th International Conference on Genetic Algorithms*, 523-530, Morgan Kaufmann.

[Copeland, 1994] Copeland, R., 1994. *Methods for Protein Analysis – A practical guide to laboratory protocols*. M. Chapman & Hall, New York.

[Cornell *et al.*, 1995] Cornell, W. D.; Cieplak, P.; Bayly, C. I.; Gould, I. R.; Merz, Jr., K. M.; Ferguson, D. M.; Spellmeyer, D. C.; Fox, T.; Caldwell, J. W. & Kollman, P. A., 1995. A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules. *Journal of the American Chemistry Society,* 117, 5179-5197.

[Cutello *et al.*, 2005] Cutello, V.; Narzisi, G. & Nicosia, G., 2005. A class of Pareto archived evolution strategy algorithms using immune inspired operators for ab initio protein structure prediction. *Lecture Notes in Computer Science,* 3449, 54-63, Springer.

[da Silva *et al.*, 2001] da Silva, F.L.B.; Olivares-Rivas, W.; Degrève, L. & Akesson, T., 2001. Application of a new reverse Monte Carlo algorithm to polyatomic molecular systems. I. Liquid water. *The Journal of Chemical Physics,* 114, 2:907-914, AIP.

[Da Silva *et al.*, 2004] da Silva, R.A.; Degreve, L. & Caliri, A., 2004. LMProt: An Efficient Algorithm for Monte Carlo Sampling of Protein Conformational Space. *Biophysical Journal,* 87, 3:1567-1577, Biophysical Soc.

[Dandekar & Argos, 1994] Dandekar, T. & Argos, P., 1994. Folding the main chain of small proteins with the genetic algorithm. *Journal of Molecular Biology*, 3, 236:844-861.

[Dayalan *et al.*, 2005] Dayalan, S.; Bevinakoppa, S. & Schroder, H., 2005. Homology Based Structure Extractor for Protein Structure Prediction. *International Journal of Lateral Computing* 2, 1:56-61, World Federation on Lateral Computing.

[Davis, 1991] Davis, L., 1991. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, NY.

[Darwin, 2004] Darwin, C., 2004. *A Origem das Espécies*. Ediouro, Rio de Janeiro.

[Day *et al.*, 2002] Day, R.O.; Zydallis, J.B.; Lamont, G.B. & Pachter, R., 2002. Solving the protein structure prediction problem through a multiobjective genetic algorithm. *Nanotechnology,* 2, 32-35.

[Deb, 2001] Deb, K., 2001. *Multi-Objective Optimization using Evolutionary Algorithms.* John Wiley and Sons, Chichester.

[Delano, 2002] DeLano, W.L., 2002. *PyMOL User's Manual*. DeLano Scientific, San Carlos, CA.

[Dobzhansky, 1982] Dobzhansky, T. 1982. *Genetics and the Origin of Species*. Columbia University Press.

[Drews, 2000] Drews, J., 2000. Drug discovery: a historical perspective. *Science,* 287, 1960–1964.

[Faccioli, 2007] Faccioli, R.A., 2007. *Algoritmo Híbrido Multi-Objetivo para Predição de Estrutura Terciária de Proteínas.* Dissertação (Mestrado em Engenharia Elétrica) – Programa de Pós-Graduação em Engenharia Elétrica, Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, SP.

[Falcão *et al.*, 2002] Falcão, P.K.; Baudet, C.; Higa, R.H. & Neshich, G., 2002. Incorporação das Propriedades Rotâmeros e Ocupância em Métodos de Análise Estrutural de Proteínas. *Comunicado Técnico,* 34. Embrapa, Campinas/SP.

[Floudas *et al.*, 2006] Floudas, C.A.; Fung, H.K.; McAllister, S.R.; Mönnigmann, M. & Rajgaria, R., 2006. Advances in protein structure prediction and de novo protein design: A review. *Chemical Engineering Science,* 61, 3:966-988.

