Universidade Federal do Rio Grande - FURG
Centro de Ciências Computacionais - C3
Programa de Pós-Graduação em Computação
Mestrado em Engenharia de Computação

Thyago Salvá

Inferência de Redes Regulatórias utilizando Algoritmos de Estimação de Distribuição

Rio Grande/RS

28 de março de 2014

Thyago Salvá

Inferência de Redes Regulatórias utilizando Algoritmos de Estimação de Distribuição

Dissertacão de Mestrado apresentada ao Programa de Pós-Graduação em Computaçãoda Universidade Federal do Rio Grande - FURG, como requisito parcial para a obtencão do grau de Mestre em Engenharia de Computação.

Orientador(a): Prof. Dr. Adriano Velasque Werhli

Coorientador(a): Prof. Dr. Leonardo Ramos Emmendorfer

FURG

Rio Grande/RS

2014

THYAGO SALVÁ

Inferência de Redes Regulatórias utilizando Algoritmos de Estimação de Distribuição

Dissertacão de Mestrado apresentada ao Programa de Pós-Graduação em Computaçãoda Universidade Federal do Rio Grande - FURG, como requisito parcial para a obtencão do grau de Mestre em Engenharia de Computação.

Aprovado em: 28 03 2014

BANCA EXAMINADORA

Adriano V. Werhli

Prof. Dr. Adriano Velasque Werhli (Orientador)
Centro de Ciências Computacionais – FURG

Prof. Dr. Leonardo Ramos Emmendorfer (Coorientador)
Centro de Ciências Computacionais – FURG

Prof. Dr. Rogério Malta Branco
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul – IFRS

Karina Machado

Profa. Dra. Karina dos Santos Machado
Centro de Ciências Computacionais – FURG

Profa. Dra. Sílvia Silva da Costa Botelho
Centro de Ciências Computacionais – FURG

Dedico mais essa conquista à minha família e à minha noiva,
por todo amor, carinho, incentivo e compreensão pelos
momentos de ausências durante a relização deste trabalho.

Agradecimentos

Em primeiro lugar, a Deus, pela vida, inteligência e por iluminar meu caminho na busca dos meus objetivos.

Aos meus pais, Santiago Gonçalves Salvá e Rosi Salvá, meus irmãos, Thayane Salvá e Thaygor Salvá, meus avós e minha noiva, Caroline Tomasini, por todo amor incondicional, carinho, conforto nos momentos difíceis e principamente, por sempre acreditarem em mim.

Ao meu orientador, Professor Dr. Adriano Velasque Werhli e ao meu coorientador, Professor Dr. Leonardo Ramos Emmendorfer meu sincero agradecimento pela disponibilidade, dedicação, ideias e esclarecimentos além da liberdade e confiança em relação a este trabalho.

Aos Professores Dr. Rogério Malta Branco, Dra. Karina dos Santos Machado e Dra. Silvia Silva da Costa Botelho, membros da Banca Examinadora, por terem aceito ao convite para desempenhar este papel, dispondo de seu tempo e conhecimento para analisar este trabalho.

E aos demais que de alguma forma contribuíram na elaboração desta dissertação.

"Alguns homens vêem as coisas como são, e dizem 'Por quê?' Eu sonho com as coisas que nunca foram e digo 'Por que não?'"

Geroge Bernard Shaw

Resumo

Inferência de redes de regulação gênica a partir de dados de expressão esparsos e na presença de ruído ainda é um desafio nos dias de hoje. Nesse trabalho, foram utilizados dois Algoritmos de Estimação de Distribuição distintos para inferir uma Rede de Regulação Gênica. A fim de avaliá-los, os algoritmos foram aplicados em três tipos de dados: (i) dados simulados a partir de uma distribuição Gaussiana multivariada, (ii) dados simulados a partir de um simulador realista, GeneNetWeaver e (iii) dados a partir de experimentos de citometria de fluxo. Os métodos de inferências em questão apresentam um desempenho comparável com algoritmos de inferência tradicionais em termos de precisão na reconstrução da rede.

Palavras-chave: Redes de Regulação Gênica, Algoritmo de Estimação de Distribuição, Redes Bayesianas.

Abstract

Inference of Gene Regulatory Networks from sparse and noisy expression data is still a challenge nowadays. In this work two different Estimation of Distribuition Algorithms were used to infer Gene Regulatory Networks. In order to evaluate them, the algorithms were applied to three types of data: (i) data simulated from a multivariate Gaussian distribution, (ii)data simulated from a realistic simulator, GeneNetWeaver and (iii) data from flow cytometry experiments. The proposed inference method shows a performance comparable with traditional inference algorithms in terms of the network reconstruction accuracy.

Keywords: Gene Regulatory Networks, Estimation of Distribution Algorithm, Bayesian Networks

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AUC	*Area Under the ROC Curve*
BBs	*Building Blocks*
BIC	*Bayesian Information Criterion*
BMDA	*Bivariate Marginal Distribution Algorithm*
BN	*Bayesian Network*
BOA	*Bayesian Optimization Algorithm*
cGA	*Compact Genetic Algorithm*
DAG	*Directed Acyclic Graph*
DNA	*Deoxyribonucleic acid*
EAs	*Evolutionary Algorithms*
EC	*Evolutionary Computation*
EBNA	*Estimation of Bayesian Network Algorithm*
ECGA	*Bivariate Marginal Distribution Algorithm*
EDAs	*Estimation Distribution Algorithms*
EGA	*Equilibrium Genetic Algorithm*
FDA	*Factorized Distribution Algorithm*
FN	*False Negative*
FP	*False Positive*

GAs	*Genetic Algorithms*
GNW	*GeneNetWeaver*
GRN	*Gene of Regulatory Network*
LL	*Linkage Learning*
MCMC	*Markov Chain Monte Carlo*
MIMIC	*Mutual-Information-Maximizing Input Clustering*
MDL	*Minimum Description Length*
MPMs	*Marginal Product Models*
mRNA	*messenger RNA*
PBIL	*Population-Based Incremental Learning*
PMBGAs	*Probabilistic Model Building Genetic ALgorithm*
PV	*Probability Vector*
RNA	*Ribonucleic acid*
ROC	*Receiver Operating Characteristic*
rRNA	*Ribosomal RNA*
TN	*True Negative*
TP	*True Positive*
tRNA	*transfer RNA*
UMDA	*Univariate Marginal Distribution Algorithm*

Sumário

Capítulo 1

Introdução

Todas as células dos organismos vivos carregam o mesmo conjunto de genes. Entretanto, elas possuem propriedades e funcionalidades distintas, devido a quais genes estão sendo expressos. Quando se diz que um gene é expresso, significa que os produtos do genes funcionais são sintetizados em produtos finais, como por exemplo as proteínas. As proteínas são compostas por aminoácidos que são ligados entre si através de ligações peptídicas. Elas desempenham papel fundamental nos organismos como: catalização de reações bioquímicas, defesa do organismo, ativação ou desativação de determinado conjunto de genes e apoio em funções estruturais [AZ12].

Compreender o funcionamento do sistema celular é um dos grandes desafios na Biologia Computacional. A importância dos genes nesse sistema é conhecida e, entender as relações entre eles, pode trazer inúmeros benefícios. Por exemplo, pode ser possível rastrear o desenvolvimento de um câncer em uma célula, o aparecimento de outras doenças além de fornecer soluções para tratá-las.

Nesses últimos anos, os avanços tecnológicos na área Genômica têm gerado enormes quantidades de dados que, manualmente, não são possíveis de serem analisados. Uma maneira de capturar como os genes interagem entre si dentro do organismo é através de uma Rede de Regulação Gênica (*Gene of Regulatory Network* - GRN).

Existem diversos métodos diferentes que podem ser utilizados para modelar essas redes. O modelo mais detalhado e fiel para representar a GRN é os sistemas de equações diferenciais [Wer07]. Porém, a utilização desse modelo requer uma enorme quantidade de conhecimento prévio sobre o sistema sob investigação. Outra alternativa é a utiliza-

ção de métodos de agrupamento que, apesar de seu baixo custo computacional, não são adequados para inferir a estrutura detalhada das vias de sinalização bioquímicas.

Neste trabalho foi analisada a aplicação de dois Algoritmos de Estimação de Distribuição (*Estimation of Distribution Algorithm* - EDA) distintos na tarefa de inferir GRNs modeladas como Redes Bayesianas (*Bayesian Networks* - BN). O primeiro algoritmo foi proposto em [Emm07] e emprega o uso de algoritmo de agrupamento. O segundo foi proposto neste trabalho e envolve duas populações e um operador de similaridade utilizado para comparar soluções criadas com a população. Foi utilizado um esquema de inferência baseada em pontuação em que a mesma é atribuída a um determinado modelo considerando os dados observados. Para avaliar o desempenho do algoritmo, foram empregados três fontes distintas de dados: (i) dados gerados a partir de uma distribuição Gaussiana multivariada, (ii) dados gerados com a ferramenta GeneNetWeaver e (iii) dados reais a partir de experimentos de citometria de fluxo.

