Pós-Graduação em Ciência da Computação

Jnom: uma ferramenta para encontrar motifs

Por

José Edson de Albuquerque Filho

Dissertação de Mestrado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, 06/2005

Universidade Federal de Pernambuco

Centro de Informática

José Edson de Albuquerque Filho

JNOM: UMA FERRAMENTA PARA ENCONTRAR MOTIFS

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientadora: *Katia Silva Guimarães*

Recife

27 de julho de 2005

Albuquerque Filho, José Edson de

JNOM : uma ferramenta para encontrar motifs / José Edson de Albuquerque Filho. – Recife : O Autor, 2005

xii, 121 folhas : il., fig., tab.

Dissertação (mestrado) – Universidade Federal de Pernambuco. CIn. Ciência da Computação, 2005.

Inclui bibliografia e apêndices.

1. Teoria da computação – Bioinformática. 2. Redes bayesianas – Natureza combinatorial. 3. Gene – Regulação gênica – Fatores de transição – Identificação de mofits (motivos). I. Título.

004.032.26	**CDU (2.ed.)**	**UFPE**
006.4	**CDD (22.ed.)**	**BC2005-544**

1% inspiração, 99% transpiração

—LEONARDO DA VINCI

RESUMO

A regulação gênica é muito importante para o desenvolvimento dos seres vivos, pois é através dela que os organismos conseguem sintetizar proteínas. Durante a síntese de proteínas, há fatores que regulam a expressão gênica pela conexão em posições específicas no genoma.

Os fatores de transcrição conectam-se a subseqüências específicas de DNA, que podem, com dificuldade, ser determinados por análises biológicas. Esse alto grau de dificuldade motiva os cientistas a procurarem meios computacionais mais rápidos e eficientes para solucionar o problema da busca pelos sítios de ligação dessas regiões promotoras. A identificação de sítios de ligação envolve duas etapas principais: aprender modelos de sítios de ligação e buscar sítios em novas seqüências.

A natureza combinatória dos fatores de transcrição é o mecanismo pelo qual as células dos organismos superiores atuam para controlar a expressão de conjuntos inteiros de genes. O objeto deste trabalho é investigar essa natureza combinante e construir uma ferramenta capaz de considerar tal ação combinada para encontrar novos *motifs* a partir de alguns já conhecidos.

Palavras-chave: Motif, Gene, Natureza Combinatorial, Transcrição, Regulação Gênica.

ABSTRACT

The gene regulation is very important for the development of the beings livings creature. It is through this regulation that the organisms synthecize proteins. During the protein synthesis, it has factors that they regulate the genic expression for connections in specific positions in genoma.

The transcription factors connect it specific DNA subsequences. It is hard to determinate these positions by biological analyses. This high degree of difficulty motivates the scientists to look efficient and faster computational ways to solve the search problem for promotional regions binding sites. The binding sites identification involves two main stages: Learning binding sites models and to search it in new sequences.

The transcription factors combinatorial nature is the mechanism for which the cells of the superior organisms acts to control the expression of entire sets of genes. The object of this work is to investigate this combination and to construct a tool capable to consider such combined action to find new motifs from some already known.

Keywords: Motif, Gene, Combinatorial Nature, Transcription, Gene Regulation.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO 1

INTRODUÇÃO

Este capítulo apresenta os objetivos e a motivação desta dissertação, analisando aspectos inerentes ao processo de reconhecimento de *motifs*, que são regiões específicas de seqüências genômicas. A Seção 1.1 expõe brevemente a motivação de nosso trabalho, que será posteriormente aprofundada na dissertação. Já na Seção 1.2 serão sucintamente revelados os objetivos deste trabalho, enquanto a Seção 1.3 contém uma rápida descrição do trabalho realizado na dissertação. Finalmente, a Seção 1.4 fornece uma visão dos demais capítulos desta dissertação.

1.1 MOTIVAÇÃO

A regulação gênica está intimamente ligada com a síntese de proteínas. Esse mecanismo é muito importante para o desenvolvimento dos seres vivos, pois é através dele que os organismos conseguem sintetizar proteínas. Recentemente foram realizadas experiências de clonagem e produção de órgãos utilizando-se deste conhecimento[ABJ+99].

Um interessante problema da biologia moderna é o entendimento de mecanismos da regulação da transcrição[1]. Muitos aspectos dessa regulação envolvem fatores de transcrição (proteínas ligantes ao DNA). Esses fatores regulam a expressão gênica pela conexão em posições específicas de regiões do genoma (conjunto de genes de uma espécie) que podem estar próximas ou não, como veremos em maiores detalhes oportunamente.

Os fatores de transcrição conectam-se a subseqüências especificas de DNA, os promotores, que podem, com dificuldade, ser determinados por análises biológicas. Esse alto

[1] A regulação da transcrição é o mecanismo pelo qual os organismos sintetizam proteínas.

grau de dificuldade vêm motivando os cientistas a procurarem meios computacionais mais rápidos e eficientes para solucionar o problema da busca pelos sítios de ligação dos promotores[2].

O crescente aumento da disponibilidade de seqüências completas de genoma motiva tentativas de entender e modelar o mecanismo regulatório através de análises computacionais. A identificação de sítios de ligação envolve duas etapas principais: aprender modelos de sítios de ligação e buscar sítios em novas seqüências.

1.2 OBJETIVOS

O objetivo principal deste trabalho é desenvolver um meio alternativo para encontrar motifs em seqüências de DNA.

Inicialmente foi desenvolvida uma ferramenta para auxiliar os cientistas na busca por essas regiões especiais, os *motifs*, no genoma. Como desenvolvemos essa ferramenta usando Java, combinamos o fonema inglês da letra "J"com o fonema "nom"da palavra "genome"para compor o nome da ferramenta e a chamamos de Jnom.

O objetivo é a criação de uma ferramenta com qualidade e que atendesse aos requisitos dos usuários, em sua maioria biólogos. Nesse caso, um importante fator para o sucesso da Jnom é a usabilidade, por isso usamos Applet[jav, app] que disponibiliza uma interface amigável ao usuário e ainda é portável a diversos sistemas operacionais. Para alcançarmos a qualidade almejada no desenvolvimento, instanciamos um processo baseado em meta-modelos internacionais (RUP[Inc] e XP[Bec]).

A primeira tarefa para encontrar os *motifs* é aprender modelos de sítios de ligação em potencial em um dado genoma. Usam-se exemplos de sítios de ligação verificados

[2]Os promotores são regiões que auxiliam o início da transcrição.

biologicamente e tenta-se encontrar sítios similares em outras regiões promotoras.

Em seguida, é necessário descobrir uma seqüência de *motifs*(sítios de ligação) em uma coleção de seqüências relativamente longas que são supostamente ligadas pelo mesmo fator. Um exemplo desta tarefa é examinar as regiões promotoras de um conjunto de genes que têm anotação funcional comum ou são co-expressos[3]. Neste caso, um *motif* encontrado indica um possível fator desconhecido que regula o conjunto de genes.

Essas duas tarefas requerem uma descrição do *motif* que caracteriza seqüências em promotores de fatores de transcrição. A literatura biológica sugere que as seqüências relevantes sejam relativamente curtas, de cinco a quinze pares de bases de comprimento (5pb a 15pb). Embora esses sítios de ligação sejam razoavelmente preservados, eles ainda apresentam alguma variação. Essa variação deve ser compreendida pelo algoritmo utilizado na Jnom para encontrar novos *motifs*.

A natureza combinatória dos fatores de transcrição é o mecanismo pelo qual as células dos seres superiores (eucariotes) atuam para controlar a expressão de conjuntos inteiros de genes. A combinação de fatores de transcrição atua como um comutador[4], controlando inclusive a velocidade da transcrição. Esse mecanismo é tão poderoso que um único sinal pode completar a seqüência combinatória de um conjunto de genes responsáveis pela produção de um órgão inteiro. Esse é um fator importante que parece estar em segundo plano na maioria das ferramentas computacionais desenvolvidas[Mid03]. A intenção deste trabalho é investigar essa natureza combinante e tentar utilizar esse fato para melhorar o desempenho em relação a ferramentas existentes.

Uma importante contribuição desta pesquisa é investigar alternativas para construção de uma ferramenta capaz de considerar a ação combinada dos fatores de transcrição

[3]Pode-se dizer, no contexto deste trabalho, que genes com anotação funcional ou coexpressos são genes que se relacionam. Essa relação pode ser usada para auxiliar a predição de sítios de ligação.

[4]Pode-se dizer, nesse contexto, que um comutador é um microprocessador de sinais.

através da seqüência de genes com a finalidade de encontrar novos *motifs* a partir de alguns já conhecidos.

1.3 METODOLOGIA

Este trabalho foi desenvolvido seguindo as etapas: entendimento do problema, levantamento do estado da arte, instanciação de um processo de desenvolvimento específico, implementação da ferramenta e realização de testes.

Como o problema tratado neste trabalho é interdisciplinar, foi necessário um estudo aprofundado sobre a essência biológica da regulação, para que os principais conceitos fossem entendidos e assim técnicas computacionais pudessem ser analisadas e propostas.

Em seguida, foi feito um levantamento das tecnologias e métodos utilizados atualmente na tentativa de modelar e solucionar o problema. Verificamos que a análise de dependência entre fatores era uma característica determinante na melhoria de desempenho das ferramentas mais recentes.

Após instanciar o processo de desenvolvimento, analisamos algumas técnicas alternativas para modelar as dependências entre as posições dos *motifs* e implementar a Jnom segundo tais técnicas. As principais foram as redes Bayesianas[Fri98, FGG97], seu caso particular a matriz de posição específica[Bai99] e o método de raciocínio baseado em casos[Kol92, Bar01].

Depois da ferramenta devidamente implementada realizamos testes sob condições semelhantes a outras ferramentas previamente estudadas. Nossos testes ratificaram que modelar as dependências ajuda no desempenho e obtivemos uma melhora em relação a abordagens semelhantes como poderemos observar no Capítulo 6.

1.4 ORGANIZAÇÃO DA DISSERTAÇÃO

Devido à interdisciplinaridade, o Capítulo 2 apresentará uma breve revisão sobre principais conceitos biológicos. Conceituaremos aminoácidos, nucleotídeos e explicaremos brevemente o funcionamento do mecanismo da regulação protéica celular.

No Capítulo 3 será realizado um levantamento do estado da arte. Descreveremos as principais técnicas e ferramentas atualmente utilizadas. No final do capítulo faremos um breve resumo sobre um recente experimento realizado por Friedman e Barash que considera a dependência para melhorar o desempenho de preditores.

Após entendimento do problema, faremos uma verificação das técnicas que são atualmente empregadas para a solução deste, então analisaremos métodos alternativos para implementá-las. Tais métodos estarão descritos no Capítulo 4. Antes de encerrá-lo, explicaremos detalhadamente como utilizar a fusão de algumas técnicas tais como redes Bayesianas e raciocínio baseado em casos para modelar as dependências e melhorar o desempenho da Jnom em relação a outros preditores de *motifs*.

No Capítulo 5, apresentaremos a Jnom, ferramenta desenvolvida segundo uma análise préviamente realizada e em conformidade ao processo sugerido no Apêndice D. Neste capítulo haverá uma descrição sumária de como utilizar a ferramenta, de como os dados são representados e de como é fornecida a entrada e a saída das informações. Faremos ainda uma análise sobre manutenção e evolução, indicando suas vantagens e limitações.

Apresentaremos os resultados dos testes realizados com Jnom no Capítulo 6, além de comparar algumas soluções e mostrar resumidamente os resultados obtidos por Barash e outros[BEFK03].

Os comentários gerais e considerações finais sobre o trabalho apresentado nesta dissertação são tecidas no Capítulo 7. Neste, sumarizaremos as contribuições e encerraremos a dissertação com uma discussão sobre os trabalhos futuros.

CAPÍTULO 2

ASPECTOS BIOLÓGICOS RELEVANTES

2.1 INTRODUÇÃO

Para uma melhor compreensão do trabalho, faremos uma breve introdução sobre regulação gênica e alguns aspectos e conceitos biologicamente relevantes. Alguns desses conceitos atuam como motivadores do mesmo.

Durante a fase embrionária, a célula ovo se multiplica e suas células filhas se multiplicam e também se diferenciam dando assim origem a todas as células do organismo adulto. Essa diferenciação ocorre porque cada célula pode expressar diferentes conjuntos de genes sob diferentes condições.

No final deste capítulo, mostraremos como uma célula pode especificar quais dos seus milhares de genes vão se expressar em um dado momento. Vale salientar que esse é um importante mecanismo, pois ele é responsável pela diferenciação celular. Isto quer dizer que quando um organismo se desenvolve as células vão se diferenciando e se especializando em suas funções. Podemos citar como exemplo de organismo multicelular o próprio homem, que é, no início da embriogênese, uma única célula.

2.2 ÁCIDOS NUCLÉICOS

São substâncias que regulam o processo vital básico de todos os organismos. Podemos afirmar que os ácidos nucléicos são polímeros de nucleotídeos (polinucleotídeos). A sua composição química é feita por um fosfato, uma pentose mais uma base nitrogenada.

Essa pentose pode ser a ribose para o RNA(ácido ribonucléico) ou a desoxirribose para o DNA(ácido desoxirribonucléico). As bases podem ser Adenina(A), Guanina(G), Citosina(C), Timina(T) ou Uracil(U). Esta última(Uracil), presente apenas em RNA substituindo a Timina, que aparece apenas no DNA[ABJ+99]. Isto é, nas cadeias de DNA não encontramos Uracil e nas de RNA não encontramos Timina.

Cada grupo de três bases, ou códon, do DNA forma um aminoácido[MSS+02]. É fácil notar que quatro bases (A, T, G, C) combinadas três a três formam 64 arranjos diferentes. Mas, existem apenas 20 tipos de aminoácidos na natureza. Isso significa que existem diferentes seqüências de bases para o mesmo aminoácido, como ilustra a Figura 2.1. Como o mesmo aminoácido pode ser codificado por seqüências diferentes de nucleotídeos, dizemos que o código genético é degenerado.

2.2.1 DNA

O DNA, o ácido nucléico mais importante para esta pesquisa, é constituído de uma cadeia dupla de nucleotídeos. Em cada cadeia, os nucleotídeos se ligam através de uma reação entre fosfato[1]. A ligação entre as cadeias é feita por pontes de hidrogênio entre as bases da seguinte forma: Adenina se liga com Timina e Citosina sempre com Guanina. No RNA a Adenina se liga com a Uracil. A ligação de uma porção contínua de nucleotídeos assume o formato de uma escada em espiral[MSS+02].

O Cromossomo é o DNA condensado que pode ser observado em microscópio óptico no início da divisão celular[ABJ+99].

2.2.2 RNA

O RNA, ou ácido ribonucléico, é uma molécula intermediária na síntese de proteínas que faz a intermediação entre o DNA e as proteínas. Ele é formado por uma cadeia

[1] O fosfato de um nucleotídeo e a hidroxíla do carbono 3 da ribose do próximo nucleotídeo

1ª posição	2ª posição								3ª posição
	U		C		A		G		
U	UUU	Phe (F)	UCU	Ser (S)	UAU	Tyr (Y)	UGU	Cys (C)	U
	UUC		UCC		UAC		UGC		C
	UUA	Leu (L)	UCA		UAA	TERM	UGA	TERM	A
	UUG		UCG		UAG		UGG	Trp (W)	G
C	CUU	Leu (L)	CCU	Pro (P)	CAU	His (H)	CGU	Arg (R)	U
	CUC		CCC		CAC		CGC		C
	CUA		CCA		CAA	Gln (Q)	CGA		A
	CUG		CCG		CAG		CGG		G
A	AUU	Ile (I)	ACU	Thr (T)	AAU	Asn (N)	AGU	Ser (S)	U
	AUC		ACC		AAC		AGC		C
	AUA		ACA		AAA	Lys (K)	AGA	Arg (R)	A
	AUG	Met (M)	ACG		AAG		AGG		G
G	GUU	Val (V)	GCU	Ala (A)	GAU	Asp (D)	GGU	Gly (G)	U
	GUC		GCC		GAC		GGC		C
	GUA		GCA		GAA	Glu (E)	GGA		A
	GUG		GCG		GAG		GGG		G

Figura 2.1. Composição dos Aminoácidos

de ribonucleotídeos, que, por sua vez, são formados por um grupo fosfato, um açucar (ribose), e uma base nitrogenada conorme ilustra a Figura 2.2. O DNA pode sintetizar RNA, a este processo dá-se o nome de transcrição. Existem basicamente três tipos básicos de RNA: o transportador, o mensageiro e o ribossômico.

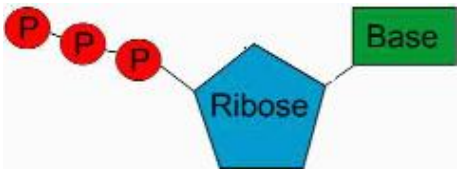


Figura 2.2. Representação esquemática do RNA

RNAt - Transportador

As moléculas de RNA transportador, ou simplesmente RNAt, têm uma conformação espacial característica e possuem duas regiões especiais (ver representação esquemática na Figura 2.3). Em uma delas, os RNAt se ligam a aminoácidos; em outra há uma seqüência de três bases nitrogenadas que usamos para identificar o RNAt. As ligações do RNAt com aminoácidos são específicas, isto é, alguns tipos de RNAt só se ligam a certos tipos de aminoácidos. Por exemplo, o RNAt que tem em uma das extremidades a base AAC liga-se, na outra extremidade, ao aminoácido leucina.

Em alguns casos, mais de um tipo de RNAt pode se ligar ao mesmo tipo de aminoácido. Por exemplo, o aminoácido leucina pode se ligar aos RNAt que têm, na extremidade correspondente, as bases CUC, CUA, CUG e CUU. No total são vinte aminoácidos que podem se ligar a RNAt com diferentes combinações de três bases nitrogenadas[ABJ+99].

RNAm - Mensageiro

Os segmentos de RNAm diferem entre si de acordo com as bases nitrogenadas que contêm. Por exemplo: o RNAm composto pela seqüência AAC AGU CAA CCC AUA

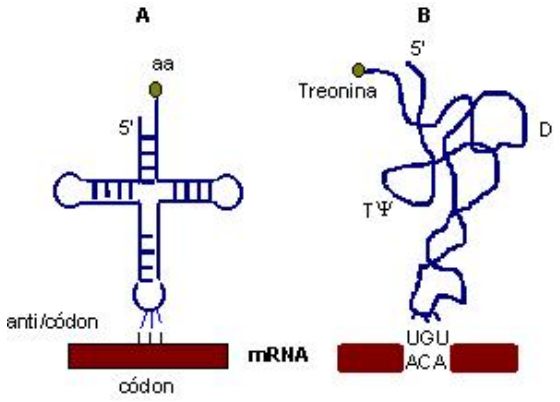


Figura 2.3. A - Estrutura secundária de um RNAt. Posição do anti-códon e do braço aceptor de aminoácido estão indicadas. B - Estrutura tridimensional determinada por difração de Raio X.

GGC é diferente do RNAm composto pela seqüência CGU CUU ACC CAA AAA UUU. A cada conjunto de três bases nitrogenadas do RNA mensageiro corresponde um conjunto de três bases nitrogenadas nos RNA transportadores. Entre o RNA mensageiro e o transportador pode haver uma ligação das bases nitrogenadas segundo certa correspondência: a base A (adenina) se liga à base U (uracila) e a base C (citosina) se liga à base G (guanina).

Cada seqüência de três bases nitrogenadas do RNA mensageiro recebe o nome de códon. A seqüência de três bases nitrogenadas do RNA transportador correspondente (isto é, aquela que se liga ao RNAm) recebe o nome de anticódon. Assim o códon GUU é complementar ao anticódon CAA.

O RNA transportador liga-se ao RNA mensageiro sempre da mesma forma: o anticódon se encaixa no códon. Os RNA transportadores que tiverem os conjuntos de três bases (anticódon) UUG e UCA vão se ligar na região do RNA mensageiro onde estiverem os conjuntos (códon) AAC e AGU. O RNAm serve para codificar as proteínas e deve ter seus códons lidos durante o processo de tradução.

RNAr - Ribossômico

Moléculas de RNA ribossômico unem-se a proteínas e formam o ribossomo. O ribossomo é uma estrutura presente no citoplasma da célula e também o local onde ocorre a síntese de proteínas[2]. Uma representação esquemática do RNA ribossômico efetuando uma tradução em conjunto com um RNAm pode ser vista na Figura 2.4.

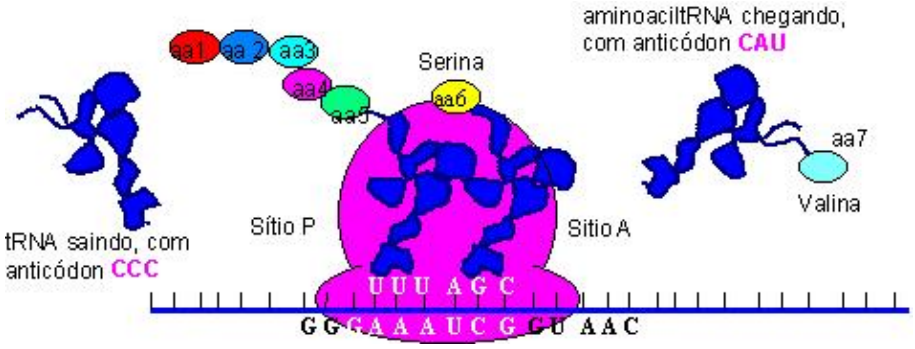


Figura 2.4. Mecanismo de adição de um único aminoácido à cadeia nascente de polipeptídio durante a tradução do mRNA

Neste tipo de RNA há uma região onde o RNA mensageiro se liga e dois locais onde podem se combinar com RNA transportadores. Estes dois últimos locais recebem o nome de sítio A (do aminoácido) e sítio P (da proteína).

2.3 AMINOÁCIDOS

Os aminoácidos, ou simplesmente AA, são moléculas orgânicas que apresentam em sua estrutura química pelo menos um radical amina (-NH2) e um radical carboxílico (-COOH) [ABJ+99].

[2]Apesar de as moléculas de RNA serem constituídas de vários conjuntos de substâncias, elas são muito pequenas e não são vistas nem mesmo ao microscópio. Para estudá-las usam-se corantes e substâncias marcadoras. Os ribossomos, constituídos por RNA e proteínas, são estruturas que podem ser vistas ao microscópio como granulações no citoplasma da célula[ABJ+99].

Os aminoácidos se ligam para formar proteínas através de uma ligação chamada ligação peptídica, que é feita entre a hidroxila e um hidrogênio da amina, liberando dessa forma, moléculas de água(H_2O). Por analogia, se um nucleotídeo equivalesse a uma pérola, um colar de pérolas seria uma seqüência de nucleotídeos. Dois colares de pérolas, lado a lado, torcidos em espiral, equivalem a uma cadeia de DNA. Se a cadeia for simples (só um colar) e tiver ribose e uracila em nucleotídeos, será de RNA.

Como já dito anteriormente, existem apenas 20 aminoácidos diferentes na natureza. Mesmo com esse número reduzido, algumas espécies não conseguem sintetizar todos os aminoácidos necessários a seu metabolismo[3].

2.4 PROTEÍNAS

As proteínas são macromoléculas (polímeros) e possuem uma grande quantidade de aminoácidos em sua composição. Elas estão envolvidas no controle de uma série de atividades no organismo, como por exemplo: atividades plásticas, energéticas, defesa, hormonal, transporte, enzimática.

As proteínas podem ser classificadas usando-se os aminoácidos de sua composição através dos tipos, quantidade, seqüência e estrutura (forma).

Podemos estudar as proteínas em diferentes níveis de detalhe. Estes níveis dividem a análise das proteínas em: primárias, secundárias, terciárias e quaternárias, como pode ser visto na Figura 2.5.

Outra forma bastante utilizada para se classificar as proteínas é através de sua com-

[3]Os seres humanos não conseguem sintetizar todos os aminoácidos de que necessitam, esses aminoácidos não sintetizados chamados essenciais, devem vir pela alimentação, são eles: Leucina, IsoLeucina, Fenilalanina, Triptófano, Treonina, Lisina, Metionina e Valina.