[Fogel, 1994] Fogel, D., 1994. An Introduction to Simulated Evolutionary Computation. *IEEE Transactions on Neural Networks,* 5:3-14.

[Fukuyama *et al.*, 1996] Fukuyama, Y.; Chiang, H. & Miu, K., 1996. Parallel genetic algorithm for service restoration in electric power distribution systems. International *Journal of Electrical Power and Energy Systems,* 18, 2:111–119.

[Futuyma *et al.*, 2002] Futuyma, D.J.; de Vivo, M. & Sene, F.M., 2002. *Biologia Evolutiva.* Ed. FUNPEC-RP, Ribeirão Preto/SP.

[Ginalski *et al.*, 2005] Ginalski, K.; Grishin, N.V.; Godzik, A. & Rychlewski, W., 2005. Practical lessons from protein structure prediction, *Nucleic Acids Research,* 33, 1874–1891.

[Goldberg, 1989] Goldberg, D.E., 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

[Han & Kambert, 2001] Han, J. & Kambert, M., 2001. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco, CA.

[Herrmann & Suhai, 1995] Herrmann, F. & Suhai, S., 1995. Energy minimization of peptide analogues using genetic algorithms. *Journal of Computational Chemistry*, 16, 11:1434-1444.

[Holland, 1975] Holland, J.H., 1975. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

[Holm & Sander, 1992] Holm, L. & Sander, C., 1992. Fast and simple Monte Carlo algorithm for side-chain optimization in proteins: application to model building by homology. *Proteins: Structure, Function and Genetics,* 14, 213–223.

[Humphrey *et al.*, 1996] Humphrey, W.; Dalke, A. & Schulten, K., 1996. VMD - Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14, 33-38.

[Jannuzzi *et al.*, 2008] Jannuzzi, A. H. L.; Vasconcellos, A.G. & Souza, C. G., 2008. Especificidades do patenteamento no setor farmacêutico: modalidades e aspectos da proteção intelectual. *Cadernos de Saúde Pública,* 24, nº 6. doi 10.1590/S0102-311X2008000600002.

[Jorgensen & Tirado-Rives, 1988] Jorgensen, W.L. & Tirado-Rives, J., 1988. The OPLS Potential Functions for Proteins. Energy Minimizations for Crystals of Cyclic Peptides and Crambin, *Journal of the American Chemistry Society,* 110, 1657-1666.

[Kaiser *et al.*, 1997] Kaiser Jr.; C.E., Lamont; G.B., Merkle; L.D., Gates Jr. & G.H., Patcher, R., 1997. Polypeptide structure prediction: Real-valued versus binary hybrid genetic algorithms. *Proceedings of the ACM Symposium on Applied Computing* (SAC), pages 279–286, San Jose, CA.

[Kihara & Skolnick, 2003] Kihara, D. & Skolnick, J., 2003. The PDB is a covering set of small protein structures. *Journal of Molecular Biology,* 334, 793–802.

[Koehl & Delarue, 1994] Koehl, P. & Delarue, M., 1994. Application of a self-consistent mean field theory to predict protein side-chains conformation and estimate their conformational entropy. *Journal of Molecular Biology,* 239, 249–275.

[Lazaridis & Karplus, 2000] Lazaridis, T. & Karplus, M., 2000. Effective energy functions for protein structure prediction, *Current Opinion in Structural Biology* 10, 2:139-145, DOI: 10.1016/S0959-440X(00)00063-4.

[Lehninger *et al.*, 2005] Lehninger, A.L.; Nelson, D.L. & Cox, M.M., 2005. *Principles of Biochemistry* 4 ed., Freeman, New York.

[Lesk & Lrdk, 2001] Lesk, A.M. & Lrdk, A.M., 2001. *Introduction to protein architecture*. Oxford University Press, New York, NY.

[Lilliefors, 1967] Lilliefors, H.W., 1967. On the Kolmogorov-Smirnov test for normality with mean and variance unknown. *Journal of the American Statistical Association*, 62, 318:399-402, JSTOR.