A estrutura deste trabalho esta organizada da seguinte forma: a próxima seção apresenta conceitos básicos das áreas de estudo exigidas para o desenvolvimento do algoritmo. No Capítulo 3, serão detalhadas as informações sobre o algoritmo proposto. Uma avaliação sobre a aplicação do algoritmo é apresentada no Capítulo 4. Finalmente, no Capítulo 6 são feitas as considerações finais sobre este trabalho.

Capítulo 2

Fundamentação Teórica

2.1 Aspectos Biológicos de Redes de Regulação Gênica

2.1.1 Estrutura do DNA

O ácido desoxirribonucleico (*deoxyribonucleic acid* - DNA) é um polímero formado por longas sequências de nucleótideos. Os nucleótideos são compostos por três componentes como ilustrado na Figura 2.1: um açúcar desoxirribose, que é uma pentose, um radical fosfato, proveniente do ácido fosfórico e uma base nitrogenada [AZ12]. No DNA, a base nitrogenada é ligada ao carbono 1' da pentose, e um ou até três grupos de fosfatos são ligados ao carbono 5' da pentose. O primeiro grupo ligado é denominado α (alfa), o segundo β (beta) e o terceiro γ (gama). No RNA, a desoxirribose é substituída pela ribose.

A Figura 2.2 ilustra os quatro tipos principais de bases nitrogenadas: citosina (C), timina (T), adenina (A) e guanina (G). Adenina e guanina são compostos de anel duplo classificadas como base púricas (derivadas da purinas) e estão presentes tanto no DNA quanto no RNA. A citosina e a timina, classificadas como base pirimídicas (derivadas de pirimidinas), são compostos de anel único e estão presentes no DNA. No RNA, a uracila (U) é encontrada no lugar da timina (T).

Os nucleotídeos formam cadeias através de ligações covalentes entre si. São estabelecidas pontes fosfodiéster entre o radical fosfato (5'-PO4) e o grupamento hidroxílico (3'-OH) do carbono 3' do nucleótideo seguinte. Essa característica define a orientação

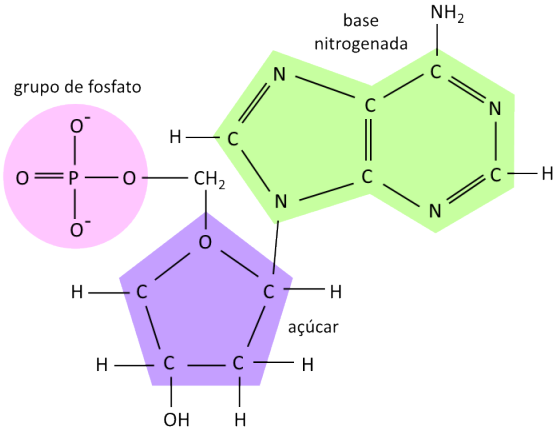

Figura 2.1: Estrutura geral de um Nucleótideo. A base nitrogenada é ligada ao carborno 1' da pentose e o radical fosfato é ligado ao carbono 5'. [Adaptado de [AZ12]]

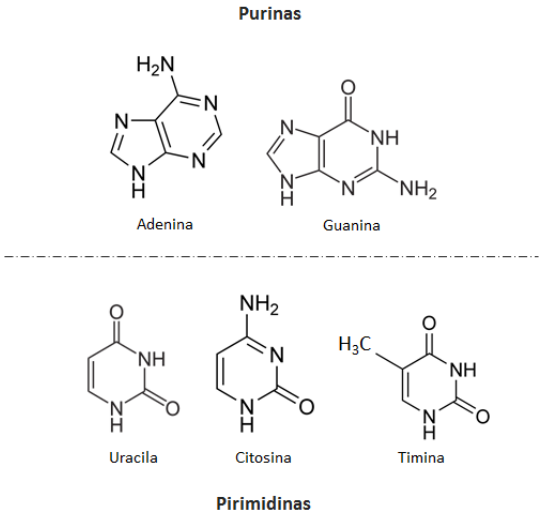

Figura 2.2: Tipos de Bases Nitrogenadas. Na parte superior, as bases purinas: Adenina (A) e Guanina (G). Na parte inferior, as bases pirimidinas: Uracila (U), Citosina (C) e Timina (T). As bases A, G e C aparecem tanto no DNA quanto no RNA. Já a base T somente no DNA e a base U apenas no RNA.

5' $\rightarrow$ 3' da cadeia formada. Uma vez que a cadeia terá em uma extremidade (carbono 5') o grupo fosfato e na outra (carbono 3') o grupamento hidroxílico. Sempre que for adicionado um novo nucleotídeo na cadeia, o mesmo será adicionado na extremidade 3' [AZ12]. Na Figura 2.3, é possível observar a orientação da cadeia formada.

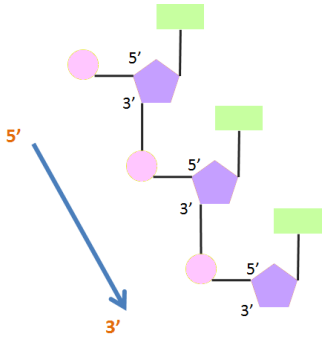

Figura 2.3: Orientação do crescimento do DNA. Ao adicionar um nucleotídeo na cadeia, ele será adiconado na extremidade 3'.

A estrutura do DNA, em forma de dupla hélice, foi descoberta em 1953 pelo norte-americano James Watson e pelo britânico Francis Crick. Nesse modelo, ilustrado na Figura 2.4, dois filamentos de DNA são enrolados entre si em forma de hélice. A estrutura açúcar-fosfato ficam externas em relação às bases nitrogenadas, exposta ao ambiente aquoso, com formato helicoidal como se fossem o corrimão de uma escada em espiral [AZ12]. Os dois filamentos estão em sentidos opostos unidos por pontes de hidrogênios entre os pares das bases nitrogenadas. Um filamento está na direção 5' $\rightarrow$ 3' e o outro na direção 3' $\rightarrow$ 5', sendo, desse modo, antiparalelos. A base A, na maioria das vezes, está pareada com a base T enquanto a base C forma ligação com a base G. Devido a essa regra, os filamentos de DNA são ditos complementares. Portanto, conhecendo a sequência de nucleotídos de um filamento, automaticamente conhecemos a sequência do outro filamento.

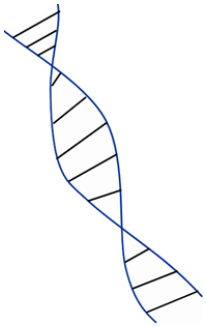

Figura 2.4: Estrutura do DNA. Dois filamentos de DNA são enrolados entre si em forma de hélice.

2.1.2 Dogma Central da Biologia Molecular

As instruções genéticas presentes nos nucleotídeos, são responsáveis pelo bom funcionamento de todos os seres vivos. Os organismos vivos dependem da capacidade da célula de replicar, transferir e traduzir as instruções genéticas codificadas no DNA. Um filamento de DNA é constituído por milhares de genes que transportam a informação necessária para a produção de RNA e proteínas.

RNA é estruturalmente semelhante ao DNA. A diferença consiste em dois aspectos principais: O DNA possui uma base nitrogenada Timina (T), enquanto o RNA possui a base nitrogenada Uracila (U) e o açúcar no DNA é uma desoxirribose, enquanto no RNA é uma ribose. Outra diferença presente é em relação ao formato. Enquanto o DNA apresenta uma organização em fita dupla o RNA é, usualmente, disposto em um único filamento.

As proteínas são os principais componentes funcionais dos organismos. São compostas por monómeros chamados aminoácidos que são ligados entre si através de ligações peptídicas. Elas desempenho papel fundamental nos organismos como: catalização de reações bioquímicas, defesa do organismo, ativação ou desativação de determinado conjunto de genes e apoio em funções estruturais. No organismo, todas as células presentes carregam o mesmo DNA. Entretanto, devido a regulação genética, as proteínas sintetizadas podem ser totalmente diferentes. A quantidade de proteína sintetizada é regulada por mecanismos de controle em diferentes estágios, os quais formam o núcleo do chamado dogma central da biologia molecular [Cri70].

