UNIVERSIDADE DE SÃO PAULO
FFCLRP - DEPARTAMENTO DE FÍSICA E MATEMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA APLICADA À
MEDICINA E BIOLOGIA

Identificação de domínios em proteínas com redes complexas

LUIZ CARLOS BÜTTNER MOSTAÇO-GUIDOLIN

Tese apresentada à Faculdade de Filosofia,
Ciências e Letras de Ribeirão Preto da USP,
como parte das exigências para a obtenção
do título de Doutor em Ciências.

Área: Física Aplicada à Medicina e Biologia.

Ribeirão Preto - SP
2011

UNIVERSIDADE DE SÃO PAULO
FFCLRP - DEPARTAMENTO DE FÍSICA E MATEMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA APLICADA À
MEDICINA E BIOLOGIA

Identificação de domínios em proteínas com redes complexas

LUIZ CARLOS BÜTTNER MOSTAÇO-GUIDOLIN

Orientador: Prof. Dr. Nelson Augusto Alves

Tese apresentada à Faculdade de Filosofia,
Ciências e Letras de Ribeirão Preto da USP,
como parte das exigências para a obtenção
do título de Doutor em Ciências.

Área: Física Aplicada à Medicina e Biologia.

Ribeirão Preto - SP
2011

FICHA CATALOGRÁFICA

Guidolin, Luiz Carlos Büttner Mostaço

Identificação de domínios em proteínas com redes complexas.
Ribeirão Preto, 2011.

235p. : il. ; 30cm

Tese de Doutorado apresentada à Faculdade de Filosofia
Ciências e Letras de Ribeirão Preto - Universidade de São Paulo.
Área de concentração: Física Aplicada a Medicina e Biologia.

Orientador: Nelson Augusto Alves.

1 - redes complexas 2 - domínios em proteínas
3 - otimização extrema 4 - modularidade

Folha de Aprovação

Luiz Carlos Büttner Mostaço-Guidolin
Identificação de domínios em proteínas com redes complexas

Tese apresentada à Faculdade de Filosofia,
Ciências e Letras de Ribeirão Preto da USP,
como parte das exigências para a obtenção
do título de Doutor em Ciências.

Área: Física Aplicada à Medicina e Biologia.

Aprovado em:

Banca Examinadora:

Prof. Dr.

Instituição: Assinatura:

Prof. Dr.

Instituição: Assinatura:

Prof. Dr.

Instituição: Assinatura:

Prof. Dr.

Instituição: Assinatura:

Prof. Dr.

Instituição: Assinatura:

Universidade de São Paulo
Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto

Identificação de domínios em proteínas com redes complexas

LUIZ CARLOS BÜTTNER MOSTAÇO-GUIDOLIN

À memória de meu
pai, Benedito, à minha
mãe, Dirce, aos meus
irmãos Pedro e Branca
e ao meu grande amor,
Leila.

Copyright © 2011 by Luiz Carlos Büttner Mostaço-Guidolin
Alguns Direitos Reservados

A presente obra encontra-se licenciada sob a licença *Creative Commons Attribution-ShareAlike 2.5 Brazil*. Para visualizar uma cópia da licença, envie uma carta para Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA ou visite <http://creativecommons.org/licenses/by-sa/2.5/br/>.

Atribuição – Compartilhamento pela mesma Licença

Você pode:

Compartilhar – copiar, distribuir, exibir e executar a obra

Modificar – criar obras derivadas

Sob as seguintes condições:

Atribuição – Você deve dar crédito ao autor original, da forma especificada pelo autor ou licenciante (mas sem sugerir que eles aprovam você ou seu uso da obra).

Compartilhamento pela mesma Licença – Se você alterar, transformar, ou criar outra obra com base nesta, você somente poderá distribuir a obra resultante sob uma Licença idêntica.

Compreendendo que:

Renúncia – Qualquer uma destas condições podem ser renunciadas, desde que você obtenha permissão do detentor dos direitos autorais.

Uso Legítimo – A licença não interfere nas limitações do direito de autor, tais como para a reflexão acadêmica e exercício da liberdade de expressão.

Direitos Morais – De nenhuma forma a licença afeta os direitos morais do autor.

Notificação – Para cada novo uso ou distribuição, você deve deixar claro para outros os termos da licença desta obra. A melhor maneira de fazê-lo é com um apontador para a página <http://creativecommons.org/licenses/by-sa/2.5/br/>.

Resumo

A utilização de redes complexas para a descrição de diversos sistemas naturais e artificiais, compreendidos nas mais diversas áreas do conhecimento humano, tem se mostrado uma abordagem poderosa para a redução da complexidade inerente a tais sistemas. Em muitos casos, tal complexidade resulta do número de componentes envolvidos e de suas intrincadas relações. Uma forma de reduzir a complexidade associada a tais sistemas, consiste em identificar e agrupar componentes que possuam características similares. Sendo assim, desenvolvemos nesta tese métodos de identificação de comunidades em redes complexas. Tais métodos se baseiam na ideia de que comunidades surgem quando grupos de vértices possuem um número mais elevado de conexões entre os vértices do mesmo grupo do que com vértices externos à este grupo. Além disso, utilizamos a função modularidade como função objetivo e como forma de avaliação e comparação dos resultados obtidos nesta tese com resultados previamente reportados na literatura. Uma vez estabelecido um método de identificação de comunidades, utilizamos a abordagem de redes complexas para a determinação de domínios estruturais de proteínas. Para tal, criamos redes de contato entre os aminoácidos de uma proteína buscando representar apenas as ligações relevantes do ponto de vista topológico. Por meio destas representações, aplicamos os métodos de identificação de comunidades desenvolvidos nesta tese, no intuito de identificar domínios estruturais de cadeias proteicas. Por fim, desenvolvemos um método específico para a identificação de domínios em proteínas com dois domínios sequencias, concluindo desta forma, os objetivos propostos nesta tese.

Palavras-chave: redes complexas, domínios em proteínas, otimização extrema, modularidade.

Abstract

The use of complex networks for the representation of various natural and artificial systems in the most diverse fields of human knowledge, has proven to be a powerful approach for the reduction of the complexity in the study of such systems. In many cases, this complexity emerges from the number of components of the system and from the intricate relationship between them. A reduction in this complexity is made possible by the identification and grouping of the components of the system with similar characteristics. In this way, we developed in this thesis, methods for community identification in complex networks. Such methods are based on the notion that communities arise when groups of vertices are more densely connected with vertices of their same group, than with vertices belonging to other groups in the network. Moreover, the modularity function has been used as an objective function, and as a score for the evaluation and comparison of the results obtained in this thesis with the results reported in the literature of complex networks. Upon the establishment of a method for community detection, we used the framework of complex networks to the determination of structural protein domains. Therefore, we have created contact networks of amino acids of protein chains, focusing on the representation of only the most relevant interactions between them, from a topological point of view. We have applied to these networks the methods for community identification developed in this thesis, aiming to identify the structural domains of proteins. Finally, we have developed a specific method for the identification of protein domains in protein chains with two sequential domains, concluding in this way, the objectives proposed in this thesis.

Key words: complex networks, protein domains, extremal optimization, modularity.

Sumário

Sumário	iv
Lista de Figuras	vii
Lista de Tabelas	xiii
Lista de Algoritmos	xvi
1 Introdução	1
1.1 Estrutura desta tese	3
1.2 Aplicativos desenvolvidos	4
2 Motivações e objetivos	7
3 Redes complexas	9
3.1 Descrição de redes complexas	9
3.2 Distribuição dos graus de uma rede	11
3.2.1 Correlação do grau e assortatividade	12
3.3 Caminhadas aleatórias	13
3.4 Modelos de redes complexas	13
3.5 Comunidades em redes complexas	16
3.5.1 Modularidade	18
3.5.2 Métodos de identificação de comunidades	20
4 Proteínas	23
4.1 Proteínas: base biológica	23
4.1.1 Composição e formação das proteínas	24
4.1.2 Estrutura das proteínas	27
4.1.3 Conformação proteica: estrutura secundária	28
4.1.4 A estrutura α -hélice	29
4.1.5 A estrutura folha- β	31
4.1.6 Conformação proteica: estrutura terciária	31
4.1.7 Domínios	32
4.1.8 Domínios como modelos evolucionários	33
4.2 PDB – <i>Protein Data Bank</i>	33
4.3 Métodos de classificação de domínios de proteínas	35
4.3.1 CATH – <i>Class, Architecture, Topology, Homology</i>	35
4.3.2 SCOP – <i>Structural Classification of Proteins</i>	37
4.3.3 DP - <i>Domain Parser</i>	39

5	Identificação de comunidades em redes complexas	43
5.1	O método AL	43
5.2	Agrupamento aglomerativo hierárquico	48
5.2.1	HAC++	49
5.3	Implementação	52
5.3.1	graphstk – <i>The Graphs Toolkit</i>	53
5.3.2	HAC++	58
5.4	Testes de desempenho	62
5.5	Aplicação em redes sociais	67
6	Otimização extrema generalizada	73
6.1	Otimização extrema generalizada	77
6.2	Implementação	80
6.3	Descrição experimental	83
6.3.1	Experimento 1	87
6.3.2	Experimento 2	92
6.3.3	Experimento 3	97
6.3.4	Experimento 4	102
6.4	Aplicação em redes sociais	103
7	ALGO – Método AL otimizado	105
7.1	Implementação	108
7.2	Aplicação em redes sociais	111
7.2.1	Experimento 1	112
7.2.2	Experimento 2	117
7.2.3	Experimento 3	121
8	Identificação de domínios em proteínas	125
8.1	Transcrição de proteínas em redes complexas com o cvtpdb	129
8.2	Proteínas como redes de contato	130
8.2.1	Estudo do <i>backbone</i> proteico a partir da matriz adjacência	133
8.3	Propriedades das redes proteicas	138
8.4	Identificação de comunidades com o método ALGO	146
8.5	Identificação de comunidades em proteínas com dois domínios	152
9	Conclusões e perspectivas	161
	Apêndices	169
A	Parâmetros calculados para as redes das cadeias de proteínas	169
A.1	Grau dos vértices	169
A.2	Assortatividade	171
A.3	Modularidade	176
A.4	Domínios identificados	178
A.4.1	$C_\alpha : R_c = 7,0\text{\AA}$	178
A.4.2	$C_\alpha : R_c = 7,5\text{\AA}$	180
A.4.3	$C_\alpha : R_c = 8,0\text{\AA}$	182
A.4.4	DP	183
A.4.5	CR	185

B <i>The GNU General Public License</i>	187
Referências Bibliográficas	195
Glossário	203

Lista de Figuras

3.1	Exemplos de redes com 8 vértices e 10 arestas. Em (a) temos um grafo simples (não direcionado e não ponderado), em (b) um grafo direcionado e não ponderado e em (c) um grafo não direcionado mas ponderado. . . .	10
3.2	Coefficiente de <i>clustering</i> do vértice representado por um quadrado, nos casos: (a) não há conexões entre os vizinhos, (b) os vizinhos estão conectados, sendo que os vértices dentro da região delimitada pela linha pontilhada formam um clique com 4 vértices e (c) corresponde a um grafo completo.	15
3.3	Redes complexas com $N = 200$ vértices e grau médio $\langle k \rangle = 2$. Em (a) uma rede aleatória, em (b) uma rede livre de escala com <i>hubs</i> representados por vértices de forma quadrada.	16
3.4	Exemplo de uma rede simples com três comunidades, na qual o vértice destacado pelo sinal de interrogação é de difícil classificação uma vez que o mesmo está conectado a duas comunidades pelo mesmo número de arestas.	18
4.1	Estrutura básica dos aminoácidos. Estes são compostos por um C_α , no qual um hidrogênio, um grupo amina e um grupo carboxila são ligados. Além destes elementos, há uma cadeia lateral para cada aminoácido. A cadeia lateral é a responsável pela identidade de cada um dos 20 tipos de aminoácidos existentes.	24
4.2	Diagrama esquemático da cadeia polipeptídica, também denominada esqueleto (ou <i>backbone</i>) das proteínas.	26
4.3	Ilustração de uma hemácia sadia e de uma hemácia falciforme. Esta deformação é causada pela alteração de um único aminoácido ao longo da cadeia peptídica (imagem adaptada: http://uofmchildrenshospital.org).	26
4.4	A mioglobina foi a primeira proteína a ter sua estrutura tridimensional revelada. Em 1958 John Kendrew e seus colegas determinaram a estrutura da mesma através da cristalografia por raios-X de alta resolução.	28
4.5	Esquema ilustrativo da estrutura α -hélice. Esta estrutura encontra-se presente na estrutura secundária dos níveis de organização das proteínas, possuindo o esqueleto polipeptídico estreitamente enrolado ao longo do maior eixo da molécula e tendo os grupos R dos resíduos de aminoácidos projetados para fora do esqueleto helicoidal. Fonte: http://ftp.unb.br/pub/UNB/cbsp/Download/Imagens/300_dpis/Biomoleculas-300/prot/ . . .	30
4.6	A estrutura <i>folha-β</i> é formada a partir da combinação de diversas regiões da cadeia polipeptídica, contrastando com a α -hélice, que é formada a partir de uma região contínua. Fonte: http://www.structuralbioinformatics.com/chapter2.shtml	31

4.7	Crescimento do número total de estruturas na base de dados PDB-RCSB (fonte: http://www.pdb.org/pdb/statistics/).	34
4.8	Representação esquemática da (C)lasse, (A)rquitetura e (T)opologia – níveis de agrupamento da base de dados CATH. Fonte: http://koehllab.genomecenter.ucdavis.edu/teaching/ecs129/07	37
4.9	Esquema da hierarquia SCOP. Todos os níveis principais, incluindo classe, enovelamento, super-família, família e domínio são apresentados. Também são mostradas as proteínas individuais e com o menor nível, sendo o PDB a referência das entradas. (Steven E. Brenner, 1994)	38
4.10	Exemplos de atribuições de domínios pelo método DomainParser.	40
5.1	Dendrograma obtido para a rede do clube de karatê usando a escala de dissimilaridades fornecida pelo método AL. Esta rede é composta por 34 vértices e 78 arestas.	45
5.2	A máxima modularidade obtida pelo método AL para a rede do clube de karatê revela 3 comunidades tendo $Q = 0,399$. As três comunidades são definidas por meio da forma geométrica utilizada para representar os vértices. As linhas pontilhadas são utilizadas para mostrar a divisão em duas comunidades como determinada pelo método AL e considerada como verdadeira por Zachary.	47
5.3	Representação esquemática da estrutura de dados desenvolvida para utilização no algoritmo de agrupamento hierárquico – HAC++.	49
5.4	Representação esquemática da situação da estruturas de dados após ocorrer a primeira aglomeração.	50
5.5	Representação esquemática da estrutura de dados na situação final do processo aglomerativo.	52
5.6	Grafo exemplo, contendo um subgrafo com estrutura de árvore (vértices 5, 6 e 7).	55
5.7	Grafo exemplo, no qual o subgrafo com estrutura de árvore foi retirado (vértices 5, 6 e 7), originando o <i>multivértice</i> 5*.	57
5.8	Grafo exemplo, no qual a folha (<i>multivértice</i> 5*) foi retirada, originando o novo <i>multivértice</i> 3*.	57
5.9	Avaliação da complexidade e tempo de processamento das funções membro da classe <code>ALGraphs_C</code> , necessárias para a implementação do método AL (Figs. 5.9a–5.9i) e 5.9j método de agrupamento aglomerativo.	64
5.10	Fração do tempo utilizado no processamento de cada componente do método AL + HAC++, listados na Tab. 5.1.	66
5.11	Avaliação da complexidade e tempo de processamento total do método AL, incluindo a rotina de agrupamento aglomerativo.	66
5.12	Avaliação da complexidade e tempo de processamento da função <code>hclust</code> , fornecida pelo ambiente R, em comparação com o método HAC++, desenvolvido neste trabalho.	67
5.13	Máxima modularidade obtida pelo método AL para rede a rede do clube de karatê, revelando 3 comunidades e $Q = 0,399$	69
5.14	Máxima modularidade obtida pelos método VP, para a rede do clube de karatê, revelando 4 comunidades e $Q = 0,420$	69
5.15	Dendrograma da rede do clube de karatê obtido pelo método AL no qual a linha tracejada apresenta a região de corte para se obter 4 comunidades.	70

6.1	Ciclo básico de algoritmos evolucionários. (Adaptação da ref. [115], pág. 96).	76
6.2	Rede do clube de karatê com 34 vértices, 78 arestas, dividida em 4 comunidades correspondente à sua máxima modularidade $Q = 0,41979$.	83
6.3	Distribuição do número de vértices n em uma comunidade, para o conjunto das $N_{div} = 500$ divisões (pseudo-aleatórias) da rede do clube de karatê. Estas divisões são usadas como ponto de partida para determinação da modularidade máxima utilizando o método τ -EO.	86
6.4	Modularidade máxima obtida a partir da rede inicial número 111, em cada uma das $N_{exec} = 500$ execuções do método τ -EO (configuração do experimento 1), para a rede do clube de karatê.	87
6.5	Distribuição dos valores da modularidade máxima obtidos em cada uma das $N_{rep} = 250\ 000$ repetições do método τ -EO (configuração do experimento 1) para a rede do clube de karatê.	89
6.6	Frequência de ocorrência dos os valores máximos obtidos para a função modularidade, considerando apenas o valor máximo obtido para cada série de $N_{exec} = 500$ execuções da rotina de otimização τ -EO (configuração do experimento 1), para cada uma das $N_{div} = 500$ divisões iniciais da rede do clube de karatê.	89
6.7	Distribuição do número de comunidades associadas aos valores máximos da modularidade obtidas pelo método τ -EO (configuração do experimento 1) para a rede do clube de karatê.	90
6.8	Distribuição do número de ocorrências do valor máximo em cada série de $N_{exec} = 500$ execução do método τ -EO (configuração do execução 1) para cada uma das $N_{div} = 500$ divisões iniciais da rede do clube de karatê.	91
6.9	Histograma correspondente ao número da execução na qual o máximo valor da modularidade foi obtido, sendo que este não necessariamente corresponde a Q_{max} , mas sim, ao máximo obtido para a dada divisão inicial da rede do clube de karatê.	92
6.10	Modularidade máxima obtida a partir da rede inicial número 111, em cada uma das $N_{exec} = 500$ execuções do método τ -EO, com troca de vértices pelo método de Boettcher e função objetivo local de Duch e Arenas, para a rede do clube de karatê.	93
6.11	Distribuição dos valores da modularidade obtidos em cada uma das $N_{rep} = 250\ 000$ execuções do método τ -EO (configuração do experimento 2) para a rede do clube de karatê.	94
6.12	Frequência de ocorrência dos os valores máximos obtidos para a função modularidade, considerando apenas o valor máximo obtido para cada série de $N_{exec} = 500$ execuções da rotina de otimização τ -EO (configuração do experimento 2), para cada uma das $N_{div} = 500$ divisões iniciais da rede do clube de karatê.	95
6.13	Distribuição do número de comunidades associadas aos valores máximos da modularidade obtidas pelo método τ -EO (configuração do experimento 2) para a rede do clube de karatê.	95
6.14	Distribuição do número de ocorrências do valor máximo em cada série de $N_{exec} = 500$ execuções do método τ -EO (configuração do execução 2) para cada uma das $N_{div} = 500$ divisões iniciais da rede do clube de karatê.	96

6.15	Histograma correspondente ao número da execução na qual o máximo valor da modularidade foi obtido, sendo que este não necessariamente corresponde a $Q_{\max}$, mas sim, ao máximo obtido para a dada divisão inicial da rede do clube de karatê.	97
6.16	Modularidade máxima obtida a partir da rede inicial número 111, em cada uma das $N_{\text{exec}} = 500$ execuções do método τ -EO (configuração do experimento 3), para a rede do clube de karatê.	98
6.17	Distribuição dos valores da modularidade obtidos em cada uma das $N_{\text{rep}} = 250\,000$ execuções do método τ -EO (configuração do experimento 3) para a rede do clube de karatê.	99
6.18	Frequência de ocorrência dos valores máximos obtidos para a função modularidade, considerando apenas o valor máximo obtido para cada série de $N_{\text{exec}} = 500$ execuções da rotina de otimização τ -EO (configuração do experimento 3), para cada uma das $N_{\text{div}} = 500$ divisões iniciais da rede do clube de karatê.	100
6.19	Distribuição do número de comunidades associadas aos valores máximos da modularidade obtidas pelo método τ -EO (configuração do experimento 3) para a rede do clube de karatê.	101
6.20	Histograma correspondente ao número da execução na qual o máximo valor da modularidade foi obtido, sendo que este não necessariamente corresponde a $Q_{\max}$, mas sim, ao máximo obtido para a dada divisão inicial da rede do clube de karatê.	101
7.1	Diagrama esquemático do processo de otimização, proposto nesta tese para determinação da divisão ótima de uma rede em comunidades.	107
7.2	Modularidade máxima obtida em cada uma das 500 execuções do método ALGO para a rede do clube de karatê.	114
7.3	Fração dos valores máximos obtidos pelo método ALGO para a função modularidade, calculada para a rede do clube de karatê em uma série de 500 execuções.	115
7.4	Tempo de processamento utilizado para executar apenas a rotina de otimização do método ALGO para a rede do clube de karatê, considerando o número de comunidades envolvidas em cada processo, para a configuração ALGO($3N, 3, 1000, 25$).	116
7.5	Avaliação do desempenho da rotina de otimização do método ALGO com configuração ALGO($3N, 3, 1000, 25$) para as redes sociais avaliadas.	116
7.6	Modularidade máxima obtida em cada uma das 500 execuções do método ALGO, para a rede do clube de karatê.	119
7.7	Fração dos valores máximos obtidos pelo método ALGO para a função modularidade calculada na rede do clube de karatê em uma série de $N_{\text{exec}} = 500$ execuções.	120
7.8	Tempo de processamento utilizado para executar apenas a rotina de otimização do método ALGO para a rede do clube de karatê, considerando o número de comunidades envolvidas em cada processo, com a configuração ALGO($3N, 3, 1000, 25$).	120
7.9	Avaliação do desempenho da rotina de otimização do método ALGO com configuração ALGO($3N, 3, 1000, 25$) para as redes sociais utilizadas.	121

7.10	Otimização da função modularidade pelo método ALGO para as redes sociais analisadas nesta tese para as divisões iniciais com 2 a 13 comunidades fornecidas pelo método AL	123
8.1	Representação sob a forma de rede complexa da proteína PDB ID 2SNV-A , utilizando distâncias entre C_α e raio de corte $R_c = 8,0\text{\AA}$. Esta rede contém 151 vértices e 737 arestas.	128
8.2	Representação sob a forma de rede complexa da proteína PDB ID 2SNV-A , utilizando o critério de interação entre resíduos com $I_c = 4,0\%$. Esta rede contém 151 vértices e 511 arestas.	128
8.3	Distribuição do número de cadeias de proteínas disponíveis na base de dados CATH em função do número de domínios.	131
8.4	Distribuição do número de aminoácidos em cada uma das 56 cadeias de proteínas analisadas nesta tese.	132
8.5	Exemplo de (a) grafo não direcionado e não ponderado, composto por 10 vértices e 12 arestas; (b) matriz adjacência do respectivo grafo; (c) representação alternativa da matriz adjacência, onde cada pixel preenchido corresponde a um valor não nulo da matriz.	133
8.6	Imagem representativa das matrizes adjacência das redes: (a) clube de karatê, (b) Golfinhos, (c) Football, (d) PolBooks, (e) Jazz e (f) LesMis.	135
8.7	Matrizes adjacência referente à proteína 1AD0-B , cuja transcrição para redes complexas foi efetuada utilizando os critérios de distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).	136
8.8	Matrizes adjacência referente à proteína 1A6C-A , cuja transcrição para redes complexas foi efetuada utilizando os critérios de distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).	137
8.9	Distribuição do grau dos vértices para as redes da proteína 1AD0-B , utilizando as redes geradas utilizando os critérios de distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).	139
8.10	Distribuição dos valores médios dos graus dos vértices obtidos para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para cada critério de criação das redes.	140
8.11	Distribuição dos valores de graus máximos dos vértices obtidos para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para cada critério de criação das redes.	141
8.12	Distribuição dos valores de assortatividade obtidos para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para cada critério de criação das redes.	143
8.13	Distribuição dos valores de assortatividade obtidos para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para cada critério de criação das redes.	144
8.14	Comparação da assortatividade para os conjuntos de proteínas com diferentes números de domínios.	145

8.15	Diagrama ilustrativo do procedimento desenvolvido para identificação de comunidades em proteínas com dois domínios sequenciais. Dada uma cadeia polipeptídica com N aminoácidos, explora-se todas as divisões possíveis em duas comunidades sequenciais simplesmente quebrando esta cadeia em duas partes e calculando-se o valor da função modularidade Q referente a tal quebra.	152
8.16	Valores de modularidade obtidos para as possíveis divisões em dois domínios contínuos da rede referente à proteína 1AD0-B, utilizando as redes geradas com os critérios de distâncias entre C_α com raios de corte $R_c = 7, 0, 7, 5$ e $8, 0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).	154
8.17	Valores de modularidade obtidos para as possíveis divisões em dois domínios contínuos da rede referente à proteína 1A6C-A, utilizando as redes geradas com os critérios de distâncias entre C_α com raios de corte $R_c = 7, 0, 7, 5$ e $8, 0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).	155
8.18	Valores de modularidade obtidos para as possíveis divisões em dois domínios contínuos da rede referente à proteína 3FKS-A, utilizando a rede gerada utilizando os critério de distâncias entre C_α com raio de corte $R_c = 7, 0\text{\AA}$	156
8.19	Sobreposição das curvas de modularidade para identificação de domínios para as proteínas 1AD0-B e 1A6C-A.	157
8.20	Distribuição dos valores de máxima modularidade correspondente às divisões em dois domínios para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para os diferentes critérios de construção das redes.	158
8.21	Distribuição dos valores de erros relativos correspondente aos erros cometidos na determinação do ponto de divisão das redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para os diferentes critérios de construção das redes.	159

Lista de Tabelas

4.1	Relação das 20 cadeias laterais com seus respectivos códigos de uma e três letras, classificadas de acordo com as categorias: hidrofóbicas, polares e carregadas.	25
5.1	Componentes do método AL avaliadas em termos de desempenho.	65
5.2	Relação das redes avaliadas nesta tese, com o nome pelo qual são referenciadas no texto, número de vértices e número de arestas, bem como nome pelo qual são referenciadas na literatura e os artigos de referência.	68
5.3	Máxima modularidade calculada para diversas redes sociais, pelos métodos propostos por Alves [14] (AL), Agarwal e Kempe [23] (VP), Liu e Murata [24] (LPAm+), Girvan e Newman [39] (GN), e por Guimerà e Amaral [40] (SA). Entre colchetes são apresentados os números de comunidades obtidas por cada um dos métodos.	68
6.1	Máxima modularidade calculada para diversas redes sociais, pelos métodos propostos por Alves [14] (AL), Boettcher e Percus [137] (τ -EO), Agarwal e Kempe [23] (VP), Liu e Murata [24] (LPAm+), Girvan e Newman [39] (GN) e por Guimerà e Amaral [40] (SA). Entre colchetes são apresentados os números de comunidades identificadas por cada um dos métodos.	103
7.1	Máxima modularidade calculada para diversas redes sociais, pelos métodos propostos por Alves [14] (AL), Alves e Guidolin (ALGO), Boettcher e Percus [137] (τ -EO), Agarwal e Kempe [23] (VP), Liu e Murata [24] (LPAm+), Girvan e Newman [39] (GN) e por Guimerà e Amaral [40] (SA). Entre colchetes são apresentados os números de comunidades obtidas por cada um dos métodos.	112
7.2	Máxima modularidade determinada para diversas redes sociais, pelo método ALGO com as configurações dos experimentos 1 e 2. Q é o valor da modularidade com o correspondente número de comunidades entre colchetes; Mult. corresponde à multiplicidade, ou seja, número de vezes que o resultado foi obtido e Repet. corresponde ao número da repetição no qual o máximo foi determinado pela primeira vez.	117
7.3	Máxima modularidade determinada para diversas redes sociais, pelo método τ -EO e pelo método ALGO, sendo Q o valor da modularidade (com correspondente número de comunidades entre colchetes), Mult. corresponde à multiplicidade, ou seja, número de vezes que o resultado foi obtido e, Repet. corresponde ao número da repetição no qual o máximo foi determinado pela primeira vez. O método τ -EO foi executado 250 000 vezes e o método ALGO foi executado apenas 500 vezes.	118

8.1	Fatores de normalização N_i utilizados para determinar a interação entre resíduos.	127
8.2	Relação dos PDB ID das 56 cadeias de proteínas analisadas nesta tese. . .	132
8.3	Nível descritivo p obtido para os testes de hipótese ANOVA e Kruskal-Wallis (KW) entre os conjuntos com i e j domínios.	146
8.4	Divisões em comunidades para a proteína 1EZM-A obtidas pelos métodos AL e ALGO para os cinco critérios de transcrição das proteínas em redes complexas, são eles: distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, DP e CR. Na coluna AL apresentamos a divisão em duas comunidades obtidas pelo método AL, na coluna ALGO apresentamos a divisão em duas comunidades obtidas pelo método ALGO, na coluna $Q_{\max}$, apresentamos o valor máximo de modularidade e, na coluna Divisão $\rightarrow Q_{\max}$, apresentamos a divisão da rede correspondente a $Q_{\max}$	150
8.5	Divisões em comunidades para a proteína 1D5M-A obtidas pelos métodos AL e ALGO para os cinco critérios de transcrição das proteínas em redes complexas, são eles: distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, DP e CR. Na coluna AL apresentamos a divisão em duas comunidades obtidas pelo método AL, na coluna ALGO apresentamos a divisão em duas comunidades obtidas pelo método ALGO, na coluna $Q_{\max}$, apresentamos o valor máximo de modularidade e, na coluna Divisão $\rightarrow Q_{\max}$, apresentamos a divisão da rede correspondente a $Q_{\max}$	151
A.1	Grau médio e máximo das 56 cadeias de proteínas utilizadas nesta tese, sendo $\langle k \rangle$ o grau médio e $k_{\max}$ o grau máximo identificado para cada rede criada segundo os critérios de transcrição da proteína de acordo com as distâncias entre C_α utilizando raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, de acordo com o critério do Domain Parser com $R_c = 4,0\text{\AA}$ e critério de contato com $I_c \leq 4\%$	170
A.2	Valores de assortatividade para as 56 cadeias de proteínas utilizadas nesta tese, para cada rede criada a partir dos critérios de distâncias entre C_α , com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, do Domain Parser com $R_c = 4,0\text{\AA}$ e critério de contato entre resíduos com $I_c \leq 4\%$	172
A.3	Relação da 90 proteínas com um domínio, para as quais foi efetuado o cálculo da assortatividade da rede correspondente.	173
A.4	Relação da 90 proteínas com dois domínios, para as quais foi efetuado o cálculo da assortatividade da rede correspondente.	174
A.5	Relação da 90 proteínas com três domínios, para as quais foi efetuado o cálculo da assortatividade da rede correspondente.	175
A.6	Relação da 90 proteínas com quatro domínios, para as quais foi efetuado o cálculo da assortatividade da rede correspondente.	176
A.7	Valores das modularidades máximas para as 56 cadeias de proteínas utilizadas nesta tese, para cada rede criada a partir dos critérios de distâncias entre C_α , com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, de acordo com o critério do Domain Parser com $R_c = 4,0\text{\AA}$ e critério de contato entre resíduos com $I_c \leq 4\%$	178

A.8	Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério $C_{\alpha} : R_c = 7,0\text{\AA}$; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).	180
A.9	Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério $C_{\alpha} : R_c = 7,5\text{\AA}$; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).	181
A.10	Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério $C_{\alpha} : R_c = 8,0\text{\AA}$; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).	183
A.11	Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério DP; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).	185
A.12	Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério CR; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).	186

Lista de Algoritmos

5.1	Declaração da classe <code>Graphs_C</code> em C++.	54
5.2	Declaração da classe <code>ALGraphs_C</code> em C++.	56
5.3	Procedimento do metodo <code>AL</code> .	58
5.4	Declaração da classe <code>Clusters_C</code> em C++.	59
5.5	Declaração do método <code>SeekCommunities()</code> em C++.	61
6.1	Código fonte em C++ da rotina de otimização cerne do método τ -EO.	81
6.2	Script em shell, <code>gen500Membership.sh</code> , para geração de $N_{\text{div}} = 500$ divisões pseudo-aleatórias de uma rede complexa em duas comunidades.	85
7.1	Declaração da rotina de otimização do método <code>ALGO</code> em C++.	109

Capítulo 1

Introdução

Uma das propriedades mais proeminentes observadas em diversos sistemas naturais e artificiais é a sua complexidade. Um simples ser vivo unicelular, como uma bactéria, é o resultado de inúmeros processos bioquímicos inter-relacionados por uma intrincada cadeia de eventos, as quais tornam possível a sua existência. O entendimento deste sistema constitui um grande desafio à capacidade humana.

O estudo das propriedades de sistemas físicos envolvendo um elevado número de constituintes, é uma tarefa muitas vezes inviável, a qual nos força a recorrer à simplificações que viabilizem seu estudo. Esta complexidade, entretanto, não está restrita a sistemas naturais.

A *world wide web*, por exemplo, contém mais de 10^{11} páginas mapeadas que possuem referências de umas para as outras dando origem à grande rede mundial [1]. Este número representa apenas uma fração da internet, uma vez que a *world wide web* corresponde à apenas uma parte dos protocolos de comunicação utilizados na internet. Por outro lado, com mais de 6 bilhões de seres humanos habitando o planeta Terra, a rede de relacionamentos sociais entre indivíduos também pode ser entendida como um sistema de complexidade elevada.

Os sistemas citados no parágrafo anterior são apenas alguns exemplos cuja complexidade apresenta desafios com relação ao seu entendimento. Entretanto, estes podem ser traduzidos em sistemas compostos por unidades nas quais a presença ou ausência de interação entre as mesmas surge como uma das características fundamentais de tais sistemas. Esta propriedade nos remete às origens das *redes complexas*, as quais constituem uma ferramenta capaz de lidar com tal complexidade em inúmeros problemas.

As unidades básicas das redes complexas são os chamados vértices, os quais podem ser conectados por meio de arestas. Arestas são adicionadas, conectando dois vértices, quando desejamos representar algum tipo de relacionamento entre eles. Por exemplo, nas chamadas redes sociais, os vértices representam pessoas e as arestas representam algum tipo de relacionamento social, como a amizade. Em proteínas os vértices são utilizados

para representar os aminoácidos e as arestas são utilizadas para representar suas ligações químicas.

A teoria de redes complexas utiliza o formalismo da *teoria de grafos* a fim de matematizar sua descrição. A origem da teoria de grafos, por sua vez é atribuída a Leonhard Euler, cujo formalismo foi desenvolvido para solucionar o famoso problema das pontes de Königsberg [2] em 1736. A cidade de Königsberg, atualmente Kaliningrad, ocupa as duas margens do rio Pregel, e uma ilha existente neste rio. Todas as partes da cidade que são divididas pelo rio, são conectadas por 7 pontes. O problema envolvia determinar a possibilidade de passar por todas as 7 pontes apenas uma vez e retornar para casa. Euler provou matematicamente que isto era impossível.

Nos dois séculos seguintes, o estudo de redes foi conduzido apenas por matemáticos, culminando no trabalho de Paul Erdős e Alfréd Rényi. Eles definiram o conceito de grafos aleatórios e cunharam a teoria de grafos “moderna” [3].

Nas últimas duas décadas, o interesse pela teoria dos grafos foi renovada em áreas como Ciência da Computação, Física, Ciências Sociais, dentre outras, principalmente por causa do crescimento da capacidade computacional, bem como da disponibilidade de dados experimentais.

Em particular, ferramentas de Física Estatística originalmente desenvolvidas para a área de Física da Matéria Condensada, mostraram-se bem sucedidas na descrição de diversas características observadas em tais redes. Exemplos destas características são a distribuição de graus, coeficientes de aglomeração e centralidade [4, 5, 6]. Estas características, dentre outras, vêm sendo utilizadas para caracterizar redes complexas. Além disso, esforços para categorizar os vários tipos de sistemas, resultou em esforços para o desenvolvimento de modelos gerais de redes complexas. Dois destes modelos são o modelo de redes de mundo pequeno [4] e redes livres de escala [7].

Por outro lado, a análise de sistemas cujo número de constituintes seja superior a 10^5 ainda é um desafio. Mesmo com o poder computacional disponível atualmente, os algoritmos atuais escalam com potências da ordem do tamanho do sistema. Sendo assim, procura-se reduzir o tamanho do sistema verificando-se se há um número de entidades similares o suficiente, tal que seja possível considerá-las como sendo apenas uma. Neste ponto, surge a necessidade do desenvolvimento de métodos de detecção de comunidades em redes complexas, tal que o procedimento de diminuição da complexidade possa ser efetuado automaticamente.

Como exemplo, proteínas podem ser representadas por meio de redes complexas considerando-se que cada aminoácido corresponde a um vértice e as forças de interação entre estes aminoácidos são representadas por meio das arestas. Através desta abordagem, a complexidade do problema que antes consistia em utilizar todos átomos da proteína é reduzida, tornando possível a identificação de comunidades em tais redes. Esta identificação tem o potencial de revelar os domínios estruturais de tal proteína, os

quais são responsáveis pelas funções da mesma, desempenhando um papel importante no seu enovelamento.

Considerando-se o poder da modelagem de sistemas através de redes complexas e da importância biológica relacionada a caracterização de proteínas, o método proposto neste trabalho baseia-se na representação de uma proteína por meio de uma rede complexa, com o objetivo de determinar a melhor divisão em comunidades para a dada rede.

1.1 Estrutura desta tese

Esta tese está dividida em 9 capítulos, sendo que no Cap. 2 são apresentados os objetivos desta tese. No Cap. 3 é feita uma breve revisão dos conceitos envolvendo redes complexas e métodos de identificação de comunidades. Neste capítulo são abordados apenas os conceitos estritamente relacionados aos assuntos abordados no restante desta tese. No Cap. 4, é feita uma breve introdução a respeito de proteínas, focando em sua estrutura, uma vez que a identificação de domínios estruturais é um dos objetivos desta tese. Ainda neste capítulo, apresentamos a base de dados de estruturas de proteínas (PDB), além de alguns métodos de classificação e identificação de domínios das mesmas. Tais métodos foram utilizados como referência para validação dos resultados obtidos nesta tese.

No Cap. 5 apresentamos um método de identificação de comunidades em redes complexas, denominado **AL** bem como um método para agrupamento aglomerativo hierárquico, desenvolvido nesta tese e denominado **HAC++**. Estes métodos são aplicados a diferentes redes sociais e seu custo computacional foi avaliado experimentalmente. No intuito de estabelecer uma estratégia de otimização dos resultados fornecidos pelo método **AL**, apresentamos no Cap. 6 as implementações de dois métodos de otimização, baseados na heurística conhecida como otimização extrema generalizada. Combinando características destes métodos, duas novas abordagens híbridas são propostas. Tais abordagens são aplicadas à identificação de comunidades em redes sociais para a avaliação dos mesmos. Com isso, a estratégia de otimização a ser empregada no método **AL** ficou estabelecida.

No Cap. 7 o método de detecção de comunidades **AL** é integrado ao método de agrupamento aglomerativo **HAC++**. Os resultados fornecidos por estes métodos são então otimizados pelo método denominado **ALGO**, o qual também foi desenvolvido nesta tese. Este último, baseia-se no método de otimização extrema generalizada híbrida e em algoritmos genéticos. O método **ALGO** foi aplicado ao problema de identificação de domínios em redes complexas, sendo que sua validação foi efetuada em diversas redes sociais. No Cap. 8 foram construídas redes complexas que representem a estrutura de proteínas, dando origem a redes de contatos entre aminoácidos. Estas redes foram avaliadas segundo critérios comumente utilizados, tais como a distribuição dos graus dos vértices e assortatividade. Uma vez que as redes proteicas foram caracterizadas,

aplicamos o método **ALGO** na identificação de domínios estruturais de proteínas. Ainda neste capítulo, propomos um novo método de identificação de domínios estruturais de proteínas usando redes complexas, o qual pode ser aplicado a proteínas com dois domínios sequenciais. Finalmente, são apresentadas no Cap. 9 as conclusões gerais e perspectivas para trabalhos futuros baseados nos resultados obtidos nesta tese.

1.2 Aplicativos desenvolvidos

Desenvolvemos nesta tese uma série de aplicativos os quais foram registrados e disponibilizados como software livre e de código aberto, sob a licença GPLv3 sendo que instruções para *download* encontram-se na seguinte URL: <http://atlas.ffclrp.usp.br>. Dentre tais aplicativos, temos o conjunto de ferramentas desenvolvidas para a manipulação de redes complexas, desenvolvidas na linguagem C++, o qual chamamos de **graphstk** (Sec. 5.3.1). Os principais componentes do **graphstk** compreendem:

- classes para a manipulação de grafos;
- mecanismos para geração de arquivos DOT (usados na representação gráfica de grafos);
- o método de detecção de comunidades **AL** (Cap. 5);
- o método de agrupamento aglomerativo hierárquico **HAC++** (Sec. 5.2.1);
- o método de otimização **ALGO** (Cap. 7);
- o método de identificação de comunidades em redes complexas baseado no procedimento de otimização extrema generalizada, chamado τ -EO híbrido (Cap. 6).

O método **AL** foi implementado com base em seu artigo original. Nesta implementação buscamos aprimorar seu desempenho computacional e torná-lo mais portátil, facilitando assim sua distribuição. O método τ -EO é uma implementação baseada em dois trabalhos, porém com alterações que permitiram obter resultados melhores do que aqueles reportados nos artigos originais; por este motivo, o chamamos de método τ -EO híbrido. Por outro lado, os métodos **HAC++** e **ALGO** são métodos originais desenvolvidos nesta tese.

Outro aplicativo desenvolvido nesta tese, chamado **cvtpdb** (Sec. 8.1), escrito na linguagem Python, é destinado à manipulação de arquivos de estruturas de proteínas disponíveis na base de dados PDB. Sua funcionalidade principal corresponde à tradução das informações estruturais das proteínas em redes complexas. Neste aplicativo foram implementados diversos critérios de transcrição, como o critério de distâncias entre C_α , distâncias entre centros de massa de resíduos, o critério adotado pelo método **Domain**

Parser e o critério de contatos entre resíduos. Além destas possíveis conversões, o `cvtpdb` efetua o cálculo do raio de giração de proteínas e gera arquivos DOT para a representação gráfica das mesmas sob a forma de grafos.

Além destes aplicativos, foi desenvolvido um método para a identificação de domínios em proteínas com dois domínios sequenciais, o qual é apresentado na Sec. 8.5. Maiores detalhes a respeito destes aplicativos, bem como citações pertinentes, são fornecidos nas respectivas seções.

Capítulo 2

Motivações e objetivos

O avanço do poder computacional das últimas duas décadas impulsionou consigo a área de redes complexas, pois sistemas antes virtualmente impossíveis de serem tratados tornaram-se viáveis de serem manipulados. Tal progresso propiciou também o surgimento de bases de dados de proporções gigantescas, tanto em termos do número de registros quanto ao nível de detalhamento dos mesmos. Isto fez com que novos problemas surgissem, os quais novamente beiram a intratabilidade.

Abordar sistemas com um grande número de componentes inter-relacionados por meio de redes complexas, permite capturar a essência de tais relacionamentos e reescrever tais problemas em termos mais simples, permitindo assim sua análise. Em alguns casos, dado o elevado número de componentes envolvidos, a simplificação advinda da representação por meio de redes complexas pode não ser suficiente.

Neste ponto, diversos trabalhos vêm surgindo no intuito de minimizar ainda mais a complexidade inerente a tais sistemas; uma forma de redução passível de ser aplicada consiste na identificação de comunidades em redes complexas.

Nesta tese efetuamos a implementação de um método destinado à determinação de comunidades. Este método foi proposto por Alves e é aqui denominado por **AL**. Muitos dos métodos mais atuais e que podem ser considerados como sendo o *estado da arte* na detecção de comunidades, estão baseados na determinação de particionamentos das redes, as quais forneçam o maior valor possível para a chamada função modularidade.

Em trabalhos anteriores, foi demonstrado que o método **AL** possui potencial para ser empregado com sucesso em diversas redes; porém, este sucesso não estará completo se tal método não for capaz de fornecer divisões da rede compatíveis com aquelas obtidas a partir de outras abordagens propostas, já bem aceitas na área.

Vale ressaltar porém, que muitos destes métodos partem de divisões arbitrárias da rede, buscando cegamente o particionamento da mesma. Tal busca visa simplesmente atingir o valor máximo da função modularidade. No entanto, a função modularidade possui limitações e nem sempre fornece a divisão considerada verdadeira para as mais

diversas redes.

Considerando-se que o método **AL** fornece, em alguns casos, as divisões consideradas verdadeiras e que estas muitas vezes estão próximas às soluções de máxima modularidade, tivemos como primeiro objetivo desta tese, desenvolver um método de otimização capaz de utilizar as divisões fornecidas pelo método **AL**. A partir destas divisões, visamos ainda obter uma que maximizasse a função modularidade.

No caso de sistemas biológicos, como proteínas, o número de átomos envolvidos pode ser da ordem de milhares. Se formos considerar as interações entre todos os átomos, o sistema resultante será inquestionavelmente difícil de ser tratado. Por outro lado, diversos átomos podem ser agrupados em aminoácidos. Alternativamente podemos representar as proteínas por meio de seus aminoácidos, o que reduz o número de componentes a serem tratados.

Além disso, considerando-se as distâncias as quais ocorrem interação entre tais aminoácidos, podemos criar redes complexas que representem a estrutura de ligações entre os mesmos. Esta representação pode, por sua vez, fornecer informações a respeito da estrutura da proteína e potencialmente possibilitar a identificação de seus domínios estruturais. Este problema, torna-se ainda mais relevante considerando-se que atualmente há aproximadamente 69 655 estruturas de proteínas mapeadas; a determinação de seus domínios é de suma importância, não só para a determinação das funções de tal proteína, mas também para estudos envolvendo seu enovelamento. Sendo assim, como segundo objetivo desta tese, buscamos aplicar a abordagem de redes complexas à modelagem de proteínas. Com a finalidade de extrair suas características topológicas essenciais, através do método **AL** com otimização da função modularidade, buscamos a identificação de domínios estruturais em proteínas.

Capítulo 3

Redes complexas

A descrição, caracterização e detecção de comunidades em redes complexas requer a introdução de alguns parâmetros estatísticos destas redes. Muitos destes conceitos vêm da teoria de grafos e mais recentemente de áreas como Física Estatística, a partir das quais um grande número de ferramentas, modelos e algoritmos foram originados. Ao invés de fornecer uma descrição detalhada de redes complexas, indicamos ao leitor interessado as seguintes referências: [3, 6, 8, 9, 10], as quais fornecem uma visão mais abrangente e completa a respeito de redes complexas.

3.1 Descrição de redes complexas

Uma rede complexa $G(V, E)$ é definida como sendo o conjunto de vértices V , conectados por meio das arestas (ou arcos) do conjunto E . Esta definição é equivalente àquela encontrada em textos matemáticos a respeito de teoria de grafos. A designação “rede complexa” é utilizada para se referir a sistemas naturais ou artificiais que possuem características que os torna complexos de serem analisados, e não simplesmente às entidades abstratas da teoria de grafos.

Em uma rede complexa, os vértices correspondem aos constituintes de interesse do sistema o qual estamos representado. Já as arestas, as quais conectam um vértice a outro, representam o relacionamento entre tais vértices. Sendo assim, uma rede complexa é completamente caracterizada pelo conhecimento de seus vértices e arestas. Uma forma de escrever uma rede é simplesmente enumerar seus vértices e listar suas arestas. Esta descrição, entretanto, é pouco prática para manipulações matemáticas. Sendo assim, as redes são geralmente descritas por meio de uma matriz, a qual chamamos de matriz adjacência $\mathbf{A}$. Nesta matriz, cada elemento a_{ij} denota o valor assumido pela aresta que conecta os vértices i e j . Em geral, $a_{ij} \neq 0$ denota a presença de uma aresta entre os vértices i e j , ao passo que $a_{ij} = 0$ indica a sua ausência. Consequentemente, podemos distinguir tipos de redes simplesmente por meio dos elementos a_{ij} . Há três tipos básicos

de redes que podemos reconhecer, são elas: redes simples, redes ponderadas e redes direcionadas. Representações sob a forma de grafos de cada um destes tipos de rede são mostradas na Fig. 3.1.

Figura 3.1: Exemplos de redes com 8 vértices e 10 arestas. Em (a) temos um grafo simples (não direcionado e não ponderado), em (b) um grafo direcionado e não ponderado e em (c) um grafo não direcionado mas ponderado.

Nesta figura, podemos observar ainda, a existência de um vértice que não está conectado a nenhum outro vértice. Nesta situação, dizemos que este grafo é não conexo. Para que um grafo seja conexo, para quaisquer dois vértices i e j é necessário existir pelo menos um caminho de i para j .

A matriz adjacência de uma rede simples é simétrica e assume valores binários. Ou seja, esta matriz é composta por 0 e 1, indicando a ausência ou presença de uma aresta entre os vértices.

O grau k_i do vértice i é definido como sendo o número de arestas conectadas a i

$$k_i = \sum_{j=1}^N a_{ij}, \quad (3.1)$$

sendo N o número total de vértices na rede. Como em uma rede simples, não há distinção se a aresta vai do vértice i para o vértice j ou do vértice j para o vértice i , a matriz adjacência é simétrica.

Contrariamente a uma rede simples, em uma rede ponderada as arestas podem assumir diferentes valores, os quais correspondem ao peso, w_{ij} , atribuído à relação entre os vértices i e j . Em geral redes ponderadas são utilizadas para representar interações entre vértices com diferentes níveis de intensidade. Desta forma, definimos como peso de um vértice, a soma dos pesos de suas arestas; ou seja,

$$w_i = \sum_{j=1}^N w_{ij}. \quad (3.2)$$

Em uma rede ponderada o grau k_i de um vértice i é simplesmente definido como

sendo o número de arestas que um vértice possui, sem incluir seus pesos. Apesar de ser menos descritivo do que o peso do vértice, o grau k_i indica o número de vértices com os quais o vértice i está relacionado.

Redes direcionadas, por sua vez, são utilizadas para casos em que a relação entre os vértices não são simétricas. Ou seja, o vértice i pode estar relacionado ao vértice j , porém o vértice j pode não estar relacionado ao vértice i . Por este motivo, em redes direcionadas a matriz adjacência, em geral, não é simétrica. Além disso, podemos ter redes que são simultaneamente ponderadas e direcionadas.

Como em redes direcionadas há distinção entre o sentido das relações entre os vértices, definimos então o grau de entrada de um vértice como sendo

$$k_i^{\text{in}} = \sum_{j=1}^N a_{ij}, \quad (3.3)$$

ao passo que o grau de saída é dado por

$$k_i^{\text{out}} = \sum_{j=1}^N a_{ij}. \quad (3.4)$$

Em geral, não utilizamos redes nas quais haja vértices com laços, ou seja, vértices que possuem arestas conectando-os a si mesmos; sendo assim, teremos $a_{ij} = 0$ para $i = j$. Desta forma, todos os elementos da diagonal principal da matriz adjacência serão nulos.

Considerando-se um grafo simples e sem laços, o número máximo de arestas possíveis em tal grafo, corresponde a

$$M_{\text{máx}} = \frac{N(N-1)}{2}. \quad (3.5)$$

No caso em que o número de arestas M corresponde a $M_{\text{máx}}$, dizemos que o grafo é completo. Em grafos direcionados, como há distinção entre as arestas de i para j e de j para i , o número máximo de arestas corresponde a $M_{\text{máx}} = N(N-1)$.