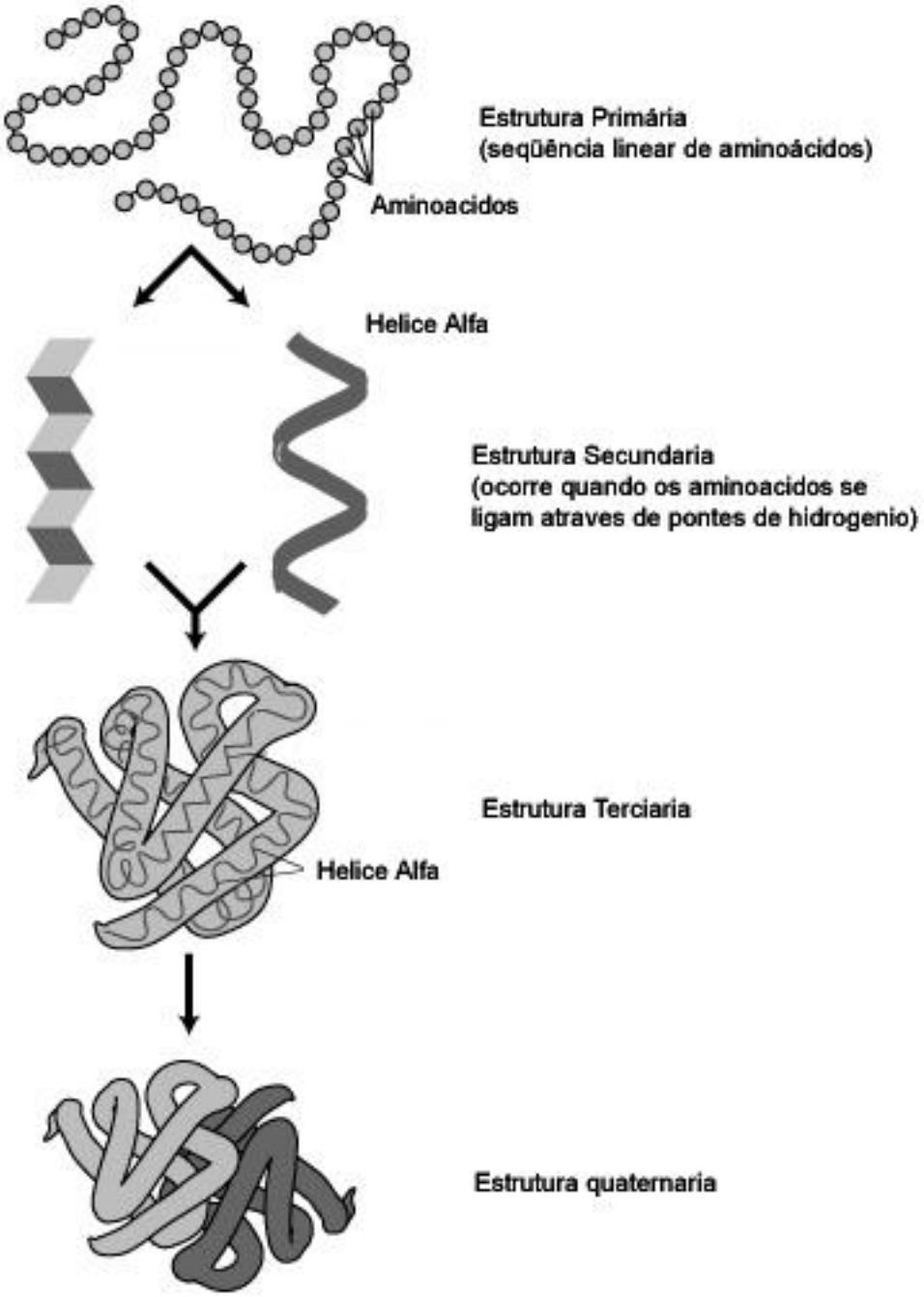


Figura 2.5. Níveis de detalhe de uma Proteína

posição. Uma proteína é dita simples se for formada exclusivamente por aminoácidos. Por exemplo, albumina, colágeno, globulinas. Já a proteína composta apresenta um grupo prostético além dos aminoácidos, como por exemplo, as nucleoproteínas que possuem como grupo prostético DNA ou RNA, ou as cromoproteínas que têm pigmentos em sua composição.

2.4.1 Síntese de Proteínas

A transcrição e a tradução são os meios pelos quais as células interpretam, ou expressam, suas instruções genéticas, os genes. A transcrição é a cópia de uma subseqüência do DNA em RNA. Ao produto desse RNA transcrito dá-se o nome de tradução.

O DNA controla a produção de proteínas que ocorre nos ribossomos. Podemos dizer que a seqüência de aminoácidos de uma proteína depende da seqüência de nucleotídeos de um segmento de DNA que codifica essa proteína.

Dessa forma, de acordo com a seqüência de bases nitrogenadas (A, T, C, G) de uma das hélices do DNA, a célula orienta a seqüência de aminoácido produzindo uma proteína específica.

2.4.2 Proteínas Estruturais x Proteínas Especializadas

É importante lembrar que muitas proteínas estão presentes em todas as células do organismo, como pode ser comprovado pela técnica de eletroforese bidimensional[4]. Essas proteínas são chamadas de proteínas estruturais ou *housekeeping*. Os genes que

[4]A eletroforese bidimensional é a técnica que se aplica à separação de um grande número de proteínas ao mesmo tempo e com alto poder de resolução. Utiliza basicamente duas propriedades: a carga elétrica da molécula e sua massa molecular. A combinação dessas duas propriedades físico-químicas permite separar milhares de proteínas de um modo relativamente simples

codificam tais proteínas são chamados genes *housekeeping*. Como exemplo podemos citar as proteínas do citoesqueleto ou as proteínas ribossomais.

Existe outro grupo de proteínas, são as proteínas especializadas responsáveis pelas diferentes funções celulares. Nos mamíferos, por exemplo, a hemoglobina é produzida apenas nos reticulócitos.

2.5 CROMATINA X CROMOSSOMOS

As duas estruturas(Cromatina e Cromossomos) são compostas pelo mesmo material, o DNA, que é uma seqüência dupla de nucleotídeos. Nucleotídeos são aquelas unidades básicas formadas por uma pentose (ribose, no RNA, ou desoxirribose, no DNA), um grupo fosfato e uma base nitrogenada (adenina, guanina, timina, citosina ou uracila no RNA). Os nucleotídeos encadeados formam um ácido nucléico (DNA ou RNA). É importante entender essa diferença pois a observação do DNA é feita durante a a divisão celular e a síntese de proteínas ocorre na célula interfásica.

O material das duas estruturas é o mesmo, a diferença é uma questão de momento. A cromatina é um filamento de DNA muito longo e muito fino, localizado no núcleo da célula interfásica (não em divisão). Na célula humana, contam-se 46 desses filamentos. Quando a célula inicia seu processo de divisão (mitose ou meiose), esses filamentos se espiralizam (enrolam-se sobre si mesmos) e se condensam, transformando-se nos cromossomos. Ou seja, eles são semelhantes, porém com estruturas diferentes e observados em diferentes fases da vida celular. O cromossomo é a cromatina condensada pronta para ser duplicada. Essa relação pode ser ilustrada com a Figura 2.6.

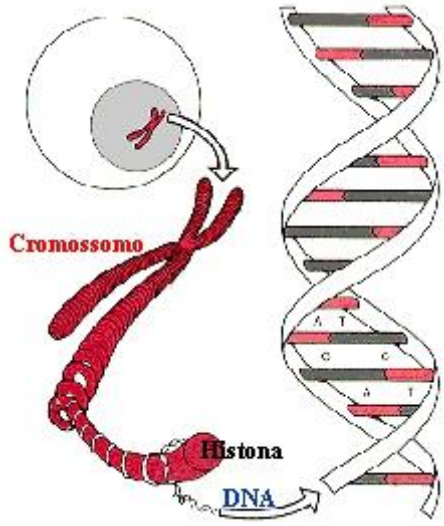


Figura 2.6. Da cromatina ao DNA

2.6 REGULAÇÃO GÊNICA

Agora que já relembramos os principiais conceitos biológicos, seremos mais específicos. Vamos falar um pouco sobre regulação gênica.

As células selecionam os genes que serão expressos sem alterar a seqüência do DNA. Note que, se o DNA fosse alterado irreversivelmente, os cromossomos de uma célula diferenciada não seriam capazes de guiar o desenvolvimento de outro tipo de célula, nem de um organismo inteiro. Para comprovar a afirmação que o DNA está intacto, mesmo nas células diferenciadas, a seguinte experiência foi realizada: um núcleo de uma célula epitelial de uma rã adulta foi injetado em um ovo de rã cujo núcleo foi previamente removido. Em alguns casos o ovo se desenvolveu normalmente. Com esse experimento pode-se observar que a célula epitelial não perdeu nenhuma seqüência importante do DNA. Experimentos feitos em plantas [ABJ+99]levaram a conclusões parecidas. Recentemente, um experimento semelhante foi realizado com sucesso em ovelhas causando bastante polêmica.

2.6.1 Autoregulação Celular

O controle da produção de proteínas pelas células é feito de diversas formas, dentre elas destacamos:

i) Controlando quando e com qual freqüência um certo gene é transcrito.

ii) Controlando como o transcrito primário sofre *splicing*[5] ou é de outra forma processado.

iii) Selecionando quais RNAm são traduzidos pelos ribossomos

iv) Ativando ou inativando proteínas depois que elas foram sintetizadas.

Embora existam várias formas de controle, como o controle do processamento, tradução ou controle da atividade protéica, o controle soberano é feito no início da transcrição[ABJ+99].

Devido a esse controle soberano, vamos enfatizar as próximas seções nos componentes de controle da transcrição.

2.6.2 Combinação das Proteínas Reguladoras

O maior ponto de controle é no início da transcrição, mais precisamente na região promotora do DNA, onde a RNA polimerase se liga e inicia a transcrição. Dentro do promotor temos uma região iniciadora que em bactérias é chamada operador (ver detalhes na Figura 2.7). Num segmento do DNA, correspondente a um gene que codifica uma determinada proteína, são encontradas regiões codificadoras (exons) alternando-se com regiões não-codificadoras (introns). O transcrito resultante não é funcional e só

[5] *splicing* é um processo de junção e remoção de partículas no RNA chamadas de exons e introns respectivamente.

poderá ser traduzido se for devidamente montado, descartando-se os introns e unindo-se os exons em seqüência ordenada. Este tipo de modificação do transcrito primário é denominado *splicing* (cortar e colar; montagem) e ocorre dentro do núcleo. O transcrito processado e pronto para migrar para o citoplasma é o RNA mensageiro.

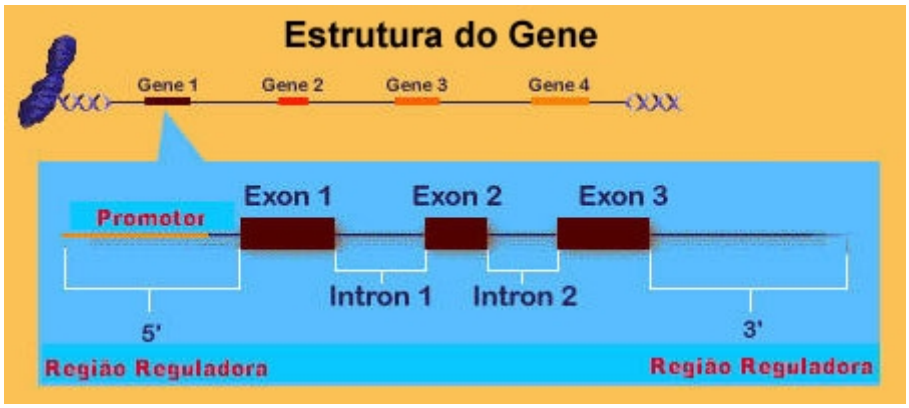


Figura 2.7. Representação esquemáticas das regiões reguladoras

É no operador que são ligados os ativadores ou os repressores. É importante destacar que existem outros fatores influenciando no controle da transcrição, tais como, tipo da célula, idade, vizinhança ou sinais externos.

Não é raro encontrar seqüências regulatórias curtas, com cerca de 10 pares de bases (10pb), que respondem a sinais específicos. Nos eucariotes, esse número é bem diferente. Algumas seqüências reguladoras chegam a ter mais de 10000 pares e funcionam como microprocessadores moleculares respondendo a vários sinais e controlando inclusive a velocidade de início da transcrição.

Na Figura 2.8 pode-se observar a formação do complexo de iniciação da transcrição em eucariontes. Considere, apenas para título de ilustração, que TFXXX são fatores de transcrição. No esquema representado por esta ilustração, TFIID liga-se a região TATA, possibilitando a ligação de TFIIB. Isso é seguido pela ligação de TFIIF e RNA-polimerase II. TFIIE, TFIIH e TFIIJ então se juntam ao complexo. TFIIH usa ATP para fosforilar

a RNA-polimerase II, mudando a sua conformação de forma que a RNA-polimerase é liberada do complexo e é capaz de iniciar a transcrição [ABJ+99].

Essas seqüências de regulação não funcionam sozinhas, elas precisam se ligar às proteínas regulatórias. É esse conjunto combinado de proteína-DNA que forma o comutador de controle da transcrição.

A ligação da proteína com o DNA é facilitada pelo perfeito ajuste das superfícies ligantes entre os fatores e a fita de DNA. Esses fatores são específicos e, em geral, as proteínas se ajustam na fenda maior da dupla hélice de DNA. A ligação é feita através de diversas pontes de hidrogênio, ligações iônicas e interações hidrofóbicas. A Figura 2.9 ilustra a ligação entre uma Guanina e uma Citosina enfatizando as pontes de hidrogênio.

2.6.3 Procariontes x Eucariontes

Antes de discutimos como os genes são ligados e desligados é importante saber a diferença entre procariontes e eucariontes.

Os procariontes foram a primeira forma de vida celular na Terra. São microorganismo sem organelas e sem um núcleo celular bem definido. Na Figura 2.10 pode ser vista a representação esquemática de uma célula procarionte e na Figura 2.11 a representação esquemática de uma célula de um eucarionte.

Existe uma teoria, chamada de simbiogênese, que tenta explicar o processo de evolução dos eucariontes. A simbiogênese é a formação de novos organismos por cooperação de organismos pré-existentes. A simbiogênese como motor da evolução foi proposta pela primeira vez por K. S. Mereschkovsky[Mer26] e por Ivan Wallin[Wal27].

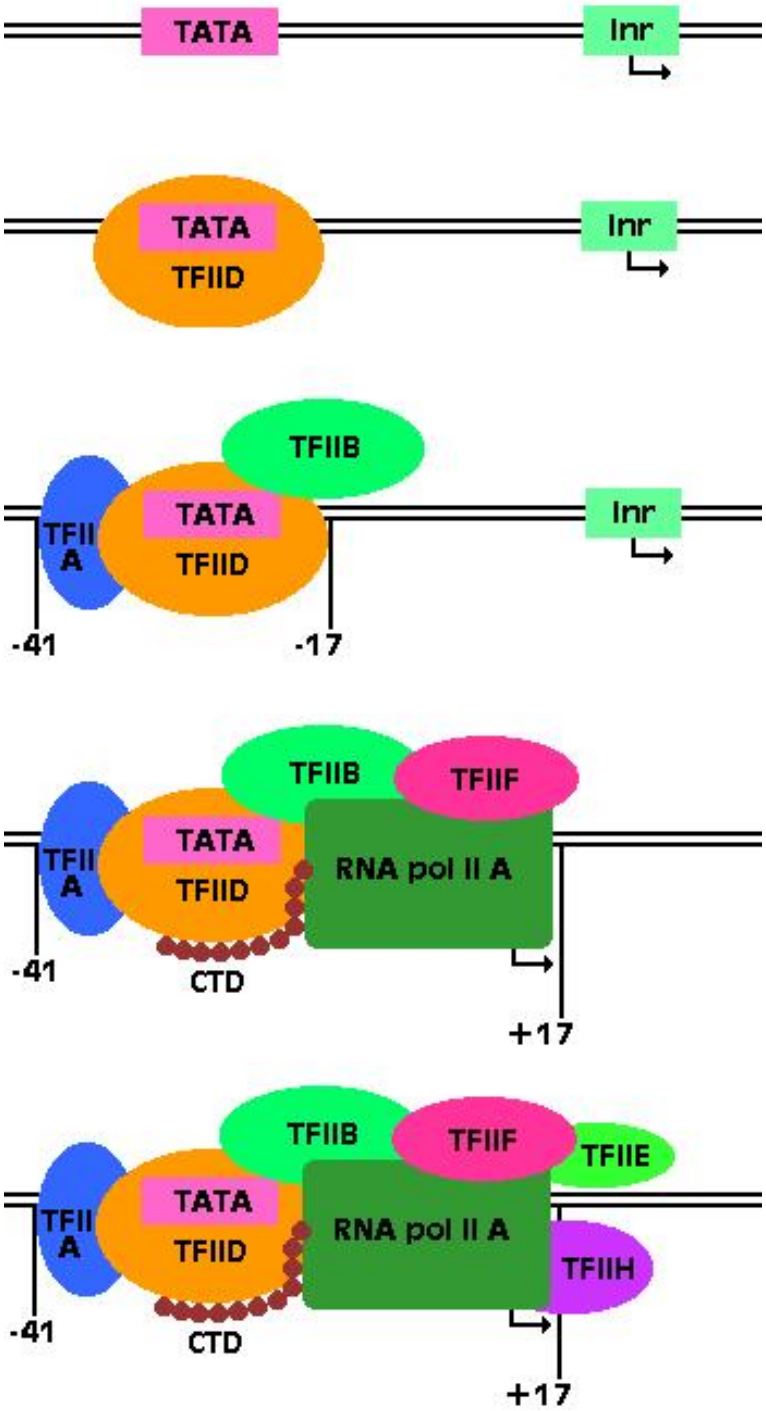


Figura 2.8. Inicio da transcrição em eucariotes

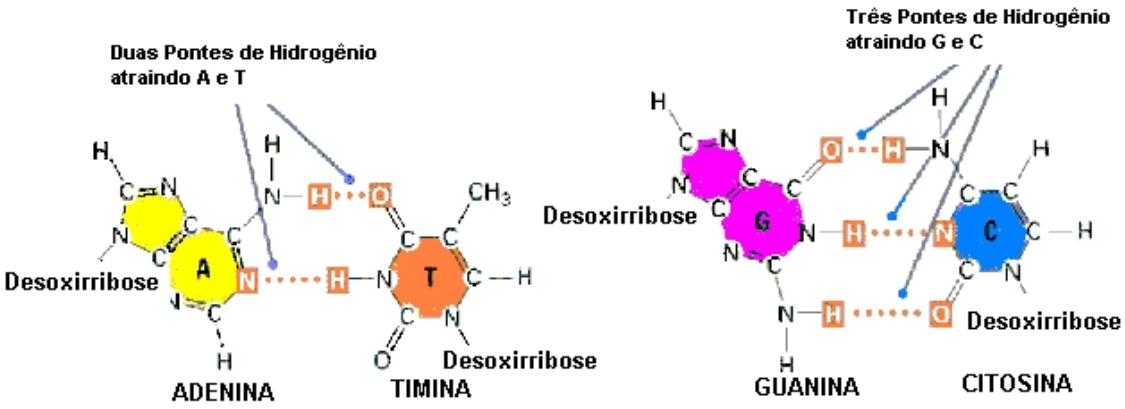


Figura 2.9. Ligação Proteína-DNA

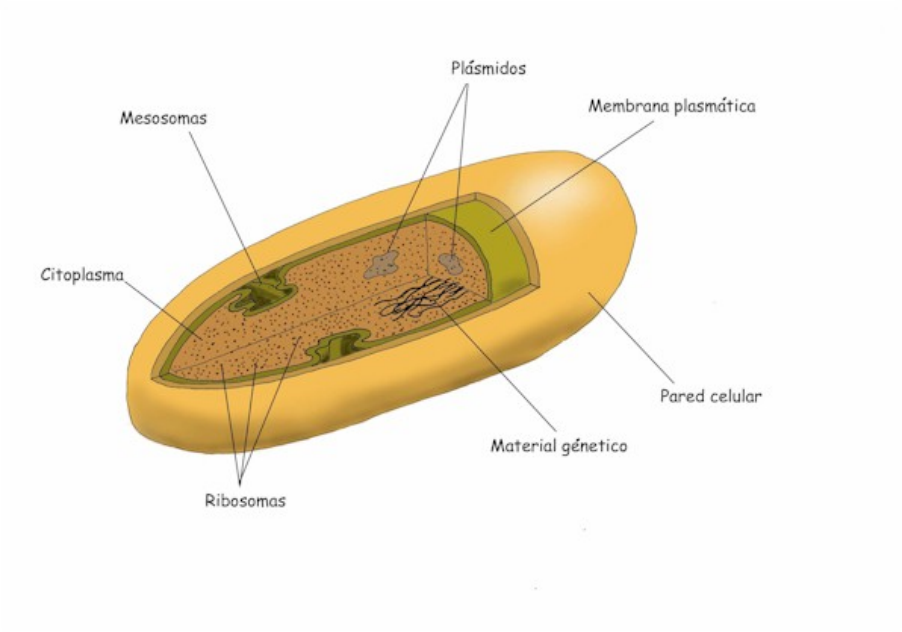


Figura 2.10. Representação esquemática de uma célula procarionte.

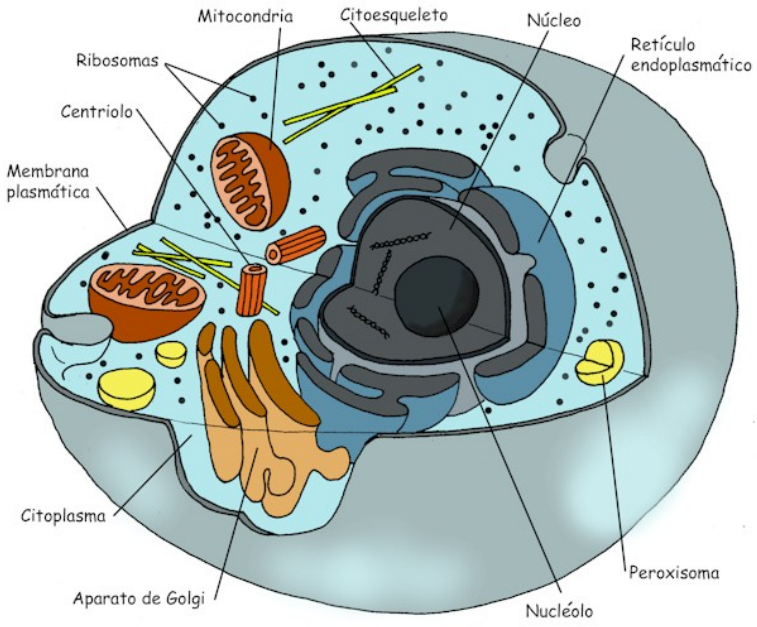


Figura 2.11. Representação esquemática de uma célula de um eucarionte.

2.6.4 Mecanismos Procarióticos de Ativação Gênica

Os procariontes possuem os mecanismos mais simples de regulação conhecidos. Podemos citar como exemplo a *Escherichia coli*, que possui cerca de 4,6 milhões de nucleotídeos e codifica aproximadamente 4000 proteínas. A Figura 2.12 ilustra de forma sucinta o controle da expressão gênica em procariontes.

No metabolismo da lactose por *Escherichia coli*, são necessárias duas enzimas que fazem parte de um operon chamado lac. A primeira enzima é a β-Galactosidase. Essa enzima é capaz de clivar a lactose em glicose e galactose que assim servirão como fonte de carbono para a célula (ver Figura 2.13). A β-galactosidase é dita uma enzima indutível, ou seja, sua expressão varia com as necessidades celulares. Isto é, caso a bactéria esteja crescendo em meio rico em lactose, sua expressão será alta; caso a fonte de carbono seja outro carboidrato, sua expressão será reduzida.

Para produzir Triptofano, cinco genes codificam enzimas simultaneamente na *Es-*

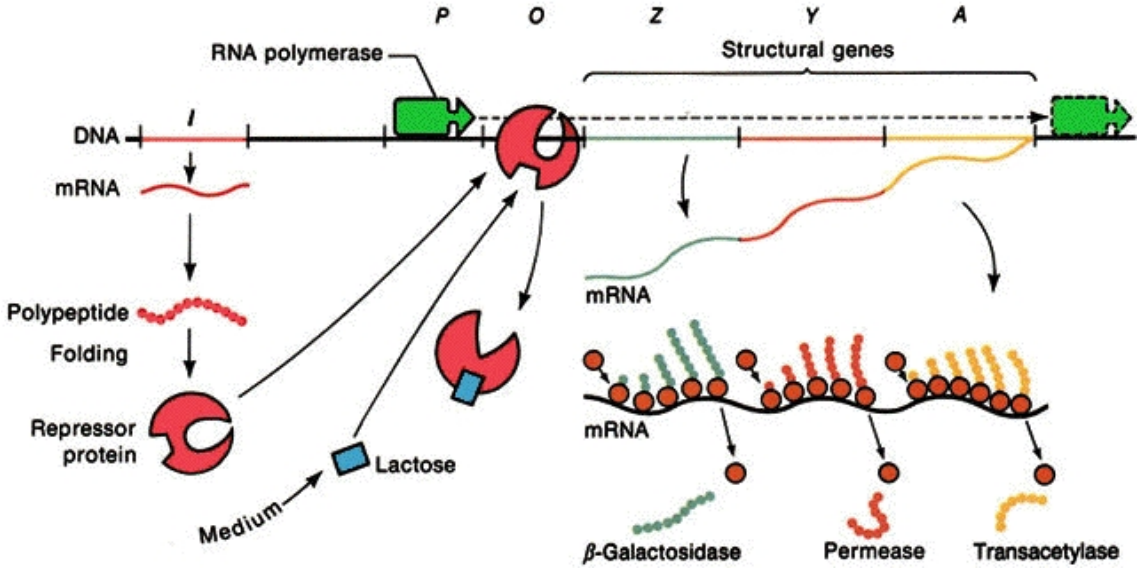


Figura 2.12. Controle da expressão gênica em procariontes

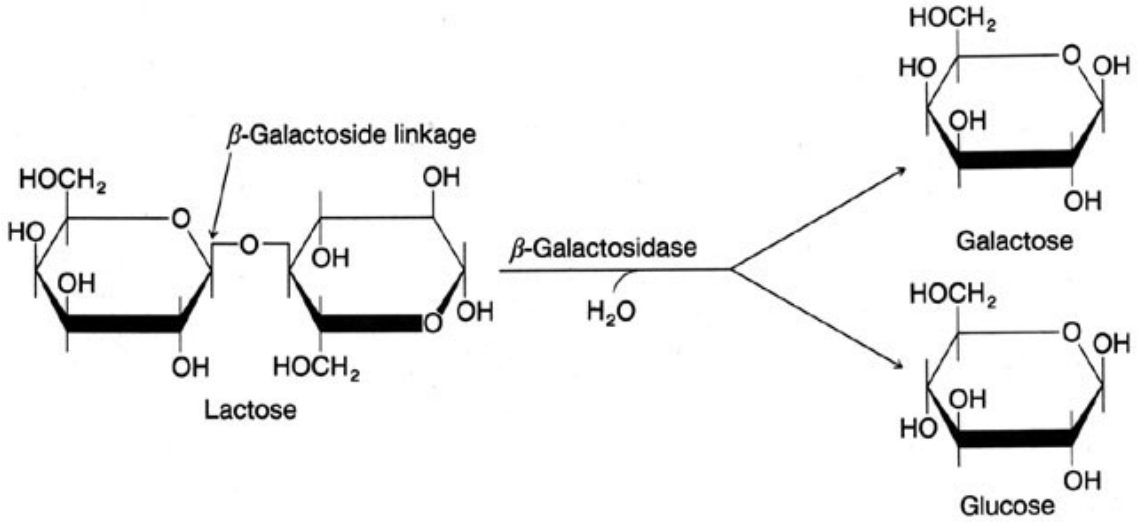


Figura 2.13. Quebra da Galactose.

cherichia coli. Esses genes ficam reunidos num único grupamento no cromossomo e são transcritos num único RNA. Chamamos os genes desse tipo de óperons. Esse é um mecanismo simples freqüentemente encontrado nas bactérias, mas dificilmente encontrado nos eucariotes. Quando o Triptofano está presente no meio, as bactérias não mais necessitam produzi-lo e então a produção é encerrada.