O Dogma Central foi proposto por Francis Crick anos depois de ter descoberto a estrutura do DNA. O objetivo era descobrir como a informação presente nas moléculas de DNA fluía para gerar as proteínas. Crick estudou a relação entre a informação contida no DNA, as proteínas e os ácidos ribonucléicos (tRNA, rRNA e mRNA). Baseado nessas informações, ele postulou o que conhecemos hoje como Dogma Central da Biologia Molecular.

A Figura 2.5 ilustra os processos do Dogma Central da Biologia Molecular. O DNA, além da capacidade de autoduplicar é capaz de originar uma fita de RNA que possui o código para a síntese de aminoácidos para formar a cadeia polipeptídica.

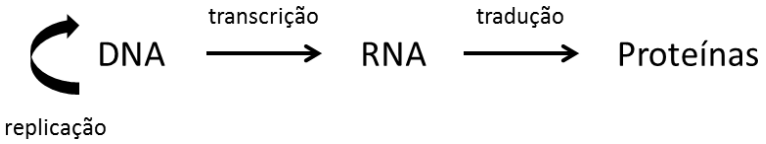

Figura 2.5: Dogma Central da Biologia Molecular. Representação do fluxo de informações entre DNA, RNA e proteínas

Replicação do DNA

O processo no qual o DNA consegue originar cópias de si mesmo é chamado replicação semiconservativa do DNA. A replicação é dita semiconservativa pois cada filamento original do DNA é mantido intacto e atua como molde para a síntese de um novo filamento. Meselson e Stahl [MS58], através de um experimento, demostraram que no processo de duplicação são geradas duas moléculas novas, no qual cada molécula é constituída de dois filamentos: um original e outro complementar, sendo antiparalelos entre si. Esse processo ocorre em todos os organismos vivos e garante a produção de células geneticamente idênticas e a passagem de material genético para a geração seguinte.

Nos seres eucariontes a replicação dá-se no interior do núcleo das células, e nos seres procariontes dá-se no citoplasma. O processo tem início em pontos específicos do DNA, conhecidos como origens, no qual uma molécula de um complexo enzimático, DNA-helicase, quebra as pontes de hidrogênio entre as bases nitrogenadas e desenrolam um pequeno segmento de DNA. Uma vez iniciado o processo, surgem pontos dinâmicos chamados de forquilhas de replicação, no qual os filamentos de DNA são separados. A partir desses pontos, a replicação continua em direções opostas. A replicação é bidirecional [Bro02].

A adição de nucleotídos livres é feita com o auxílio de uma enzima, a DNA Polimerase. Essa adição respeita a regra apresentada anteriormente: A base A sempre está pareada com a base T enquanto a base C forma ligação com a base G.

Transcrição

O processo pelo qual as moléculas de RNA são sintetizadas a partir de um molde de DNA, ou seja, com base na informação contida em uma molécula de DNA é chamado de transcrição. Através desse processo, são sintetizados todos os tipos de de RNAs: RNA

mensageiro ($mRNA$), RNA ribossômico ($rRNA$), RNA transportador ($tRNA$) entre outros. Durante a transcrição, os filamentos do DNA se separam e um serve de molde para o RNA, enquanto o outro fica inativo. No final do processo, os filamentos voltam a se unir.

A transcrição é um processo seletivo, uma vez que apenas pequenas porções do filamento são copiadas. O processo começa quando a RNA polimerase é conectada a uma das extremidades do DNA chamada de *promotor*. Quando isso ocorre, os filamentos do DNA começam a se separar, do mesmo modo que acontece na replicação, quebrando as pontes de hidrogênios entre as bases nitrogenadas. Em seguida, as bases do DNA são transcritas em RNA na sequência 3'→5'. Esse processo respeita a mesma regra de emparelhamento de base do DNA. O RNA resultante é antiparalelo ao filamento do DNA utilizado como molde e igual ao filamento codificado na replicação do DNA (5'→3'), exceto pela substituição das bases nitrogenadas T por U. O processo ocorre até que encontram a outra extremidade do DNA que determina o fim da transcrição.

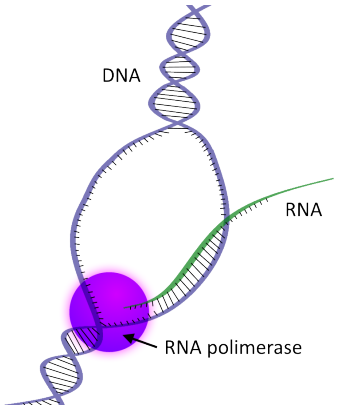

Figura 2.6: Transcrição

Tradução

O processo de síntese de proteínas é chamado de tradução. Ele ocorre no citoplasma e tem a participação de RNA e aminoácidos. A mensagem presente no $mRNA$ é decodificada e o ribossomo a utiliza para sintetizar a proteína de acordo com a informação dada. Os códons são a parte principal nesse processo. Um códon é constituído por três bases nitrogenadas consecutivas e codificam um aminoácido. Porém, o inverso não é verdadeiro uma vez que

um único aminoácido pode ser codificado por diferentes códons. A proteína é formada pelas ligações polipeptídicas entre os aminoácidos.

Os aminoácidos são capturados pelo $tRNA$, que possuem os anti-códons. Os anti-códons são formados por três bases nitrogenadas que se paream com os códons presentes na molécula de $mRNA$. Uma vez que o aminoácido é incorporado na cadeia por meio de uma ligação peptídica, o $tRNA$ volta ao citoplasma para se unir a outro aminoácido. Esse processo é repetido até que seja atingido o códon de terminação. Neste momento, a cadeia de polipéptido é liberada do ribossomo para o citoplasma da célula. A figura 2.7 ilustra esse processo.

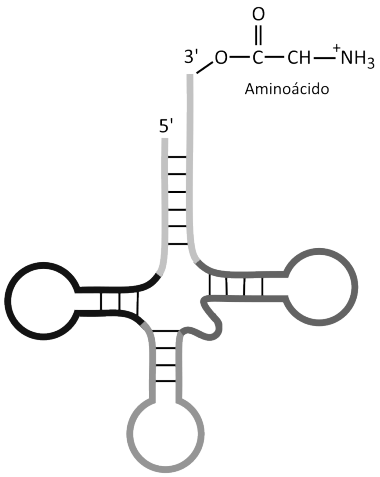

Figura 2.7: Tradução

2.1.3 Redes de Regulação Gênica

Na célula existem diversos fatores que influenciam em quais genes serão transcritos e em que quantidade. Isso ocorre por meio da regulação da expressão gênica. A expressão gênica é o processo em que a informação presente no gene é utilizada na síntese de um produto do gene funcional, como as proteínas.

Uma Rede de Regulação Gênica (*Gene Regulatory Network* - GRN) é um conjunto de genes que interagem entre si dentro do organismo. O objetivo de um GRN é capturar as dependências entre as entidades moleculares dentro da célula. Em uma GRN os

genes controlam a expressão de outros genes, ou seja, controlam quais outros genes são transcritos em mRNA. Entretanto, não ocorre de todos os genes interagirem entre si. Na verdade, um gene controla um subconjunto de outros genes e ele pode ser controlado por outro subconjunto. Além disso, ele pode se auto-regular [Wer07]. Graficamente, uma GRN pode ser vista como um grafo M em que os nós representam os genes e as arestas indicam as interações entre esses genes. Por exemplo, se existe uma aresta do nó A para o nó B, significa que o gene A regula o gene B.

Para compreender as relações entre todos os genes dentro de uma célula é necessário avaliar todos os componentes ao mesmo tempo, o que é inviável [Wer07].

2.2 Redes Bayesianas

Nesses últimos anos, foi possível observar um crescente aumento na busca por métodos robustos com a finalidade de extrair a estrutura completa da rede de regulação gênica. Entretanto, o fato dos dados experimentais serem esparsos e a presença de ruído neles tem dificultado esta tarefa. Assim, as Redes Bayesianas vem surgindo como uma alternativa de ferramenta para realizar a análises desses dados.

As Redes Bayesianas (*Bayesian Networks* - BNs) são o casamento entre a teoria da probabilidade e a teoria dos grafos. Elas são uma forma de representar o conhecimento sobre um domínio do qual não se tem certeza de quais varíaveis estão presentes. Em particular, são uma ferramenta bastante útil para representar as variáveis e suas relações de um determinado problema. A representação é feita através de um grafo, em que cada um dos nós representam as variáveis aleatórias $X = X_1, ..., X_n$ enquanto os arcos representam a influência direta de um nó em outro por meio das dependências probabilísticas entre essas variáveis.