3.2 Distribuição dos graus de uma rede

Enquanto que o grau de um vértice é uma quantidade local que caracteriza o número de conexões que um vértice possui, a distribuição dos graus de uma rede é uma propriedade global. Historicamente, o trabalho de Barabási e Albert [7] atribui à distribuição do grau um papel importante no estudo de redes complexas. Se $N(k)$ corresponde ao número de vértices com grau k em uma rede, a distribuição dos graus será

$$P(k) = \frac{N(k)}{N}, \quad (3.6)$$

logo, o grau médio de uma rede será dado por

$$\langle k \rangle = \sum_{k=1}^{k_{\text{máx}}} kP(k), \quad (3.7)$$

sendo que a flutuação do grau acerca do valor médio é dada por

$$\langle k^2 \rangle - \langle k \rangle^2 = \sum_{k=1}^{k_{\text{máx}}} k^2 P(k) - \left[\sum_{k=1}^{k_{\text{máx}}} kP(k) \right]^2. \quad (3.8)$$

3.2.1 Correlação do grau e assortatividade

Outro aspecto importante para a caracterização de redes complexas é a determinação de como os vértices com valores elevados de grau se conectam. Tem sido mostrado em redes sociais, que vértices com alto grau tendem a se conectar a vértices também com alto grau. Em outros tipos de redes, tais como redes tecnológicas, vértices com elevados graus podem estar conectados a vértices com menores valores de grau [11]. Sendo assim, a correlação entre os graus é dada pelo grau médio dos seus primeiros vizinhos ($k_{nn}(k)$),

$$k_{nn}(k)(k) = \frac{1}{kN(k)} \sum_{i=1}^N \sum_{j=1}^{k_i} k_{i \rightarrow j} \delta_{k_i, k}, \quad (3.9)$$

na qual $k_{i \rightarrow j}$ é o grau do j -ésimo vizinho do vértice i . Sendo assim, definindo $P(k'|k)$ como sendo a probabilidade condicional de uma aresta pertencendo a um vértice com grau k , estar conectada a um vértice com grau k' , temos então, que a média dos graus dos primeiros vizinhos do vértice com grau k será dada por

$$k_{nn}(k) = \sum_{k'} P(k'|k). \quad (3.10)$$

Dessa forma, se $k_{nn}(k)$ for uniforme, a rede é não correlacionada. Entretanto, se $k_{nn}(k)$ cresce em função de k , implica que os vértices com graus elevados possuem, em média, vizinhos também com graus elevados. Neste caso, dizemos que a rede possui grau de correlação positivo. Contrariamente, esta rede terá um grau de correlação negativo se $k_{nn}(k)$ decrescer em função de k .

Complementarmente ao comportamento de $k_{nn}(k)$, a correlação em uma rede pode ser estudada através do coeficiente de assortatividade (r), [12]. Escolhendo-se aleatoriamente uma aresta, a distribuição do grau de um vértice no final desta aresta (não considerando a aresta pela qual chegamos) será dada por

$$e(k) = \frac{(k+1)P(k+1)}{\sum_j jP(j)}. \quad (3.11)$$

Escrevendo $e(k', k)$ para a distribuição do grau de ambos os vértices nas extremidades

da aresta escolhida (ainda desconsiderando tal aresta), a assortatividade será definida como

$$r = \frac{\sum_{k,k'} kk' \left[e(k, k') - e(k)e(k') \right]}{\sum_k k^2 e(k) - \left[\sum_k k e(k) \right]^2}. \quad (3.12)$$

Foi demonstrado que $r = 0$ corresponde a redes perfeitamente não correlacionadas, enquanto que $r > 0$ indica correlação positiva, ou seja, redes assortativas. Por outro lado, $r < 0$ indica correlação negativa, logo redes não assortativas [13].

3.3 Caminhadas aleatórias

Caminhadas aleatórias são muito utilizadas para descrever uma grande variedade de fenômenos, geralmente referidos como processos estocásticos. Em redes complexas, as caminhadas aleatórias constituem um conceito fundamental para a descrição de processos difusivos, de transporte, de procura e até mesmo para identificação de comunidades em redes [14, 15, 16]. Em sua forma discreta e mais simples, uma caminhada aleatória constitui-se de deslocamentos aleatórios de um vértice para outro adjacente a ele. Sendo assim, podemos caracterizar uma caminhada aleatória por meio de sua matriz de probabilidade de transição $\mathbf{P}$, na qual cada elemento é dado por

$$p_{ij} = \frac{a_{ij}}{\sum_{i=1}^N a_{ij}}. \quad (3.13)$$

Chamamos esta matriz $\mathbf{P}$ de matriz estocástica [17], a qual corresponde à matriz adjacência normalizada por colunas. Iniciando uma caminhada aleatória no vértice i , no instante t , obtemos a equação mestra [16] para a probabilidade p_{ij} de chegar ao vértice j no instante t

$$p_{ij}(t+1) = \sum_l \frac{a_{lj}}{k_l} p_{il}(t). \quad (3.14)$$

O método AL, apresentado no Cap. 5 desta tese, estende a ideia de caminhadas aleatórias, que aliada a uma rede de resistores define uma escala de dissimilaridades que possibilita a identificação de comunidades.

3.4 Modelos de redes complexas

O chamado **grafo aleatório** é considerado como o primeiro modelo de grafo. Este modelo foi proposto por Paul Erdős e Alfred Rényi em 1959 [18]. Até recentemente,

este modelo permaneceu como sendo o paradigma da teoria de grafos. Tal modelo, assume simplesmente que para cada par de vértices, haverá uma aresta conectando-os com probabilidade $p \leq 1$. Logo, a distribuição de graus de tal grafo será dada por

$$P(k) = \binom{N}{k} p^k (1-p)^{N-k} \approx \frac{\langle k \rangle^k e^{-\langle k \rangle}}{k!}, \quad (3.15)$$

na qual $\langle k \rangle = Np$, corresponde ao grau médio da rede. Dado o caráter aleatório da existência de uma aresta conectando dois vértices, um grafo aleatório não deve, a princípio, apresentar nenhuma estrutura topológica relevante. Por este motivo, logo ficou claro que o modelo de grafo aleatório era simples demais para representar redes reais. Este fato foi evidenciado por Duncan Watts e Steven Strogatz [4], através de seus esforços para caracterizar quantitativamente o experimento de Stanley Milgram [19]. Em tal experimento Milgram documentou que em uma rede social, na qual as arestas representam relacionamentos sociais, a distância média entre dois vértices é de aproximadamente 6 vértices. Ou seja, são necessários em geral, apenas seis contatos para conectar quaisquer duas pessoas no hemisfério ocidental [20]. Sendo assim, redes que possuam tal característica são chamadas de **redes de mundo pequeno**.

Em uma rede de mundo pequeno, um vértice arbitrário possuirá $\langle k \rangle$ vizinhos, $\langle k \rangle^2$ segundos vizinhos, $\langle k \rangle^3$ terceiros vizinhos, etc. Sendo assim, uma pessoa que possua 100 amigos ou colegas, estará conectada a 10^{12} pessoas em apenas seis passos; considerando que há mais de 6 884 472 357 pessoas no mundo¹, este valor é suficiente para corroborar o resultado de Milgram.

O problema de se utilizar uma rede aleatória como modelo para descrever uma rede social está no fato de que redes sociais apresentam a propriedade de aglomeração ou *clustering*. Ou seja, uma rede social tende a apresentar um maior número de triângulos do que uma rede aleatória. Um triângulo consiste na manifestação do fato de dois vértices vizinhos a um terceiro, também serem conectados. O coeficiente de *clustering*, c_i , do vértice i , é dado pelo número de arestas entre os vizinhos do vértice i , dividido pelo número de arestas que poderia existir entre eles [4],

$$c_i = \frac{2}{k_i(k_i - 1)} \sum_{j,l=1}^N \frac{1}{2} a_{ij} a_{jl} a_{li}. \quad (3.16)$$

A soma corresponde ao número total de arestas entre os vizinhos do vértice i , enquanto que $k_i(k_i - 1)/2$ corresponde ao número máximo de arestas possíveis entre o grupo formado pelos vértices vizinhos de i . Com isso, o coeficiente de *clustering* da rede (C) é determinado como sendo o valor médio dos coeficientes de agrupamento considerando todos os vértices da rede, ou seja,

¹Valor obtido do U.S. Bureau of the Census, o qual estimava que este fosse o tamanho da população mundial em 28 de novembro de 2010 às 21:36 UTC (EST+5).

$$C = \frac{1}{N} \sum_{i=1}^N c_i. \quad (3.17)$$

Na Fig. 3.2 são apresentadas três redes com valores de coeficiente de *clustering* distintos. Vemos que quanto maior o número de conexões entre vizinhos, maior será o valor do coeficiente de *clustering*. O resultado $c_i = 1$ implica que todos os vizinhos estão completamente conectados. Esta ideia leva à definição de cliques em uma rede. Em uma rede simples, um n -clique é definido como sendo o conjunto de n vértices no qual cada vértice é conectado a todos os outros vértices do conjunto (Fig. 3.2b). No caso $n = 3$, os cliques correspondem a triângulos. Se a rede inteira puder ser considerada como um clique, então esta rede é dita completa (Fig. 3.2c).

Figura 3.2: Coeficiente de *clustering* do vértice representado por um quadrado, nos casos: (a) não há conexões entre os vizinhos, (b) os vizinhos estão conectados, sendo que os vértices dentro da região delimitada pela linha pontilhada formam um clique com 4 vértices e (c) corresponde a um grafo completo.

Grafos aleatórios apresentam distribuições de vértices com decaimento exponencial, enquanto diversas outras redes apresentam um decaimento muito mais lento. Tal decaimento pode ser melhor representado por uma lei de potências, conhecida também por lei de Zipf [21, 22]

$$P(k) \sim k^{-\gamma}. \quad (3.18)$$

Redes que apresentam tal comportamento são chamadas **redes livres de escala** [6, 8, 9, 10]. Em tais redes, o comportamento livre de escala influencia a topologia da rede, implicando em uma alta probabilidade de existência de vértices com alto valor de grau, ou seja, vértices com muitos vizinhos.

Barabási e Albert [7], desenvolveram um modelo de crescimento de redes no qual vértices são adicionados continuamente. Entretanto, ao invés de conectá-los aleatoriamente a outros vértices, as conexões são efetuadas com probabilidade proporcional ao grau de cada vértice. Este fato faz com que surja uma conexão preferencial na qual novos

vértices tendem a se conectar a vértices já altamente conectados, exemplificando desta forma o lema de que o “rico fica cada vez mais rico”. Analisando-se as redes formadas de acordo com este algoritmo, chega-se à distribuições que seguem leis de potência com $\gamma = 3$ [7]. Como mostrado na Fig. 3.3, contrariamente a redes aleatórias em que os vértices são praticamente equivalentes, as redes geradas seguindo tal estratégia apresentam vértices com alto grau, os quais formam concentradores (*hubs*).

Figura 3.3: Redes complexas com $N = 200$ vértices e grau médio $\langle k \rangle = 2$. Em (a) uma rede aleatória, em (b) uma rede livre de escala com *hubs* representados por vértices de forma quadrada.

Em geral, redes de computadores apresentam distribuições de suas máquinas que se assemelham a redes livres de escala. Sendo assim, em caso de um ataque a máquinas efetuado de forma aleatória, dificilmente impactará na rede como um todo. Esta vantagem entretanto, também apresenta uma desvantagem caso o atacante conheça quais são os concentradores de tal rede e concentre seu ataque a tais máquinas.

3.5 Comunidades em redes complexas

A ideia por detrás de procedimentos de agrupamento ou detecção de comunidades em redes complexas, consiste na determinação dos conjuntos de vértices que possuem alguma característica comum. Por meio de tal característica, deve ser possível classificá-los e organizá-los em grupos naturais. Considerando como exemplo uma lista de cidades ao redor do mundo, podemos naturalmente associá-las a países, e tais países a continentes. Este processo de associação mostra que dada uma grande quantidade de dados (cidades), tendemos naturalmente a usar diferentes níveis de organização a fim de lidar com a complexidade do sistema. Por meio de tal simplificação, somos capazes de determinar as

características principais do conjunto de dados inicial, sem incluir informações específicas de cada um de seus constituintes.

Por outro lado, tendo em mãos um conjunto de dados sobre o qual não se tem informações a respeito de sua organização, utilizamos métodos de agrupamento tais que seja possível extrair os diferentes níveis de organização dos dados. Para isto, associamos cada vértice a um determinado grupo (ou comunidade) com base em similaridades ou diferenças entre eles. A informação a respeito de tal similaridade entre vértices está geralmente codificada sob a forma de redes. Portanto, estratégias de detecção de comunidades tornam-se indispensáveis para a compreensão dos mais diversos sistemas.

Considerando um exemplo em biologia envolvendo proteínas, utilizando-se informações como similaridade estrutural e identidade sequencial, é possível classificá-las por exemplo, em classes como proteínas compostas por hélices- α , por folhas- β ou estruturas compostas por ambas.

Uma forma possível de agrupamento consiste em simplesmente efetuar uma inspeção visual e manualmente atribuir cada vértice a uma comunidade apropriada. Este procedimento, entretanto, é inviável uma vez que os sistemas em estudo, em geral, são compostos de bem mais do que algumas poucas dezenas de vértices. Por este motivo, estratégias automatizadas de agrupamento são extremamente importantes no âmbito de redes complexas.

O estudo de comunidades em redes complexas baseia-se na observação de que, contrariamente aos grafos aleatórios de Erdős e Rényi, uma rede complexa geralmente é constituída de *grupos nos quais os vértices são mais conectados entre si, do que com o resto da rede*. Estes grupos são denominados comunidades, em analogia à terminologia praticada em redes sociais. Sendo assim, definimos comunidade da seguinte maneira:

Definição 1. *Uma comunidade é composta por um conjunto de vértices que possuem um maior número de arestas conectando-os a vértices da mesma comunidade, do que arestas conectando-os a vértices em outras comunidades.*

Na Fig. 3.4, apresentamos um exemplo de rede com estrutura de comunidades. Vemos claramente que esta rede possui três comunidades bem definidas; entretanto, há um vértice (marcado com sinal de interrogação) sobre o qual é difícil inferir a respeito de qual comunidade ele pertence, uma vez que este está conectado a duas comunidades distintas pelo mesmo número de arestas. Neste exemplo, a estrutura de comunidade é facilmente determinada, à exceção do vértice marcado por sinal de interrogação. Para redes reais, pode haver diversos vértices cuja classificação seja duvidosa. Além disso, similarmente à assortatividade, o grau do vértice é utilizado como parâmetro classificatório em comunidades, uma vez que esta informação está presente em qualquer rede.

Figura 3.4: Exemplo de uma rede simples com três comunidades, na qual o vértice destacado pelo sinal de interrogação é de difícil classificação uma vez que o mesmo está conectado a duas comunidades pelo mesmo número de arestas.

3.5.1 Modularidade

Diversos métodos têm sido propostos com a finalidade de classificar vértices em comunidades [23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34], e com eles a necessidade de se determinar a qualidade das divisões propostas. Um destes métodos, proposto por Girvan e Newman (GN) baseia-se em identificar as arestas que são centrais – *edge betweenness* [25]. A centralidade de uma aresta i é definida como sendo o número de menores caminhos entre quaisquer dois vértices que passam por esta aresta. Assim, esta quantidade mede a influência da aresta i sobre o fluxo de informação entre vértices. Se a rede contiver comunidades que estejam fracamente conectadas, então todos os menores caminhos entre comunidades distintas devem passar por umas poucas arestas. Desta forma, tais arestas terão uma alta centralidade. Como procedimento, calcula-se a centralidade de todas as arestas da rede e sucessivamente remove-se aquela mais centrais. Entretanto, a cada remoção de aresta torna-se necessário recalcular o valor da centralidade de todas as arestas remanescentes uma vez que a remoção de uma única aresta altera o sistema como um todo. O procedimento é então finalizado quando não houver mais arestas a serem removidas.

Neste mesmo artigo, Girvan e Newman propõe uma função cujo objetivo é medir a qualidade das divisões resultantes de métodos de identificação de comunidades. A esta função deram o nome de modularidade [25]. A função modularidade considera a divisão da rede em s comunidades distintas. Definido-se um matriz simétrica $\mathbf{e}$, $s \times s$, cujos elementos e_{ij} correspondem à fração de todas as arestas na rede que conectam vértices da comunidade i a vértices da comunidade j . O traço desta matriz, dado por

$$\text{Tr } \mathbf{e} = \sum_i e_{ii}, \quad (3.19)$$

fornece a fração de arestas da rede que conectam vértices de uma mesma comunidade. Sendo assim, uma boa divisão da rede deve ter um alto valor para seu traço. Entretanto, este parâmetro sozinho não é um bom indicador pois colocando todos os vértices em uma mesma comunidade resulta em seu valor máximo $\text{Tr } \mathbf{e} = 1$.

É conveniente definir então, a soma a_i em linha (ou coluna), como sendo

$$a_i = \sum_j e_{ij}, \quad (3.20)$$

a qual representa a fração das arestas que se conectam a vértices na comunidade i . Temos que $e_{ij} = a_i a_j$, representa a fração de arestas entre vértices, desconsiderando a qual comunidade eles pertencem. Dessa forma, define-se modularidade como sendo

$$Q = \sum_i (e_{ii} - a_i^2) = \text{Tr } \mathbf{e} - \|\mathbf{e}^2\|, \quad (3.21)$$

na qual $\|\mathbf{x}\|$ representa a soma dos elementos da matriz $\mathbf{x}$. Assim, este parâmetro fornece a fração de arestas na rede que conectam vértices do mesmo tipo (ou seja, número de arestas intra-comunidades), menos o valor esperado para esta mesma quantidade em uma rede com a mesma divisão em comunidades, mas com vértices conectados aleatoriamente. Se o número de arestas intra-comunidades não for melhor que a situação aleatória, teremos $Q = 0$. Por outro lado, se cada vértice for classificado em uma comunidade diferente de tamanho 1, teremos $Q \leq 0$. Valores de Q tendendo a um (máximo), indicam redes com forte caráter de comunidade. Uma forma alternativa de apresentar a função modularidade é defini-la como

$$Q = \sum_{s=1}^S \left[\frac{l_s}{M} - \left(\frac{d_s}{2M} \right)^2 \right], \quad (3.22)$$

na qual l_s corresponde ao número de arestas entre vértices da comunidade s , d_s é a soma dos graus dos vértices da comunidade s e S corresponde ao número total de comunidades.

Neste ponto algumas considerações se fazem necessárias. Um particionamento aleatório em S comunidade tem modularidade próxima a $Q = 0$, ao passo que um particionamento, em geral considerado como *ótimo*, tem valores de $Q \rightarrow 1$. Deve ser ressaltando entretanto que valores elevados de modularidade implicam que o particionamento difere de um particionamento aleatório. Em particular, isto não necessariamente implica que para uma rede aleatória, a princípio sem estrutura de comunidades, todos os possíveis particionamentos terão modularidade pequena. Este fato foi inicialmente observado por Guimerà e Amaral [35]. A partir deste estudo foi demonstrado por Reichardt [36, 37] que redes

puramente aleatórias apresentam uma modularidade intrínseca e pode ser particionada tal que forneça valores elevados de Q . Isto é atribuído à flutuações aleatórias que fazem com que exista uma estrutura modular em tais redes com uma certa probabilidade. Fortunato e Barthélemy [38] em seu trabalho apresentam também uma discussão a respeito do limite de resolução da função modularidade. Tal limitação faz com que a função modularidade seja incapaz de revelar a divisão real de uma rede em comunidades. Sendo assim, um valor alto de Q não garante que a rede seja realmente modular, ou ainda que esta seja a divisão real da rede, como será discutido ao longo desta tese.

A função modularidade, é uma forma natural de avaliar a estrutura de comunidades de uma rede. Por este motivo, diversos métodos surgiram nos últimos anos cujo foco foi a maximização desta função [23, 24, 28, 29, 30, 31]. Entretanto deve-se tomar o cuidado com o fato de que valores elevados da modularidade apenas garantem que o particionamento em consideração é o mais diferente possível de um particionamento em comunidades totalmente aleatório e que muitas vezes pode fornecer divisões compatíveis com as divisões reais das redes em estudo.

3.5.2 Métodos de identificação de comunidades

Para podermos avaliar os resultados obtidos nesta tese, utilizamos os resultados reportados por quatro métodos de identificação de comunidade em redes complexas, o método proposto por Girvan e Newman (GN), o método baseado na otimização da modularidade por *Simulated Annealing*, o método conhecido como LPAm+ e o método VP. Por este motivo faremos uma breve descrição a respeito destes métodos nesta seção.

Dentre todos os métodos de identificação de comunidades em redes complexas desenvolvidos nos últimos anos, o método desenvolvido por Girvan e Newman (GN) [39] (explicado na Sec. 3.5.1) merece ser mencionado em primeiro lugar. Logo após sua publicação, este método tornou-se muito popular e impulsionou o interesse pelo problema de determinação de comunidades em redes complexas, o que acarretou no surgimento de um grande número de métodos relacionados a este problema. Este método não é o que possui o melhor desempenho, nem tampouco é o mais acurado. Entretanto, o método GN tem sido muito utilizado como base de comparação para os diversos métodos que vêm sendo desenvolvidos.

Guimerà e Amaral [40] por sua vez, utilizaram o método de otimização *Simulated Annealing* (SA) a fim de obter a divisão de uma rede que forneça o maior valor de modularidade possível. Sendo assim, os autores utilizaram a ideia de que vértices com mesmas características devem ter propriedades topológicas similares. Esta ideia, derivada de redes sociais, considera que em uma fábrica, por exemplo, os operadores de máquinas tendem a ter mais contatos sociais com outros operadores, ao passo que os diretores tendem a ter mais contatos com outros diretores. Dada esta característica, deve ser

possível identificar dois grupos bem definidos. Em termos de redes complexas, podemos traduzir esta ideia por meio da definição de comunidades.

Para a determinação de tais comunidades, os autores classificam os vértices de uma rede de acordo com seu grau z intra-comunidade e seu coeficiente de participação P , o qual estabelece o quão bem distribuídas suas arestas estão em relação às outras comunidades. Além disso, são definidos sete perfis básicos divididos por regiões no espaço de parâmetros $z - P$. Por meio de simulações utilizando o método de otimização *Simulated Annealing* são efetuadas buscas das configurações no espaço de parâmetros tal que o parâmetro modularidade seja maximizado [35, 41].

O método LPAm+, proposto por Liu e Murata [24] consiste em inicialmente atribuir um número de comunidade a cada vértice da rede. Durante as iterações, cada vértice determina qual é a comunidade mais frequente dentre seus vizinhos e se muda para esta comunidade. Caso haja mais de uma comunidade com mesma frequência de ocorrência, o vértice muda para a comunidade que maximize o valor da função modularidade. Caso o algoritmo encontre valores de modularidades iguais, ele passa a fundir comunidades em busca de valores mais elevados de modularidade. Este processo se repete até que não haja mais incremento no valor da modularidade.

O método LPAm+ fornece alguns dos melhores resultados já obtidos para diversas redes sociais comumente utilizadas nos testes de detecção de comunidades.

O quarto método utilizado como comparação, proposto por Agarwal e Kempe [23], o qual denominamos VP, é baseado em técnicas de programação matemática. Este método produz uma árvore aglomerativa hierárquica e, partindo de uma única comunidade passa a efetuar divisões desta buscando particionamentos que forneçam incremento na função modularidade.

Este método, até o momento de escrita desta tese é o que fornece os valores mais elevados de modularidade para as diversas redes sociais utilizadas para testes dos métodos propostos nesta tese e por este motivo é considerado como *padrão-ouro*.

Capítulo 4

Proteínas

4.1 Proteínas: base biológica

Num primeiro momento é difícil aceitar a ideia de que cada um dos organismos vivos seja meramente um sistema químico. Até o século XIX foi amplamente aceito que o animais tinham uma força - um “animus” - que era responsável pelas suas características [42].

Sabe-se, atualmente, que não há nada nos organismos vivos que não obedeça as leis da química e da física. Os organismos vivos são fundamentalmente baseados em compostos de carbono, as células são formadas por 70% de água e a vida depende quase que exclusivamente de reações que ocorrem em soluções aquosas. A química das células é extremamente complexa - mesmo a mais simples das células possui uma química muitíssimo mais complexa do que qualquer outro sistema químico conhecido. Além disso, todas as características de uma célula dependem das moléculas que a mesma possui. Uma molécula é definida como um conjunto de átomos que são mantidos unidos por meio de diversas ligações químicas. Moléculas biológicas são encontradas livres em solução, sendo algumas utilizadas na construção de grandes macromoléculas, tais como as proteínas, os ácidos nucleicos e os grandes polissacarídeos.

Grande parte da composição celular está relacionada com as proteínas [43]. As proteínas constituem mais da metade do peso celular e são responsáveis por determinar a forma e a estrutura da célula, assim como servem de instrumentos para o reconhecimento molecular e processos de catálise. Embora o DNA armazene a informação requerida para a formação de uma célula, as proteínas é que são responsáveis por uma enorme quantidade de processos celulares.

4.1.1 Composição e formação das proteínas

As proteínas são as macromoléculas estruturalmente mais complexas conhecidas. Cada tipo de proteína tem a sua própria estrutura e função. Polímeros são definidos como sendo grandes moléculas compostas por subunidades idênticas ou semelhantes, denominadas monômeros. Sendo assim, podemos definir proteínas como polímeros compostos por 20 tipos de aminoácidos, ou seja, os monômeros proteicos [44].

Os aminoácidos são moléculas que contêm, simultaneamente, os grupos funcionais amina e ácido carboxílico; As moléculas amina e carboxila estão ligadas a um carbono alifático, denominado carbono- α (C_α) [43, 44]. Ao menos um átomo de hidrogênio está ligado a este carbono. As ligações do C_α são completadas com a presença de uma cadeia lateral, distinta para cada tipo de aminoácido. Um esquema ilustrando a configuração geral de um aminoácido é apresentada na Fig. 4.1. Os aminoácidos formam um conjunto de 20 tipos, os quais constituem todas as proteínas e peptídeos.

Figura 4.1: Estrutura básica dos aminoácidos. Estes são compostos por um C_α , no qual um hidrogênio, um grupo amina e um grupo carboxila são ligados. Além destes elementos, há uma cadeia lateral para cada aminoácido. A cadeia lateral é a responsável pela identidade de cada um dos 20 tipos de aminoácidos existentes.

As 20 cadeias laterais que aparecem nas proteínas podem ser divididas em três classes, sendo elas: hidrofóbicas, carregadas e polares. Na Tab. 4.1 estão listadas todas as cadeias laterais classificadas de acordo com a categoria em que se enquadram, além dos códigos de uma e de três letras, os quais são comumente utilizados para referenciá-las.

Classe	Nome	Código
hidrofóbica	Alanina	A ALA
	Valina	V VAL
	Leucina	L LEU
	Glicina	G GLY
	Isoleucina	I ILE
	Fenilalanina	F PHE

(continua na próxima página)

Classe	Nome	Código	
carregada	Prolina	P	PRO
	Metionina	M	MET
	Aspartato	D	ASP
	Glutamato	E	GLU
polar	Lisina	K	LYS
	Arginina	R	ARG
	Serina	S	SER
	Treonina	T	THR
	Cisteína	C	CYS
	Asparagina	N	ASN
	Glutamina	Q	GLN
	Histidina	H	HIS
	Tirosina	Y	TYR
	Triptofano	W	TRP

Tabela 4.1: Relação das 20 cadeias laterais com seus respectivos códigos de uma e três letras, classificadas de acordo com as categorias: hidrofóbicas, polares e carregadas.

Em soluções aquosas de pH neutro, os aminoácidos podem existir em duas formas. Uma pequena fração encontra-se em uma forma eletricamente neutra, ou seja, com o grupo amina desprotonado ($-NH_2$) e o grupo carboxila protonado ($-COOH$). A maioria encontra-se, no entanto, numa forma ionizada, em que o grupo amina se encontra protonado ($-NH_3^+$) e o ácido carboxílico desprotonado ($-COO^-$) [43, 44].

Os aminoácidos podem ligar-se entre si com uma ligação amida, sendo esta neste caso especificamente denominada como ligação peptídica. Esta ligação ocorre entre o átomo de carbono do grupo carboxila e o nitrogênio do grupo amina; neste processo uma molécula de água é liberada, sendo que a ligação final ocorre entre o carbono de um grupo carboxila e o hidrogênio de uma amina secundária. Como consequência, observa-se a formação de uma cadeia peptídica, ou seja, uma cadeia formada por diversos aminoácidos ligados por ligações peptídicas, a qual terá um grupo amina numa extremidade (denominada N-terminal) e um grupo carboxila na extremidade oposta (denominada C-terminal). Por este motivo, define-se que a cadeia polipeptídica vai da amina terminal à carboxila terminal [45].

A cadeia peptídica é designada como sendo a cadeia principal ou esqueleto *backbone* da proteína, a partir da qual diversas cadeias laterais ou resíduos (R), se projetam. Uma representação esquemática de tal configuração é mostrada na Fig. 4.2, na qual podemos observar a existência de uma unidade que se repete ao longo de toda a cadeia principal

e é comum a todos os aminoácidos [42, 43, 44, 45].

Figura 4.2: Diagrama esquemático da cadeia polipeptídica, também denominada esqueleto (ou *backbone*) das proteínas.

Um ponto importante com relação à ligação peptídica está no fato de que a mesma possui uma geometria planar [43].

A geometria formada devido à ação das diversas ligações químicas existentes na cadeia polipeptídica, assim como a sequência de aminoácidos em cada polipeptídeo é exclusiva para cada tipo de proteína. A estrutura tridimensional formada é única e denominada conformação nativa. Se um único aminoácido na sequência é alterado, este pode modificar potencialmente a função da proteína em questão. Por exemplo, a anemia falciforme é causada por uma alteração em um único nucleotídeo na sequência do DNA, que faz com que apenas um aminoácido, em uma das cadeias polipeptídicas da hemoglobina seja diferente. Devido à isso, a célula sanguínea acaba sendo deformada e incapaz de transportar oxigênio corretamente, conforme ilustrado na Fig. 4.3.

Figura 4.3: Ilustração de uma hemácia sadia e de uma hemácia falciforme. Esta deformação é causada pela alteração de um único aminoácido ao longo da cadeia peptídica (imagem adaptada: <http://uofmchildrenshospital.org>).

A sequência dos aminoácidos possui um papel chave com relação à bioquímica das proteínas, porém a conformação das mesmas também deve ser considerada. Tendo em vista que algumas proteínas contêm apenas uma cadeia polipeptídica, e outras (como a

hemoglobina), contêm várias cadeias unidas através de diferentes conformações, podemos dizer que a determinação destas cadeias, com um certo grau de acurácia, é extremamente importante no estudo e compreensão dos diversos papéis desempenhados pelas proteínas.

4.1.2 Estrutura das proteínas

Estudos estruturais através de difração de raios-X foram responsáveis por uma grande contribuição na tarefa de transformar a Química, que antes era considerada uma ciência meramente descritiva, numa ciência na qual é possível prever propriedades de novos compostos baseando-se em sólidas teorias. Quando B.L. Bragg [46] observou a primeira estrutura cristalina, de uma pedra de sal (NaCl), os resultados previamente obtidos mudaram completamente: conceitos relacionados às forças de ligação em compostos iônicos foram finalmente considerados.

A primeira cristalografia estrutural por raios-X reportada, de uma molécula de proteína globular, foi a da mioglobina em 1958 [47, 48]. Proteínas globulares são formadas por cadeias polipeptídicas que se enovelam, adquirindo formas esféricas ou globulares. A maioria dessas proteínas é solúvel em sistemas aquosos. Nesta categoria situam-se as proteínas ativas (como por exemplo as enzimas), as proteínas transportadoras (como a hemoglobina), dentre outras. Uma ilustração da estrutura encontrada é apresentada na Fig. 4.4. Esta descoberta veio como um choque para aqueles que tinham desejado que a estrutura e função proteica fosse algo simples, com princípios gerais bem estabelecidos, de maneira análoga àquela simples dupla hélice da estrutura do DNA, a qual havia sido determinada cinco anos antes por James Watson e Francis Crick [49].

John Kendrew, no Conselho de Pesquisas Médicas de Biologia Molecular, em Cambridge na Inglaterra, foi quem determinou a estrutura da mioglobina [50], expressando o seu desapontamento com relação à complexidade da estrutura encontrada nas seguintes palavras:

... talvez a mais importante característica desta molécula seja a complexidade da mesma e esta falta de simetria. Este arranjo parece ser quase que totalmente ausente de regularidades, as quais alguém instintivamente pudesse ter antecipado; esta estrutura é mais complicada do que tem sido predita por qualquer teoria de estrutura de proteínas.¹

¹original: ...perhaps the most remarkable features of the molecule are its complexity and its lack of symmetry. The arrangement seems to be almost totally lacking in the kind of regularities which one instinctively anticipates, and it is more complicated than has been predicted by any theory of protein structure.

Figura 4.4: A mioglobina foi a primeira proteína a ter sua estrutura tridimensional revelada. Em 1958 John Kendrew e seus colegas determinaram a estrutura da mesma através da cristalografia por raios-X de alta resolução.

Atualmente, é fácil observar que esta irregularidade estrutural é altamente necessária para que as proteínas desempenhem as mais diversas funções. O armazenamento de informação e transferência da mesma, pela molécula de DNA, é essencialmente linear; moléculas de DNA com as mais diversas informações podem possuir basicamente a mesma estrutura. Em contrapartida, proteínas necessitam reconhecer milhares de centenas de diferentes moléculas na célula, através de interações tridimensionais detalhadas, as quais requerem diversas irregularidades estruturais das proteínas. Apesar destes requerimentos, há aspectos regulares na estrutura proteica, sendo a mais importante delas, suas estruturas secundárias [42, 43, 44, 45].

4.1.3 Conformação proteica: estrutura secundária

Quando estudos à respeito da mioglobina foram realizados com alta resolução, Kendrew notou que os aminoácidos situados no interior da proteína possuíam, quase que exclusivamente, cadeias laterais hidrofóbicas. Este foi um dos mais importantes princípios gerais emergentes do estudo da estrutura das proteínas. A força principal para o enovelamento de moléculas globulares de proteínas solúveis em água, seria para embalar as cadeias laterais hidrofóbicas no interior da molécula, criando um centro hidrofóbico e uma superfície hidrofílica [44].

Ao criar um núcleo hidrofóbico a partir de uma sequência de aminoácidos – denominada também como estrutura primária – um problema maior aparece: para manter as cadeias laterais no centro hidrofóbico, a cadeia principal também tende a ser empacotada em direção ao interior da molécula [43, 44].

Num ambiente hidrofóbico, estas cadeias principais de grupos polares podem ser neutralizadas pela formação de ligações do tipo pontes de hidrogênio. Este problema é solucionado de uma forma muito interessante, através da formação de estruturas secundárias regulares no interior da molécula proteica. Tais estruturas secundárias são geralmente de dois tipos: α -hélice ou folha- β . Ambos os tipos são caracterizados por ligações do tipo pontes de hidrogênio entre os grupos NH e C=O das cadeias principais; tais ligações são formadas quando um certo número de resíduos consecutivos possuem o mesmo ângulo ϕ e ψ . Uma terceira categoria, denominada **volta β** , pode ser incluída como sendo um elemento da estrutura secundária apesar de não apresentar ângulos regulares de torção [44]. Nas **volta β** a cadeia é forçada na direção oposta, invertendo o sentido da mesma e fazendo com que esta seja o mais compactada possível [51].

Os elementos das estruturas secundárias, formados desta forma e mantidos unidos pelo núcleo hidrofóbico, formam uma rígida e estável estrutura. Eles exibem uma flexibilidade relativamente pequena com relação de uns aos outros e são as estruturas mais bem definidas através das técnicas de raios-X e RMN. Grupos funcionais da proteína são conectados a esta estrutura, sendo estes tanto diretamente ligados às cadeias laterais quanto, mais frequentemente, ligados em regiões curvas que se conectam sequencialmente à elementos da estrutura secundária adjacente [43, 45].

4.1.4 A estrutura α -hélice

A α -hélice é o elemento clássico da estrutura proteica. Esta estrutura foi descrita primeiramente por Linus Pauling em 1951, enquanto ele trabalhava no Instituto de Tecnologia da Califórnia, no EUA. Ele predisse que havia uma estrutura a qual seria estável e energeticamente favorável em proteínas. Ele fez esta importante predição com base em acurados parâmetros geométricos, os quais ele havia derivado das unidades peptídicas de resultados oriundos de análises cristalográficas das estruturas de uma certa quantidade de moléculas pequenas [43]. Esta predição foi recebida, quase que imediatamente, com um forte apoio experimental de padrões de difração obtidos por Max Perutz em Cambridge, para cristais de hemoglobina e fibras de queratina. Esta previsão foi completamente verificada por John Kendrew em seu experimento de alta resolução para identificação de estruturas da mioglobina, a qual possui toda estrutura secundária em forma helicoidal [45].

As estruturas α -hélice em proteínas são encontradas quando um trecho de resíduos possuem um par de ângulos, ϕ e ψ , de aproximadamente -60° e -50° . A α -hélice possui 3,6 resíduos por volta, com pontes de hidrogênio entre o C=O do resíduo n e o NH do resíduo $n + 4$, conforme ilustrado na Fig. 4.5.

Figura 4.5: Esquema ilustrativo da estrutura α -hélice. Esta estrutura encontra-se presente na estrutura secundária dos níveis de organização das proteínas, possuindo o esqueleto polipeptídico estreitamente enrolado ao longo do maior eixo da molécula e tendo os grupos R dos resíduos de aminoácidos projetados para fora do esqueleto helicoidal. Fonte: http://ftp.unb.br/pub/UNB/cbsp/Download/Imagens/300_dpis/Biomoleculas-300/prot/

Desta forma, todos os grupos NH e C=O possuem uma ligação do tipo ponte de hidrogênio, exceto o primeiro grupo NH e o último grupo C=O, no final da α -hélice. Como consequência, o final das α -hélices são polares e quase sempre localizam-se na superfície da proteína.

Variações da estrutura α -hélice, onde as cadeias são mais relaxadas ou mais compactas, com pontes de hidrogênio localizadas entre os resíduos $n + 5$ ou $n + 3$ ao invés de $n + 4$, são denominadas π hélice ou 3_{10} hélice, respectivamente. A 3_{10} hélice possui três resíduos por volta e contém 10 átomos entre o hidrogênio doador e receptor, sendo o seu nome derivado deste fato [44]. Tanto a π hélice quanto a 3_{10} hélice ocorrem raramente e geralmente, somente no final das α -hélices. Estas configurações não são energeticamente favoráveis, uma vez que os átomos do esqueleto são muito rigidamente compactados numa 3_{10} hélice, ou muito frouxamente compactados, no caso da π hélice. Somente a α -hélice possui os átomos do esqueleto configurados de maneira a formar uma estrutura estável [43, 45].

Em proteínas globulares, as estruturas hélices α variam consideravelmente em comprimento, variando de quatro ou cinco aminoácidos até mais de 40 resíduos. O comprimento médio é de cerca de dez resíduos, correspondendo a três voltas. O aumento por resíduo numa α -hélice é $1,5\text{\AA}$ com relação ao eixo vertical; isto corresponde a cerca de $15\text{\AA}$, considerando-se o comprimento do início ao fim de uma estrutura α -hélice média.

4.1.5 A estrutura folha- β

O segundo principal elemento estrutural encontrado em proteínas globulares é a folha- β . Esta estrutura é formada a partir da combinação de diversas regiões da cadeia polipeptídica, contrastando com a α -hélice, que é formada a partir de uma região contínua [45]. As folhas β encontram-se alinhadas adjacente uma às outras, através de pontes de hidrogênio entre os grupos C=O de uma folha e os grupos NH de outra, conforme apresentado na Fig. 4.6.

Figura 4.6: A estrutura folha- β é formada a partir da combinação de diversas regiões da cadeia polipeptídica, contrastando com a α -hélice, que é formada a partir de uma região contínua. Fonte: <http://www.structuralbioinformatics.com/chapter2.shtml>.

As folhas β são formadas a partir de diversas voltas- β ; estas são “pregueadas” com átomos de C_α sucessivamente situados ligeiramente acima e abaixo do plano da folha- β em questão. As cadeias laterais seguem este mesmo padrão, de tal forma que a cada volta, estas localizam-se acima e abaixo da folha- β [43, 45].

4.1.6 Conformação proteica: estrutura terciária

Diversas proteínas possuem apenas uma cadeia principal, outras entretanto, possuem diversas cadeias polipeptídicas idênticas, correspondentes às chamadas estruturas quaternárias. Os domínios podem ser definidos como sendo a própria cadeia polipeptídica – ou parte dela – que independentemente se arranja em uma estrutura terciária estável. Frequentemente, diferentes domínios de uma proteína estão associados à diferentes funções. Existem diversos casos em que uma determinada função biológica de uma espécie de organismo é desempenhada por cadeias polipeptídicas separadas, enquanto que em outras espécies, tais funções são desempenhadas por uma única proteína com diversos domínios.

A estrutura terciária de uma proteína é uma descrição da forma de como toda a cadeia

(incluindo as estruturas secundárias) enovela-se em seu formato final tridimensional. A estrutura terciária é a forma geométrica final e específica que cada proteína assume. Esta forma final é determinada por uma variedade de interações de ligação entre as cadeias laterais dos aminoácidos. Essas interações podem ser mais fortes do que as ligações de pontes de hidrogênio entre os grupos amida as quais mantêm a estrutura helicoidal. Como resultado, as interações entre as cadeias laterais pode causar uma série de dobras, curvas, “loops” na cadeia da proteína.

Muitas proteínas têm um design modular, com vários domínios distintos resultantes da fusão de genes durante a evolução. Um domínio com uma determinada estrutura e função pode ser parte da estrutura de várias proteínas de uma forma distinta.

4.1.7 Domínios

O domínio de uma proteína é uma parte da sequência de aminoácidos e da estrutura proteica, a qual pode evoluir, funcionar e existir independentemente do restante da cadeia proteica. Cada domínio forma uma estrutura compacta tridimensional e muitas vezes podem ser independentemente estáveis e enovelados individualmente. Muitas proteínas consistem de vários domínios estruturais. Um domínio pode aparecer em uma variedade de proteínas diferentes, uma vez que a evolução molecular usa domínios como blocos de construção, os quais podem ser recombinados em diferentes arranjos, para criar proteínas com funções diferentes. Domínios variam em comprimento, podendo ser compostos por cerca de 25 aminoácidos até 500 aminoácidos [52, 53, 54, 55]. O conceito de domínio foi proposto pela primeira vez em 1973 por Wetlaufer após estudos cristalográficos por raios-X de lisozima [56] e papaína [57] e por estudos da proteólise limitada de imunoglobulinas [58, 59]. Domínios têm sido descritos como unidades de estrutura compacta [60], unidades de função e evolução [61] ou unidades de enovelamento. Entretanto, cada definição é válida e se sobrepõe, isto é, um domínio de estrutura compacta é encontrado em diversas proteínas, enovelando-se de acordo com o seu ambiente estrutural. A natureza frequentemente apresenta diversos domínios combinados, a fim de formar proteínas com multi-domínios e multifuncionais, possuindo assim um vasto número de possibilidades [62].

Em uma proteína multi-domínio, cada domínio pode realizar sua própria função independentemente ou de maneira combinada com os seus vizinhos. Domínios podem servir como módulos para a construção de um grande conjunto de biomoléculas, tais como partículas virais ou fibras musculares; estes podem ainda realizar catálises específicas ou ligar-se em determinadas regiões, tais como aquelas presentes em enzimas ou proteínas regulatórias.

4.1.8 Domínios como modelos evolucionários

Segundo um artigo publicado na revista Science por F.Jacob em 1977, *Nature is a tinkerer and not an inventor* (a natureza é um funileiro e não um inventor) [63]: novas sequências são adaptadas a partir de sequências preexistentes e não inventadas. Os domínios são os materiais comuns utilizados pela natureza para gerar novas sequências; eles podem ser considerados como unidades geneticamente móveis, conhecidos como “módulos”. Muitas vezes, os carbonos e nitrogênios terminais dos domínios estão juntos no espaço, permitindo que facilmente sejam “encaixados” em estruturas-mãe, durante o processo de evolução.

Muitas famílias de domínios são encontradas em quase todas as formas de vida. Os domínios que são repetidamente encontrados em diversas proteínas são muitas vezes referidos como módulos; podem ser citados como exemplos as proteínas extracelulares associadas com a fibrinólise, coagulação, matriz extracelular, as moléculas de superfície de adesão celular e receptores de citocinas [64].

A evolução molecular deu origem a famílias de proteínas, sendo estas relacionadas de acordo com a sequência e estrutura das mesmas. No entanto, as semelhanças sequenciais podem ser extremamente baixas entre as proteínas que compartilham a mesma estrutura. Já as estruturas proteicas podem ser semelhantes, pois as proteínas divergiram de um ancestral comum. Como alternativa, alguns enovelamentos podem ser mais favorecidos em algumas proteínas do que em outras e, como estes representam um regime estável de estruturas secundárias, algumas proteínas puderam convergir para estes enovelamentos ao longo da evolução.

Existem atualmente cerca de 69 655¹ estruturas de proteínas determinadas experimentalmente; suas estruturas tridimensionais encontram-se depositadas no *Protein Data Bank* (PDB) [43]. No entanto, este conjunto contém uma grande quantidade de estruturas idênticas ou muito semelhantes. Embora todas as proteínas devam ser classificadas de acordo com as famílias estruturais, para melhor compreender suas relações evolutivas, comparações estruturais são melhor alcançadas ao nível do domínio. Por esta razão, muitos algoritmos foram desenvolvidos para a determinação automática de domínios a partir de proteínas com estruturas tridimensionais conhecidas.

4.2 PDB – *Protein Data Bank*

O banco de dados de proteínas (*Protein Data Bank* - PDB) é um repositório de dados de estruturas tridimensionais de grandes moléculas biológicas, tais como proteínas e ácidos nucleicos. Os dados, geralmente obtidos por cristalografia de raios-X ou espectroscopia de RMN submetidos por biólogos e bioquímicos de todo o mundo, são livremente

¹de acordo com o PDB (<http://www.pdb.org/>) em 30 de Novembro de 2010 as 4 PM PST.

acessíveis na Internet. O PDB é supervisionado por uma organização chamada *Worldwide Protein Data Bank* (wwPDB).

O PDB surgiu de um esforço de base [56], em 1971, quando Walter Hamilton do Laboratório Nacional Brookhave decidiu criar o banco de dados em Brookhaven. Após a morte de Hamilton em 1973, Tom Koeztle assumiu a direção do PDB e em Outubro de 1998 [65], o PDB foi transferido para o *Research Collaboratory for Structural Bioinformatics* (RCSB). A transferência foi concluída em Junho de 1999, sendo então Helen M. Berman, da Universidade Rutgers (uma das instituições membro da RCSB), nomeada diretora [66]. Em 2003, com a formação do wwPDB [66], o PDB tornou-se uma organização internacional. Cada um dos quatro membros da wwPDB pode atuar como centro de depósito, processamento de dados e de distribuição de dados do PDB. O processamento de dados refere-se ao fato de que pessoas ligadas ao wwPDB revisam e anotam cada trabalho inscrito. O constante crescimento do PDB é um reflexo de toda investigação e pesquisa que vem ocorrendo nos laboratórios de todo o mundo, conforme mostrado no gráfico da Fig. 4.7. No PDB estão disponíveis estruturas para muitas das proteínas e ácidos nucleicos envolvidos nos processos centrais da vida, sendo esta base de dados um importante pilar e um marco em diversas pesquisas.

Figura 4.7: Crescimento do número total de estruturas na base de dados PDB-RCSB (fonte: <http://www.pdb.org/pdb/statistics/>).

O PDB é um recurso chave nas áreas de biologia estrutural, tal como a genômica estrutural. A maioria das principais revistas científicas, e algumas agências de financiamento, tais como o NIH, nos EUA, exigem agora que os cientistas enviem seus dados estruturais para o PDB. Se o conteúdo do PDB é considerado como dados primários, existem diversas bases de dados derivadas (ou seja, secundárias), que classificam as proteínas de formas diferentes. Por exemplo, tanto SCOP quanto o CATH classificam as estruturas de acordo com o tipo de estrutura, assumindo relações evolutivas.

4.3 Métodos de classificação de domínios de proteínas

Obter a função biológica de uma proteína é essencial para determinar o seu potencial como alvo terapêutico e sua utilidade como parte de drogas baseadas em seu design. Além disso, a compreensão da função biológica de uma proteína fornece a base para explorar a sua atividade celular. Um dos resultados dos esforços na área de genômica tem sido um enorme crescimento de sequências de proteína experimentalmente definidas, porém com uma grande lacuna com relação à classificação de suas funcionalidades.

A homologia na sequência das proteínas tem sido rotineiramente utilizada como uma abordagem rápida para atribuir a função biológica das mesmas [66]. Esta abordagem é baseada em um paradigma da biologia estrutural, que consiste no fato de que a similaridade na sequência (30%) implica em uma similaridade correspondente tanto na estrutura quanto na função proteica. Na melhor das hipóteses, a homologia da sequência permite uma atribuição funcional para cerca de 50% das proteínas identificadas [66, 67].

A homologia estrutural é um método sensível para a atribuição de funções, uma vez que existem inúmeros exemplos de proteínas com enovelamentos similares que não possuem qualquer homologia sequencial significativa [62]. Este fato é consistente com a observação geral de que as estruturas terciárias são significativamente mais estáveis do que as sequências evolutivas das proteínas [68].

Sendo assim, uma vez que a identificação correta das estruturas terciárias (domínios) das proteínas parece ser a chave para a caracterização das funções proteicas, diversos esforços têm sido feitos neste sentido. Novas bases de dados dedicadas à identificação de domínios vêm surgindo, a fim de auxiliar diversos pesquisadores das mais variadas áreas biológicas; alguns dos métodos de classificação mais bem estabelecidos são o *Structural Classification of Proteins* ou SCOP, o *Class, Architecture, Topology, Homology* ou CATH, o *Protein Domain Parses* (ou PDP) e o “Domain Parser” (Domain Parser).

4.3.1 CATH – *Class, Architecture, Topology, Homology*

CATH é uma base de dados para classificação de estruturas de proteínas baseada num método semiautomático de classificação hierárquica de domínios da proteína, publicado em 1997 por Christine Orengo, Janet Thornton e colegas [69]. O CATH divide muitas características gerais com o SCOP, porém também há muitas áreas em que a classificação detalhada de ambos é muito diferente. A base de dados CATH é uma classificação hierárquica de domínio das estruturas das proteínas do Protein Data Bank. Apenas estruturas cristalinas observadas com uma resolução melhor do que 4 Å são considerados, em conjunto com estruturas de RMN. Todas as não proteínas, modelos e estruturas com mais de 30% de “somente C_α ” são excluídas do CATH. As estruturas de proteínas são classificadas usando uma combinação de processos automatizados e manuais. Há quatro principais níveis de hierarquia: Classe, Arquitetura, Topologia (família

de enovelamento) e super-família homóloga (Orengo *et al.*, 1997). Cada nível será descrito abaixo, juntamente com os métodos utilizados para definir limites de domínio e atribuição de estruturas para uma família específica.

Toda a classificação é realizada em domínios individuais da proteína. Para dividir uma proteína de acordo com os seus domínios constituintes, são utilizadas uma combinação de técnicas automáticas e manuais. Se uma dada cadeia de proteína possui uma similaridade estrutural e identidade sequencial suficientemente alta (ou seja, 80% de similaridade da sequência e pontuação SSAP¹ ≥ 80), com uma cadeia que tenha sido previamente destacada, a atribuição dos limites do domínio é feita automaticamente, herdando os limites da outra cadeia previamente analisada. Caso contrário, os limites do domínio são atribuídos manualmente, com base na análise dos resultados provenientes de uma série de algoritmos que incluem métodos estrutura base tais como CATHE-DRAL [66], SSAP [71, 70], DETECTIVE [62], PUU [72], DOMAK [73].

A classificação das estruturas do domínio da proteína através do método CATH foi estabelecido como sendo um agrupamento hierárquico de estruturas do domínio da proteína, dependendo da sequência e similaridade de estrutura. Existem quatro níveis principais de classificação, correspondentes à classe de proteínas, arquitetura, topologia ou enovelamento e famílias homólogas (Fig. 4.8).