Esse mecanismo de interrupção é devido a uma seqüência curta de aminoácidos, chamada operador, que fica situada dentro do promotor conforme ilustram a Figuras 2.14 e 2.7.

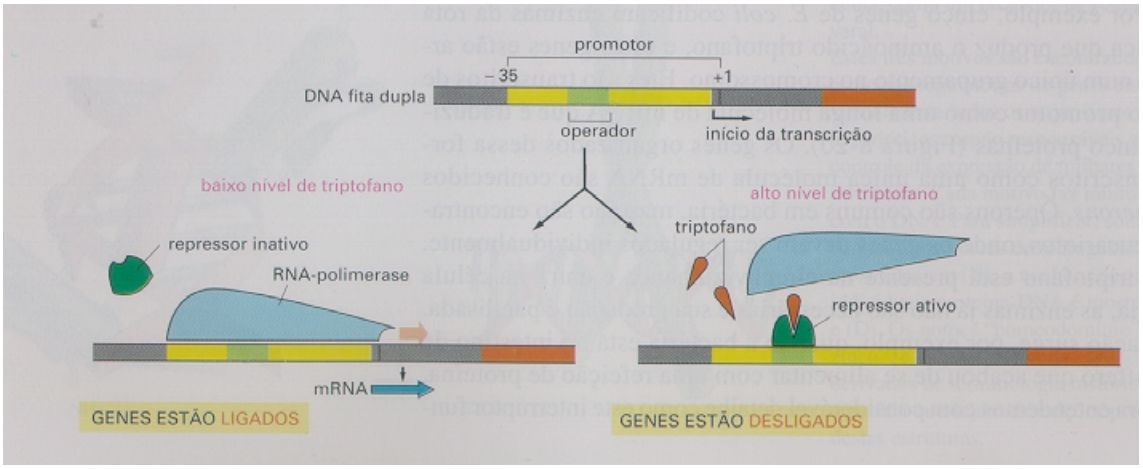


Figura 2.14. Atuação do Repressor

Quando a proteína repressora se liga ao operador, pode-se observar que a RNA polimerase fica impedida de fazer sua ligação. Desse modo a transcrição não ocorre. A taxa de Triptofano no meio causa uma discreta mudança na estrutura tridimensional no repressor tornando-o ativo ou inativo. Dessa forma o repressor é um mecanismo rápido e simples para ligar ou desligar um conjunto de genes em procariontes.

Dizemos que esse mecanismo é rápido, pois a proteína repressora está sempre presentes na célula, isto é, existem genes produzindo essa proteína continuamente. A essa

produção não regulada damos o nome de constitutiva.

Assim como existem os repressores, de maneira análoga há os ativadores. Essas proteínas se ligam ao operador, a polimerase se ajusta e transcreve os genes como mostra a Figura 2.15.

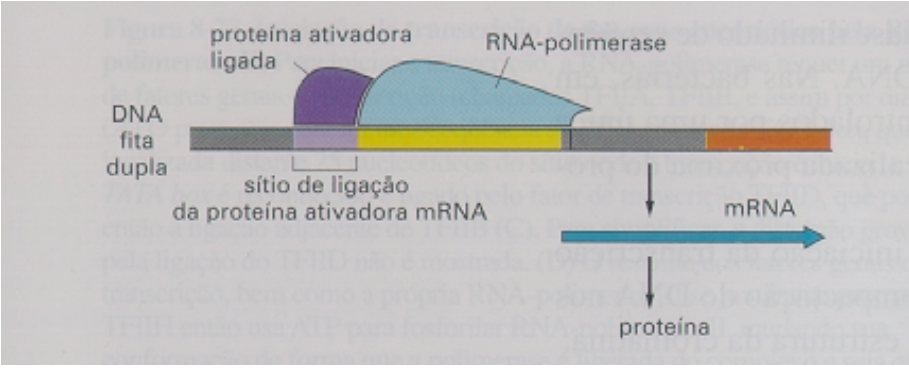


Figura 2.15. Atuação do Ativador

2.6.5 Mecanismos Eucarióticos de Ativação Gênica

Como vimos na seção anterior, os procariontes tem mecanismos simples e práticos. Nos eucariotes o processo é um pouco mais complexo, pois um único gene pode responder a diversos sinais diferentes. O início da transcrição eucariótica difere principalmente devido a três principais fatores:

i) Número de RNA polimerase. Os procariontes só possuem uma enquanto os eucariontes possuem três.

ii) A RNA polimerase dos eucariontes requer proteínas (fatores gerais de transcrição) para começar a transcrição enquanto a RNA dos procariontes pode começar a transcrição sozinha.

iii) Nos eucariontes, os genes reguladores podem estar a milhares de pares de base de distância dos genes regulados. Já nos procariontes, os genes reguladores estão juntos no mesmo agrupamento.

2.7 CONSIDERAÇÕES FINAIS

Para auxiliar no melhor entendimento dessa dissertação relembramos alguns conceitos de biologia. Agora, os conceitos sobre os genes, o DNA, cromossomos, proteínas e aminoácidos devem estar mais sedimentados.

No final desse capítulo, mostramos como uma célula pode especificar quais dos seus milhares de genes vão se expressar em um dado instante. Vale salientar mais uma vez que esse é um importante mecanismo, pois ele é responsável pela diferenciação celular.

Isto quer dizer que, quando um organismo se desenvolve, as células vão se diferenciando e se especializando em suas funções. Podemos citar como exemplo de organismo multicelular o próprio homem. Durante a fase embrionária a célula ovo dos seres humanos se multiplicou e suas células filhas se multiplicaram e se diferenciaram dando origem a todas as células do organismo adulto. Essa diferenciação ocorre porque cada célula pode expressar diferentes conjuntos de genes sob diferentes condições.

No próximo capítulo descreveremos os trabalhos relacionados à predição de *motifs*.

CAPÍTULO 3

TRABALHOS RELACIONADOS À PREDIÇÃO DE MOTIFS

3.1 INTRODUÇÃO

Várias técnicas de tratamento ajudam a identificar a estrutura e a função de proteínas, especialmente quando trabalhamos com todo o genoma. Muitas das técnicas que ajudam na identificação de similaridades estruturais entre proteínas com seqüências diferentes são baseadas nas predições de estruturas secundárias. Predizer mapas de contato a partir da estrutura primária, secundária ou outras características também é uma estratégia bastante utilizada para predizer a função da proteína. A partir dos anos 90, as técnicas de predição de estrutura secundária evoluíram muito com a sofisticação dos métodos de aprendizado e com informações evolucionárias obtidas das divergências entre proteínas da mesma família.

Com o aumento da quantidade de dados disponíveis para treinamentos, os algoritmos tornaram-se cada vez mais precisos. As melhoras nas predições, na maioria das vezes, resultam da combinação de preditores. A combinação do uso de métodos mais sensíveis, tais como, derivação evolucionária de perfis com arquiteturas de aprendizado mais flexíveis de vários tipos vêm melhorando a precisão dos preditores. Alguns métodos atuais de predição de estrutura secundária tipicamente combinam múltiplas redes neurais.

A habilidade de produzir perfis que incluem homologia crescente usando PSI-BLAST[1]

[1]PSI-BLAST é uma ferramenta variante do BLAST que permite comparar proteínas fracamente relacionadas, mas com regiões bem conservadas.

tem contribuído bastante para a melhoria do desempenho. Perfis evolucionários divergentes não só contém informação suficiente para melhorar substancialmente a predição, mas também servem para predizer longos intervalos de resíduos idênticos observados em estruturas secundárias independente dos locais [PPRB02].

Com o passar dos anos, as ferramentas e técnicas de busca por sítios de ligação foram melhorando e atualmente podemos encontrar basicamente duas linhas distintas que consideram ou não a independência das posições do *motif*. Essas duas linhas podem ser representadas pela matriz de pesos (*position specific score matrix*) ou simplesmente PSSM e, mais recentemente, pelas redes Bayesianas. Diversos outros algoritmos auxiliares são utilizados em cada uma dessas técnicas. A Figura 3.1 ilustra um exemplo de PSSM.

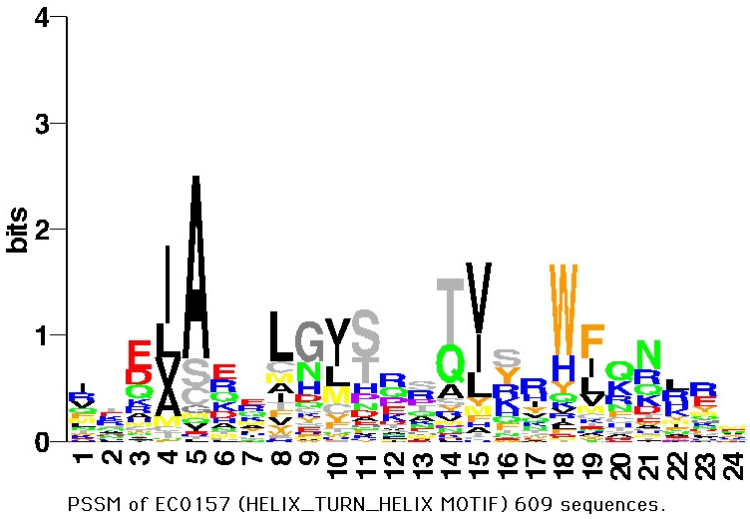


Figura 3.1. Exemplo de PSSM

Algumas ferramentas analisadas utilizam a representação PSSM e assumem que as posições dentro do *motif* são independentes umas das outras. Entretanto, é uma questão biologicamente em aberto se essa forte suposição de independência é razoável. Resultados de experimentos recentes indicam que, em casos específicos, existem dependências entre

posições. Essas dependências podem ser modeladas com as redes Bayesianas [Mid03].

Esse capítulo apresenta o esforço da comunidade científica, algumas ferramentas e técnicas usadas atualmente. Inicialmente falaremos brevemente sobre um banco de dados público e disponível na internet. Em seguida discutiremos sobre alguns dos principais métodos usados para busca por *motifs*, dentre eles a busca por resultado esperado e as redes Bayesianas.

Apresentaremos mais adiante uma breve análise de algumas ferramentas semelhantes à Jnom. Encerraremos o capítulo com uma discussão a respeito de um recente trabalho envolvendo a análise de dependência dos fatores de transcrição[BEFK03]. Esse trabalho foi recentemente publicado e mostra o ganho de desempenho quando são consideradas as dependências entre posições nas seqüências.

3.2 TRANSFAC

TRANSFAC é um banco com dados sobre de regulação de seqüências de DNA em eucariotes e fatores de ligação. Nos próximos parágrafos, detalharemos um pouco mais o que pode ser encontrado neste banco.

Para se converter seqüências de genomas em informações biologicamente úteis, necessita-se de um grande esforço computacional. Por isso, a comunidade científica tem trabalhado insistentemente para desenvolver sofisticadas ferramentas computacionais para permitir deduzir funcionalidades biológicas a partir apenas das seqüências de DNA.

A função dos genes é codificar produtos específicos, mas saber qual gene produz o que e supor suas funcionalidades biológicas é apenas uma parte da tarefa. Outra parte importante dessa tarefa é decifrar o código de regulação, ou seja, descobrir sobre quais condições essa informação é expressa. Por isso, estudar o mecanismo de expressão gênica é uma das maiores tarefas da biologia molecular atualmente. Dado a enorme e

crescente quantidade de informação produzida devido ao esforço da comunidade científica é necessário sistemas computacionais especializados para tratar e armazenar essas informações.

De forma bastante sucinta, TRANSFAC é um banco que coleta dados que são relevantes para expressão gênica em relação à transcrição. Há algum tempo, um grande número de coleções são publicadas nesse banco descrevendo seus fatores de transcrição e as seqüências com que estas seqüências interagem.

O mecanismo básico de controle da transcrição opera através de interações com seqüências específicas de uma classe especial de proteínas, os fatores de transcrição. Quando se transfere esse conhecimento em um modelo de dados relacional apropriado, informações sobre duas constituintes básicas aparecem em duas tabelas, Sítios e Fatores (no TRANSFAC chamam-se SITES e FACTORS).

Essas entidades possuem relações NxM, uma vez que vários sítios podem interagir com vários fatores, e todos os fatores conhecidos podem se ligar a mais que apenas um sítio. A tabela de sítios nos fornece a posição de um sítio de regulação em particular, o gene a que este sítio pertence e a espécie biológica da qual esse gene derivou.

Há um campo de texto livre para associar informações desestruturadas, tais como, constantes de dissociação. Existem também atributos para registrar os métodos e seus parâmetros usados para identificar o sítio.

Dentre os parâmetros dos métodos, destaca-se o que indica a qualidade da amostra. Além dos sítios e dos fatores há informações sobre as células e os métodos em entidades relacionais separadas, são as CELLS e os METHODS, porém ligadas por relacionamentos com os sítios. Quando possível ainda há ligações individuais entre as seqüências e o

EMBL data library[Pre].

A tabela de fatores descreve as propriedades de fatores de transcrição em particular: a espécie biológica de que eles derivaram, qualquer especificidade celular, dados sobre o seu tamanho, propriedades estruturais e funcionais. Há dois campos para informação desestruturada que são gravadas como texto. Relacionadas diretamente aos fatores estão as tabelas com os seus respectivos sinônimos e relacionados com os respectivos fatores de interação. Esses fatores de interação são importantes porque os fatores agem como homo ou heterodímeros e a distinção desse tipo de parceria pode determinar o efeito biológico.

3.3 MÉTODO DE MAXIMIZAÇÃO DO RESULTADO ESPERADO

Lawrence e Reilly [CA90] introduziram o método de maximização do resultado esperado (EM) como meio de solucionar a mais simples versão do problema de busca por motifs. O método EM toma como entrada um grupo de seqüências desalinhadas e o comprimento do *motif*. Como saída, este método devolve um modelo probabilístico do *motif* comum. A idéia por trás do método é que cada seqüência do grupo de dados contém um exemplo do *motif*. O método não sabe onde o *motif* aparece dentro da seqüência (onde está localizado o nucleotídeo inicial) em cada exemplo.

Neste método, supõe-se que cada exemplo de *motif* é gerado por uma seqüência aleatória de variáveis independentes, multinomiais, então as freqüências observadas das letras nas colunas são as máximas chances de distribuição das variáveis aleatórias. Como as seqüências originais do conjunto de dados são desalinhadas, as posições relativas das bases dentro das seqüências, *offsets*, não são conhecidas, devendo assim ser estimados. Para fazer isso, o algoritmo EM estima a probabilidade $Poff_{i,j}$ de que o *motif* comece na posição j, da seqüência i, usando os dados e uma suposição inicial da descrição do *motif*. Então o $Poff_{i,j}$ é usado para re-estimar a freqüência da letra l na coluna c do *motif*, $Freq_{l,c}$, para cada letra no alfabeto.

O algoritmo EM alternadamente re-estima $Poff_{i,j}$ e $Freq_{l,c}$ até $Freq_{l,c}$ ficar abaixo de um limiar pré-estabelecido a cada iteração. A notação $Poff_{i,j}$ é usada para referir à matriz de probabilidades. Da mesma forma, $Freq_{l,c}$ refere-se à matriz de freqüência de letra. A descrição do pseudocódigo do algoritmo EM de Lawrence e Reilly é dada a seguir.

Entrada:

```
   Grupo de dados das seqüências;
Comprimento do motif comum;
```

Algoritmo:

```
   Escolher o ponto de partida (freq);
Do {
re-estimar Poff_{i,j} de Freq_{l,c} com regra de Bayes;
re-estimar Freq_{l,c} de Poff_{i,j};
} until (alteração na Freq_{l,c} < LIMIAR );
```

EM começa a partir de uma estimativa de $Freq_{l,c}$ provida pelo usuário ou gerada aleatoriamente. Simultaneamente, o algoritmo EM descobre um modelo de *motif* (a seqüência de variáveis aleatórias multinomiais independentes com parâmetros $Freq_{l,c}$) e estima a probabilidade de cada ponto de partida possível dos exemplos do *motif* nas seqüências no conjunto de dados ($Poff_{i,j}$).

Os algoritmos de maximização de resultados esperados, EM, encontram valores para os parâmetros dado o modelo nos quais a função probabilidade supõe um máximo local[Bai99]. É razoável supor que a solução correta para o problema de caracterizar o

motif comum ocorre no máximo global da função de probabilidade.

A versão original do método EM sofre com algumas limitações. Em primeiro lugar, não é claro como escolher o ponto inicial (um valor inicial de $Freq_{l,c}$) nem quando parar de tentar diferentes pontos iniciais. Isto torna difícil encontrar satisfatoriamente o correto *motif* comum.

Além disso, EM pressupõe que cada seqüência no grupo de dados contém exatamente uma aparição do *motif* comum. Isto significa que seqüências com múltiplas aparições subcontribuirão, e seqüências com nenhuma aparição supercontribuirão para a caracterização do *motif* [Bai99]. Havendo muitas seqüências, com nenhuma aparição do *motif* no conjunto de dados, pode tornar impossível para o EM encontrar o *motif* comum.

Finalmente, EM pressupõe que há somente um *motif* comum na seqüência, e não continua procurando por mais *motifs* após caracterizar o primeiro. Isto faz EM incapaz de encontrar *motifs* com inserções de variáveis no comprimento e incapaz de descobrir múltiplos *motifs* que possam ocorrer nas mesmas ou em diferentes seqüências num grupo de dados[Bai99].

Estas limitações no EM fazem o algoritmo mais suscetível à confusão no conjunto de dados, menos apto a encontrar padrões mais complexos dos dados, e menos útil para explorar grupo de dados que possam conter exemplos de diversos motifs diferentes. Utilizamos a técnica de raciocínio baseado em casos para superar essas limitações.

3.4 FERRAMENTAS ANALISADAS

Existem várias ferramentas computacionais desenvolvidas para localizar supostos sítios de ligação para os fatores de transcrição tais como:

- MEME
- BioProspector
- AlignAce
- MDSCAN
- MatInspector
- MSCAN
- COBIND

3.4.1 MEME

Multiple EM for Motif Elicitation (MEME) é baseada no algoritmo EM (*Expectation Maximization* [CA90]). Esse algoritmo faz uma estimativa inicial do alinhamento e a usa para criar uma matriz de pontos(PSSM) que representa a distribuição das bases no *motif*. Em seguida a matriz é comparada a cada seqüência e então atualizada para maximizar o alinhamento entre a matriz e as seqüências. Este passo é repetido iterativamente até a matriz convergir. MEME usa um algoritmo gradiente descendente e pode ficar preso em máximos locais. Várias heurísticas são empregadas para tentar atenuar esse problema, mas não é possível garantir o retorno do melhor *motif*. A vantagem do MEME é que ele pode encontrar vários *motifs* e não é necessário que toda seqüência o contenha. O MEME requer ainda que o usuário estime inicialmente o tamanho do *motif*.

O algoritmo MEME é uma versão modificada do método MEME[CA90]. O algoritmo EM é executado repetidamente com diferentes pontos de partida. Os pontos de partida são derivados de subseqüências atuais que ocorrem no conjunto de dados de entrada. EM é executado em poucas iterações, não para convergência de cada ponto de partida, mas para poupar tempo. Cada execução do EM produz um modelo probabilístico de um possível *motif* comum. O ponto inicial que produz o modelo com maior probabilidade é

escolhido e EM é executado para convergência a partir deste ponto. O modelo do *motif* comum é, desse modo, descoberto e impresso. Finalmente, todas as aparições do *motif* comum no grupo de dados são eliminados. Todo o processo é repetido para descobrir mais motifs comuns.

Algoritmo MEME

Input:

```
   dataset of sequences;
LSITE (length of shared motifs);
PASSES (number of distinct shared motifs to find);
NITER (number of iterations to run EM);
MAXP (number appearances of each shared motif expected in dataset);

   Algorithm:

   for motif = 1 to PASSES {

         for each subsequence in dataset {
run modified EM for NITER iterations with starting point derived from
this subsequence;
}
}
```

3.4.2 BioProspector e AlignAce

BioProspector e AlignAce são especializações da amostragem de Gibbs. Amostragem de Gibbs é um algoritmo capaz de gerar amostras aleatórias através de um processo de remontagem das variáveis para um parâmetro θ [dUMdFdLR+02], que nesse caso é o tamanho do *motif*. Assim como em outras ferramentas esse tamanho deve ser previamente conhecido.

Com pequenos ajustes para busca de sítios de ligação, a amostragem de Gibbs é uma técnica que funciona da seguinte forma: dado um conjunto que se acredita conter o *motif*, retira-se uma seqüência aleatória desse conjunto e alinham-se as seqüências restantes levando em consideração a probabilidade para cada posição. Em seguida, otimiza-se o alinhamento ajustando a seqüência para frente ou para trás. Colocá-se o *motif* de volta e uma nova seqüência é escolhida observando-se uma matriz de pesos (PSSM). Pesos para cada segmentos são atribuídos usando o *motif*. BioProspector incorpora uma série de melhorias ao algoritmo de Gibbs, tais como não precisar que toda seqüência contenha o *motif* ou o tratamento a múltiplas cópias do mesmo *motif* dentro de uma seqüência. O BioProspector também pode procurar por *motifs* separados em dois blocos, desde que haja uma pequena seqüência entre eles que não interfira no *motif*. Para que o BioProspector funcione, é importante conhecer previamente o modelo ideal de formação para o motif, pois uma cadeia de Markov(HMM) é criada baseada nesse modelo. É importante ressaltar que o BioProspector pode prender-se em máximos locais e é uma ferramenta relativamente lenta[dUMdFdLR+02].

A amostragem de Gibbs é um método eficaz para encontrar sinais delicados e complexos, o que faz essas ferramentas muito sensíveis a fatores de transcrição e sítios de ligação.

3.4.3 MDscan

Diferente do BioProspector, o MDScan usa um algoritmo determinístico que sempre converge para o mesmo resultado em um mesmo conjunto de dados. O MDscan funciona melhor quando as seqüências puderem ser separadas em dois grupos: o grupo que contém o motif e o que não contém. O MDScan começa procurando n-meros (dímeros, trímeros,...). Em seguida, enumera cada semente (n-mero não redundante) e por sua vez essas sementes são usadas para criar uma matriz de pesos. Cada matriz é avaliada e o processo se repete até que se estabeleça um *motif*. Esse algoritmo é muito mais rápido $O(x^2)$ [LBL01b] que a amostragem de Gibbs e pode ser usado em genomas inteiros.

3.4.4 MatInspector

Essa ferramenta difere do MDScan, BioProspector e AlignAce, porque procura por sinais de fatores previamente conhecidos para encontrar novos *motifs* para esses fatores de transcrição. As outras três ferramentas são usadas para encontrar sítios em seqüências, enquanto MatInspector serve para encontrar ocorrências de um dado *motif* numa seqüência que pode ou não conter os sítios ligantes. A vantagem de usar uma ferramenta como esta é que bancos de dados biológicos como o TRANSFAC representa o sítio de ligação como uma matriz de pesos (PSSM). Isso permite ao MatInspector encontrar rapidamente os sítios de ligação. Entretanto, essa abordagem de matriz de pesos dificulta a busca por sítios de ligação de fatores de transcrição desconhecidos.

3.4.5 MSCAN

MSCAN não precisa de treinamento para estimar a significância de agrupamentos de supostos sítios de ligação de fatores de transcrição. Este algoritmo recebe como entrada um conjunto de modelos de sítios de ligação dos fatores de transcrição passados através de matrizes de pesos (PSSM), um limiar e uma seqüência de genoma. O agrupamento mais significante de supostos sítios de ligação dentro de uma janela de seqüência é iden-

tificado e aquelas janelas são relatadas se o nível de significância ficar abaixo do limiar estabelecido. Para se estabelecer o limiar é preciso conhecer o modelo previamente. Este conhecimento prévio não é simples de ser obtido.

Os métodos de detecção de fatores de transcrição em seqüências de genomas devem diferenciar pequenos segmentos contendo alta concentração de *motifs* dentro de uma janela de seqüência. O algoritmo MSCAN avalia a significância estatística combinada de conjuntos de sítios de ligação em potencial e usa um valor limiar previamente definido para fornecer as saídas.

No MSCAN há três itens para se considerar: medir a significância de um padrão individual com um motif, calculá-la combinada para qualquer conjunto de supostos sítios de ligação e determinar o conjunto ótimo de *motifs*.

3.4.6 COBIND

Nenhuma das ferramentas computacionais consideram o fato de haver efeitos combinatoriais. Entretanto, o COBIND[GS01] procura por supostos sítios de dois fatores cooperativamente ligados. O algoritmo maximiza a vizinhança de dois sítios de ligação. Os sítios são representados por duas PSSM que são computadas com uma função objetivo derivada da energia de ligação termodinâmica. O problema do COBIND é checar apenas por dois sítios, mas o controle combinatorial pode envolver vários fatores de transcrição. A distância também é um fator desfavorável do COBIND. Segundo este algoritmo, a probabilidade de uma seqüência fazer parte do sítio diminui com a distância e isso não é totalmente uma verdade biológica[ABJ+99]. Há sítios de ligação que estão linearmente distantes e o COBIND não consegue tratar. O COBIND usa a função termodinâmica tendo a vantagem de quantificar a afinidade das ligações.