2.2.1 Estrutura de uma Rede Bayesiana

Formalmente, uma BN é constituída por um grafo M, uma família de distribuição de probabilidades condicionais F e seus parâmetros q, os quais representam a distribuição conjunta para cada variável do conjunto de interesse. O grafo M é um grafo dirigido acíclico (*directed acyclic graph* - DAG). Os DAG são grafos no qual as arestas possuem uma

única direção e não possuem ciclos, isto é, para qualquer nó n, não existe uma ligação dirigida começando e terminando em n. Em virtude dessa propriedade, a distribuição conjunta dos nós do grafo M pode ser expandida em termos de probabilidades condicionais mais simples, o qual segue a hipótese de Markov: Cada nó é condicionalmente independente de seus não descendentes dado seus pais. Assim, podemos escrever a regra de cadeia das probabilidades na forma de produto [FLNP00]:

$$P(X_1, \cdots, X_n) = \prod_{k=1}^{n} P(X_i | Pa(X_i)) \tag{2.1}$$

em que, $X_1, ...X_n$ são as variáveis aleatórias e P_a representa os nós pais de X_i.

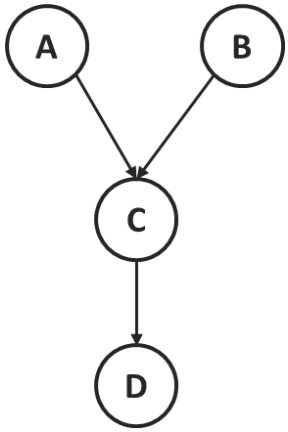

Figura 2.8: Exempo de uma Rede Bayesiana

Na Figura 2.8, um simples exemplo de Rede Bayesiana é ilustrado. Neste exemplo, a rede R é constituída pelo conjunto de nós N = {A, B, C, D} e pelo conjunto de arestas E = {(A,C), (B,C), (C,D)}, os quais representam as dependências entre as variáveis. A relação de dependência entre duas variáveis é representada pela seta que as conecta. Assim, o nó de onde parte a seta é denominado pai e o nó o qual a seta está orientada é chamado filho. Na Figura 2.8, o nós A e B são pais do nó C. A falta de seta entre dois nós, equivale a independência condicional entre duas variáveis.

Aplicando a Equação 2.1 na BN da Figura 2.8, a distribuição conjunta na forma de produto é obtida:

$$P(A, B, C, D) = P(A)P(B)P(C|A, B)P(D, C) \tag{2.2}$$

para definir a distribuição conjunta completa, é necessário determinar uma representação para cada probabilidade condicional na forma de produto (Equação 2.1). Existem diferentes tipos de representações para a família de distribuição condicionais F [FLNP00]. A escolha da representação depende do tipo de variável que estamos lidando, sendo mais comum utilizar distribuição de Gauss para variáveis contínuas e distribuição multinomial para as variáveis discretas.

2.2.2 Aprendizado de uma Rede Bayesiana

A aprendizagem de Rede Bayesiana consiste no processo de criar um modelo M de BN a partir de um determinado conjunto de dados de treinamento D. Portanto, o objetivo é determinar o modelo que melhor represente as dependências presentes nos dados disponíveis.

Existem dois passos distintos para realizar a aprendizagem. Primeiro, a estrutura da rede é aprendida, ou seja, como as entidades estão ligadas. Sendo $\mathbb{M}$ o espaço de todos os modelos possíveis, o primeiro objetivo é encontrar um modelo $M^* \in \mathbb{M}$ que melhor represente o conjunto de dados D:

$$M^* = \text{argmax}_M P(M|D) \tag{2.3}$$

O segundo passo consiste na aprendizagem do conjunto de parâmetros q associados com as arestas, se as entidades regulam e qual sua intensidade. Portanto, de posse da melhor estrutura M^* e os dados D, é possível encontrar os melhores parâmetros:

$$q = \text{argmax}_q P(q|M^*, D) \tag{2.4}$$

Se for aplicado o Teorema de Bayes na Equação 2.3, então:

$$P(M|D)\alpha P(D|M)P(M) \tag{2.5}$$

onde a probabilidade marginal, que calcula a média das probabildades dos dados D sobre todos os possíveis parâmetros atribuídos a M é uma integral:

$$P(D|M) = \int P(D|q, M)P(q|M)dq \tag{2.6}$$

A Equação 2.6 representa uma maneira de determinar o quanto uma determinada estrutura M explica o conjunto de dados D. Em [Hec94], foi proposta uma métrica denominada de *Bayesian Gaussian likelihood equivalent* (BGe) *score*. Essa métrica, assume que os dados estão associados a uma distribuição Gaussiana multivariada. Assim, aplicando o BGe *score*, é possível atribuir uma pontuação para uma determinada estrutura gráfica dado um conjunto de dados. Entretanto, a busca por estruturas com pontuações altas não é trivial. Uma vez que o número de estruturas aumenta exponencialmente com o número de nós presentes na rede (Tabela 2.1). Ademais, considerando que o conjunto de dados são esparsos (ver Figura 2.9), este não será representado por apenas uma única estrutura M^*.

Tabela 2.1: **Número de nós x número de estruturas.** O número de redes cresce exponencialmente com o número de nós (Adaptado de [Mur01]).

Número de nós	2	4	6	8	10
Número de redes	3	543	3.7 x 10^6	7.8 x 10^{11}	4.2 x 10^{18}

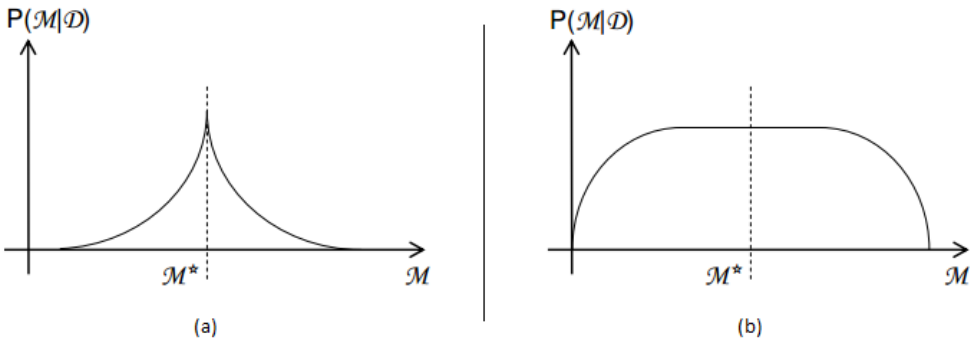

Figura 2.9: Inferência Bayesiana. O eixo vertical indica a probabilidade $P(H|D)$ e o eixo horizontal representa as estruturas de rede M. (a) A melhor estrutura de $M*$ está bem definida na distribuição de probabilidade uma vez que o conjunto de dados é bem informativo. (b) O conjunto de dados é pequeno e menos informativo, existindo muitas estruturas com altos valores para função de pontuação. Consequentemente, não é possível determinar a melhor estrutura. [reproduzida de [Wer07]].

2.3 Computação Evolutiva

2.3.1 Aprendizado de Ligação

Nos sistemas biológicos, ligação entre genes refere-se a situação na qual o nível de associação na herança de dois ou mais genes é maior que o esperado pela Lei Mendeliana da segregação independente [HJ98]. O nível de ligação é a medida da distância entre dois genes que estão no mesmo cromossomo e existe uma maior probabilidade de que sejam herdados juntos após o cruzamento [YpTlKG07]. Os EAs, para a solução de problemas, usam uma metodologia que é baseada na teoria de seleção natural de Darwin oriunda dos GAs. Assim, compreender as semelhanças entre o sistema biológico e os EAs pode ajudar a entender a importância do aprendizado de ligação (*Linkage Learning* - LL).

A troca de material genético durante a meiose é conhecida como recombinação genética. Durante esta fase ocorre o *crossover*, o qual incorpora, em cada novo descendente, genes provenientes de cada um de seus pais. Na Figura 2.10 estão ilustrados dois casos de *crossover*. No primeiro caso, quanto mais próximos os genes estiverem, maior a probabilidade de que eles sejam herdados juntos pelo descendente. Já no segundo cenário, devido a distância, é mais provável que os genes sejam divididos após o cruzamento.

Quando aplicado em GAs, os cromossomos são cadeias de caracteres com um alfabeto finito, geralmente binário. Os GAs não são capazes de capturar essas ligações entre os genes com muita facilidade. Eles precisam de um conhecimento prévio sobre o problema, para ordenar os genes adequadamente no cromossomo. Esse procedimento é realizado durante a codificação.

Codificar os genes adequadamente no cromossomo significa manter as variáveis que interagem próximas uma das outras. Entretanto, ter o conhimento prévio sobre o problema, para ordenar adequadamente os genes, não é uma tarefa simples. E, para problemas em que existem interações de ordem do tamanho do problema, essa abordagem não obtem sucesso [Emm07].