Uma vez que o CATH está baseado em quatro níveis de classificação, estes possuem certas regras e descrições, as quais são apresentadas resumidamente a seguir:

- **Classe ou nível C:** é determinado de acordo com a composição em termos das estruturas secundárias e de empacotamento da estrutura. Três classes principais são reconhecidas; aquelas compostas por estruturas α , por estruturas β , e, finalmente estruturas compostas por folhas β e hélices α . Uma quarta classe ainda é considerada, definida como contendo proteínas de domínios que possuem baixo conteúdo de estruturas secundárias.
- **Arquitetura ou nível A:** este nível descreve a forma geral da estrutura de domínio, conforme determinado por orientações das estruturas secundárias, porém ignorando a conectividade entre as estruturas secundárias. Atualmente, este é atribuído manualmente usando uma simples descrição do arranjo da estrutura secundária, como por exemplo o barril ou o sanduíche de três camadas.
- **Topologia ou nível T:** as estruturas são agrupadas de acordo com as que compartilham da mesma topologia ou enovelamento no núcleo do domínio, ou seja, se estas

¹SSPA - *Sequential structure alignment program for protein structure comparison*. O método SSAP usa programação dinâmica dupla para produzir um alinhamento estrutural baseado em vetores de átomo para átomo da estrutura. Em vez dos C_α , geralmente usados no alinhamento estrutural, o método SSAP constrói seus vetores a partir de C_β , levando em conta a rotação de cada resíduo, bem como a sua localização ao longo do esqueleto [70].

compartilham a mesma forma geral das estruturas secundárias e de conectividade no núcleo de domínio.

- **Famílias homólogas (super-famílias) ou nível H:** este nível agrupa domínios da proteína os quais se acredita que compartilham um ancestral comum e podem ser descritos como homólogos. As semelhanças são identificadas quer pela alta identidade de sequência ou pela comparação da estrutura utilizando-se o SSAP.

Figura 4.8: Representação esquemática da (C)lasse, (A)rquitetura e (T)opologia – níveis de agrupamento da base de dados CATH. Fonte: <http://koehl1ab.genomecenter.ucdavis.edu/teaching/ecs129/07>

Desde 1995, as informações sobre esses grupos de famílias estruturais de proteínas está acessível através do sítio <http://www.cathdb.info/>, além de informações resumidas sobre cada estrutura da proteína individual apresentadas no PDB.

4.3.2 SCOP – *Structural Classification of Proteins*

A classificação estrutural de proteínas presente na base de dados SCOP é realizada predominantemente de maneira manual, classificando a maior parte dos domínios com base em semelhanças de suas sequências de aminoácidos e estruturas tridimensionais. Originalmente publicado em 1995 [74], a base de dados SCOP é normalmente atualizada pelo menos uma vez por ano por Alexei G. Murzin *et al.* [65, 66], cuja competência cabe à classificação dos domínios.

A base de dados SCOP visa fornecer uma descrição detalhada e abrangente das relações estruturais e evolutivas entre todas as proteínas cuja estrutura é conhecida, incluindo todas as entradas do PDB.

Uma vez que as ferramentas de sequência automática e de comparação de estrutura existentes não podem identificar todas as relações estruturais e evolutivas entre proteínas, a classificação SCOP foi construída manualmente através de inspeção visual e comparação de estruturas, mas com o auxílio de ferramentas para tornar a tarefa factível e ajudar a fornecer generalidade. O trabalho se torna ainda mais desafiador e teoricamente, assustador pelo fato de que as entidades a serem organizadas não são homogêneas: às vezes faz mais sentido organizá-las por domínios individuais, e outras vezes, por completas proteínas com multi-domínios.

As proteínas são classificadas com o objetivo de refletir tanto as relações estruturais quanto as relações evolucionárias. Existem diversos níveis hierárquicos, porém os principais níveis são família, super-família e enovelamento, que serão descritos abaixo e estão esquematizados na Fig. 4.9. A posição exata dos limites destes níveis possuem um certo grau de subjetividade.

Figura 4.9: Esquema da hierarquia SCOP. Todos os níveis principais, incluindo classe, enovelamento, super-família, família e domínio são apresentados. Também são mostradas as proteínas individuais e com o menor nível, sendo o PDB a referência das entradas. (Steven E. Brenner, 1994)

A classificação evolutiva do SCOP é geralmente conservadora: quando aparece qualquer dúvida sobre a existência de parentesco, novas divisões no nível da família e da super-família são feitas. Assim, alguns pesquisadores preferem se concentrar nos níveis superiores da árvore de classificação, onde as proteínas com similaridade estrutural estão agrupadas. Os diferentes níveis mais importantes na hierarquia são:

- **Família** (claras relações evolutivas): proteínas agrupadas em famílias estão claramente relacionadas evolutivamente. Geralmente, isso significa que as identidades de pares de resíduos são de pelo menos 30%. No entanto, em alguns casos, as funções e estruturas semelhantes podem oferecer a prova definitiva da origem comum, na falta de alta identidade da sequência, como por exemplo as globinas, que formam uma família em que alguns membros têm identidades de sequência de apenas 15%.
- **Super-família** (provável origem evolutiva comum): proteínas que têm baixas identidades sequenciais mas cujas características estruturais e funcionais sugerem uma provável origem evolutiva comum, são colocadas juntas em super-famílias.
- **Enovelamento** (principalmente semelhanças estruturais): as proteínas são definidas como tendo um enovelamento comum, possuindo a maior parte das estruturas secundárias semelhantes, com a mesma ordem e com as mesmas conexões topológicas. Diversas proteínas com o mesmo enovelamento frequentemente possuem elementos periféricos de estrutura secundária e regiões de voltas que podem diferir em tamanho e conformação. Em alguns casos, essas regiões periféricas diferentes podem incluir apenas meia estrutura. Proteínas colocadas juntas na mesma categoria de enovelamento, as vezes não possuem uma origem evolutiva comum: as semelhanças estruturais podem surgir a partir da física e da química das proteínas, favorecendo certas modalidades de empacotamento e topologias de rede.

Esta classificação é significativamente mais baseada na experiência humana do que o método CATH (semiautomático). Geralmente, é aceito que o SCOP apresenta uma classificação melhor justificada. A experiência humana é necessária para decidir se certas proteínas estão evolutivamente relacionadas e portanto, devem ser atribuídas à super-família, ou se sua semelhança é resultado dos efeitos estruturais e logo, pertencem ao mesmo tipo de enovelamento.

4.3.3 DP - *Domain Parser*

O método **Domain Parser** consiste em um algoritmo para a divisão de uma proteína com múltiplos domínios em domínios estruturais individuais. O princípio subjacente usado é o contato entre resíduos uma vez que estes são mais densos do que no caso dos contatos entre domínios. O problema da divisão de domínios, foi abordado com sendo um problema de redes complexas, no qual cada resíduo é representado por um vértice em uma rede e cada contato entre os resíduos é representado como uma aresta, com uma capacidade específica, dependendo do tipo de contato [75].

Figura 4.10: Exemplos de atribuições de domínios pelo método DomainParser.

Uma conexão entre dois resíduos é adicionada quando um destes encontra-se próximo a outro considerando-se a estrutura tridimensional da proteína. A proximidade espacial entre os dois resíduos é estabelecida caso a distância entre pelo menos um par de átomos (um de cada resíduo) seja de até $4,0\text{\AA}$. A força da conexão (referida como capacidade) entre dois vértices é proporcional à força da interação entre os dois resíduos representados por tais vértices.

Um problema de decomposição em dois domínios é solucionado buscando-se uma divisão da rede; o algoritmo de Ford-Fulkerson clássico é utilizado para localizar uma divisão mínima de uma rede. Para proteínas com múltiplos domínios, o algoritmo repetidamente aplica o algoritmo de Ford-Fulkerson para bipartição da rede, formando uma árvore de partição até que algum critério de parada seja satisfeito [75].

A etapa de pós-processamento é utilizada para fundir os segmentos que podem ter

sido cortados, com base em certas propriedades geométricas e físicas de um domínio. Os parâmetros incluem a compactação, o raio de giração, o número de segmentos não adjacentes por domínio e a distribuição de tamanhos de domínio. Um exemplo das atribuições de domínio através do método **Domain Parser** é apresentada da Fig. 4.10.

Por fim, o código fonte do programa referente ao **Domain Parser** não é disponibilizado e os autores do **Domain Parser** não fazem menções sobre a adequação do deste a qualquer finalidade. Tal programa é fornecido “como está” e sem garantia expressa ou implícita [75].

Capítulo 5

Método de identificação de comunidades em redes complexas

5.1 O método AL

O método desenvolvido por Alves [14] para identificação de comunidades em redes complexas, o qual designamos AL, é um método determinístico, inspirado no método espectral e que combina caminhadas aleatórias em grafos de redes elétricas. Neste método as arestas entre os vértices são consideradas como sendo resistores e, a partir disto, é possível definir uma matriz de probabilidade de transição Markoviana em termos das condutâncias elétricas.

Por meio da matriz adjacência $\mathbf{A}$ do grafo G , obtemos o grau de cada vértice, definido na Eq. 3.1, o qual utilizamos para calcular a matriz Laplaciana $\mathbf{L}(G)$ [76], dada por

$$L_{ij} = \begin{cases} k_i, & \text{caso } i = j, \\ -1, & \text{caso } i \text{ e } j \text{ sejam adjacentes,} \\ 0, & \text{caso contrário.} \end{cases} \quad (5.1)$$

Em um grafo conexo, o agrupamento dos vértices em comunidades, pode ser obtido utilizando-se as componentes do autovetor correspondente ao primeiro autovalor não nulo da matriz Laplaciana [77]. Entretanto, o sucesso de tal método depende do quão separado este autovalor está em relação aos demais.

Considerando o grafo como sendo uma rede elétrica na qual as arestas são representadas por resistores, a caminhada aleatória em G corresponde à sequência de estados (vértices) escolhidos dentre seus vizinhos adjacentes.

A partir da matriz Laplaciana, podemos definir a resistividade r_{ij} entre todos os pares de vértices em função de seus n autovalores λ_i e autovetores v_i [78], como sendo

$$r_{ij} = \sum_{k=1}^{n-1} \frac{1}{\lambda_k} (v_{ki} - v_{kj})^2, \quad (5.2)$$

a partir da qual definimos, como uma generalização natural, a condutância efetiva entre os pares de vértices (i, j) ,

$$c_{ij} = \frac{1}{r_{ij}}, \quad \text{para } i \neq j. \quad (5.3)$$

Como consequência deste resultado, é possível estender o conceito usual de caminhada aleatória entre dois estados adjacentes i e j para saltos no grafo. Desta forma, definimos a matriz de probabilidade de transição $\mathbf{P}$ entre dois vértices (i, j) quaisquer, como sendo

$$P_{ij} = \frac{c_{ij}}{c_i}, \quad \text{com } c_i = \sum_j c_{ij}. \quad (5.4)$$

É importante ressaltar que devemos considerar apenas grafos conexos para que tal probabilidade esteja definida.

Dada esta probabilidade, podemos definir a distância média entre dois vértices (i, j) , como sendo

$$\bar{d}_{ij} = \frac{\sum_{k \neq i, j}^n |P_{ik} - P_{jk}|}{n - 2}. \quad (5.5)$$

Uma vez que P_{ij} não é necessariamente igual a P_{ji} , esta distância não é adequada para definir uma escala para classificação hierárquica dos vértices em comunidades. Entretanto, verificou-se que a flutuação S_{ij} de $|P_{ik} - P_{jk}|$ desempenha o papel mais importante em tal classificação. Sendo assim, podemos escrever o desvio padrão entre dois vértices como sendo

$$S_{ij} = \left(\frac{1}{n - 3} \sum_{k \neq i, j}^n \left[|P_{ik} - P_{jk}| - \bar{d}_{ij} \right]^2 \right)^{\frac{1}{2}}. \quad (5.6)$$

O desvio padrão S_{ij} fornece a descrição da similaridade entre os vértices. Explorando o comportamento de P_{ij} , é possível notar que uma baixa probabilidade de transição do estado i para o estado j significa que j é menos acessível por parte do estado i . Contrariamente, uma alta probabilidade de transição entre estados define classes de estados facilmente conectados.

Uma vez que os elementos de P_{ij} não são necessariamente simétricos, é possível descrever o quão próximos os vértices i e j estão definindo-se distância sob a forma $1/P_{ij}^{\max}$, na qual $P_{ij}^{\max}$ corresponde ao máximo valor entre P_{ij} e P_{ji} .

Similarmente, $P_{ij}^{\min}$ corresponde ao valor mínimo entre P_{ij} e P_{ji} . Dessa forma, a matriz dissimilaridade $\mathbf{D}$ (ou alternativamente, “matriz distância”), pode ser definida

como

$$D_{ij} = S_{ij} \frac{\max\{\Delta_{ij}, P_{ij}^{\min}\}}{P_{ij}^{\max}}, \quad \text{na qual} \quad \Delta_{ij} \equiv P_{ij}^{\max} - P_{ij}^{\min}. \quad (5.7)$$

Os termos Δ_{ij} e $P_{ij}^{\min}$ são levados em consideração sob a forma de penalidade quando $P_{ij}^{\min}$ é elevado, contribuindo assim, para a definição de uma escala de similaridade. Desta forma, verifica-se que a matriz $\mathbf{D}$ é capaz de revelar a estrutura hierárquica dos estados, ou seja, a estrutura de comunidades da rede.

Definida uma escala de dissimilaridade, podemos representar a estrutura hierárquica dos vértices de uma rede por meio de uma árvore hierárquica, ou dendrograma [79]. Na Fig. 5.1 é apresentado o dendrograma para a rede do clube de karatê [80], com 34 vértices e 78 arestas. Nesta rede, podemos observar que para diferentes níveis de dissimilaridade, divisões distintas da mesma rede são possíveis, o que resulta em comunidades distintas. Como exemplo, ao escolher o valor de dissimilaridade entre aproximadamente 75–100%, vemos que, de acordo com métrica definida pela matriz de distâncias, esta rede é dividida em duas comunidades. Escolhendo níveis de dissimilaridade inferiores a 75%, novas divisões desta rede tornam-se possíveis. Se considerarmos o valor zero da escala de dissimilaridade, cada vértice então, passa a ser considerado como sendo uma comunidade distinta.

Figura 5.1: Dendrograma obtido para a rede do clube de karatê usando a escala de dissimilaridades fornecida pelo método AL. Esta rede é composta por 34 vértices e 78 arestas.

Para a geração de um dendrograma, a matriz dissimilaridade deve ser processada por um algoritmo de agrupamento aglomerativo que organize tal informação para análise. No caso da Fig. 5.1, a matriz dissimilaridade foi fornecida como entrada no programa

de análise estatística R [81] e a árvore hierárquica foi produzida utilizando-se o método aglomerativo hierárquico de ligação completa (*complete linkage*).

Em seu estudo, Zachary concluiu que esta mesma divisão (em dois grupos distintos), foi o resultado da cisão do referido clube [80]. Esta divisão, a qual consideramos como divisão verdadeira, corresponde a dois grupos de alunos. De um lado, os alunos que permaneceram no clube por ter maior afinidade com o administrador do mesmo, e do outro, o grupo de alunos que seguiu o instrutor de karatê em seu novo clube. Desta forma, consideramos que o método AL foi capaz de revelar a verdadeira divisão da rede do clube de karatê.

Ao analisar um dendrograma, é desejável que a divisão escolhida como correta esteja suficientemente resolvida, ou seja, a diferença de dissimilaridade entre a melhor divisão e a próxima divisão possível seja superior a quaisquer outras divisões. Sendo assim, definimos esta diferença de dissimilaridades como sendo a coesão da tal divisão. Logo, quanto maior esta diferença, maior será a coesão da referida comunidade.

Considerando ainda o dendrograma mostrado na Fig. 5.1, vemos que a divisão em duas comunidades corresponde à melhor divisão desta rede. Observando a escala de dissimilaridades no dendrograma, vemos que a divisão em duas comunidades está compreendida no intervalo de $[100, 0 - 74, 6[$, cuja diferença corresponde a 25,4. Considerando a divisão em três comunidades, a qual compreende-se no intervalo de $[53, 7 - 74, 6]$, a diferença corresponde a 20,9. Já as demais divisões candidatas a soluções possuem valores de diferença bastante inferiores quando comparados àquelas das divisões em 2 e 3 comunidades, portanto não são consideradas candidatas à melhor solução. Como a diferença na escala de dissimilaridades para 2 comunidades é ligeiramente superior à diferença referente a 3 comunidades (aproximadamente 4,5), consideramos que esta seja a melhor divisão.

Em outras redes, entretanto, a determinação da melhor divisão utilizando a escala de dissimilaridade pode não ser tão evidente. Isto faz com que seja necessário considerar outros fatores para a determinação da melhor divisão. Um exemplo deste fato seria medir a qualidade de uma dada divisão por meio de alguma função objetivo. Outro ponto a ser considerado, consiste no fato de que excedendo-se o limite de algumas dezenas de vértices, a inspeção visual da rede, bem como a inspeção visual do dendrograma gerado torna-se pouco prática ou até mesmo inviável. Este fato reforça a necessidade de se utilizar alguma função objetivo capaz de revelar a melhor divisão em comunidades de uma dada rede.

Uma possível candidata à função objetivo, a qual vem sendo amplamente utilizada é a função modularidade (Q) proposta por Newman [25] e definida na Eq. 3.21. Dada uma possível divisão da rede em comunidades, calcula-se o valor desta função e espera-se que o maior valor possível de Q para uma rede, corresponda à sua melhor divisão em comunidades considerando-se a definição de comunidades (Def. 1 apresentada na

Sec. 3.5).

No caso do método AL (que fornece diversas possíveis divisões para uma rede), pode-se calcular o valor da modularidade para cada nível de dissimilaridade que resulte em uma divisão distinta para a dada rede. A melhor a divisão corresponderia ao máximo de Q .

Figura 5.2: A máxima modularidade obtida pelo método AL para a rede do clube de karatê revela 3 comunidades tendo $Q = 0,399$. As três comunidades são definidas por meio da forma geométrica utilizada para representar os vértices. As linhas pontilhadas são utilizadas para mostrar a divisão em duas comunidades como determinada pelo método AL e considerada como verdadeira por Zachary.

Neste caso, contrariando o que é evidenciado no dendrograma através das diferenças entre níveis de dissimilaridade, a função modularidade máxima obtida dentre as possíveis divisões fornecidas pelo método AL, corresponde a $Q = 0,399$, ou seja, 3 comunidades. Analisando esta divisão por inspeção visual da rede mostrada na Fig. 5.2 e considerando simplesmente a definição de comunidades, podemos concluir esta divisão é superior à divisão em duas comunidades. Pois, nesta divisão, o número de arestas internas à comunidade definida pelos vértices 5, 6, 7, 11 e 17 totaliza $m = 6$, o que é superior ao número de arestas externas à comunidade $m = 4$.

Neste ponto, devemos ressaltar que de acordo com a definição de comunidades, o valor máximo da modularidade, dentre as alternativas fornecidas pelo método AL, não fornece a divisão real da rede assim como entendida por Zachary, mas sim a configuração que melhor divide a rede segundo a definição de comunidades.

Na Sec. 5.5 serão apresentados e discutidos os resultados obtidos pelo o método AL para outras redes sociais comumente utilizadas.

5.2 Agrupamento aglomerativo hierárquico

Como mencionado na Sec. 5.1, o método AL apresenta os seus resultados sob a forma de uma matriz de dissimilaridades, a qual define uma métrica que pode ser utilizada para identificar comunidades. Para tal identificação, é necessário utilizar algum método de agrupamento aglomerativo hierárquico, para construir uma árvore hierárquica, como a mostrada na Fig. 5.1.

Diversos aplicativos especializados em análise estatística disponibilizam rotinas para agrupamento hierárquico e geração de dendrogramas. Entretanto, utilizar aplicativos específicos que fornecem ambientes para análise de dados como R, Scilab ou Matlab, apresentam alguns inconvenientes do ponto de vista computacional. Alguns exemplos destes inconvenientes são: interação por meio de uma interface, gráfica ou linha de comando, utilização de linguagens interpretadas, difícil integração com aplicativos que utilizem linguagens de mais baixo nível e baixo desempenho computacional.

A interface com o usuário, por meio de uma interface gráfica ou até mesmo linha de comando, limita a interação com o programa por meio de aplicativos desenvolvidos utilizando linguagens de mais baixo nível como C++, por exemplo. Esta limitação, em geral, pode ser amenizada por meio da elaboração de scripts, os quais podem ser executados por meio de chamadas de sistema e que por sua vez introduzem outros inconvenientes. Outro fator a ser considerado, é o desempenho computacional inferior inerente à utilização de aplicativos externos e, acima de tudo, de linguagens interpretadas (usadas em scripts). No caso do Matlab, o fato dele ser um aplicativo proprietário de código fechado, torna-o uma “caixa preta” cuja utilização prejudica a transparência e divulgação de trabalhos científicos.

O ambiente R – *R Language and Environment for Statistical Computing* [82, 81, 83], amplamente utilizado para análise dos dados desta tese, é capaz de gerar dendrogramas como o mostrado na Fig. 5.1, por meio da função `hclust` [84, 85]. Esta função, implementada por Fionn Murtagh na linguagem FORTRAN e integrada ao R por meio da biblioteca `StatLib`¹, é descrita em detalhes no trabalho intitulado *Multidimensional Clustering Algorithms* (Algoritmo E, página 105), publicado em 1985 [84], tendo por sua vez, complexidade assintótica estimada em $\mathcal{O}(n^2)$. Fornecendo uma matriz de distâncias para a função `hclust`, o resultado é apresentado sob a forma gráfica em um dendrograma o qual pode ser analisado por inspeção visual. Em casos cujo interesse é utilizar as informações contidas no dendrograma de forma automatizada – como no caso do cálculo do valor da função modularidade para cada comunidade possível – a representação gráfica não é adequada.

Por estes motivos, desenvolvemos um método de agrupamento aglomerativo hierárquico, o qual chamamos de HAC++, ou *Hierarchical Agglomerative Clustering in C++*, capaz

¹<http://lib.stat.cmu.edu/>

de lidar com tais questões a um custo computacional inferior. A motivação principal para tal desenvolvimento é poder utilizar a informação estrutural do dendrograma de forma integrada ao método AL sem que haja a necessidade de efetuar inspeções visuais. Isto resulta na possibilidade de obter as divisões possíveis da rede, tornar viável o cálculo direto de funções objetivo e transcrever a informação estrutural sob uma representação adequada que torne possível fornecê-la a algum outro método para manipulações posteriores.

5.2.1 HAC++ – *Hierarchical Agglomerative Clustering in C++*

Algoritmos de agrupamento aglomerativo hierárquico são objetos de estudo em áreas de estatística, mineração de dados, análise de imagens e bioinformática. Dado que o tamanho dos conjuntos de dados a serem analisados vêm crescendo, esforços têm sido feitos no intuito de criar algoritmos cada vez mais eficientes [86, 87, 88].

O método desenvolvido neste trabalho baseia-se naquele proposto por Walter *et al.* [88], o qual foi adaptado para utilizar uma estrutura de dados que forneça um melhor desempenho na manipulação de grandes quantidades de dados.

A estrutura de dados desenvolvida, e implementada em C++, consiste na combinação de duas estruturas de dados comumente utilizadas: listas duplamente ligadas e árvores binárias [89]. Uma representação esquemática de tal estrutura de dados é mostrada na Fig. 5.3. Para a implementação desta estrutura de dados foi criada uma classe chamada `Clusters_C` a qual será apresentada em detalhes na Sec. 5.3.

Figura 5.3: Representação esquemática da estrutura de dados desenvolvida para utilização no algoritmo de agrupamento hierárquico – HAC++.

Esta estrutura utiliza quatro ponteiros, sendo dois para a lista duplamente ligada (ponteiros `previous` e `next`) e dois para a árvore binária (ponteiros `left` e `right`). Os apontadores `previous` e `next`, como seus nomes sugerem, mantêm referências para os objetos anteriores e posteriores da lista, respectivamente. Os apontadores `left` e `right`, por sua vez, mantêm referências para os ramos à esquerda e à direita da árvore hierárquica, respectivamente. Além destes ponteiros, cada objeto dessa classe possui um identificador único, o qual corresponde ao número do vértice principal (`main_node`) armazenado em tal objeto. É considerado como vértice principal aquele cujo número na rede seja o menor dentre os vértices armazenados no respectivo objeto. Ou seja, se

um objeto da classe `Clusters_C` é responsável pelos vértices 2, 5 e 15, o `main_node` será o vértice 2. Além disso, a lista dos vértices agrupados ficará armazenada no vetor `member_nodes`. Outra informação importante, armazenada em tais objetos, corresponde ao valor da dissimilaridade em que o agrupamento ocorreu. Sendo assim, no início do processo aglomerativo, haverá uma instância da classe `Clusters_C` para cada vértice da rede, ao nível de dissimilaridade 0, sendo que cada objeto possui referências para os objetos anteriores e posteriores, não havendo nesse momento ramos na árvore hierárquica. Na Fig.5.4 está apresentado um esquema da lista de vértices após ocorrer o primeiro agrupamento.

Figura 5.4: Representação esquemática da situação da estruturas de dados após ocorrer a primeira aglomeração.

Neste esquema vemos a estrutura de lista duplamente ligada com o vértice 1 a encabeçando. Por este motivo, o apontador `previous` deste objeto não está sendo usado (apontamento nulo) e o apontador `next` possui referência para o objeto 2. Além disso, o objeto `main_node = 1` está ao nível de dissimilaridade 0, ao passo que o objeto `main_node = 2`, apontado pelo objeto `main_node = 1`, está a um nível de dissimilaridade diferente de 0, uma vez que há ramos abaixo dele considerando a árvore binária. Quando dois objetos são aglomerados, ou seja, quando

$$d(i, j) \leq D, \quad (5.8)$$

sendo $d(i, j)$ a dissimilaridade (ou distância) entre os vértices i, j e D o nível de dissimilaridade em consideração, é criado um novo objeto que assume o lugar do objeto de menor valor de `main_node`, mas que tem como `member_nodes` a relação dos `main_node` de todos os objetos aglomerados. No exemplo da Fig. 5.4, vemos que há dois objetos com `main_node = 2`, entretanto estes objetos são completamente diferentes. O objeto

`main_node = 2` que é apontado pelo apontador `left` do outro objeto com `main_node = 2`, tem `cutoff_height = 0` e `member_nodes = 2`, ao passo que o `main_node = 2` que é apontado pelo objeto com `main_node = 1`, possui `cutoff_height ≠ 0` e `member_nodes = 2` e 4.

Além disso, quando ocorre o agrupamento de objetos, a matriz de dissimilaridades deve ser atualizada de modo a refletir a situação do grupo aglomerado em relação ao restante dos objetos. Esta atualização corresponde à remoção da linha e coluna referente a um dos objetos, e atualização da linha e coluna do objeto remanescente, de acordo com o critério de aglomeração adotado. Ao utilizar o método de **agrupamento por ligação completa**, e utilizando o objeto `main_node = 2` (que corresponde ao grupo de objetos 2 e 4, por exemplo), os elementos da matriz correspondentes aos vértices 2 e 4, D_2 e D_4 , são comparados e aglomerados no novo elemento D_2^* seguindo o critério

$$D_{2j}^* = \begin{cases} D_{2j} & \text{caso } D_{2j} > D_{4j}, \\ D_{4j} & \text{caso } D_{2j} < D_{4j}. \end{cases} \quad (5.9)$$

O procedimento é finalizado quando todos os objetos forem agrupados. Em termos do algoritmo implementado nesta tese, no início do procedimento é criado um objeto da classe `Clusters_C` para cada vértice, sendo que tais objetos são adicionados à lista duplamente ligada. Como a cada aglomeração de objetos, a lista é reduzida em um objeto, e são adicionados dois ramos à árvore binária, o final da rotina ocorre quando a lista ligada tiver apenas um único objeto e por sua vez, a árvore binária estiver completa. Na Fig. 5.5 é apresentado um diagrama esquemático da situação da estrutura de dados no final do processo aglomerativo.

Vemos, neste caso, que todos os apontadores da lista ligada não estão sendo mais utilizados (apontamentos nulos) e que apenas os apontadores da lista binária estão em uso, sendo que os objetos ao nível 0 de dissimilaridade não possuem ramos (apontamentos `left` e `right` nulos).

Após o processo de agrupamento ter sido completado, a varredura da árvore resultante pode ser efetuada, extraindo-se quaisquer informações pertinentes à análise de um dendrograma, como nível de dissimilaridade, agrupamentos em cada nível, etc. Isto permite que para cada nível de dissimilaridade seja efetuado o cálculo da função objetivo de interesse e até mesmo a geração de vetores que representem a estrutura de comunidades.

Figura 5.5: Representação esquemática da estrutura de dados na situação final do processo aglomerativo.

5.3 Implementação

O problema de determinação de comunidades em redes complexas tem recebido grande atenção por parte da comunidade científica dado que este campo possui aplicações em uma grande variedade de áreas e uma vasta gama de aplicações tecnológicas. Em biologia por exemplo, pode-se utilizar redes complexas para modelar as interações intra e entre espécies de animais, como na rede criada por Lusseau [90] a qual estuda os relacionamentos em uma população de golfinhos e seus padrões de migração. Em epidemiologia, redes de contato são utilizadas para estudar como certa doença se espalha em uma população, sendo a detecção de comunidades algo de extrema importância para a determinação de políticas de vacinação [91, 92, 93]. Do ponto de vista tecnológico, existem diversos exemplos de aplicações de redes complexas e identificações de comunidades. Tais exemplos são: mecanismos de busca de páginas de internet, corretores ortográficos, redes de telefonia e até mesmo as mais recentes tecnologias de bancos de dados denominada **NoSQL**. Nesta última, as tabelas convencionais são substituídas por grafos o que resulta em uma enorme escalabilidade e excelente desempenho [94, 95, 96, 97, 98, 99].

Uma grande fração dos trabalhos envolvendo redes complexas descrevem estratégias interessantes para a detecção de comunidades, porém estes trabalhos não fornecem detalhes suficientes para que seja possível implementá-las de forma adequada. Além disso, para que seja possível efetuar comparações relevantes entre métodos, cada interessado necessita implementar tais estratégias, o que consome tempo e muitas vezes não se atinge

o mesmo grau de eficiência reportado nos artigos de origem, prejudicando a reprodutibilidade dos resultados obtidos.

Por estes motivos, todos os algoritmos desenvolvidos, ou simplesmente implementados neste trabalho, foram disponibilizados como software livre e de código aberto, registrados sob a licença GPLv3 – *GNU General Public License* (Ap. B). Desta forma, garantimos livre acesso às ferramentas aqui desenvolvidas propiciando que outros autores a utilizem, descubram seus pontos fortes, bem como os pontos fracos, e façam adaptações. Como resultado, esperamos encorajar que novos métodos sejam divulgados abertamente e que os progressos nesta área sejam acelerados.

5.3.1 graphstk – *The Graphs Toolkit*

As aplicações desenvolvidas neste trabalho foram agrupadas, em um projeto de software livre e de código aberto, o qual nomeamos **graphstk** – *The Graphs Toolkit*. As informações a respeito deste projeto, bem como os respectivos códigos fontes, podem ser obtidas através do endereço eletrônico <http://code.google.com/p/graphstk>, o qual corresponde à página oficial do projeto **graphstk**.

Escolhemos a linguagem C++ como padrão para implementação dos algoritmos. Os principais motivos desta escolha foram: desempenho, escalabilidade, disponibilidade de recursos, orientação a objeto e ampla adoção pela comunidade científica. Além disso, **graphstk** utiliza recursos oferecidos por duas outras bibliotecas, também disponíveis como software livre e de código aberto, que oferecem diversos recursos pertinentes às implementações, como *arrays* n-dimensionais, ponteiros inteligentes – *smart pointers* – geradores de números pseudo-aleatórios, além de rotinas para o cálculo de autovalores e paralelização. Tais bibliotecas são a GNU-GSL [100] – *GNU Scientific Library* – e *Sco-pira* [101].

O ponto central da **graphstk** é o modo de manipulação dos grafos. Esta manipulação é efetuada por meio da classe **Graphs_C**, na qual utilizamos um *array* de duas dimensões para armazenar a matriz adjacência e para construir diversos métodos para manipulação da mesma. Além disso, cálculos básicos, como a determinação do grau dos vértices, assortatividade, bem como verificações de simetria e consistência também foram implementados. Dessa forma, cada método que utilize esta classe pode simplesmente estendê-la, adicionando especializações pertinentes a cada caso. Na implementação do método AL, por exemplo, tais especializações compreendem o cálculo e armazenamento das matrizes de condutâncias, probabilidade de transição e distância. A listagem 5.1, mostra a declaração da classe **Graphs_C**.

```

1  /* License: GPLv3 * Copyright Luiz C. B. Mostaco-Guidolin */
2  class graphstk::Graphs_C : public scopira::tool::object
3  {
4  private:
5      size_t num_of_nodes_ui_;
6      size_t num_of_edges_ui_;
7
8      narray< size_t, 1 > vertices_degree_na_;
9      narray< size_t, 2 > adjacency_matrix_na_;
10     narray< size_t, 2 > laplacian_matrix_na_;
11
12 public:
13     Graphs_C();
14     ~Graphs_C();
15
16     void CalcVerticesDegree();
17     void GenLaplacianMatrix();
18     void GenListOfLinks();
19     void LoadAdjacencyMatrix
20     (
21         string file_name_s__,
22         size_t num_of_nodes_ui__
23     );
24     void LoadAdjacencyMatrix( narray< size_t, 2> &A );
25     void PrintAdjacencyMatrix();
26     void PrintVerticesDegree();
27     void PrintSummary();
28     void Resize( size_t num_of_nodes );
29     void SetEdgeValue
30     (
31         size_t node_A_ui__,
32         size_t node_B_ui__,
33         size_t value_ui__
34     );
35     void TestAdjacencyMatrixSymetry();
36     void TestAdjacencyMatrixDiagonal();
37
38     size_t GetEdgeValue( size_t node_A_ui__, size_t node_B_ui__ );
39     size_t GetNodeDegree( size_t node_ui__ );
40     size_t GetNumOfEdges();
41     size_t GetNumOfNodes();
42
43     narray< size_t, 2 > GetAdjacencyMatrix();
44
45     Graphs_C& operator= ( Graphs_C& rhs );
46 };

```

Algoritmo 5.1: Declaração da classe Graphs_C em C++.

Esta classe, herda elementos da classe `scopira::tool::object`, a qual fornece a infraestrutura básica para que seja possível criar *smart pointers* da classe `Graphs_C`. Na listagem 5.1, vemos alguns métodos, tais como: `LoadAdjacencyMatrix()`, `GetEdgeValue()` e `GenListOfLinks()`, por exemplo. Tais métodos são utilizados, respectivamente, para ler a matriz adjacência a partir de um arquivo, acessar os valores das arestas na matriz adjacência e, converter a matriz adjacência para a representação de lista de arestas. A representação sob a forma lista de arestas é outra forma bastante utilizada na distribuição dos dados de uma determinada rede, comumente utilizada por diversos autores. Além disso, diversos autores utilizam a matriz Laplaciano em seus métodos de detecção de comunidades, sendo por isso o suporte a esta matriz incluso direto na classe `Graphs_C`.

Considerando o método `AL`, foi necessário implementar diversos outros métodos para manipulação das redes. O termo método pode ser usado para designar uma metodologia ou algoritmo, bem como para designar uma função membro de uma classe.

A classe `ALGraphs_C`, mostrada na listagem Alg. 5.2, é uma extensão classe `Graphs_C`, na qual foram introduzidas especializações como: espaços de armazenamento e rotinas para cálculos das matrizes de condutância, probabilidade de transição, distância média entre vértices, desvio padrão da matriz de distâncias médias e, finalmente, matriz dissimilaridade. As definições correspondentes encontram-se nas linhas 9 a 13, 27 a 29, 31 e 32 da listagem Alg. 5.2.

Uma particularidade do método `AL` é a remoção das chamadas folhas, ou seja, vértices com grau $k = 1$, uma vez que classes de estados transientes bem definidas somente são possíveis em grafos sem estruturas locais de árvore [102]. Na Fig. 5.6, é apresentado um grafo como exemplo, contendo um subgrafo formado pelos vértices 5, 6 e 7.

Figura 5.6: Grafo exemplo, contendo um subgrafo com estrutura de árvore (vértices 5, 6 e 7).

Considerando que há apenas um caminho possível para se chegar a cada um dos vértices 6 e 7 de grau 1, ou seja, necessariamente passando pelo vértice 5, e que estes têm grau 1, então eles pertencerão à mesma comunidade a que o vértice 5 pertencer. Sendo assim, podemos *remover* os vértices 6 e 7, simplesmente carregando-os junto com o vértice 5.

```

1  /* License: GPLv3 * Copyright Luiz C. B. Mostaco-Guidolin */
2  class graphstk::ALGraphs_C : public graphstk::Graphs_C
3  {
4      private:
5          int seed_i_;
6          narray< int, 1 > node_index_list;
7          narray< double, 1> EigenValue;
8          narray< double, 2> Conductance_Matrix;
9          narray< double, 2> Dissimilarity_Matrix;
10         narray< double, 2> Probability_Matrix;
11         narray< double, 2> Average_Distance_Matrix;
12         narray< double, 2> StdDev_Average_Distance_Matrix;
13         narray< double, 2> Eigenvectors;
14
15         gsl_rng* rand_num_generator;
16
17         count_ptr< Clusters_C > leaves_list_ptr_;
18         count_ptr< Clusters_C > discarded_leaves_ptr_;
19         count_ptr< Clusters_C > cluster_list_ptr_;
20     public:
21         ALGraphs_C();
22         ~ALGraphs_C();
23
24         void CalcAverageDistance();
25         void CalcConductance();
26         void CalcDissimilarityMatrix();
27         void CalcEigenv( narray< double, 2 > &Matrix__ );
28         void CalcProbabilityMatrix();
29         void CalcStddevAverageDistance();
30         void InitClusterList();
31         void OrganizeLeaves();
32         void PrintAverageDistanceMatrix();
33         void PrintConductanceMatrix();
34         void PrintEigenvectors();
35         void PrintEigenvalues();
36         void PrintLaplacianMatrix();
37         void PrintNodeNumbering();
38         void PrintProbabilityMatrix();
39         void PrintRemovedLeaves();
40         void PrintSimilarityMatrix();
41         void PrintSimilarityMatrix( string filename__ );
42         void PrintStddevAverageDistanceMatrix();
43         void RemoveLeaves();
44         void SeekCommunities();
45
46         bool FindLeaves();
47 };

```

Algoritmo 5.2: Declaração da classe ALGraphs_C em C++.

O método `RemoveLeaves()` em conjunto com a classe `Clusters_C`, busca recursivamente os vértices caracterizados como folhas e os combina com o vértice ao qual os vértices folha estão conectados. A classe `Clusters_C` será apresentada em detalhes na Sec. 5.3.2, uma vez que ela é a classe central do método para agrupamento aglomerativo desenvolvido nesta tese. Na Fig. 5.7, o vértice 5 é substituído pelo vértice 5^* , que por sua vez é um *multivértice*, objeto da classe `Clusters_C`, que contém a numeração original do vértice 5, além dos vértices carregados por ele.

Figura 5.7: Grafo exemplo, no qual o subgrafo com estrutura de árvore foi retirado (vértices 5, 6 e 7), originando o *multivértice* 5^* .

Após a primeira iteração do método `RemoveLeaves()`, o vértice 5^* , também se torna uma folha, uma vez que seu novo grau passou de $k = 3$ a $k = 1$. Sendo assim, o vértice 5^* deve ser combinado ao vértice 3, o qual, por sua vez, passará também a ser um *multivértice*, denominado 3^* , contendo referências para o vértice 5^* . Dessa forma, o grafo da Fig. 5.6 reduzir-se-á ao grafo mostrado na Fig. 5.8.

Figura 5.8: Grafo exemplo, no qual a folha (*multivértice* 5^*) foi retirada, originando o novo *multivértice* 3^* .

Deve-se ressaltar que durante o processo de remoção de folhas, a matriz adjacência deve ser atualizada, removendo-se os vértices combinados em *multivértices*, tomando-se o cuidado quanto ao controle da numeração atual e original dos vértices envolvidos.

Finalmente, através dos recursos de orientação a objetos, a implementação do método AL utilizando-se a classe `ALGraphs_C` torna-se bastante simples, pois a complexidade da implementação deste método é deixada para a implementação de cada método da classe `ALGraphs_C`. O algoritmo do método AL é apresentado na listagem 5.3.

```

1  /* License: GPLv3 * Copyright Luiz C. B. Mostaco-Guidolin */
2
3  Graphs_C network;
4  network.ReadAdjacencyMatrix( input_file , num_of_nodes );
5  network.TestAdjacencyMatrixSimetry();
6
7  // Search for leaves
8  if( network.FindLeaves() )
9  {
10     network.RemoveLeaves();
11 }
12
13 network.CalcVerticesDegree();
14 network.GenLaplacianMatrix();
15 network.CalcConductance();
16 network.CalcProbabilityMatrix();
17 network.CalcAverageDistance();
18 network.CalcStddevAverageDistance();
19 network.CalcDissimilarityMatrix();

```

Algoritmo 5.3: Procedimento do metodo AL.

Após o cálculo da matriz dissimilaridade, os resultados podem ser analisados por meio de um algoritmo de classificação hierárquica que revelará a estrutura de comunidades segundo a métrica estabelecida pela matriz de dissimilaridades. Para este fim, foi desenvolvido o método HAC++, o qual será apresentado na Sec. 5.3.2.

5.3.2 HAC++

O método de agrupamento hierárquico aglomerativo desenvolvido neste trabalho é parte do projeto `graphstk` por meio da classe `Clusters_C` a qual implementa a infraestrutura básica de listas duplamente ligadas, árvores binárias além dos métodos relevantes ao agrupamento aglomerativo. A listagem da classe `Clusters_C` é mostrada no Alg. 5.4.

O procedimento para agrupamento aglomerativo é por sua vez, implementado como sendo um método da classe `ALGraphs_C`, denominado `SeekCommunities()`. Este método manipula objetos da classe `Clusters_C`. O código fonte do método `SeekCommunities()` é mostrado em Alg. 5.5.

```

1  /* License: GPLv3 * Copyright Luiz C. B. Mostaco-Guidolin */
2  class graphstk::Clusters_C : public scopira::tool::object
3  {
4      private:
5          static size_t nr_of_clusters_in_list_ui_;
6          static size_t nr_of_vertices_ui_;
7          size_t main_node_ui_;
8
9          double cutoff_height_d_;
10
11         narray< size_t, 1 > member_nodes_v_;
12         static narray< int, 1 > node_numbering_v_;
13         static narray< double, 2 > Dissimilarity_Matrix_;
14
15         count_ptr< Clusters_C > right_node_ptr_;
16         count_ptr< Clusters_C > left_node_ptr_;
17         count_ptr< Clusters_C > next_node_ptr_;
18         count_ptr< Clusters_C > previous_node_ptr_;
19
20     public:
21         Clusters_C();
22         Clusters_C
23         (
24             size_t main_ui_,
25             double cutoff_d_,
26             size_t member_ui_
27         );
28         ~Clusters_C();
29
30         size_t GetClusterListSize();
31         size_t GetMainNode();
32         size_t FindBestMatch();
33
34         void Destroy();
35         void CopyMembers
36         (
37             count_ptr< Clusters_C > A_ptr_,
38             count_ptr< Clusters_C > B_ptr_
39         );
40         void CopyMembersFrom( count_ptr< Clusters_C > A_ptr_ );
41         void FreeDissimilarityMatrix();
42         void UpdateNodeNumbering( size_t node_to_remove_ui_ );
43         void AddMemberNode( size_t new_member_ui_ );
44         void PrintNodeInfo();
45         void PrintList();
46         void PrintTree();
47         void AllocDissimilarityMatrix
48         (
49             narray< double, 2 > Dimilarity_Matrix,
50             size_t dimension_ui_

```

```

51     );
52     void SetMainNode( size_t new_main_node_ui__ );
53     void UpdateDissimilarityMatrix
54     (
55         size_t index_to_remain_ui__,
56         size_t index_to_remove_ui__
57     );
58
59     count_ptr< Clusters_C > GetNextNode();
60     count_ptr< Clusters_C > PushBack
61     (
62         size_t node_ui__,
63         double cutoff_d__,
64         size_t member_ui__
65     );
66     count_ptr< Clusters_C > Pop();
67     count_ptr< Clusters_C > Merge
68     (
69         count_ptr< Clusters_C > A_ptr__,
70         count_ptr< Clusters_C > B_ptr__
71     );
72     count_ptr< Clusters_C > TakePlaceOf
73     (
74         count_ptr< Clusters_C > cluster_being_replaced
75     );
76     count_ptr< Clusters_C > FindByMainNode( size_t Main_Node_ui__
77         );
77     count_ptr< Clusters_C > FindByMemberNode( size_t
78         Member_Node_ui__ );
78 };

```

Algoritmo 5.4: Declaração da classe `Clusters_C` em C++.

```

1  /* License: GPLv3 * Copyright Luiz C. B. Mostaco-Guidolin */
2  void graphstk::SeekCommunities()
3  {
4      size_t best_match_ui;
5      count_ptr< Clusters_C > list_position = NULL;
6      count_ptr< Clusters_C > A = NULL;
7      count_ptr< Clusters_C > B = NULL;
8      count_ptr< Clusters_C > C = NULL;
9
10     InitClusterList();
11
12     cluster_list_ptr->AllocDissimilarityMatrix
13     (
14         Dissimilarity_Matrix,
15         nr_of_nodes_ui_
16     );
17
18     A = cluster_list_ptr_;
19     best_match_ui = A->FindBestMatch();
20     B = ClusterList->FindByMainNode( best_match_ui );
21
22     while( A->GetClusterListSize() > 1 )
23     {
24         best_match_ui = B->FindBestMatch();
25         C = cluster_list->FindByMainNode( best_match );
26
27         assert( C->is_alive_object() );
28         if( A->GetMainNode() == C->GetMainNode() )
29         {
30             C = new Clusters_C();
31             list_position = C->Merge( A, B );
32
33             if( list_position->is_alive_object() )
34             {
35                 cluster_list = list_position;
36             }
37
38             A = C;
39             best_match_ui = A->FindBestMatch();
40             B = cluster_list->FindByMainNode( best_match_ui );
41         }
42         else
43         {
44             A = B;
45             B = C;
46         }
47     }
48 }

```

Algoritmo 5.5: Declaração do método SeekCommunities() em C++.

Após o procedimento de agrupamento hierárquico aglomerativo é possível extrair as informações da árvore hierárquica, tais como nível de dissimilaridade e comunidades em cada nível, possibilitando assim dar continuidade às análises, tais como a determinação do valor da função modularidade para cada nível de dissimilaridade que resulte em uma configuração distinta da rede em estudo.

5.4 Testes de desempenho

Sempre que novos algoritmos são propostos, a pergunta que surge naturalmente diz respeito à sua complexidade. Por este motivo, avaliamos experimentalmente o desempenho dos diversos componentes referentes ao método **AL**, bem como da rotina de agrupamento aglomerativo. O procedimento adotado em tal avaliação seguiu a seguinte metodologia:

1. a rotina completa (método **AL** e agrupamento aglomerativo) foram executadas para 10 redes artificiais com estrutura de comunidades e número de vértices entre $n = 100$ e $n = 1000$];
2. para cada rede, o método foi executado 100 vezes;
3. foi medido o tempo de execução de cada sub-rotina e o tempo execução total;
4. para cada rede, foi calculado o valor médio e o desvio da média considerando todas as 100 execuções.

O programa **AL** foi compilado utilizando o compilador **g++** versão 4.4.3 (4.4.3-4ubuntu5), sendo que para a compilação foram utilizados os parâmetros `-O3-NDEBUG-march=native`. Todos os testes foram executados em uma máquina com processador Intel Xeon E5410 (2.33GHz) com 8 núcleos de processamento (cpu family 6, model 23, stepping 6). Considerando que o método **AL** não foi implementado usando recursos de processamento distribuído, apenas um núcleo de processamento foi utilizado por execução.

Os resultados obtidos são apresentados nas Figs. 5.9a–5.9j, as quais correspondem a cada uma das funções membro avaliadas e respectiva função ajustada afim de se determinar experimentalmente a complexidade do algoritmo.

(a) ReadAdjacencyMatrix

(b) TestAdjacencyMatrix

(c) CalcVerticesDegree

(d) CalcEigenv

(e) CalcConductances

(f) CalcProbabilityMatrix

(Continua na próxima página)

Figura 5.9: Avaliação da complexidade e tempo de processamento das funções membro da classe `ALGraphs_C`, necessárias para a implementação do método AL (Figs. 5.9a–5.9i) e 5.9j método de agrupamento aglomerativo.

Analisando-se os gráficos mostrados nas Figs. 5.9a a 5.9j e na Tab. 5.1, vemos que a complexidade dos métodos variou entre $\mathcal{O}(n^2)$ e $\mathcal{O}(n^3)$. Os métodos `ReadAdjacencyMatrix()`, `TestAdjacencyMatrix()` e `CalcVerticesDegree()` possuem complexidade $\mathcal{O}(n^2)$, como esperado, uma vez que suas implementações requerem dois laços para varrer a matriz adjacência inteira. O método `CalcEigenv()`, para o cálculo de autovalores e autovetores, utiliza a função `gsl_eigen_symmv()` da biblioteca GNU-GSL. Esta função, específica para matrizes simétricas, utiliza o método QR simétrico e envolve um custo computacional da ordem de $\mathcal{O}(n^2)$ [103, 104]. Apesar do método QR ter complexidade $\mathcal{O}(n^2)$, ele destrói a matriz sob a qual está operando. Com o intuito de preservar a matriz Laplaciano, é efetuada uma cópia da mesma, a qual é utilizada para o cálculo dos autovetores e autovalores. Além disso, a função `gsl_eigen_symmv_sort()`, (que faz parte da biblioteca GNU-GSL), é utilizada para a ordenação dos autovalores. Sendo assim, a ordem observada para a função `CalcEigenv()` torna-se $\mathcal{O}(n^3)$.

Código	Método	Complexidade
A	ReadAdjacencyMatrix()	$\mathcal{O}(n^2)$
B	TestAdjacencyMatrix()	$\mathcal{O}(n^2)$
C	CalcVerticesDegree()	$\mathcal{O}(n^2)$
D	CalcEigenv()	$\mathcal{O}(n^3)$
E	CalcConductance()	$\mathcal{O}(n^3)$
F	CalcProbabilityMatrix()	$\mathcal{O}(n^2)$
G	CalcAverageDistance()	$\mathcal{O}(n^3)$
H	CalcStddevAverageDistance()	$\mathcal{O}(n^3)$
I	CalcDissimilarityMatrix()	$\mathcal{O}(n^2)$
H	SeekCommunities()	$\mathcal{O}(n^2)$

Tabela 5.1: Componentes do método AL avaliadas em termos de desempenho.

Na tabela 5.1, estão listados todos os métodos analisados, e suas respectivas complexidades, além de um código alfabético que será utilizado na Fig. 5.10.

O cálculo das condutâncias, o qual é efetuado com base nas Eqs. 5.2 e 5.3, envolve três laços sob todos os vértices. Tais laços são necessários para iterar sob todos os vértices por todos os vértices e por todos os autovalores e autovetores. Por este motivo sua complexidade é, obviamente, $\mathcal{O}(n^3)$.

Similarmente, o cálculo das distâncias e desvios padrão segundo as Eqs. 5.5 e 5.6, também envolvem três laços sobre todos os vértices e por isso também possuem complexidade $\mathcal{O}(n^3)$.

No gráfico da Fig. 5.10 é apresentada a comparação da fração de tempo de processamento utilizado por cada componente do método AL, incluindo o método HAC++. Neste gráfico, cada função membro é referenciada pelo código de letra listado na Tab. 5.1. Observando este gráfico, vemos que os métodos que mais consomem tempo de processamento são os métodos de ordem de complexidade $\mathcal{O}(n^3)$. Para melhorar o desempenho do método AL, considerando que não podemos reduzir a complexidade referente aos cálculos das resistências, distâncias médias e desvios padrão das distâncias médias, podemos, alternativamente, utilizar técnicas de processamento distribuído para agilizar cada etapa de cálculos. No caso do cálculo dos autovetores e autovalores é possível utilizar também o método de Jacobi em paralelo [105, 106, 107]¹.

¹<http://code.google.com/p/parallel-jacobi/>

Figura 5.10: Fração do tempo utilizado no processamento de cada componente do método AL + HAC++, listados na Tab. 5.1.

Na Fig. 5.11 é mostrado o tempo de processamento total envolvido na execução de toda a rotina do método AL e HAC++, cuja complexidade final é $\mathcal{O}(n^3)$.

Figura 5.11: Avaliação da complexidade e tempo de processamento total do método AL, incluindo a rotina de agrupamento aglomerativo.