3.5 PESQUISAS RELACIONADAS

Não somos os primeiros a modelar dependências entre seqüências biológicas de motifs. Agarwal e Bafna[AB98] sugeriram o modelo de rede de árvore e discutiram algoritmos para aprendê-los. Friedman e outros[BEFK03] também fizeram experimentos com dependência e concluíram vantagens nesses modelos. Há outros trabalhos sendo desenvolvidos nos últimos anos e podemos destacar os seguintes.

Em 1999 a equipe de Wagner estudou uma forma de caracterizar o posicionamento dos fatores de transcrição dentro da região regulatória para procurar por relações que possam sugerir quais proteínas podem interagir [Wag99].

Na maioria dos trabalhos olha-se apenas para os segmentos das estruturas dos fatores de transcrição que estão em contato com o DNA. Este fato permite-nos estudar como o fator de transcrição interage com o DNA, mas não se sabe a conformação do resto do complexo [BT99a]. Uma das principais razões pela qual não são levadas em consideração as naturezas combinatórias dos fatores de transcrição é que esse processo não é biologicamente bem compreendido ainda.

Um trabalho foi feito por Pipel[P+01] sobre o efeito combinatorial (dependência). O resultado de experimentos de expressão gênica sob diferentes posições foi usado para gerar mapas de associação de *motifs* baseados nos padrões de expressão. Nessas experiências foram identificados várias associações transcricionais ratificando a ação combinada dos fatores[P+01].

Para identificar combinações de *motifs* que controlam a expressão gênica, Pipel estabeleceu um banco com motifs conhecidos e com supostos motifs de regulação, em seguida identificou cada *motif* e seus promotores. Para cada *motif* ou combinação, Pipel calculou a coerência, similaridade e perfis de todos os genes que continham *motifs*. Uma abordagem estatística foi utilizada para caracterizar a sinergia entre motifs e identificar

combinações funcionais.

Davidson, em 2001, fez promissores estudos sobre o relacionamento combinatorial de fatores de transcrição usando o fato de que eles se agrupam em módulos[Dav01]. Alguns estudos usam esse conceito para ajudar a localizar sítios de ligação como pode ser visto no desenvolvimento da *Drosophila melanogaster*[B+02, M+02].

Outros grupos de cientistas analisaram o TRANSFAC para encontrar pares de fatores de transcrição conhecidos para interagir e medir a preferência para localizar, à distância específica, e predizer novos sítios de ligação em genoma humano[HL02].

3.6 EXPERIMENTO DE BARASH E FRIEDMAN

Um dos trabalhos mais recentes é o de Friedman e Barash, nos seus trabalhos eles falam sobre duas questões fundamentais: dependências entre posições são relevantes nos dados biológicos e aprender modelos de dependências de posição pode melhorar a descoberta de novos sítios de ligação[BEFK03].

O objetivo de Barash et. al. é testar se a modelagem das dependências melhora o resultado nas tarefas computacionais de perceber sítios de ligação e de descobrir motifs. O trabalho sugere existência de dependências entre posições, pelo menos para alguns fatores de transcrição. A principal questão técnica é como enfraquecer a suposição de independência. Com este propósito foi modelada a distribuição conjunta de todas as posições. Entretanto, a representação direta requer um grande numero de parâmetros (exponencial no caso do tamanho do *motif*). A escolha da representação leva a uma questão difícil: modelos mais baratos não podem representar dependências complexas, mas têm representação sucinta e podem aprender muito com apenas alguns exemplos. Modelos mais caros podem representar dependências complexas, mas envolvem muitos parâmetros e requerem um grande numero de exemplos para aprendizado.

Neste trabalho foram examinados modelos que ponderam esses fatos. Estes modelos foram baseados em redes Bayesianas. Barash descreveu procedimentos que aprendem estas representações a partir de dados. Nos sítios verificados biologicamente, o conjunto flexível de modelos generaliza melhor que o modelo PSSM. De acordo com testes empíricos, os modelos aprendidos são mais precisos ao predizerem os supostos sítios de ligação no genoma da *S. cerevisiae*.

Barash modelou os *motifs* representando os sítios de ligações dos fatores de transcrição com a intenção de analisar os termos comuns entre os diferentes sítios. A maneira encontrada para fazer isso foi representar a distribuição probabilística das seqüências que designam altas probabilidades às seqüências que provavelmente serão encontradas nos sítios de ligação. Assumiu-se que os sítios de ligação são de tamanho K. Logo, quer-se representar a distribuição de probabilidade sobre todos os 4^K possíveis elementos que podem aparecer em um sítio de ligação. Formalmente, precisa-se de uma distribuição $P(X_1, ..., X_k)$ onde X_i é uma variável aleatória que representa os nucleotídeos na posição i do k-ésimo elemento. Uma representação de tal distribuição simplesmente lista as probabilidades de todos os 4^K designados. Esta representação é computacionalmente pouco viável para. Assim, estamos interessados em maneiras mais sucintas de representar tais distribuições. Para tentar encontrar uma representação ideal, Barash tentou algumas técnicas como segue em 3.6.1, 3.6.2, 3.6.3, 3.6.4 e 3.6.5 conforme ilustra também a Figura 3.2.

3.6.1 Matriz de valores de posição especifica (PSSM)

Como podemos observar em outras abordagens, uma maneira comum de se representar um sítio de ligação é assumir que o nucleotídeo em uma posição é independente dos ncleotídeos de todas as outras posições. Esta suposição implica que:

$$P(X_i, ..., X_k) = \Pi_{t=1}^{K} P(X_t)$$

Onde $P(X_i)$ é o limiar de probabilidade de cada nucleotídeo X_i na distribuição.

3.6.2 Mistura de PSSM

Supondo que os fatores de transcrição podem fazer diversos tipos de ligação, cada fator com algumas preferências diferentes, pode-se ligar a qualquer seqüência que encaixe em uma dessas configurações. Para modelar este caso, Barash assumiu que existe uma variável aleatória T adicional, não observada, que descreve o tipo de ligação. Tem-se uma probabilidade primária $P(T)$, que ele afirma ser uniforme sobre o tipo de ligação e assumiu que para cada tipo as posições são independentes, assim como no caso da PSSM. Isto requer uma distribuição $P(X_i \mid t)$ sobre o nucleotídeo na posição i dado o valor t de T. A probabilidade conjunta das posições observadas requer a soma de todos os valores possíveis de T. Isto significa que essa distribuição é uma mistura de PSSMs cuja probabilidade da mistura é dada por $P(T)$.

$$P(X_i, ..., X_k) = \sum_{t=1}^{C} [P(t)\Pi_{t=1}^{K} P(X_t \mid t)]$$

A Figura 3.2 mostra exemplos de modelos de diferentes redes Bayesianas para um *motif* com 5 posições. Para cada modelo, há um exemplo de uma estrutura e de uma representação correspondente da distribuição conjunta.

Os modelos híbridos têm diversos benefícios para representação e interpretação semântica. Um dos principais benefícios é que o número de parâmetros é pequeno: $C-1$ parâmetros para $P(T)$, e $3KC$ para as probabilidades condicionais $P(X_i \mid T)$. Embora os nucleotídeos sejam dependentes quando T é desconhecida, eles são independentes quando T é observada. Desta forma, T desempenha o papel de um mecanismo biológico escondido que apresenta as posições independentes. Entender a interação entre o mecanismo escondido T e os nucleotídeos em cada posição pode oferecer idéias a respeito das

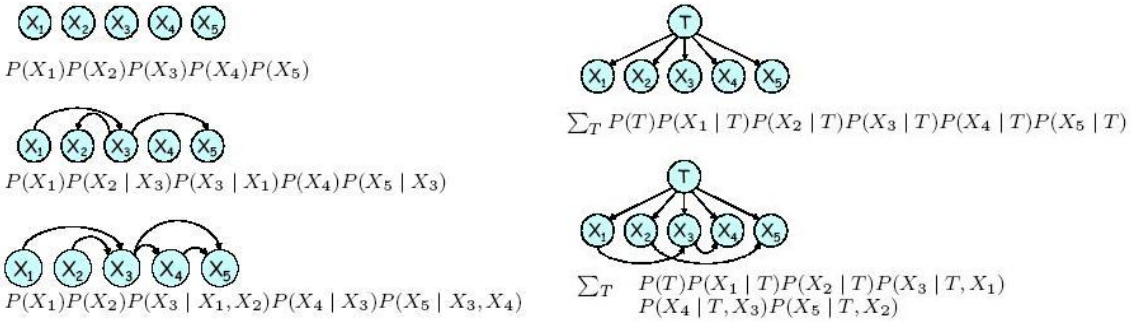


Figura 3.2. Modelos usados por Barash et. al.

interações física da proteína de DNA.

3.6.3 Redes Bayesianas

Misturas de PSSM capturam dependências entre todas as posições via a variável T. Uma abordagem alternativa para descrever dependências é considerar cada posição dependente das outras. Por exemplo, um nucleotídeo em particular na posição 1 pode causar uma mudança da configuração de um radical de um aminoácido em particular. Isto, por outro lado, afeta a configuração de outros aminoácidos nos sítios de ligação e pode ter um efeito sobre a preferência de ligação na posição 3.

Uma representação projetada a capturar as dependências locais é a rede Bayesiana. Como se pode ver, em maiores detalhes, na seção 4.2, usamos um grafo direcionado acíclico G para representar as dependências. Os vértices de G correspondem a variáveis aleatórias $X_1 ... X_k$ e a parametrização que descreve a distribuição condicional para cada variável dado seus pais imediatos em G.

Em termos de independência condicional, nas redes Bayesianas, cada posição X_i é independente de seus não-descendentes no grafo G. Em geral, quanto mais arestas temos em G, mais complexa são as dependências entre as posições. A rede mais simples não

tem arestas, como a primeira representação da Figura 3.2. O número de parâmetros nas redes depende do número de arestas[Fri98].

3.6.4 Redes de Árvores Bayesianas

Uma classe derivada da rede Bayesiana é a de redes de árvores Bayesianas. Nesse modelo cada posição tem no máximo um pai, tornando G uma floresta. Redes de árvores também generalizam as cadeias de Markov de primeira ordem (onde G é $X_1...X_k$). Elas provêem uma linguagem flexível para modelar dependências. Outro benefício importante desta classe de modelos é que existem algoritmos eficientes para aprender a melhor estrutura de árvore[BEFK03].

3.6.5 Combinação de Árvores

Em alguns casos, uma rede estruturada em árvore pode ser muito limitada. Uma abordagem possível de enriquecer a representação é combinar os benefícios de uma estrutura de árvore com a diversidade adicionada por um mecanismo escondido. Isto nos leva a uma extensão natural, similar a mistura de PSSM que é uma mistura de árvores. Cada X_i tem como pais a variável escondida T e no máximo outro nucleotídeo. Intuitivamente, a variável não observada T ganha a habilidade do modelo de árvore para modelar dependências adicionais apenas multiplicando o numero de parâmetros por um fator C. Uma vantagem importante da composição de árvores é que existem algoritmos eficientes para se aprender a melhor estrutura[BEFK03].

3.6.6 Aprendizado da Estrutura

Além de estimar parâmetros, é necessário aprender a dependência da estrutura do grafo G, isto é, que arestas incluir. Para realizar o aprendizado da estrutura foram con-

siderados modelos mais ricos (os que têm mais arestas) e podem alcançar similaridades mais altas. Desse modo, há o risco de *overfitting*, aprendendo um modelo aparentemente bom nos dados de treinamento, mas executando mal em novas instâncias. Por isso, ao invés de maximizar a função de probabilidade maximizou-se o valor estatístico baseado em considerações Bayesianas. Este valor pode ser visto como probabilidade penalizada por um termo que leva em conta diferenças na complexidade dos modelos.

Em modelos que não há variável escondida T, encontrar o melhor grafo é um problema de otimização. Para redes de árvores, este problema pode ser reduzido ao algoritmo de maximizar o peso da floresta (*maximum weighted forest*) e resolvido eficientemente. Em redes Bayesianas em geral, este problema é intratável, por isso tentou-se várias heurísticas de busca.

Em modelos onde uma variável escondida T está presente, a situação é mais complicada. Primeiro, deve-se decidir a cardinalidade de T. Além disso, o valor da estrutura se torna intratável, por isso deve-se recorrer a uma aproximação. Barash usou a aproximação de Cheeseman - Stutz do valor de BDeu[CH97] para escolher a cardinalidade de T e, em seguida, estimou o valor de cada cardinalidade. Para uma combinação de modelos de árvores, também encaramos o problema de aprendizado de estrutura.

O método de aprendizado descrito por Barash e Friedman é similar, em sua arquitetura geral, a outros métodos baseados em EM, tais como o MEME[Bai99]. O mesmo diferencia-se do MEME em vários aspectos. Primeiramente, conectamos uma rede Bayesiana geral, que aprende um modelo dado uma representação (árvores, combinação de PSSM, etc.). Segundo, o processo de aprendizado considera observações parciais acerca de cada seqüência. Terceiro, o método utiliza um modelo fixo e previamente conhecido durante o aprendizado e não como um passo de pós processamento. Finalmente, como um ponto inicial, foi usado um método de busca combinatorial diferente. Este método de

busca tenta encontrar soluções iniciais que melhor distinguem as seqüências supostamente reguladas a partir daquelas supostamente não reguladas.

3.6.7 Avaliação dos Métodos

Para se avaliar os métodos, foram construídos vários conjuntos de dados, cada um consistindo de promotores positivos (seqüências em que se encontram *motifs* de sítios de ligação), e negativos. Para simular o problema biológico em questão mais fielmente, os dados foram amostrados a partir de modelos treinados em sítios de ligação conhecidos do fator de transcrição de Human LUN da base de dados TRANSFAC (V$LUN1_01). Em cada configuração, criaram-se dois conjuntos paralelos, um amostrado a partir de uma rede de árvores que contém posições de dependência, e o outro a partir de um modelo de PSSM. Para efeturamos testes comparáveis, utilizamos os mesmos dados que Friedman e Barash usaram em seus testes.

As seqüências promotoras foram amostradas a partir de um modelo de distribuição de Markov[2] de ordem 3, treinado em regiões promotoras de humanos. Para simular ruídos, os grupos de dados foram contaminados com outro grupo de promotores falsos positivos, em que nenhum motif verdadeiro foi implantado.

Assim como Barash, testamos nosso método em uma variedade de configurações, mudando o tamanho dos promotores (de 250 a 500bp), e a composição de promotores positivos de 100 positivos verdadeiros sem falsos, ate 25 positivos verdadeiros com 75 falsos. Os conjuntos de dados sintéticos estão disponíveis em [BEFK]. Os resultados podem ser vistos nas tabelas 6.1, 6.2, 6.3 e 6.4.

Pode-se notar que todos os métodos trazem resultados similares em dados gerados

[2]O modelo escondido de Markov é um conjunto finito de estados em que cada um deles está asssociado a uma distribuição de probabilidade, geralmente multidimensional. As transições entre os estados são governadas por um conjunto de probabilidades chamadas de probabilidades de transição[Ján].

a partir de uma PSSM. Os dados gerados a partir da rede de árvores (tree network), mostram a diferença entre o desempenho do aprendido a partir de uma rede de árvores e o modelo PSSM. O modelo de combinação de PSSM, que são incapazes de modelar as dependências, mostram o pior desempenho em relação aos demais. O melhor desempenho é o tree network.

Na prática, pretende-se obter seqüências que contenham o motif aprendido. Para fazer isso, selecionam-se seqüências em que o valor calculado está abaixo de um limiar pré-especificado. Nos experimentos, usou-se o limiar de 0.01. A razão para este limiar estrito é que em exames de grandes genomas, milhares de seqüências, a maioria não contém o fator de transcrição em questão. Ao se escolher níveis de significância estritos, controlam-se o numero de falsos positivos entre as seqüências obtidas.

3.7 CONSIDERAÇÕES FINAIS

Atualmente, há uma grande quantidade de técnicas para predizer a função de proteínas. Muitas delas surgiram ou tiveram seu desempenho melhorado com o aumento da quantidade de dados disponíveis na sociedade. Um dos bancos de dados mais utilizado pelos cientistas é o TRANSFAC.

As ferramentas atualmente desenvolvidas estão melhores a cada dia. Uma das principais razões para este fato é a combinação de preditores que muitas vezes combinam vários métodos diferentes para conseguir tal melhora.

Neste capítulo tivemos a oportunidade de conhecer melhor alguns métodos e ferramentas disponíveis atualmente. Dentre os métodos podemos destacar o método de maximização do resultado esperado e as redes Bayesianas.

Tivemos ainda a oportunidade de analisar brevemente algumas das ferramentas atu-

almente difundidas no meio acadêmico, dentre elas o MEME, BioProspector, AlignAce, MDScan, MatInspector, MScan e COBIND. Elaboramos um breve resumo sobre o trabalho de Barash e Friedman de onde retiramos importantes aspectos sobre a dependência entre as posições e as idéias sobre o uso de redes Bayesianas para modelar tais dependências.

No próximo cápítulo falaremos sobre os algoritmos utilizados no desenvolvimento da ferramenta Jnom.

CAPÍTULO 4

A FERRAMENTA JNOM

4.1 INTRODUÇÃO

Para se desenvolver a ferramenta Jnom, objeto deste trabalho, considerou-se práticas adequadamente adaptadas dentre meta modelos de desenvolvimento de software, principalmente o RUP(*Rational Unified Process*)[Inc] e o XP(*Extreme Programming*)[Bec]. Isto significa que, para o desenvolvimento da Jnom, desenvolvemos um processo alternativo baseados em práticas sugeridas nesses dois meta modelos de processo.

Conseguimos garantir a produção de uma ferramenta de alta qualidade[1] que atende às necessidades dos usuários dentro de um prazo estabelecido. Mais informações sobre o processo de desenvolvido especialmente para a criação da Jnom pode ser visto no apêndice D.

Dentre as várias técnicas utilizadas destacamos Redes Bayesianas, PSSM[2] e raciocínio baseado em caso(RBC). Optamos por estas tecnologias e métodos devido a suas características que veremos em detalhes a seguir.

[1] Qualidade segundo a International Organization for standardization - ISO - é definida como o grau de conformidade com os requisitos[iso].

[2] PSSM é a abreviação da sigla em inglês *Position Specific Score Matrix*.

4.2 REDES BAYESIANAS

Uma forma alternativa e recentemente utilizada para representação e análise de dados são as redes Bayesianas. Vamos mostrar algumas características do uso dessas redes.

Redes Bayesianas podem naturalmente tratar conjuntos incompletos de dados. Por exemplo, considere um problema de classificação onde duas das variáveis de entrada possuem forte anticorrelação. Esta correlação não é um problema para técnicas de aprendizado supervisionado se sempre fornecermos todas as entradas. Quando não fornecemos uma das entradas, a maioria dos algoritmos que usam essa técnica produz uma saída imprecisa, pois eles não codificam a correlação entre as variáveis de entrada. Já as redes Bayesianas oferecem uma forma natural de codificar essas dependências[FGG97].

Essas redes permitem aprendizado de relações causais. Aprender sobre relações causais é importante pois o processo é útil quando estamos tentando entender o domínio do problema. Além disso, o conhecimento de relações causais nos permite fazer predições e inferências na presença de intervenções. Por exemplo, um biólogo pode querer saber se um dado aminoácido influencia ou não em um sítio de ligação. Para responder essa pergunta o biólogo pode determinar se esse aminoácido influencia no sítio de ligação e em com qual intensidade. As redes Bayesianas podem responder questões como essa mesmo sem precisar de nenhuma experiência prática sobre o efeito do aminoácido no *motif*.

Redes Bayesianas em conjunto com técnicas estatísticas facilitam a combinação entre o domínio do conhecimento e os dados. Não se deve esquecer que o conhecimento de especialista é muito útil quando os dados são escassos ou caros. As redes Bayesianas implementam as relações causais com probabilidade e na definição dessas probabilidades é muito útil um conhecimento prévio. Esse conhecimento pode ser combinado com os dados através de técnicas estatísticas[APR99].

Métodos Bayesianos oferecem eficientes formas para evitar *overffiting* por isso não

há necessidade de separar conjuntos de teste e validação como nas redes neurais. Na abordagem Bayesiana, os modelos podem ser ajustados de forma a usar todos os dados disponíveis para treinamento[Fri98].

As redes Bayesianas podem ser implementadas como misturas de PSSM que capturam dependências globais entre todas as posições via a variável aleatória T independente. Uma abordagem alternativa para descrever dependências é considerar como cada posição depende das outras. Por exemplo, um nucleotídeo na posição 1 pode causar uma mudança da configuração de um radical de um aminoácido em particular. Isto, por outro lado, afeta a configuração de outros aminoácidos nos sítios de ligação e pode ter um efeito sobre a preferência de ligação na posição 3.

Em uma representação mais elaborada, usa-se um grafo direcionado acíclico G para representar as dependências. Os vértices de G correspondem às variáveis aleatórias $X_1 ... X_k$ e a parametrização que descreve a distribuição condicional para cada variável dado seus pais imediatos em G correspondem às arestas.

A distribuição probabilística conjunta correspondente decompõe-se na forma de produto:

$$P(X_i, ..., X_k) = \times P(X_t \mid P(A_i^G),$$

onde $P(A_i^G)$ é o conjunto de pais de X_i em G.

Cada posição X_i é independente de seus não-descendentes em G. Em geral, quanto mais arestas têm em G, mais complexa são as dependências entre as posições. A rede mais simples é aquela que não tem arestas e pode ser vista como uma PSSM. Por isso, diz-se que a PSSM é um caso particular das redes Bayesianas.

4.3 MATRIZ DE PESOS DE POSIÇÕES ESPECÍFICAS

A matriz de pesos de posições específicas, ou simplesmente PSSM, é construída levando-se em consideração a freqüência de cada resíduo (nucleotídeo) em uma posição específica de um alinhamento múltiplo. A PSSM pode ser computada da seguinte forma: a freqüência de todo resíduo em cada posição é comparada com a probabilidade da ocorrência daquele resíduo em uma seqüência aleatória.

O Placar (*Score*) é derivado da taxa das freqüências observadas para as esperadas. Mais precisamente o logaritmo dessa taxa é tomado e referido como sendo a taxa da vizinhança (*loglikelihood ratio*).

$$Score_{i,j} = log(\frac{f'_{i,j}}{q_i})$$

O $Score_{i,j}$ é o score do resíduo i na posição j, $f'_{i,j}$ é a freqüência relativa do resíduo i na posição j e q_i é freqüência relativa esperada do resíduo i em uma seqüência aleatória.

Pode-se supor que PSSM é uma boa opção para regiões pequenas, estáveis e com tamanho fixo. Esse algoritmo é relativamente fácil de implementar. Os pesos podem ser interpretados baseados em teorias estatísticas. Há limitações em relação a inserções e remoções. Seqüências longas não são bem tratadas por esse método[Bai99].

4.3.1 Limitações da Matriz

Para construir esta matriz, considera-se geralmente a quantidade relativa de cada nucleotídeo em cada posição no sítio. Isso significa que se em uma dada posição do sítio a Adenina aparece em torno de 70% das vezes e as outras bases apenas 30%. Nesse caso a matriz de freqüência dará com 70% certeza que nessa posição isoladamente encontraremos Adenina nos *motifs* do organismo em questão. Note que a matriz de

freqüência erraria cerca de 30% dos *motifs* se olhasse apenas para essa posição. O problema aumenta se propagarmos esse erro para todo o sítio que pode ter até cerca de 40 bases de comprimento perdendo assim muita precisão [ZDSS99]. Há algumas formas de minimizar esse problema uma delas é explicitada a seguir.

Um meio para contornar esse problema é observar a combinação dos nucleotídeos na seqüência devido a ação combinada dos fatores de transcrição, isto é, não vamos olhar para cada posição isoladamente, mas olharemos sempre para um contexto (combinação). Por exemplo: uma combinação freqüente de bases que representa uma região promotora em bactérias é TAT (Timina, Adenina, Timina)[Lew97]. Nesse caso devemos olhar sempre para as três bases juntas, não para cada uma delas isoladamente. Para aumentar o espaço de busca e tentar melhorar os resultados desejados, espaços (gaps) podem ser introduzidos na combinação, tornando a busca mais elaborada, pois a seqüência TxxAT poderia ser uma combinação procurada onde x pode ser qualquer outra base, nesse exemplo há um gap de tamanho três.

Houve algumas abordagens em trabalhos recentes que conseguiram uma melhora na precisão usando redes Bayesianas[BEFK03]. Essas redes visavam capturar dependência entre as posições do motif. Este trabalho propõe uma forma alternativa de implementar tais redes e capturar essas dependências de forma espacial usando uma fusão de idéias entre a técnica do raciocínio baseado em casos e as redes Bayesianas. Essa forma alternativa custa menos computacionalmente e empiricamente se mostrou mais precisa que algumas implementações de caráter mais estatístico das redes.

4.4 RACIOCÍNIO BASEADO EM CASOS

Raciocínio Baseado em Casos[Kol92, vWvW03], ou apenas RBC, é um método de solução de problemas usando adaptações de soluções anteriores e similares a estes problemas. Sistemas baseados em conhecimento podem adaptar soluções conhecidas para

encontrar novas soluções.

Neste contexto, o RBC pode funcionar inclusive como um modelo cognitivo para se entender alguns aspectos do pensamento e comportamento humano, além de ser uma tecnologia para construir sistemas computacionais inteligentes e resolver problemas reais[vWvW03].

Tais sistemas podem usar casos conhecidos para explicar novas situações e criticar novas soluções. Esta abordagem usa raciocínio baseado em conhecimento previamente adquirido para interpretar uma nova situação ou criar uma solução apropriada para um problema desconhecido[Bar01, Kol92].

O raciocínio baseado em casos também é usado extensivamente para raciocínio de senso comum no dia-a-dia. Por exemplo, quando planejamos nossas atividades, nos recordamos do que funcionou e do que falhou, e usamos isso para criar nossos planos. Quando montado um sistema, cada diferente solução ou interpretação do problema, é um novo caso. Para este trabalho, cada *motif* é considerado um caso.