O processo de conhecer as relações entre as variáveis do problema é chamado aprendizado de ligação. Em [GB90], os autores já se preocupavam com a ordenação dos genes e idealizaram um operador de reordernamento. A importância de aprender sobre as relações entre as variáveis foi sugerida em [Hol75], no qual o autor destaca a preservação e

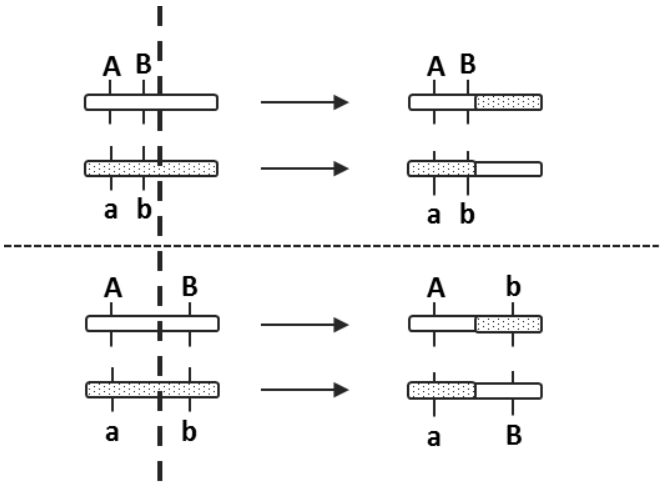

Figura 2.10: Meiose e *Crossover*. A distância entre os genes A e B no cromossomo indica o grau de sua ligação. (Adaptado de [YpTlKG07])

combinação de subestruturas chamada de hipótese de blocos construtores.

A importância do aprendizado de ligação para ajudar na resolução de problemas é foco em trabalhos de Computação Evolutiva (*Evolutionary Computation* - EC) [YpTlKG07, Har97]. O autor em [YpTlKG07] sugere que as técnicas de aprendizado de ligação podem ser classificadas como: ligação física e ligação virtual.

- **Ligação Física** Um EA é dito usar ligação física quando ligação de genes emerge a partir da localização física dos genes do indivíduo.
- **Ligação Virtual** O EA baseado em ligação virtual tenta capturar informações sobre a estrutura do problema e não tenta alterar a codificação utilizada. Esse tipo de aprendizado parece ser mais eficiente que o baseado na ligação física, mesmo tendo menor embasamento biológico.

2.3.2 Blocos Construtores - BBs

Como uma maneira de explicar como os GAs funcionam, Holland [Hol75] propôs a teoria dos esquemas. Um esquema é uma regra capaz de representar diversos individuos simultaneamente pela similaridade [Gol98]. Um esquema utiliza um alfabeto binário e finito (S), no qual S $\in \{0, 1, *\}$. O caracter *, também chamado de *don't care*, indica que aquele gene não está especificado no esquema.

Por exemplo, o esquema [1 * 0 1] representa os indíviduos [1 0 0 1] e [1 1 0 1]. Já o esquema [1 * 0 *], representa 4 indivíduos: [1 0 0 0], [1 0 0 1], [1 1 0 0] e [1 1 0 1]. O primeiro esquema especifica uma regra no qual o primeiro gene é 1 e os dois últimos são 0 e 1. O gene * não está especificado na regra.

O número de indivíduos representados por um esquema depende da quantidade de genes que não estão especificados. Um esquema representa 2r̂ indivíduos, no qual r é o número de genes não especificados.

A ordem de um esquema S, o(S), é o número de posições especificadas, isto é, o número de posições definidas com 0's e 1's. A ordem representa a especificidade do esquema, ou seja, quanto maior a ordem, mais específico é o esquema. Já o comprimento é a maior distância entre as posições fixas do do indivíduo, ou seja, a distância entre a primeira e a última posição especificada no esquema. Esta propriedade indica o quanto a informação está compactada no esquema. Assim, quanto menor o comprimento, menor a chance de romper o esquema após o *crossover*.

Além dessas duas propriedades, o esquema (S) possui o *fitness*, o qual é definido como o *fitness* médio de todos os indivíduos que são representados por (S). Considerando que os GAs tendem a selecionar indivíduos com *fitness* mais alto, então esquemas com *fitness* alto terão maior probabilidade de serem selecionados. Portanto, esquemas com ordem baixa, comprimento de definição curto e *fitness* acima da média têm grandes probabilidades de estarem presentes nas gerações subsequentes [Hol75]. Esquemas com essas características são chamados de blocos construtores (*Building Blocks* - BBs).

2.3.3 Algoritmos de Estimação de Distribuição

Nas últimas décadas diversas técnicas de Algoritmos Evolucionários (*Evolutionary Algorithms* - EAs) têm sido propostas: Algoritmos Genéticos (*Genetic Algorithms* - GAs) [Hol75] e Algoritmos de Estimação de Distribuíção (*Estimation of Distribution Algorithms* - EDAs) [MP96] são algumas delas. Elas são consideradas ferramentas importantes para resolver problemas de otimização e possuem duas características básicas: estratégias para seleção de soluções promissoras e geração de novas soluções de modo a formar uma nova população. Esses processos são repetidos até que os critérios de paradas sejam alcançados.

GAs são, possivelmente, uma das técnicas mais utilizadas de EAs. São inspirados na

teoria de seleção natural de Darwin e na recombinação genética. Um típico GA é composto por uma população de soluções, inicialmente, gerada de forma aleatória. É esperado que essa população evolua durante determinado número de gerações. A cada geração, um conjunto de soluções (normalmente duas) são selecionadas, de acordo com algum mecanismo de seleção, para gerar uma solução da geração subsequente. A qualidade de uma solução é calculada através do seu *fitness*, que representa o quanto a solução otimiza a função objetivo. O processo de criação envolve os mecanismos de cruzamento e mutação. O Cruzamento é responsável por combinar as soluções selecionadas através do método de seleção enquanto que mutação é utilizado para introduzir pequenas e aleatórias mudanças, tanto boas quanto ruins, na solução. Esse último, ajuda a preservar a diversidade da população, evitando uma convergência prematura para ótimos locais.

EDAs, também conhecido como Algoritmos Genéticos baseados na Construção de Modelos Probabilisticos (*Probabilistic Model Building Genetic ALgorithms* - PMBGAs) [MPL99], são técnicas de busca estocásticas baseadas em modelos probabilisticos. Esses modelos podem considerar a independência entre as variáveis do problema ou capturar as dependências de ordem igual ou superior a 2 entre as variáveis. Aplicando modelos capazes de capturar essas dependências, os EDAs são aptos a resolver um dos principais problemas dos GAs: a quebra dos blocos construtores. A principal diferença de outras técnicas de EAs, como GAs é que a evolução da população ocorre pela estimação da distribuição de probabilidade das soluções promissoras e, em seguida, da amostragem do modelo probabilístico [Emm07]. Essa diferença implica na redução do número de parâmetros utilizados no algoritmos, uma vez que o uso dos operadores de cruzamento e mutação é evitado.

Em EDAs, a população de soluções é representada por vetores de tamanho fixo, onde cada índice desse vetor corresponde a uma variável. Alguns EDAs, como PBIL e cGA, não mantêm uma população explícita de soluções como os GAs. Eles representam a população através de um vetor de probabilidade denominado PV (*Probability Vector*). Consequentemente, a complexidade de memória é significativamente reduzida.

As melhores soluções são selecionadas a partir de uma população inicialmente gerada de forma aleatória. A partir dessas soluções selecionadas, é realizada a inferência do modelo probabilístico. Em seguida, novas soluções são geradas através da amostragem

do modelo inferido. Esses procedimentos são realizados até que a convergência ocorra. A Figura 2.11 ilustra esse comportamento.