Por fim, avaliamos o desempenho do método de agrupamento hierárquico aglomerativo fornecido pelo ambiente R (função `hclust`), o qual é mostrado na Fig. 5.12.

Figura 5.12: Avaliação da complexidade e tempo de processamento da função `hclust`, fornecida pelo ambiente R, em comparação com o método `HAC++`, desenvolvido neste trabalho.

Apesar do reportado na literatura [84, 85], a rotina `hclust` apresentou complexidade $\mathcal{O}(n^3)$, ao invés de $\mathcal{O}(n^2)$. Apesar da rotina `hclust` ter uma implementação bastante eficiente, o método `HAC++` é visivelmente superior apesar de haver espaço para aperfeiçoamentos, como a utilização da classe `std::map` para a localização de objetos da lista ($\mathcal{O}(\log n)$), e `kd-trees` para a determinação dos objetos mais próximos ($\mathcal{O}(\log n)$).

5.5 Aplicação em redes sociais

No intuito de avaliar os resultados fornecidos pelo método `AL` e para comparações futuras, utilizamos 6 redes sociais bastante estudadas na literatura de redes complexas. Tais redes estão listadas na Tab. 5.2, na qual são apresentados os nomes pelos quais nos referenciaremos a elas durante o texto e suas características: número de vértices e arestas. Incluímos também, o nome encontrado na literatura e a referência para os trabalhos que originaram tais redes.

Para toda divisão possível fornecida pelo método `AL`, foi efetuado o cálculo da modularidade Q e, considerado que seu valor máximo revela a suposta melhor divisão para a rede. Além disso, os valores obtidos foram comparados com aqueles determinados por outros métodos de identificação de comunidades, propostos por diversos autores, sendo eles o *Advanced modularity-specialized label propagation algorithm* (LPAm+) proposto por Liu e Murata [24], o método proposto por Girvan e Newman (GN), por Agarwal e Kempe [23] (VP), e o *Simulated Annealing* (SA) por Guimerà e Amaral [40], descritos anteriormente na Sec. 3.5.2.

Os resultados obtidos por cada método são apresentados na tabela 5.3.

Rede	# de vértices	# de arestas	Nome de Origem	Referência
Karatê	34	78	<i>Zachary's Karate Club</i>	[80]
Golfinhos	62	159	<i>Bottlenose Dolphins</i>	[90]
PolBooks	105	441	<i>American Political Books</i>	[108]
Football	115	613	<i>College Football</i>	[39]
Jazz	198	2742	<i>Jazz Musicians</i>	[109]
LesMis	77	254	<i>Les Misérables</i>	[110]

Tabela 5.2: Relação das redes avaliadas nesta tese, com o nome pelo qual são referenciadas no texto, número de vértices e número de arestas, bem como nome pelo qual são referenciadas na literatura e os artigos de referência.

Observamos na tabela 5.3 que os valores obtidos pelo método AL para o número de comunidades (valores entre colchetes) diferem daqueles encontrados pelos outros métodos em 4 das 6 redes estudadas. Além disso, os valores máximos de modularidade obtidos por AL são inferiores aos obtidos pelos outros métodos, o que mostra que este método não determina as divisões das redes que maximizem a função modularidade. Observamos ainda que, em diversos casos, os métodos determinam valores de Q distintos para uma mesma rede. Estes fatos evidenciam a dificuldade em se determinar qual é a melhor divisão em comunidades para redes sociais uma vez que sua real divisão, em geral, é um tanto quanto subjetiva, não estando simplesmente restrita à definição de comunidades (enunciada na Def. 1).

Rede	AL	VP	LPAm+	GN	SA
Karatê	0,399[3]	0,420[4]	0,420[4]	0,401[4]	0,420[4]
Golfinhos	0,471[5]	0,529[5]	0,529[5]	0,520[5]	0,528[5]
PolBooks	0,499[4]	0,527[5]	0,527[5]	—×—	0,527[5]
Football	0,571[8]	0,605[10]	0,605[10]	0,601[10]	0,605[10]
Jazz	0,273[4]	0,445[4]	0,445[4]	0,405[4]	0,445[4]
LesMis	0,093[19]	0,560[11]	—×—	0,540[11]	—×—

Tabela 5.3: Máxima modularidade calculada para diversas redes sociais, pelos métodos propostos por Alves [14] (AL), Agarwal e Kempe [23] (VP), Liu e Murata [24] (LPAm+), Girvan e Newman [39] (GN), e por Guimerà e Amaral [40] (SA). Entre colchetes são apresentados os números de comunidades obtidas por cada um dos métodos.

Ao analisarmos os dados apresentados na tabela 5.3, referentes à rede do clube de karatê, constatamos que a suposta melhor divisão, de acordo com o critério da modularidade para o método AL, é obtida para 3 comunidades com $Q = 0,399$. Em contrapartida, os outros métodos determinam que 4 comunidades é a melhor divisão com Q variando entre 0,401 e 0,420. Considerando que a divisão natural para esta rede é em 2 comunidades, podemos concluir que o parâmetro modularidade não é totalmente adequado para avaliar a melhor divisão desta rede, uma vez que este é simplesmente baseado na Def. 1.

Além disso, analisando-se o dendrograma gerado a partir da matriz dissimilaridade fornecida pelo método AL (Fig. 5.1) concluímos que a melhor divisão corresponde às duas comunidades, assim como descrito por Zachary. Dessa forma, a simples maximização de Q não revela a verdadeira divisão da rede; por outro lado, este parâmetro serve como uma escala para comparação de resultados e possíveis divisões baseadas na definição de comunidades adotada.

Nas Fig. 5.13 e 5.14 apresentamos a rede do clube de karatê com as divisões obtidas pelo método AL e VP, respectivamente.

Figura 5.13: Máxima modularidade obtida pelo método AL para rede a rede do clube de karatê, revelando 3 comunidades e $Q = 0,399$.

Figura 5.14: Máxima modularidade obtida pelos método VP, para a rede do clube de karatê, revelando 4 comunidades e $Q = 0,420$.

Utilizamos o dendrograma das possíveis divisões desta rede pelo método AL, o qual é mostrado na figura 5.15 no intuito de aprofundar a comparação entre os dois métodos. Vemos que, para 4 comunidades (dissimilaridade de corte representada pela linha pontilhada), o algoritmo AL revela que a quarta comunidade é composta pelos vértices 33–34. Com uma simples inspeção visual da Fig. 5.13 podemos concluir que esta divisão não seria adequada uma vez que tais vértices possuem grau elevado em comparação com os demais. Sendo os graus de tais vértices $k = 12$ e $k = 17$ respectivamente, notamos que tais valores, são comparáveis apenas aos vértices 1 e 3, cujos graus são $k = 16$ e $k = 10$ respectivamente. Este fato torna tais vértices pontos centrais das comunidades à qual pertencem. Separá-los em uma comunidade com apenas estes dois vértices, claramente não faz sentido e contradiz a ideia de que o número de arestas entre vértices de uma mesma comunidade deve ser maior do que o número de vértices entre comunidades diferentes. Além disso, a diferença na dissimilaridade é muito pequena, mostrando que mais do que 3 divisões não apresentam subgrafos com vértices coesos entre si e que possam realmente representar comunidades.

Figura 5.15: Dendrograma da rede do clube de karatê obtido pelo método AL no qual a linha tracejada apresenta a região de corte para se obter 4 comunidades.

Por outro lado, analisando-se a possível divisão em 2 comunidades, o valor obtido para a modularidade pelo método AL é $Q = 0,3715$. Neste caso, as comunidades são constituídas pelos vértices do lado esquerdo (fundindo as duas comunidades à esquerda na figura 5.13) e a outra, pelos vértices do lado direito. Esta divisão corresponde à divisão natural desta rede. Considerando-se a divisão em duas comunidades, fornecida pelo método VP, esta difere da divisão natural apenas pelo vértice 10, assim como mencionado na Ref. [23], resultando em $Q = 0,3718$, ou seja, o valor de Q ainda é ligeiramente superior ao obtido pelo método AL. Vale ressaltar que, assim como o método VP no

caso de duas comunidades, diversos outros métodos [34, 111, 112] também classificam o vértice 10 como pertencendo à comunidade errada, segundo Zachary. Este fato confere ao método AL uma vantagem importante em relação aos demais.

Com base nestes fatos, podemos concluir que o método AL foi capaz de identificar a divisão natural desta rede e que o parâmetro modularidade, apesar de ser útil para definir uma escala para base de comparação, sua maximização não revela por si só a melhor divisão desta rede. De fato, Fortunato e Barthélemy [38] mostraram que funções objetivo, como a modularidade, possuem uma resolução intrínseca e que por este motivo, podem falhar no momento de revelar comunidades menores que a sua resolução.

Capítulo 6

Otimização extrema generalizada

Um dos princípios mais fundamentais da natureza é a constante busca pelo “estado ótimo”. Em Física, podemos entender este conceito através da observação, por exemplo, de partículas de um gás confinadas em um certo volume. Nesta situação, as partículas de tal gás buscam ocupar todo o espaço disponível a elas, de forma a minimizar a energia livre do sistema e consequentemente maximizar sua entropia [113]. Na Biologia, indivíduos que possuem alguma vantagem de adaptação ao ambiente têm maior chance de sobrevivência do que os menos adaptados. Este processo, conhecido como seleção natural ou sobrevivência dos mais adaptados, dita a evolução das espécies [114]. Em Economia, empresas buscam maximizar vendas e lucros ao mesmo tempo que tentam minimizar custos. Processos de otimização estão presentes tanto em sistemas naturais quanto em sistemas e processos artificiais criados pelo ser humano. Por este motivo, a otimização é uma das ciências mais antigas.

Diversas técnicas de otimização vêm sendo criadas e estudas pela área de otimização em matemática aplicada e análise numérica. O objetivo da otimização é determinar o melhor elemento x^* possível do espaço $\mathbb{X}$ de acordo com um conjunto de critérios $F = \{f_1, f_2, \dots, f_n\}$. Estes critérios são expressos por meio de funções matemáticas ou algoritmos, e são chamados também de funções objetivo. Os elementos do domínio $\mathbb{X}$ de f podem ser de qualquer tipo, como números, listas, plantas de construção, ou até mesmo redes complexas. [115]

Existem diversas estratégias de otimização as quais podem ser classificadas como determinísticas ou probabilísticas. Métodos determinísticos são empregados quando há uma clara relação entre as características das soluções possíveis e a sua utilidade para o dado problema. Neste caso, o espaço de soluções pode ser eficientemente explorado utilizando-se algoritmos de divisão e conquista, os quais recursivamente reduzem o espaço de soluções até que a solução ótima seja determinada. Métodos probabilísticos, por outro lado, são utilizados quando não há uma clara relação entre a solução candidata e sua adequação ao problema ou quando a dimensionalidade do espaço de procura é muito

grande tornando o problema difícil de resolver deterministicamente. Tais métodos se baseiam em alguma heurística, de modo a selecionar elementos do espaço de soluções a serem testados como solução. Os métodos probabilísticos podem não fornecer o ótimo global para o problema em estudo. Neste caso, teremos uma solução *um pouco inferior* que por sua vez, assume o caráter de ótima se considerarmos que determinar a solução verdadeiramente ótima poderia eventualmente levar 10^{100} .

A busca da melhor divisão de uma rede complexa em comunidades, por exemplo, se enquadra na categoria de problemas em que o número de possíveis soluções a serem avaliadas é tão grande que seria inviável testá-las todas. Sendo assim, é natural optarmos por um método de otimização probabilístico para tal busca. Para isso, é necessário utilizar alguma função **objetivo** a ser maximizada ou minimizada. No caso de redes complexas, uma função objetivo amplamente utilizada é aquela proposta por Girvan e Newman [25] e conhecida como modularidade (Q). Esta função capta a essência de comunidades em redes complexas em sua definição mais básica, ou seja, que uma comunidade deve ter um maior número de arestas conectando vértices pertencentes à mesma comunidade do que arestas conectando vértices pertencentes à comunidades distintas. Por este motivo, esta função tem sido o ponto central de diversos métodos de otimização especializados em sua maximização e que vêm sendo desenvolvidos nos últimos anos [116, 24, 23, 117, 118, 30, 31, 32, 3, 119, 29].

No Cap. 5, apresentamos o método **AL** e efetuamos o estudo da sua aplicação em redes sociais. Tal método fornece uma matriz de distâncias, a qual estabelece uma métrica que pode ser utilizada para a determinação de comunidades utilizando algum método de agrupamento aglomerativo. O método **AL** é determinístico, e possui um custo computacional relativamente baixo [14] e que em alguns casos, como para a rede do clube de karatê, fornece soluções condizentes com a realidade de tais redes. Entretanto, este método nem sempre revela a divisão da rede que resulte no valor máximo da função modularidade. Considerando que as soluções propostas pelo método **AL** se aproximam das soluções ótimas (segundo a função **objetivo** modularidade) para as redes consideradas, acreditamos que utilizar tais soluções (e não soluções iniciais aleatórias) como objetos a serem otimizados, possa ser uma abordagem eficiente para a obtenção da configuração ótima segundo a função **objetivo**.

Partindo-se desta ideia, vemos claramente que o método **AL** fornece um conjunto inicial de candidatos a soluções e que o processo de otimização está relacionado com variações conformacionais destas. Portanto, os métodos de otimização a serem empregados devem ser capazes de utilizar o conjunto de soluções iniciais e variar suas conformações. Existem pelo menos três métodos que seguem estes princípios e que podem ser empregados neste problema de otimização. São eles, os algoritmos genéticos (**AG**), - recozimento simulado (**SA**) – *Simulated Annealing*, e a otimização extrema generalizada (τ -**EO**) – *Generalized Extremal Optimization*.

O método de otimização conhecido como recozimento simulado, foi desenvolvido por Kirkpatrick *et al.* na década de 1980 [120] inspirado pelo método desenvolvido por Metropolis *et al.* [121] em 1953 para calcular as propriedades de substâncias que podem ser consideradas como compostas de moléculas individuais interagentes [115]. O SA se baseia no procedimento de metalurgia conhecido como recozimento. Neste procedimento, o metal é aquecido a uma temperatura suficientemente alta tal que os íons que o compõem podem se deslocar e desfazer pequenos defeitos na estrutura cristalina do metal. No procedimento de recozimento, a temperatura inicial não pode ser muito baixa e o resfriamento deve ser suficientemente lento para que o sistema não fique preso em algum meta-estado não cristalino que represente um mínimo de energia. Mais precisamente, cada conjunto de posições de todos os átomos do sistema é ponderado de acordo com o fator da probabilidade de Boltzmann $e^{-E/k_B T}$, no qual E é a energia da dada configuração, T é a temperatura (em Kelvin) e k_B é a constante de Boltzmann. No caso de problemas de otimização, a energia E corresponde à função objetivo a ser otimizada. A cada iteração, uma nova configuração (próxima à anterior) é pseudo-aleatoriamente gerada. A energia desta nova configuração é então calculada e a diferença de energia ΔE entre o estado atual e a nova geometria é determinada. Caso esta variação seja positiva ($\Delta E > 0$), a probabilidade desta nova configuração ser aceita é dada pelo fator de Boltzmann, caso contrário $P(\Delta E) = 1$ e a transição será aceita pois a nova energia é menor que a atual. No caso da variação $\Delta E < 0$, um número pseudo-aleatório é gerado a partir de uma distribuição uniforme e a transição será aceita caso o número escolhido seja menor ou igual ao fator de probabilidade de Boltzmann. Em altas temperaturas, esta probabilidade estará próxima de 1, o que faz com que muitas transições de acréscimo sejam aceitas. A aceitação de tais transições faz com que o algoritmo seja capaz de escapar de mínimos locais. À medida que a temperatura diminui, o número de transições aceitas também diminui. Como consequência, o sistema não deve escapar de regiões locais e possivelmente atingirá o mínimo global ao atingir a temperatura mínima ($T = 0K$).

O SA é fortemente dependente da estratégia de resfriamento empregada. Foi demonstrado entretanto, que o SA converge para o ótimo global se a estratégia de resfriamento escolhida for adequada [122, 123], em especial se forem efetuadas um número de iterações t suficientes ($t \rightarrow \infty$). No mesmo trabalho foi demonstrado que o SA encontrará o ótimo global quando o número de iterações exceder a cardinalidade do espaço do problema, o que pode ser muito lento. Entretanto, isto não significa que o SA será lento em todos os casos.

Algoritmos genéticos por sua vez, se enquadram na categoria de algoritmos evolutivos. Estes algoritmos, propostos por John Holland em 1975 [124], são baseados em populações e sua heurística é inspirada em mecanismos da biologia evolutiva, cujos princípios foram inicialmente formulados por Charles Darwin [114]. Os mecanismos envolvidos em tais al-

goritmos tentam imitar os processos biológicos associados à evolução das espécies como: *crossing-over* (ou recombinação), *mutação*, *seleção natural* e sobrevivência do mais adaptado. Da mesma forma, termos da biologia como *gene*, *genoma*, *genótipo* e *fenótipo*, são utilizados na descrição dos algoritmos evolutivos.

Em problemas de otimização envolvendo métodos evolucionários, denotamos por $\mathbb{G}$ o espaço de procura, ou *genoma*. Um elemento $g \in \mathbb{G}$ corresponde a um *genótipo*, muitas vezes referenciado como *cromossomo*. Genótipos por sua vez, correspondem à sequência de *genes* os quais contém as características elementares do indivíduo, ou seja, seu *fenótipo*. Como cada indivíduo é uma instância do seu *genótipo*, os candidatos a solução, (ou *fenótipos*, onde $x \in \mathbb{X}$ no espaço do problema $\mathbb{X}$), são formadas a partir do *mapeamento genótipo-fenótipo* $x = gpm(g)$, ou *genotype-phenotype mapping*, do inglês.

Nos algoritmos genéticos, os elementos g (*genótipos*) do espaço de procura $\mathbb{G}$ (*genoma*) são tipicamente representados, do ponto de vista computacional, por sequências binárias ou vetores de algum outro tipo de dados. Estas representações são utilizadas nas operações de reprodução, *mutação* e *crossing-over*. Ao efetuar o *mapeamento genótipo-fenótipo*, as possíveis soluções da função objetivo $f \in F$, podem ser avaliadas. Na Fig. 6.1 é mostrado um fluxograma do ciclo de algoritmos evolutivos para ilustrar seu funcionamento.

Figura 6.1: Ciclo básico de algoritmos evolucionários. (Adaptação da ref. [115], pág. 96).

O início do processo de otimização ocorre com a criação de uma população inicial (*genótipos*) na qual os *genes* são escolhidos de forma (pseudo) aleatória. Uma vez criada tal população, o valor da função objetivo é calculado, a partir de cada *genótipo*. Com base no valor da função objetivo para cada candidato à solução, estes são categorizados e posteriormente são selecionados os mais adaptados. Dentre os selecionados como melhores candidatos (mais adaptados), ocorrem os processos de *crossing-over* e *mutação*, os quais dão origem a uma nova geração de candidatos a soluções, a partir da qual se espera que tenha ocorrido uma melhora em relação à geração anterior. Este processo se repete

até que o ótimo desejado tenha sido alcançado, tenha sido criado o número máximo de gerações permitidas, ou algum outro critério de parada tenha sido atingido.

Os processos de mutação e recombinação podem ser efetuados segundo diversas estratégias e de forma probabilística. Para recombinação, dados dois **cromossomos**, escolhe-se um ponto (**gene**) a partir do qual os **genes** subsequentes serão simplesmente trocados entre os **cromossomos** envolvidos. Alternativamente, podem ser escolhidos dois ou mais pontos de quebra, caracterizando assim, uma estratégia de recombinação dual-ponto ou multi-ponto. No caso de mutações, um ou mais **genes** são escolhidos aleatoriamente para terem seus valores alterados. O processo de mutação é extremamente importante para que se preserve a variabilidade da espécie, ou no caso do AG, a variabilidade dos candidatos a soluções. O termo aleatório, frequentemente empregado no texto, deve ser interpretado neste trabalho, como pseudo-aleatório sempre que envolver algum processo de escolha feito por meio de um computador, uma vez que utilizar números realmente aleatórios envolveria utilizar dispositivos especiais, tais como o `/dev/random` (disponível em sistemas GNU/Linux), ou algum outro tipo de hardware específico [125, 126, 127, 128].

6.1 Otimização extrema generalizada

O processo de otimização conhecido como otimização extrema generalizada – *Generalized Extremal Optimization* – proposta por Stefan Boettcher [129], explora a ideia de que os processos de otimização existentes na natureza apresentam um caráter extremo. Este extremismo toma forma considerando que espécies biológicas encontram-se em um constante jogo no qual um processo comparativo global persistentemente elimina os menos adaptados fazendo com que o ótimo surja naturalmente a partir da simples seleção dos indivíduos extremamente menos adaptados [129]. Este método se baseia no modelo de evolução biológica proposto por Kim Sneppen e Per Bak *et al.*, o qual ficou conhecido como Bak-Sneppen [130, 131]. A ideia central deste modelo é que a evolução biológica não é gradual mas sim episódica ou abrupta, na qual longos períodos de estase são interrompidos por surtos de atividades [130]. Em simulações computacionais deste modelo, espécies são colocadas em uma grade sendo que a cada espécie é associado um valor de aptidão entre 0 e 1. A cada iteração, a espécie com o menor valor de aptidão é selecionada para ter seu valor atualizado aleatoriamente, tendo seu novo valor amostrado de uma distribuição uniforme entre 0 e 1. Entretanto, a alteração na aptidão de uma espécie influencia a aptidão de espécies inter-relacionadas. No advento desta alteração, todas as espécies vizinhas na grade também têm seu valor de aptidão atualizado aleatoriamente a partir de uma distribuição uniforme. Após um número suficiente de iterações o sistema atinge um estado de alta correlação conhecido como estado de criticalidade auto-organizado. Neste estado, quase todas as espécies atingiram um nível de aptidão acima de um certo limite. Entretanto, estas espécies possuem um *equilíbrio pontuado*, ou

seja, períodos de estase ocasionalmente interrompidos por surtos de evolução [132, 133]. Sendo assim, uma única espécie pouco adaptada pode provocar alterações nos estados de seus vizinhos, ocasionando um efeito cascata que provoca a atividade coevolucionária, potencialmente fazendo com que qualquer estado evolutivo se torne acessível. Apesar desta coevolução não ter como objetivo exclusivo a otimização, ela serve como um poderoso paradigma e por este motivo foi utilizado para a criação da heurística de otimização denominada otimização extrema [129].

Tendo como inspiração o modelo de Bak-Sneppen, a otimização extrema simplesmente atualiza aleatoriamente os elementos piores adaptados, sem buscar a otimização dessa adaptabilidade. Contrariamente aos SA e AG, a otimização extrema utiliza as grandes flutuações inerentes ao sistema para desviá-lo de estados de equilíbrio [129]. A diferença entre a otimização extrema e o recozimento simulado é que no segundo as configurações são aceitas ou rejeitadas a uma dada temperatura de acordo com algoritmo de Metropolis [121], forçando um equilíbrio dinâmico e requerendo um esquema de resfriamento cuidadosamente elaborado. A otimização extrema, por outro lado, inicia-se com o sistema longe do equilíbrio e não aplica nenhum critério de decisão, o que faz com que toda nova configuração seja aceita indiscriminadamente. Esta busca aparentemente aleatória e pouco eficiente atinge rapidamente as soluções quase-ótimas pela persistente seleção do menos adaptado [134].

A diferença entre a otimização extrema e os algoritmos genéticos, por sua vez, é que o AG utiliza um conjunto de cromossomos, representando estados possíveis, a partir dos quais são selecionados aqueles que serão utilizados para gerar uma nova população mais adaptada. O τ -EO por sua vez, atualiza uma única cópia do sistema por meio da simples *remoção* do menos adaptado.

A estratégia de otimização extrema foi originalmente aplicada ao problema de particionamento de grafos por Stefan Boettcher e Allon Percus [134] e posteriormente adaptada por Jordi Duch e Alex Arenas [31]. A primeira implementação, a qual referenciaremos como implementação de Boettcher, consiste na seguinte heurística:

1. escolhe-se um estado do sistema que corresponda a uma partição arbitrária do grafo em duas partições, com $N/2$ vértices em cada;
2. atribui-se um valor de aptidão λ_i a cada vértice i , sendo

$$\lambda_i = \frac{g_i}{g_i + b_i}, \quad (6.1)$$

na qual g_i corresponde ao número de arestas (*boas*) que conectam vértices de uma mesma partição e b_i é o número de arestas (*ruins*) que conectam vértices de partições distintas;

3. cria-se uma lista dos vértices ordenada segundo o valor de aptidão em ordem crescente, sendo w a posição do vértice nesta lista e sendo $w = 1$ o vértice com menor valor de aptidão;
4. com base no valor de aptidão, seleciona-se um vértice da lista de vértices em cada partição segundo a lei de potências

$$P(w) \propto w^{-\tau}, \quad (6.2)$$

na qual $P(w)$ corresponde à probabilidade de um vértice com aptidão w ser selecionado e τ corresponde ao expoente da lei de potências [135, 136];

5. os vértices selecionados são então trocados de partição;
6. repete-se o procedimento a partir do item 2.

Esta estratégia requer $\mathcal{O}(N)$ iterações e fornece como resultado a divisão da rede em duas partições. Espera-se que tal divisão corresponda à melhor divisão em duas comunidades para esta rede.

A implementação proposta por Duch e Arenas [31] difere da implementação de Boettcher em diversos aspectos. Em primeiro lugar, a função utilizada para determinar a aptidão de cada vértice decorre da função modularidade proposta por Newman (Eq. 3.21), a qual corresponde à contribuição individual de cada vértice para a soma no cálculo da função modularidade. Esta função objetivo local é expressa como sendo

$$\lambda_i = \frac{k_{r(i)}}{k_i} - a_{r(i)} = \frac{k_{r(i)}}{k_i} - \frac{k_i}{2m}, \quad (6.3)$$

na qual $k_{r(i)}$ corresponde ao número de arestas do vértice i que o conectam a vértices da mesma partição r . k_i é ao grau do vértice i e $a_{r(i)}$ corresponde à fração de arestas do vértice i e m é número total de arestas na rede.

A segunda diferença corresponde ao fato de que a cada iteração apenas um vértice é trocado de partição, sendo o que o vértice também é selecionado segundo a lei de potências da Eq. 6.2. A cada troca é avaliada uma função objetivo global, sendo esta a função modularidade (Q). Quando incrementos na função modularidade não forem mais possíveis, esta rede é então quebrada em duas novas redes, uma para cada partição, sendo que os vértices que conectam estas duas redes são simplesmente removidos. O procedimento segue então para as duas novas redes, sendo que o valor da função modularidade é avaliado para a rede inteira, considerando as novas comunidades que vêm sendo determinadas. O método segue até que incrementos na função modularidade (objetivo global) não sejam mais possíveis. Esta estratégia, contrariamente do método de

Boettcher, determina a melhor divisão em comunidades para uma dada rede e não a simples bipartição da rede.

Ambas as estratégias possuem abordagens interessantes e, infelizmente, não está claro nos trabalhos originais se uma abordagem é mais adequada do que a outra para a determinação de comunidades de uma rede uma vez que ambos os métodos foram aplicados a redes distintas e possuíam objetivos distintos. No método de Boettcher, o objetivo era determinar a bipartição da rede, ao passo que o método de Duch e Arenas visava obter a melhor divisão em comunidades da rede segundo a função objetivo modularidade.

Nesta tese, visamos explorar a metodologia proposta por Duch e Arenas incorporando aspectos do método proposto por Boettcher. Neste sentido elaboramos um código para otimização baseado no τ -EO capaz de explorar tais abordagens. Na Sec. 6.2, apresentamos a implementação de tal método e na Sec. 6.3 propomos quatro experimentos cujo intuito é explorar as variações de tais métodos, cujos resultados serão apresentados nas Sec. 6.3.1 a Sec. 6.3.4.

6.2 Implementação

Os diversos códigos criados nesta tese para manipulação de redes complexas e detecção de comunidades foram agrupados na biblioteca **graphstk**, apresentada na Sec. 5.3.1. O procedimento de otimização τ -EO, foi implementado como sendo uma classe desta biblioteca e chamada de **TauExtremalOptimizers_C**. Esta classe implementa, a abordagem proposta por Duch e Arenas uma vez que troca dois vértices de suas comunidades e recursivamente divide a rede em duas novas redes quando a divisão ótima em duas comunidades tiver sido alcançada. Nesta implementação é possível substituir as funções objetivo, local e global, bem como alterar a estratégia de trocas de vértices de partição. Dessa forma, podemos explorar aspectos do método proposto por Boettcher segundo a estratégia criada por Duch e Arenas, bem como explorar puramente o método proposto por Duch e Arenas.

O método central da classe **TauExtremalOptimizers_C** corresponde à rotina que executa a otimização propriamente dita, a qual chamamos de **Optimize()**. Esta rotina é mostrada na listagem 6.1. Vemos nesta listagem que o método **Optimize()** é implementado de forma recursiva. Como não há sentido em tentar otimizar redes de entrada com apenas um vértice, este fato é usado como condição de parada deste método.

```

1  /* License: GPLv3
2  * Copyright Luiz C. B. Mostaco-Guidolin */
3  void graphstk::TauExtremalOptimizers_C::Optimize
4  (
5      count_ptr< Graphs_C > G_ptr
6  )
7  {
8
9      size_t commu_A;
10     size_t commu_B;
11     size_t ranked_A;
12     size_t ranked_B;
13
14     Graphs_C &G = (*G_ptr);
15
16     if( (G.GetNumOfNodes() > 1) && (G.GetNumOfCommunities() == 2) )
17     {
18         G.optimal_commu_membership_na_ = G.commu_membership_na_;
19
20         // Optimize:
21         for( size_t i = 0; i < G.GetNumOfNodes(); i++ )
22         {
23             CalcLocalFitness( G_ptr );
24             ranked_A = ChooseNodeToSwap( G_ptr );
25             commu_A = G.fitness_na_[ranked_A]->GetCommuNum();
26
27             do // Choose a node from a different community than node_A
28             {
29                 ranked_B = ChooseNodeToSwap( G_ptr );
30                 commu_B = G.fitness_na_[ranked_B]->GetCommuNum();
31             } while( commu_A == commu_B );
32
33             SwapNodes( G_ptr, ranked_A, ranked_B );
34             UpdateOptimalIfImproved( G_ptr );
35         }
36
37         count_ptr< Graphs_C > MG_copy_ptr = new Graphs_C();
38         (*MG_copy_ptr) = (*main_graph_ptr_);
39
40         UpdateCopyCommuMembership( MG_copy_ptr, G_ptr );
41
42         if
43         (
44             isgreater
45             (
46                 CalcGlobalFitness( MG_copy_ptr ),
47                 CalcGlobalFitness( main_graph_ptr_ )
48             )
49         )
50         {

```

```

51     UpdateMainGraphWith( G_ptr );
52
53     count_ptr < Graphs_C > sub_G_A_ptr = new Graphs_C();
54     count_ptr < Graphs_C > sub_G_B_ptr = new Graphs_C();
55
56     BreakCommunitiesIntoTwoGraphs
57     (
58         G_ptr,
59         sub_G_A_ptr,
60         sub_G_B_ptr
61     );
62
63     Optimize( sub_G_A_ptr );
64     Optimize( sub_G_B_ptr );
65 }
66 }
67 }

```

Algoritmo 6.1: Código fonte em C++ da rotina de otimização cerne do método τ -EO.

Como ponto de partida, é fornecido ao método `Optimize()` um *smart pointer* da classe `Graphs_C`, o qual contém as informações da rede a ser dividida em comunidades. Considerando que a condição de parada não seja satisfeita, a divisão inicial desta rede é considerada como sendo a divisão ótima inicial e a partir deste ponto, é iniciado o laço que efetuará a otimização. O processo de otimização se inicia calculando-se o valor da função **objetivo** local de cada vértice (método `CalcLocalFitness()`). Este método pode ser adaptado para utilizar qualquer função **objetivo** pertinente. Em seguida, é escolhido um vértice de baixa aptidão (*fitness*) para ser trocado de comunidade (método `ChooseNodeToSwap()`). Caso tenha sido adotada a abordagem de otimização proposta por Duch e Arenas, o passo de escolher um segundo vértice deve ser retirado da rotina de otimização. Caso contrário, ou seja, tenha sido escolhido o método proposto por Boettcher, escolhe-se um segundo vértice de baixo **objetivo**, porém de comunidade oposta à do primeiro vértice escolhido. Uma vez escolhido (ou não) o segundo vértice a ser trocado de comunidade, efetua-se a troca (método `SwapNodes()`) e, então é avaliado se a troca resultou em uma melhor divisão para a rede ou não (método `UpdateOptimalIfImproved()`). Caso a troca tenha sido aceita como favorável, a nova divisão ótima é armazenada e o laço continua até que todas as iterações tenham ocorrido ($t = N$).

Ao final das iterações é criada uma cópia (`MG_copy_ptr`) do objeto `Graphs_C` principal que contém a rede antes das alterações que acabaram de ser determinadas como ótima. Esta cópia será utilizada para avaliar se a nova configuração proposta é melhor do que aquela atualmente considerada ótima armazenada no objeto `main_graph_ptr`. Na sequência, a nova proposta de configuração ótima contida no objeto `G_ptr` é aplicada no objeto cópia (`MG_copy_ptr`). Neste ponto, é calculado o valor da função **objetivo** global

(modularidade, nesse caso) para a rede anterior e para a nova rede proposta. Caso o valor da função **objetivo** global na nova configuração seja superior à configuração anterior, esta passa a ser a nova configuração ótima. Como a rede que está sendo otimizada, em cada passo da otimização, pode corresponder a uma fração da rede inicial, esta deve ser atualizada com as informações desta sub-rede otimizada (método `UpdateMainGraphWith()`).

Considerando ainda, que foi possível obter um valor superior para a função **objetivo** global, a rede em foco da otimização é dividida em duas sub-redes independentes e desconexas (método `BreakCommunitiesIntoTwoGraphs()`) as quais são fornecidas como ponto de partida para o método de otimização, fechando assim o ciclo de recursão deste método.

6.3 Descrição experimental

Para testar o algoritmo de Otimização Extrema Generalizada (τ -EO), utilizamos a rede mais famosa da literatura referente ao clube de Karatê. Esta rede (Fig. 6.2) é composta de 34 vértices que representam os membros do clube e 78 arestas representando os laços sociais entre estes membros [80].

Figura 6.2: Rede do clube de karatê com 34 vértices, 78 arestas, dividida em 4 comunidades correspondente à sua máxima modularidade $Q = 0,41979$.

Nos experimentos elaborados e descritos nas seções subsequentes, utilizamos a referida rede e buscamos obter sua melhor divisão em comunidades utilizando o algoritmo τ -EO. Consideramos e definimos neste caso como *melhor*, a divisão tal que a função objetivo

modularidade (Q) atinja o valor máximo possível para a dada rede. Diversos trabalhos têm mostrado que este máximo corresponde a $Q = 0,41979$, o que equivale a quatro comunidades como mostrado na Fig. 6.2.

O procedimento de otimização τ -EO pode ser efetuado utilizando estratégias ligeiramente distintas como as descritas na Sec. 6.1. O primeiro destes procedimentos foi proposto por Boettcher [137, 134] e posteriormente adaptado por Duch e Arenas [31]. A diferença básica entre estas duas abordagens foram apresentadas na Sec. 6.1. Entretanto, nesta tese, iremos nos referir ao método de Boettcher, para indicar o uso da função **objetivo** local apresentada na Eq. 6.1 e indicar que em cada iteração são escolhidos dois vértices para serem trocados de comunidade, uma vez que o procedimento τ -EO implementado nesta tese possui maior grau de similaridade com o método proposto por Duch e Arenas. Entretanto, iremos nos referir ao método proposto por Duch e Arenas, para indicar que será efetuada a troca de comunidade de apenas um vértice e utilizar-se-á como função **objetivo** local aquela apresentada na Eq. 6.3.

Para as investigações efetuadas nesta tese, elaboramos quatro arranjos distintos, sendo eles:

1. utilizar o método de Boettcher, ou seja, utilizar a função **objetivo** local proposta por Boettcher (Eq. 6.1) em conjunto com o esquema de troca de dois vértices, também proposta por ele;
2. utilizar o esquema de troca de vértices proposto por Boettcher em conjunto da função **objetivo** local proposta por Duch e Arenas (Eq. 6.3);
3. utilizar o método proposto por Duch e Arenas, ou seja, utilizar a função **objetivo** local proposta por eles (Eq. 6.3) e trocar apenas um vértice de partição de cada vez;
4. utilizar o esquema de troca de vértices proposto por Duch e Arenas em conjunto da a função **objetivo** local proposta por Boettcher (Eq. 6.1).

Cada situação será referenciada como sendo um experimento respectivamente numerado de 1 a 4.

Boettcher, em seu trabalho original a respeito de τ -EO em redes complexas, não chegou a explorar a divisão das redes em mais de uma comunidade. Seu trabalho ficou restrito ao problema de bipartição das mesmas. Além disso, não utilizou a rede do clube de karatê em seus testes e não utilizou a função modularidade, uma vez que esta ainda não havia sido proposta. Sendo assim, não temos como comparar os resultados obtidos aqui aos dele. Duch e Arenas por sua vez, utilizaram a rede do clube de karatê em seus testes e obtiveram o maior valor da modularidade obtido até então, $Q = 0,4188$ (ano de 2005). Este valor foi posteriormente superado por trabalhos que utilizaram métodos distintos

para determinação de comunidades mas também baseados na otimização da função custo modularidade. Assim sendo, utilizaremos tais valores como base para comparação.

O ponto de partida para os testes aqui propostos na aplicação da otimização τ -EO é sempre gerar uma divisão aleatória da rede em estudo em duas comunidades de tamanhos aproximadamente iguais $n = N/2$. Os experimentos executados nesta seção foram geradas $N_{\text{div}} = 500$ divisões (pseudo) aleatórias e distintas respeitando-se a imposição de duas comunidades. Para a criação de tais redes foi elaborado um simples script de shell escrito em Bash [138], o qual é apresentado na listagem 6.2.

```

1 #!/bin/bash
2 # License: GPLv3
3 # Copyright Luiz C. B. Mostaco-Guidolin
4
5 if [ "$1" == "" ]; then
6     echo "Usage: $0 num_of_vertices"
7     exit
8 fi
9
10 # num. of vertices
11 n=$1
12 let n--
13
14 memberships=1
15
16 > Memberships.dat
17
18 while [ $memberships -le 500 ]; do
19     > membership.dat
20
21     for i in $(seq 0 $n); do
22         echo -ne "$((expr $RANDOM % 2)) " >> membership.dat;
23     done
24
25     grep "$(cat membership.dat)" Memberships.dat
26
27     if [ $? -eq 1 ]; then
28         cat membership.dat >> Memberships.dat
29         echo >> Memberships.dat
30         let memberships++
31         echo $memberships
32     fi
33
34 done

```

Algoritmo 6.2: Script em shell, `gen500Membership.sh`, para geração de $N_{\text{div}} = 500$ divisões pseudo-aleatórias de uma rede complexa em duas comunidades.

Para cada divisão (ou configuração) inicial o programa de otimização foi executado $N_{\text{exec}} = 500$ vezes para cada uma das $N_{\text{div}} = 500$ divisões iniciais, totalizando $N_{\text{rep}} =$

250 000 repetições. Sendo que, os valores máximos obtidos para a função modularidade, o número de comunidades correspondentes a este máximo e a informação de quais vértices pertencem a quais comunidades, foram armazenados. Dado o caráter estocástico desta rotina, a repetição da mesma é essencial, e deve-se considerar como resultado final o máximo obtido dentre todas as realizações.

Deve-se ressaltar também que a rotina de otimização foi executada utilizando $\tau = 1,25$ para o expoente da lei de potências na Eq. 6.2 para seleção dos nós a serem trocados de comunidade de acordo com a função **objetivo** local utilizada. Este valor foi escolhido com base nos trabalhos de Boettcher e também de Duch e Arenas por ser o valor mais adequado para a rotina de otimização proposta [137, 31].

O método τ -EO parte do princípio de que a rede está dividida em duas comunidades de mesmo tamanho. As redes utilizadas neste experimento foram geradas pseudo-aleatoriamente selecionando-se apenas aquelas com configurações distintas. Apesar de não ter sido colocada a imposição de que ambas as partições devessem ter o mesmo número de vértices é natural esperar que as divisões geradas dividam as redes em dois conjuntos com aproximadamente mesmo número de vértices, havendo apenas uma variação dada a aleatoriedade do processo. Para a rede do clube de karatê, composta por $N = 34$ vértices, espera-se que o número de vértices em cada comunidade seja aproximadamente $n = 17$.

Na Fig. 6.3 está apresentada a distribuição dos valores de n para as N_{div} divisões da rede de karatê utilizada nos experimentos.

Figura 6.3: Distribuição do número de vértices n em uma comunidade, para o conjunto das $N_{\text{div}} = 500$ divisões (pseudo-aleatórias) da rede do clube de karatê. Estas divisões são usadas como ponto de partida para determinação da modularidade máxima utilizando o método τ -EO.

6.3.1 Experimento 1

Neste experimento utilizamos a rede do clube de karatê com seus 34 vértices e 78 arestas [80]. Buscamos obter a melhor divisão (máxima modularidade) em comunidades para esta rede utilizando o algoritmo τ -EO, de acordo com as especificações do experimento denominado como experimento 1 descrito na Sec. 6.3.

Para ilustrar o comportamento típico deste método escolhemos uma das redes fornecidas como ponto de partida para o algoritmo. Esta rede, designada rede #111, não possui nenhuma característica especial em relação às outras e foi escolhida a esmo a título de meramente ilustrar o comportamento do método.

Na Fig. 6.4 são mostrados os valores máximos obtidos em cada uma das $N_{\text{exec}} = 500$ execuções da rotina τ -EO para a rede inicial #111 com $n = 16$. Dentre estas execuções encontramos 4 vezes a máxima modularidade conhecida ($Q_{\text{max}} = 0,420$). O valor máximo obtido para a função modularidade apresentado em diversos trabalhos [23, 24, 40] é sempre mostrado com apenas três casas decimais. A partir do trabalho de Bagrow e Bollt [139], o qual apresenta a divisão da rede correspondente a este valor, constatamos que este é fruto do arredondamento de $Q = 0,4197869$. Considerando que não faz sentido calcular incertezas relacionadas à modularidade, a não ser aquelas advindas de cálculos computacionais envolvendo pontos flutuantes [140], não adotamos um critério rígido quando ao número de algarismos significativos apresentados. Sendo assim, este pode variar durante o texto à conveniência, dependendo do aspecto a ser ressaltado em cada situação.

Figura 6.4: Modularidade máxima obtida a partir da rede inicial número 111, em cada uma das $N_{\text{exec}} = 500$ execuções do método τ -EO (configuração do experimento 1), para a rede do clube de karatê.

Como mencionado acima, a Fig. 6.4, mostra que a modularidade máxima, representada pela linha superior (pontilhada), foi obtida apenas em 4 das $N_{\text{exec}} = 500$ execuções da rotina de otimização, sendo que o valor da modularidade da divisão inicial fornecida era de aproximadamente $Q = -0,09698$, mostrada na linha inferior (tracejada). Com isso, podemos concluir que, embora o valor máximo da modularidade tenha sido obtido, o método não converge para a melhor solução em todas as execuções. De fato, para a rede #111, o máximo (Q_{max}) foi obtido em apenas 0,8% das execuções. Vale ressaltar que Duch e Arenas [31] efetuaram testes de seu algoritmo usando a rede do clube de karatê e obtiveram $Q = 0,4188$ como modularidade máxima, ao passo que nossa implementação foi capaz de determinar o máximo reportado na literatura até a presente data. Além disso, na época da publicação por Duch e Arenas (ano de 2005), o valor da modularidade obtido por eles passou a ser adotado como sendo o maior até então conhecido. Este valor foi superado apenas em 2009 por Noack e Rotta [116] que obtiveram o valor de $Q = 0,420$, o qual passou a ser adotado como máximo possível desde então.

Tendo em vista os resultados acima, podemos levantar algumas questões referentes a este método,

1. qual é a fração de realizações que fornecem o valor máximo da modularidade?
2. qual é o intervalo de valores máximos obtidos?
3. quantas vezes estes valores máximos são obtidos para uma mesma rede de entrada?
4. quantas realizações são necessárias para se obter o valor máximo?
5. o valor máximo é obtido para todas as configurações fornecidas como ponto de partida?

Para responder a tais questões analisemos os resultados advindos do conjunto de $N_{\text{rep}} = 250\,000$ realizações compreendendo as $N_{\text{div}} = 500$ configurações iniciais. Na Fig. 6.5, é apresentada a distribuição dos valores máximos obtidos no universo de realizações efetuadas.

Analisando a Fig. 6.5 vemos que todos os valores máximos obtidos no universo das $N_{\text{rep}} = 250\,000$ realizações efetuadas estão compreendidos no intervalo $Q \in [0, 2259 - 0, 4198]$. Podemos observar ainda, que o valor máximo obtido por Duch e Arena ($Q = 0,4188$) ocorreu em aproximadamente 0,21% dos casos, ao passo que o valor máximo ($Q_{\text{max}} = 0,4198$) ocorreu em aproximadamente 0,18% dos casos.

Considerando apenas o valor máximo obtido em cada série de $N_{\text{exec}} = 500$ execuções da rotina de otimização para cada uma das $N_{\text{div}} = 500$ divisões da rede fornecidas como solução inicial, obtivemos a distribuição apresentada na Fig. 6.6.

Figura 6.5: Distribuição dos valores da modularidade máxima obtidos em cada uma das $N_{\text{rep}} = 250\,000$ repetições do método τ -EO (configuração do experimento 1) para a rede do clube de karatê.

Figura 6.6: Frequência de ocorrência dos os valores máximos obtidos para a função modularidade, considerando apenas o valor máximo obtido para cada série de $N_{\text{exec}} = 500$ execuções da rotina de otimização τ -EO (configuração do experimento 1), para cada uma das $N_{\text{div}} = 500$ divisões iniciais da rede do clube de karatê.

Vemos neste gráfico que, ao contrário dos 1690 valores máximos possíveis (apresentados na Fig. 6.5) retornados nas $N_{\text{rep}} = 250\,000$ realizações, apenas 30 valores máximos foram obtidos. Destes, observamos que o segundo maior máximo, ou seja, o máximo determinado por Duch e Arenas $Q = 0,4188$, é obtido como resultado em aproximadamente 26,3% das $N_{\text{div}} = 500$ redes iniciais e que a máxima modularidade possível ($Q_{\text{max}} = 0,4198$) é obtida em aproximadamente 30,5% desta redes.

Associado ao valor máximo encontrado para a modularidade está o número de comunidades correspondentes. Na Fig. 6.7, é apresentada a distribuição do número de comunidades identificadas em todas as $N_{\text{rep}} = 250\,000$ repetições da rotina de otimização.

Figura 6.7: Distribuição do número de comunidades associadas aos valores máximos da modularidade obtidas pelo método τ -EO (configuração do experimento 1) para a rede do clube de karatê.

Neste gráfico observamos que em aproximadamente 54,7% dos casos foram determinadas 3 comunidades correspondentes ao máximo das realizações. Similarmente, em aproximadamente 36,7% dos casos, o número de comunidades determinadas foram 4. Neste ponto, vale lembrar que o valor máximo da modularidade ($Q_{\text{max}} = 0,420$) corresponde a 4 comunidades, sendo que a divisão relativa a este valor é aquela apresentada na Fig. 6.2. Entretanto, a divisão em 4 comunidades pode ser obtida para valores da modularidade diferentes de Q_{max} , ou seja, nem todas as divisões em 4 comunidades correspondem à divisão mostrada na Fig. 6.2.

Outro aspecto relevante desta análise diz respeito ao número de vezes que o valor máximo da modularidade foi obtido para uma determinada configuração inicial. Na Fig. 6.8 é apresentada a distribuição do número de vezes que tal valor máximo foi obtido para cada uma das $N_{\text{div}} = 500$ configurações iniciais da rede.

Figura 6.8: Distribuição do número de ocorrências do valor máximo em cada série de $N_{\text{exec}} = 500$ execução do método τ -EO (configuração do execução 1) para cada uma das $N_{\text{div}} = 500$ divisões iniciais da rede do clube de karatê.

Observamos neste gráfico que em $N_{\text{exec}} = 500$ execuções consecutivas para uma dada configuração inicial, em aproximadamente 33,5% dos casos, o valor máximo é obtido apenas uma vez e que a probabilidade desse máximo se repetir cai exponencialmente. Considerando esse comportamento, resta determinar quantas repetições do processo de otimização para uma mesma configuração inicial são necessárias para se obter o valor máximo. Na Fig. 6.9 está apresentado o histograma no qual é apresentada a distribuição no número da repetição (em cada série), em que o máximo foi alcançado. Neste histograma podemos observar que em, por exemplo, aproximadamente 12,1% dos casos, o máximo valor da modularidade foi obtido no intervalo de tentativas compreendido em $[50,75[$. Isto significa que foi necessário executar a rotina de otimização τ -EO mais de 50 vezes para se obter o maior valor da modularidade possível para a dada divisão inicial. Apesar desta ter sido o intervalo de maior ocorrência do máximo, com base na distribuição apresentada na Fig. 6.9, podemos concluir que não há relação apreciável entre a obtenção do máximo e o número da realização. Sendo assim, para atingirmos o valor máximo é possível que seja necessário efetuar pelo menos $N_{\text{exec}} = 500$ execuções para uma dada configuração inicial. Além disso, é razoável considerar que o método τ -EO implementado nesta tese e baseado no método de Boettcher, seja capaz de determinar Q_{max} para todas as divisões iniciais da rede se o executarmos um número suficiente de vezes. Entretanto, com base nos resultados apresentados nesta seção, vemos que este Q_{max} pode ser obtido variando-se a divisão inicial da rede. Sendo assim, a generalização proposta por Boettcher, ou seja, selecionar os vértices a serem trocados de comunidades

utilizando uma distribuição de lei de potências, não elimina a dependência na divisão inicial utilizada. Entretanto a variação do método τ -EO proposta nesta tese, superou as expectativas uma vez que foi capaz de determinar $Q_{\max}$ para a rede do clube do karatê o que não existe reportado na literatura.

Figura 6.9: Histograma correspondente ao número da execução na qual o máximo valor da modularidade foi obtido, sendo que este não necessariamente corresponde a $Q_{\max}$, mas sim, ao máximo obtido para a dada divisão inicial da rede do clube de karatê.

6.3.2 Experimento 2

Analogamente ao experimento 1, utilizamos a rede do clube de karatê e a abordagem de trocas de vértices proposta por Boettcher. A diferença no procedimento desta seção em comparação com o da seção 6.3.1 é que nesta utilizamos como função de **objetivo** local aquela proposta por Duch e Arenas, apresentada nesta tese como Eq. 6.3. Além disso, para os experimentos executados nesta seção, foram utilizadas novamente as mesmas $N_{\text{div}} = 500$ divisões aleatórias e distintas com duas comunidades, cuja distribuição de vértices em uma comunidade foi apresentada na Fig. 6.3. Novamente, para cada divisão inicial, o programa de otimização foi executado $N_{\text{exec}} = 500$ vezes, utilizando $\tau = 1,25$, para o expoente da lei de potências utilizada (Eq. 6.2) para seleção dos nós a serem trocados de comunidade de acordo com a função de **objetivo** local (Eq. 6.3).

Novamente utilizando a rede inicial #111 para ilustrar o comportamento deste arranjo, mostramos na Fig. 6.10 os valores máximos obtidos em cada uma das $N_{\text{exec}} = 500$ execuções da rotina τ -EO, para a rede inicial ($n = 16$).

Figura 6.10: Modularidade máxima obtida a partir da rede inicial número 111, em cada uma das $N_{\text{exec}} = 500$ execuções do método τ -EO, com troca de vértices pelo método de Boettcher e função objetivo local de Duch e Arenas, para a rede do clube de karatê.

Vemos na Fig. 6.10, que a modularidade máxima, representada pela linha pontilhada (superior), não foi obtida ao longo das $N_{\text{exec}} = 500$ execuções da rotina de otimização, sendo que o valor da modularidade da divisão inicial fornecida era de aproximadamente $Q = -0,09698$ (representada pela linha tracejada inferior). O valor máximo atingido para esta rede, utilizando este arranjo, foi de $Q = 0,4156$. Além disso, se compararmos os resultados do experimento 1, Fig. 6.4, com o resultado mostrado na Fig. 6.10, vemos que neste último, além da modularidade máxima não ter sido obtida nenhuma vez, os valores máximos obtidos no segundo caso estão compreendidos em um intervalo maior do que no primeiro caso. No experimento 1, os valores obtidos estavam compreendidos no intervalo de $[0,2838 - 0,4198]$ e no experimento 2 ficaram compreendidos no intervalo de $[0,1695 - 0,4156]$.