[Lima, 2006] Lima, T., 2006. *Algoritmos Evolutivos para Predição de Estruturas de Proteínas*. Dissertação (Mestrado em Ciência da Computação e Matemática Computacional) – Programa de Pós-Graduação em Ciência da Computação e Matemática Computacional, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, SP.

[Lima *et al.*, 2007] Lima, T.; Gabriel, P.; Delbem, A.; Faccioli, R. & Silva, I., 2007. Evolutionary algorithm to ab initio protein structure prediction with hydrophobic interactions. *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, 612–619.

[Linden, 2006] Linden, R., 2006. *Algoritmos Genéticos*. Ed. Brasport, Brasil.

[Lodish *et al.*, 2004] Lodish, H.; Berk, A.; Matsudaira, P.; Kaiser, C.A.; Krieger, M. & Scott, M., 2004. *Biologia Celular e Molecular*. Artmed, Brasil.

[MacKerel Jr. *et al.*, 1998] MacKerel Jr., A.D; Brooks III, C.L.; Nilsson, L; Roux, B; Won, Y & Karplus, M., 1998. CHARMM: The Energy Function and Its Parameterization with an Overview of the Program. In *The Encyclopedia of Computational Chemistry,* 1, 271-277, John Wiley & Sons, Chichester.

[Malthus, 1809] Malthus, T.R., 1809. *An Essay on the Principle of Population, as it Affects the Future Improvement of Society*. Disponível em http://books.google.com/books?hl=pt-BR&lr=&id=0sOuqFl0fh8C&oi=fnd&pg=PA9&dq=malthus&ots=87IT0Zn-YQ&sig=kVPeNDCIAj5VkJVv4mJwyCLH8Xc . Acesso em 30/10/2008.

[Mathworks, 1992] Mathworks, 1992. *MATLAB, User's Guide*. The MathWorks, Inc., Natick, MA 01760.

[Mendel, 1865] Mendel, G., 1865. Experiments on plant hybrids. *The origin of genetics: a Mendel sourcebook*, 1-48.

[Merkle *et al.*, 1996] Merkle, L.D.; Gaulke, R.L.; Lamont, G.B.; Gates Jr, G.H. & Pachter, R., 1996. Hybrid genetic algorithms for polypeptide energy minimization. *Proceedings of the 1996 ACM symposium on Applied Computing*, 305-311, ACM New York, NY, USA.

[Michalevicz & Fogel, 2002] Michalevicz, Z. & Fogel, D.B., 2002. *How to Solve It: Modern Heuristics* 1.ed., Springer-Verlag, Berlim, Alemanha.

[Mitchell, 1996] Mitchell, M., 1996. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA.

[MDP, 2008] Médicos de Portugal. 2008. *Glossário.* Disponível em http://medicosdeportugal.saude.sapo.pt/action/10/glo_id/4531/menu/2/. Acesso em 20/12/2008.

[Morrison *et al.*, 2006] Morrison, J.L.; Breitling, R.; Higham, D.J. & Gilbert, D.R., 2006. A lock-and-key model for protein-protein interactions. *Bioinformatics,* 22, 16:2012-2019.

[Morse, 1929] Morse, P. M., 1929. Diatomic molecules according to the wave mechanics. ii. Vibrational levels. *Physical Review,* 34, 1:57-64.

[Moult, 1997] Moult, J., 1997. Comparison of database potentials and molecular mechanics force fields. *Current Opinion in Structural Biology,* 7, 2:194-199, Elsevier.

[Nemethy & Scheraga, 1977] Nemethy, G. & Scheraga, H., 1977. Protein Folding. *Quarterly reviews in Biophysics*, 10:239-352.

[Nicosia & Stracquadanio, 2008] Nicosia, G. & Stracquadanio, G., 2008. Generalized Pattern Search Algorithm for Peptide Structure Prediction. *Biophysical Journal*. doi:10.1529/biophysj.107.124016

[PDBJ, 2008] Protein Data Bank Japan, 2008. *Encyclopedia of Protein Structures*. Disponível em http://eprots.protein.osaka-u.ac.jp/eProtS/Chain.do?from=group&lang=en&pdb_id=1CRN. Acesso em 20/12/2008.