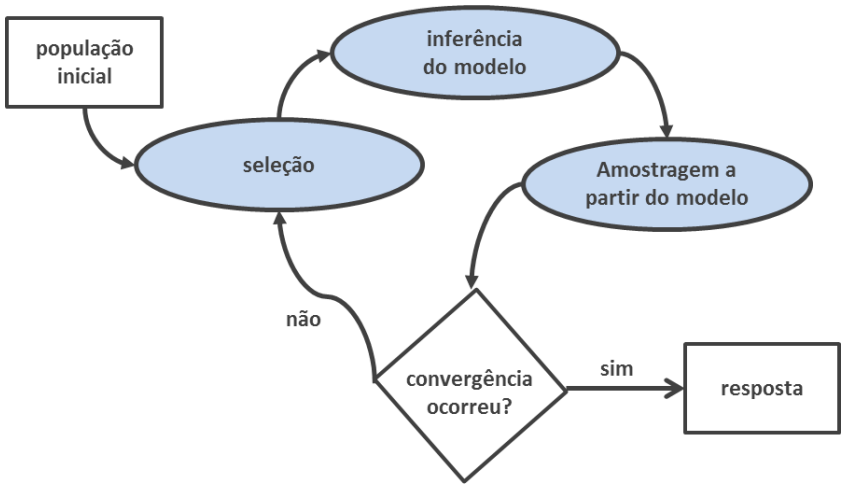

Figura 2.11: Comportamento de um EDA

O processo de inferência é uma das principais diferenças entre os EDAs. Os EDAs que utilizam modelos probabilísticos que assumem independência entre as variaveis possuem comportamento similar aos algoritmos genéticos com cruzamento uniforme. Esses modelos são mais simples e computacionalmente mais econômicos visto que a estrutura do modelo é sempre a mesma [GRHG99]. No entando, em problemas reais seu desempenho não é satisfatório. De outro lado, os EDAs que tentam capturar a estrutura do problema, utilizando modelos mais complexos que na maioria das vezes envolve a busca por uma fatoração adequada, conseguem ser mais eficientes que os GAs simples [Emm07]. Esses modelos adotam estatísticas de ordem maior que um, visto que interações de ordem maior que um entre as variáveis são consideradas. Entretanto, estimar a estrutura do problema e todas as interações presentes nele não é uma tarefa fácil e, possui um custo computacional elevado para encontrar a fatoração adequada [YSG05].

Os EDAs combinam informações das soluções candidatas a fim de obter bons resultados. Entretanto, se o modelo utilizado não for capaz de capturar a estrutura do problema, a combinação de informações pode levar a resultados ruins. Então, a busca por modelos mais complexos visa obter informações sobre a estrutura do problema.

Uma das formas de classificar os EDAs é através da complexidade do modelo probabilístico utilizado. Assim, eles podem ser classificados como: EDAs univaráiveis, EDAs

bivariáveis e EDAs multivariáveis [PLnS13].

EDAs univaríaveis

É considerada a forma mais simples de EDAs. O modelo utilizado observa as frequências de cada variável do problema para gerar a nova população. As variáveis são definidas como independentes. Para problemas lineares, em que não são consideradas as interações entre as variáveis, essa classe de EDAs tem atingindo resultados satisfatórios, pois permite misturar os blocos de construção (*building blocks* - BBs) de forma eficiente, além de possibilitar que eles apareçam em gerações subsequentes [Emm07]. Entre os representantes dessa classe estão: *Population-Based Incremental Learning* (PBIL) [Bal94], *Compact Genetic Algorithm* (cGA) [GRHG99] e *Univariate Marginal Distribution Algorithm* (UMDA) [Müh97].

PBIL foi inspirado no *Equilibrium Genetic Algorithm* (EGA) [JBS93], ele supera os GAs em vários problemas de otimização, tanto em velocidade quanto em precisão. A população é representada através de um vetor de probabilidade, o qual mantêm um modelo probabilístico das soluções promissoras. O vetor é utilizado para gerar as novas gerações e é atualizado considerando o valor de fitness das soluções. Já que somente as probabilidades univariadas são consideradas, ele mantêm para cada posição as probabilidades de 1Šs. Inicialmente todas as posições são definidas com 0.5 e durante a execução do algoritmo elas são deslocadas para 0 ou para 1. A cada geração, um número fixo N de soluções binárias são geradas a partir do vetor de probabilidade (PV). Essas soluções são avaliadas e as melhores são selecionadas para atualizar o PV. Além de N, os critérios de parada e a taxa de mudança do PV são parametros do algoritmo.

O cGA é equivalente a um GA de ordem 1 com cruzamento uniforme. Ele funciona semelhante ao PBIL. Também utiliza um PV para representar a população. Entretanto, a cada iteração dois indivíduos são gerados a partir do PV. Em seguida, é definido qual é o melhor entre os dois baseado na função objetivo. O PV é atualizado da seguinte forma: Se a posição i do melhor indivíduo for diferente do pior, então o PV na mesma posição será incrementado em $\frac{1}{n}$, onde n é o tamanho da população. Caso contrário, a posição do PV é decrementada em $\frac{1}{n}$. Isso indica a contribuição de um indivíduo para cada posição do PV.

O UMDA, ao contrário do PBIL e do cGA, mantêm a população de soluções. As soluções são binárias e possuem tamanho fixo N. A cada geração, a distribuição do conjunto de soluções selecionadas é estimada com um modelo linear denominado distribuição marginal univariada. Nesse modelo, as frequências dos valores para cada posição são calculadas. A nova população é gerada baseada nessas frequências. Esse processo é repetido até que os critérios de parada sejam atingidos.

EDAs bivariáveis

Apesar dos EDAs univariáveis obterem bons resultados em problemas lineares, o mesmo desempenho não é obtido quando existe interações entre as variáveis. Então, alguns esforços foram realizados afim de permitir que os EDAs fossem eficiente nesses problemas. Nessa classe, cada variável do problema depende apenas de uma outra. Uma variável é definida como raiz e possui distribuição univariada. As demais variáveis dependem da variável que estiver na posição imediatamente anterior. Então, a distribuição dessas variáveis é condicionada ao valor presente na posição anterior. *Mutual-Information-Maximizing Input Clustering* (MIMIC) [BIV97] e *Bivariate Marginal Distribution Algorithm* (BMDA) [PM99] são alguns dos representantes dessa categoria.

No MIMIC, a estrutura da solução é na forma de cadeia. Onde a primeira variável é definida independente. As demais variáveis dependem apenas de uma outra na cadeia. Para construir a cadeia de variáveis, é utilizado um algoritmo guloso o qual não garante a otimização global da distribuição [MPL99]. Em cada interação, uma nova solução é gerada utilizando uma distribuição univariada para a primeira variável e, para as demais variáveis, uma probabilidade condicional relativa ao valor da variável imediatamente anterior.

BMDA é outro representante dessa classe. É uma extensão do UMDA. Ao contrário do UMDA, ele utiliza uma distribuição bivariada, a qual permite identificar dependências mais importantes. A distribuição é estimada com o objetivo de preservar essas dependências. Para problemas lineares e de ordem 2, o algorítmo possui desempenho satisfatório o qual não é observado ém problemas com dependências de ordem superiores.

EDAs multivariáveis

A utilização de EDAs bivariáveis não é suficiente para resolver problemas com dependências de ordem superior a dois entre as varáveis, o que pode ser visto em [PM99]. Então, surgiu a necessidade de pesquisar modelos capazes de cobrir essas interações multivariáveis. Alguns dos algoritmos que estão presente nessa classe são: *Extended Compact Genetic Algorithm* (ECGA) [Har99], *Factorized Distribution Algorithm* (FDA) [HMOR99], *Bayesian Optimization Algorithm* (BOA) [PGCP00] e *Estimation of Bayesian Network Algorithm* (EBNA) [EL99].

No ECGA, a distribuição de probabilidade utilizada é conhecida como modelo de produtos de marginais (marginal product models - MPMs). Ela é capaz de representar a distribuição para mais de uma variável ao mesmo tempo. Nesse modelo, as variáveis são divididas em grupos (clusters), os quais são considerados mutualmente exclusivos. Na busca pelo melhor modelo, o ECGA utiliza a métrica de comprimento mínimo de descrição (minimum description length - MDL). O MDL penaliza os modelos complexos e os imprecisos, fazendo com que a distribuição se aproxime do ótimo. Para a criação do modelo é utilizado um algoritmo de busca guloso. Inicialmente, as variáveis são consideradas independentes e a cada iteração, grupos vão sendo unidos até que não haja mais melhoras. O problema surge para problemas com blocos de construção altamente sobrepostos, pois não são corretamente modelados utilizando esse método [MPL99].

O desempenho do FDA pode ser observado em [HMOR99]. É mostrado que os problemas são resolvidos de modo rápido, confiável e preciso. Entretando, é necessário conhecer a estrutura do problema, pois o algoritmo não é capaz de aprende-la. A distribuição e sua fatorização são dadas por um especialista. No entanto, em problemas reais, essa informação não está disponível. Assim, o algoritmo está limitado a problemas em que se conhece a estrutura do problema.

No BOA a representação da população de melhores soluções é realizada através de redes Bayesianas. A população evolui a cada geração por meio da construção e amostragem dessas redes. Na maioria das vezes, a população inicial é gerada de forma aleátoria. Em cada iteração, são selecionadas as melhores soluções a partir da população atual utilizando um método de seleção qualquer. Em seguida, uma rede Bayesiana é criada a partir das soluções selecionadas. Então, novas soluções são geradas pela amostragem dessa rede.

Por último, as soluções geradas são misturadas a população. Diferente do FDA, o BOA não requer nenhuma informação sobre o problema, ele é capaz de descobri-la sozinho.