Analisando-se os resultados obtidos para o conjunto de $N_{\text{rep}} = 250\,000$ realizações compreendendo as $N_{\text{div}} = 500$ configurações iniciais. Na Fig. 6.11, é apresentada a distribuição dos valores máximos obtidos no universo de realizações efetuadas. Nesta figura vemos que os valores máximos obtidos no universo das $N_{\text{rep}} = 250\,000$ realizações efetuadas, estão compreendidos no intervalo $Q \in [0,0463 - 0,4198]$. Podemos observar, ainda que o valor máximo obtido por Duch e Arena ($Q = 0,4188$) ocorreu em aproximadamente 0,17% dos casos e, o máximo ($Q = 0,4198$) ocorreu em aproximadamente 0,08% dos casos. Novamente constatamos que o intervalo de valores obtidos no experimento 2, $Q \in [0,0463 - 0,4198]$, é consideravelmente maior do que aquela obtida no experimento

1, $Q \in [0, 2259 - 0, 4198]$.

Considerando apenas o valor máximo obtido em cada série de $N_{\text{exec}} = 500$ execuções da rotina de otimização para as $N_{\text{div}} = 500$ redes de entrada, obtemos a distribuição apresentada na Fig. 6.12. Neste gráfico, estão apresentados apenas os valores máximos obtidos em cada série de $N_{\text{exec}} = 500$ execuções. Ao contrário dos 3 297 valores máximos retornados nas $N_{\text{rep}} = 250\,000$ realizações (Fig. 6.11), se considerarmos os máximos de cada série, teremos apenas 19 valores máximos. Destes, observamos que o segundo maior máximo ($Q = 0,4188$) é obtido como resultado em aproximadamente 29,9% das 500 séries, e a modularidade máxima possível ($Q = 0,4198$) é obtido em aproximadamente 28,1% dos casos. Comparando com os resultados obtidos no experimento 1, no qual o máximo é obtido em 30,5% dos casos, podemos concluir que apesar da diferença de obtenção do máximo não ser muito inferior do que no experimento 1, esta é aproximadamente 2,4% inferior, correspondendo a 6 000 execuções a mais da rotina de otimização. Isto representa um custo computacional superior, e portanto desnecessário a menos que haja algum tipo de ganho em relação ao arranjo utilizado no experimento 1.

Figura 6.11: Distribuição dos valores da modularidade obtidos em cada uma das $N_{\text{rep}} = 250\,000$ execuções do método τ -EO (configuração do experimento 2) para a rede do clube de karatê.

Figura 6.12: Frequência de ocorrência dos os valores máximos obtidos para a função modularidade, considerando apenas o valor máximo obtido para cada série de $N_{\text{exec}} = 500$ execuções da rotina de otimização τ -EO (configuração do experimento 2), para cada uma das $N_{\text{div}} = 500$ divisões iniciais da rede do clube de karatê.

Associado ao valor máximo encontrado para a modularidade, temos o número de comunidades correspondentes. Na Fig. 6.13, é apresentada a distribuição do número de comunidades identificadas em todas as $N_{\text{rep}} = 250\,000$ execuções da rotina de otimização.

Figura 6.13: Distribuição do número de comunidades associadas aos valores máximos da modularidade obtidas pelo método τ -EO (configuração do experimento 2) para a rede do clube de karatê.

Neste gráfico observamos que em aproximadamente 28,2% dos casos foram determinadas 4 comunidades, correspondentes ao máximo das realizações. Neste experimento, entretanto, vemos que o número de comunidades ficou mais bem distribuído em comparação com aquele obtido no experimento 1 (Fig. 6.7). Novamente, o experimento 1 se mostra superior ao arranjo do experimento 2, pois os números de comunidades determinadas no primeiro se concentrou principalmente em 3 e 4 comunidades. Porém no experimento 2 os valores obtidos ficaram mais dispersos do que no experimento 1.

Na Fig. 6.14, é apresentada a distribuição do número de vezes que o valor máximo foi obtido para cada configuração da rede fornecida como ponto de partida para o programa de otimização.

Figura 6.14: Distribuição do número de ocorrências do valor máximo em cada série de $N_{\text{exec}} = 500$ execuções do método τ -EO (configuração de execução 2) para cada uma das $N_{\text{div}} = 500$ divisões iniciais da rede do clube de karatê.

Observamos neste gráfico que em $N_{\text{exec}} = 500$ execuções consecutivas para uma dada configuração inicial, em aproximadamente 66,1% dos casos o valor máximo é obtido apenas uma vez. Outro ponto a ser ressaltado está relacionado com o fato que a probabilidade desse máximo se repetir cai acentuadamente, sendo que duas obtenções do valor máximo ocorre em apenas 18,5% dos casos. Novamente, o arranjo do experimento 1 se mostra superior, considerando-se que é desejável obter o valor máximo no maior número de vezes possível, tal que possamos executar a rotina de otimização o menor número de vezes possível.

Considerando esse comportamento, resta determinar quantas repetições do processo de otimização (para uma mesma configuração inicial) são necessárias para se obter o valor máximo. Na Fig. 7.7 está apresentado o histograma da distribuição do número da repetição (em cada série) na qual o máximo foi determinado.

Figura 6.15: Histograma correspondente ao número da execução na qual o máximo valor da modularidade foi obtido, sendo que este não necessariamente corresponde a $Q_{\max}$, mas sim, ao máximo obtido para a dada divisão inicial da rede do clube de karatê.

Neste histograma podemos observar que, por exemplo, em aproximadamente 8,6% dos casos, o valor máximo da modularidade foi obtido no intervalo de $[25 - 50[$ tentativas. Apesar deste ter sido o intervalo de maior ocorrência do máximo, vemos que diversos outros intervalos possuem valores dessa ordem. Assim como no arranjo do experimento 1, não é possível afirmar qual é o número de repetições mínimas necessárias para a obtenção da maximização da função objetivo.

Com base nos dados apresentados nesta seção e aqueles apresentados no experimento 1, concluímos que o arranjo proposto no experimento 1 é mais adequado do que o proposto no experimento 2.

6.3.3 Experimento 3

No arranjo proposto como experimento 3, utilizaremos a abordagem proposta por Duch e Arenas para a troca dos vértices, bem como a função **objetivo** local também proposta por eles [31]. A diferença desta abordagem consiste em a cada iteração selecionar e trocar de partição apenas um vértice, em oposição à troca de dois vértices de comunidades distintas como proposto por Boettcher.

Além disso, para os experimentos executados nesta seção, foram utilizadas novamente as mesmas $N_{\text{div}} = 500$ divisões aleatórias e distintas com duas comunidades, cuja distribuição de vértices em uma comunidade foi apresentado na Fig. 6.3. Novamente, para cada divisão inicial, o programa de otimização foi executado $N_{\text{exec}} = 500$ vezes. Utiliza-

mos $\tau = 1,25$ para o expoente da lei de potências utilizada (Eq. 6.2) para seleção dos nós a serem trocados de partição de acordo com a função de objetivo local (Eq. 6.3).

Utilizamos novamente a rede inicial 111 para ilustrar o comportamento deste arranjo. Na Fig. 6.16 são mostrados os valores máximos obtidos em cada uma das $N_{\text{exec}} = 500$ execuções da rotina τ -EO (configuração do experimento 3), para a rede inicial (com $n = 16$).

Figura 6.16: Modularidade máxima obtida a partir da rede inicial número 111, em cada uma das $N_{\text{exec}} = 500$ execuções do método τ -EO (configuração do experimento 3), para a rede do clube de karatê.

Vemos na Fig. 6.16, que a máxima modularidade, representada pela linha pontilhada (superior), não foi obtida ao longo das $N_{\text{exec}} = 500$ execuções da rotina de otimização, sendo que o valor da modularidade da divisão inicial fornecida era de aproximadamente $Q = -0,09698$ (linha tracejada inferior). O valor máximo atingido para esta rede utilizando este arranjo, foi de $Q = 0,3358$, o que é consideravelmente inferior àqueles obtidos nos experimentos 1 e 2. Além disso, o intervalo de valores máximos resultantes apresenta uma variação comparável àquela obtida no experimento 2 $[0,1695 - 0,4156]$, ficando compreendido no intervalo de $[0,1103 - 0,3358]$. Ao compararmos os resultados dos três experimentos fica claro, até o momento, que o experimento 3 é o que forneceu resultados mais insatisfatórios para a rede 111.

Analisando os resultados obtidos para o conjunto de $N_{\text{rep}} = 250\,000$ realizações compreendido pelas $N_{\text{div}} = 500$ configurações iniciais, temos a distribuição dos valores máximos obtidos no universo de realizações efetuadas, a qual é mostrada na Fig. 6.17.

Figura 6.17: Distribuição dos valores da modularidade obtidos em cada uma das $N_{\text{rep}} = 250\,000$ execuções do método τ -EO (configuração do experimento 3) para a rede do clube de karatê.

Na Fig. 6.17 é possível notar que os valores máximos obtidos no universo das $N_{\text{rep}} = 250\,000$ realizações efetuadas, estão compreendidos no intervalo $Q \in [0 - 0,3673]$. Vemos que nem o valor máximo obtido por Duch e Arena ($Q = 0,4188$), nem o máximo conhecido ($Q = 0,4198$) foram obtidos em nenhuma das realizações. Este fato, já seria suficiente para descartar tal abordagem, dada sua ineficiência. Entretanto, isto não significa que não é possível obter o máximo ($Q = 0,4188$) obtido pelos autores, mas sim que as configurações iniciais escolhidas não foram adequadas para este fim. Esta forte relação entre as configurações iniciais e os máximos obtidos podem explicar o porquê que os autores não foram capazes de obter a máxima modularidade ($Q = 0,4198$) em seu trabalho original. Este fato é evidenciado uma vez que ambos os métodos apresentados como experimentos 1 e 2 foram capazes de obtê-la, inclusive no experimento 2, que utiliza a função **objetivo** local proposta por Duch e Arenas. Embora esta abordagem já tenha apresentado resultados que justifiquem a sua não utilização, a fim de que sua caracterização seja completa daremos continuidade às análises.

Considerando apenas o valor máximo obtido em cada série de $N_{\text{exec}} = 500$ execuções da rotina de otimização para as $N_{\text{div}} = 500$ redes de entrada, obtemos a distribuição apresentada na Fig. 6.18.

Figura 6.18: Frequência de ocorrência dos valores máximos obtidos para a função modularidade, considerando apenas o valor máximo obtido para cada série de $N_{\text{exec}} = 500$ execuções da rotina de otimização τ -EO (configuração do experimento 3), para cada uma das $N_{\text{div}} = 500$ divisões iniciais da rede do clube de karatê.

Neste gráfico, está apresentado o histograma dos valores máximos obtidos em cada série de $N_{\text{exec}} = 500$ execuções. Ao contrário dos 2669 valores máximos retornados nas $N_{\text{rep}} = 250\,000$ realizações (Fig. 6.17), se considerarmos os máximos de cada série, teremos 166 valores máximos. Este resultado é bem superior aos 30 valores máximos obtidos no experimento 1 ou aos 19 obtidos no experimento 2. É possível ver ainda, que ao contrário dos experimentos 1 e 2, nos quais as ocorrências mais frequentes foram para os dois maiores valores da modularidade obtidos, no experimento 3 as maiores ocorrências concentraram-se na região central do histograma (região compreendida entre $[0,3240 - 0,3406]$).

Associado ao valor máximo encontrado para a modularidade está o número de comunidades correspondentes. Na Fig. 6.19, é apresentada a distribuição do número de comunidades identificadas em todas as $N_{\text{rep}} = 250\,000$ execuções da rotina de otimização. Neste gráfico observamos que em aproximadamente 35,6% dos casos, foram determinadas 3 comunidades correspondentes ao máximo das realizações e 31,1% para 4 comunidades.

Quanto ao número de vezes em que o máximo para uma determinada configuração inicial foi obtido, constatamos que tal máximo foi obtido apenas uma vez em 96,6% das realizações e foi obtido 2 vezes nas 3,4% realizações restantes. Considerando esse comportamento, nos resta determinar quantas repetições do processo de otimização para uma mesma configuração inicial são necessárias para se obter o valor máximo. Na Fig. 6.20 está apresentado o histograma da distribuição do número da repetição (em cada série),

em que o máximo foi alcançado.

Figura 6.19: Distribuição do número de comunidades associadas aos valores máximos da modularidade obtidas pelo método τ -EO (configuração do experimento 3) para a rede do clube de karatê.

Figura 6.20: Histograma correspondente ao número da execução na qual o máximo valor da modularidade foi obtido, sendo que este não necessariamente corresponde a $Q_{\max}$, mas sim, ao máximo obtido para a dada divisão inicial da rede do clube de karatê.

Neste histograma podemos observar que, por exemplo, em aproximadamente 7,9% dos casos, o valor máximo da modularidade foi obtido no intervalo de tentativas compreendida entre 475 e 500. Assim como nos experimentos 1 e 2, não é possível estimar qual é o número mínimo de realizações para que o máximo seja alcançado uma vez que a distribuição apresentada na Fig. 6.20 é aproximadamente uniforme.

6.3.4 Experimento 4

Neste experimento, assim como nos anteriores, utilizamos a rede do clube de karatê e buscamos determinar o valor máximo da função modularidade. Assim como no experimento 3 (Sec. 6.3.3), utilizamos a abordagem proposta por Duch e Arenas para o método de otimização extrema generalizada. Ao contrário do experimento 3, utilizamos como função **objetivo** local aquela proposta por Boettcher (Eq. 6.1).

Seguindo o padrão estabelecido para os experimentos apresentados neste capítulo, utilizamos novamente as mesmas 500 divisões aleatórias e distintas com duas comunidades da rede do clube de karatê, cuja distribuição de vértices em 4 comunidades foi apresentada na Fig. 6.3. Novamente, para cada divisão inicial, tentou-se executar o programa de otimização 500 vezes. Entretanto, apenas uma tentativa para cada rede foi efetuada pois o programa não convergiu para nenhuma rede inicial. Assim como nos demais experimentos, utilizamos $\tau = 1,25$, para o expoente da lei de potências utilizada (Eq. 6.2) para seleção dos nós a serem trocados de comunidade de acordo com a função de **objetivo** local (Eq. 6.1).

Como no caso do experimento 3, os resultados obtidos nesta seção são insatisfatórios. Na realidade, o arranjo proposto como experimento 4, não é capaz de otimizar a função modularidade, pois dada uma configuração inicial com duas comunidades, a troca de apenas um vértice de comunidade aliado à função **objetivo** local proposta por Boettcher faz com que a configuração inicial seja reduzida a apenas uma única comunidade. Considerando a natureza recursiva do algoritmo, e dado que a cada divisão da rede otimizada é gerada uma nova sub-rede com uma nova divisão em duas comunidades, se o método faz com que a rede inicial convirja para uma única comunidade, a nova sub-rede será então uma nova rede inicial. Dessa forma, o método entra em um laço infinito e não gera a otimização desejada.

Tal comportamento, aliado à dificuldade de reproduzir o resultado de Duch e Arenas, pode colocar em dúvida a implementação do método efetuada nesta tese (apesar dos bons resultados obtidos nos experimentos 1 e 2). Entretanto, considerando que os experimentos 1 e 2 forneceram resultados superiores aquelas reportados por Duch e Arenas, consideramos que o método desenvolvido nesta tese está adequado e pode ser utilizado, sugerindo que a melhor abordagem é utilizar o esquema de troca de vértices e função **objetivo** local propostas por Boettcher, como descrito no experimento 1.

6.4 Aplicação em redes sociais

Uma vez determinado o melhor esquema de otimização utilizando o método τ -EO, aplicamos este método nas mesmas redes utilizadas para os testes do método AL (Sec. 5.5, Tab. 5.2). O esquema adotado para o método τ -EO foi aquele utilizado na Sec. 6.3.1, ou seja, a abordagem baseada na proposta por Boettcher tanto para a troca de vértices, quanto para a função objetivo local.

Seguindo o mesmo procedimento experimental, para cada rede social, foram geradas $N_{\text{div}} = 500$ redes iniciais distintas e que dividem as tais redes em duas partições com aproximadamente o mesmo número de vértices em cada partição. Para cada uma das redes de entrada a rotina de otimização foi executada $N_{\text{exec}} = 500$ vezes e o máximo valor encontrado para a modularidade foi considerado como sendo a melhor divisão possível. O parâmetro τ da função objetivo local não foi alterado, permanecendo $\tau = 1, 25$.

Os resultados obtidos são apresentados na Tab. 6.1 em conjunto com aqueles anteriormente apresentados na Tab. 5.3 para comparação.

Rede	AL	τ -EO	VP	LPAm+	GN	SA
Karatê	0,399[3]	0,420[4]	0,420[4]	0,420[4]	0,401[4]	0,420[4]
Golfinhos	0,471[5]	0,529[5]	0,529[5]	0,529[5]	0,520[5]	0,528[5]
PolBooks	0,499[4]	0,525[6]	0,527[5]	0,527[5]	—×—	0,527[5]
Football	0,571[8]	0,600[9]	0,605[10]	0,605[10]	0,601[10]	0,605[10]
Jazz	0,273[4]	0,434[4]	0,445[4]	0,445[4]	0,405[4]	0,445[4]
LesMis	0,093[19]	0,560[6]	0,560[11]	—×—	0,540[11]	—×—

Tabela 6.1: Máxima modularidade calculada para diversas redes sociais, pelos métodos propostos por Alves [14] (AL), Boettcher e Percus [137] (τ -EO), Agarwal e Kempe [23] (VP), Liu e Murata [24] (LPAm+), Girvan e Newman [39] (GN) e por Guimerà e Amaral [40] (SA). Entre colchetes são apresentados os números de comunidades identificadas por cada um dos métodos.

Observamos na Tab. 6.1 que o método τ -EO foi capaz de obter a divisão correspondente a Q_{max} para as redes do clube de karatê e para a rede dos Golfinhos. Além disso, para ambas as redes, os números de comunidades referentes aos Q_{max} para cada rede foi o mesmo daqueles obtidos pelos métodos VP e LPAm+. Sendo assim, o método τ -EO está em completa concordância com os dois métodos citados.

Por outro lado, para a rede LesMis o método τ -EO determinou Q_{max} igual ao determinado pelo método VP porém com um número de comunidades inferior; apenas 6 comunidades, ao contrário das 11 comunidades identificadas pelo método VP. Este fato revela dois pontos que valem ser ressaltados. O primeiro refere-se à não unicidade da função modularidade, ou seja, é possível obter o mesmo valor de Q para configurações totalmente distintas. Em segundo lugar, é possível notar a dificuldade em se definir qual é a melhor divisão da rede LesMis sem recorrer a inspeções visuais, ou até mesmo

informações adicionais. De fato, ser capaz de determinar a real divisão de uma rede é o grande desafio na área de identificação de comunidades em redes complexas, uma vez que a definição de comunidade pode variar ligeiramente de uma rede para outra dado o contexto em que ela se encontra. Para as redes PolBooks, Football e Jazz, os valores máximos obtidos para a função modularidade foram inferiores a $Q_{\max}$. Isto não implica que o método τ -EO não seja capaz de identificá-los, mas sim que ele não foi capaz de identificá-las utilizando as N_{div} redes utilizadas como ponto de partida.

Capítulo 7

ALGO – Método AL otimizado

O método AL, descrito no Cap. 5, define uma métrica a partir da qual é possível estabelecer comunidades em redes complexas. Esta métrica, representada pela matriz dissimilaridade, ao ser processada por um algoritmo de agrupamento aglomerativo hierárquico, como o implementado no método HAC++, fornece uma configuração para cada número de comunidades (de 2 a N comunidades). Analisando o nível de coesão de cada configuração, podemos determinar qual é a melhor divisão para a rede em análise, com base na métrica utilizada. Esta divisão, entretanto, não corresponde necessariamente à melhor divisão segundo a definição de comunidades apresentada na Def. 1. Utilizar a função modularidade como auxiliar ao método AL pode ser interessante do ponto de vista de obtenção de divisões da rede que correspondam à definição de comunidades. Por outro lado, a busca cega pela configuração que forneça $Q_{\max}$ para uma determinada rede não fornece necessariamente sua melhor divisão. Este fato, deve-se à limitação de resolução da função modularidade, como é o caso da divisão em quatro comunidades da rede do clube de karatê com $Q = 0,420$. Ainda assim, a função modularidade tem sido utilizada como função objetivo em diversos métodos de identificação de comunidades, uma vez que, até o momento, é a função que fornece resultados mais próximos da definição de comunidades. Aliado a este fato, o método AL é capaz de fornecer bons resultados tanto no quesito determinação da configuração real de uma rede quanto no quesito divisão segundo a definição de comunidades, como pudemos comprovar através do estudo da rede do clube de karatê. Sendo o método AL um método determinístico, cujo desempenho pode ser sensivelmente aprimorado e capaz de fornecer resultados próximos a $Q_{\max}$, podemos utilizar tais resultados como ponto de partida em métodos de otimização tais como o τ -EO apresentado no Cap. 6 e os algoritmos genéticos.

Poderíamos considerar algoritmos genéticos como método de otimização, uma vez que temos uma população inicial que já está razoavelmente adaptada. Por outro lado, considerando que as conformações iniciais são bastante diferentes entre si, ou seja, cada qual com um número de comunidades distinto, simplesmente efetuar operações de *crossing*-

over, poderia gerar novas soluções bem menos adaptadas do que às que as originaram em uma grande fração dos casos. Sendo assim, a cada geração teríamos muito mais retrocessos do que evoluções. Poderíamos argumentar que seria possível desenvolver um processo de *crossing-over*, tal que este preservasse as características das comunidades interessantes do ponto de vista da evolução para a solução ótima. Entretanto, para isto, deveríamos conhecer as características que levam a uma solução verdadeiramente ótima. Outra possibilidade seria criar, para cada número de comunidades, outras soluções iniciais e recombinar apenas soluções com números de comunidades iguais. Desse ponto de vista, não teríamos vantagem em partir das soluções oriundas do método AL, consequentemente seria mais conveniente partir de soluções iniciais totalmente arbitrárias e usar puramente algoritmos genéticos. A possibilidade que resta com relação a utilização de algoritmos genéticos, seria simplesmente utilizar mutações nos cromossomos iniciais sem utilizar *crossing-over*. Esta alternativa, entretanto, seria pouco eficiente.

O método τ -EO, por sua vez, mostrou-se entretanto capaz de revelar a divisão em comunidades tal que a função modularidade seja máxima. Constatamos que este método é dependente da configuração inicial fornecida como ponto de partida para a otimização. Utilizando a rede do clube de karatê por exemplo, dada a natureza estocástica do método τ -EO, foram necessárias 6 101 execuções. Este resultado correspondeu a tentar $N_{\text{div}} = 12$ configurações iniciais, até que Q_{max} fosse obtido. Além disso, este máximo foi obtido em apenas 0,18% das $N_{\text{rep}} = 250\,000$ repetições, o que torna este método caro do ponto de vista computacional, especialmente para redes com mais de algumas poucas dezenas de vértices. Por fim, o τ -EO utiliza como configuração inicial divisões em apenas duas comunidades e de tamanhos aproximadamente iguais, o que limita sua aplicação. Poderíamos utilizar apenas a configuração com duas comunidades fornecida pelo método AL, mas neste ponto não estaríamos obtendo vantagem real em utilizar o método AL, considerando que as outras $[N(N+1)/2] - 1$ soluções resultantes deste método estariam sendo desprezadas. Ou seja, estaríamos gastando poder computacional com um método, atualmente $\mathcal{O}(n^3)$, para gerar uma única solução inicial, ao passo que poderíamos criar diversas soluções iniciais da mesma forma que fizemos no Cap. 6.

Procurando aproveitar o melhor que cada método descrito acima pode oferecer, desenvolvemos nesta tese um método que utiliza os resultados fornecidos pelo método AL para obter a conformação da rede que forneça Q_{max} , utilizando para isso um método de otimização inspirado no τ -EO e algoritmos genéticos, o qual denominamos método ALGO. Na Fig. 7.1, é mostrada uma representação esquemática deste processo de otimização. Nesta figura vemos que, dada uma rede complexa, esta é processada pelo método AL, o qual fornecerá uma matriz de dissimilaridades como resultado. Esta matriz é processada pelo método HAC++ de agrupamento aglomerativo hierárquico, o qual irá fornecer um cromossomo para cada nível de dissimilaridade que corresponda a conformações distintas da rede. Sendo assim, haverá um cromossomo para a divisão em duas comunidades, um

para a divisão em três comunidades, e assim por diante, até chegarmos ao cromossomo para a divisão em N comunidades.

Figura 7.1: Diagrama esquemático do processo de otimização, proposto nesta tese para determinação da divisão ótima de uma rede em comunidades.

Neste contexto, consideramos um **cromossomo** como sendo um vetor de números inteiros, no qual cada componente do vetor corresponde a um vértice; o valor armazenado em tal posição corresponde ao número da comunidade à qual este vértice pertence. Desta forma, a divisão em $c = 3$ comunidades de uma rede contendo $N = 20$ vértices será representada por um vetor M de tamanho 20. Cada posição M_i do vetor corresponde ao vértice v_i da rede e cujo valor assumido será $M_i = c_i$, sendo $c_i = 0, 1$ ou 2 , dependendo da comunidade à qual o vértice v_i pertença. Criados os cromossomos pelo método HAC++, estes serão processados pelo método ALGO o qual irá determinar qual conformação corresponde ao máximo valor da modularidade, o qual idealmente espera-se que corresponda à melhor divisão em comunidades.

O método ALGO, atua independentemente em cada um dos cromossomos, buscando determinar quais alterações podem ser feitas em tal cromossomo de modo a gerar incrementos na função modularidade. Esta busca é realizada de forma similar ao método τ -EO, ou seja, dada uma configuração inicial (**cromossomo**), é calculada a função objetivo local dada pela Eq. 6.1, a qual é a mesma utilizada por Boettcher. Em seguida, é gerada uma lista de vértices em ordem crescente de adaptação, ou seja, os vértices mais mal classificados estarão encabeçando tal lista. A partir desta lista, sorteia-se dois vértices segundo a lei de potências da Eq. 6.2, com $\tau = 1,25$, assim como utilizado no método de Boettcher, respeitando-se o fato de que os vértices devem pertencer a comunidades distintas. Satisfeita esta condição, é introduzida uma mutação neste cromossomo com probabilidade $P_m = 0,25$, sendo P_m a probabilidade de mutação. Esta mutação

corresponde, simplesmente, ao sorteio de um vértice pseudo-aleatório e trocando-o de comunidade, escolhendo a comunidade de destino também pseudo-aleatoriamente. Esta mutação, inspirada em algoritmos evolutivos, tem como objetivo desviar o processo de otimização de seu curso natural desatrelando, até certo ponto, a resposta final do cromossomo inicial.

Esta mutação foi inclusa observando que assim como o método τ -EO, o método ALGO também possuía uma forte dependência da configuração inicial o que prejudicava a obtenção do máximo valor da função modularidade. Além disso, sem esta mutação o método ALGO apresentava uma convergência para o ótimo bastante lenta à exemplo do método τ -EO. Diferentemente do método τ -EO, no qual uma vez determinada a melhor divisão da rede em duas comunidades, esta era dividida em duas novas redes independentes e o processo de otimização era reiniciado, no método ALGO, uma vez determinada a conformação de máxima modularidade para o dado número de comunidades imposto pelo cromossomo inicial, este simplesmente passava para a otimização do próximo cromossomo. Sendo assim, não era efetuada a quebra do cromossomo otimizado em duas novas redes desacopladas.

Desta forma, o método ALGO, desenvolvido neste tese, consiste na extensão do método τ -EO, tal que seja possível aplicá-lo a soluções iniciais com qualquer número de comunidades, não ficando restrito a apenas duas. Além disso, como vimos anteriormente, o método τ -EO é bastante dependente da solução inicial, podendo não atingir a solução ótima para um número factível de execuções do método. Por este motivo, introduzimos também mutações durante o processo de troca de vértices de comunidades. Isto fez com que o método ALGO convergisse para soluções ótimas a um custo extremamente inferior ao do método τ -EO.

O método ALGO incorpora influências dos algoritmos genéticos pois considera uma população inicial de soluções e utiliza mutações e, ao mesmo tempo, estende o conceito de otimização proposto pelo método τ -EO. Dado que o processo de otimização de cada cromossomo independe dos outros cromossomos, o método ALGO é inerentemente passível de ser paralelizado. Dessa forma, acreditamos que esta *mutação* do método τ -EO, tenha aplicações nos mais diversos problemas de otimização, não ficando restrito apenas a aplicações em redes complexas.

7.1 Implementação

Na listagem Alg. 7.1, é apresentado o código fonte da rotina de otimização do método ALGO. Esta rotina também foi incorporada ao projeto **graphstk** como função membro da classe **ALGraphs_C**.

```

1  /* License: GPLv3 * Copyright Luiz C. B. Mostaco-Guidolin */
2  double graphstk::ALGraphs_C::Optimize
3  (
4      narray< size_t, 1 > &membership,
5      size_t num_of_commus
6  )
7  {
8      size_t ranked_A;
9      size_t ranked_B;
10     size_t commu_A;
11     size_t commu_B;
12     double optimal_modularity;
13     vector< size_t > optimal_membership;
14     vector< size_t > original_membership;
15
16     original_membership = membership;
17     optimal_membership = membership;
18     optimal_modularity = CalcModularity( optimal_membership );
19
20     CalcLocalFitness( membership );
21
22     size_t stop = 1;
23     size_t really_stop = 1;
24     while( isless( fitness_na_[0]->GetFitness(), 0.6 ) )
25     {
26         for( size_t step = 0; step < membership.size(); step++ )
27         {
28             ranked_A = SelectNodeToSwap();
29             commu_A = fitness_na_[ranked_A]->GetCommuNum();
30
31             do // Choose a node from a different community than node_A
32             {
33                 ranked_B = SelectNodeToSwap();
34                 commu_B = fitness_na_[ranked_B]->GetCommuNum();
35             } while( commu_A == commu_B );
36
37             SwapNodes( membership, ranked_A, ranked_B );
38
39             // Mutation
40             if
41             (
42                 gsl_rng_uniform_int( rand_num_generator, 100000 ) < 25000
43             )
44             {
45                 size_t node =
46                     gsl_rng_uniform_int( RNG, membership.size() );
47                 size_t commu = gsl_rng_uniform_int( RNG, num_of_commus );
48                 membership[node] = commu;
49             }
50

```

```

51     {
52         double new_optimal = CalcModularity( membership );
53         if( isgreater( new_optimal, optimal_modularity ) )
54         {
55             optimal_modularity = new_optimal;
56             optimal_membership = membership;
57         }
58     }
59     CalcLocalFitness( membership );
60 }
61
62 if( stop == 250 )
63 {
64     if( really_stop == 2 ) { break; }
65     membership = original_membership;
66     really_stop++;
67     stop = 1;
68 }
69 stop++;
70 }
71
72 membership = optimal_membership;
73
74 return optimal_modularity;
75 }

```

Algoritmo 7.1: Declaração da rotina de otimização do método ALGO em C++.

Vemos nesta listagem que o processo de otimização de uma rede tem como critério de parada a condição de que o adequação local do vértice menos adaptado seja superior a 0,6 (linha 24). Este número foi escolhido considerando-se que se o vértice menos adaptado já está 60% bem adaptado, neste caso dificilmente será possível melhorar o seu posicionamento. Este valor, entretanto, não é de grande importância para o método, uma vez que ele dificilmente será alcançado como veremos a seguir. Uma alternativa a este critério consistiria em simplesmente criar um laço infinito, uma vez que reduzir este valor pode comprometer a qualidade do resultado.

O laço mostrado na linha 26, por sua vez, determina o número de trocas de vértices para uma determinada divisão inicial. Neste ponto, foi escolhido o mesmo número de vértices da rede, o que por sua vez já pode ser demais uma vez que a solução inicial já deve estar próxima do ótimo. Entretanto, este valor foi mantido pela influência do método τ -EO e para garantir que um número suficiente de iterações tenha sido efetuado.

Na linha 40, é introduzida a mutação em algum dos vértices a uma taxa de 25% de ocorrência. Esta mutação é de suma importância considerando-se que o método τ -EO é consideravelmente dependente da divisão inicial. Sendo assim, mutações causam distúrbios que podem resultar em melhoras do resultado final e permitem escapar das limitações inerentes à configuração inicial, que por ventura existam. Por este motivo,

a rede resultante do laço de otimização é utilizada, neste exemplo, por 250 vezes consecutivas (linha 62), o que deixa margem para que mutações ocorram e o espaço de soluções ótimas seja explorado. Ao mesmo tempo que estas mutações podem aprimorar o resultado, é possível que elas simplesmente gerem soluções inferiores. Por este motivo, a rede inicial é fornecida duas vezes ao laço de otimização (linha 64). Assim, o laço principal do método **ALGO**, no exemplo apresentado, é executado 500 vezes para cada divisão inicial da rede, sendo que esta é fornecida duas vezes e processada 250 vezes, com 25% de mutações e N trocas de vértices. A fim de facilitar a comunicação do arranjo experimental, introduzimos a notação $\text{ALGO}(N, 2, 250, 25)$, a qual corresponde a N trocas de vértices, fornecendo a solução inicial duas vezes, iterando 250 vezes a uma taxa de mutação de 25%.

Outro parâmetro importante para este método é o expoente τ da lei de potências utilizada para selecionar os vértices a serem trocados de comunidade. Este parâmetro é utilizado no método `SelectNodeToSwap()`, sendo que o valor adotado foi $\tau = 1,25$. Como este valor foi o mesmo utilizado, tanto nos testes do método τ -EO quanto nos testes do método **ALGO**, não há necessidade de incluí-lo na notação introduzida acima.

7.2 Aplicação em redes sociais

Uma vez estabelecido o método de otimização implementado em **ALGO**, aplicamos este método nas mesmas redes utilizadas para testes dos métodos **AL** e τ -EO (Sec. 5.5, Tab. 5.2). O procedimento experimental adotado foi similar ao procedimento utilizado para os testes do método τ -EO. Entretanto, há uma diferença significativa entre os dois procedimentos. No método τ -EO gerávamos $N_{\text{div}} = 500$ redes iniciais distintas com divisão em duas comunidades e com número de vértices em cada comunidade aproximadamente iguais. No método **ALGO**, as redes iniciais não são geradas aleatoriamente, mas sim obtidas através do processamento da matriz dissimilaridade fornecida pelo método **AL** utilizando o método **HAC++**. Dessa forma, existem $N(N-1)/2$ divisões possíveis, as quais compreendem divisões com uma comunidade (nível de dissimilaridade 100) e indo até divisões com N comunidades (nível de dissimilaridade 0). Dado que estamos partindo de soluções próximas do ótimo, esperamos que o método **ALGO** rapidamente determine este ótimo. Sendo assim, para iniciar a rotina de otimização, fornecemos como entrada a divisão com duas comunidades. Após determinar a melhor configuração para esta divisão, o algoritmo passava à otimização da próxima configuração inicial, ou seja, com 3 comunidades e assim por diante. Para cada divisão inicial, a heurística de otimização foi executada obtendo-se o melhor valor para a função modularidade. Ao final da otimização de cada uma das redes iniciais, o valor da modularidade daquela rede foi computado e comparado com o valor determinado para a rede anterior. Caso a função modularidade tivesse sofrido acréscimo, o procedimento era continuado, caso contrário

o procedimento era finalizado. Desta forma, apenas uma fração $i \ll N(N-1)/2$ de redes iniciais possíveis seria otimizada.

Estabelecida a metodologia, três experimentos foram realizados visando caracterizar o método ALGO,

1. executar o método ALGO para todas as redes sociais exaustivamente, afim de determinar se o método ALGO é capaz de revelar a configuração de máxima modularidade;
2. uma vez estabelecida a máxima modularidade fornecida pelo método ALGO, determinar a configuração de parâmetros do método ALGO capaz de obter os mesmos resultados do item 1;
3. deixar o método ALGO explorar o universo de configurações iniciais afim de determinar se é possível obter valores de modularidade superiores aos obtidos nos itens 1 e 2.

Cada um desses experimentos serão apresentados nas seções seguintes e referenciados como experimentos 1 a 3.

7.2.1 Experimento 1

O experimento apresentado nesta seção corresponde à configuração ALGO($3N, 3, 1000, 25$). Em testes preliminares, estes parâmetros mostraram estar muito além do necessário para se determinar o máximo possível para uma dada rede. Além de utilizar o método ALGO com os parâmetros descritos acima, este foi executado $N_{\text{exec}} = 500$ vezes para cada rede sendo que o valor máximo da modularidade determinado dentre as N_{exec} execuções foi considerado como resultado ótimo para a dada rede. Os resultados ótimos obtidos para cada rede social estudada são apresentados na Tab. 7.1 em conjunto com aqueles anteriormente apresentados na Tab. 6.1 a título de comparação.

Rede	AL	ALGO	τ -EO	VP	LPAm+	GN	SA
Karatê	0,399[3]	0,420[4]	0,420[4]	0,420[4]	0,420[4]	0,401[4]	0,420[4]
Golfinhos	0,471[5]	0,527[5]	0,529[5]	0,529[5]	0,529[5]	0,520[5]	0,528[5]
PolBooks	0,499[4]	0,522[3]	0,525[6]	0,527[5]	0,527[5]	—×—	0,527[5]
Football	0,571[8]	0,595[6]	0,600[9]	0,605[10]	0,605[10]	0,601[10]	0,605[10]
Jazz	0,273[4]	0,444[3]	0,434[4]	0,445[4]	0,445[4]	0,405[4]	0,445[4]
LesMis	0,093[19]	0,548[5]	0,560[6]	0,560[11]	—×—	0,540[11]	—×—

Tabela 7.1: Máxima modularidade calculada para diversas redes sociais, pelos métodos propostos por Alves [14] (AL), Alves e Guidolin (ALGO), Boettcher e Percus [137] (τ -EO), Agarwal e Kempe [23] (VP), Liu e Murata [24] (LPAm+), Girvan e Newman [39] (GN) e por Guimerà e Amaral [40] (SA). Entre colchetes são apresentados os números de comunidades obtidas por cada um dos métodos.

Vemos na Tab. 7.1, que o método **ALGO** foi capaz de identificar a configuração que fornece $Q_{\max}$ para a rede do clube de karatê. Este resultado mostra que o método **ALGO** foi capaz de aprimorar o resultado fornecido pelo método **AL**. Este aprimoramento diz respeito à maximização da função modularidade, uma vez que a divisão verdadeira desta rede (com duas comunidades) foi corretamente identificada pelo método **AL**. Em todas as redes estudadas, o método **ALGO** foi capaz de determinar configurações da rede com valores da modularidade superiores aos valores obtidos para as divisões resultantes do método **AL**, atingindo o objetivo proposto. Entretanto, ao ser comparando com o método τ -EO, desconsiderando-se a rede do clube de karatê já analisada, vemos que o método **ALGO** não foi capaz de igualar, ou até mesmo superar, os resultados obtidos pelo método τ -EO, exceto para a rede de Jazz. Para esta rede o método **ALGO** determinou modularidade $Q = 0,444$, que está mais próximo do valor máximo conhecido $Q_{\max} = 0,445$ e superior ao que o método τ -EO determinou $Q = 0,434$. Nota-se entretanto, que o método **ALGO** determinou este valor para uma divisão da rede com apenas 3 comunidade e não 4 comunidades como determinado pelos demais métodos. Neste caso, vemos que este é provavelmente o máximo possível de se obter com o método **ALGO** para os cromossomos iniciais oriundos do método **AL**. Este fato é corroborado, considerando-se que o método **ALGO** chegou a utilizar a divisão com 4 comunidades, porém esta não forneceu valores superiores àqueles obtidos para 3 divisões, o que ocasionou o encerramento da rotina de otimização.

Fato similar ocorreu para a rede LesMis, cujo valor não se igualou àquele obtido pelo método τ -EO. Neste caso, assim como para as redes do PoolBooks e Football, o método **ALGO** não chegou a ser executado para as divisões em 11, 5 e 10 comunidades (para cada uma das respectivas redes) as quais poderiam potencialmente fornecer o máximo valor da modularidade para estas redes. Para a rede dos Golfinhos, por outro lado, o método **ALGO** simplesmente não chegou a atingir o maior valor da modularidade $Q_{\max}$.

Estes resultados, nos levam a concluir que (i) o método **ALGO**, assim como o método τ -EO, também é dependente da rede fornecida como divisão inicial apesar de que as mutações introduzidas permitem que ele explore outras configurações similares à configuração inicial, o que faz com que a dependência na configuração inicial seja amenizada. (ii) Podemos considerar que os resultados obtidos são bons, dado que o método **ALGO** foi capaz de se aproximar e, em alguns casos igualar ou mesmo superar o método τ -EO baseando-se apenas um número reduzido de configurações iniciais fornecidas pelo método **AL**, ao passo que o método τ -EO fez uso de $N_{\text{div}} = 500$ redes distintas para isso. Sendo assim, é possível concluir que o método **ALGO** é menos suscetível à configuração inicial do que o método τ -EO, apesar desta também limitá-lo. (iii) Por fim, utilizar como critério de parada a diminuição no valor da função modularidade pode não permitir que o método **ALGO** explore divisões da rede com número de comunidades maiores do que $c + 1$, sendo c o número de comunidades que forneceu o valor máximo da modularidade.

Explorar as demais configurações iniciais tem o potencial de revelar $Q_{\max}$, o que poderia indicar que o método ALGO pode ter ficado preso em máximos locais.

A Fig. 7.2 mostra o valor máximo da modularidade obtida em cada uma das $N_{\text{exec}} = 500$ execuções do método ALGO para a rede do clube de karatê. Vemos que o valor máximo foi obtido diversas vezes e que a variação em torno deste máximo foi pequena em comparação com a variação apresentada pelo método τ -EO, como mostrada na Fig. 6.4. Além disso, em todas as $N_{\text{exec}} = 500$ execuções os máximos obtidos correspondiam a divisões da rede em 4 comunidades. A linha inferior tracejada no gráfico da Fig. 7.2 corresponde ao valor da modularidade da rede com 4 comunidades, fornecida pelo método AL e que foi utilizada como rede inicial para otimização.

Figura 7.2: Modularidade máxima obtida em cada uma das 500 execuções do método ALGO para a rede do clube de karatê.

Na Fig. 7.3 vemos que configuração do método ALGO utilizada neste experimento, determinou o máximo valor da modularidade $Q_{\max}$, em 69,4% das execuções. Além disso, este método ficou restrito a apenas 10 valores máximos da modularidade, ao contrário dos 1960 valores máximos retornados pelo método τ -EO (configuração de Boettcher – Experimento 1, Sec. 6.3.1). Ressaltamos ainda que foram necessárias apenas 3 execuções do método ALGO para que $Q_{\max}$ fosse obtido, ao contrário do método τ -EO (configuração de Boettcher) que só retornou $Q_{\max}$ na tentativa número 6 101, o que corresponde à rede inicial #12.

Figura 7.3: Fração dos valores máximos obtidos pelo método ALGO para a função modularidade, calculada para a rede do clube de karatê em uma série de 500 execuções.

Dessa forma, podemos concluir que o método ALGO, através da avaliação da rede do clube de karatê mostrou-se bastante eficiente quanto à sua convergência para o valor máximo da modularidade, uma vez que necessitou de apenas 3 tentativas para obter tal valor. Por outro lado, a configuração utilizada $\text{ALGO}(3N, 3, 1000, 25)$, apesar de fornecer resultados muito bons em poucas tentativas, pode ser consideravelmente lenta uma vez que os parâmetros utilizados visavam efetuar uma busca exaustiva pelo máximo. No experimento 2 deste capítulo, estes mesmos aspectos são avaliados para uma configuração que exija um custo computacional inferior ao requerido neste experimento.

Para avaliação do desempenho da rotina de otimização, o método ALGO completo foi executado 500 vezes para cada rede estudada, sendo que o tempo de execução da rotina de otimização foi determinado. Para isso, em cada uma das 500 execuções a rotina de otimização foi executada 3 000 vezes, sendo que em cada uma foram efetuadas $3N$ trocas de vértices entre comunidades além de 25% de mutações.

Na Fig. 7.4 apresentamos os valores médios dos tempos de processamento, compreendendo as $N_{\text{exec}} = 500$ execuções do método ALGO, para divisões iniciais com 2 a 5 comunidades.

Figura 7.4: Tempo de processamento utilizado para executar apenas a rotina de otimização do método ALGO para a rede do clube de karatê, considerando o número de comunidades envolvidas em cada processo, para a configuração $\text{ALGO}(3N, 3, 1000, 25)$.

Observamos na Fig. 7.4 que o tempo de processamento utilizado para todas as $N_{\text{rep}} = 3\,000$ repetições para a mesma configuração inicial, não depende do número de comunidades de tal configuração. Ou seja, o número de comunidades na configuração inicial não altera a complexidade do método. Sendo assim, o processo total de otimização levou $119,23(7)s$ para todo processo de otimização, para o dado arranjo experimental.

Na Fig. 7.5 é mostrado o gráfico do tempo de processamento referente à rotina de otimização para cada uma das redes estudadas e considerando a configuração $\text{ALGO}(3N, 3, 1000, 25)$.

Figura 7.5: Avaliação do desempenho da rotina de otimização do método ALGO com configuração $\text{ALGO}(3N, 3, 1000, 25)$ para as redes sociais avaliadas.

A curva ajustada mostrada na Fig. 7.5 é da ordem $\mathcal{O}(n^3)$.

Com o arranjo utilizado neste experimento pudemos avaliar o comportamento do método ALGO, bem como avaliar seu desempenho para a configuração utilizada. No experimento 2 visamos verificar a possibilidade de obter os mesmos resultados porém a um custo computacional inferior ao despendido neste experimento. No experimento 3 permitiremos que o método ALGO explore configurações além dos máximos locais determinados, como no caso das redes PolBooks, Football e LesMis.

7.2.2 Experimento 2

O experimento apresentado nesta seção corresponde à configuração $\text{ALGO}(N, 2, 250, 25)$ a qual corresponde a 18 vezes menos iterações do que a configuração utilizada no experimento 1. Da mesma forma que no experimento 1, o método ALGO foi executado 500 vezes para cada rede sendo que o máximo valor da modularidade registrado dentre as 500 execuções foi considerado como resultado ótimo para a dada rede. Os resultados obtidos com esta configuração são apresentados na Tab. 7.2 em conjunto com os resultados obtidos no experimento 1 a título de comparação. Nesta tabela apresentamos, para cada rede, o valor máximo obtido para a função modularidade em conjunto com o correspondente número de comunidades (em colchetes), a multiplicidade de tal resultado, ou seja, o número de vezes que ele foi obtido e por fim o número da repetição no qual este resultado foi obtido.

Rede	Experimento 1			Experimento 2		
	Q	Mult.	Repet.	Q	Mult.	Repet.
Karatê	0,420[4]	347	3	0,420[4]	15	11
Golfinhos	0,527[5]	186	3	0,527[5]	7	6
PolBooks	0,522[3]	303	1	0,522[3]	19	56
Football	0,595[6]	1	494	0,589[5]	2	300
Jazz	0,444[3]	250	2	0,444[3]	5	8
LesMis	0,548[5]	1	172	0,543[5]	1	137

Tabela 7.2: Máxima modularidade determinada para diversas redes sociais, pelo método ALGO com as configurações dos experimentos 1 e 2. Q é o valor da modularidade com o correspondente número de comunidades entre colchetes; Mult. corresponde à multiplicidade, ou seja, número de vezes que o resultado foi obtido e Repet. corresponde ao número da repetição no qual o máximo foi determinado pela primeira vez.

Com base nesta tabela vemos que a configuração do experimento 2 foi suficiente para determinar os mesmos resultados obtidos com a configuração do experimento 1 para 4 das 6 redes testadas, não tendo alcançado o máximo para as redes do Football e LesMis. Estas duas redes, entretanto, já se mostraram bastante desafiadoras mesmo utilizando-se a configuração do experimento 1 uma vez que o máximo obtido com tal configuração foi

obtido apenas uma vez, para ambas as redes. Este fato indica que a configuração inicial fornecida pelo método AL não é adequada para a determinação da configuração que forneça $Q_{\max}$ para estas duas redes. Isto pode ser observado não só pela multiplicidade 1 destes resultados, mas também pelo número da repetição em que tal resultado foi obtido. No caso da rede do Football, o resultado máximo foi obtido apenas na tentativa 494, o que indica que este só foi possível por causa das mutações. Sendo assim, aumentando ligeiramente o número de repetições talvez já seja possível determinar o mesmo valor máximo do experimento 1, considerando que mais mutações poderão ocorrer. Ademais, a configuração utilizada no experimento 2 se mostrou capaz de substituir a configuração do experimento 1 para as outras 4 redes estudadas, principalmente se considerarmos, como exemplo, a rede de Jazz. Os experimentos referentes a esta rede utilizando a configuração 1, levaram aproximadamente 48 dias para serem concluídos, ao passo que o mesmo resultado foi reproduzido utilizando os parâmetros do experimento 2, o qual levou apenas 48 horas.

Na Tab. 7.3 está apresentada a comparação entre os resultados obtidos pelos métodos τ -EO e ALGO utilizando os parâmetros do experimento 2. Nesta, podemos observar que os valores máximos da modularidade obtidos pelo método τ -EO foram superados pelo método ALGO apenas na rede de Jazz, foram igualados na rede do clube de karatê e ficaram inferiores nas demais redes.

Rede	τ -EO			ALGO		
	Q	Mult.	Repet.	Q	Mult.	Repet.
Karatê	0,420[4]	439	6 101	0,420[4]	15	11
Golfinhos	0,529[5]	18	7 905	0,527[5]	7	6
PolBooks	0,525[6]	2	149 878	0,522[3]	19	56
Football	0,600[9]	2	200 669	0,589[5]	2	300
Jazz	0,434[4]	2	15 592	0,444[3]	5	8
LesMis	0,560[6]	2	155 780	0,543[5]	1	137

Tabela 7.3: Máxima modularidade determinada para diversas redes sociais, pelo método τ -EO e pelo método ALGO, sendo Q o valor da modularidade (com correspondente número de comunidades entre colchetes), Mult. corresponde à multiplicidade, ou seja, número de vezes que o resultado foi obtido e, Repet. corresponde ao número da repetição no qual o máximo foi determinado pela primeira vez. O método τ -EO foi executado 250 000 vezes e o método ALGO foi executado apenas 500 vezes.

Considerando que o método τ -EO usou 500 configurações iniciais para cada rede e o método ALGO usou entre 3 e 6 configurações iniciais fornecidas pelo método AL e que cada uma destas configurações correspondia a um número de comunidades distinto, podemos concluir que o método ALGO atinge valores da modularidade comparáveis aos obtidos pelo método τ -EO, mas está limitado pelo número de configurações iniciais que o método AL pode fornecer, uma vez que este se trata de um método determinístico. Avaliando

os resultados das Tab. 7.2 e 7.3, vemos que o método ALGO rapidamente determina a melhor configuração possível dada uma configuração inicial. Por outro lado, comparar os métodos τ -EO e ALGO não é totalmente adequado uma vez que o método τ -EO é, por si mesmo, um método de determinação de comunidades baseado na otimização da função modularidade. O método ALGO, por sua vez é um otimizador das soluções fornecidas pelo método AL cuja heurística é inspirada no método τ -EO.

Comparando o gráfico mostrado na Fig. 7.6, o qual apresenta o valor máximo obtido para a função modularidade em cada uma das 500 execuções do método AL para a rede do clube de karatê, com o gráfico referente ao experimento 1, mostrado na Fig. 7.2, vemos que as reduções dos parâmetros constantes do experimento 2 faz com que haja uma maior variação nos valores máximos obtidos, e no número de obtenções de $Q_{\max}$.

Figura 7.6: Modularidade máxima obtida em cada uma das 500 execuções do método ALGO, para a rede do clube de karatê.