[Pearlman *et al.*, 1995] Pearlman, D.A.; Case, D.A.; Caldwell, J.W.; Ross, W.S.; Cheatham III, T.E.; DeBolt, S.; Ferguson, D.; Seibel, G. & Kollman, P., 1995. AMBER, a Package of Computer Programs for Applying Molecular Mechanics, Normal Mode Analysis, Molecular Dynamics and Free Energy Calculations to Simulate the Structural and Energetic Properties of Molecules, *Computer Physics Communications,* 91, 1-41.

[Pedersen & Moult, 1996] Pedersen, J. & Moult, J., 1996. Genetic algorithms for protein structure prediction. *Current Opinion in Structural Biology,* 6, 2:227–231.

[Pedersen & Moult, 1997] Pedersen, J.T. & Moult; J., 1997. Ab initio protein folding simulations with genetic algorithms: Simulations on the complete sequence of small proteins. *Proteins: Structure, Function and Genetics,* 29, S1:179-184, Wiley-Liss, Inc.

[Petsko & Ringe, 2003] Petsko, G.A. & Ringe, D., 2003. *Protein Structure and Function.* Sinauer Associates, Sunderland, Massachusetts, USA.

[Piccolboni & Mauri, 1998] Piccolboni, A. & Mauri, G., 1998. Application of Evolutionary Algorithms to Protein Folding Prediction. *Lecture Notes in Computer Science*, 1363, 123-136, Springer.

[Pierce & Winfree, 2002] Pierce, N.A. & Winfree, E., 2002. Protein Design is NP-hard. *Protein Engineering*, 15, 10:779-782.

[Ponder *et al.*, 1998] Ponder, J. *et al.*, 1998. *TINKER: Software Tools for Molecular Design*. Department of Biochemistry and Molecular Biophysics, Washington University School of Medicine, St. Louis, MO, 1998.

[Ramachandran & Sasisekharan, 1968] Ramachandran, G.N. & Sasisekharan, V., 1968. Conformation of polypeptides and proteins. *Advances in Protein Chemistry,* 23, 283-438.

[Santana *et al.*, 2008] Santana, R.; Larrañaga, P. & Lozano, J.A., 2008. Protein Folding in Simplified Models With Estimation of Distribution Algorithms. *IEEE Transactions on Evolutionary Computation*, in press.

[Sulloway, 1982] Sulloway, F.J., 1982. Darwin's conversion: The Beagle voyage and its aftermath. *Journal of the History of Biology,* 15, 3:325-396.

[Schulze-Kremer, 1993] Schulze-Kremer, S., 1993. Genetic Algorithms for Protein Tertiary Structure Prediction. *Lecture Notes in Computer Science*: *Machine Learning: ECML-93*, 262-279.

[Schulze-Kremer & Tiedemann, 1994] Schulze-Kremer, S. & Tiedemann, U., 1994. Parameterizing genetic algorithms for protein folding simulation. System Sciences, 1994. *Proceedings of the Twenty-Seventh Hawaii International Conference on Biotechnology Computing*, 5, 345:354.

[Sheik *et al.*, 2003] Sheik, S.S.; Ananthalakshmi, P.; Bhargavi, G.R. & Sekar, K., 2003. CADB: Conformation Angles DataBase of proteins. *Nucleic Acids Research,* 31, 1:448-451, Oxford University Press.

[Sheik *et al.*, 2005] Mohan, S.; Sheik, S.S.; Ramesh, J.; Balamurugan, B.; Jeyasimhan, M.; Mayilarasi, C. & Sekar, K., 2005. CADB-2.0: Conformation Angles Database. *Biological Crystallography,* D61, 637-639.