O EBNA também adota Redes Bayesianas como modelo probabilistico. Ele não possui restrições quanto ao número de pais que as variáveis podem ter. É baseado em uma pontuação (score) e um método de busca. O *Bayesian Information Criterion* (BIC) é um dos scores utilizados pelo autor e indica a qualidade de determinada Rede Bayesiana em representar as interdependencias entre as variáveis. É utilizado pelo algoritmo guloso que realiza o procedimento de busca pela melhor representação.

2.4 Métodos Relacionados

Basicamente, o processo de inferência de redes envolve duas etapas distintas: (i) aprendizado da estrutura da rede (otimização da estrutura) e (ii) estimação dos parâmetros (otimização de parâmetros). Enquanto que o primeiro está relacionado ao problema de encontrar o modelo da rede que melhor explique os padrões de expressão observados nos dados, o segundo está ligado a identificação dos parâmetros do modelo [HLT+09].

Na tarefa de inferir GRNs, a principal etapa é a da otimização de estrutura, uma vez que existem poucos parâmetros envolvidos [HLT+09]. Na maioria das vezes, essa etapa, envolve a avaliação de um conjunto de modelos distintos de rede aplicando uma função de pontuação (*score*).

A melhor maneira envolve a utilização de força bruta, a qual testa todos os possíveis modelos de rede. Entretanto, o número total de redes está relacionado ao número de genes presente na mesma. Em um cenário do mundo real, as redes incluem centenas de genes, tornando a hipótese de testar todas as possibilidades inviável.

Devido a quantidade limitada de dados disponíveis e a presença de ruídos nos dados, a inferência de um modelo a partir dos dados de experimentos torna-se uma tarefa difícil. Outra dificuldade relacionada a escassez de dados é a possibilidade de diversos modelos de rede terem pontuação semelhante. Dessa forma, os algoritmos de aprendizagem empregam heurísticas para encontrar uma solução boa, quando a ótima não é possível, em tempo de execução viável. As heurísticas utilizadas aplicam uma função de pontuação para auxiliar na busca por soluções boas.

Entre os métodos que aplicam heurísticas, os que são frequentemente utilizados na tarefa de inferência de GRNs são: algoritmos evolucionários (algoritmos genéticos - GAs e algoritmos de estimação de distribuição - EDAs), *simulated annealing* - SA e *Markov Chain Monte Carlo* - MCMC.

De maneira geral, a heurística desses métodos são similares. O funcionamento é baseado na escolha probabilística, a cada iteração, em permanecer no estado atual s ou deslocar-se para um novo estado s'. Em GAs e EDAs, o novo estado pode ser obtido pela combinação de duas ou mais soluções. A escolha probabilística pelo caminho que o algoritmo irá seguir é o que possibilita essa abordagem escapar de ótimos locais, buscando aproximação com ótimos globais.

Devido a incerteza presente nos dados de experimentos, vários modelos distintos podem ter uma mesma pontuação. Consequentemente, a seleção de apenas um único modelo é prejudicada. Assim, o melhor é retornar mais de um modelo [GC03]. E, por exemplo, calcular a média entre eles. MCMC funciona desta maneira e foi aplicado na inferência de BNs [MY93] e no cenário de GRNs [FK00].

Em GAs, normalmente, as soluções são representadas como matrizes de peso. São, geralmente, avaliadas com base na comparação de padrões de expressão gênica entre as redes inferidas e a rede alvo, na tentativa de minimizar a diferença entre suas dinâmicas [CLA03]. Outras abordagens também foram propostas em GAs. Entretanto, mesmo com o auxílio significativo na identificação de redes com diversas dezenas de genes, a aplicação dessas abordagens são limitadas devido ao grande número de parâmetros a serem inferidos.

Outra abordagem em GA é modelar as soluções como BNs (grafos acíclicos), que compõem a ordem topológica e a relação entre os genes. [Dav10] avaliou as soluções baseado em funções de pontuação implementadas pelo Weka[1], denominadas de *Akaike Information Criterion* (AIC) e *Minimum Description Length* (MDL). Embora o algoritmo tenha previsto a ordenação topológica correta, o conjunto de relações entre nós não pode ser totalmente reconstruído. Consequentemente, os resultados não foram muitos satisfatórios.

Aplicação de métodos de otimização estocásticos e busca heurísticas tem possibilitado uma importante contribuição na tarefa de inferência de GRNs. Entretanto, um dos problemas encontrados nessa abordagem é o tempo de convergência. Por exemplo, para

[1] Weka é uma coleção de algoritmos de aprendizado de máquina para as tarefas de mineração de dados.

[ADHR04], um MCMC corretamente implementado deveria ser capaz de escapar de mínimos locais. Entretanto, esse processo pode demandar quantidade de tempo inviável. Consequentemente, afetando a convergência do algoritmo.

Além do MCMC, GAs também podem ter problemas de convergência. O algoritmo pode sofrer convergência prematura ou retornar ótimos locais em vez do ótimo global [RN99].

Existem diversos modelos que podem ser aplicados na tarefa de inferir redes regulatórias, entre eles: redes Booleanas [Kau69], redes de associação [BKK00] e redes Bayesianas [FK00]. O primeiro e o segundo são brevemente resumidos a seguir. O terceiro, o qual foi utilizado neste trabalho, foi explicado na seção 2.2.

Redes Booleanas: foram propostas como modelos para representação de GRNs em [Kau69]. Nesta proposta os genes são representados por variáveis binárias: 1 (ativo) e 0 (inativo). Nesse modelo, a GRN é definida como um grafo dirigido $M(V, F)$ em que $V = \{v_1, v_2, ..., v_n\}$ é o conjunto de nós que representam os genes que fazem parte da rede e $F = \{f_1, f_2, ..., f_n\}$ é o conjunto de funções de transição que representam as interações entre os genes. Quando os dados são contínuos eles precisam ser discretizados.

Redes de Associação: é, provavelmente, o modelo mais simples para representar as GRNs. Diferente das redes booleanas, esse modelo é um grafo não dirigido, o qual descreve a estrutura da GRN mas sem considerar qual gene regulam e quais são regulados. As interações na rede são baseadas em um coeficiente de correlação, obtido a partir de níveis de expressão de genes. As interações são assumidas quando o coeficiente está acima de um limiar, quanto maior o limiar, mais esparsa será a GRN inferida [HLT+09].

Além da escolha pelo modelo de Redes Bayesianas, neste trabalho, foi proposto como método de busca os algoritmos explicados em 3.1 e 3.2.

Capítulo 3

Abordagem Proposta

Neste trabalho foram aplicados dois algoritmos na tarefa de inferir redes Bayesianas a partir dos dados dos experimentos, ou seja, encontrar o modelo que melhor represente as dependências presentes nos dados disponíveis. O primeiro foi proposto em [Emm07] e o segundo foi desenvolvido neste trabalho. Os dois algoritmos serão chamados daqui em diante Algoritmo1 e Algoritmo2 respectivamente.

Como mencionado na Seção 2.3.3, no EDA uma função de *fitness* é utilizada para determinar a qualidade de uma solução. Tanto no Algoritmo1 quanto no Algoritmo2, foi utilizada a métrica BGe *score*, descrita no Capítulo 2, como função de *fitness*. Cada solução é uma estrutura de BN. E, para cada estrutura, é associado um *score* que define o quanto essa estrutura possa ter sido responsável para gerar o conjunto de dados. Em ambos algoritmos, o objetivo é encontrar a estrutura com melhor pontuação.

3.1 Algoritmo 1

O Algoritmo1 é um EDA que emprega agrupamento e aprendizado de modelos probabilísticos. Neste algoritmo foi proposto um novo operador de combinação, o *cg-recombination*. Na primeira etapa são geradas $N0$ soluções de forma aleatória. No qual cada solução da população representa uma rede e é composta por um vetor binário. Esse vetor equivale a matriz de adjacência da rede. Assim, caso a rede tenha n nós, o tamanho do vetor será de 2^n. Na Figura 3.1, um exemplo para o caso com 2 nós é ilustrado.

A partir desse momento, é aplicado o K-means [Mac67] na população obtendo k grupos.

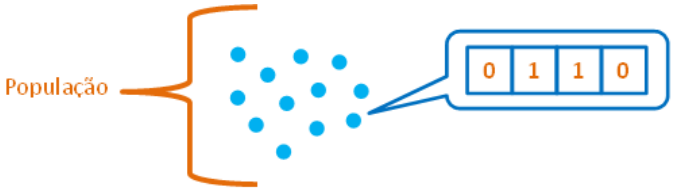

Figura 3.1: População de soluções. Cada solução representa uma rede, no qual o valor 1 em cada posição significa a presença de uma aresta entre os nós.