Para a construção de um Sistema Baseado em Conhecimento, necessitamos de uma forma coerente de representação do conhecimento. O conhecimento é representado na forma de caso. Um caso é a principal parte do conhecimento nos sistemas RBC.

A representação dos casos é uma tarefa complexa e importante para o sucesso de sistemas RBC. Em nosso trabalho utilizamos a matriz de freqüência para armazenar os casos.

A escolha de um caso semelhante para a solução de problemas exige o uso de uma medida que determine a proximidade da suposta solução em relação ao caso em

análise[Bar01]. Essa medida é chamada grau de semelhança.

Inúmeras funções matemáticas para cálculo de distâncias em espaço métrico podem ser usadas na avaliação de semelhança entre os casos, dentre elas destacamos a distância de Hamming ou Manhatan e a Distância Euclidiana[Bar01]. A distância de Hamming utiliza somente a soma ponderada dos módulos da diferença ao invés da raiz quadrada da soma quadrática das distâncias como na Euclidiana. Isto lhe confere um menor custo computacional aliado à vantagem de que os valores extremos (outliers) não são amplificados como no caso do modelo Euclidiano onde as diferenças são elevadas ao quadrado.

Jnom considera o mesmo peso para todas as unidades da seqüência, mas isso pode ser alterado para se considerar dependências mais fortes entre posições específicas. Neste caso, a distância de Hamming entre duas seqüências de nucleotídeos x, y pode ser dada por:

$$dH(x, y) = \sum \mid x_i - y_i \mid$$

Onde:

$dH(x, y)$ é a distância de Hamming entre as seqüências x e y.
x_i é um caracter dentro da seqüência x.
y_i é um caracter dentro da seqüência y.

Essa expressão devolve como resultado a soma das posições coincidentes entre duas seqüências. Jnom utiliza essa soma para atribuir o grau de semelhança. Pesos entre posições específicas podem ser considerados para modelar combinações entre posições específicas apenas adicionado termos ao lado direito da igualdade. Esses pesos podem ser implementados através das redes Bayesianas, como fez Friedman[BEFK03], ou pela própria distância de Hamming que possui artifícios para isso[Bar01].

4.5 FUSÃO DE TÉCNICAS NA JNOM

Suponha que queremos aprender modelos de *motif* a partir de dados. Assumimos que nossa entrada é um conjunto de sítios de ligação do fator de transcrição. Nossa tarefa é aprender um modelo que captura os aspectos comuns destas seqüências. Isto é uma instância do problema bem conhecido de treinar as redes Bayesianas a partir de dados.

Para persistir o conhecimento, usamos a matriz de freqüência(PSSM) como caso particular das redes Bayesianas. Esta matriz armazena o conhecimento adquirido através da apresentação de padrões (casos). As dependências são encontradas através da semelhança obtida com a distância de Hamming [Bar01, Kol92]. Após os vários casos serem observados e suas dependências devidamente absorvidas pela matriz através das observações, as buscas por novos motifs podem começar.

A matriz de freqüência para motifs de tamanho K possui K colunas onde cada coluna representa uma posição no *motif* e 4 linhas representando as quatro diferentes bases (A, T, C, G). Dizemos que a posição $M_{i,j}$ representa a freqüência da base da linha i na posição j do *motif*. O aprendizado é feito contando-se a freqüência relativa de cada base em cada *motif* de treinamento que é apresentado a matriz. Vamos ilustrar com o seguinte pseudo-código:

```
i)   função aprender(motif){
ii)      quantidadeDeMotifs++;
iii)     for (de i = 0; enquanto i < motif.length(); i++) {
iv)          matrix[motif.baseAt(i)][i]+= 1/quantidadeDeMotifs;
v)       }
vi)  }
```

	T	C	C	C	A	G	C	A	C	T	T	T	G	G	G	A	G
A	3,0%	4,0%	5,0%	2,0%	84,0%	5,0%	3,0%	63,0%	23,0%	1,0%	1,0%	13,0%	13,0%	18,0%	7,0%	81,0%	9,0%
T	89,0%	2,0%	7,0%	5,0%	5,0%	8,0%	4,0%	31,0%	7,0%	71,0%	77,0%	65,0%	4,0%	1,0%	2,0%	5,0%	6,0%
C	2,0%	87,0%	87,0%	88,0%	5,0%	6,0%	85,0%	0,0%	65,0%	24,0%	10,0%	9,0%	5,0%	7,0%	2,0%	2,0%	9,0%
G	6,0%	7,0%	1,0%	5,0%	6,0%	81,0%	8,0%	6,0%	5,0%	4,0%	12,0%	13,0%	78,0%	74,0%	89,0%	12,0%	76,0%

Figura 4.1. Um exemplo de matriz de freqüência para um *motif* com 17 bases. As bases no topo das colunas indicam qual base é mais freqüente naquela posição do *motif*.

Depois de apresentarmos alguns *motifs*, a matriz deve ter a configuração semelhante a matriz da Figura 4.1

Se olharmos novamente para a fórmula geral que implementa as redes Bayesianas e tentam modelar dependência, podemos observar que cada rede possui dependência fixa. Isto significa que a rede não consegue aprender as dependências, elas devem ser modeladas. Se as seqüências possuem dependências entre as posições x e y, a rede que modela essas seqüências sempre considerará essa dependência, quer ela exista ou não nas seqüências de teste.

Fórmula Geral das Redes Bayesianas com Independência entre Posições.

$$P(X_i, ..., X_k) = \times P(X_t \mid P(A_i^G))$$

onde:

$P(A_i^G)$ é o conjunto de pais de X_i em G.

Artifícios matemáticos e funções estatísticas podem ser utilizadas no lado direito da igualdade para modelar as dependências. O objetivo é fazer com que o algoritmo aprenda as dependências sem a necessidade do modelo prévio. A abordagem escolhida para essa absorção de conhecimento a partir dos dados é a busca por semelhança espacial, pois é relativamente fácil de observar empiricamente que se em uma seqüência $s_1 = x_1, ..., x_n$ a posição x_k depende de outra posição $x_{(k+z)}$ essa dependência aparece visualmente quando

observamos o comportamento de x_k ou de $x_{(k+z)}$.

A semelhança espacial pode ser obtida com a fusão das equações de Hamming com fórmula geral das redes Bayesianas. Com a fusão destes dois conceitos, obtivemos uma equação que permite-nos informar dependências específicas sem perder o poder de encontrar dependências a partir dos dados de treinamento.

$$dH(X, Y) = \sum \mid (X_i = Y_i) \rightarrow 1 \vee (X_i \neq Y_i) \rightarrow 0 \mid$$

onde:

$dH(X, Y)$ é a distância de Hamming entre as seqüências X e Y.
X_i é um caracter dentro da seqüência X.
Y_i é um caracter dentro da seqüência Y.

Note que essa equação retorna a soma algébrica da quantidade de nucleotídeos que se alinham perfeitamente entre as seqüências x e y. Pode-se ainda acrescentar parâmetros adicionais para implementar dependências específicas que podem ser conhecidas a priori. Neste trabalho usamos a distância de Hamming (acima) com uma variante da expressão que implementa as redes Bayesianas. A expressão a seguir mostra $P(X_i, ..., X_k)$ calculada a partir de $dH(X, Y)$. Esta fusão de técnicas provê um aumento no desempenho das predições da Jnom através da modelagem de dependência a partir dos dados. Chamamos essa modelagem de semelhança espacial.

$$P(X_i, ..., X_k) = \frac{dH(X,Y)}{k}$$

onde:

$dH(X, Y)$ é a distância de Hamming entre as seqüências X e Y.
X é a menor seqüência.
k é o tamanho de X.

4.6 CONSIDERAÇÕES FINAIS

Neste capítulo, detalhamos algumas tecnologias importantes para o desenvolvimento da Jnom. Além disso, discutimos alguns algoritmos e métodos relevantes para a arquitetura e implementação.

Percebemos que a fusão das técnicas de raciocínio baseado em caso(RBC) com as redes Bayesianas conseguem modelar de forma adequada o problema e efetuar a busca pelos sítios. Essa fusão acontece quando usamos o cálculo da semelhança (distância de Hamming) e a matriz de freqüência. O cálculo da semelhança é uma importante característica de modelos RBC enquanto a matriz de freqüência é um caso particular de redes Bayesianas.

CAPÍTULO 5

IMPLEMENTAÇÃO E USABILIDADE NA JNOM

5.1 INTRODUÇÃO

O desempenho é um fator importante na Jnom, por isso para aumentar a precisão usamos redes bayesianas para modelar as dependências. Criamos uma ferramenta computacionalmente mais barata usando essas redes e a técnica de raciocínio baseado em casos. Além disso, um dos requisitos não funcionais mais importantes considerados foi a usabilidade, isto é, Jnom foi criada para ser fácil de usar. Por isso escolhemos uma aplicação visual, como pode ser visto na Figura 5.1.

Neste capítulo mostraremos como a Jnom funciona e como ocorre a interação com o usuário. Mostraremos ainda os parâmetros envolvidos. Os formatos dos arquivos de entrada e saída de informações também serão detalhados. Encerraremos o capítulo com uma breve análise sobre a manutenção, as vantagens, limitações e problemas conhecidos da ferramenta.

5.2 INSTRUÇÕES DE USO

De forma bastante sucinta, basta escolher o arquivo de aprendizado e o arquivo de testes e pressionar o botão "Testar", como ilustra a figura 5.2. Entretanto, há alguns parâmetros que podem ser escolhidos para ajustar o desempenho de acordo com a intenção do usuário.

O primeiro destes parâmetros é a indicação de quais *motifs* devem ser usados para aprendizado. A opção padrão é usar todos disponíveis, mas o usuário pode optar por

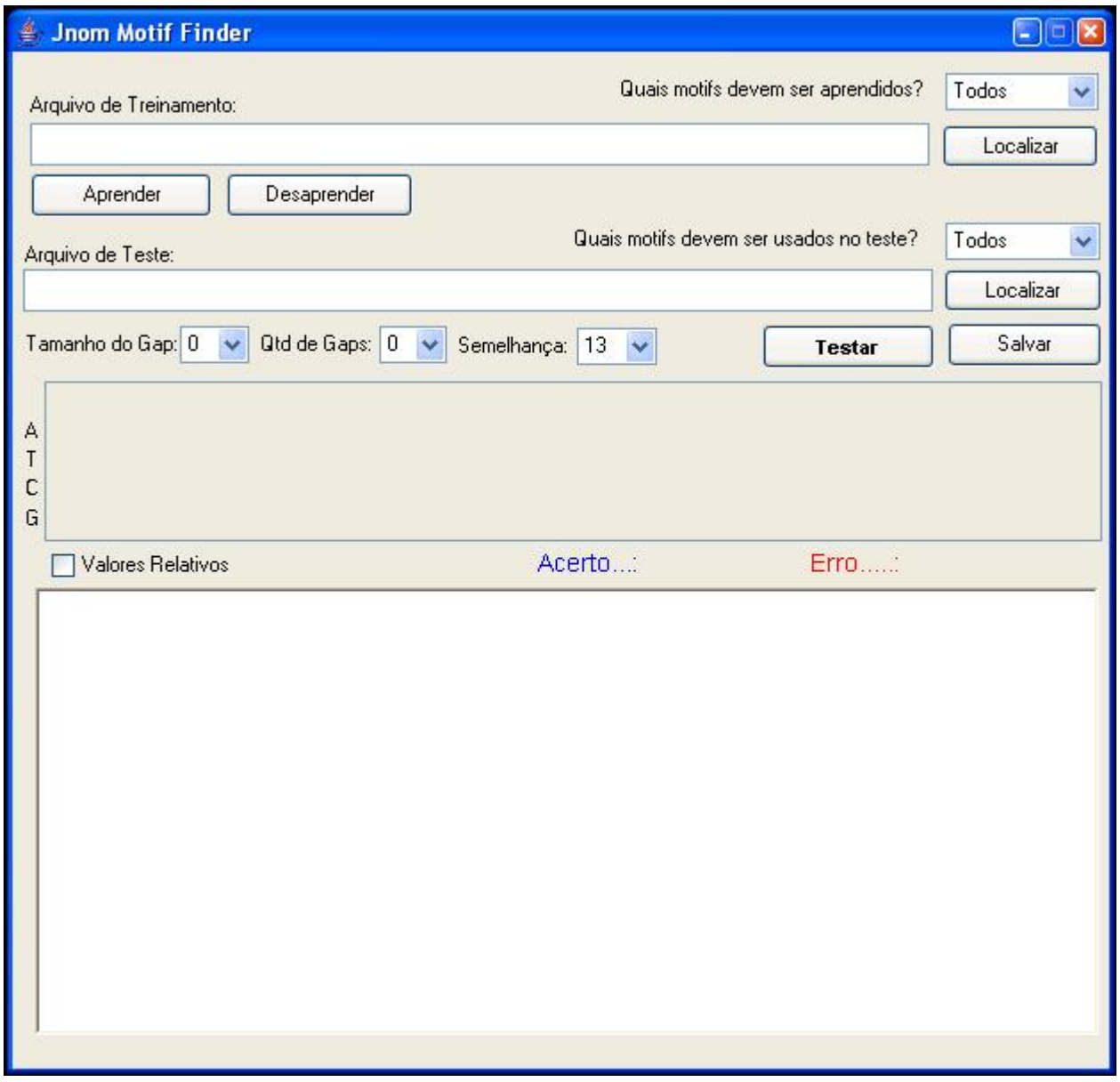


Figura 5.1. Tela Principal da Ferramenta

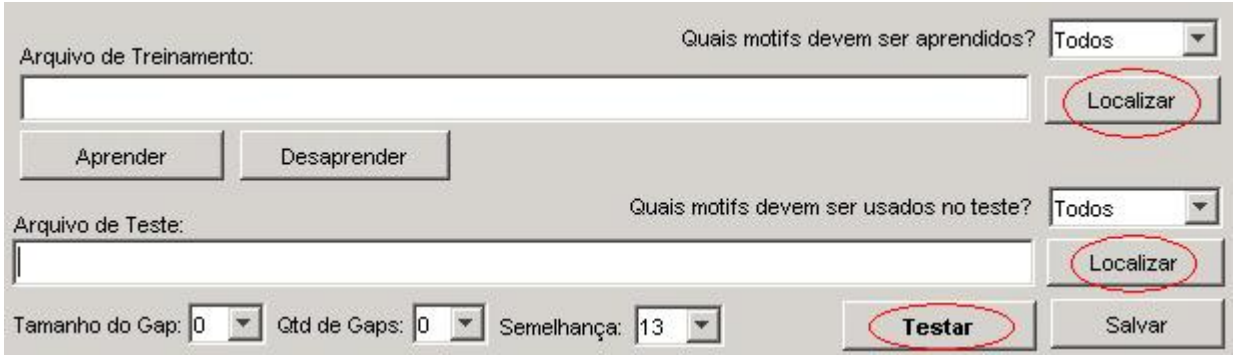
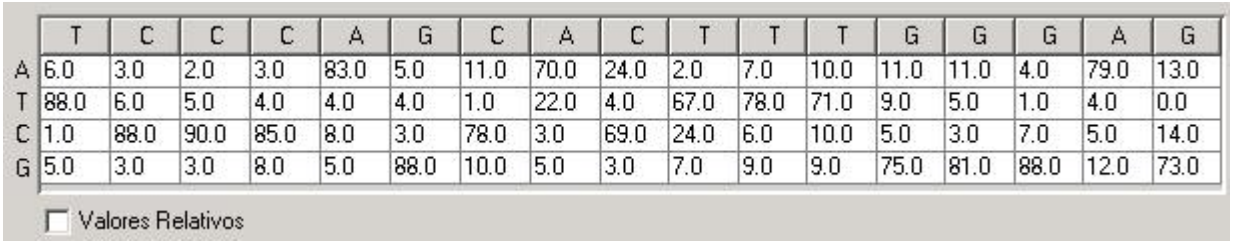


Figura 5.2. Controles fundamentais da Jnom.

aprender apenas os verdadeiros (TP), ou apenas os falsos positivos (FP). Para isso basta escolher a opção "Apenas TP"ou "Apenas FP"respectivamente, no combo "Quais motifs devem ser aprendidos?", conforme Figura 5.6.

A cada vez que o botão "aprender"é pressionado, a ferramenta lê o arquivo indicado na caixa de texto "Arquivo de treinamento"e adiciona à matriz informações especificadas no combo "Quais motifs devem ser aprendidos?"(ver figura 5.2). Isso significa que o conhecimento pode ser constantemente adicionado. Novos motifs podem ser aprendidos sem haver necessidade de criação de novos arquivos para treinar novamente a ferramenta. Caso o usuário deseje recomeçar o aprendizado o botão desaprender pode ser usado e tudo que foi aprendido é descartado.

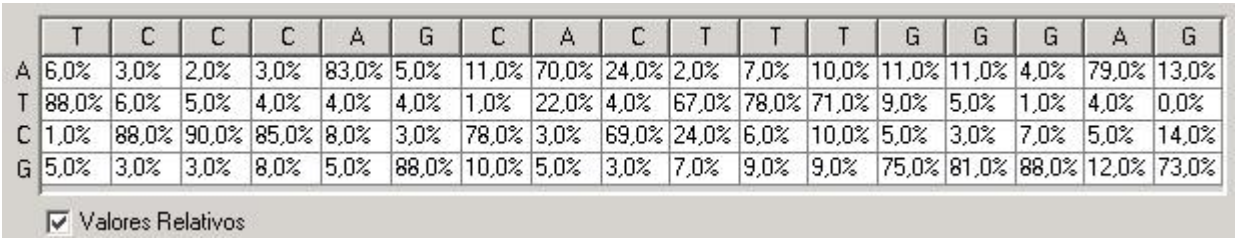
Os valores da matriz de aprendizado podem ser visualizados de duas formas diferentes. A forma padrão é em valores absolutos, mas se a caixa "Valores Relativos"for marcada, aparecerão os valores relativos em formas percentuais como ilustram as figuras 5.3 e 5.4 respectivamente. A matriz conta a quantidade de ocorrência de cada base em nas posições do *motif*. No modo de visualização em valores relativos, a quantidade de ocorrência de cada base é dividida pela soma da ocorrência de todas as bases. Essa relatividade é utilizada para se evitar *overfitting*.



	T	C	C	C	A	G	C	A	C	T	T	T	G	G	G	A	G
A	6.0	3.0	2.0	3.0	83.0	5.0	11.0	70.0	24.0	2.0	7.0	10.0	11.0	11.0	4.0	79.0	13.0
T	88.0	6.0	5.0	4.0	4.0	4.0	1.0	22.0	4.0	67.0	78.0	71.0	9.0	5.0	1.0	4.0	0.0
C	1.0	88.0	90.0	85.0	8.0	3.0	78.0	3.0	69.0	24.0	6.0	10.0	5.0	3.0	7.0	5.0	14.0
G	5.0	3.0	3.0	8.0	5.0	88.0	10.0	5.0	3.0	7.0	9.0	9.0	75.0	81.0	88.0	12.0	73.0

Valores Relativos

Figura 5.3. Forma padrão em valores absolutos após aprendizagem



	T	C	C	C	A	G	C	A	C	T	T	T	G	G	G	A	G
A	6,0%	3,0%	2,0%	3,0%	83,0%	5,0%	11,0%	70,0%	24,0%	2,0%	7,0%	10,0%	11,0%	11,0%	4,0%	79,0%	13,0%
T	88,0%	6,0%	5,0%	4,0%	4,0%	4,0%	1,0%	22,0%	4,0%	67,0%	78,0%	71,0%	9,0%	5,0%	1,0%	4,0%	0,0%
C	1,0%	88,0%	90,0%	85,0%	8,0%	3,0%	78,0%	3,0%	69,0%	24,0%	6,0%	10,0%	5,0%	3,0%	7,0%	5,0%	14,0%
G	5,0%	3,0%	3,0%	8,0%	5,0%	88,0%	10,0%	5,0%	3,0%	7,0%	9,0%	9,0%	75,0%	81,0%	88,0%	12,0%	73,0%

Valores Relativos

Figura 5.4. Forma alternativa em valores relativos após aprendizagem

5.3 APRENDIZADO NA FERRAMENTA

O usuário pode escolher a partir de qual arquivo o sistema deve aprender. O arquivo de aprendizado inicial serve como ponto de partida para o descobrimento de similaridades a fim de posteriormente aplicarmos a novas seqüências e identificarmos supostos motifs. Para escolher qual será o arquivo de aprendizagem inicial, basta pressionar o botão "Localizar"situado ao lado da caixa de texto "Arquivo de treinamento", ou digitar diretamente o caminho do arquivo na caixa de texto conforme a Figura 5.5.

Figura 5.5. Caixa de escolha do arquivo de aprendizado

Há algumas opções extras para a etapa de aprendizado, tais como, "Quais motifs devem ser aprendidos?"que apresenta as opções "Todos", "Apenas TP"e "Apenas FP",

como ilustra a figura 5.6. Nessa *comboBox*, pode-se escolher quais motifs serão considerados para aprendizagem quando o arquivo estiver sendo lido. Se for escolhida a opção "Todos"tanto os motifs verdadeiros(TP) quanto os falsos(FP) serão considerados para fins de aprendizagem. Nesta versão da ferramenta as seqüências TN(verdadeiramente negativas) não são consideradas para aprendizagem, conforme ilustra a Figura 5.6

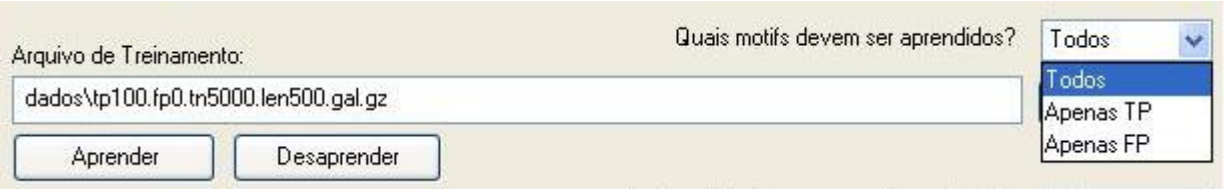


Figura 5.6. Caixa de escolha do arquivo de aprendizado mostrando o detalhe do tipo de *motif* que será aprendido

O botão "Aprender"adiciona o conhecimento dos motifs do arquivo selecionado ao conhecimento já aprendido sem desconsiderar o conhecimento previamente adquirido. Para reiniciar a matriz, é preciso pressionar o botão "Desaprender".

Mesmo se apenas um conjunto de dados for utilizado para sucessivos aprendizados dentro de um só experimento, não há possibilidade de ocorrer *overfitting*, isto é, Jnom não vai "decorar"o conjunto de entrada e perder a capacidade de generalização, pois o algoritmo trabalha com valores percentuais e por isso não sofrem variação absoluta. Esse mecanismo de evitar *overfitting* pode ser verificado empiricamente bastando-se para isso pressionar em "Aprender"várias vezes seguidas para o mesmo arquivo e observar que os valores da tabela não sofrem grande variação, conforme ilustram as Figuras 5.3 e 5.4. As Figuras 5.7 e 5.8 mostram o estado da matriz de freqüências antes e depois do aprendizado respectivamente.

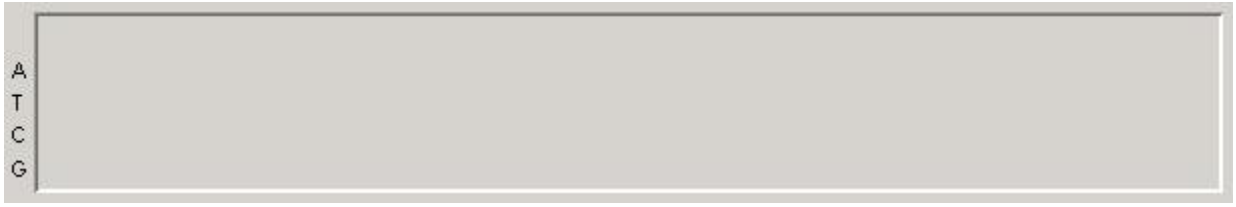


Figura 5.7. Composição da semelhança das posições antes do aprendizado (composição é vazia).

	T	C	C	C	A	G	C	A	C	T	T	T	G	G	G	A	G
A	6,0%	3,0%	2,0%	3,0%	83,0%	5,0%	11,0%	70,0%	24,0%	2,0%	7,0%	10,0%	11,0%	11,0%	4,0%	79,0%	13,0%
T	88,0%	6,0%	5,0%	4,0%	4,0%	4,0%	1,0%	22,0%	4,0%	67,0%	78,0%	71,0%	9,0%	5,0%	1,0%	4,0%	0,0%
C	1,0%	88,0%	90,0%	85,0%	8,0%	3,0%	78,0%	3,0%	69,0%	24,0%	6,0%	10,0%	5,0%	3,0%	7,0%	5,0%	14,0%
G	5,0%	3,0%	3,0%	8,0%	5,0%	88,0%	10,0%	5,0%	3,0%	7,0%	9,0%	9,0%	75,0%	81,0%	88,0%	12,0%	73,0%

☑ Valores Relativos

Figura 5.8. Composição da semelhança das posições

5.4 REPRESENTAÇÃO DOS DADOS

A seqüência de um sítio de ligação é composta pela concatenação das bases individualmente. A distribuição de probabilidade das fitas de DNA pode ser aproximada pela distribuição gaussiana [Sch97], entretanto pode-se observar empiricamente que a representação de Gauss traz pouca informação relevante para modelarmos o sítio quando comparado com a matriz de freqüência.

5.4.1 Representação para Motifs sem intervalos

Uma abordagem possível é contar o número de posições onde a base do suposto *motif* é igual à respectiva base mais freqüente em cada coluna da matriz (ver na parte superior da matriz como ilustra a Figura 5.8) e se a soma dessa quantidade for maior que um limiar, chamado de grau de semelhança, afirmamos que encontramos um *motif*. Em outras palavras: ao apresentarmos um genoma à matriz, procuramos por subseqüências que se alinham ao *motif* padrão da matriz de freqüência com uma dada medida de semelhança. Quanto maior a medida de semelhança mais criteriosa se torna a busca e

menos *motifs* serão encontrados. Analogamente se diminuirmos o grau de semelhança a matriz encontrará vários motifs, mas não necessariamente verdadeiros. Empiricamente obtivemos resultados ótimos com semelhança por volta de 60% a 70%.