[Schwyzer, 1995] Schwyzer, R., 1995. 100 Years lock-and-key concept: Are peptide keys shaped and guided to their receptors by the target cell membrane? *Biopolymers,* 37, 1:5-16. John Wiley & Sons, Inc.

[Snow *et al.*, 2002] Snow, C.D.; Nguyen, H.; Pande, V.S. & Gruebele, M., 2002. Absolute comparison of simulated and experimental protein-folding dynamics. *Nature,* 420, 6911:102-106.

[Taniguchi *et al.*, 1999] Taniguchi, E.; Noritake, M.; Yamada, T. & Izumitani, T., 1999. Optimal size and location planning of public logistics terminals. *Transportation Research Part E,* 35, 3:207–222.

[Tinós & Yang, 2007] Tinós, R. & Yang, S., 2007. A self-organizing random immigrants genetic algorithm for dynamic optimization problems. *Genetic Programming and Evolvable Machines,* 8, 3:255-286, Springer Netherlands.

[Tragante & Tinós, 2007] Tragante, V. & Tinós, R, 2007. Strategies to increase solution variability in Protein Folding in Silico via Genetic Algorithms. In: X-Meeting 2007, São Paulo.

[Tragante & Tinós, 2008] Tragante, V. & Tinós, R, 2008. Impact of Database Sorting on the Efficiency of Genetic Algorithms in Protein Structure Prediction. In: VIII BIOMAT International Symposium on Mathematical and Computational Biology, 2008, Campos do Jordão.

[Unger & Moult, 1993] Unger, R. & Moult, J., 1993. On the applicability of genetic algorithms to protein folding. In Trevor N.Mudge, Veljko Milutionovic, and Lawrence Hunter, editors, *Proceedings of the 26th Hawaii International Conference on Systems Science (HICSS-26),* volume 1, pages 715–725, Wailea, HI, 5.-8. January 1993. IEEE Computer Society press, Los Alamitos, CA.

[Vavak & Fogarty, 1996] Vavak, F. & Fogarty, T.C., 1996. A comparative study of steady state and generational genetic algorithms for use in nonstationary environments. T. C. Fogarty (editor), AISB Workshop on Evolutionary Computing, *Lecture Notes in Computer Science*, 1143, 297–304, Springer.

[Verli, 2008] Verli, H., 2008. *Bioinformática Estrutural.* Slides de aula. Disponível em http://www.cbiot.ufrgs.br/bioinfo/SAEF_03.pdf. Acesso em 19/01/2009.

[Wilcoxon *et al.*, 1963] Wilcoxon, F.; Katti, S.K. & Wilcox, R.A., 1963. Critical Values and Probability Levels for the Wilcoxon Rank Sum Test and the Wilcoxon Signed Rank Test. *Selected Tables in Mathematical Statistics I*, 171-260, Providence, RI, American Mathematics Society.

[Yang & Honavar, 1998] Yang, J. & Honavar, V., 1998. Feature Subset Selection Using a Genetic Algorithm. *IEEE Intelligent Systems* 13, 2:44-49.