Apesar da configuração do experimento 2 promover maiores variações dos valores máximos em comparação com o experimento 1, ainda assim, tais variações são inferiores às mostradas na Fig. 6.4, referentes ao método τ -EO, experimento 1. Na Fig. 7.7 é apresentada a fração de ocorrências de cada um dos 56 valores máximos obtidos no experimento 2. Neste gráfico observamos que o máximo $Q = 0,3991$ ocorre em aproximadamente 17,5% das $N_{\text{exec}} = 500$ execuções do método ALGO. Isto significa que o resultado fornecido pelo método AL não foi aprimorado nestes casos. Uma possibilidade para este resultado ter sido possível, é considerar que as trocas de vértices de comunidades referentes às mutações podem, na realidade, ter atrapalhado o processo de otimização nestes casos.

Figura 7.7: Fração dos valores máximos obtidos pelo método ALGO para a função modularidade calculada na rede do clube de karatê em uma série de $N_{\text{exec}} = 500$ execuções.

Da mesma forma que no experimento 1, vemos na Fig. 7.8 que não há diferença no custo computacional ao se processar soluções iniciais com números de comunidades distintos.

Figura 7.8: Tempo de processamento utilizado para executar apenas a rotina de otimização do método ALGO para a rede do clube de karatê, considerando o número de comunidades envolvidas em cada processo, com a configuração $\text{ALGO}(3N, 3, 1000, 25)$.

Por fim, na Fig. 7.9 é apresentada a curva referente ao custo computacional do arranjo utilizado no experimento 2. Assim como no experimento 1, a curva ajustada é da ordem $\mathcal{O}(n^3)$. Entretanto o tempo de processamento necessário para cada rede é apreciavelmente inferior no experimento 2 quando comparado com o tempo de execução demandado no experimento 1.

Figura 7.9: Avaliação do desempenho da rotina de otimização do método ALGO com configuração $\text{ALGO}(3N, 3, 1000, 25)$ para as redes sociais utilizadas.

Utilizando a rede de Jazz para comparações, e o arranjo de parâmetros do experimento 1, o tempo necessário para executar todas as 500 execuções do método ALGO é de aproximadamente 48 dias. Contrariamente, utilizando o arranjo de parâmetros do experimento 2, o tempo necessário para o mesmo número de execuções, é de aproximadamente 48 horas. Sendo assim, apesar do arranjo proposto no experimento 2 não ter sido capaz de reproduzir os resultados do experimento 1, para 2 das 6 redes analisadas, o custo computacional associado é apenas aproximadamente 4,2% do custo computacional necessário para o experimento 1.

7.2.3 Experimento 3

Nesta seção exploramos a possibilidade de que o método ALGO pode não ter sido capaz de determinar $Q_{\max}$ para as redes PolBooks, Football e LesMis, pois o método foi interrompido antes de processar redes iniciais com número de comunidades adequados. Ou seja, para a rede de PolBooks, por exemplo, o método ALGO processou apenas até a rede inicial com 4 comunidades, sendo que o máximo que ele chegou a determinar para a função modularidade correspondeu a 3 comunidades. Sendo assim, considerando que

o $Q_{\max}$ conhecido para esta rede, corresponde à divisão com 5 comunidades, é possível que o método ALGO não tivesse obtido $Q_{\max}$ uma vez que ele não chegou a otimizar a solução inicial com 5 comunidades fornecida pelo método AL. De forma similar, as redes do FootBall e LesMis se enquadram nesta situação.

Sendo assim, nos experimentos desta seção, alteramos o critério de parada da rotina de otimização. A nova condição de parada corresponde ao processamento das 13 divisões iniciais. Esta condição consiste em efetuar a otimização das configurações da rede com 2 a 14 comunidades, ao invés da rotina de otimização parar quando a modularidade máxima da rede atual fosse inferior ao valor da modularidade máxima da rede anterior. Este valor foi escolhido por ser superior ao número de comunidades correspondente às divisões que fornecem $Q_{\max}$ para todas as redes em questão, inclusive para a rede LesMis a qual a divisão de $Q_{\max}$ corresponde a 11 comunidades.

Neste experimento, entretanto, não nos restringimos ao processamento das três redes citada anteriormente, mas sim efetuamos os testes para todas as redes, de modo a compreender o real comportamento do método de otimização ALGO face às configurações iniciais fornecidas pelo método AL. Para isso, para cada rede, o método foi executado 500 vezes utilizando como critério de parada em cada execução o número de comunidades da solução inicial. Ou seja, quando o número de comunidades desta fosse maior ou igual a 15 comunidades o método ALGO era encerrado.

Nas Figs. 7.10a a 7.10f, são apresentados os resultados referentes aos testes para todas as redes sociais analisadas. Em todas as redes analisadas vemos que o método ALGO foi capaz de otimizar a solução inicial proposta pelo método AL para praticamente todas as soluções iniciais. Entretanto, para algumas redes esta otimização foi mais acentuada do que para outras. Vemos por exemplo, que para a rede do clube de karatê, as duas soluções iniciais propostas pelo método AL correspondem ao máximo da função modularidade para as divisões em 2 e 3 comunidades. Vemos ainda, que o método ALGO determinou como solução de máxima modularidade a divisão em 4 comunidades, cujo valor obtido corresponde a $Q_{\max}$. Após a divisão de 4 comunidades, o valor da função modularidade, apesar de superior aos valores das soluções iniciais propostas pelo método AL, decrescem. Sendo assim, para a rede do clube de karatê o critério de parada inicial, ou seja, interromper a otimização quando novas divisões da rede não correspondem a incremento na função modularidade, é adequado. Desta forma, não há necessidade real de continuar otimizando soluções com maiores números de comunidades.

Esta característica está presente em praticamente todas as outras redes analisadas à exceção da rede do FootBall (Fig. 7.10d), na qual o valor máximo determinado havia sido para 6 comunidades, tendo o processo de otimização sido interrompido quando a modularidade sofreu decremento ao considerar a divisão com, 7 comunidades. Entretanto, ignorando este fato e prosseguindo com as otimizações vemos que um novo máximo foi determinado. Este corresponde a $Q = 0,590$ em uma divisão com 9 comunidades.

Figura 7.10: Otimização da função modularidade pelo método ALGO para as redes sociais analisadas nesta tese para as divisões iniciais com 2 a 13 comunidades fornecidas pelo método AL.

Sendo assim, para a rede do FootBall o critério de parada original não é adequado. Observamos ainda, que as soluções iniciais para esta rede, propostas pelo método AL e as soluções otimizadas pelo método ALGO não oferecem configurações tão distintas do ponto de vista da função modularidade, uma vez que os valores de Q são bastante próximos para todos, exceto para 3 divisões desta rede, as quais correspondem às divisões em 4, 5 e 6 comunidades. Diferentemente do que ocorre para as demais redes, nas quais é possível observar que houve ganhos em praticamente todos os casos. Para a rede LesMis, por sua vez, a rodada de execuções do método ALGO para obtenção dos dados para o experimento 3, acabaram fornecendo um valor da modularidade superior àquele obtido no experimento 2. O novo valor, o qual também corresponde à divisão em 5 comunidades, é de $Q = 0,545$. Entretanto, este valor não supera aquele obtido com a configuração 1, cujo valor foi $Q = 0,548$, também para 5 comunidades.

Com base nos resultados obtidos no experimento 3, à exceção da rede do FootBall, não há necessidade de se alterar o critério de parada do método ALGO como utilizado nos experimentos 1 e 2. Considerando esta rede ainda, vemos que com as soluções fornecidas pelo método AL, o método ALGO não é capaz de realizar grandes otimizações que gerem divisões da rede correspondentes a valores da modularidade mais elevados.

Capítulo 8

Identificação de domínios em proteínas

A estrutura de biopolímeros, como a das proteínas, é regida em grande parte por interações não covalentes. Interações por meio de forças de Van der Waals e pontes de hidrogênio conferem às proteínas estruturas tridimensionais únicas. O entendimento da topologia das proteínas é de suma importância para estudos relacionados à sua função e enovelamento (*folding*). Para isso, a extração de informações como o agrupamento de tipos específicos de aminoácidos é de grande valia. A teoria de grafos tem sido utilizada na modelagem de proteínas uma vez que esta proporciona mecanismos que permitem representar os constituintes das proteínas, bem como suas interações. Representando-as sob a forma de redes de contatos é possível extrair diversas informações, como por exemplo, a estrutura de comunidades. A estrutura de comunidades emerge considerando que as interações entre os resíduos dos aminoácidos pertencentes a um mesmo domínio é maior do que as interações entre resíduos em domínios distintos.

Comunidades em redes de contato entre resíduos de uma proteína, por sua vez, podem ser entendidas como correspondendo aos domínios estruturais desta proteína utilizando sua estrutura terciária. Por este motivo, aplicamos os métodos de identificação de comunidades em redes complexas desenvolvidos nesta tese para a determinação de domínios estruturais de proteínas. A ideia de usar redes complexas para a determinação de domínios em proteínas, foi originalmente aplicada por Vishveshwara *et al.* [141, 142], utilizando o método espectral para identificação de comunidades em redes de contatos entre os aminoácidos de uma proteína. Posteriormente, Ying Xu *et al.* [75], utilizaram o método de mínimo corte/máximo fluxo (originário de problemas de fluxo em redes), para a identificação de domínios em proteínas. Tal método ficou conhecido pelo nome de Domain Parser [75]. Ambos os métodos apresentaram resultados bastante interessantes, porém estes utilizam métodos de particionamento de redes complexas que datam de 1962 e 1970, respectivamente. Dado o grande avanço na área de redes complexas e

determinação de comunidades nos últimos anos, novos métodos (por sua vez, mais robustos), vêm surgindo com o potencial de revelar domínios estruturais em proteínas. Dessa forma, o segundo objetivo principal desta tese foi buscar a criação de uma metodologia que utilizasse os recentes avanços na área de determinação de comunidades em redes complexas para identificar domínios em proteínas.

O primeiro passo neste sentido corresponde a obter uma representação da proteína sob a forma de uma rede complexa. Esta representação deveria capturar a essência da informação topológica da proteína que fosse relevante à determinação de seus domínios. Para isto, existem pelo menos duas abordagens básicas, sendo elas: (i) considerar contatos entre C_α [143, 144] ou (ii) considerar contatos entre resíduos [145, 146]. Na representação da proteína sob a forma de um grafo, na situação (i) os vértices correspondem aos C_α e na situação (ii) estes correspondem aos resíduos. A adição de arestas entre os vértices é efetuada considerando-se a interação entre os objetos representados pelos vértices, sendo esta avaliada para cada caso. De forma geral, a simplicidade faz com que se defina a existência de uma interação em termos da distância Euclideana entre os vértices.

No caso (i), um grafo é construído utilizando-se as distâncias entre C_α . Formalmente temos o grafo $G(V, E)$, como sendo o conjunto de vértices V , conectados por arestas contidas no conjunto de arestas E . Cada C_α corresponde a um vértice $v_i \in V$, tal que exista uma aresta, $e_{ij} = 1$ com $e_{ij} \in E$, entre os vértices i e j , caso a distância entre tais C_α seja menor que um dado raio de corte R_c , ou seja, $d(v_i, v_j) \leq R_c$ [143, 144]. Dessa forma, variando o raio de corte podemos obter um grafo com mais ou menos arestas. Esta distância tem sido tipicamente utilizada no intervalo de $R_c \in [6, 5 - 8, 5] \text{Å}$ [144, 145, 146, 147, 148, 149].

A outra abordagem considera os vértices do grafo como sendo as cadeias laterais, ou resíduos, da proteína. Neste ponto, há pelo menos três metodologias possíveis. A primeira delas corresponde considerar as distâncias entre os centros de massa de cada resíduo, tal que $e_{ij} = 1$, caso $d(r_{cm}^i, r_{cm}^j) \leq R_c$, sendo r_{cm}^i a posição do centro de massa do resíduo i [146]. A segunda possibilidade é considerar que dois resíduos estão conectados, ou seja, $e_{ij} = 1$, quando pelo menos um átomo do resíduo i esteja próximo a algum átomo do resíduo j segundo um raio de corte R_c . Este último critério corresponde à abordagem adotada por Ying Xu *et al.* em seu método **Domain Parser**, no qual utilizaram $R_c = 4,0 \text{Å}$ [75].

Finalmente, a terceira possibilidade é contar o número de contatos C_{ij} entre todos os átomos dos resíduos i e j ; um contato é considerado caso a distância Euclideana entre dois átomos seja menor que um dado raio de corte estabelecido, R_c . No caso do método espectral aplicado por Vishveshwara *et al.*, este raio de corte correspondia a $R_c = 4,5 \text{Å}$ [141, 142]. Considerando-se o número de contatos entre os resíduos i e j , teremos $e_{ij} = 1$ caso o critério de interação $l_{ij} = (C_{ij}/N_i) \times 100$, esteja dentro de um

limite de corte l_c , sendo N_i o fator de normalização correspondente ao tipo do resíduo i , cujos valores para cada resíduo são mostrados na Tab. 8.1 [141].

Aminoácido	N_i	Aminoácido	N_i
ALA	55,7551	LEU	72,2517
ARG	93,7891	LYS	69,6096
ASN	73,4097	MET	69,2569
ASP	75,1507	PHE	93,3082
CYS	54,9528	PRO	51,3310
GLN	78,1301	SER	61,3946
GLU	78,8288	THR	63,7075
GLY	47,3129	TRP	106,703
HIS	83,7357	TYR	100,719
ILE	67,9452	VAL	62,3673

Tabela 8.1: Fatores de normalização N_i utilizados para determinar a interação entre resíduos.

Novamente, baseado no estudo realizado por Vishveshwara *et al.*, temos que o critério de corte considerado foi $l_c = 4,0\%$.

Dependendo do método e parâmetros utilizados para transcrição da proteína, os grafos resultantes podem diferir potencialmente, facilitando ou dificultando a identificação de comunidades por um determinado método. Nesta tese, o termo transcrição da proteína refere-se ao processo de criar uma rede complexa a partir da posição dos átomos da proteína e não ao processo biológico de criação de uma proteína. Dessa forma é necessário identificar qual metodologia é mais apropriada para transcrever a proteína em redes complexas.

Para ilustrar a diferença entre as redes geradas utilizando-se dois critérios distintos, apresentamos na Fig. 8.1 a rede gerada para a proteína 2SNV-A (PDB ID: 2SNV cadeia A), considerando as distâncias entre C_α e raio de corte $R_c = 8,0\text{\AA}$ (ou $C_\alpha - 8,0\text{\AA}$). Vemos na Fig. 8.2, a rede gerada para a mesma proteína, porém utilizando o critério de interação entre resíduos (CR). Nestes exemplos, ambas as redes apresentam 151 vértices, uma vez que cada C_α corresponde a um resíduo, entretanto na rede mostrada na Fig. 8.1 há 737 arestas, ao passo que na rede da Fig. 8.2 há apenas 511 arestas. Dessa forma, nesta segunda representação um menor número de ligações entre os constituintes da proteína está sendo considerado.

Segundo a base de dados de classificação de proteínas e identificação de domínios CATH, esta proteína possui dois domínios que compreendem os vértices de 0 a 64 e de 65 a 150. Entretanto, com uma simples inspeção visual destas redes não é possível identificar tais limites; pelo menos não com o arranjo espacial dos vértices apresentada.

Figura 8.1: Representação sob a forma de rede complexa da proteína PDB ID 2SNV-A, utilizando distâncias entre C_α e raio de corte $R_c = 8,0\text{\AA}$. Esta rede contém 151 vértices e 737 arestas.

Figura 8.2: Representação sob a forma de rede complexa da proteína PDB ID 2SNV-A, utilizando o critério de interação entre resíduos com $I_c = 4,0\%$. Esta rede contém 151 vértices e 511 arestas.

Na realidade a numeração PDB dos aminoácidos desta proteína, tem como numeração do vértice inicial (aminoácido inicial) como sendo o número 114 e não 0, como nas redes

apresentadas. Sendo assim, de acordo com o CATH, a divisão correta corresponde a 114 a 178 e 179 a 264.

Do ponto de vista de redes complexas, a numeração dos vértices não interfere na análise dos resultados. A notação que será utilizada neste capítulo referente à apresentação das divisões em comunidades é introduzida a seguir.

No exemplo da proteína 2SNV-A, passaremos a apresentar os limites de seus domínios da seguinte forma: (114-178)(179-264). Nesta notação, os números entre parênteses correspondem aos números dos aminoácidos em cada domínio. Como há dois conjuntos de números entre parênteses, significa que nesta cadeia da proteína há dois domínios. Além disso, o hífen é utilizado para representar a sequência de aminoácidos, ou seja, 114-178, corresponde a todos os aminoácidos desde o aminoácido 114 até o aminoácido 178. No caso, em que haja domínios não sequenciais, o caractere ';' será utilizado para representar tal quebra, correspondendo ao designador 'e'. Sendo assim, domínios definidos como (1-57;115-170)(58-114) indicam que o primeiro domínio contém os aminoácidos de 1 a 57 e 115 a 170, enquanto o segundo domínio corresponde aos aminoácidos 58 a 114.

Para a transcrição das proteínas em redes complexas, desenvolvemos nesta tese o aplicativo denominado **cvtpdb**. Este aplicativo manipula arquivos PDB e gera a representação da proteína em redes complexas de acordo com o critério de transcrição escolhido. A descrição e características deste aplicativo serão apresentadas na Sec. 8.1.

8.1 Transcrição de proteínas em redes complexas com o cvtpdb

Como mencionado na seção anterior, a transcrição de uma proteína sob a forma de redes complexas pode ser efetuada de diversas maneiras. A criação de uma aresta entre dois vértices da rede está relacionada com o fato de haver, ou não, uma ligação química entre os constituintes da proteína (considerados como sendo os vértices da rede) forte o suficiente para que esta seja considerada uma ligação relevante do ponto de vista de representar a topologia de tal proteína. Esta distância, ou raio de corte R_c , tem sido utilizada no intervalo compreendido entre $R_c = 6,5$ e $8,5\text{\AA}$ [144, 145, 146, 147, 148, 149].

Para que pudéssemos avaliar possíveis diferenças entre os métodos de criação das redes criamos um aplicativo, o qual chamamos de **cvtpdb** – *Convert PDB*, desenvolvido utilizando a linguagem de programação Python e distribuído sob a licença GPLv3, o qual o caracteriza como software livre e de código aberto. Este aplicativo pode ser obtido pelo seguinte endereço na internet <http://atlas.ffclrp.usp.br/cvtpdb>.

Este aplicativo, de linha de comando, possibilita efetuar o download de um arquivo PDB [150] simplesmente fornecendo o código de identificação (PDB ID) da estrutura de interesse. Além disso, ele oferece a possibilidade de transcrever a proteína sob a

forma de redes complexas utilizando os critérios de: distâncias entre C_α com raio de corte especificados pelo usuário, distâncias entre os centros de massa dos resíduos com raios de corte também especificados pelo usuário, critério de contato de interação entre resíduos utilizando os fatores de normalização fornecidos na Tab. 8.1 e, contato entre resíduos seguindo o critério estabelecido pelo método *Domain Parser*. Às redes de contato é possível adicionar ainda, contatos adicionais referentes às estruturas secundárias, como α -hélice e folha- β .

Além de gerar as redes de contato, o *cvtpdb*, é capaz de calcular o raio de giração da proteína (R_G), o qual fornece informação de quão compacta é a estrutura, bem como gerar um arquivo no formato DOT [151], o qual é utilizado para gerar representações gráficas de redes complexas, como a apresentada na Fig. 8.1.

Como subprodutos, é possível obter arquivos das redes de contato, sob a forma de matriz adjacência, em arquivos com formatação específica para FORTRAN ou com uma formatação mais livre que pode ser usada, como por exemplo, em linguagem C++. Além disso, são gerados um arquivo para cada cadeia da proteína além de arquivos de informações contendo o número de resíduos, resíduos faltantes, raio de giração e resolução. Outros arquivos compreendem arquivos contendo apenas os registros ATOM desconsiderando os hidrogênios e arquivos referentes aos registros HELIX e SHEET. Os referidos registros são descritos no dicionário do formato de arquivos PDB [150].

Sendo assim, o *cvtpdb* é uma ferramenta versátil e que pode ser amplamente utilizada e aprimorada uma vez que é distribuída como software livre e de código aberto. Todas as proteínas apresentadas neste trabalho foram manipuladas por meio do *cvtpdb*.

8.2 Proteínas como redes de contato

As proteínas utilizadas nesta tese foram escolhidas considerando-se a disponibilidade de informação a respeito de seus domínios na base de dados CATH. Os mantenedores desta base de dados disponibilizam em seu sítio na internet um arquivo chamado *CathDomall*¹ o qual contém as informações a respeito dos limites dos domínios para cada cadeia das proteínas disponíveis na base de dados do PDB. Na versão 3.3.0 desta base de dados, estão disponíveis 97 643 cadeias de proteínas, as quais apresentam um número variado de domínios, compreendidos no intervalo de [1 – 9] domínios. Na Fig. 8.3, é apresentada a distribuição do número de estruturas em função do número de domínios que estas apresentam. Vemos que aproximadamente 63% das cadeias mapeadas são compostas de um único domínio estrutural. Nesta tese, entretanto, nosso interesse é nas proteínas que possuem mais de um domínio estrutural, ou seja as 37% restantes. Para os testes efetuados nesta tese, focamos nossos esforços em proteínas com dois domínios o que

¹O arquivo *CathDomall* pode ser obtido através do endereço: <http://release.cathdb.info/v3.3.0/CathDomall>

corresponde a aproximadamente 28,5% das cadeias disponíveis, ou alternativamente, aproximadamente 77% das cadeias que apresentam mais de um domínio.

Figura 8.3: Distribuição do número de cadeias de proteínas disponíveis na base de dados CATH em função do número de domínios.

Além de considerar o número de domínios, utilizamos nesta tese apenas cadeias não fragmentadas. O termo fragmento empregado para indicar partes das sequências de aminoácidos da qual não se possui informação devido à dificuldades experimentais. Esta falta de informação, em última instância, corresponde à ausência do registro **ATOM** (no arquivo PDB da respectiva proteína) para os aminoácidos do fragmento. Do número total de cadeias com dois domínios, aproximadamente 48% são fragmentadas.

Outro aspecto a ser considerado corresponde às cadeias com domínios sequenciais (e não sequenciais). Um domínio sequencial, ou equivalentemente, um segmento, corresponde a um domínio formado por uma sequência contínua de aminoácidos considerando a numeração dos mesmos no *backbone* da cadeia. Como exemplo de domínios sequenciais, utilizamos novamente a proteína 2SNV-A. Seus domínios são (114-178)(179-264) ou seja, estes podem ser representados por apenas uma quebra no *backbone* da cadeia. Em oposição, domínios não sequenciais são compostos por mais de um segmento do *backbone*. Utilizando novamente o exemplo hipotético apresentado anteriormente, nos quais os domínios correspondiam à (1-57;115-170)(58-114), vemos neste caso, que os limites do primeiro domínio correspondem a dois segmentos distintos do *backbone*, logo representa uma cadeia com domínios não sequenciais. Considerando-se cadeias com apenas dois domínios e não fragmentadas, temos que aproximadamente 29,0% destas são sequenciais e aproximadamente 24,0% são não sequenciais (Fig. 8.3).

Nesta tese, efetuamos testes com 56 cadeias selecionadas dentre as 7 986 cadeias não fragmentadas e com domínios sequenciais. A relação dos PDB ID e respectivas cadeias

destas proteínas estão relacionadas na Tab. 8.2.

1A02-N	1A0H-A	1A0P-A	1A16-A	1A20-A	1A3L-H	1A41-A	1A7-A
1A45-A	1A4J-B	1A4K-A	1A5D-A	1A6A-A	1A6C-A	1A6Q-A	1A0F-A
1A6T-C	1A79-A	1A87-A	1A8D-A	1A8P-A	1A8R-A	1A8V-A	1APG-A
1A9C-A	1A9D-A	1ABR-A	1ABW-A	1ABY-A	1ACM-B	1ACY-L	1AQD-A
1AD0-B	1AD9-B	1ADM-A	1ADQ-A	1AE6-H	1AF7-A	1AF9-A	1EZM-A
1AGX-A	1AHA-A	1AHB-A	1AHC-A	1AHJ-B	1AHJ-F	1AHW-B	15C8-H
1AI1-L	1AI6-A	1AIF-A	1AIG-M	1AIJ-M	1AMM-A	1AMY-A	1D5M-B

Tabela 8.2: Relação dos PDB ID das 56 cadeias de proteínas analisadas nesta tese.

A distribuição do número de aminoácidos (N) das cadeias das proteínas utilizadas é apresentada na Fig. 8.4, sendo que os valores específicos para cada cadeia estão apresentados na Tab. A.1, no Ap. A.1. Nesta tabela, observamos que foram utilizadas cadeias de proteínas com $N = 157$ até cadeias com $N = 532$. Entretanto, pela distribuição do número de vértices (Fig. 8.4), observamos que aproximadamente 32,7% das cadeias analisadas são compostas por $N \in [200 - 225[$ aminoácidos.

Figura 8.4: Distribuição do número de aminoácidos em cada uma das 56 cadeias de proteínas analisadas nesta tese.

Para cada uma das proteínas listadas na Tab. 8.2 foram geradas suas respectivas representações em redes complexas utilizando-se para isso o critério de transcrição baseado nas distâncias entre C_α . Consideramos em nosso estudo três raios de corte distintos $R_c = 7,0, 7,5$ e $8,0\text{\AA}$. Além do critério de distâncias entre C_α , geramos as representações utilizando os critérios do Domain Parser e o critério que leva em consideração o número de contatos entre átomos de resíduos. Neste último, o raio de corte adotado foi $R_c = 4,5\text{\AA}$ e critério de corte para interações $I_c = 4,0\%$.

Adotaremos as notações a seguir para nos referirmos às redes geradas usando um desses critérios. Para a distância entre C_α com um determinado raio de corte, como por exemplo $R_c = 7,5\text{\AA}$, usaremos a notação, $C_\alpha : R_c = 7,5\text{\AA}$; para redes geradas pelo critério do Domain Parser, usaremos simplesmente a notação DP, da qual fica subentendido $R_c = 4,0\text{\AA}$ e, para o critério de contato entre átomos de resíduos, utilizaremos a notação CR.

Apresentamos nas seções seguintes as análises envolvendo estas transcrições.

8.2.1 Estudo do *backbone* proteico a partir da matriz adjacência

Considerando-se as proteínas utilizadas neste estudo como redes complexas, estas são representadas por suas respectivas matrizes adjacência. Como cada proteína foi transcrita para sua forma de rede de contatos utilizando-se os critérios mencionados, teremos cinco matrizes adjacência distintas para cada proteína. A partir de cada matriz adjacência podemos calcular diversos parâmetros através dos quais é possível caracterizar tais redes.

Geralmente, as informações contidas nesta matriz são utilizadas apenas como sendo uma base para cálculos, dificilmente sendo observadas além da sua representação sob a forma de um grafo. No caso das proteínas entretanto, a visualização da matriz adjacência em sua forma bruta, ou seja, em sua forma matricial, evidencia uma característica associada a tais redes que contrasta com as redes sociais analisadas até o momento. A fim de facilitar a comparação visual entre as matrizes adjacência das redes sociais e das redes de contatos das proteínas, introduzimos como exemplo a ilustração apresentada na Fig. 8.5.

Figura 8.5: Exemplo de (a) grafo não direcionado e não ponderado, composto por 10 vértices e 12 arestas; (b) matriz adjacência do respectivo grafo; (c) representação alternativa da matriz adjacência, onde cada pixel preenchido corresponde a um valor não nulo da matriz.

Na Fig. 8.5a, vemos um simples grafo composto por 10 vértices e 12 arestas, previamente utilizado no Cap. 5, o qual corresponde a um grafo não direcionado e não ponderado. Logo, a matriz adjacência correspondente a este grafo contém apenas os valores 0 e 1 e é apresentada na Fig. 8.5b. Podemos, entretanto, visualizá-la alternativamente sob a forma de uma imagem representativa dos valores contidos nesta matriz (Fig. 8.5c).

Nesta representação como imagem, cada par de vértices corresponde à posição de um pixel na imagem, sendo que a intensidade desse pixel é definida pelo valor da aresta que conecta tais vértices. Sendo assim, a imagem resultante será uma imagem em branco e preto (0 e 1). Uma vez que o valor 0 indica que não há contato entre entidades representadas pelos vértices, e 1 representa tal contato, os pontos pretos apresentados na imagem exibida na Fig. 8.5c são simplesmente as conexões existentes entre tais entidades. Baseado nesta ideia, imagens representando as matrizes adjacência das diversas redes sociais previamente estudadas neste trabalho, são apresentados nas Figs. 8.6a a 8.6f.

Analizando-se tais imagens, é difícil inferir suas características uma vez que, por exemplo, as redes de Jazz e dos Golfinhos não apresentam padrões apreciáveis. Além disso, se a ordem dos vértices for embaralhada, porém preservando a relação de adjacência entre os mesmos, padrões distintos podem surgir nestas representações. Dessa forma, a validade desta representação é deveras restrita.

Entretanto, no caso das redes geradas para as proteínas, na qual a numeração dos vértices corresponde à numeração dos aminoácidos no *backbone* proteico, a representação em imagem fornece uma primeira característica destas redes, a qual pode ser observada com uma simples inspeção visual. Nas Figs. 8.7a a 8.7e são mostradas as representações das matrizes adjacência geradas para a proteína 1AD0-A por meio dos cinco critérios anteriormente estabelecidos. Nas Figs. 8.8a a 8.8e, são apresentadas as imagens geradas para as matrizes adjacência para a proteína 1A6C-A, a título de comparação.

Comparando-se tais imagens, vemos que existe um padrão que se preserva, não só entre os diferentes critérios de criação das redes, mas também entre ambas as proteínas. Este padrão corresponde à linha que se forma próximo à diagonal principal da imagem. Mais precisamente, na primeira diagonal superior. Dado que a matriz adjacência é simétrica, este mesmo padrão está presente na primeira diagonal inferior. Tal padrão origina-se da forte relação entre aminoácidos sequenciais no *backbone* da proteína, uma vez que dois a dois eles são adjacentes. Ou seja, $e_{ij} = 1$ para os aminoácidos i e j , sendo $j = i + 1$. Sendo assim, esta característica topológica da proteína é preservada na sua representação em redes complexas. Como veremos na Sec. 8.5, esta característica é crucial para a identificação das comunidades que correspondam aos domínios da proteína.

Podemos observar ainda nestas imagens que, variando-se o critério de criação das redes, a estrutura básica destas se preservam. Este fato é evidenciado uma vez que todas as imagens referentes aos cinco critérios para uma dada proteína são semelhantes. A diferença entre as imagens é que, dependendo do R_c ou do I_c , o número de arestas se altera. Entretanto, esta variação não elimina um número suficiente de contatos tal que a forma básica destas imagens (ou topologia das redes) sofra alterações significativas. Este fato evidencia que, ao aplicarmos métodos de identificação de comunidades, deve ser possível obter resultados semelhantes, ou até mesmo iguais, por meio de qualquer uma destas redes, independentemente do critério adotado.

Figura 8.6: Imagem representativa das matrizes adjacência das redes: (a) clube de karatê, (b) Golfinhos, (c) Football, (d) PolBooks, (e) Jazz e (f) LesMis.

Figura 8.7: Matrizes adjacência referente à proteína 1AD0-B, cuja transcrição para redes complexas foi efetuada utilizando os critérios de distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).

Figura 8.8: Matrizes adjacência referente à proteína 1A6C-A, cuja transcrição para redes complexas foi efetuada utilizando os critérios de distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).

8.3 Propriedades das redes proteicas

Uma forma de classificar redes complexas é, em geral, por meio da distribuição do grau de seus vértices. De acordo com esta distribuição, podemos classificar as redes como sendo redes aleatórias ou redes livres de escala. Em uma rede aleatória, cada aresta pode estar presente, ou ausente, com igual probabilidade [6]. Desta forma, a distribuição do grau dos vértices se aproxima de uma distribuição de Poisson no limite de grafos com $N \rightarrow \infty$. Por outro lado, redes construídas a partir de sistemas reais em geral, diferem das redes aleatórias. Considerando-se a distribuição do grau dos vértices destas redes, estas frequentemente apresentam uma cauda com queda acentuada e valores muito superiores ao valor médio. Neste caso, as redes em geral seguem uma distribuição sob a forma de lei de potências e são ditas livres de escala.

A fim de determinar se as redes correspondentes às cadeias das proteínas seguem alguma das distribuições mencionadas, ou algum outro padrão, mostramos nas Figs. 8.9a a 8.9e, as distribuições dos graus dos vértices das redes obtidas para a proteína 1AD0-B. Nestas figuras são apresentadas as distribuições oriundas de cada uma das redes geradas segundo os cinco critérios de transcrição adotados nesta tese. Nestes gráficos, observamos que a distribuição dos graus dos vértices não seguem as distribuições de Poisson, Gaussiana e, tampouco de lei de potências (os parâmetros e ajustes referentes às curvas normal e de Poisson ajustadas estão apresentadas nos próprios gráficos). Sendo assim, as redes não podem ser consideradas como aleatórias e nem livres de escala uma vez que observamos que a distribuição não se assemelha a uma lei de potências.

Na realidade, a forma adequada para representar a distribuição dos graus dos vértices é por meio de gráficos com escala logarítmica, nos quais uma distribuição de lei de potências corresponderia a uma reta. Esta representação não foi apresentada, porém a análise de tais redes foi efetuada com gráficos em escala log-log e log-linear, no intuito de buscar eventuais distribuições exponenciais.

Nas Figs. 8.10a a 8.10e estão apresentadas as distribuições dos valores médios obtidos para todas as 56 cadeias analisadas, para cada critério de transcrição da rede. Vemos nestes gráficos que os valores médios são próximos aos valores médios daqueles obtidos para a proteína 1AD0-B, para cada critério de transcrição. Considerando como exemplo as distribuições mostradas nas Figs. 8.9a e 8.10a, temos que os valores médios, $\langle k \rangle = 7,41(16)$ e $\langle k \rangle = 7,72(42)$ respectivamente, são compatíveis em até em $z = 0,75$ desvios padrão.

Figura 8.9: Distribuição do grau dos vértices para as redes da proteína 1AD0-B, utilizando as redes geradas utilizando os critérios de distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).

Figura 8.10: Distribuição dos valores médios dos graus dos vértices obtidos para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para cada critério de criação das redes.

Figura 8.11: Distribuição dos valores de graus máximos dos vértices obtidos para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para cada critério de criação das redes.

Outro fator importante à análise da distribuição dos graus dos vértices corresponde ao grau máximo destas redes. No caso da proteína 1AD0-B, o maior grau observado, dentre as redes criadas compreendendo os cinco critérios de transcrição, correspondeu a $k_{\max} = 16$. Considerando-se todas as proteínas analisadas, o maior grau observado correspondeu a $k_{\max} = 20$, sendo que as distribuições compreendendo todos os valores máximos são mostradas nas Figs. 8.11a a 8.11e. Com base nestes fatos, podemos concluir que existe um limite de conexões que um aminoácido pode ter. Este fato pode ser explicado, considerando que existe uma limitação física ao número de aminoácidos que podem ocupar uma certa região do espaço a uma determinada distância de um aminoácido arbitrário considerando-se a estrutura da proteína [149].

Os valores de grau médio $\langle k \rangle$ e grau máximo $k_{\max}$ para todas as cadeias proteicas utilizadas nesta tese estão apresentadas na Tab. A.1 em apêndice (Ap. A.1).

Outra característica bastante analisada em redes complexas é a chamada assortatividade. Este termo, oriundo da literatura de biologia e epidemiologia, corresponde ao viés para o estabelecimento de conexões entre vértices de características similares [13]. Este viés pode ser exemplificado utilizando redes sociais que mapeiam os laços de amizade entre indivíduos. Nestas redes pode ser observada uma preferência de associação entre os indivíduos dirigida pelo fato de que os padrões de amizade entre indivíduos são fortemente afetados por fatores como etnia, idioma, idade, etc. Sendo assim, no caso de redes em que indivíduos preferem se associar a indivíduos com características similares, dizemos que tais redes são assortativas. Caso a rede não apresente esta característica, ou seja, caso não haja nenhum tipo de viés (ou preferência) para o estabelecimento de conexões, dizemos que tais redes são não assortativas.

A assortatividade está sempre relacionada a alguma característica que propicie a distinção entre os vértices. No caso em que esta distinção se dá por causa do idioma, dizemos que tal rede é assortativa pelo idioma. Entretanto, em grande parte dos casos a assortatividade é considerada em termos dos graus dos vértices, ou seja, assortativa pelo grau. Nestas redes existe a preferência de vértices com graus elevados possuírem conexões com outros vértices de grau também elevado. Desta forma, a assortatividade pode exercer grande influência sobre as características estruturais das redes. Por exemplo, redes assortativas por algum tipo de característica tendem a apresentar estrutura de comunidades considerando-se tal característica [13]. Contrariamente ao grau do vértice, que fornece apenas características pontuais de cada vértice, a assortatividade pelo grau relaciona tais vértices por meio de seus graus. Sendo assim, o caráter assortativo, ou não assortativo é uma informação útil ao estudo de redes complexas.

A fim de determinar se as redes formadas pelas cadeias de proteínas avaliadas nesta tese apresentam caráter assortativo, efetuamos o cálculo da assortatividade para cada uma das cinco redes geradas segundo os critérios de transcrição previamente estabelecidos. A distribuição dos valores de assortatividade por critério de criação das redes

é apresentada nas Figs. 8.12a a 8.12e. Estas distribuições foram criadas com base nos valores apresentados na Tab. A.2 em apêndice (Ap. A.2).

Figura 8.12: Distribuição dos valores de assortatividade obtidos para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para cada critério de criação das redes.

Figura 8.13: Distribuição dos valores de assortatividade obtidos para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para cada critério de criação das redes.

Avaliando-se os valores presentes em tal tabela e nos gráficos das distribuições, vemos que as redes das cadeias das proteínas utilizadas nesta tese apresentam caráter assortativo. Este fato pode ser explicado considerando-se que cada vértice corresponde a um aminoácido distinto que pode apresentar caráter hidrofóbico ou polar; dada esta característica, o aminoácido encontrar-se-á na região mais interna da proteína ou mais externa. Sendo assim, intuitivamente, os aminoácidos hidrofóbicos, por serem mais internos, devem apresentar um maior número de ligações com outros aminoácidos a fim de se isolarem das moléculas de água que os circundam. De fato Alves e Martinez [146], em seu estudo das propriedades topológicas de redes de aminoácidos determinaram as probabilidades de estabelecimento de conexões entre aminoácidos dados seus caracteres hidrofóbico ou

polar, são estas: 0,424(5) para pares de aminoácidos hidrofóbicos-hidrofóbicos, 0,419(2) hidrofóbicos-polares e 0,157(3) para polares-polares [146]. Dessa forma, os resultados obtidos nesta tese estão de acordo com os obtidos pelos autores.

Até o momento, apresentamos resultados referentes a proteínas com apenas dois domínios, o que levanta a questão quanto ao valor da assortatividade para proteínas com apenas um domínio, ou alternativamente, com mais de dois domínios. Por este motivo, selecionamos, aleatoriamente, na base de dados do CATH, 90 cadeias de proteínas não fragmentadas de cada um dos seguintes grupos: cadeias com apenas 1 domínio, com 2, 3 e 4 domínios, totalizando 360 proteínas. Para cada uma das cadeias foram geradas as respectivas redes utilizando apenas o critério de transcrição CR, uma vez que este corresponde ao modelo mais refinado utilizado nesta tese, do ponto de vista de contatos entre resíduos. As cadeias selecionadas, e os correspondentes valores de assortatividade, são apresentados nas Tabs. A.3 a A.6 em apêndice (Ap. A.2). Nas Figs. 8.13a a 8.13d, são apresentadas as distribuições obtidas para as assortatividades dos conjuntos de proteínas, separadas por número de domínios.

No gráfico da Fig. 8.14 é apresentado um gráfico de caixa (*box plot*) correspondente aos conjuntos de dados para cada número de comunidades. Neste gráfico, podemos observar que as medianas dos conjuntos com 2, 3 e 4 comunidades são bastante próximas quando comparadas entre si, em oposição à mediana do conjunto com apenas uma comunidade.

Figura 8.14: Comparação da assortatividade para os conjuntos de proteínas com diferentes números de domínios.

Afim de avaliar se os valores médios de cada conjunto de dados são compatíveis entre si, efetuamos o teste de análise da variância (ANOVA) [152]. Neste teste, consideramos

$H_0 : \mu_i = \mu_j$, sendo H_0 a hipótese nula e μ_i e μ_j os valores médios dos conjuntos i e j , os quais correspondem ao conjunto analisado e cujos valores são atribuídos segundo o número de domínios do conjunto. Além disso, como não podemos dizer que as distribuições apresentadas nas Figs. 8.13a a 8.13d seguem distribuições normais, utilizamos também o teste de Kruskal-Wallis [153]. Este teste é utilizado quando não se acredita que as distribuições sejam gaussianas, porém considerando-se que os conjuntos de dados sigam distribuições iguais. Nesse caso, a mediana das amostras é avaliada, sendo a hipótese nula a igualdade destas medianas. Na Tab. 8.3 são apresentados os valores do nível descritivo (p) obtidos em cada teste.

(i, j)	$p - ANOVA$	$p - KW$
(1,2)	0,00021	0,00031
(1,3)	0,0011	0,0027
(1,4)	0,0033	0,0071
(2,3)	0,30	0,14
(2,4)	0,24	0,19
(3,4)	0,79	0,96

Tabela 8.3: Nível descritivo p obtido para os testes de hipótese ANOVA e Kruskal-Wallis (KW) entre os conjuntos com i e j domínios.

Nesta tabela, podemos observar que o parâmetro assortatividade é capaz de distinguir entre proteínas com um domínio de proteínas com mais de um domínio, a um nível de significância $\alpha = 0,05$, em ambos os testes. A assortatividade, entretanto, não é capaz de distinguir o número de domínios quando a proteína apresenta mais de um domínio. Considerando que aproximadamente 77% das proteínas listadas na base de dados CATH apresentam um único domínio, este parâmetro pode ser utilizado como auxiliar em prever se a proteína possui um ou mais domínios. Este fato é importante, se considerarmos que os métodos de identificação de domínios utilizando redes complexas não são capazes de identificar apenas um domínio.

8.4 Identificação de comunidades com o método ALGO

Nesta seção passamos ao problema de identificação de comunidades nas redes geradas para proteínas utilizando os cinco critérios de transcrição adotados neste capítulo. Para a identificação dos domínios, utilizamos o método ALGO utilizando o conjunto de parâmetros $ALGO(N, 2, 250, 25)$. Por meio deste método obtivemos todas as divisões das redes em comunidades até que $Q_{\max}$ fosse obtido. Sendo assim, dentre as 56 proteínas analisadas nesta tese, selecionamos duas proteínas, 1EZM-A e 1D5M-A, para ilustrar o comportamento médio obtido para as demais proteínas.

Na Tab. 8.4, apresentamos os resultados referentes à proteína 1EZM-A. Nesta tabela, estão apresentados os resultados referentes aos cinco critérios de transcrição das proteínas em redes complexas. Considerando que a proteína 1EZM-A, segundo a base de dados CATH, é composta por dois domínios, nas colunas AL e ALGO são apresentadas as divisões das proteínas em dois domínios, obtidas pelo método AL e otimizada pelo método ALGO. Na coluna $Q_{\max}$ é apresentado o valor máximo obtido para a função modularidade. Logo, na coluna “Divisão $\rightarrow Q_{\max}$ ”, apresentamos a divisão correspondente ao $Q_{\max}$.

Os domínios da proteína 1EZM-A, fornecidos pelo método CATH são: (1-152)(153-298). Comparando-se esta divisão, com as configurações de máxima modularidade apresentadas na Tab. 8.4, vemos que estas correspondem a divisões da rede entre 4 a 6 domínios. Além disso, tais domínios não são sequenciais. Considerando que este resultado corresponde aos resultados típicos obtidos para as 56 proteínas analisadas nesta tese, podemos concluir que a simples determinação da divisão de máxima modularidade para as redes criadas para as proteínas, utilizando os métodos de transcrição testados nesta tese, não fornecem resultados compatíveis com a divisão real destas proteínas. Entendemos como divisão real aquela fornecida pelo método CATH, adotado nesta tese como sendo o *padrão-ouro*.

Considerando que a proteína 1EZM-A possui supostamente dois domínios, analisando as divisões em dois domínios obtidos pelo método AL, vemos que os resultados são igualmente insatisfatórios. Dentre as divisões fornecidas por este método, o resultado que mais se aproximou da divisão do método CATH, corresponde àquela da rede criada pelo critério $C_\alpha : R_c = 8,0\text{\AA}$. Este resultado, entretanto, apresenta domínios não sequenciais; além disso, o aminoácido limítrofe superior entre os domínios corresponde ao aminoácido 199, enquanto que o CATH identifica este como sendo o aminoácido número 152, o que resulta no erro relativo $\delta_E = 30,92\%$. Considerando-se os resultados de duas comunidades, otimizados pelo método ALGO, os resultados que mais se aproximam daqueles fornecidos pela base de dados CATH correspondem às redes $C_\alpha : R_c = 7,0$ e $8,0\text{\AA}$. Novamente, tais resultados fornecem domínios não sequenciais e também cometem erros da ordem de 30% na identificação dos aminoácidos limítrofes superiores.

Por outro lado, considerando-se a base de dados SCOP, esta estabelece que a proteína 1EZM-A possui apenas 1 domínio. Se considerarmos os cinco valores de assortatividade desta proteína, podemos efetuar um teste de hipóteses a fim de verificar se esta proteína possui 1 ou mais domínios. Para isto, efetuamos dois testes t, um considerando a distribuição dos valores de assortatividade de proteínas com 1 domínio e, o segundo considerando os valores de assortatividade para proteínas com 2 domínios. Para o primeiro teste, consideramos $H_0 : \mu_1 = \mu_e$ e $H_A : \mu_e > \mu_1$, sendo μ_1 o valor médio da distribuição para proteínas com 1 domínio e μ_e o valor médio da distribuição referente à proteína 1EZM-A. Para o segundo teste, consideramos $H_0 : \mu_2 = \mu_e$ e $H_A : \mu_e < \mu_2$, sendo μ_2 o valor médio da distribuição para proteínas com 2 domínios e, novamente, μ_e o valor

médio da distribuição referente à proteína 1EZM-A. Em ambos os testes as hipóteses nulas não foram rejeitadas com $p = 0,3054$ e $p = 0,2042$, respectivamente. Sendo assim, não podemos afirmar, com base nestes testes, se a proteína 1EZM-A tem 1 ou 2 domínios. Entretanto, o fato da hipótese nula do primeiro teste não ter sido rejeitada, isto indica a possibilidade desta proteína ter apenas 1 domínio conforme descrito na base de dados SCOP.

Da mesma forma que para a proteína 1EZM-A, apresentamos na Tab. 8.5, os resultados obtidos para a proteína 1D5M-A. Para esta proteína, constatamos novamente que a divisão em comunidades tal que o valor da função modularidade seja máximo, divide a proteína em um número maior de domínios do que o número considerado como real, de acordo com a base de dados CATH. Esta base de dados fornece os domínios (4-81)(82-181). A base de dados SCOP fornece para esta proteína a mesma divisão que a base de dados CATH. Infelizmente, a base de dados do método Domain Parser não fornece informações a respeito de ambas as proteínas.

Analizando os resultados fornecidos pelo método AL para duas comunidades, vemos que os limites dos domínios desta proteína não estão corretamente estabelecidos e, além disso, os domínios encontram-se segmentados. Considerando os resultados fornecidos pela otimização efetuada pelo método ALGO, obtemos resultados condizentes com o esperado para todos os métodos de transcrição, exceto pelo método DP. Vemos ainda, que utilizando os métodos de transcrição de distâncias entre C_α , o erro na identificação de domínios corresponde a apenas um aminoácido mal classificado, sendo que para raios de corte $R_c = 7,0$ e $7,5\text{\AA}$, este erro é para mais e, para o raio de corte $R_c = 8,0\text{\AA}$, este erro é para menos. Entretanto, no caso do método de transcrição CR, os limites são corretamente identificados. Devemos ressaltar, que os resultados obtidos para a proteína 1D5M-A não correspondem aos resultados frequentemente obtidos nesta tese, sendo o resultado obtido para a proteína 1EZM-A considerado como um exemplo que melhor representa os resultados obtidos para a maior fração das proteínas analisadas.

Com base nos resultados apresentados para as proteínas 1EZM-A e 1D5M-A e, considerando que estes representam resultados típicos obtidos para as 56 proteínas analisadas nesta tese, concluímos que utilizar o método AL/ALGO com função objetivo modularidade, nas redes criadas a partir dos cinco critérios utilizados, não é adequado para a determinação de domínios em proteínas. É uma possibilidade, entretanto, que as comunidades referentes a $Q_{\max}$, identificadas por estes métodos estejam identificando grupos de aminoácidos segundo características mais específicas uma vez que ocorre uma segmentação dentro de cada domínio. Uma possível característica a ser observada é o caráter hidrofóbico/hidrofílico dos aminoácidos; mas, esta possibilidade requer estudos mais aprofundados e por este motivo é proposta como possível estudo futuro.

Outro fator que certamente influencia na identificação de domínios em proteínas, utilizando os critérios de transcrição aqui adotados, é o fato de que dois a dois, os

aminoácidos são conectados ao longo do *backbone* proteico. Esta característica, ausente nas redes sociais estudadas nesta tese, possivelmente exerce influência sobre os métodos de identificação de comunidades aqui propostos. Sendo assim, para a utilização dos algoritmos de identificação de comunidades em redes formadas por aminoácidos, duas possibilidades merecem ser exploradas. A primeira possibilidade corresponde à criação de redes ponderadas que capturem a essência da estrutura de domínios em proteínas e, em segundo lugar a criação de funções **objetivo** ou heurísticas diferenciadas, que levem em consideração fatores como tamanho mínimo de domínios, estruturas secundárias, além da característica do *backbone* proteico.

Os resultados obtidos para a proteína 1D5M-A, por sua vez, corrobora a possibilidade de identificar-se corretamente os domínios das proteínas pelo método AL/ALGO. Entretanto, um critério alternativo para criação das redes deve ser desenvolvido.

PDB ID: 1EZM-A

Critério : R_c	AL	ALGO	$Q_{\max}$	Divisão $\rightarrow Q_{\max}$
$C_\alpha : 7, 0\text{\AA}$	(1-142;167-182;187-197)(143-166;183-186;198-298)	(1-134;187-197)(135-186;198-298)	0,693	(1-39;53-77)(40-52;99-134;185-197)(78-98;135-152;166-184)(153-165;198-231;260-262)(232-259;263-298)
$C_\alpha : 7, 5\text{\AA}$	(1-146;167-182;189-195)(147-166;183-188;196-298)	(1-133;189-195)(134-188;196-298)	0,684	(1-1;25-41;51-56;73-73;76-77;99-104;115-120)(2-24;57-66;68-72;74-75)(42-50;67-67;105-114;121-134;189-195)(78-98;135-146;167-182)(147-154;250-298)(155-166;183-188;196-249)
$C_\alpha : 8, 0\text{\AA}$	(1-146;166-199)(147-165;200-298)	(1-134;187-197)(135-186;198-298)	0,665	(1-31;55-66;68-75;77-78)(32-54;67-67;76-76;98-132)(79-97;133-146;167-197)(147-154;250-298)(155-166;198-249)
DP	(1-93;96-141;166-181;185-198)(94-95;142-165;182-184;199-298)	(1-86;96-142;171-183;185-198)(87-95;143-170;184-184;199-298)	0,658	(1-8;13-37;53-65;70-82)(9-12;66-69;109-109;111-111;124-133;185-209)(83-97;134-150;166-168;170-183)(38-52;98-108;110-110;112-123)(151-165;210-230;259-270;293-298)(169-169;184-184;231-258;271-292)
CR	(1-141;169-197;298-298)(142-168;198-297)	(1-88;91-142;184-197;298-298)(89-90;143-183;198-297)	0,660	(1-9;12-33;36-38;52-65;67-77;132-132;298-298)(10-11;66-66;107-111;122-131;186-196)(34-35;39-51;95-106;112-121)(78-94;133-141;169-185;197-197)(142-168;198-234;259-262)(235-258;263-297)

Tabela 8.4: Divisões em comunidades para a proteína 1EZM-A obtidas pelos métodos AL e ALGO para os cinco critérios de transcrição das proteínas em redes complexas, são eles: distâncias entre C_α com raios de corte $R_c = 7, 0, 7, 5$ e $8, 0\text{\AA}$, DP e CR. Na coluna AL apresentamos a divisão em duas comunidades obtidas pelo método ALGO, na coluna $Q_{\max}$, apresentamos o valor máximo de modularidade e, na coluna Divisão $\rightarrow Q_{\max}$, apresentamos a divisão da rede correspondente a $Q_{\max}$.