Usando a matriz de freqüências dessa forma, já conseguimos uma melhora sensível na precisão da busca pelos sítios em relação às redes Bayesianas propostas por Barash et. al. [BEFK03] como ilustram as tabelas 6.1, 6.2, 6.3 e 6.4.

5.4.2 Introduzindo Intervalos na Estrutura do Motif

Outra abordagem é incrementar a busca da matriz com informações adicionais tais como a consideração de intervalos(gaps) na estrutura do genoma para formação do *motif*. Considerar intervalos significa que as bases que formam o *motif* não estão necessariamente imediatamente uma após a outra. Por exemplo, para o *motif* TAAATCCCG o genoma não precisa ser algo do tipo xxxxTAAATCCCGxxxx, mas pode ter a seguinte organização linear xxxxxxxTAxxxAATCCCGxxxxxx, onde x pode ser qualquer base.

Jnom está preparada para lidar com *gaps* na estrutura das seqüências. Como experimento, redistrubuimos alguns dos dados para considerar a existência de intervalos. Entretanto, como nossos antecessores não prepararam dados com *gaps* e o tempo de execução aumenta consideravelmente com a introdução de intervalos, não relatamos os experimentos realizados com *gaps* no corpo dessa dissertação.

5.5 ARQUIVOS DE ENTRADA E SAÍDA

Os arquivos usados nos testes dos experimentos seguem o formato especificado por Friedman e Barash. Esses arquivos estão disponíveis em [BEFK] e cada uma das suas linhas devem obedecer as seguintes regras:

Os primeiros três caracteres são, em ordem da esquerda para direita, sinal "maior

que"(>) seguido do tipo do *motif* encontrado no genoma que esta linha contem. Este tipo pode ser TP para *True Positive*, FP para *False Positive* ou TN para *True Negative*.

- TP significa que o genoma apresentado terá um *motif* verdadeiro.
- FP significa que o genoma apresentado terá um falso *motif*, isto é, terá uma seqüência muito parecida.
- TN significa que o genoma apresentado não terá motifs reais.

Os arquivos de aprendizado podem ter dois formatos distintos(simples ou completo) como veremos a seguir. Os arquivos para testes devem sempre possuir o formato completo, isto é, com cabeçalho e seqüência, conforme detalharemos nas seções subseqüentes.

5.5.1 Arquivos para Aprendizado ou Teste

Após a identificação do tipo, deve haver um ponto e um número inteiro para identificação da linha. Esse número é um contador que é reiniciado cada vez que o tipo de *motif* é alterado. A seguir há uma ilustração de um exemplo de como poderia ser o início das linhas do arquivo:

$> TP.1$
$> TP.2$
$> TP.3$
...
$> TP.48$
$> TP.49$
$> TP.50$
$> FP.1$

$> FP.2$

$> FP.3$

$> FP.4$

...

Após o identificador de linha, há um espaço em branco. Se o Tipo for TP ou FP o espaço será seguido de um "abre colchetes"([). Após o colchete é colocado o *motif* (TP ou FP, conforme consta no início da linha) seguido da posição em que esse *motif* será encontrado na seqüência que virá no final da linha. A posição do *motif* na seqüência deve ficar depois do *motif* e precedido por espaço em branco seguido de "at"seguido de espaço em branco. Depois do número inteiro que indica a posição do *motif* deve vir um "fecha colchetes"(]) como mostra o exemplo:

[GTCTTTAGGTTAAAGGC at 10]

Nesse exemplo, o *motif* é GTCTTTAGGTTAAAGGC e será encontrado na posição 10 da seqüência que estará no final da linha.[1]

Após o símbolo de "fecha colchetes"(]) há uma tabulação, seguida da probabilidade de regulação, seguido por outra tabulação e finalmente a seqüência de cujas informações no início da linha se referem.

De forma sucinta, a linha típica explicada acima tem o seguinte formato:

>Tipo.Id [motif at posição] (tab) probabilidade (tab) seqüência.

Pode-se exibir como exemplo o início da oitava linha do arquivo PSSM-tp50.fp50.tn5000.len500.gal.gz:

[1]Para esta definição de formato considera-se o índice da primeira posição como sendo um e não zero. É válido esclarecer, pois isto poderia confundir os programadores de linguagens como C ou Java que consideram zero como índice da primeira posição de arrays.

>TP.8 [TCGCAGCTCTTAGAGAG at 9] 0.99 TGCTGCCCCTCGCAGCTCTTAGA-GAG...

A linha do arquivo começa no sinal de "maior que"(>) e termina com o fim da seqüência "...TAGAGAGCTGCTC". Note o que o *motif* no cabeçalho da linha(dentro dos colchetes) está na posição nove da seqüência como informado pelo próprio cabeçalho.

5.5.2 Arquivos Exclusivos para Aprendizagem

Há um segundo formato que pode ser usado apenas para a fase de aprendizagem. Estes arquivos não precisam conter a seqüência, bastam os cabeçalhos. Isto é, cada linha precisa conter apenas a primeira parte de cada linha do formato completo. Este formato diminui o tamanho físico necessário para cada arquivo. Elucidando: bastariam os arquivos possuírem linhas como o exemplo abaixo:

>TP.1 [TAGCAGCGAATTGGGTG at 342]
>TP.2 [CCGCAGCACTTTAGGAA at 284]
>TP.3 [CCGCAGGAATTTGGGAG at 164]
>TP.4 [CCCCGTCAATTTGGGAG at 51]
>TP.5 [TCCCAGAACCGGGGGAC at 356]
>TP.6 [TTCGAGCACCTGGGGAG at 285]
>FP.1 [TCCGGCTCCGCGGCTAA at 336]
>FP.2 [ACACATTAAGGTATTTC at 373]
>FP.3 [GTGTCTCAAGTAGTACC at 12]
>FP.4 [AAAAGGTCAGGAGTAAA at 372]

Para este tipo de arquivo, consideram-se apenas as linhas indicadas por TP e FP. As linhas identificadas por TN são ignoradas no aprendizado. As linhas TN são descartadas porque se deseja aprender padrões de motifs para sabermos classificá-los como verdadeiros

ou falsos. A ausência de motifs indicada por TN será posteriormente classificada se não forem identificados como TP ou FP.

Como os arquivos no formato simplificado podem ser substituídos pelo formato completo e todos os arquivos gerados por Friedman e Barash já estavam nesse modo, usamos apenas os arquivos no formato estendido(completo) nos experimentos.

5.5.3 Nomenclatura dos Arquivos de Entrada

O nome do arquivo segue um padrão que dá algumas informações importantes sobre seu conteúdo.

Os nomes dos arquivos seguem a seguinte regra de formação: primeiro é o radical que informa como foram gerados os dados contidos pelo arquivo. Usou-se PSSM ou TREE para radical.

Após o radical, há um hífen seguido da indicação do tipo em letra minúscula e da quantidade de cada tipo de *motif* que o arquivo contém, não necessariamente em ordem de aparição, mas a informação dos três tipos está sempre presente e entre cada tipo coloca-se um ponto para separar os tipos distintos (TP, FP e TN). Um exemplo dessa parte é -tp50.fp75.tn5000

Nesse exemplo, o arquivo deve conter 50 motifs verdadeiros indicado por tp50, 75 falsos positivos indicado por fp75 e 5000 seqüências sem motifs indicado por tn5000. Nos arquivos usados para os experimentos, a ordem em que os tipos aparecem no nome do arquivo é a ordem em que os motifs estão dispostos dentro do arquivo.

Por fim, coloca-se o tamanho das seqüências, precedido das letras "len"de tamanho em inglês(length), por exemplo: len300 e acrescenta-se o sufixo .gal.gz para identificar

o tipo do arquivo para o sistema operacional e facilitar buscas e classificações. Como exemplo do nome completo do arquivo temos:

PSSM-tp50.fp25.tn5000.len500.gal.gz

Pelo nome do arquivo, podemos dizer que ele foi gerado por algoritmos baseados em PSSM e tem 50 seqüências positivas verdadeiras nas primeiras 50 linhas seguidas por 25 linhas que contém seqüências com motifs falsos seguidas por 500 seqüências sem motifs. Podemos dizer ainda que as seqüências em cada linha desse arquivo possuem 500 bases de comprimento.

É válido ressaltar que o nome do arquivo é apenas uma convenção e não precisa ser obedecido, pois a ferramenta não faz nenhuma conferência de formato, entretanto é aconselhável seguir este padrão, pois facilita bastante na localização e determinação do conteúdo dos arquivos.

5.5.4 Arquivos de Saída

Jnom disponibiliza três formas de saída de resultados e informações. A principal delas é a própria janela da aplicação construída usando *applet*. Essa característica a torna facilmente portável e bastante intuitiva. Outra forma de saída, é um arquivo somente texto (com a extensão *.txt). Esse arquivo guarda o resultado de cada execução do sistema em formato texto unicode. Esse arquivo é ideal para leitura e interpretação manual, ou seja, esse arquivo funciona como um relatório de execução e pode facilmente ser lido por qualquer pessoa. Nele, é sempre adicionado o resultado da última execução ao final do arquivo de acordo com o formato abaixo:

```
DD/MM/AA HH:MM:SS jnom.data.Log logResult
INFO:
Arquivo de Treinamento.............:  <nome do arquivo de treinamento>
Motifs Aprendidos durante Treinamento.:  <motifs usados para o
```

```
treinamento:  Todos, TP ou FP>
Arquivo de Teste.....:  <nome do arquivo de teste>
Motifs usados para teste.....:  <motifs usados para o teste:
Todos, TP ou FP>
Quantidade máxima de Gaps.....:  <especificado na GUI>
Tamanho máximo de cada Gap.....:  <especificado na GUI>
Grau de semelhança.....:  <especificado na GUI>
Motif Padrão.....:  <motif padrão aprendido>
Acertos.....:  <percentual de erro dentre os
motifs usados para teste>
Erros.....:  <percentual de acerto dentre os
motifs usados para teste>
```

Como exemplo pode-se mostrar:

```
   22/08/2004 09:24:42 jnom.data.Log logResult
INFO:
Arquivo de Treinamento.....:  tp100.fp0.tn5000.len500.gal.gz
Motifs Aprendidos durante Treinamento.:  Todos
Arquivo de Teste.....:  tp50.fp50.tn5000.len500.gal.gz
Motifs usados para teste.....:  Todos
Quantidade máxima de Gaps.....:  0
Tamanho máximo de cada Gap.....:  0
Grau de semelhança.....:  13
Motif Padrão.....:  TCCCAGCACTTTGGGAG
Acertos.....:  97,84%
Erros.....:  02,16%
```

Uma terceira forma de saída implementada, nesta versão da ferramenta, é o arquivo XML que contém algumas informações a mais conforme mostraremos a seguir. Esse formato de saída é ideal para fazer a integração com outros sistemas ou para enviar as informações para computadores remotos para fins de análise automática ou para guardar em bancos de dados. O formato é simples e também permite a interpretação manual dos dados de saída. Assim como no arquivo texto de log, os resultados são adicionados ao XML a cada execução automaticamente.

No cabeçalho do XML encontramos:

```
<?xml version="1.0"encoding="windows-1252"standalone="no"?>
<!DOCTYPE log SYSTEM "logger.dtd≫
```

Durante a execução dos testes, Jnom gera um arquivo XML com o resumo do que está acontecendo nos testes. No corpo do arquivo encontramos registros com informações sobre as execuções. Tais informações estão dispostas no arquivo como ilustrado a seguir (a cada execução de um teste todas as informações são gravads).

```
<log>
<record>
<date>2004-08-22T09:24:42</date>
<millis>1093177482703</millis>
<sequence>0</sequence>
<logger>JModel.log</logger>
<level>INFO</level>
```

```
<class>jnom.data.Log</class>
<method>logResult</method>
<thread>10</thread>
<message> resultado da execução </message>
</record>
</log>
```

5.6 VANTAGENS

A facilidade de uso é uma das características marcantes da Jnom, tudo pode ser feito usando apenas o mouse e em uma única janela. É importante lembrar que a simplicidade de uso não inibe a flexibilidade. A ferramenta, ao ser iniciada, é carregada com uma série de parâmetros que podem ser usados como padrão, ou podem ser alterados para atenderem às necessidades específicas dos usuários.

Outra grande vantagem é a facilidade de modificar a implementação das classes principais, pois todo o desenvolvimento foi feito usando interfaces para garantir a independência de implementação.

5.7 LIMITAÇÕES

O uso de gaps proporciona aumento exponencial no número de operações em função da quantidade e do tamanho do intervalo, pois para cada gap de tamanho n, é preciso fazer n comparações a mais para cada *motif*. Essa alta taxa de crescimento no tempo de execução limita bastante a ferramenta em relação a buscas com gaps. A interface gráfica com applet implementada não está preparada para motifs muito grandes, pois não caberiam na tela. Essa é uma limitação apenas visual da interface implementada.

Apesar de fornecer XML como saída, não foi implementada a leitura usando esse

tipo de arquivo. Levando-se em consideração a arquitetura extensível desenvolvida, seria relativamente fácil implementar a entrada de dados via XML.

Como a ferramenta foi desenvolvida usando *applet*, todo o processamento é feito no cliente. esse fato faz com que o tempo de resposta dependa da máquina cliente. É relativamente fácil migrar a aplicação para rodar em servidores mais robustos e de forma distribuída. Para isso poderiam ser usados outros recursos, tais como, *threads* e *servlets*.

Outro problema conhecido é o lançamento de uma exceção em tempo de execução. Esse erro ocorre em qualquer plataforma, quando se pressiona "Testar"e não há pelo menos um *motif* especificado em "Quais motifs devem ser usados", a máquina virtual lança a seguinte exceção:

java.util.NoSuchElementException

Se na caixa "quais motifs devem ser usados"no teste estiver selecionada a opção "Apenas TP"e no arquivo de teste não existir nenhum *motif* TP, a exceção é levantada e a ferramenta precisa ser reiniciada. Esse bug não aborta subitamente a aplicação e a matriz de aprendizado pode ser salva sem problemas. O sintoma que esse erro ocorreu é sentido através da inabilitação da tabela inferior onde são exibidos os resultados, como ilustra a Figura **??**.

5.8 CONSIDERAÇÕES FINAIS

Conseguimos criar uma ferramenta intuitiva implementada com Applet e saídas XML que facilitam a integração. Utilizamos redes bayesianas e raciocínio baseado em casos para modelar dependências encontradas entre os nucleotídeos dos motifs. Preferimos usar uma arquitetura baseada em padrões de projeto para facilitar a manutenção corretiva e evolutiva da ferramenta.

Apesar da ferramenta suportar busca de motifs com intervalos(gaps), a implementação do algoritmo de busca, com esse parâmetro diferente de zero, faz o tempo de resposta aumentar exponencialmente de acordo com o parâmetro. Os testes com motifs contendo intervalos não foram foco do nosso trabalho ficando isso como evolução posterior.

CAPÍTULO 6

ANÁLISE DOS RESULTADOS

6.1 INTRODUÇÃO

Neste documento, expandimos a representação probabilística de motifs de DNA. Por isso, usamos a linguagem de redes Bayesianas com o auxílio de matrizes de posição específica. Nos modelos cognitivos usamos a busca por semelhança para considerar a ação combinada das proteínas. Nossa estrutura permite expansão e agregação de outros modelos independente da utilização das técnicas abordadas. Descrevemos métodos para aprender modelos a partir de dados limitados e observamos a relação da ação combinada com o desempenho em relação aos métodos com independência total ou apenas dependência determinada.

Apresentamos métodos para encontrar supostos sítios de ligação usando um modelo genérico e descrevemos uma abordagem efetiva para encontrar sítios candidatos. Finalmente, mostramos como realizamos a descoberta de novos de *motifs* em seqüências de genomas não alinhados. Através de uma avaliação empírica, comparamos a eficiência de alguns modelos de dependência e independência presentes na literatura.

Como vimos anteriormente, os genes são controlados por proteínas que se ligam a pontos específicos na seqüência de DNA. Nosso objetivo é encontrar esses pontos. Uma característica empiricamente marcante dos *motifs* é a uniformidade, isto significa que há um certo grau de homogeneidade nos sítios de ligação dos organismos. É importante enfatizar que quanto mais próximo biologicamente o organismo usado para o treinamento (construir a matriz) mais conservados serão os seus *motifs*[ABJ+99]. Esse fato, ajuda a

inferir regras para treinar algoritmos de busca para localizar tais *motifs*.

A seguir, faremos uma breve comparação entre soluções, analisaremos os experimentos realizados com a família GAL[1] e consolidaremos os resultados empíricos.

6.2 COMPARANDO SOLUÇÕES

A PSSM pode ser vista como um caso particular das redes Bayesianas. É o caso onde não há dependência específica entre posições. No caso mais extremo, pode-se considerar que haja dependência entre todas as posições. Essa dependência completa pode tornar os algoritmos baseados em cálculos Bayesianos muito lentos e de certa maneira os engessam[Fri98]. Como a idéia original é modelar as dependências entre posições, propusemos uma forma alternativa de implementar tais redes através da técnica de raciocínio baseado em casos, usando suas características, para modelar as dependências e aumentar o desempenho em relação a tentativas anteriores.

Matematicamente, pode-se associar a seqüência de variáveis aleatórias $s = X_1, ..., X_n$ uma probabilidade P, como vimos anteriormente, e sobre essa seqüência podemos tentar encontrar dependências entre suas posições, ou seja, relações entre as variáveis que compõem a seqüência. Parte de nossa tarefa é encontrar quais variáveis da seqüência são dependentes e de quem elas dependem.

Para encontrar essas relações entre as variáveis no *motif*, usamos a técnica de raciocínio baseado em caso. Imagine que um observador externo resolveu alinhar vários *motifs* e colocá-los um sobre o outro. Em seguida, o observador começa a ordenar e contar quais bases aparecem em cada posição. Visualmente, não é difícil para o observador enxergar as dependências.

[1] A família GAL é uma família de proteínas quem contém galactose[yea].

As redes Bayesianas trabalham de forma semelhante ao observador. A diferença é que, nos algoritmos estudados, as dependências já são informadas indiretamente através do modelo da rede e são desconsiderados o caráter espacial da semelhança.

Para modelar essa semelhança usamos a distância de Hamming[Bar01, Kol92]. A maioria das ferramentas optam pela abordagem estatística das redes. Nesse caso, elas precisam de um modelo prévio. Jnom enfatiza o caráter da semelhança espacial do conjunto de treinamento. Com isso, é possível identificar as dependências sem a necessidade de fornecer o modelo como entrada. Para um melhor entendimento, detalharemos sobre essa forma de trabalho conforme exemplo a seguir.

Considere os seguintes motifs alinhados:

TCCCAGCAATAGAG**GA**G
TCCCAGCAATTGGG**GA**G
TCCCACCCCTTTGG**GA**G
ACACAGCACTTTGG**GA**A
TCCGAGGACTTTGA**GA**T
TCCCAGCACGTTAG**GA**G
TGCAAGCTCTTGAG**GA**G
CCCCAGGACTTTGG**GA**G
TCCCACCACTTAGG**GA**C

Não é complicado observar visualmente que a quinta e a penúltima posição da esquerda para direita é sempre Adenina(A) e a antepenúltima é sempre Guanina(G). Este fato, nos faz supor que essas posições independem de qualquer outra, pois elas permanecem constantes.

Vamos observar a seguinte série de motifs:

TCCCAGCCGGTAGGGAA
TCCCAGCATTTTGGGGG
TC**TG**AGCGACTTGGGAG
TCCCAGTACTTTGGGAG
TC**TG**AGCACGTTGGGAG
TCCCAGGACTTTGGGAG
TCCCAGCACTATTGGAG
TCCTAGCCCCTTGGCAG
TC**TG**AGCACTGGTAGTC
TCCCACAACTTTGGGAG

Note, mais uma vez, que há posições inalteradas, tais como a primeira posição onde sempre há Timina(T). Esse exemplo possui algumas outras características peculiares. Observe a terceira e a quarta posição. Sempre que há uma Timina na terceira fileira, há uma Guanina na quarta. Esse tipo de observação faz o desempenho melhorar. Nossa ferramenta usa esse tipo de observação espacial para indicar os resultados.

6.3 RESULTADOS RESUMIDOS DE BARASH

Para facilitar a comparação de resultados, Jnom usou a mesma base de dados que Barash e Friedman utilizaram. Essa base pode ser encontrada em [BEFK]. Para avaliar os resultados, foram usadas duas medidas chamadas sensibilidade e especificidade como veremos a seguir.

6.3.1 Sensibilidade

Esta medida informa a taxa de acerto nas seqüências que realmente contém o *motif*. Isto é, percentual de motifs positivos obtidos dentre todas as seqüências que realmente os contêm. Por exemplo, se usarmos um arquivo para testes em que TP = 100, independentemente do número de TF e TN, a sensibilidade será a quantidade dos 100 motifs TP preditos corretamente. Daí a fórmula geral:

Sensibilidade = (Quantidade de TP encontrado / Quantidade de TP total) * 100

6.3.2 Especificidade

Esta outra medida informa a taxa de acerto nas seqüências que contêm motifs independente de serem falsos positivos ou não. Ou seja, a especificidade indica o percentual de seqüências que possuem motifs verdadeiros dentre todas as seqüências que contêm *motifs* sejam eles verdadeiros ou falsos. Por exemplo, se usarmos um arquivo para testes em que TP = 75 e FP = 25 a especificidade será o número de motifs TP predito corretamente, independente do número de TN. Daí a fórmula geral:

Especificidade = (Quantidade de TP encontrado / Quantidade de (TP+FP) total) * 100

6.3.3 Tabelas de Resultados

Como se pode observar, com as Tabelas 6.1, 6.2, 6.3 e 6.4, quando os dados não contêm seqüências com falsos positivos(FP), todos os modelos predizem sem muita diferença. Nos dados gerados a partir de árvores de dependência, o modelo PSSM é o pior. A mistura de PSSM é melhor. A melhor predição ficou com o mistura de árvores.

Tabela 6.1. Sensibilidade usando dados gerados a partir de PSSM

Método de Busca	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	68%	65%	64%
Árvores	68%	65%	66%
Mistura de PSSM	67%	61%	53%
Mistura de árvores	60%	43%	40%

Tabela 6.2. Sensibilidade usando dados gerados a partir de árvores com dependências

Método de Busca	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	68%	68%	67%
Árvores	85%	82%	80%
Mistura de PSSM	70%	66%	67%
Mistura de árvores	72%	67%	64%

Tabela 6.3. Especificidade usando dados gerados a partir de PSSM

Método de Busca	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	56%	51%	57%
Árvores	56%	52%	56%
Mistura de PSSM	55%	48%	46%
Mistura de árvores	48%	38%	48%

Tabela 6.4. Especificidade usando dados gerados a partir de árvores com dependências

Método de Busca	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	65%	65%	63%
Árvores	66%	70%	66%
Mistura de PSSM	64%	59%	57%
Mistura de Árvores	62%	54%	56%

6.4 RESULTADOS COM DADOS SINTÉTICOS

Realizamos vários experimentos com os mesmos dados usados por Barash, entretanto a Jnom possui alguns parâmetros ajustáveis não providos pelo algoritmo anterior. Estes parâmetros são semelhança, quantidade de gaps, tamanho do gap, *motifs* aprendido e *motifs* usados para teste, assumindo respectivamente os valores 13, 0, 0, Todos e Todos. Estes valores foram obtidos empiricamente. O principal desses parâmetros é o grau de semelhança. De acordo com o tipo das seqüências, este parâmetro faz o resultado variar consideravelmente. Essa variação ocorre devido ao fato de que o nosso alfabeto é muito limitado e a rede Bayesiana usada na implementação com distância de Hamming não considera características biológicas adicionais.

Na prática, se o usuário disser que a semelhança entre *motifs* é pequena a Jnom considerará a ocorrência de *motifs* mesmo quando as seqüências forem biologicamente muito diferentes. Isso significa que a Jnom poderá encontrar muitos *motifs* em seqüências contendo falsos positivos. Entretanto, se o grau de semelhança utilizado for alto demais, Jnom só conseguirá encontrar novos *motifs* verdadeiros se estes forem biologicamente muito próximos.

Essa grande variação é esperada, uma vez que estamos estudando a composição

espacial dos *motifs*, e essa composição possui um determinado grau de homogeneidade. Como conhecemos, a princípio, a composição de cada arquivo de entrada, é possível ajustar os parâmetros para se obter o máximo de acerto. Entretanto, isso não retrataria a realidade porque nos experimentos reais não sabemos quantos *motifs* existem, nem onde eles estão, por isso, realizamos uma série de experimentos e determinamos empiricamente parâmetros ótimos que apresentam bons resultados em todos os conjuntos de dados. Estes parâmetros ótimos são os carregados como padrão quando a ferramenta é iniciada.

Uma média dos resultados com esses parâmetros, segundo os mesmos critérios de Barash e Friedman, podem ser resumidos nas tabelas 6.5 e 6.6. É importante salientar que os dados sintéticos foram gerados a partir dois modelos distintos: PSSM e árvore.

Tabela 6.5. Sensibilidade na Jnom

Dados	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	71%	65%	67%
Árvore	71%	55%	75%

A análise de semelhança das posições espaciais independe de como foram gerados os *motifs* de aprendizagem, por isso, a sensibilidade e a especificidade são semelhantes para ambos os casos.

Lembramos que não possuímos modelos intermediários, tais como mistura de árvores ou misturas de PSSM, pois a Jnom encontra as estruturas de dependências usando um único modelo, através da semelhança entre as posições.