APÊNDICE A – CABEÇALHO DO CAMPO DE FORÇA CHARMM

```
    ##############################
    ##                          ##
    ##  Force Field Definition  ##
    ##                          ##
    ##############################

forcefield            CHARMM27

vdwtype               LENNARD-JONES
radiusrule            ARITHMETIC
radiustype            R-MIN
radiussize            RADIUS
epsilonrule           GEOMETRIC
vdw-14-scale            1.0
chg-14-scale            1.0
dielectric            78.7

    ##############################
    ##                          ##
    ##  Literature References  ##
    ##                          ##
    ##############################

A. D. MacKerrell, Jr., et al., "All-Atom Empirical Potential for
Molecular Modeling and Dynamics Studies of Proteins", J. Phys. Chem. B,
102, 3586-3616 (1998)

N. Foloppe and A. D. MacKerell, Jr., "All-Atom Empirical Force Field
for Nucleic Acids: I. Parameter Optimization Based on Small Molecule
and Condensed Phase Macromolecular Target Data", J. Comput. Chem.,
21, 86-104 (2000)

Current parameter values are available from the CHARMM parameter site
in Alex MacKerell's lab at UMBC, http://www.pharmacy.ab.umd.edu/~alex/

    ##############################
    ##                          ##
    ##  Atom Type Definitions  ##
    ##                          ##
    ##############################

  ######################################################
  ##                                  ##
  ##  TINKER Atom Class Numbers to CHARMM Atom Names  ##
  ##                                  ##
  ##    1  HA     11  CA     21  CY     31  NR3   ##
  ##    2  HP     12  CC     22  CPT    32  NY    ##
  ##    3  H      13  CT1    23  CT     33  NC2   ##
  ##    4  HB     14  CT2    24  NH1    34  O     ##
  ##    5  HC     15  CT3    25  NH2    35  OH1   ##
  ##    6  HR1    16  CP1    26  NH3    36  OC    ##
  ##    7  HR2    17  CP2    27  N      37  S     ##
  ##    8  HR3    18  CP3    28  NP     38  SM    ##
  ##    9  HS     19  CH1    29  NR1            ##
  ##   10  C      20  CH2    30  NR2            ##
  ##                                  ##
  ######################################################

atom     1    1   HA     "Nonpolar Hydrogen"        1     1.008     1
atom     2    2   HP     "Aromatic Hydrogen"        1     1.008     1
atom     3    3   H      "Peptide Amide HN"         1     1.008     1
atom     4    4   HB     "Peptide HCA"              1     1.008     1
```

APÊNDICE B – EXEMPLO DE ARQUIVO GERADO PELOS AGS

arquivo.dat
Crambin (THR-THR-CYS-CYS-PRO-SER-ILE-VAL-ALA-ARG-SER-ASN-PHE-ASN-VAL-CYS-ARG-LEU-PRO-GLY-THR-PRO-GLU-ALA-ILE-CYS-ALA-THR-TYR-THR-GLY-CYS-ILE-ILE-ILE-PRO-GLY-ALA-THR-CYS-PRO-GLY-ASP-TYR-ALA-ASN)
THR -70.9 169.6 180 62.5
THR -80.2 -19.2 180 62.5
CYS -78 142 180 -67.5
CYS -123 152.5 180 -67.5
PRO -63.3 157.1 180
SER -78.6 -27.1 180 -62.5
ILE -60.6 -44.4 180 -57.5 -62.5
VAL -121.2 118.8 180 177.5
ALA -134.9 133.8 180
ARG -122.8 165.8 180 72.5 -67.5 -172.5 -77.5 180
SER -156.1 151.6 180 -62.5
ASN -111.3 25.8 180 -67.5 -37.5
PHE -64.2 -44.1 180 -72.5 172.5
ASN -97.9 170.8 180 -67.5 -37.5
VAL -108.4 134.3 180 -62.5
CYS -126.6 131.1 180 -67.5
ARG -49.4 -43.9 180 -177.5 177.5 177.5 177.5 180
LEU -50.9 122 180 -62.5 177.5
PRO -72.9 -7.7 180
GLY -53.8 -46.6 180
THR -121.9 119.8 180 62.5
PRO -55.4 134.9 180
GLU -68.2 -44.3 180 -67.5 177.5 177.5
ALA -140.8 160.3 180
ILE -111.1 125.2 180 -62.5 87.5
CYS -97.7 134.5 180 -67.5
ALA -75.3 147 180
THR -105.7 139.7 180 -62.5
TYR -130.7 163.2 180 -67.5 82.5
THR -86.1 -158.7 180 62.5
GLY 107.5 -8.5 180
CYS -125.9 160.4 180 62.5
ILE -126.7 147.9 180 -172.5 167.5
ILE -113.9 112 180 -57.5 -62.5
ILE -113 105.1 180 -62.5 172.5
PRO -67.4 142.1 180
GLY 84.1 -175.4 180
ALA -107.9 13 180
THR -112.7 116.2 180 -62.5
CYS -93.5 156.4 180 -67.5
PRO -55.1 144.1 180
GLY -62.3 -44.5 180
ASP -59.4 -23.6 180 -72.5 167.5
TYR -95.4 -6 180 177.5 77.5
ALA -92.5 -6.9 180
ASN -106.5 119.5 180 -67.5 -37.5