PDB ID: 1D5M-A

Critério : R_c	AL	ALGO	$Q_{\max}$	Divisão $\rightarrow Q_{\max}$
$C_\alpha : 7, 0\text{\AA}$	(4-11;21-58)(12-20;59-181)	(4-83)(84-181)	0,713	(4-31)(32-54)(55-81)(81-90;107-116;134-149;167-172)(91-106;129-133;150-159;180-181)(117-128;160-166;173-179)
$C_\alpha : 7, 5\text{\AA}$	(4-12;21-63)(13-20;64-181)	(4-83)(84-181)	0,702	(4-33)(34-57)(91;117-129;157-181)(92-107;130-134;148-156)(58-81)(82-90;108-116;135-147)
$C_\alpha : 8, 0\text{\AA}$	(4-67;82-181)(68-81)	(4-81)(82-181)	0,679	(4-10;23-57)(11-22;58-80)(81-90;108-115;135-148)(116-128;156-181)(91-107;129-134;149-155)
DP	(4-66;81-181)(67-80)	(82-138;140-181)(4-81;139)	0,656	(4-12;20-42;54;56;136;138-139)(93-105;120-133;151-162;177-181)(13-19;55;57-81)(43-53;89-92;105-108;118-119;134-135;147-150;163-166;172-176)(81-88;109-117;137-140-146;167-171)
CR	(4-70;81-181)(71-80)	(4-82)(83-181)	0,671	(4-33;41-54)(83-88;110-116;135-147;167-171)(34-40;55-81)(93-104;122-127;153-161;177-181)(82;89-92;105-109;117-121;128-134;148-152;162-166;173-176)

Tabela 8.5: Divisões em comunidades para a proteína 1D5M-A obtidas pelos métodos AL e ALGO para os cinco critérios de transcrição das proteínas em redes complexas, são eles: distâncias entre C_α com raios de corte $R_c = 7, 0, 7, 5$ e $8, 0\text{\AA}$, DP e CR. Na coluna AL apresentamos a divisão em duas comunidades obtidas pelo método AL, na coluna ALGO apresentamos a divisão em duas comunidades obtidas pelo método ALGO, na coluna $Q_{\max}$, apresentamos o valor máximo de modularidade e, na coluna Divisão $\rightarrow Q_{\max}$, apresentamos a divisão da rede correspondente a $Q_{\max}$.

8.5 Identificação de comunidades em proteínas com dois domínios

Alternativamente aos métodos AL e ALGO, desenvolvemos nesta tese um método puramente baseado na função modularidade e especializado na determinação dos limites dos domínios nas cadeias de proteínas com dois domínios sequenciais. O método ALGO, como mostrado na Sec. 8.4, baseia-se na identificação da divisão de uma rede tal que o valor da função modularidade associado a tal divisão seja máximo. Como resultado, encontram-se divisões com números de comunidades superiores ao número de domínios reais da proteína em questão. Além disso, as soluções fornecidas pelos métodos AL e ALGO correspondentes ao número correto de domínios, na maior parte dos casos analisados, fornece divisões da rede com domínios não sequenciais. Estes, novamente, não correspondem à realidade das cadeias analisadas.

Como mencionado na Sec. 8.4, o *backbone* proteico exerce influência na identificação de domínios nas proteínas. Sendo assim, diferentemente do método ALGO, cujo objetivo é determinar a divisão da rede que maximize a função modularidade, o método proposto nesta seção utiliza o fato de que os domínios das cadeias analisadas nesta tese são sequenciais. Além disso, utiliza-se o fato de que os aminoácidos também são sequenciais ao longo do *backbone* proteico. Sendo assim, dada uma cadeia polipeptídica, explora-se todas as divisões possíveis em duas comunidades, simplesmente quebrando esta cadeia em duas partes e calculando-se o valor da função modularidade referente a tal quebra. Na Fig. 8.15 é apresentado um diagrama ilustrativo do procedimento.

Figura 8.15: Diagrama ilustrativo do procedimento desenvolvido para identificação de comunidades em proteínas com dois domínios sequenciais. Dada uma cadeia polipeptídica com N aminoácidos, explora-se todas as divisões possíveis em duas comunidades sequenciais simplesmente quebrando esta cadeia em duas partes e calculando-se o valor da função modularidade Q referente a tal quebra.

Considerando uma proteína com N aminoácidos, o procedimento é iniciado com a divisão da rede em duas comunidades, sendo que a primeira comunidade corresponde apenas ao primeiro aminoácido. Para esta divisão, é calculado o valor da modularidade, o qual passa a ser considerado como máximo. Na próxima iteração o vértice 2 é trocado de comunidade passando a pertencer à mesma comunidade do vértice 1. Novamente, a função modularidade é avaliada. Caso esta modularidade seja maior que a máxima modularidade já obtida, esta passa a ser considerada como máxima modularidade. Repetindo-se o procedimento $N - 1$ vezes, até que a segunda comunidade corresponda apenas ao n -ésimo aminoácido, a curva do valor da função modularidade para a cada divisão é construída, sendo que o valor máximo corresponde à divisão em dois domínios da proteína em análise.

Este procedimento foi executado para todas as 56 proteínas listadas na Tab. 8.2. Apresentamos nas Figs. 8.16a a 8.16e e 8.17a a 8.17e, as curvas resultantes para as proteínas 1AD0-B e 1A6C-A, respectivamente, além de cada um dos métodos de transcrição das proteínas em redes complexas utilizados nesta tese. Por meio destes gráficos fica ilustrado o comportamento médio obtido para as demais proteínas investigadas.

Podemos observar nas Figs. 8.16a a 8.16e, que o método proposto nesta seção foi capaz de identificar corretamente os limites dos domínios da proteína 1AD0-B. Isto, considerando-se os limites do método CATH como padrão ouro. Vale ressaltar que para esta proteína o método SCOP fornece o mesmo resultado que o CATH, enquanto que a base de dados do método Domain Parser não fornece informações a respeito desta cadeia.

Para o caso da proteína 1A6C-A, os resultados apresentados nas Figs. 8.17a a 8.17e, mostram que o método proposto nesta seção não identificou corretamente os domínios quando comparado com a base de dados do CATH. O erro relativo percentual na determinação dos limites dos domínios, utilizando os valores da base de dados CATH como referência é da ordem de 43,10%. Neste caso, o método SCOP indica que esta proteína na realidade é composta por 3 domínios, cujos limites são: (1-177)(178-348)(349-513). O método Domain Parser, por sua vez fornece os seguintes limites: (1-175)(176-349;510-513)(350-509), os quais por sua vez diferem de ambos os métodos e fornece resultados não sequenciais.

Considerando os resultados fornecidos pelo método proposto nesta tese, vemos que as curvas de modularidade apresentam dois picos e, que estes correspondem, aproximadamente, aos limites identificados pelo método SCOP. Este fato pode induzir-nos a pensar que haverá tantos picos quantos domínios, uma vez que, neste caso houve uma certa concordância com o método SCOP e Domain Parser. Este comportamento, entretanto, não é reproduzido para proteínas com 3 domínios.

Figura 8.16: Valores de modularidade obtidos para as possíveis divisões em dois domínios contínuos da rede referente à proteína 1AD0-B, utilizando as redes geradas com os critérios de distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0 \text{ \AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).

Figura 8.17: Valores de modularidade obtidos para as possíveis divisões em dois domínios contínuos da rede referente à proteína 1A6C-A, utilizando as redes geradas com os critérios de distâncias entre C_α com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, contato entre resíduos (CR) e critério do Domain Parser (DP).

Como exemplo, mostramos na Fig. 8.18, a curva obtida para a proteína 3FKS-A utilizando apenas o critério de transcrição de distâncias entre C_α com raio de corte $R_c = 7,0\text{\AA}$. Neste gráfico, em oposição àqueles obtidos para as proteínas 1AD0-B e 1A6C-A, não há um pico, ou picos, bem definidos. Ao invés disso, a curva apresenta forma alargada e esta sim, corresponde à curva típica para proteínas com três domínios.

Figura 8.18: Valores de modularidade obtidos para as possíveis divisões em dois domínios contínuos da rede referente à proteína 3FKS-A, utilizando a rede gerada utilizando os critério de distâncias entre C_α com raio de corte $R_c = 7,0\text{\AA}$.

Por outro lado, se considerarmos o valor da função modularidade referente à divisão em três comunidades: (24-95)(96-381)(382-509), teremos $Q = 0,5151$, cujo valor é superior a todos os valores obtidos no experimento com apenas dois domínios sequencias. Este fato indica a possibilidade de desenvolvermos um método de varredura da proteína baseado na função modularidade que seja capaz de determinar a verdadeira divisão da proteína em domínios. Tal abordagem entretanto, necessita de uma estratégia diferenciada para tal varredura, sendo que a complexidade do método aumenta pelo menos da ordem de n para cada número de domínios a serem varridos.

Considerando novamente os resultados obtidos para as proteínas 1AD0-B e 1A6C-A, mostramos na Fig. 8.19a todas as curvas previamente apresentadas na Fig. 8.16, porém sobrepostas. Da mesma forma, na Fig. 8.19b é apresentada a sobreposição das curvas da Fig. 8.17. Comparando-se as curvas em ambos os gráficos, fica claro que existe apenas uma pequena diferença entre as curvas obtidas por um ou outro critério de transcrição da proteína. Vemos inclusive, que as posições dos valores máximos dos picos não mudam de posição, como pode ser constatado nos gráficos para cada método de transcrição em separado. Este fato indica que, sob a ótica do método proposto nesta seção, pode

não haver diferença significativa ao utilizarmos qualquer um dos métodos de transcrição empregados neste capítulo.

Figura 8.19: Sobreposição das curvas de modularidade para identificação de domínios para as proteínas 1AD0-B e 1A6C-A.

Apresentamos em apêndice (Ap. A.3) uma tabela (Tab. A.7) contendo todos os valores das modularidades máximas obtidas para as 56 proteínas analisadas nesta tese, para os cinco critérios de transcrição da proteína em redes complexas. Nas Figs. 8.20a a 8.20e, são apresentadas as distribuições dos valores máximos de modularidade obtidos para todas as 56 proteínas para cada método de transcrição. Observamos nestes gráficos que os valores máximos de modularidade obtidos estão compreendidos, em geral no intervalo de $[0, 38 - 0, 50]$.

Analisando as distribuições dos erros relativos cometidos na identificação dos domínios de cada uma das proteínas analisadas, mostradas nas Figs. 8.21a a 8.21e, vemos que as distribuições dos erros, para todos critérios de transcrição são bastante semelhantes. Entretanto, o método de transcrição CR é o que apresentou maior número de acertos (erros relativos no intervalo $[0, 0 - 0, 05]$), correspondendo a aproximadamente 62,26% das proteínas analisadas. Por outro lado, este valor é comparável com aqueles obtidos pelos métodos DP e $C_\alpha : R_c = 8, 0\text{Å}$, os quais apresentaram 61,82% e 60,38%, respectivamente. Dessa forma, concluímos que para o método proposto nesta seção (para identificação de domínios em proteínas com dois domínios sequenciais), podemos escolher qualquer um dos métodos de transcrição, apesar dos métodos $C_\alpha : R_c = 7, 0$ e $7, 5\text{Å}$ terem uma taxa de acerto um pouco inferior com relação aos outros três métodos.

Figura 8.20: Distribuição dos valores de máxima modularidade correspondente às divisões em dois domínios para as redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para os diferentes critérios de construção das redes.

Figura 8.21: Distribuição dos valores de erros relativos correspondente aos erros cometidos na determinação do ponto de divisão das redes de contatos geradas para as 56 cadeias de proteínas utilizadas neste tese, para os diferentes critérios de construção das redes.

Com base nestes resultados podemos concluir que o método proposto nesta seção é adequado para a indentificação de domínios em proteínas não fragmentados e com dois domínios sequenciais. Além disso, é possível desenvolver métodos similares para a indentificação de domínios para proteínas com mais de dois domínios.

Capítulo 9

Conclusões e perspectivas

O método proposto por Alves (AL) define uma escala de dissimilaridades. Esta escala propicia divisões de redes complexas em diversas configurações distintas, as quais correspondem a diferentes níveis de coesão. Analisando-se a rede do clube de karatê em cada nível de dissimilaridade, observamos que o método AL define como sendo a configuração mais coesa desta rede, àquela correspondente à sua divisão real. Tal divisão foi apresentada por Zachary como sendo a divisão de fato ocorrida no referido clube.

Partindo das divisões fornecidas pelo método AL, foi possível avaliar tais divisões por meio da função modularidade. Com isso, constatamos que a divisão em três comunidades correspondia à divisão de máxima modularidade. Tal divisão está de acordo com a definição de comunidades amplamente adotada por diversos autores, mas que no caso da rede do clube de karatê, não corresponde à sua divisão real. Sendo assim, o método AL foi capaz de fornecer ambas as soluções: tanto a divisão real desta rede quanto a melhor divisão, segundo a definição de comunidades.

Considerando-se as configurações fornecidas pelo método AL, a função objetivo modularidade forneceu a melhor divisão segundo a definição de comunidades; Entretanto, a simples maximização desta função pode fornecer configurações que falham ao revelar tanto a solução real para uma dada rede, quanto em fornecer a solução ótima segundo a definição de comunidades.

Este é o caso para a rede do karatê, na qual $Q_{\max}$ corresponde à divisão em 4 comunidades. A quarta comunidade identificada claramente não corresponde a uma comunidade segundo a definição de comunidades. Este fato é um exemplo da limitação de resolução da função modularidade, conforme discutida por Fortunato *et al.* e Good *et al.*, os quais mostram que a simples maximização desta função não necessariamente reflete a melhor divisão de uma rede em comunidades.

A otimização extrema generalizada (τ -EO), aplicada ao problema de determinação de comunidades em redes complexas, consiste em uma estratégia bastante atrativa quando comparada ao recozimento simulado (SA) e algoritmos genéticos (AG). Tal preferência

deve-se ao fato de que, ao contrário do SA, os novos estados são gerados a partir de correções direcionadas por funções objetivo locais e não simplesmente por variações aleatórias. No caso do AG em redes complexas, o processo de recombinação envolvendo dois cromossomos possivelmente ocasionaria um embaralhamento indesejável dos vértices, o que poderia desviar a solução do rumo ao ótimo. Apesar de ser possível desenvolver estratégias de recombinação que preservem as qualidades da solução do ponto de vista de comunidades, esta abordagem faria com que estivessemos introduzindo algum tipo de objetivo local. Desta forma, estaríamos recaindo na otimização extrema generalizada, porém com uma heurística possivelmente mais complicada.

As considerações acima nos levam a crer que o método τ -EO (dentre os citados) é mais adequado para a aplicação em problemas de determinação de comunidades em redes complexas. Verificamos que o método τ -EO híbrido, proposto e implementado nesta tese, foi capaz de obter resultados comparáveis àqueles obtidos pelos métodos considerados como *estado da arte* na detecção de comunidades. Sendo assim, optamos pela utilização do método τ -EO híbrido para otimizar os resultados fornecidos pelo método AL, uma vez que este se mostrou superior quando comparado aos demais métodos baseados na otimização extrema generalizada.

Os resultados obtidos pelo τ -EO híbrido para as redes sociais avaliadas, foram comparáveis àqueles obtidos por métodos propostos recentemente, como o VP e LPAm+. Entretanto, o τ -EO híbrido não obteve $Q_{\max}$ para três das redes avaliadas (PolBooks, Football e Jazz) e revelou um dos pontos fracos da função modularidade: a sua degenerescência.

No que diz respeito à função modularidade, diversos exemplos de onde ela falha são conhecidos, entretanto ainda não há uma função objetivo que forneça resultados superiores. O método τ -EO híbrido, apesar de não fornecer $Q_{\max}$ em todos os casos, é independente da função objetivo utilizada. No advento do surgimento de uma função que seja capaz de substituir a função modularidade, esta pode ser facilmente integrada ao método τ -EO. Além disso, o τ -EO híbrido constitui-se em um método de otimização, para o caso de redes complexas, mais simples do que o SA ou AG.

Considerar a combinação do método AL com a maximização da função modularidade (ou qualquer outra função objetivo capaz de revelar comunidades), é uma abordagem interessante. Considerando que o método AL fornece divisões próximas a divisões correspondendo a $Q_{\max}$, torna-se vantajoso submeter tais divisões a otimização ao invés de partir de soluções iniciais aleatórias.

O método ALGO, por sua vez, é inspirado no método de otimização extrema generalizado híbrido. Este método utiliza os resultados fornecidos pelo método AL como soluções iniciais a serem otimizadas por meio da função objetivo modularidade. Em analogia aos algoritmos genéticos, o método ALGO utiliza tais soluções como sendo a população inicial de candidatos soluções, tratando-as como sendo um cromossomo. Diferentemente do

método AG, o método ALGO não efetua operações de *crossing-over* nem qualquer outro tipo de mistura entre cromossomos; este apenas efetua operações de **mutação**.

Como a evolução de cada cromossomo ocorre independentemente das outras, o método ALGO é intrinsecamente paralelizável, o que é uma característica desejável em qualquer método que lide com sistemas cuja complexidade ou quantidade de dados, seja crescente. Da mesma forma que o método τ -EO híbrido, o método ALGO se baseia na substituição dos componentes menos adaptados de cada cromossomo, na busca pela solução ótima.

Aplicando o método ALGO às redes sociais, vimos que este foi capaz de determinar a solução ótima para a rede do clube de karatê. Excetuando-se esta última e a rede de Jazz, através do método ALGO obtivemos resultados ligeiramente inferiores aos obtidos pelo método τ -EO híbrido. Este fato mostra que assim como o τ -EO híbrido, o método ALGO também é dependente da solução inicial. Considerando que o método τ -EO híbrido utilizou 500 soluções iniciais e o método ALGO utilizou aproximadamente apenas uma dezena, consideramos que o método ALGO foi bem sucedido em sua otimização.

Para que fosse possível aplicar o método ALGO às soluções fornecidas pelo método AL, foi necessário o desenvolvimento de um método de agrupamento aglomerativo hierárquico capaz de processar a matriz de dissimilaridades fornecidas pelo método AL. Geradas as respectivas árvores hierárquicas (dendrogramas) os cromossomos foram extraídos e fornecidos como soluções iniciais para o método ALGO. Este método, denominado HAC++, foi validado e teve seu desempenho comparado com o da função `hclust` fornecida pelo ambiente de análise estatística R. A validação do método HAC++ foi efetuada através da comparação com os resultados fornecidos pela função `hclust`, a qual é extremamente bem estabelecida. A análise do desempenho dos métodos HAC++ e `hclust`, mostrou que o método HAC++ é superior à função `hclust`; sendo assim este torna-se uma excelente alternativa.

Desenvolvido o ferramental para análise e determinação de comunidades em redes complexas, aplicamos os métodos desenvolvidos nesta tese ao problema da determinação de domínios em proteínas. Para isto, foram avaliados cinco critérios de transcrição das proteínas sob a forma de redes complexas. Tais critérios foram: o critério de distâncias entre C_α , com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$; o critério DP, utilizado pelo **Domain Parser** e, o critério de contato entre resíduos (CR) utilizado por Vishveshwara *et al.*. Estes métodos, os quais são amplamente utilizados nos estudos de proteínas como redes complexas, mostraram-se equivalentes em todos os aspectos avaliados nesta tese. Exemplos destes aspectos são as distribuições dos graus dos vértices destas redes, bem como os graus máximos e médios, além da assortatividade. De uma forma qualitativa, utilizamos ilustrações da matriz adjacência a fim de avaliar as características das redes de aminoácidos. A partir desta avaliação, foi possível notar que de um método para outro o que varia é o número de arestas; por outro lado foi observado que a estrutura básica da rede se preserva. Além disso, comparando-se as redes de contato de

aminoácidos com as redes sociais, constatamos que as redes proteicas refletem a relação entre os aminoácidos do *backbone* proteico; o que potencialmente influi na detecção de comunidades ou domínios.

Para a análise da assortatividade das redes proteicas, utilizamos 360 proteínas distintas distribuídas igualmente entre proteínas com 1, 2, 3 e 4 domínios. A partir desta análise, constatamos que todas apresentam assortatividade positiva, o que indica preferência para a constituição de ligações. Esta preferência pode ser associada ao caráter hidrofóbico/hidrofílico dos aminoácidos, como reportado por Alves e Martinez. Além disso, o parâmetro assortatividade mostrou-se capaz de classificar proteínas em dois grupos, ou seja, o de proteínas com 1 domínio e o grupo de proteínas com mais de um domínio. Esta capacidade, entretanto, é um tanto quanto limitada dada a proximidade dos valores de assortatividade dentre as populações de proteínas analisadas.

A aplicação dos métodos AL/ALGO (com maximização da função modularidade) na identificação de domínios em proteínas infelizmente não foi capaz de identificar corretamente os limites dos domínios proteicos para a maior parte das proteínas analisadas. Este resultado até certo ponto inesperado, se deve ao fato de que tanto o método AL quanto o método ALGO possuem uma sensibilidade elevada, uma vez que identificam mais comunidades do que o número adotado como correto para as cadeias analisadas. Embora estes métodos não tenham sido capazes de fornecer a informação esperada a respeito dos domínios das proteínas, eles podem estar fornecendo informações mais finas a respeito da estrutura das proteínas, tais como o agrupamento de aminoácidos com base em seu caráter hidrofóbico/hidrofílico, ou até mesmo estruturas secundárias. Esta possibilidade deverá ser explorada em trabalhos futuros.

O excesso de resolução do método AL/ALGO é evidenciado quando consideramos que o método proposto por Vishveshwara *et al.* tem como base o método espectral, o qual de certa forma também é base para o método AL. É fato que o método espectral tem uma baixa resolução para a identificação de comunidades em redes complexas. Já o método AL possui uma resolução maior que a do método espectral. Por estes motivos, o método de Vishveshwara *et al.* aplicado a proteínas é mais bem sucedido do que o método AL, uma vez que uma menor resolução faz com que não seja possível detectar as comunidades menos evidentes de uma rede. Partindo desta hipótese, o método AL/ALGO têm o potencial de revelar os domínios de proteínas de forma satisfatória. Para isso, é necessário o desenvolvimento de redes que forneçam uma descrição mais acurada da estrutura proteica, tal que esta maior resolução dos métodos AL/ALGO seja traduzida em um método de identificação de domínios confiável e totalmente automatizado. Este aprimoramento das redes, possivelmente será alcançado através da transcrição de proteínas sob a forma de redes ponderadas, uma vez que através dos pesos das arestas deve ser possível decidir os melhores agrupamentos.

Outra alternativa à utilização deste método, é a criação de uma função objetivo (ou

heurística) específica para proteínas. Esta deve levar em consideração também as características das proteínas, tais como estruturas secundárias e caráter hidrofóbico/hidrofílico. Esta abordagem, entretanto, caminha no mesmo sentido da criação de redes ponderadas, o que por sua vez pode ser mais simples do que a criação de uma heurística específica.

Como exemplo de heurística, desenvolvemos nesta tese um método baseado na função modularidade, que utiliza a informação de que as proteínas estudadas possuem dois domínios sequenciais. Este método foi aplicado à 56 proteínas e obteve aproximadamente 75% de acertos com erro relativo no intervalo de $\delta_E \in [0 - 0, 1]$. Neste caso, esta simples heurística foi capaz de aproveitar as atuais técnicas de transcrição da proteína em redes complexas e a existente função modularidade para determinar os domínios das proteínas a um custo computacional $\mathcal{O}(n)$. Este método, apesar de ter sido aplicado para proteínas não fragmentadas, deve ser capaz de identificar domínios em proteínas fragmentadas. Com algumas adaptações, este método possui um potencial real de identificar domínios não sequenciais, além de poder ser aplicado em proteínas com mais de dois domínios.

Os resultados obtidos nesta tese criam novas possibilidades passíveis de serem exploradas em trabalhos futuros. Alguns exemplos de trabalhos possíveis consistem na exploração do método de otimização ALGO em problemas não apenas relacionados a redes complexas, uma vez que a exemplo da otimização extrema generalizada, a sua aplicabilidade não está restrita a redes complexas.

No caso do método ALGO aplicado em redes complexas, é possível explorar quais seriam os resultados se seguíssemos o mesmo procedimento experimental utilizado no método τ -EO híbrido. Ou seja, para cada cromossomo inicial buscar a solução ótima seguindo o processo recursivo, adotado no τ -EO híbrido. Utilizando esta estratégia, estaríamos explorando todos os cromossomos em todo seu potencial. Desta forma, poderíamos eventualmente obter resultados superiores aos fornecidos pelo método τ -EO híbrido.

Em relação à assortatividade como indicador de número de domínios, é necessário aumentar o número de proteínas em cada categoria de forma a obter uma melhor amostragem. Além disso, sugere-se realizar testes de sensibilidade e especificidade e verificar sua real aplicabilidade ao problema de determinação do número de domínios de uma proteína, possivelmente combinando-o com outros parâmetros. Esta possibilidade merece uma atenção especial, considerando que os métodos de identificação de domínios em redes complexas não conseguem determinar se uma rede é composta de apenas um domínio, uma vez que esta é a solução trivial de tais métodos.

Proteínas constituem uma categoria particular de redes, sendo portanto possível que a assortatividade seja capaz de determinar se a solução trivial é adequada.

O método AL/ALGO representa um bom método para a identificação de domínios em proteínas. A possibilidade deles estarem fornecendo informações estruturais mais detalhadas das proteínas deve ser explorada, além da sua utilização com diferentes funções

objetivo e diferentes metodologias de transcrição das proteínas em redes complexas. Por fim, o método desenvolvido nesta tese, para identificação de domínios em proteínas com dois domínios, deve ser generalizado para o caso de proteínas com mais de dois domínios.

Com base nestes resultados, concluímos que muitos objetivos foram alcançados nesta tese e que os horizontes foram ampliados revelando novas possibilidades a serem exploradas.

Apêndices

Apêndice A

Parâmetros calculados para as redes das cadeias de proteínas

A.1 Grau dos vértices

PDB ID	N	$C_\alpha(7, 0\text{\AA})$		$C_\alpha(7, 5\text{\AA})$		$C_\alpha(8, 0\text{\AA})$		$DP(4, 0\text{\AA})$		$CR(4\%)$	
		$\langle k \rangle$	$k_{\max}$	$\langle k \rangle$	$k_{\max}$	$\langle k \rangle$	$k_{\max}$	$\langle k \rangle$	$k_{\max}$	$\langle k \rangle$	$k_{\max}$
1A0H-A	157	6,03	13	6,91	17	7,55	17	5,98	14	5,40	13
1A0P-A	271	7,68	14	8,35	14	9,12	15	8,20	16	7,75	14
1A16-A	439	8,13	16	9,06	16	10,10	18	8,42	15	7,94	14
1A20-A	347	8,37	16	9,46	18	10,60	20	8,28	17	7,66	14
1A3L-H	212	7,34	12	8,25	13	9,14	15	6,68	15	6,13	14
1A41-A	221	7,56	14	8,18	14	8,94	16	8,03	15	7,68	15
1A45-A	173	7,90	15	9,07	15	10,19	17	7,58	18	7,01	14
1A4J-B	217	7,78	12	8,56	13	9,57	16	7,15	16	6,47	14
1A4K-A	216	7,83	13	8,79	14	9,55	15	7,23	15	6,51	14
1A5D-A	173	8,01	14	9,22	15	10,22	17	7,88	18	7,17	14
1A6A-A	176	7,20	13	8,11	14	8,85	15	7,20	14	6,48	13
1A6C-A	513	7,83	14	8,89	15	10,01	18	7,37	15	6,69	14
1A6Q-A	363	8,30	14	9,27	17	10,25	20	8,31	16	7,78	15
1A6T-C	210	7,84	13	8,69	14	9,48	15	7,12	15	6,36	15
1A87-A	297	8,03	13	8,74	14	9,47	16	8,17	15	7,69	14
1A8D-A	452	7,97	14	8,96	15	9,78	16	7,68	15	6,98	13
1A8P-A	257	7,83	14	8,77	15	9,59	16	7,72	14	6,94	14
1A8R-A	221	7,62	14	8,42	16	9,26	16	7,78	16	7,20	16
1A8V-A	115	7,28	14	8,15	14	8,82	15	7,28	13	6,72	14
1A9C-A	221	7,68	14	8,43	16	9,24	16	7,61	16	7,13	16

(continua na próxima página)

PDB ID	N	$C_\alpha(7, 0\text{\AA})$		$C_\alpha(7, 5\text{\AA})$		$C_\alpha(8, 0\text{\AA})$		$DP(4, 0\text{\AA})$		$CR(4\%)$	
		$\langle k \rangle$	$k_{\max}$	$\langle k \rangle$	$k_{\max}$	$\langle k \rangle$	$k_{\max}$	$\langle k \rangle$	$k_{\max}$	$\langle k \rangle$	$k_{\max}$
1A9D-A	273	7,76	12	8,58	15	9,41	16	7,61	14	6,90	15
1ABR-A	251	7,97	15	8,74	15	9,77	15	7,89	15	7,47	14
1ABW-A	283	7,78	13	8,44	14	9,22	16	8,40	13	8,06	13
1ABY-A	283	7,80	13	8,33	14	9,07	16	8,23	14	7,76	13
1ACM-B	146	7,44	12	8,36	15	9,09	17	7,02	13	6,29	11
1ACY-L	211	7,90	13	8,69	15	9,59	16	7,05	14	6,38	14
1AD0-B	211	7,34	12	8,34	14	9,30	16	6,92	14	6,30	14
1AD9-B	212	7,35	12	8,28	13	9,11	16	6,69	15	6,04	15
1ADM-A	415	7,59	14	8,57	16	9,62	17	7,71	16	7,11	16
1ADQ-A	206	7,39	12	8,40	14	9,17	15	6,93	16	6,17	14
1AE6-H	211	7,32	12	8,22	13	9,08	15	6,46	14	5,88	13
1AF7-A	274	7,32	12	8,05	14	8,99	17	7,98	14	7,31	13
1AF9-A	441	8,00	14	8,94	15	9,90	16	7,65	14	6,88	13
1AGX-A	331	8,18	15	9,33	18	10,37	20	7,99	14	7,26	13
1AHA-A	246	8,05	12	9,00	14	10,00	17	8,21	15	7,66	13
1AHB-A	246	8,04	13	8,90	14	10,04	16	8,20	15	7,57	14
1AHC-A	246	8,02	13	8,89	14	10,00	16	8,22	15	7,50	13
1AHJ-B	212	7,08	14	7,80	15	8,49	16	7,19	13	6,68	14
1AHW-B	214	7,75	13	8,72	13	9,62	16	6,81	16	6,32	17
1AI1-L	211	7,75	12	8,63	14	9,61	16	6,87	14	6,22	12
1AI6-A	195	7,42	13	8,04	14	8,71	17	7,48	14	7,02	14
1AIF-A	215	7,52	12	8,58	15	9,34	15	6,50	13	5,90	12
1AIG-M	299	7,96	13	8,51	14	9,35	15	8,03	15	7,64	13
1AIJ-M	301	7,90	13	8,52	14	9,19	15	8,05	15	7,55	14
1AMM-A	174	8,17	14	9,26	15	10,48	18	8,12	17	7,37	15
1AMY-A	403	8,22	14	9,26	15	10,38	18	8,43	15	7,72	15
1AN7-A	136	7,52	12	8,27	15	9,07	16	7,31	12	6,60	11
1AOF-A	532	8,46	17	9,58	18	10,7	19	8,11	16	7,31	15
1APG-A	267	7,67	13	8,59	14	9,65	15	7,86	15	7,11	14
1AQD-A	179	7,29	13	8,14	14	8,78	15	7,06	13	6,59	12
1EZM-A	298	8,54	15	9,35	15	10,48	18	8,50	16	8,17	15
15C8-H	211	7,42	12	8,32	14	9,02	14	6,62	16	6,06	15

Tabela A.1: Grau médio e máximo das 56 cadeias de proteínas utilizadas nesta tese, sendo $\langle k \rangle$ o grau médio e $k_{\max}$ o grau máximo identificado para cada rede criada segundo os critérios de transcrição da proteína de acordo com as distâncias entre C_α utilizando raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, de acordo com o critério do Domain Parser com $R_c = 4,0\text{\AA}$ e critério de contato com $I_c \leq 4\%$.

A.2 Assortatividade

PDB ID	Assortatividade				
	$C_{\alpha}(7, 0\text{\AA})$	$C_{\alpha}(7, 5\text{\AA})$	$C_{\alpha}(8, 0\text{\AA})$	$DP(4, 0\text{\AA})$	$CR(4\%)$
1A0H-A	0,447	0,531	0,542	0,319	0,370
1A0P-A	0,345	0,291	0,280	0,284	0,286
1A16-A	0,322	0,360	0,398	0,242	0,249
1A20-A	0,372	0,378	0,382	0,246	0,272
1A3L-H	0,339	0,327	0,293	0,166	0,152
1A41-A	0,197	0,216	0,228	0,302	0,346
1A45-A	0,274	0,257	0,223	0,096	0,168
1A4J-B	0,285	0,283	0,258	0,157	0,103
1A4K-A	0,313	0,300	0,252	0,124	0,137
1A5D-A	0,279	0,234	0,221	0,115	0,134
1A6A-A	0,310	0,299	0,293	0,223	0,215
1A6C-A	0,340	0,357	0,368	0,243	0,240
1A6Q-A	0,348	0,394	0,417	0,197	0,243
1A6T-C	0,318	0,290	0,246	0,145	0,150
1A87-A	0,325	0,307	0,309	0,281	0,325
1A8D-A	0,405	0,431	0,402	0,214	0,164
1A8P-A	0,303	0,329	0,330	0,200	0,216
1A8R-A	0,430	0,434	0,407	0,324	0,313
1A8V-A	0,311	0,301	0,304	0,204	0,240
1A9C-A	0,440	0,435	0,400	0,306	0,280
1A9D-A	0,333	0,342	0,312	0,251	0,257
1ABR-A	0,277	0,287	0,250	0,270	0,315
1ABW-A	0,340	0,356	0,301	0,204	0,270
1ABY-A	0,351	0,334	0,295	0,213	0,235
1ACM-B	0,363	0,311	0,339	0,170	0,207
1ACY-L	0,305	0,276	0,271	0,175	0,140
1AD0-B	0,374	0,372	0,349	0,213	0,165
1AD9-B	0,361	0,382	0,332	0,235	0,164
1ADM-A	0,377	0,378	0,347	0,240	0,276
1ADQ-A	0,334	0,299	0,273	0,192	0,188
1AE6-H	0,400	0,343	0,284	0,227	0,231
1AF7-A	0,419	0,427	0,432	0,250	0,241
1AF9-A	0,395	0,423	0,397	0,226	0,208
1AGX-A	0,418	0,436	0,439	0,344	0,330

(continua na próxima página)

PDB ID	Assortatividade				
	$C_\alpha(7, 0\text{\AA})$	$C_\alpha(7, 5\text{\AA})$	$C_\alpha(8, 0\text{\AA})$	$DP(4, 0\text{\AA})$	$CR(4\%)$
1AHA-A	0,242	0,222	0,225	0,280	0,262
1AHB-A	0,240	0,238	0,232	0,334	0,257
1AHC-A	0,249	0,237	0,228	0,293	0,264
1AHJ-B	0,422	0,430	0,451	0,365	0,322
1AHJ-F	0,417	0,439	0,462	0,359	0,334
1AHJ-H	0,418	0,434	0,454	0,358	0,344
1AHW-B	0,296	0,318	0,274	0,164	0,150
1AHW-E	0,277	0,319	0,273	0,186	0,153
1AI1-L	0,327	0,305	0,247	0,137	0,120
1AI6-A	0,536	0,528	0,509	0,426	0,425
1AIF-A	0,364	0,334	0,308	0,143	0,194
1AIG-M	0,341	0,347	0,326	0,319	0,331
1AIJ-M	0,349	0,341	0,358	0,333	0,340
1AMM-A	0,277	0,260	0,201	0,166	0,175
1AMY-A	0,287	0,309	0,330	0,251	0,297
1AN7-A	0,290	0,337	0,322	0,283	0,289
1AOF-A	0,345	0,443	0,447	0,194	0,216
1APG-A	0,329	0,346	0,358	0,352	0,361
1AQD-A	0,296	0,314	0,297	0,262	0,253
1EZM-A	0,285	0,305	0,305	0,335	0,305
15C8-H	0,352	0,354	0,323	0,200	0,174

Tabela A.2: Valores de assortatividade para as 56 cadeias de proteínas utilizadas nesta tese, para cada rede criada a partir dos critérios de distâncias entre C_α , com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, do Domain Parser com $R_c = 4,0\text{\AA}$ e critério de contato entre resíduos com $I_c \leq 4\%$.

PDB ID	N	r	PDB ID	N	r	PDB ID	N	r
1HHN-A	101	0,2400	1JOG-C	129	0,100	1YHU-J	144	0,240
1C6N-A	162	0,4200	1JPX-D	57	-0,038	2BV8-D	171	0,130
250L-A	162	0,2500	2PUH-A	283	0,390	2FHG-D	220	0,260
1BWH-A	129	0,3900	1Z99-A	42	0,480	2IWQ-A	92	0,300
2GC4-K	105	0,2900	4CYT-R	103	0,290	1WDX-C	63	0,340
1PDU-A	230	0,2800	2FIF-C	73	0,250	1T7J-A	99	0,480
1L4A-A	66	0,3900	1RVV-B	154	0,300	1UIX-A	68	-0,031
3EZN-A	236	0,3100	1XHO-A	106	0,110	1KDU-A	80	0,510

(continua na próxima página)

PDB ID	N	r	PDB ID	N	r	PDB ID	N	r
1ZXB-A	289	0,4400	2YQW-D	118	0,270	1N4S-E	314	0,350
1N5K-B	204	0,2400	1IRN-A	53	0,210	1LTG-F	103	0,310
1L3C-A	178	0,1500	1SZ9-A	143	0,240	1T00-B	142	0,100
104T-A	115	0,2300	2PAV-P	139	0,420	3BJQ-G	289	0,240
2AXY-A	69	0,2100	1JNU-B	104	0,230	2QDK-E	248	0,340
1G60-B	228	0,2900	1VPS-A	285	0,380	2ZB0-I	86	0,300
1T60-F	222	0,2900	2Z6Y-B	214	0,300	2D2N-B	142	0,240
3BP9-U	120	0,2100	2CNA-A	237	0,240	2RDR-A	264	0,320
1DYE-A	162	-0,0062	1HBV-A	99	0,220	1CU3-A	162	0,250
2BWB-G	45	-0,0230	1YAJ-B	532	0,430	1RGD-A	71	0,240
1XK9-B	198	0,4400	1TOU-A	131	0,370	1LVW-D	291	0,330
2BWE-F	45	-0,0230	2ZJQ-H	134	0,400	2BJY-A	150	0,240
200P-A	112	0,2100	1TWS-A	273	0,340	2Q8R-F	63	0,240
1P32-C	176	0,3700	2GM5-B	111	0,160	1JS2-D	89	0,210
1ANI-B	446	0,3700	1TX3-B	254	0,300	1ALH-A	446	0,380
1ZLA-A	98	-0,0100	2Z3H-B	126	0,460	109K-G	194	0,240
2A5C-B	123	0,5100	2HYN-C	52	-0,020	2JZ0-D	165	0,320
2QFX-C	410	0,3500	1R8I-A	187	0,270	2TSR-C	281	0,330
1UPH-A	131	0,3300	3CX5-N	385	0,410	1HOT-A	266	0,370
1WE0-E	166	0,2900	1PW0-A	124	0,170	1Y03-A	85	0,420
1X7Z-A	391	0,3200	1NZ0-A	108	0,200	2BWA-B	226	0,240
2JHT-A	138	0,2500	2Z6P-A	160	0,380			

Tabela A.3: Relação da 90 proteínas com um domínio, para as quais foi efetuado o cálculo da assortatividade da rede correspondente.

PDB ID	N	r	PDB ID	N	r	PDB ID	N	r
1NBE-A	310	0,428	3FIO-H	291	0,120	1C72-D	217	0,371
2NU3-E	177	0,364	3CKM-A	309	0,420	2GK1-A	492	0,270
1SOV-B	299	0,368	1EU4-A	204	0,200	2HI9-C	346	0,281
2PYE-D	195	0,352	1F1Z-A	260	0,320	1YL6-B	245	0,422
1KEV-C	351	0,323	1GQ2-J	541	0,300	2HX1-D	275	0,270
1IK9-A	207	0,344	2GLF-B	437	0,320	1T2C-A	298	0,239
1NU8-A	728	0,355	3CIE-B	338	0,360	1LG2-A	359	0,404
2HRC-B	359	0,378	3D46-E	405	0,370	1WBP-A	350	0,323
1UP4-B	396	0,274	2OMW-A	461	0,370	1VNS-A	574	0,356
1U0M-B	348	0,392	1FSU-A	474	0,340	1FL3-H	208	0,264

(continua na próxima página)

PDB ID	N	r	PDB ID	N	r	PDB ID	N	r
3BYU-A	256	0,429	2APL-A	147	-0,021	3BEH-A	222	0,274
2D44-A	482	0,326	1TSS-A	194	0,240	1NWU-A	362	0,430
1T4H-B	193	0,254	2J61-A	216	0,220	1XV8-A	495	0,268
1APX-D	249	0,473	2DUU-Q	322	0,330	2QYE-D	369	0,397
3D47-F	405	0,364	1TR8-B	86	0,230	1TED-B	359	0,384
2NS8-B	204	0,246	1P41-D	270	0,370	1QOR-B	326	0,403
1I48-F	396	0,480	1R71-C	112	0,240	1PY5-A	301	0,403
1H81-A	459	0,304	1GKR-B	450	0,280	3BI2-A	494	0,341
2P16-A	227	0,435	2GUG-D	361	0,260	2AYP-A	268	0,335
1PT0-B	196	0,331	1CG2-D	389	0,440	2ZFP-H	228	0,448
1FJ1-F	251	0,361	1VGS-C	237	0,400	1IXI-A	321	0,377
1L5V-A	796	0,357	3BOX-A	402	0,370	1HL8-B	421	0,288
1Y18-E	213	0,269	1BGY-Q	196	0,340	3D46-B	405	0,370
2OEZ-A	236	0,326	2DPP-B	227	0,350	2P9D-B	265	0,331
3BFJ-D	382	0,286	1RPQ-C	166	0,490	2D3A-C	353	0,395
2BE3-A	197	0,226	207E-A	514	0,380	1IUG-B	347	0,334
2DF7-M	412	0,352	3BL7-A	286	0,370	3CMC-O	332	0,326
2HZV-A	129	0,393	2A3E-B	394	0,300	2HZ4-A	262	0,313
2C5B-C	291	0,467	2D3C-C	353	0,430	2VCV-E	219	0,273
1YWG-O	337	0,362	10C4-A	298	0,330			

Tabela A.4: Relação da 90 proteínas com dois domínios, para as quais foi efetuado o cálculo da assortatividade da rede correspondente.

PDB ID	N	r	PDB ID	N	r	PDB ID	N	r
1NBE-A	310	0,428	3FIO-H	291	0,120	1C72-D	217	0,371
2NU3-E	177	0,364	3CKM-A	309	0,420	2GK1-A	492	0,270
1SOV-B	299	0,368	1EU4-A	204	0,200	2HI9-C	346	0,281
2PYE-D	195	0,352	1F1Z-A	260	0,320	1YL6-B	245	0,422
1KEV-C	351	0,323	1GQ2-J	541	0,300	2HX1-D	275	0,270
1IK9-A	207	0,344	2GLF-B	437	0,320	1T2C-A	298	0,239
1NU8-A	728	0,355	3CIE-B	338	0,360	1LG2-A	359	0,404
2HRC-B	359	0,378	3D46-E	405	0,370	1WBP-A	350	0,323
1UP4-B	396	0,274	20MW-A	461	0,370	1VNS-A	574	0,356
1UOM-B	348	0,392	1FSU-A	474	0,340	1FL3-H	208	0,264
3BYU-A	256	0,429	2APL-A	147	-0,021	3BEH-A	222	0,274
2D44-A	482	0,326	1TSS-A	194	0,240	1NWU-A	362	0,430

(continua na próxima página)

PDB ID	N	r	PDB ID	N	r	PDB ID	N	r
1T4H-B	193	0,254	2J61-A	216	0,220	1XV8-A	495	0,268
1APX-D	249	0,473	2DUU-Q	322	0,330	2QYE-D	369	0,397
3D47-F	405	0,364	1TR8-B	86	0,230	1TED-B	359	0,384
2NS8-B	204	0,246	1P41-D	270	0,370	1QOR-B	326	0,403
1I48-F	396	0,480	1R71-C	112	0,240	1PY5-A	301	0,403
1H81-A	459	0,304	1GKR-B	450	0,280	3BI2-A	494	0,341
2P16-A	227	0,435	2GUG-D	361	0,260	2AYP-A	268	0,335
1PT0-B	196	0,331	1CG2-D	389	0,440	2ZFP-H	228	0,448
1FJ1-F	251	0,361	1VGS-C	237	0,400	1IXI-A	321	0,377
1L5V-A	796	0,357	3BOX-A	402	0,370	1HL8-B	421	0,288
1Y18-E	213	0,269	1BGY-Q	196	0,340	3D46-B	405	0,370
2OEZ-A	236	0,326	2DPP-B	227	0,350	2P9D-B	265	0,331
3BFJ-D	382	0,286	1RPQ-C	166	0,490	2D3A-C	353	0,395
2BE3-A	197	0,226	207E-A	514	0,380	1IUG-B	347	0,334
2DF7-M	412	0,352	3BL7-A	286	0,370	3CMC-O	332	0,326
2HZV-A	129	0,393	2A3E-B	394	0,300	2HZ4-A	262	0,313
2C5B-C	291	0,467	2D3C-C	353	0,430	2VCV-E	219	0,273
1YWG-O	337	0,362	10C4-A	298	0,330			

Tabela A.5: Relação da 90 proteínas com três domínios, para as quais foi efetuado o cálculo da assortatividade da rede correspondente.

PDB ID	N	r	PDB ID	N	r	PDB ID	N	r
2DR9-A	437	0,347	1CSJ-A	519	0,315	1LNQ-G	301	0,342
9WGA-B	170	0,141	1DTZ-A	689	0,294	1M6B-B	584	0,303
2GWS-E	326	0,281	2ZH2-A	437	0,377	1URZ-E	382	0,257
3CJ2-A	557	0,313	1HCI-A	475	0,372	1WZK-A	585	0,377
1QQ0-A	378	0,399	3EPI-A	373	0,290	1WIO-A	363	0,215
1XSK-C	773	0,351	2HG0-A	400	0,271	2BCS-A	324	0,386
1BLF-A	685	0,304	9WGA-A	170	0,158	1QSS-A	539	0,401
3CJB-A	360	0,289	3DC2-B	523	0,376	3DLK-B	409	0,360
1JOK-B	588	0,338	1OJY-C	253	0,102	1N8Z-C	581	0,336
1V0F-C	666	0,318	2HNY-B	401	0,386	3FN9-B	672	0,285
1EA9-D	583	0,336	1URZ-B	387	0,241	2I9G-A	325	0,325
1M3I-D	465	0,316	1M3J-A	471	0,325	2QTS-F	412	0,417
1XSK-A	773	0,339	2NQS-A	403	0,473	2NQQ-D	403	0,345
1D2P-A	373	0,186	1BVZ-A	585	0,365	1URZ-C	385	0,249

(continua na próxima página)

PDB ID	N	r	PDB ID	N	r	PDB ID	N	r
2P3Y-A	447	0,438	1LFI-A	691	0,330	2QTS-E	421	0,430
1LNQ-B	301	0,344	1V0E-A	666	0,284	1KAY-A	378	0,387
1V0E-C	666	0,297	1QZE-A	214	0,273	1MCX-A	304	0,387
3C5G-A	324	0,329	1DKV-A	699	0,370	2ZH5-A	437	0,406
1ORD-A	730	0,345	3CJ5-A	562	0,300	1J0J-B	588	0,345
2NQV-A	405	0,496	2F2H-C	773	0,325	1NGI-A	378	0,378
1R49-A	548	0,326	1OK2-A	254	0,187	1G8K-C	822	0,381
1VFU-B	585	0,374	2C9M-A	994	0,411	1LLW-A	1475	0,389
1LNQ-A	301	0,344	2UWZ-A	170	0,195	7WGA-B	170	0,181
2NYA-F	791	0,343	3CIZ-A	562	0,267	1BF5-A	544	0,405
1MIV-A	385	0,344	1LCF-A	691	0,333	2DPI-A	372	0,251
1DJH-B	559	0,309	2NQQ-C	403	0,374	3C5X-A	390	0,273
2NQU-B	404	0,337	2NQU-A	403	0,336	2FMS-A	326	0,251
1YY9-A	613	0,341	1T6B-X	676	0,286	2QQK-A	541	0,311
1SKR-A	669	0,299	1KTQ-A	534	0,483	1LNQ-C	301	0,335
2UV0-B	170	0,233	1G44-A	223	0,200			

Tabela A.6: Relação da 90 proteínas com quatro domínios, para as quais foi efetuado o cálculo da assortatividade da rede correspondente.

A.3 Modularidade

PDB ID	Modularidade				
	$C_\alpha(7, 0\text{\AA})$	$C_\alpha(7, 5\text{\AA})$	$C_\alpha(8, 0\text{\AA})$	$DP(4, 0\text{\AA})$	$CR(4\%)$
1A0H-A	0,46819	0,46029	0,45015	0,47378	0,47160
1A0P-A	0,43870	0,42882	0,42028	0,43207	0,43680
1A16-A	0,46633	0,45999	0,45334	0,45396	0,45097
1A20-A	0,45747	0,45461	0,45045	0,45012	0,45427
1A3L-H	0,49424	0,49429	0,49049	0,48776	0,49042
1A41-A	0,48163	0,47353	0,45605	0,46065	0,46055
1A45-A	0,48521	0,47946	0,47158	0,46194	0,46835
1A4J-B	0,48829	0,48885	0,48935	0,48688	0,48779
1A4K-A	0,49331	0,48972	0,48675	0,48301	0,44910
1A5D-A	0,48561	0,47728	0,46637	0,45476	0,45997
1A6A-A	0,48713	0,48209	0,47278	0,45945	0,47119
1A6C-A	0,42766	0,42215	0,26773	0,41579	0,41600

(continua na próxima página)

PDB ID	Modularidade				
	$C_{\alpha}(7, 0\text{\AA})$	$C_{\alpha}(7, 5\text{\AA})$	$C_{\alpha}(8, 0\text{\AA})$	$DP(4, 0\text{\AA})$	$CR(4\%)$
1A6Q-A	0,40740	0,39539	0,38921	0,41042	0,41569
1A6T-C	0,49391	0,49018	0,49008	0,48807	0,49215
1A79-A	0,48252	0,48059	0,45578	0,45466	0,45845
1A87-A	0,46647	0,45532	0,44356	0,46721	0,46294
1A8D-A	0,47518	0,47382	0,47018	0,46216	0,46908
1A8P-A	0,43136	0,42911	0,42716	0,40943	0,41827
1A8R-A	0,42916	0,41464	0,40347	0,40769	0,41566
1A8V-A	0,46286	0,45093	0,44593	0,45460	0,45522
1A9C-A	0,43077	0,41571	0,40830	0,41609	0,41943
1A9D-A	0,47599	0,46917	0,46150	0,43899	0,44790
1ABR-A	0,44228	0,43442	0,42515	0,40492	0,41416
1ABW-A	0,49726	0,49583	0,49237	0,49499	0,49390
1ABY-A	0,49729	0,49574	0,49529	0,49569	0,49543
1ACM-B	0,44318	0,43512	0,43200	0,43035	0,43040
1ACY-L	0,48462	0,48100	0,47872	0,48135	0,48364
1AD0-B	0,49153	0,48910	0,48611	0,48535	0,48203
1AD9-B	0,49328	0,49263	0,49244	0,48833	0,48966
1ADM-A	0,48791	0,48541	0,48032	0,47803	0,47991
1ADQ-A	0,48685	0,48242	0,48016	0,47977	0,48062
1AE6-H	0,49191	0,49147	0,49167	0,49330	0,49525
1AF7-A	0,43454	0,42424	0,40567	0,40012	0,40427
1AF9-A	0,47744	0,47590	0,47262	0,46232	0,46704
1AGX-A	0,44763	0,44317	0,43908	0,43499	0,43891
1AHA-A	0,44996	0,43458	0,42173	0,39372	0,40109
1AHB-A	0,45074	0,43792	0,42104	0,39087	0,40539
1AHC-A	0,45129	0,43569	0,42306	0,39405	0,40715
1AHJ-B	0,47872	0,46568	0,45947	0,46209	0,46642
1AHJ-F	0,47628	0,46470	0,45908	0,46328	0,46706
1AHW-B	0,48122	0,47619	0,47511	0,47657	0,47482
1AI1-L	0,48303	0,48052	0,47883	0,47800	0,48184
1AI6-A	0,45044	0,44506	0,42720	0,42404	0,42492
1AIF-A	0,49445	0,49034	0,48965	0,49047	0,49348
1AIG-M	0,46440	0,45877	0,45009	0,45825	0,46067
1AIJ-M	0,46463	0,45918	0,45168	0,45656	0,46224
1AMM-A	0,48574	0,47728	0,46877	0,45512	0,46375

(continua na próxima página)

PDB ID	Modularidade				
	$C_\alpha(7, 0\text{\AA})$	$C_\alpha(7, 5\text{\AA})$	$C_\alpha(8, 0\text{\AA})$	$DP(4, 0\text{\AA})$	$CR(4\%)$
1AMY-A	0,46888	0,46430	0,45931	0,45903	0,46293
1AN7-A	0,47405	0,47209	0,45706	0,42831	0,44451
1AOF-A	0,46030	0,45420	0,45225	0,44348	0,44623
1APG-A	0,44501	0,43266	0,41853	0,40044	0,40865
1AQD-A	0,48622	0,48171	0,47690	0,46692	0,47230
1EZM-A	0,46828	0,46696	0,45950	0,43808	0,45638
15C8-H	0,49351	0,49263	0,49262	0,49047	0,49270

Tabela A.7: Valores das modularidades máximas para as 56 cadeias de proteínas utilizadas nesta tese, para cada rede criada a partir dos critérios de distâncias entre C_α , com raios de corte $R_c = 7,0, 7,5$ e $8,0\text{\AA}$, de acordo com o critério do Domain Parser com $R_c = 4,0\text{\AA}$ e critério de contato entre resíduos com $I_c \leq 4\%$.