Tabela 6.6. Especificidade na Jnom

Dados	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	71%	82%	81%
Árvore	71%	76%	88%

6.5 RESULTADOS COM A FAMÍLIA GAL

Efetuamos alguns testes com seqüências da família GAL, infelizmente há poucos dados disponíveis sobre esta família, por isso apenas um arquivo foi construído. Todos os parâmetros permaneceram idênticos aos dos experimentos nos dados sintéticos e ainda assim a ferramenta mostrou índices satisfatórios de sensibilidade e especificidade conforme ilustra a Tabela 6.7. O aprendizado neste experimento foi a partir de 40% dos *motifs* encontrados na própria família GAL.

Tabela 6.7. Resultados com a Família GAL

Medida	TP = 30, FP = 30
Sensibilidade	71%
Especificidade	65%

Foram utilizadas algumas seqüências da família GAL obtidas em [**?**]. Essas seqüências foram submetidas ao MEME[Bai99] para que seus *motifs* fossem devidamente encontrados. As seqüências são:

- GAL1/YBR020W no cromossomo II, nas posições 279021 a 280607.
- GAL2/YLR081W no cromossomo XII, nas posições 290213 a 291937.
- GAL3/YDR009W no cromossomo IV, nas posições 463431 a 464993.

- GAL4/YPL248C (GAL81) no cromossomo XVI, nas posições 82356 a 79711.
- PGM2/YMR105C (GAL5) no cromossomo XIII, nas posições 477605 a 475896
- LAP3/YNL239W (BLH1, GAL6, YCP1) no cromossomo XIV, nas posições 200568 a 201932.
- GAL7/YBR018C no cromossomo II nas posições, 275527 a 274427.
- GAL10/YBR019C no cromossomo II from coordinates 278352 a 276253.
- GAL11/YOL051W (RAR3, SDS4, SPT13, ABE1, MED15) no cromossomo XV, nas posições 234938 a 238183.
- GAL80/YML051W no cromossomo XIII, nas posições 171594 a 172901.
- GAL83/YER027C (SPM1) no cromossomo V, nas posições 210231 a 208978.
- SIN4/YNL236W (BEL2, GAL22, SDI3, SSF5, SSN4, TSF3, RYE1, MED16) no cromossomo XIV, nas posições 206929 a 209853.

As Tabelas 6.8 e 6.9 ilustram esquematicamente as entradas utilizadas.

Mesmo após buscar na internet, ainda possuíamos poucas seqüências para testes. Para solucionar esse problema, modificamos aleatoriamente cerca de 10% a 20% das bases de cada seqüências obtendo 24 para trabalharmos nos testes. Adicionalmente, utilizamos mais 8 seqüências previamente classificadas pela equipe do biolab [bio]. Essas últimas seqüências foram previamente analisadas e seus *motifs* descobertos através de análises computacionais em outras ferramentas. Dessa forma totalizaram-se 32 seqüencias. Pode-se encontrar mais detalhes sobre tais seqüências em [vHRCV00].

Ao submetermos as 12 seqüências originais encontradas no endereço do sítio do *yeastgenome* em [yea] à ferramenta de busca MEME, encontramos vários *motifs* para cada seqüência. Parte desses *motifs* encontrados foram utilizados como entrada para o

Tabela 6.8. Seqüências da família GAL

Nome	AC	Nome do Gene	Descrição	Tam.
BLH1_YEAST	Q01532	BLH1, GAL6, LAP3, YCP1, YNL239W, N1118	Cisteína proteinase 1 (EC 3.4.22.40) (Y3) (Bleomicina hidrolase) (BLM hidrolase)	454
GAL10_-YEAST	P04397	GAL10, YBR019C, YBR0301	GAL10 proteína bifuncional [Inclue: UDP-glicose 4-epimerase (EC 5.1.3.2) (Galactowaldenase); Aldose 1-epimerase (EC 5.1.3.3) (Mutarotase)]	699
GAL11_-YEAST	P19659	GAL11, RAR3, SPT13, YOL051W	Regulação Transcricional proteína GAL11	1081
GAL1_YEAST	P04385	GAL1, YBR020W, YBR0302	Galactokinase (EC 2.7.1.6) (Galactose kinase)	527
GAL2_YEAST	P13181	GAL2, IMP1, YLR081W, L9449.6	Transportadore de Galactose (Galactose permease)	574
GAL3_YEAST	P13045	GAL3, YDR009W, YD8119.14	proteína GAL3	520

Tabela 6.9. Seqüências da família GAL (cont.)

Nome	AC	Nome do Gene	Descrição	Tam.
GAL4_YEAST	P04386	GAL4, YPL248C	Proteína Regulatória GAL4	881
GAL7_YEAST	P08431	GAL7, YBR018C, YBR0226	Galactose-1-fosfato uridililtransferase (EC 2.7.7.12) (Gal-1-P uridililtransferase) (UDP-glicose-hexose-1-phosphate uridililtransferase)	365
GAL80_YEAST	P04387	GAL80, YML051W, YM9958.12, YM9827.01	Galactose/lactose metabolismo regulatorio proteína GAL80	435
GAL83_YEAST	Q04739	GAL83, SPM1, YER027C	Proteína repressora de Glicose GAL83 (proteína SPM1)	417
PGM2_YEAST	P37012	PGM2, GAL5, YMR105C, YM9718.04C	Fosfoglicomutase 2 (EC 5.4.2.2) (Glicose fosfomutase 2) (PGM 2)	569
SAG1_YEAST	P20840	SAG1, AGAL1, YJR004C, J1418	Alpha-aglutinina precursora (AG-alpha-1)	650
SIN4_YEAST	P32259	SIN4, BEL2, GAL22, SSF5, TSF3, YNL236W, N1135	Regulação transcricional global SIN4	974

aprendizado da Jnom. Isto significa que Jnom conseguiu generalizar e manter o nível de acerto anteriormente verificado nos dados sintéticos, conforme ilustra a Tabela 6.7.

Vamos detalhar o comportamento da Jnom através de um exemplo. Vale lembrar que o treinamento foi feito com trinta *motifs* encontrados na própria família Gal com a ajuda da ferramenta MEME[Bai99]. Consideramos os *motifs* alinhados da Tabela 6.10 para realização deste experimento.

Tabela 6.10. Motifs para aprendizado na família GAL

Motifs	*Motifs*
CGGGCGACAGCCCTCCG	CGGACAACTGTTGACCG
CGGAGGAGAGTCTTCCG	CGGTCAACAGTTGTCCG
CGGAAGACTCTCCTCCG	CGGCGCACTCTCGCCCG
CGGATTAGAAGCCGCCG	CGGATTAGAAGCCGCCG
CGGGCGACAGCCCTCCG	CGGAAGACTCTCCTCCG
CGGATTAGAAGCCGCCG	CGCGCCGCACTGCTCCG
CGGGCGACAGCCCTCCG	CGGAGGAGAGTCTTCCG
CGGATTAGAAGCCGCCG	CGCGCCGCACTGCTCCG
CGGGCGACAGCCCTCCG	CGGAAGACTCTCCTCCG

Vamos analisar o comportamento da submissão da seqüência ilustrada pela Tabela 6.11 à Jnom. Como se pode observar, na primeira linha da tabela, há 1101 nucleotídeos na GAL7_YEAST. Segundo o MEME, há ainda 5 motifs com 17 bases de comprimento cada nas posições 118, 892, 309, 554 e 1042. Com os parâmetros nos valores padrão,

Jnom foi capaz de encontrar 3 dos 4 *motifs* existentes, ou seja, 75%.

Se aumentarmos o grau de semelhança, o índice de acerto diminui, pois a distância de Hamming calculada para essa semelhança torna a busca mais criteriosa. Isto pode significar que os *motifs* precisam ter suas características mais conservadas, em relação ao conjunto de aprendizado, para serem encontrados. Este aumento no grau de semelhança pode ser útil na busca por *motifs* específicos em seqüências de organismos biologicamente semelhantes.

Por outro lado, quando diminuímos o grau de de semelhança, a Jnom encontra todos os *motifs*, mas em contrapartida também encontra falsos positivos, pois a distância de Hamming calculada para essa semelhança torna a busca menos criteriosa. Isto significa que a Jnom ficará tendenciosa a encontrar *motifs* com as características pouco conservadas em relação ao conjunto de aprendizado. Resumindo, a diminuição da semelhança torna a Jnom propícia a encontrar falsos positivos. Este decremento pode ser útil na busca por *motifs* em seqüências de organismos biologicamente muito diferentes.

Como conseqüência da técnica de RBC em conjunto com a distância de Hamming, a diminuição da semelhança pode ser adequada para uma varredura inicial. Essa passagem preliminar, pode excluir grandes trechos da seqüência que não contém *motifs*, facilitando assim análises subseqüentes mais específicas através da diminuição da quantidade de nucleotídeos que precisariam ser considerados. Este comportamento da Jnom, em relação à alteração no grau de semelhança, pôde ser verificado empiricamente como regra geral nas demais seqüências da família Gal e nos experimentos realizados com dados sintéticos.

6.6 CONSOLIDAÇÃO DOS RESULTADOS

É possível comparar os resultados da Jnom em relação ao método descrito por Barash Friedman uma vez que utilizam a mesma base de dados. As Tabelas 6.12, 6.13, 6.14 e

Tabela 6.11. Seqüência GAL7_YEAST

Info: 1101 BP; 350 A; 217 C; 205 G; 329 T; 0 other; 544892515 CRC32;	
atgactgctg aagaatttga tttttctagc cattcccata gacgttacaa tccactaacc	60
gattcatgga tcttagtttc tccacacaga gctaaaagac cttggttagg tcaacaggag	120
gctgcttaca agcccacagc tccattgtat gatccaaaat gctatctatg tcctggtaac	180
aaaagagcta ctggtaacct aaacccaaga tatgaatcaa cgtatatttt ccccaatgat	240
tatgctgccg ttaggctcga tcaacctatt ttaccacaga atgattccaa tgaggataat	300
cttaaaaata ggctgcttaa agtgcaatct gtgagaggca attgtttcgt catatgtttt	360
agccccaatc ataatctaac cattccacaa atgaaacaat cagatctggt tcatattgtt	420
aattcttggc aagcattgac tgacgatctc tccagagaag caagagaaaa tcataagcct	480
ttcaaatatg tccaaatatt tgaaaacaaa ggtacagcca tgggttgttc caacttacat	540
ccacatggcc aagcttggtg cttagaatcc atccctagtg aagtttcgca agaattgaaa	600
tcttttgata aatataaacg tgaacacaat actgatttgt ttgccgatta cgtcaaatta	660
gaatcaagag agaagtcaag agtcgtagtg gagaatgaat cctttattgt tgttgttcca	720
tactgggcca tctggccatt tgagaccttg gtcatttcaa agaagaagct tgcctcaatt	780
agccaattta accaaatggt gaaggaggac ctcgcctcga ttttaaagca actaactatt	840
aagtatgata atttatttga aacgagtttc ccatactcaa tgggtatcca tcaggctcct	900
ttgaatgcga ctggtgatga attgagtaat agttggtttc acatgcattt ctacccacct	960
ttactgagat cagctactgt tcggaaattc ttggttggtt ttgaattgtt aggtgagcct	1020
caaagagatt taacttcgga acaagctgct gaaaaactaa gaaatttaga tggtcagatt	1080
cattatctac aaagactgta a	1101

6.15 ilustram a relação de desempenho entre essas duas técnicas.

De maneira geral, nos experimentos realizados, o pior desempenho ficou com a modelagem de mistura de árvores e o melhor com a Jnom que obteve o melhor desempenho em quase todos os testes realizados.

Pode-se observar que a Jnom só teve o seu desempenho vencido em um único caso, na Tabela 6.13 a melhor sensibilidade ficou com o método de árvore, com dependências fixas. Neste caso, a Jnom ficou com a segunda melhor pontuação média. Deve-se notar ainda que a especificidade na Jnom foi a melhor em todos os casos.

Como já comentamos anteriormente, esses experimentos foram todos realizados com os mesmos parâmetros. Isso significa que poderíamos obter resultados melhores se trabalhássemos com ajustes específicos para cada conjunto de dados. Entretanto, visando a uma comparação mais adequada, não alteramos os parâmetros em nenhum dos experimentos realizados para a coleta dos dados ora analisados.

Tabela 6.12. Consolidação da Sensibilidade usando dados gerados a partir de PSSM

Método de Busca	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	68%	65%	64%
Árvores	68%	65%	66%
Mistura de PSSM	67%	61%	53%
Mistura de Árvores	60%	43%	40%
Jnom	71%	65%	67%

Uma vez comprovado o resultado/desempenho da ferramenta Jnom foram realizados

Tabela 6.13. Consolidação da Sensibilidade usando dados gerados a partir de árvores com dependências

Método de Busca	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	68%	68%	67%
Árvores	85%	82%	80%
Mistura de PSSM	70%	66%	67%
Mistura de Árvores	72%	67%	64%
Jnom	71%	55%	75%

Tabela 6.14. Consolidação da Especificidade usando dados gerados a partir de PSSM

Método de Busca	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	56%	51%	57%
Árvores	56%	52%	56%
Mistura de PSSM	55%	48%	46%
Mistura de Árvores	48%	38%	48%
Jnom	71%	82%	81%

Tabela 6.15. Consolidação da Especificidade usando dados gerados a partir de árvores com dependências

Método de Busca	TP = 100, FP = 0	TP = 50, FP = 50	TP = 25, FP = 25
PSSM	65%	65%	63%
Árvores	66%	70%	66%
Mistura de PSSM	64%	59%	57%
Mistura de Árvores	62%	54%	56%
Jnom	71%	76%	88%

experimentos com a família Gal por ser composta de dados biológicos reais. Os resultados obtidos com essa família foram semelhantes em termos de taxa de acerto em relação aos dados sintéticos. Esta foi mais uma forma de ratificar que a modelagem de dependência entre posições pode melhorar a predição de novos *motifs* e a ferramentas Jnom apresentou-se consistente para tal finalidade.

6.7 CONSIDERAÇÕES FINAIS

Comparamos as soluções adotadas existentes e como produto da nossa ponderação para os modelos cognitivos, utilizamos a busca por semelhança para considerar a natureza combinada das proteínas. Conseguimos melhorar sensivelmente o desempenho em relação a técnicas previamente analisadas, principalmente devido a generalização obtida com a implementação das redes bayesianas através do raciocínio baseado em casos usando distância de Hamming. Esta implementação tornou a rede mais flexível, pois as dependências podem ser reveladas através dos exemplos de treinamento e não precisam ser previamente descritas antes do início da aprendizagem como acontece em outras ferramentas.

Este capítulo apresentou uma análise sobre os experimentos realizados com a intenção de mostrar a ratificação de que a combinação é uma característica importante para modelagem e busca de novos *motifs*. Esse fato, nos ajudou a inferir modelos e regras computacionais para treinar o nosso algoritmo de busca e localizar tais *motifs*. Oportunamente mostramos o comportamento da Jnom nos diversos experimentos realizados.

CAPÍTULO 7

COMENTÁRIOS GERAIS

7.1 INTRODUÇÃO

Parte do trabalho foi desenvolver uma ferramenta para auxiliar os cientistas na busca por essas regiões especiais, os *motifs*, no genoma. Pode-se dizer, nesse contexto, que genes com anotação funcional ou coexpressos são genes que se relacionam. Essa relação pode ser usada para auxiliar a predição de sítios de ligação.

Neste capítulo, comentaremos as contribuições, argüiremos sobre trabalhos futuros e então encerraremos com as considerações finais.

7.2 SUMÁRIO DAS CONTRIBUIÇÕES

Durante o decurso deste trabalho, construímos uma ferramenta. Para desenvolvê-la, com padrões internacionais de qualidade, instanciamos um processo baseado em práticas e padrões mundialmente recomendados por modelos de processos consagrados de engenharia de software tais como, RUP[Inc] e XP[Bec]. Nosso processo enfatiza a fase de requisitos e valoriza a arquitetura e construção(implementação) do sistema. Mais detalhes sobre o processo criado pode ser encontrado no Apêndice D.

Em nosso estudo de caso, para a implementação da Jnom, pudemos homologar e aperfeiçoar o processo de desenvolvimento utilizado. Percebemos que este processo se adaptava adequadamente ao tipo de ferramenta almejada. Por isso, deixamo-lo docu-

mentado como parte de nossa contribuição para a comunidade.

Como produto de saída do processo, obtivemos uma ferramenta conforme padrões de qualidade almejados. A ferramenta foi desenvolvida utilizando-se uma combinação de técnicas, tais como redes Bayesianas[FGG97, Fri98] e raciocínio baseado em casos[Bar01, Kol92], visando aumentar o desempenho.

Outra característica importante é a preocupação com a facilidade de uso e integração. Pensando nisso, desenvolvemos a interface gráfica com a tecnologia Applet e estendemos a saída para suportar XML. O Applet é projetado para funcionar em plataforma web sobre vários sistemas operacionais além de ser, na maioria das vezes, graficamente amigável. O XML é um formato textual atualmente adotado como padrão para integração entre sistemas.

7.3 TRABALHOS FUTUROS

Há várias sugestões para melhorar a eficiência do nosso modelo, dentre eles destacamos a exploração de outros tipos de redes Bayesianas, ter uma maior consideração sobre as informações biológicas dos *motifs* e combinar métodos adicionais para otimizar os resultados.

A vantagem de ser capaz de modelar um *motif* usando qualquer rede Bayesiana sugere uma exploração mais profunda de tipos diferentes de modelos, assim como modelos gerais sem restrição. Isto pode incluir extensões representativas que caminham em direção a problemas de complexidade vs. expressionismo, tal como modelos de dependência de contexto específico.

Como nossa estrutura não fez nenhuma presunção no tipo de sítios de ligação, ela pode ser prontamente adaptada para descobrir outras seqüências de *motifs* tal como as

de fatores de remodelagem *splicing*[1] e *histone*[2].

A distância de Hamming, usada para o cálculo da semelhança, suporta atribuição de pesos específicos para intensificar ou enfraquecer ligações entre posições determinadas. Esses pesos podem ser atribuídos nas seqüência em que um especialista já sabe, a priori, se há ou não dependência entre tais posições no conjunto de dados em estudo. Nós não realizamos testes com pesos diferenciados, mas é bastante apropriado flexibilizar o algoritmo com a consideração de pesos nas posições visando melhorar o desempenho das predições.

Um desafio interessante é integrar nosso método com dados adicionais. Podem ser usadas informações prévias sobre a localização do sítio de ligação ou informações sobre a expressividade gênica.

Nossas análises focaram principalmente na eficiência dos resultados. Entretanto, estes resultados também têm implicações interessantes acerca das interações da proteína-DNA. O desafio é como relacionar estas dependências à função e estrutura de proteína. Para este propósito, precisamos ser capazes de avaliar nossa confiança em descobrir dependências e relacionar estas dependências com arranjos tridimensionais de complexos entre proteínas e DNA.

Outro desafio interessante é combinar nosso método com outros tipos de algoritmos, tais como sistemas híbridos[OV99] ou algoritmos genéticos[dM01, Mit96]. Outros algoritmos baseados em técnicas de Inteligência Artificial podem ser usados para tentar melhorar o desempenho[RN04]. Uma pequena revisão sobre sistemas híbridos e algoritmos genéticos podem ser encontrados nos apêndices A e B.

Pode-se ainda, para uma abordagem mais estatística, submeter os resultados a teste

[1] Procedimento de remoção de introns e ligação de exons para formar uma cadeia contínua[spl].
[2] Proteína básica importante no sistema de controle repressor-ativador do DNA[his].

de hipótese para análise dos experimentos[Mey83]. Com esses testes, pode-se obter resultados mais relevantes estatisticamente. A escolha do tipo adequado do teste depende diretamente do objetivo da pesquisa na qual ele será aplicado. Uma abordagem bastante comum é aplicar diversos testes até que um deles obtenha um achado estatisticamente relevante. Os testes de hipótese mais comuns são os paramétricos e não-paramétricos[Mey83].

7.4 CONSIDERAÇÕES FINAIS

Com a fusão das técnicas de redes Bayesianas e do raciocínio baseado em casos, usando a natureza combinatorial dos fatores de transcrição, pudemos melhorar sensivelmente o desempenho em relação a técnicas previamente analisadas, principalmente devido a generalização obtida com a implementação das redes bayesianas através do raciocínio baseado em casos usando distância de Hamming. Esta implementação torna a rede mais flexível, uma vez que as dependências são descobertas com os próprios exemplos de treinamento e não precisam ser impostas no início da aprendizagem como acontece em outras ferramentas.

Implementamos uma ferramenta que é capaz de encontrar novos *motifs* em seqüências desconhecidas e em ambientes onde os dados de treinamento são escassos. Ratificamos a suposição feita em trabalhos anteriores que modelar dependência melhora o desempenho e produzimos uma ferramenta que pode ser facilmente expandida ou integrada com outros sistemas a fim de se realizar experimentos mais específicos devido aos padrões utilizados.

Por fim, deixamos a semente de um processo de desenvolvimento padrão instanciado. Este processo pode ser adaptado para o desenvolvimento de novas ferramentas ou para a evolução e integração de outras ferramentas já existentes e não apenas a Jnom.

REFERÊNCIAS BIBLIOGRÁFICAS

[AB98] P Agarwal and V Bafna. Detecting non-adjacent correlations within signals in dna. *RECOMB*, 1998.

[ABJ+99] B Alberts, D Bray, A Johnson, J Lewis, M Raff, K Roberts, P Walter, et al. *Fundamentos da biologia celular: uma introdução à biologia molecular da célula*. ISBN 85-7307-494-9, 1999.

[AGS02] S. Ahmad, M. M. Gromiha, and A. Sarai. Prediction of DNA binding in proteins from composition, sequence and structure, 2002.

[app] *Java Applet Tutorial*. http://www.realapplets.com/tutorial/.

[APR99] R. E. Anderson, V.S. Pande, and C.J. Radke. Dynamic lattice Monte Carlo simulation of a model protein at an oil-water interface. AIChE Annual Meeting, 1999.

[B+02] P Berman et al. Exploiting transcription factor binding site clustering to identify cis-regulatory modules involved in pattering formation in the Drosophila genome. *PNAS*, vol. 99(no. 2), 2002. 757-762.

[Bai99] T. L. Bailey. Meme – multiple em for motif elicitation, 1999. www.sdsc.edu/MEME.

[Bar01] Jorde Muniz Barreto. *Inteligência artificial no limiar do século XXI*. Ed 3, 2001.

[BDS93] M Barrett, MJ Donoghue, and E Sober. Against consensus, 1993. Syst. Zool. 40, 486-493.

[Bec] Kent Beck. Extreme programming: A gentle introduction. http://www.extremeprogramming.org/.

[BEFK] Yoseph Barash, Gal Elidan, Nir Friedman, and Tommy Kaplan. Supplementary information for modeling dependencies in protein-DNA binding sites. http://www.cs.huji.ac.il/labs/compbio/TFBN/.

[BEFK03] Yoseph Barash, Gal Elidan, Nir Friedman, and Tommy Kaplan. Modeling dependencies in protein-DNA binding sites. *Seventh Annual Inter. Conf. on Computational Molecular Biology (RECOMB)*, 2003.

[BFB+01] Y Barash, N Friedman, G Bejerano, et al. A simple hyper-geometric approach for discovering putative transcription factor binding sites, 2001. WABI 2001.

[BG03] E. Blanco and Messeguer R. Guigó. Alignment of promoter regions by mapping nucleotide sequences into arrays of transcription factors binding motifs. *RECOMB*, 2003.

[bio] http://biolab.cin.ufpe.br/.

[BMV03] Ricardo Bringas, Thomas Manke, and Martin Vingron. Correlating protein-DNA and protein-protein interactions of transcription factors. *RECOMB*, 2003.

[BT99a] C Branden and J Tooze. Introduction to protein structure, 1999. 2nd Edition Garland Pub.

[BT99b] C Branden and J Tooze. Introduction to protein structure, 1999.

[CA90] Lawrence CE and Reilly AA. An expectation maximization (em) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *PUBMED*, 1990. http://www.ncbi.nlm.nih.gov.

[CH97] DM Chickering and D Heckerman. Efficient approximations for the marginal likelihood of bayesian networks with hidden variables. *Mach. Learn.*, 1997.

[Cha91] Philip K. Chan. Machine learning in molecular biology sequence analysis. Technical Report CUCS-041-91, 1991.

[cpl] *The cplusplus.com tutorial: Complete C++ language tutorial*. http://www.cplusplus.com/doc/tutorial/.

[CS04] Ernesto Costa and Anabela Simões. *Inteligência Artificial - Fundamentos e Aplicações*. ISBN: 972-722-269-2, 2004.

[CV98] Gerry Coleman and Renaat Verbruggen. A quality software process for rapid application development. *Software Quality Journal 7*, 1998.

[Dav01] E Davidson. Genomic regulatory systems. *Academic Press.*, 2001.

[DGW+96] Yuefan Deng, James Glimm, Yuan Wang, Alex Korobka, Moshe Eisenberg, and Arthur P. Grollman. Prediction of protein binding to DNA in the presence of water-mediated hydrogen bonds, 1996.

[dM01] Marcio Nunes de Miranda. *Algoritmos Genéticos: Fundamentos e Aplicações*. http://www.gta.ufrj.br/ marcio/genetic.html, 2001.

[dUMdFdLR+02] Cláudio de U. Magnabosco, Carina U. de Faria, Arcadio de Los Reyes, Raysildo B. Lôbo, Vanessa Barbosa, and Roberto Sainz. Implementação da amostragem de gibbs na estimação de parâmetros genéticos para peso ao desmame na raça nelore utilizando diferentes esquemas de cadeia amostral. *Embrapa Cerrados*, 2002.

[FGG97] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997.

[Fri98] Nir Friedman. The Bayesian structural EM algorithm. *Fourteenth Conf. on Uncertainty in Artificial Intelligence (UAI)*, 1998.

[FRS01] Cristian Follmer, Cristina Russo, and Evelyn Schroeder. Modelagem molecular, 2001. Universidade Federal do Rio Grande do Sul.