n

APÊNDICE C – EXEMPLO DE ARQUIVO XYZ

```
  648  Crambin  (THR-THR-CYS-CYS-PRO-SER-ILE-VAL-ALA-ARG-SER-ASN-PHE-ASN-VAL-CYS-ARG-LEU-
PRO-GLY-THR-PRO-GLU-ALA-ILE-CYS-ALA- THR-GLY-CYS-ILE-ILE-ILE-PRO-GLY-ALA-THR-CYS-PRO-GLY-
ASP-TYR-ALA-ASN)
    1  N3     0.000000    0.000000    0.000000   146     2     5     6     7
    2  CT     0.000000    0.000000    1.500000   156     1     3     8     9
    3  C      1.412520    0.000000    2.033747    82     2     4    17
    4  O      2.383438   -0.185743    1.318751    83     3
    5  H3     0.470538   -0.838490   -0.340483   151     1
    6  H3     0.516043    0.811278   -0.340483   151     1
    7  H3    -0.961330   -0.017785   -0.340483   151     1
    8  HC    -0.504720    0.916552    1.870526     6     2
    9  CT    -0.743787   -1.246644    2.014063    20     2    10    11    12
   10  OH    -0.030429   -2.379021    1.570279    16     9    13
   11  CT    -2.148716   -1.302498    1.385848     1     9    14    15    16
   12  HC    -0.711491   -1.263905    3.123458     6     9
   13  HO     0.706765   -2.062813    1.080221    17    10
   14  HC    -2.692520   -2.198615    1.751023     6    11
   15  HC    -2.068308   -1.354310    0.279978     6    11
   16  HC    -2.723893   -0.393808    1.660721     6    11
   17  N      1.466465    0.223158    3.353933    85     3    18    21
   18  CT     2.744633    0.261204    4.058516    74    17    19    22    23
   19  C      3.891015    0.538835    3.115738    82    18    20    31
   20  O      5.003253    0.064724    3.278670    83    19
   21  H      0.587702    0.365645    3.851803    88    17
   22  HC     2.724702    1.069705    4.818798     6    18
   23  CT     2.979143   -1.083809    4.770938    20    18    24    25    26
   24  OH     1.951918   -1.243844    5.723460    16    23    27
   25  CT     4.318301   -1.036890    5.529921     1    23    28    29    30
   26  HC     2.868615   -1.912179    4.040399     6    23
   27  HO     1.402637   -0.483398    5.663365    17    24
   28  HC     4.495738   -2.003002    6.046878     6    25
   29  HC     4.299298   -0.222007    6.283382     6    25
   30  HC     5.150617   -0.848671    4.820044     6    25
   31  N      3.545225    1.348325    2.105414    85    19    32    35
   32  CT     4.516151    1.735752    1.086197    74    31    33    36    37
   33  C      4.143842    3.045285    0.433045    82    32    34    42
   34  O      3.096839    3.623470    0.673644    83    33
   35  H      2.583292    1.685696    2.069833    88    31
   36  HC     5.515039    1.859734    1.554119     6    32
   37  CT     4.564962    0.658455   -0.013188    59    32    38    39    40
   38  SH     5.269693   -0.867953    0.683830    52    37    41
   39  HC     5.157631    1.035389   -0.872702     6    37
   40  HC     3.543858    0.498046   -0.417805     6    37
   41  HS     5.456938   -0.384305    1.919396    56    38
   42  N      5.080067    3.482090   -0.420351    85    33    43    46
   43  CT     4.905212    4.733012   -1.152621    74    42    44    47    48
   44  C      3.665493    5.470163   -0.705624    82    43    45    53
   45  O      3.698992    6.370335    0.117156    83    44
   46  H      5.918549    2.914366   -0.542978    88    42
   47  HC     5.778756    5.393516   -0.971639     6    43
   48  CT     4.757915    4.432462   -2.655809    59    43    49    50    51
   49  SH     6.338218    3.788999   -3.289052    52    48    52
   50  HC     4.441629    5.354429   -3.186883     6    48
   51  HC     3.925459    3.714239   -2.808362     6    48
   52  HS     6.993086    3.852314   -2.121688    56    49
   53  N      2.554111    5.025573   -1.307929    86    44    54    60
   54  CT     1.250901    5.612012   -1.009046    93    53    55    57    58
...
```