A.4 Domínios identificados

A.4.1 $C_\alpha : R_c = 7,0\text{\AA}$

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1A0H-A	(164-248)(249-320)	(164-252)(253-320)	0,0160
1A0P-A	(3-123)(124-273)	(3-100)(111-292)	0,2300
1A16-A	(1-185)(186-439)	(1-173)(174-440)	0,0690
1A20-A	(1-142)(143-347)	(1-133)(134-347)	0,0680
1A3L-H	(1-113)(114-212)	(1-113)(114-227)	0,0000
1A41-A	(81-194)(195-301)	(81-216)(217-310)	0,1000
1A45-A	(1-83)(84-173)	(1-83)(84-174)	0,0000
1A4J-B	(1-119)(120-217)	(1-119)(120-217)	0,0000
1A4K-A	(1-113)(114-216)	(1-107)(108-211)	0,0560
1A5D-A	(1-82)(83-173)	(1-83)(84-174)	0,0120
1A6A-A	(5-83)(84-180)	(5-81)(82-180)	0,0250
1A6C-A	(1-178)(179-513)	(1-348)(349-513)	0,4900
1A6Q-A	(2-225)(226-364)	(2-295)(296-368)	0,2400
1A6T-C	(1-106)(107-210)	(1-108)(109-211)	0,0190
1A79-A	(80-156)(157-250)	(80-179)(9-79)	0,1300
1A87-A	(91-232)(233-387)	(91-187)(188-387)	0,2400
1A8D-A	(1-232)(233-452)	(1-246)(247-452)	0,0570

(continua na próxima página)

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1A8P-A	(2-102)(103-258)	(2-96)(97-258)	0,0620
1A8R-A	(1-87)(88-221)	(1-85)(86-221)	0,0240
1A8V-A	(-1-51)(52-113)	(-1-47)(48-118)	0,0000
1A9C-A	(1-86)(87-221)	(1-85)(86-221)	0,0120
1A9D-A	(2-141)(142-274)	(2-181)(182-274)	0,2200
1ABR-A	(1-121)(122-251)	(1-166)(167-251)	0,2700
1ABW-A	(1-142)(143-283)	(1-140)(141-283)	0,0140
1ABY-A	(1-142)(143-283)	(1-140)(141-283)	0,0140
1ACM-B	(8-99)(100-153)	(8-100)(101-153)	0,0100
1ACY-L	(1-108)(109-211)	(1-108)(109-211)	0,0000
1ADO-B	(1-114)(115-211)	(1-113)(114-211)	0,0088
1AD9-B	(1-113)(114-212)	(1-113)(114-212)	0,0000
1ADM-A	(1-237)(238-415)	(1-243)(244-421)	0,0250
1ADQ-A	(238-341)(342-443)	(238-337)(338-443)	0,0120
1AE6-H	(1-114)(115-211)	(1-114)(115-211)	0,0000
1AF7-A	(11-152)(153-284)	(11-90)(91-284)	0,6900
1AF9-A	(875-1108)(1109-1315)	(875-1109)(1110-1315)	0,0009
1AGX-A	(1-149)(150-331)	(1-216)(217-331)	0,3100
1AHA-A	(1-119)(120-246)	(1-162)(163-246)	0,2700
1AHB-A	(1-120)(121-246)	(1-162)(163-246)	0,2600
1AHC-A	(1-120)(121-246)	(1-162)(163-246)	0,2600
1AHJ-B	(1-110)(111-212)	(1-110)(111-212)	0,0000
1AHJ-F	(1-110)(111-212)	(1-110)(111-212)	0,0000
1AHW-B	(1-116)(117-214)	(1-117)(118-214)	0,0085
1AI1-L	(1-108)(109-211)	(1-108)(109-211)	0,0000
1AI6-A	(3-100)(101-197)	(3-149)(150-197)	0,3300
1AIF-A	(1-108)(109-215)	(1-109)(110-215)	0,0092
1AIG-M	(3-178)(179-301)	(3-143)(144-301)	0,2400
1AIJ-M	(1-178)(179-301)	(1-143)(144-301)	0,2400
1AMM-A	(1-83)(84-174)	(1-83)(84-174)	0,0000
1AMY-A	(1-205)(206-403)	(1-345)(346-403)	0,4100
1AN7-A	(3-64)(65-138)	(3-80)(81-138)	0,2000
1AOF-A	(36-300)(301-567)	(36-135)(136-567)	1,2000
1APG-A	(1-121)(122-267)	(1-179)(180-267)	0,3200
1AQD-A	(3-83)(84-181)	(3-81)(82-181)	0,0250

(continua na próxima página)

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1EZM-A	(1-133)(134-298)	(1-152)(153-298)	0,1200
15C8-H	(1-113)(114-211)	(1-113)(114-226)	0,0000

Tabela A.8: Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério $C_\alpha : R_c = 7,0\text{\AA}$; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).

A.4.2 $C_\alpha : R_c = 7,5\text{\AA}$

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1A0H-A	(164-248)(249-320)	(164-252)(253-320)	0,0160
1A0P-A	(3-122)(123-273)	(3-100)(111-292)	0,2200
1A16-A	(1-185)(186-439)	(1-173)(174-440)	0,0690
1A20-A	(1-142)(143-347)	(1-133)(134-347)	0,0680
1A3L-H	(1-113)(114-212)	(1-113)(114-227)	0,0000
1A41-A	(81-194)(195-301)	(81-216)(217-310)	0,1000
1A45-A	(1-83)(84-173)	(1-83)(84-174)	0,0000
1A4J-B	(1-119)(120-217)	(1-119)(120-217)	0,0000
1A4K-A	(1-112)(113-216)	(1-107)(108-211)	0,0470
1A5D-A	(1-82)(83-173)	(1-83)(84-174)	0,0120
1A6A-A	(5-83)(84-180)	(5-81)(82-180)	0,0250
1A6C-A	(1-186)(187-513)	(1-348)(349-513)	0,4700
1A6Q-A	(2-225)(226-364)	(2-295)(296-368)	0,2400
1A6T-C	(1-106)(107-210)	(1-108)(109-211)	0,0190
1A79-A	(80-156)(157-250)	(80-179)(9-79)	0,1300
1A87-A	(91-232)(233-387)	(91-187)(188-387)	0,2400
1A8D-A	(1-245)(246-452)	(1-246)(247-452)	0,0041
1A8P-A	(2-102)(103-258)	(2-96)(97-258)	0,0620
1A8R-A	(1-86)(87-221)	(1-85)(86-221)	0,0120
1A8V-A	(-1-51)(52-113)	(-1-47)(48-118)	0,0000
1A9C-A	(1-86)(87-221)	(1-85)(86-221)	0,0120
1A9D-A	(2-141)(142-274)	(2-181)(182-274)	0,2200
1ABR-A	(1-111)(112-251)	(1-166)(167-251)	0,3300
1ABW-A	(1-142)(143-283)	(1-140)(141-283)	0,0140

(continua na próxima página)

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1ABY-A	(1-140)(141-283)	(1-140)(141-283)	0,0000
1ACM-B	(8-99)(100-153)	(8-100)(101-153)	0,0100
1ACY-L	(1-107)(108-211)	(1-108)(109-211)	0,0093
1ADO-B	(1-113)(114-211)	(1-113)(114-211)	0,0000
1AD9-B	(1-113)(114-212)	(1-113)(114-212)	0,0000
1ADM-A	(1-237)(238-415)	(1-243)(244-421)	0,0250
1ADQ-A	(238-340)(341-443)	(238-337)(338-443)	0,0089
1AE6-H	(1-114)(115-211)	(1-114)(115-211)	0,0000
1AF7-A	(11-152)(153-284)	(11-90)(91-284)	0,6900
1AF9-A	(875-1108)(1109-1315)	(875-1109)(1110-1315)	0,0009
1AGX-A	(1-180)(181-331)	(1-216)(217-331)	0,1700
1AHA-A	(1-109)(110-246)	(1-162)(163-246)	0,3300
1AHB-A	(1-114)(115-246)	(1-162)(163-246)	0,3000
1AHC-A	(1-109)(110-246)	(1-162)(163-246)	0,3300
1AHJ-B	(1-109)(110-212)	(1-110)(111-212)	0,0091
1AHJ-F	(1-109)(110-212)	(1-110)(111-212)	0,0091
1AHW-B	(1-117)(118-214)	(1-117)(118-214)	0,0000
1AI1-L	(1-107)(108-211)	(1-108)(109-211)	0,0093
1AI6-A	(3-100)(101-197)	(3-149)(150-197)	0,3300
1AIF-A	(1-108)(109-215)	(1-109)(110-215)	0,0092
1AIG-M	(3-151)(152-301)	(3-143)(144-301)	0,0560
1AIJ-M	(1-151)(152-301)	(1-143)(144-301)	0,0560
1AMM-A	(1-82)(83-174)	(1-83)(84-174)	0,0120
1AMY-A	(1-204)(205-403)	(1-345)(346-403)	0,4100
1AN7-A	(3-79)(80-138)	(3-80)(81-138)	0,0120
1AOF-A	(36-300)(301-567)	(36-135)(136-567)	1,2000
1APG-A	(1-121)(122-267)	(1-179)(180-267)	0,3200
1AQD-A	(3-83)(84-181)	(3-81)(82-181)	0,0250
1EZM-A	(1-133)(134-298)	(1-152)(153-298)	0,1200
15C8-H	(1-113)(114-211)	(1-113)(114-226)	0,0000

Tabela A.9: Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério $C_\alpha : R_c = 7,5\text{\AA}$; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).

A.4.3 $C_\alpha : R_c = 8,0\text{\AA}$

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1A0H-A	(164-251)(252-320)	(164-252)(253-320)	0,0040
1A0P-A	(3-107)(108-273)	(3-100)(111-292)	0,0700
1A16-A	(1-182)(183-439)	(1-173)(174-440)	0,0520
1A20-A	(1-142)(143-347)	(1-133)(134-347)	0,0680
1A3L-H	(1-113)(114-212)	(1-113)(114-227)	0,0000
1A41-A	(81-199)(200-301)	(81-216)(217-310)	0,0790
1A45-A	(1-82)(83-173)	(1-83)(84-174)	0,0120
1A4J-B	(1-119)(120-217)	(1-119)(120-217)	0,0000
1A4K-A	(1-112)(113-216)	(1-107)(108-211)	0,0470
1A5D-A	(1-82)(83-173)	(1-83)(84-174)	0,0120
1A6A-A	(5-81)(82-180)	(5-81)(82-180)	0,0000
1A6Q-A	(2-226)(227-364)	(2-295)(296-368)	0,2300
1A6T-C	(1-106)(107-210)	(1-108)(109-211)	0,0190
1A87-A	(91-232)(233-387)	(91-187)(188-387)	0,2400
1A8D-A	(1-247)(248-452)	(1-246)(247-452)	0,0041
1A8P-A	(2-100)(101-258)	(2-96)(97-258)	0,0420
1A8R-A	(1-83)(84-221)	(1-85)(86-221)	0,0240
1A8V-A	(1-50)(51-113)	(1-47)(48-118)	0,0640
1A9C-A	(1-83)(84-221)	(1-85)(86-221)	0,0240
1A9D-A	(2-144)(145-274)	(2-181)(182-274)	0,2000
1ABR-A	(1-111)(112-251)	(1-166)(167-251)	0,3300
1ABW-A	(1-142)(143-283)	(1-140)(141-283)	0,0140
1ABY-A	(1-140)(141-283)	(1-140)(141-283)	0,0000
1ACM-B	(8-99)(100-153)	(8-100)(101-153)	0,0100
1ACY-L	(1-107)(108-211)	(1-108)(109-211)	0,0093
1AD0-B	(1-113)(114-211)	(1-113)(114-211)	0,0000
1AD9-B	(1-113)(114-212)	(1-113)(114-212)	0,0000
1ADM-A	(1-237)(238-415)	(1-243)(244-421)	0,0250
1ADQ-A	(238-338)(339-443)	(238-337)(338-443)	0,0030
1AE6-H	(1-114)(115-211)	(1-114)(115-211)	0,0000
1AF7-A	(11-165)(166-284)	(11-90)(91-284)	0,8300
1AF9-A	(875-1110)(1111-1315)	(875-1109)(1110-1315)	0,0009
1AGX-A	(1-195)(196-331)	(1-216)(217-331)	0,0970
1AHA-A	(1-114)(115-246)	(1-162)(163-246)	0,3000

(continua na próxima página)

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1AHB-A	(1-109)(110-246)	(1-162)(163-246)	0,3300
1AHC-A	(1-109)(110-246)	(1-162)(163-246)	0,3300
1AHJ-B	(1-108)(109-212)	(1-110)(111-212)	0,0180
1AHJ-F	(1-108)(109-212)	(1-110)(111-212)	0,0180
1AHW-B	(1-117)(118-214)	(1-117)(118-214)	0,0000
1AI1-L	(1-107)(108-211)	(1-108)(109-211)	0,0093
1AI6-A	(3-100)(101-197)	(3-149)(150-197)	0,3300
1AIF-A	(1-108)(109-215)	(1-109)(110-215)	0,0092
1AIG-M	(3-151)(152-301)	(3-143)(144-301)	0,0560
1AIJ-M	(1-151)(152-301)	(1-143)(144-301)	0,0560
1AMM-A	(1-82)(83-174)	(1-83)(84-174)	0,0120
1AMY-A	(1-204)(205-403)	(1-345)(346-403)	0,4100
1AN7-A	(3-80)(81-138)	(3-80)(81-138)	0,0000
1AOF-A	(36-344)(345-567)	(36-135)(136-567)	1,5000
1APG-A	(1-121)(122-267)	(1-179)(180-267)	0,3200
1AQD-A	(3-81)(82-181)	(3-81)(82-181)	0,0000
1EZM-A	(1-133)(134-298)	(1-152)(153-298)	0,1200
15C8-H	(1-113)(114-211)	(1-113)(114-226)	0,0000

Tabela A.10: Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério $C_\alpha : R_c = 8,0\text{\AA}$; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).

A.4.4 DP

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1A02-N	(399-569)(570-678)	(399-569)(570-678)	0,0000
1A0H-A	(164-248)(249-320)	(164-252)(253-320)	0,0160
1A0P-A	(3-108)(109-273)	(3-100)(111-292)	0,0800
1A16-A	(1-184)(185-439)	(1-173)(174-440)	0,0640
1A20-A	(1-141)(142-347)	(1-133)(134-347)	0,0600
1A3L-H	(1-112)(113-212)	(1-113)(114-227)	0,0088
1A41-A	(81-208)(209-301)	(81-216)(217-310)	0,0370
1A45-A	(1-81)(82-173)	(1-83)(84-174)	0,0240

(continua na próxima página)

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1A4J-B	(1-118)(119-217)	(1-119)(120-217)	0,0084
1A4K-A	(1-112)(113-216)	(1-107)(108-211)	0,0470
1A5D-A	(1-83)(84-173)	(1-83)(84-174)	0,0000
1A6A-A	(5-82)(83-180)	(5-81)(82-180)	0,0120
1A6C-A	(1-176)(177-513)	(1-348)(349-513)	0,4900
1A6Q-A	(2-226)(227-364)	(2-295)(296-368)	0,2300
1A6T-C	(1-106)(107-210)	(1-108)(109-211)	0,0190
1A79-A	(80-160)(161-250)	(80-179)(9-79)	0,1100
1A87-A	(91-237)(238-387)	(91-187)(188-387)	0,2700
1A8D-A	(1-245)(246-452)	(1-246)(247-452)	0,0041
1A8P-A	(2-96)(97-258)	(2-96)(97-258)	0,0000
1A8R-A	(1-86)(87-221)	(1-85)(86-221)	0,0120
1A8V-A	(-1-50)(51-113)	(-1-47)(48-118)	0,0000
1A9C-A	(1-86)(87-221)	(1-85)(86-221)	0,0120
1A9D-A	(2-150)(151-274)	(2-181)(182-274)	0,1700
1ABR-A	(1-110)(111-251)	(1-166)(167-251)	0,3400
1ABW-A	(1-142)(143-283)	(1-140)(141-283)	0,0140
1ABY-A	(1-140)(141-283)	(1-140)(141-283)	0,0000
1ACM-B	(8-98)(99-153)	(8-100)(101-153)	0,0200
1ACY-L	(1-107)(108-211)	(1-108)(109-211)	0,0093
1AD0-B	(1-114)(115-211)	(1-113)(114-211)	0,0088
1AD9-B	(1-112)(113-212)	(1-113)(114-212)	0,0088
1ADM-A	(1-237)(238-415)	(1-243)(244-421)	0,0250
1ADQ-A	(238-338)(339-443)	(238-337)(338-443)	0,0030
1AE6-H	(1-112)(113-211)	(1-114)(115-211)	0,0180
1AF7-A	(11-115)(116-284)	(11-90)(91-284)	0,2800
1AF9-A	(875-1109)(1110-1315)	(875-1109)(1110-1315)	0,0000
1AGX-A	(1-193)(194-331)	(1-216)(217-331)	0,1100
1AHA-A	(1-120)(121-246)	(1-162)(163-246)	0,2600
1AHB-A	(1-119)(120-246)	(1-162)(163-246)	0,2700
1AHC-A	(1-120)(121-246)	(1-162)(163-246)	0,2600
1AHJ-B	(1-110)(111-212)	(1-110)(111-212)	0,0000
1AHJ-F	(1-110)(111-212)	(1-110)(111-212)	0,0000
1AHW-B	(1-116)(117-214)	(1-117)(118-214)	0,0085
1AI1-L	(1-107)(108-211)	(1-108)(109-211)	0,0093

(continua na próxima página)

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1AI6-A	(3-99)(100-197)	(3-149)(150-197)	0,3400
1AIF-A	(1-108)(109-215)	(1-109)(110-215)	0,0092
1AIG-M	(3-177)(178-301)	(3-143)(144-301)	0,2400
1AIJ-M	(1-177)(178-301)	(1-143)(144-301)	0,2400
1AMM-A	(1-83)(84-174)	(1-83)(84-174)	0,0000
1AMY-A	(1-200)(201-403)	(1-345)(346-403)	0,4200
1AN7-A	(3-81)(82-138)	(3-80)(81-138)	0,0120
1AOF-A	(36-300)(301-567)	(36-135)(136-567)	1,2000
1APG-A	(1-131)(132-267)	(1-179)(180-267)	0,2700
1AQD-A	(3-81)(82-181)	(3-81)(82-181)	0,0000
1EZM-A	(1-133)(134-298)	(1-152)(153-298)	0,1200
15C8-H	(1-112)(113-211)	(1-113)(114-226)	0,0088

Tabela A.11: Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério DP; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).

A.4.5 CR

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1A0H-A	(164-248)(249-320)	(164-252)(253-320)	0,0160
1A0P-A	(3-112)(113-273)	(3-100)(111-292)	0,1200
1A16-A	(1-188)(189-439)	(1-173)(174-440)	0,0870
1A20-A	(1-141)(142-347)	(1-133)(134-347)	0,0600
1A3L-H	(1-112)(113-212)	(1-113)(114-227)	0,0088
1A41-A	(81-208)(209-301)	(81-216)(217-310)	0,0370
1A45-A	(1-81)(82-173)	(1-83)(84-174)	0,0240
1A4J-B	(1-118)(119-217)	(1-119)(120-217)	0,0084
1A4K-A	(1-107)(108-216)	(1-107)(108-211)	0,0000
1A5D-A	(1-81)(82-173)	(1-83)(84-174)	0,0240
1A6A-A	(5-81)(82-180)	(5-81)(82-180)	0,0000
1A6C-A	(1-176)(177-513)	(1-348)(349-513)	0,4900
1A6Q-A	(2-226)(227-364)	(2-295)(296-368)	0,2300
1A6T-C	(1-106)(107-210)	(1-108)(109-211)	0,0190

(continua na próxima página)

PDB ID	Domínios		δ_E
	$Q_{\max}$	CATH	
1A79-A	(80-154)(155-250)	(80-179)(9-79)	0,1400
1A87-A	(91-237)(238-387)	(91-187)(188-387)	0,2700
1A8D-A	(1-245)(246-452)	(1-246)(247-452)	0,0041
1A8P-A	(2-97)(98-258)	(2-96)(97-258)	0,0100
1A8R-A	(1-86)(87-221)	(1-85)(86-221)	0,0120
1A8V-A	(-1-49)(50-112)	(-1-47)(48-118)	0,0000
1A9C-A	(1-86)(87-221)	(1-85)(86-221)	0,0120
1A9D-A	(2-150)(151-274)	(2-181)(182-274)	0,1700
1ABR-A	(1-111)(112-251)	(1-166)(167-251)	0,3300
1ABW-A	(1-140)(141-283)	(1-140)(141-283)	0,0000
1ABY-A	(1-140)(141-283)	(1-140)(141-283)	0,0000
1ACM-B	(8-97)(98-153)	(8-100)(101-153)	0,0300
1ACY-L	(1-107)(108-211)	(1-108)(109-211)	0,0093
1ADO-B	(1-114)(115-211)	(1-113)(114-211)	0,0088
1AD9-B	(1-113)(114-212)	(1-113)(114-212)	0,0000
1ADM-A	(1-237)(238-415)	(1-243)(244-421)	0,0250
1ADQ-A	(238-338)(339-443)	(238-337)(338-443)	0,0030
1AE6-H	(1-111)(112-211)	(1-114)(115-211)	0,0260
1AF7-A	(11-115)(116-284)	(11-90)(91-284)	0,2800
1AF9-A	(875-1108)(1109-1315)	(875-1109)(1110-1315)	0,0009
1AGX-A	(1-193)(194-331)	(1-216)(217-331)	0,1100
1AHA-A	(1-120)(121-246)	(1-162)(163-246)	0,2600
1AHB-A	(1-120)(121-246)	(1-162)(163-246)	0,2600
1AHC-A	(1-120)(121-246)	(1-162)(163-246)	0,2600
1AHJ-B	(1-110)(111-212)	(1-110)(111-212)	0,0000
1AHJ-F	(1-110)(111-212)	(1-110)(111-212)	0,0000
1AHW-B	(1-118)(119-214)	(1-117)(118-214)	0,0085
1AI1-L	(1-107)(108-211)	(1-108)(109-211)	0,0093
1AI6-A	(3-100)(101-197)	(3-149)(150-197)	0,3300

Tabela A.12: Relação dos domínios identificados para as 56 cadeias de proteínas utilizadas nesta tese, por meio da maximização da modularidade ($Q_{\max}$) da rede criada segundo o critério CR; divisões adotadas como padrão ouro (CATH) e erro relativo entre $Q_{\max}$ e CATH, em porcentagem do erro de identificação (δ_E).

Apêndice B

The GNU General Public License

Copyright © 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program—to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- (c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- (d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form

a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- (a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- (b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- (c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- (d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- (e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section

must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- (a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- (b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- (c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- (d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- (e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- (f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a

version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

Referências Bibliográficas

- [1] T. WEISE, *Global Optimization Algorithms – Theory and Application*, 2008.
- [2] N. A. ALVES, *Physical Review E* **76** (2007).
- [3] G. AGARWAL and D. KEMPE, *The European Physical Journal B* **66**, 409 (2008).
- [4] X. LIU and T. MURATA, *Physica A* **389**, 1493 (2010).
- [5] M. GIRVAN and M. E. J. NEWMAN, *Proc Natl Acad Sci U S A* **99**, 7821 (2002).
- [6] R. GUIMERÀ and L. A. NUNES AMARAL, *Nature* **433**, 895 (2005).
- [7] S. BOETTCHER and A. PERCUS, *Physical Review E* **64**, 1 (2001).
- [8] J. ALPERT and N. HAJAJ, *We knew the web was big...*, Official Google Blog, 2008.
- [9] G. L. ALEXANDERSON, *Bulletin of the American Mathematical Society* **43**, 567 (2006).
- [10] S. FORTUNATO, *Physics Reports* **486**, 75 (2010).
- [11] D. J. WATTS and S. H. STROGATZ, *Nature* **393**, 440 (1998).
- [12] U. BRANDES, *The Journal of Mathematical Sociology* **25**, 163 (2001).
- [13] M. E. J. NEWMAN, *SIAM Review* **45**, 167 (2003).
- [14] A. BARABÁSI, *Science* **286**, 509 (1999).
- [15] S. N. DOROGVTSEV, A. L. BARABASI, and J. MENDES, *Evolution of Networks: From Biological Nets to the Internet and WWW*, Oxford University Press, 2004.
- [16] S. N. DOROGVTSEV and J. F. F. MENDES, *Advances in Physics* **51**, 1079 (2002).
- [17] R. ALBERT and A. BARABASI, *Reviews of modern physics* **74**, 47 (2002).
- [18] M. NEWMAN and J. PARK, *Physical Review E* **68** (2003).
- [19] M. NEWMAN, *Physical Review Letters* **89** (2002).
- [20] M. E. J. NEWMAN, *Physical Review E* **67**, 26126 (2003).
- [21] S.-J. YANG, *Physical Review E* **71**, 1 (2005).

- [22] J. NOH and H. RIEGER, *Physical Review Letters* **92**, 1 (2004).
- [23] P. BRÉEMAUD, *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*, Springer-Verlag New York, Inc., 1998.
- [24] A. ERDÖS, PAUL, RÉNYI, *Publ. Math. Debrecen* **6** (1959).
- [25] J. TRAVERS and S. MILGRAM, *Sociometry* **32**, 425 (1969).
- [26] A. MISLOVE, M. MARCON, K. P. GUMMADI, P. DRUSCHEL, and B. BHATTACHARJEE, *Measurement and analysis of online social networks*, ACM Press, New York, New York, USA, 2007.
- [27] M. NEWMAN, *Contemporary Physics* **46**, 323 (2005).
- [28] A. CLAUSET, C. R. SHALIZI, and M. E. J. NEWMAN, *SIAM Review* **51**, 661 (2009).
- [29] M. NEWMAN and M. GIRVAN, *Physical review E* **69**, 26113 (2004).
- [30] T. N. BUI and B. R. MOON, *IEEE Transactions on Computers* **45**, 841 (1996).
- [31] G. GODZA, M. REBAUDENGO, M. S. REORDA, and U. P. BUCURESTI, *Evaluation*, 985.
- [32] M. TASGIN, A. HERDAGDELEN, and H. BINGOL, *arXiv:0711.0491v1* (2007).
- [33] A. CLAUSET, M. NEWMAN, and C. MOORE, *Physical Review E* **70**, 66111 (2004).
- [34] V. D. BLONDEL, J.-L. GUILLAUME, R. LAMBIOTTE, and E. LEFEBVRE, *Journal of Statistical Mechanics* **2008**, P10008 (2008).
- [35] J. DUCH and A. ARENAS, *Physical Review E* **72**, 27104 (2005).
- [36] B. YAN and S. GREGORY, *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*, 832 (2009).
- [37] C. LEE, F. REID, A. MCDAID, and N. HURLEY, *arXiv:1002.1827v2* (2010).
- [38] B. KARRER and M. E. J. NEWMAN, *ReCALL*, 11 (2010).
- [39] R. GUIMERA, M. SALES-PARDO, and L. AMARAL, *Physical Review E* **70**, 1 (2004).
- [40] J. REICHARDT and S. BORNHOLDT, *Physical Review E* **74**, 016110 (2006).
- [41] J. REICHARDT and S. BORNHOLDT, *Physica D* **224**, 20 (2006).
- [42] S. FORTUNATO and M. BARTHÉLEMY, *Proc Natl Acad Sci U S A* **104** (2007).
- [43] R. GUIMERÀ and L. A. N. AMARAL, *Journal of Statistical Mechanics* **2005** (2005).
- [44] B. ALBERTS, D. BRAY, A. JOHNSON, J. LEWIS, M. RAFF, K. ROBERTS, P. WALTER, and A. CAMPBELL, *Essential cell biology*, Garland Science New York, 2004.

- [45] B. ALBERTS, A. JOHNSON, J. LEWIS, M. RAFF, K. ROBERTS, and P. WALTER, *Molecular biology of the cell*. 2002, New York: Garland Science.
- [46] D. VOET, J. VOET, and C. PRATT, *Fundamentals of biochemistry*, Wiley, New York, NY, 1999.
- [47] A. LEHNINGER, D. NELSON, and M. COX, *Lehninger principles of biochemistry*, Wh Freeman, 2005.
- [48] W. BRAGG, *Nature* **91** (1913).
- [49] J. KENDREW, G. BODO, H. DINTZIS, and R. PARRISH, *Nature* **181** (1958).
- [50] J. KENDREW, *Science* **139**, 1259 (1963).
- [51] J. WATSON and F. CRICK, *Nature* **171** (1953).
- [52] J. KENDREW, *Comparative Biochemistry and Physiology* **4**, 249 (1962).
- [53] G. A. PETSKO and D. RINGE, *Protein structure and function*, New Science Press, 2004.
- [54] M. SAVAGEAU, *Proc Natl Acad Sci U S A* **83**, 1198 (1986).
- [55] J. GAREL, *Protein folding* **1** (1992).
- [56] S. J. WHEELAN, A. MARCHLER-BAUER, and S. H. BRYANT, *Bioinformatics* **16**, 613 (2000).
- [57] S. A. ISLAM, J. LUO, and M. J. STERNBERG, *Protein Engineering, Design and Selection* **8**, 513 (1995).
- [58] D. C. PHILLIPS, *Scientific American* **215**, 78 (1966).
- [59] J. DRENTH, J. N. JANSONIUS, R. KOEKOEK, H. M. SWEN, and B. G. WOLTHERS, *Nature* **218**, 929 (1968).
- [60] R. R. PORTER, *Science* **180**, 713 (1973).
- [61] G. M. EDELMAN, *Scandinavian journal of immunology* **34**, 1 (1991).
- [62] J. S. RICHARDSON, *Advances in protein chemistry* **34**, 167 (1981).
- [63] P. BORK, *Febs Letters* **286** (1991).
- [64] C. CHOTHIA, *Nature* **357**, 543 (1992).
- [65] F. JACOB, *Science* **196**, 1161 (1977).
- [66] I. D. CAMPBELL and A. K. DOWNING, *Trends in biotechnology* **12**, 168 (1994).
- [67] H. BERMAN, T. BHAT, P. BOURNE, Z. FENG, and G. GILLILAND, *Nucleic Acids Research* **28**, 235 (2000).
- [68] [HTTP://WWW.PDB.ORG](http://www.pdb.org), *RCSB The Protein Data Bank*.

- [69] A. F. YAKUNIN, A. A. YEE, A. SAVCHENKO, A. M. EDWARDS, and C. H. ARROWSMITH, *Current opinion in chemical biology* **8**, 42 (2004).
- [70] L. A. MIRNY and E. I. SHAKHNOVICH, *Journal of Molecular Biology* **291**, 177 (1999).
- [71] C. A. ORENGO, A. D. MICHIE, S. JONES, D. T. JONES, M. B. SWINDELLS, and J. M. THORNTON, *Structure* **5**, 1093 (1997).
- [72] C. A. ORENGO and W. R. TAYLOR, *SSAP: Sequential structure alignment program for protein structure comparison*, Academic Press, 1996.
- [73] W. R. TAYLOR and C. A. ORENGO, *Journal of Molecular Biology* **208**, 1 (1989).
- [74] L. HOLM and C. SANDER, *Proteins* **19**, 256 (1994).
- [75] A. S. SIDDIQUI and G. J. BARTON, *Protein science* **4**, 872 (1995).
- [76] A. MURZIN, S. BRENNER, T. HUBBARD, and C. CHOTHIA, *Journal of Molecular Biology* **247**, 536 (1995).
- [77] Y. XU, D. XU, and H. GABOW, *Bioinformatics* **16**, 1091 (2000).
- [78] R. MERRIS, *Linear Algebra and its Applications* **197-198**, 143 (1994).
- [79] M. NEWMAN, *Physical Review E* **74**, 36104 (2006).
- [80] W. XIAO and I. GUTMAN, *Theoretical Chemistry Accounts* **110**, 284 (2003).
- [81] B. F. J. MANLY, *Multivariate Statistical Methods: A Primer*, Chapman and Hall, 3rd edition, 2004.
- [82] W. W. ZACHARY, *Journal of Anthropological Research* **33** (1977).
- [83] R. IHAKA and R. GENTLEMAN, *Journal of Computational and Graphical Statistics* **5** (1996).
- [84] R. D. C. TEAM, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2010.
- [85] Y. COHEN and J. Y. COHEN, *Statistics and data with R*, John Wiley & Sons Ltd., 2008.
- [86] F. MURTAGH, *Compstat Lectures 4: Multidimensional Clustering Algorithms*, Physica-Verlag, Vienna, Austria, 1985.
- [87] J. M. CHAMBERS, *Software for Data Analysis*, volume 15 of *Statistics and Computing*, Springer New York, New York, NY, 2008.
- [88] Z. DU and F. LIN, *Parallel Computing* **31**, 523 (2005).
- [89] M. DASH, H. LIU, P. SCHEUERMANN, and K. TAN, *Data and Knowledge Engineering* **44**, 109 (2003).

- [90] B. WALTER, K. BALA, M. KULKARNI, and K. PINGALI, *IEEE Symposium on Interactive Ray Tracing (RT)* (2008).
- [91] A. DROZDEK, *Data Structures and Algorithms in C++*, Brooks/Cole Publishing Co., Pacific Grove, CA, USA, 2000.
- [92] D. LUSSEAU, K. SCHNEIDER, O. J. BOISSEAU, P. HAASE, E. SLOOTEN, and S. M. DAWSON, *Behavioral Ecology and Sociobiology* **54**, 396 (2003).
- [93] S. VAN SEGBROECK, F. C. SANTOS, and J. M. PACHECO, *PLoS Computational Biology* **6**, e1000895 (2010).
- [94] S. EUBANK, H. GUCLU, V. S. A. KUMAR, M. V. MARATHE, A. SRINIVASAN, Z. TOROCZKAI, and N. WANG, *Nature* **429**, 180 (2004).
- [95] M. SALATHÉ, M. KAZANDJIEVA, J. W. LEE, P. LEVIS, M. W. FELDMAN, and J. H. JONES, *arXiv:1006.4622*, 1 (2010).
- [96] M. CHOUDHURY, M. THOMAS, and A. MUKHERJEE, *TextGraphs-2: Graph-Based Algorithms for Natural Language Processing*, 81 (2007).
- [97] P. TALUKDAR and F. PEREIRA, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 1473 (2010).
- [98] N. LEAVITT, *Computer* **43**, 12 (2010).
- [99] A. LAKSHMAN and P. MALIK, *ACM SIGOPS Operating Systems Review* **44**, 35 (2010).
- [100] J. PUJOL, V. ERRAMILLI, G. SIGANOS, X. YANG, and N., *ACM SIGCOMM*, 375 (2010).
- [101] F. CHANG, J. DEAN, S. GHEMAWAT, W. C. HSIEH, D. A. WALLACH, M. BURROWS, T. CHANDRA, A. FIKES, and R. E. GRUBER, *ACM Transactions on Computer Systems* **26**, 1 (2008).
- [102] M. GALASSI and B. GOUGH, *GNU Scientific Library Reference Manual*, Network Theory Ltd.
- [103] A. B. DEMKO and N. J. PIZZI, *Software: Practice and Experience* **39**, 641 (2009).
- [104] A. ARENAS, J. DUCH, A. FERNÁNDEZ, and S. GÓMEZ, *New Journal of Physics* **9**, 176 (2007).
- [105] M. GALASSI, J. DAVIES, B. GOUGH, and G. JUNGMAN, *GNU Scientific Library*, Number February, Network Theory Ltd, 2003.
- [106] G. GOLUB and C. V. LOAN, *Matrix computations*, Johns Hopkins University Press, 3rd edition, 1996.
- [107] A. SAMEH, *Mathematics of Computation* **25** (1971).
- [108] H. RUTISHAUSER, *Numerische Mathematik* **9**, 1 (1966).

- [109] G. GOLUB, *Journal of Computational and Applied Mathematics* **123**, 35 (2000).
- [110] M. E. J. NEWMAN, *Proc Natl Acad Sci U S A* **103**, 8577 (2006).
- [111] P. M. GLEISER and L. DANON, *Advances in Complex Systems* **6**, 565 (2003).
- [112] D. KNUTH, The Stanford GraphBase: a platform for combinatorial algorithms, in *SODA '93: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms*, number 3, pp. 41–43, Society for Industrial and Applied Mathematics, 1993.
- [113] P. J. BICKEL and A. CHEN, *Proc Natl Acad Sci U S A* **106**, 21068 (2009).
- [114] M. ROSVALL and C. BERGSTROM, *Proc Natl Acad Sci U S A* **104**, 7327 (2007).
- [115] W. GREINER, L. NEISE, and H. STÖCKER, *Thermodynamics and statistical mechanics*, Springer-Verlag, 1995.
- [116] C. R. DARWIN, *On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life.*, Murray, John, 5 edition, 1869.
- [117] A. NOACK and R. ROTTA, *Lecture Notes in Computer Science* **5526**, 257 (2009).
- [118] R. GUIMERÀ, M. SALES-PARDO, and L. AMARAL, *Physical Review E* **70**, 1 (2004).
- [119] Y. SUN, B. DANILA, K. JOSIĆ, and K. E. BASSLER, *Europhysics Letters* **86**, 28004 (2009).
- [120] J. LIU and T. LIU, *Physica A* **389**, 2300 (2010).
- [121] S. KIRKPATRICK, C. D. GELATT, and M. P. VECCHI, *Science* **220**, 671 (1983).
- [122] N. METROPOLIS, A. W. ROSENBLUTH, M. N. ROSENBLUTH, A. H. TELLER, and E. TELLER, *The Journal of Chemical Physics* **21** (1953).
- [123] A. NOLTE and R. SCHRADER, *Mathematics of Operations Research* **25**, 476 (2000).
- [124] L. INGBER, *Mathematical and Computer Modelling* **18**, 29 (1993).
- [125] J. H. HOLLAND, *Adaptation in Natural and Artificial Systems*, MIT Press, 1992.
- [126] D. KNUTH, *The art of computer programming - Volume 2: Seminumerical Algorithms*, Addison-Wesley, 1973.
- [127] S. PARK and K. MILLER, *Communications of the ACM* **31**, 1201 (1988).
- [128] Z. GUTTERMAN, B. PINKAS, and T. REINMAN, *Analysis of the Linux Random Number Generator*, IEEE, 2006.
- [129] D. EASTLAKE 3RD, J. SCHILLER, and S. CROCKER, RFC-4086: Randomness Requirements for Security, 2005.

- [130] S. BOETTCHER and A. G. PERCUS, *Physical Review Letters* **86**, 5211 (2001).
- [131] K. SNEPPEN, P. BAK, H. FLYVBJERG, and M. H. JENSEN, *Proc Natl Acad Sci U S A* **92**, 5209 (1995).
- [132] S. BOETTCHER and M. PACZUSKI, *Physical review letters* **76**, 348 (1996).
- [133] S. J. GOULD and N. ELDREDGE, *Nature* **366**, 223 (1993).
- [134] P. BAK and M. PACZUSKI, *Proc Natl Acad Sci U S A* **92**, 6689 (1995).
- [135] S. BOETTCHER, *Artificial Intelligence* **119**, 275 (2000).
- [136] M. NEWMAN, *Contemporary Physics* **46**, 323 (2005).
- [137] M. MITZENMACHER, *Internet Mathematics* **1**, 226.
- [138] [HTTP://WWW.GNU.ORG/SOFTWARE/BASH](http://www.gnu.org/software/bash/), *The GNU Bash Reference Manual, for Bash*.
- [139] J. BAGROW and E. BOLLT, *Physical Review E* **72**, 46108 (2005).
- [140] D. GOLDBERG, *ACM Computing Surveys* **23**, 5 (1991).
- [141] N. KANNAN and S. VISHVESHWARA, *Journal of Molecular Biology* **292**, 441 (1999).
- [142] R. SISTLA, K. BRINDA, and S. VISHVESHWARA, *Proteins* **59**, 616 (2005).
- [143] K. NISHIKAWA and T. OOI, *Journal of Biochemistry* **100**, 1043 (1986).
- [144] S. MIYAZAWA and R. L. JERNIGAN, *Macromolecules* **18**, 534 (1985).
- [145] L. H. GREENE and V. A. HIGMAN, *Journal of Molecular Biology* **334**, 781 (2003).
- [146] N. A. ALVES and A. S. MARTINEZ, *Physica A* **375**, 336 (2007).
- [147] A. N. JHA, S. VISHVESHWARA, and J. R. BANAVAR, *Protein Science* **19**, 603 (2010).
- [148] A. ATILGAN, P. AKAN, and C. BAYSAL, *Biophysical Journal* **86**, 85 (2004).
- [149] G. BAGLER and S. SINHA, *Physica A* **346**, 27 (2005).
- [150] P. D. B. E. DICTIONARY, Protein Data Bank Contents Guide: Atomic Coordinate Entry Format Description Version 3.20 Document Published by the wwPDB, 2008.
- [151] E. KOUTSOFIOS, S. NORTH, and OTHERS, Drawing graphs with DOT, 1991.
- [152] S. KOTZ, *Encyclopedia of statistical sciences*, John Wiley & Sons Inc., Hoboken, NJ, 2 edition, 2006.
- [153] W. KRUSKAL and W. WALLIS, *Journal of the American Statistical Association* **47** (1952).
- [154] H. PEARSON, *Nature* **441**, 398 (2006).

Glossário

A

Matriz adjacência - 9

C_α

Carbono- α - v, vii, xi, xii, xiv, xv, 4, 24, 31, 35, 36, 126, 127, 129, 132, 134, 136, 139, 147–149, 154–157, 163, 169, 171, 176, 178, 180, 181

E

Conjunto de arestas de um grafo - 9

E

Energia - 74

G

Grafo ou rede complexa - 9

I_c

Critério de corte de interação entre resíduos de uma proteína - xi, 128, 132, 134

$M_{\text{máx}}$

Número máximo de arestas em um grafo - 11

M

Número total de arestas de uma rede - 11

$N(k)$

Número de vértices com grau k em grafo - 11

N_{div}

Número de divições iniciais de uma rede - ix, x, xvi, 85, 86, 88–90, 92–99, 103, 106, 111, 113

N_{exec}

Número de execuções - ix, x, 85, 87–100, 103, 112, 114, 115, 119

N_{rep}

Número de repetições - ix, x, 85, 88, 90, 93–95, 98–100, 106, 116

N

Número total de vértices de uma rede - xi, 10, 11, 82, 85, 86, 152

 $P(k)$

Distribuição do graus de um grafo - 11, 12

 $Q_{\max}$

Valor máximo da função modularidade para uma dada rede - ix, x, xiv, xv, 87, 88, 90–92, 96, 101, 103, 105, 106, 112–114, 117, 119, 121, 122, 146, 148, 149, 161, 162, 178, 180, 181, 183, 185

 Q

Função objetivo modularidade - viii, xi, xiii, 19, 46, 47, 67–70, 74, 79, 87, 88, 90, 93, 94, 98, 99, 103, 112, 117–119, 122, 123, 152, 156

 R_G

Raio de giração de uma proteína - 130

 R_c

Raio de corte utilizado para definir a distância limite de contato entre dois componentes de uma proteína - v, xii, xiv, xv, 132, 134, 147–149, 155–157, 163, 178, 180, 181

 T

Temperatura - 74

 V

Conjunto de vértices de um grafo - 9

 δ_E

Erro relativo - xiv, xv, 147, 165, 178, 180, 181, 183, 185

 λ

Valor de aptidão de um elemento - 78, 79

 τ

Expoente da lei de potências - 79, 86, 92, 97, 102, 103, 107, 111

 a_{ij}

Elemento da matriz adjacência - 9–11

 g

Possível solução no contexto de algoritmos evolucionários ou genótipo - 76

 k_B

Constante de Boltzmann - 74

$k_{\text{máx}}$

Grau máximo de uma rede - 12

 k_i

Grau do vértice i - 10, 11

 $k_{nn}(k)$

- 12

 k

Grau de um vértice em um grafo - 70

 m

Número de arestas de uma rede - 47

 n

Número de vértices - ix, 85–87, 92, 98, 152

 r

Assortatividade - 12, 13

 t

Número de iterações - 75, 82

 w_i

Peso de uma arestas de uma rede - 10

 w_{ij}

Peso da aresta que conecta os vértices i e j em uma rede - 10

crossing-over

processo de troca de genes entre dois cromossomos - 75, 76, 105, 162

backbone

Cadeia polipeptídica de uma proteína e também denominada esqueleto da proteína - v, vii, 25, 131, 133, 134, 148, 152, 163

smart pointer

Tipo abstrato de dados que simula um ponteiro convencional e fornece funcionalidades adicionais como: coletor de lixo e verificação de limites do tipo de dado - 82

AG

Algoritmo Genético - 74, 77, 78, 161, 162

AL

Método de detecção de comunidades em redes complexas proposto por Alves - viii, x, xiii, xiv, xvi, 3, 4, 7, 8, 13, 43, 45–48, 53, 55, 57, 58, 62, 64–70, 74, 102, 103, 105, 106, 111–114, 117–119, 121–124, 146–149, 152, 161–165

ALGO

Método de detecção de comunidades em redes complexas proposto por Alves com Otimização proposta por Guidolin - v, x, xiii, xiv, 3, 4, 106–108, 110–124, 146–149, 152, 162–165

Bash

Interpretador de comandos de sistemas Unix e Unix-like - 85

C++

Linguagem de programação C++ - xvi, 4, 48, 53, 54, 56, 60, 61, 82, 110, 130

CATH

Base de dados de classificação de proteínas e identificação de domínios - iv, viii, xi, xiv, xv, 34–37, 39, 127, 128, 130, 131, 145–148, 153, 178, 180, 181, 183, 185

CR

Critério de interação entre resíduos de uma proteína - 127

DNA

Ácido desoxirribonucleico - 23, 26–28

DOT

Linguagem para descrição de grafos - 4, 130

Domain Parser

Método de identificação de domínios estruturais em proteínas [75] - xi, xii, xiv, 4, 35, 39–41, 125, 126, 129, 132, 134, 136, 139, 148, 153–155, 163, 169, 171, 176

GN

Método de identificação de comunidades proposto por Girvan e Newman [39] - xiii, 20, 67, 68, 103, 112

GNU-GSL

GNU Scientific Library - 53, 64

GPLv3

The GNU General Public License versão 3.0 - 4, 53, 129

HAC++

hierarchical agglomerative clustering em C++ - v, viii, 3, 4, 48, 49, 58, 65–67, 105–107, 111, 163

LPAm+

Método de identificação de comunidades – *Advanced modularity-specialized label propagation algorithm* - xiii, 20, 21, 67, 68, 103, 112, 162

NoSQL

Tecnologia de banco de dados relacionais baseado na teoria de grafos - 52

PDB

The Protein Data Bank - iv, vii, viii, 3, 4, 33, 34, 37, 38, 128–131

PDB ID

Código de identificação de uma proteína no The Protein Data Bank - xi, xiv, 127–129, 131

PDP

Protein domain parser - 35

R

R Language and Environment for Statistical Computing [82] - viii, 48, 66, 67, 163

RCSB

Research Collaboratory for Structural Bioinformatics - vii, 34

RMN

Ressonância magnética nuclear - 33

SA

Recozimento simulado – *Simulated Annealing* - xiii, 20, 67, 68, 74, 75, 78, 103, 112, 161, 162

SCOP

Base de dados de classificação de proteínas e identificação de domínios - iv, viii, 34, 35, 37–39, 147, 148, 153

Scopira

Biblioteca em C++ para análise de dados na área de bioinformática - 53

StatLib

Biblioteca de bases de dados e aplicativos de estatística - 48

VP

Método de identificação de comunidades proposto por Agarwal e Kempe [23] - viii, xiii, 20, 21, 67–70, 103, 112, 162

 C

Número de contatos entre átomos de dois resíduos de uma proteína - 126

 I

Percentual de interações entre resíduos de uma proteína - 126, 127

 P_m

Probabilidade de mutação - 107

 α -hélice

Estrutura secundária de uma proteína - iv, vii, 28–31, 129

τ -EO

Otimização Extrema Generalizada, do Inglês *Generalized Extremal Optimization* - ix, x, xiii, xvi, 4, 74, 78, 80, 82–92, 94–96, 98–100, 102, 103, 105–108, 110–114, 118, 119, 161–163, 165

agrupamento por ligação completa

Critério de agrupamento no qual a aglomeração dos objetos envolvidos ocorre caso a distância entre eles seja que o valor de distância em consideração - 51

cromossomo

Conjunto de genes de um indivíduo (genótipo) - 76, 106, 107

cvtpdb

Convert PDB, aplicativo destinado à manipulação de arquivos PDB e geração de redes de contato de uma proteína - 4, 129, 130

fenótipo

Características de um indivíduo resultantes de seu genótipo - 75, 76

folha- β

Estrutura secundária de uma proteína - iv, vii, 28, 30, 31, 129

gene

Região localizável na sequência gênica que corresponde à unidade de herança de traços entre indivíduos [?] - 75, 77

genoma

Espaço de procura para métodos de otimização evolutivos - 75, 76

genótipo

Conjunto de genes de um indivíduo - 75, 76

graphstk

Graphs Toolkit – conjunto de ferramentas para estudo de redes complexas - v, 4, 53, 58, 80, 108

mapeamento genótipo-fenótipo

Mapeamento dos genótipos em fenótipos - 76

mutação

Processo pelo qual um gene de um cromossomo é alterado aleatoriamente - 75, 76, 162

objetivo

Função de qualidade utilizada em processos de otimização - ix, 74, 82–84, 86, 92, 97, 99, 102, 103, 148

seleção natural

Processo evolutivo pelo qual os indivíduos menos adaptados ao ambiente têm menor chance de sobrevivência - 75

shell

Interpretador de comandos presente em sistemas operacionais - xvi, 85

volta β

Estrutura secundária de uma proteína - 28

wwPDB

Worldwide Protein Data Bank - 33, 34

arestas

Unidade básica de um gráfico que conecta dois vértices - 1

vértices

Unidade básica de um gráfico - 1