[GK95] S. Goonatilake and S. Khebbal. Intelligent hybrid systems: Issues, classifications and future directions. intelligent hybrid systems. *John Wiley & Sons Ltd*, 1995.

[GS01] D GuhaThakurta and G Stormo. Identifying target sites of cooperatively binding factors. *Bioinformatics*, 17(7):608–621, 2001.

[his] *Drug Discovery and development*. http://www.dddmag.com.

[HL02] S Hannenhalli and S Levy. Prediction transcription factory synergism. *Nucleic Acids Research*, 20(19), 2002. pp 4278-4284.

[HS99] G. Z. Hertz and G. D. Stormo. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics, Vol 15, 563-577*, 1999.

[Inc] Rational Inc. Rational unified process. http://www-306.ibm.com/software/rational/.

[iso] *International Organization for standardization - ISO*. http://www.iso.org/iso/en/ISOOnline.frontpage.

[jav] *The Java Tutorial*. http://java.sun.com/docs/books/tutorial/applet/.

[Ján] Gergely János. Hmm - hidden markov model. http://jedlik.phy.bme.hu/ gerjanos/HMM/node4.html.

[Kol92] Janet L. Kolodner. An introduction to case-based reasoning. *Artificial Intelligence Review*, 1992.

[Lat03] David S. Latchman. *Eukaryotic Transcription Factors*. University of London, 4th edition, 2003. ISBN 0124371787.

[LBL01a] X Liu, D Brutlag, and J Liu. An algorithm for finging protein-DNA binding sites with applications to chromatin-imuoprecipitation microarray experiments. *Nature Biotechnology*, 20, 2001. pp. 835-839.

[LBL01b] X Liu, D Brutlag, and J Liu. Bioprospector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pacific Symposium on Biocomputing*, 2001. 6:127-138.

[Leh86] E Lehman. *Testing Statistical Hypothesis*. Springer-Verlag, 2 edition, 1986.

[Lew97] B Lewin. Genes vi, 1997. Oxford University Press, Oxford.

[M+02] M Markstein et al. Genome-wide analysis of clustered dorsal sites identifies putative target genes in the Drosophila embryo. *PNAS*, 99(2), 2002. 763-768.

[McC96] Steve McConnell. *Rapid Development*. ISBN 1-55615-900-5, 1996.

[Mer26] K. S. Mereschkovsky. *Symbiogenesis and the Origin of Species*. publisher, 1926.

[Mey83] P. L. Meyer. *Testes de Hipóteses, Probabilidade -Aplicações à Probabilidade - Aplicações à Estatística*. Livros Técnicos e Científicos. Rio de Janeiro - RJ, 2ed. edition, 1983. Testes de Hipóteses.

[Mid03] Jack Middleton. Combinatorial relationships in transcription factors. *BIOC 218*, March 2003.

[Mit96] M. Mitchell. An introduction to genetic algorithms. *MIT Press*, 1996.

[MKS92] S. Muggleton, R. D. King, and M. J. Sternberg. Protein engineering, 1992.

[MM+04] Alexandra Carniel Perdigao Maia, Roberto Michelan, et al. Algoritmos genéticos, 2004. http://www.din.uem.br/ia/geneticos/.

[MSS+02] Eduardo Massad, Koichi Sameshima, Paulo Sérgio Panse Silveira, et al. Herança (introdução à genética quantitativa), 2002. http://medicina.fm.usp.br/dim/genquant/index.php.

[Oss00] Antônio Carlos Osse. A proteína e a cromatina, 2000. http://www1.folha.uol.com.br/folha/educacao/ult305u5855.shtml.

[OV99] Fernando Santos Osório and Renata Vieira. *Sistemas Hibridos Inteligentes*. http://www.inf.unisinos.br, 1999.

[P+01] Y Pipel et al. Identifying regulatory networks by combinatorial analysis of promoter elements. *Nature Genetics*, vol. 29, 2001.

[PPRB01] Gianluca Pollastri, Darisz Przybylski, Burkhard Rost, and Pierre Baldi. Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles, 2001.

[PPRB02] Gianluca Pollastri, Darisz Przybylski, Burkhard Rost, and Pierre Baldi. Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. *InterScience*, 2002. http://promoter.ics.uci.edu/BRNN-PRED/.

[Pre] Oxford University Press. The embl data library. http://nar.oupjournals.org/cgi/content/abstract/14/1/5.

[Rez02] Solange Oliveira Rezende. *Sistemas Inteligentes - Fundamentos e Aplicações*. 01/11/2002, 2002. ISBN: 8520416837.

[RN04] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Hardcover, 2004.

[Sch97] TD Schneider. Information content of individual genetic sequences, 1997. J. Theor. Biol. 189 (4), 427-441.

[spl] *Imacculata University: bioinformatics glossary.* http://www.immaculata.edu.

[SSG+01] Akinori Sarai, Samuel Selvaraj, M. Michael Gromiha, Joerg-Gerald Siebers, Ponraj Prabakaran, and Hidetoshi Kono. Target prediction of transcription factors: Refinement of structure-based method, 2001.

[SSG03] A. Sarai, S. Selvaraj, and M. M. Gromiha. Target prediction of transcription factors: Application of structure-based method to yeast genome, 2003.

[Sto01] G Stormo. DNA binding sites: representation and discovery. *Bioinformatics*, 16(1):16–23, 2001.

[vHRCV00] Jacques van Helden, Alma. F. Rios, and Julio Collado-Vides. Discovering regulatory elements in non-coding sequences by analysis of spaced dyads. *Nucleic Acids Research*, 2000.

[vWvW03] Christiane Gresse von Wangenheim and Aldo von Wangenheim. *Título Raciocínio Baseado em Casos.* ISBN 1459-1, 2003.

[W+01] E Wingender et al. The TRANSFAC system on gene expression regulation, 2001.

[Wag99] A Wagner. Genes regulated cooperatively by one ore more transcription factors and their identification in whole eukaryotic genomes. *Bioinformatics*, vol. 15 no.(776-784), 1999.

[Wal27] Ivan Wallin. *Symbionticism and the Origins of Species.* publisher, 1927.

[WS03] Jane Wood and Denise Silver. *Joint Application Development*. ISBN 0-04299-4, 2003.

[xml] *Extensible Markup Language (XML)*. http://www.w3.org/XML/.

[yea] http://seq.yeastgenome.org.

[ZDSS99] M Zheng, B Doan, TD Schneider, and G Storz. Oxyr and soxrs regulation of fur, 1999. J. Bacteriol. 181, 4639-4643.

APÊNDICE A

RACIOCÍNIO BASEADO EM CASOS

Raciocínio Baseado em Casos nada mais é do que um método de soluções de problemas usando adaptações de soluções anteriores similares a estes problemas. Sistemas baseados em conhecimento podem adaptar soluções conhecidas para encontrar novas soluções. Podem usar casos conhecidos para explicar novas situações e criticar novas soluções. Raciocínios anteriores para interpretar uma nova situação ou criar uma solução apropriada para um novo problema[Bar01, Kol92]. O raciocínio baseado em casos também é usado extensivamente para raciocínio de senso comum no dia-a-dia. Por exemplo, quando planejamos nossas atividades, nos lembramos o que funcionou e o que falhou, e usamos isso para criar nossos planos. Quando montado um sistema, cada diferente solução ou interpretação do problema, é um novo caso. Em nossa abordagem cada novo motif é considerado um caso.

Para a construção de um Sistema Baseado em Conhecimento, necessitamos de uma forma coerente de representação do conhecimento. O conhecimento é representado na forma de caso. Um caso é a principal parte do conhecimento nos sistemas RBC. Este pode ser entendido como a abstração de uma experiência descrita em termos de seu conteúdo e contexto, podendo assumir diferentes formas de representação. A representação dos casos é uma tarefa complexa e importante para o sucesso do sistema RBC. Em nosso caso utilizamos a matriz de freqüência para armazenar os casos.

APÊNDICE B

SISTEMAS HÍBRIDOS

Sistemas Híbridos são algoritmos inteligentes que utilizam diferentes métodos de processamento de informações, ou seja, resultam da combinação de duas ou mais técnicas distintas, sendo pelo menos uma delas de inteligência artificial(IA), para resolver um dado problema[OV99]. A principal idéia do desenvolvimento de Sistemas Híbridos é que uma única técnica, em razão de suas limitações ou deficiências, pode não ser capaz de, por si só, resolver um dado problema. Para isso a combinação de várias técnicas pode levar a uma solução mais robusta e eficiente. Podem ser citadas como técnicas de IA: Redes Neurais, Lógica Fuzzy, Algoritmos Genéticos, Raciocínio Baseado em Casos, Raciocínio Baseado em Regras, Redes Semânticas, Redes Bayesianas, etc.

Goonatilake e Khebbal [GK95] propuseram um esquema de classificação para sistemas híbridos composto de três classes, levando em consideração funcionalidade, arquitetura de processamento e requerimentos de comunicação. As três categorias são:

- **Substituição de Função:** Utiliza-se uma técnica para implementar uma função de outra técnica. Essa forma de hibridismo não acrescenta nenhuma funcionalidade ao sistema inteligente.
- **Híbridos Intercomunicativos:** Usada para problemas complexos, onde podem ser divididos em subtarefas independentes. Cada subtarefa usa uma técnica inteligente.
- **Híbridos Polimórficos:** Uma única técnica é adaptada para realizar uma tarefa

inerente à outra técnica tendo como motivação a descoberta de novas funcionalidades e o entendimento do relacionamento de diferentes técnicas.

As seguintes fases compõem o ciclo de desenvolvimento de Sistemas Híbridos: análise do problema (identificação de subtarefas e propriedades do problema), casamento das propriedades, Seleção da categoria de hibridismo, implementação, validação e manutenção [Rez02].

Podemos citar alguns exemplos de combinações que podem ser úteis ao nosso problema: Redes Neurais (técnica fortemente baseada em dados) com Lógica Fuzzy (técnica fortemente baseada em conhecimento), Raciocínio Baseado em Casos com Redes Neurais, Algoritmos Genéticos com Redes Neurais, Redes Neurais com Estatística, Combinações de Classificadores (simbólicos e estatísticos), Redes Neurais com Linguagens Formais, etc.

APÊNDICE C

ALGORITMOS GENÉTICOS

Podemos destacar, em nossos trabalhos futuros com Sistemas Híbridos, os Algoritmos Genéticos que são uma subárea da área de Computação Evolutiva. São sistemas para a resolução de problemas que utilizam modelos computacionais baseados na teoria da evolução natural [dM01, Mit96]. São algoritmos de otimização global, baseados nos mecanismos de seleção natural e genética. Favorecem a combinação dos indivíduos mais aptos, dessa forma os indivíduos menos aptos tenderão a desaparecer. Um mecanismo de reprodução, baseado em processo evolutivo, é aplicado sobre a população atual com o objetivo de explorar o espaço de busca e encontrar melhores soluções para o problema[CS04].

Diagrama geral do ciclo de vida de um Algoritmo Genético Tradicional:

i) t = 0;

ii) Gerar população inicial P(0);

iii) Avaliar aptidão de cada indivíduo i da população atual P(t);

iv) Enquanto critérios de paradas não forem satisfeitos:

v) t = t + 1;

vi) Selecionar população P(t) a partir de P(t - 1);

vii) Aplicar operadores de cruzamento sobre P(t);

viii) Aplicar operadores de mutação sobre P(t);

ix) Avaliar P(t);

A representação dos dados pode ser feita inspirada na biologia. O cromossomo é uma estrutura de dados, geralmente vetores ou cadeias de valores binários, que representa uma possível solução do problema a ser codificado. O conjunto de todas as configurações que o cromossomo pode assumir forma o seu espaço de busca.

Como esse tipo de algoritmo é inspirado na teoria da seleção natural, ou seja, os melhores indivíduos possuem maior aptidão, são selecionados para gerar filhos através de crossover e mutação. A aptidão é baseada no valor da função de custo, que é especificada para cada problema, e mede o quão boa é a solução codificada por um indivíduo dirigindo o AG para as melhores regiões do espaço de busca[MM+04].

São Métodos de Seleção:

- **Roleta:** Método de seleção mais simples e mais utilizado. Cada indivíduo da população é representado na roleta por uma fatia proporcional ao seu índice de aptidão.
- **Torneio:** Escolhe-se **n** indivíduos aleatoriamente da população e o cromossomo com maior aptidão é selecionado, a cada iteração.
- **Amostragem universal estocástica:** Variação do Método da Roleta. Ao invés de uma única agulha, **n** agulhas são igualmente especadas assim a roleta é girada uma única vez.

Os operadores genéticos são necessários para a geração de populações sucessivas, estendendo a busca até chegar a um resultado satisfatório. Temos os seguintes operadores genéticos:

- **Mutação:** Alteração arbitrária de um ou mais componentes de uma estrutura escolhida (inversão dos valores dos bits), é um operador necessário para a introdução e manutenção da diversidade genética da população.

- **Crossover:** Permitem que as próximas gerações herdem características através da recombinação dos pais. O cruzamento pode ser de um ponto, multiponto ou uniforme. O crossover ou mutação podem destruir o melhor indivíduo, assim como prevenção de perca da melhor solução encontrada o Elitismo transfere a cópia do melhor indivíduo encontrado para a geração seguinte.

- **Parâmetros Genéticos:** Vários parâmetros influenciam o desempenho de um Algoritmo Genético. Dentre eles podemos citar: Tamanho da População, Taxa de Cruzamento, Taxa de Mutação, Intervalo de Geração, Critério de Parada. Portanto, é importante a análise de como estes podem ser utilizados diante das necessidades do problema e dos recursos disponíveis.

Podemos destacar as aplicações dos Algoritmos Genéticos com sucesso a uma grande variedade de problemas de otimização e busca. São apropriados para problemas de difícil otimização pelas técnicas convencionais, quando não existe outra técnica específica para resolver o problema [MM+04].

APÊNDICE D

O PROCESSO DE CONSTRUÇÃO

D.1 INTRODUÇÃO

Para se desenvolver um projeto de software deve-se levar em consideração os principais conceitos da engenharia de software necessários para construí-lo de forma adequada. A meta é garantir a produção de software de alta qualidade que atenda às necessidades dos usuários dentro de um cronograma previsível.

O desenvolvimento é o processo de criação de um sistema a partir dos requisitos por isso esse fluxo é tão importante. Depois que os sistemas tiverem passado pelo ciclo de desenvolvimento inicial, os desenvolvimentos subseqüentes serão o processo de adaptação do sistema aos requisitos novos ou modificados. Isso se aplica durante todo o ciclo de vida do sistema.

O processo de engenharia de software é o modelo de desenvolvimento do sistema. Daí a importância de se utilizar os conceitos e processos definidos pela engenharia de software. Devido à qualidade desejada à ferramenta, não poderia faltar tal preocupação. Mais adiante, conheceremos um pouco do processo desenvolvido especialmente para criação da Jnom. Este processo derivou de uma série de boas práticas consagradas pelos modelos de processo RUP[Inc] e XP[Bec]. Nós adaptamos algumas das práticas mais adequadas ao tipo de projeto deste trabalho.

D.2 FLUXO DE REQUISITOS

O principal objetivo do fluxo de requisitos é guiar o desenvolvedor a obter um sistema adequado às necessidades dos usuários. Para isso, é preciso elicitar os requisitos do sistema, isto é, as funcionalidades e características (features) que o sistema deve possuir (requisitos) e como o usuário interage com o sistema (casos de uso).

As atividades do fluxo de requisitos se iniciam durante a iteração preliminar, na fase de concepção. O levantamento inicial de requisitos tem o objetivo de entender o que é o sistema solicitado e suas principais funcionalidades. As demais atividades do fluxo de requisitos acontecem de forma incremental, analisando os requisitos e prototipando as interfaces a cada nova iteração. Essas atividades se estendem, normalmente, até o início da fase de construção.

D.3 TÉCNICAS DE ELICITAÇÃO DE REQUISITOS

De forma sucinta, requisitos de um sistema definem os serviços que o sistema deve oferecer e as restrições aplicáveis à sua operação. Por exemplo: necessidades de determinada plataforma ou configuração mínima de hardware e software. Os requisitos de software são aqueles dentre os requisitos de sistema que dizem respeito a propriedades do software. Tradicionalmente, os requisitos de software são classificados em requisitos funcionais e não-funcionais.

Os requisitos funcionais são as declarações das funções que o sistema deve oferecer, como o sistema se comporta com entradas particulares e como o sistema deve se comportar em situações específicas. O termo função é usado no sentido genérico da operação que pode ser realizada pelo sistema, seja por meio de comandos dos usuários, ou seja, pela ocorrência de eventos internos ou externos ao sistema. Em alguns casos, os

requisitos funcionais podem também explicitamente definir o que o sistema não deve fazer.

Os requisitos não-funcionais são as restrições nas funções oferecidas pelo sistema. Incluem restrições de tempo, restrições no processo de desenvolvimento, padrões, e qualidades globais de um software, como manutenibilidade, usabilidade, desempenho, custos e várias outras. Como exemplo desse tipo de requisito podemos citar: o tempo de resposta do sistema não deve ultrapassar 30 segundos ou o software deve ser operacionalizado no sistema Linux.

O propósito destas técnicas é a identificação, a especificação e o gerenciamento de todos os requisitos do projeto.

D.3.1 JAD - Joint Application Design

JAD (Joint Application Design ou Arquitetura de Aplicações em Conjunto) é um agrupamento de ferramentas destinadas a apoiar o desenvolvimento de sistemas de Informática (Application) nas fases de levantamento de dados, modelagem e análise (Design). Essas fases são realizadas pelos analistas de sistemas (fornecedores) em conjunto (Joint) com os usuários da Aplicação[WS03].

O JAD teve origem nos laboratórios de Software da IBM, no Canadá, no final dos anos 60. Durante esse tempo vem sendo ajustado às novas tecnologias emergentes.

JAD é um ciclo programado de reuniões nas quais analistas e usuários arquitetam uma aplicação. Em JAD há uma atenção especial dirigida ao assunto condução de reuniões.

Por definição, JAD é uma entidade abstrata. Para que o JAD se realize, para que se concretize, a implementação se dá através de uma reunião.

A técnica permeia todas as fases do método de desenvolver sistemas, desde o Levantamento Preliminar até a implantação do sistema e o treinamento dos usuários. O foco principal está na identificação de objetivos e modelagem conceitual, atividades de extrema importância para a realização de soluções que atendam às necessidades dos usuários.

D.3.2 RAD - Rapid Application Development

Rapid Application Development é um método dinâmico de desenvolvimento de sistemas (DSDM[CV98]). O RAD se aplica a projetos que têm prazos curtos, e que em geral envolvem o uso de prototipagem e ferramentas de desenvolvimento de alto nível como foi o nosso caso [CV98, McC96]. O RAD existe para facilitar o desenvolvimento de aplicações com esta característica. Esta metodologia combina o JAD (para definir rapidamente a especificação do sistema) com o uso de ferramentas CASE e de metodologias de prototipação, para chegar a um produto final em menor tempo. Vale esclarecer que o DSDM é o progenitor do Extreme Programing (XP)[Bec].

D.4 ESPECIFICAÇÃO DE CASOS DE USO

Esta atividade é uma concretização da atividade de especificação de requisitos que surge no detalhamento dos requisitos. Nesta atividade, os requisitos são efetivamente detalhados na forma de casos de uso, onde uma notação e uma estruturação tecnicamente apropriadas para o entendimento da equipe técnica de desenvolvimento são empregadas.

O objetivo do detalhamento dos requisitos serve para descrever a interação dos atores com o sistema, isto é, como os casos de uso começam, terminam e interagem com os atores. Um ator representa qualquer entidade externa ao sistema que interaja com ele para obter ou fornecer informações.

Um fator importante na hora de identificar casos de uso a partir dos requisitos funcionais é a granularidade dos detalhes. Granularidade muito alta (quantidade muito grande de casos de uso) dificulta o entendimento do documento de especificação de casos de uso e, em muitos casos, interfere fortemente na produtividade. Para melhorar a granularidade o analista deve produzir os fluxos secundários.

O fluxo de eventos principal descreve os passos que compõem o caso de uso, ou seja, descreve os eventos que ocorrem durante sua execução, para que ele possa realizar seus objetivos. Normalmente o caso de uso inicia quando o usuário faz algo, como escolher uma das opções do menu, por exemplo. O fluxo de eventos, então, descreve o que o usuário faz e o que o sistema faz em resposta, sendo estruturado na forma de um diálogo entre o usuário e o sistema.

Normalmente, o fluxo de eventos necessário para realizar uma determinada funcionalidade do sistema contém várias exceções ou caminhos alternativos. Os fluxos secundários descrevem o que deve ocorrer em situações como essas, que não representam o funcionamento normal ou rotineiro do caso de uso. Eles dividem-se em fluxos alternativos e fluxos de exceção. Os fluxos alternativos descrevem situações especiais e os fluxos de exceção descrevem situações de erro. Quando um fluxo secundário termina, os eventos do fluxo principal são retomados, a menos que se especifique o contrário.

D.5 FLUXO DE CONSTRUÇÃO

O objetivo do fluxo de construção é esclarecer os requisitos restantes e concluir o desenvolvimento do sistema. Para isso, adotamos a arquitetura em camadas como detalhado a seguir.

A separação do código em camadas independentes permite que se troque a tecnologia utilizada sem afetar as regras de negócio do sistema. Por exemplo pode-se alternar a

tecnologia da interface gráfica (Swing por HTML, por exemplo)[1] ou para o armazenamento dos dados (JDBC por ODBC[2]) . Isso facilita a reusabilidade das classes em outros projetos e permite maior flexibilidade na escolha de tecnologias para implementar a aplicação. O meio de armazenamento, por exemplo, poderia ser trocado de arquivos para um SGBD com uso de JDBC. Além disso, a própria tecnologia utilizada para um mesmo meio de armazenamento poderia ser facilmente alterada. Um exemplo disso seria a troca de JDBC por EJB para armazenamento de dados em um SGBD.

Na figura D.1 descreve-se quais são as camadas padrão da aplicação e que serviços cada uma delas deve apresentar. A tecnologia Java foi utilizada, porém grande parte das recomendações aqui presentes é genérica a ponto de poder ser seguida com outras tecnologias.

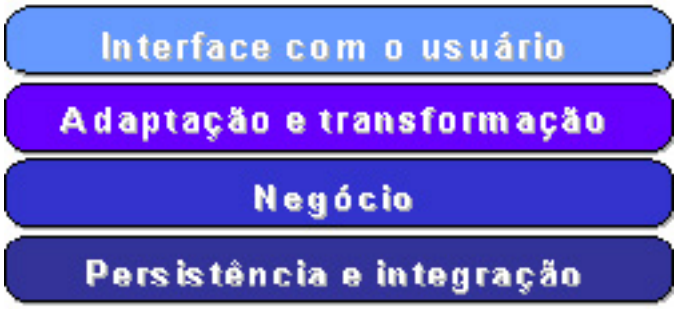


Figura D.1. As quatro camadas independentes da arquitetura

D.5.1 Camada de Interface com o Usuário

A interface com o usuário, que por simplicidade chamaremos de GUI, Graphical User Interface, é a camada de apresentação da aplicação, responsável pela criação e disposição dos objetos gráficos através dos quais o usuário interage com o sistema. Applets e componentes de APIs de interface gráfica, como Swing, são elementos desta camada.

[1] Swing e HTML são exemplos de tecnologias para interface com o usuário.
[2] JDBC e ODBC são exemplos de tecnologias para comunicação com banco de dados.

D.5.2 Camada de Adaptação e Transformação

Esta camada é responsável pelas conversões de formato das informações trocadas entre as camadas de negócio e a GUI, permitindo, por exemplo, que uma mesma informação seja exibida de forma diferente dependendo do usuário. Ela também pode ser responsável pela comunicação distribuída entre os componentes da GUI e da camada de negócios em aplicações cliente-servidor.

D.5.3 Camada de Negócio

Esta camada é o núcleo do sistema, responsável por implementar a lógica do negócio. Nela estão todas as classes inerentes ao domínio da aplicação. Os elementos mais básicos de aplicação estão nesta camada e são chamados de classes básicas. As classes básicas representam objetos básicos manipulados pelo sistema e que normalmente são, direta ou indiretamente, persistidos pelo sistema. Existem também as coleções de negócio, que são os elementos que representam conjuntos (repositórios) de classes básicas e são responsáveis pelo armazenamento, temporário ou não, das instâncias destas classes. Nas coleções de negócio estão encapsuladas regras de negócio associadas ao armazenamento e manipulação das instâncias. Elas podem encapsular as verificações e validações inerentes ao negócio (escolha do arquiteto) como, por exemplo, verificar se uma determinada instância já está criada antes de criá-la.

D.5.4 Camada de Persistência e Integração

As classes desta camada são responsáveis pela manipulação da estrutura física de armazenamento dos dados e pela integração com sistemas externos. São elas que isolam o resto do sistema do meio de armazenamento usado (memória, arquivos textos, XML, SGBD, etc.), de maneira que, se o meio de armazenamento for trocado, apenas as classes desta camada terão que ser modificadas ou substituídas.

A camada de negócio utiliza os serviços desta camada para acessar e persistir

instâncias das classes básicas. Os serviços de persistência oferecidos podem ser implementados de forma transparente ao resto da aplicação, pois os detalhes estão encapsulados nesta camada. A principal utilidade desta camada é isolar o mecanismo de armazenamento utilizado das demais classes da aplicação.

Dissertação de Mestrado apresentada por **José Edson de Albuquerque Filho** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título , **"Jnom: Uma Ferramenta para Encontrar Motifs"**, orientada pela **Profa. Katia Silva Guimarães** e aprovada pela Banca Examinadora formada pelos professores:

Profa. Patrícia Cabral de Azevedo Restelli Tedesco
Centro de Informática / UFPE

Prof. Marcos Antonio de Morais Junior
Departamento de Genética / UFPE

Profa. Katia Silva Guimarães
Centro de Informática / UFPE

Visto e permitida a impressão.
Recife, 27 de junho de 2005.

Prof. JAELSON FREIRE BRELAZ DE CASTRO
Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.