APÊNDICE D – EXEMPLO DE BASE ORDENADA – ALANINA

```
27265                   //número de combinações de ângulos existentes no arquivo
-179.75  108.98         //ângulo phi, seguido pelo ângulo psi
-179.63  63.14
-179.35  -171.6
-179.02  166.94
-178.99  170.01
-178.96  19.62
-178.85  149.1
-178.79  163.29
-178.7   172.11
-178.63  159.85
-178.56  179.67
-178.52  166.32
-178.44  146.24
-178.33  166.47
-178.16  169.16
-178.15  168.12
-178.13  158.94
-177.95  133.77
-177.27  152.04
-177.22  160.14
-177.18  176.3
-177.15  129.09
-177.14  163.93
-177.12  -4.85
-177.08  -142.2
-177.05  -89.16
-177.04  163.44
-177.01  145.92
-176.93  169.45
-176.89  156.18
-176.87  136.19
-176.66  172.85
-176.44  107.8
-176.38  156.51
-176.38  171.41
-176.35  170.74
-176.11  151.98
-176.09  158.86
-175.92  159.13
-175.92  163.02
-175.86  163.42
-175.65   42.27
-175.55  155.44
-175.54  131.61
-175.45  165.73
-175.41  145.71
-175.32  152.37
-175.13  159.17
-175.04   -30.96
-174.9    -122.45
-174.89  174.93
-174.86  132.64
-174.84  156.68
-174.67  146.8
-174.59  151.2
-174.56  171.79
-174.54  170.37
-174.53  149.37
-174.35  163.92
-174.33  154.17
```

APÊNDICE E – EX. DE BASE DESORDENADA – ALANINA

```
27265                    //número de combinações de ângulos existentes no arquivo
100.04   -70.15          //ângulo phi, seguido pelo ângulo psi
-152.23  150.49
-73.06   -51.50
-75.69   -43.93
-60.17   -48.56
-62.67   -34.09
-58.00   -57.05
-63.38   -20.31
-50.70   136.00
-52.33   -30.80
-86.50   -31.28
-66.20   -55.45
-75.37   -35.92
-67.44   -37.62
-60.94   -24.27
-53.16   -27.32
-53.70   -16.75
-61.64   -61.41
-58.53   -44.28
-95.40   -27.52
-55.20   -46.83
-91.73   -50.84
150.62   150.94
-169.31  -53.53
-58.09   -50.91
-52.50   -19.80
-79.86   144.46
-110.70  131.36
-126.75  128.24
-73.40   138.95
-72.34   152.68
-64.78   113.20
-94.08   -49.12
-45.98   -51.69
-47.82   -34.04
-59.78   -17.01
-87.71   130.29
-84.88   158.73
-105.50  12.04
-79.38   131.46
-84.60   -7.71
-76.76   -28.74
-65.03   -34.98
-86.22   164.46
-66.67   -34.45
-59.52   160.23
-106.66  152.15
-138.25  170.81
-59.29   -52.20
-53.06   -45.85
-54.86   -14.18
-103.67  167.27
-59.06   140.52
-60.93   129.02
-60.40   -31.81
-69.88   165.61
-46.07   -48.67
-74.34   -28.56
-69.12   -48.70
...
```