K é um parâmetro do algoritmo que deve ser definido inicialmente. A Figura 3.2 ilustra o resultado da aplicação do K-means na população para $k = 3$.

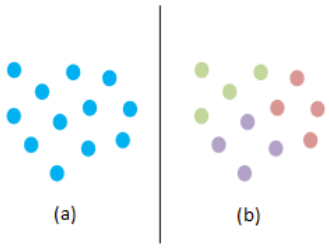

Figura 3.2: População separada em grupos. (a) A população antes do agrupamento. (b) A população divida em 3 grupos.

Cada grupo define um modelo probabilístico que é um vetor de probabilidade (PV). O PV representa o centroide de cada grupo e é formado pelas proporções de 1's para cada posição (Figura 3.3). Os PV's são armazenados em uma matriz denominada de $\hat{\pi}$. Sendo $\hat{\pi}(i)$ o PV do grupo i.

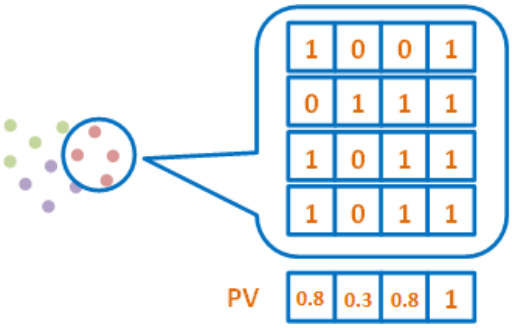

Figura 3.3: Vetor de Probabilidade (PV). Os PV são a proporção de 1's para cada posição. Representa o centroide de cada grupo e são armazenados na matriz $\hat{\pi}$.

Matriz de informação

A fim de utilizar sempre o PV mais informativo para determinada variável j, calculamos a matriz $\hat{w}$. Essa matriz representa a diferença de entropia antes (Equação 3.1) e depois (Equação 3.2) de analisar determinado grupo i.

$$\hat{h}_{.,j} = \sum_{q \epsilon \{0,1\}} \hat{p}_{j,q} \log_2(\hat{p}_{j,q}) \tag{3.1}$$

em que, $\hat{p}_{j,q}$ é a proporção de soluções em toda a população que tem o valor q na posição j.

$$\hat{h}'_{.,j} = \sum_{q \epsilon \{0,1\}} \hat{p}'_{j,q} \log_2(\hat{p}'_{j,q}) \tag{3.2}$$

em que, $\hat{p}'_{j,q}$ é a proporção de soluções em toda a população que tem o valor q na posição j, sem considerar as soluções do grupo i.

Assim, podemos definir a matriz $\hat{w}$ como:

$$\hat{w}_{i,j} = \hat{h}_{i,j} - \hat{h}'_{i,j} \tag{3.3}$$

O fluxograma a seguir (Figura 3.4), ilustra o comportamento do algoritmo enquanto o critério de parada não seja atingido.

Cruzamento

Seguinte a escolha aleatória de um PV, é verificado (através de uma probabilidade) se o operador *cg-recombination* será utilizado. Ao utilizar o operador, um novo PV será criado a partir dos dois $PV's$ anteriormente selecionados. Neste processo de criação é considerada a matriz $\hat{w}$. Cada posição j do novo PV será:

$$v_j = \begin{cases} \hat{\pi}_{A,j}, & \hat{w}_{A,j} > \hat{w}_{B,j} \\ \hat{\pi}_{B,j}, & \text{caso contrário} \end{cases} \tag{3.4}$$

A Figura 3.5 ilustra a criação de um novo PV baseado na matriz $\hat{w}$.

Nesse exemplo, o $\hat{\pi}_{A,1}$ é igual a 0.3, o que indica que 30% das soluções presentes neste grupo possuem o valor 1. Essa taxa ainda permite gerar novas soluções com o valor 0. Entretanto, essas chances seriam nulas caso o valor seja 1.0. Assim, uma forma

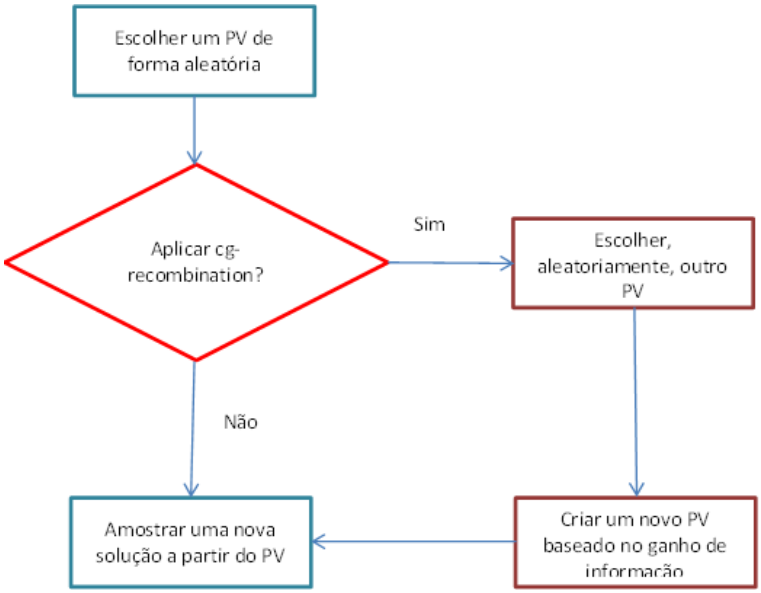

Figura 3.4: Fluxograma do Algoritmo 1. Enquanto o critério de parada não for atingido, será criada uma solução a partir de um PV ou de uma combinação de dois $PV's$.

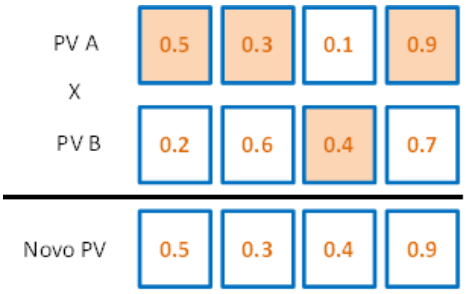

Figura 3.5: Criação do PV baseado na matriz $\hat{w}$. Para $j \in \{0,1,3\}$ é considerado o PV de A e para $j \in \{2\}$ é considerado o PV de B.

de incorporar incerteza na estimação das proporções é através do estimador de Wilson [AC98]. Deste modo, a proporção é estimada conforme a Equação 3.5.

$$\hat{\pi}_{A,i} = \frac{\text{numero de uns} + 2}{\text{numero de solucoes no grupo} + 4} \tag{3.5}$$

Inserção na população

Finalmente, é gerada uma nova solução a partir do PV. Em seguida, calcula-se o *Fitness* da nova solução e, se esse for melhor que o pior presente na população, então ocorre a

substituição da solução. Caso a população sofra a alteração, então as matrizes $\hat{\pi}$ e $\hat{w}$ precisam ser atualizadas.

Como mencionado na Figura 3.4, o Algoritmo1 é executado enquanto o critério de parada não é atingido. Para o Algoritmo1, existem dois critérios de parada: (i) no momento em que toda a população converge para a mesma solução ou (ii) ocorrência de N iterações sem nenhuma solução gerada entrar na população.

3.2 Algoritmo 2

O Algoritmo2 é um EDA que, diferentemente do algoritmo1, não emprega agrupamento. Uma vez que o agrupamento precisa ser aplicado a cada iteração, o uso dele interfere no desempenho do algoritmo. Outra diferença é a presença de uma segunda população no algoritmo. Além da população de soluções (em que cada solução representa um modelo que represente as dependências presente nos dados disponíveis), existe a população de máscaras. Cada máscara é um vetor com o mesmo tamanho do vetor da solução. Finalmente, enquanto no Algoritmo1 o processo de inclusão de uma nova solução na população se dá pela comparação com a pior solução presente na população. No Algoritmo2, a nova solução é comparada com a mais similar dentro da população, não encontrando uma solução similar, então segue a mesma regra do Algoritmo1. Na Figura 3.6 é ilustrado o fluxograma da visão geral do Algoritmo2.

Seguinte a criação da população de soluções e de máscaras, o algoritmo faz os próximos passos até que o critério de parada seja atingido. O primeiro passo consiste em verificar, de acordo com uma probabilidade de cruzamento pc, se o algoritmo usará uma máscara ou o cruzamento de duas máscaras. Uma vez que o cruzamento seja realizado (Figura 3.7a), uma máscara M_R será criada a partir de duas máscaras M_1 e M_2 selecionadas de forma aleatória da população. Caso contrário (Figura 3.7b), a máscara $M1$ será definida como M_R. A seguir (Figura 3.7c), duas soluções S_1 e S_2 são selecionadas também de modo aleatório.

Uma nova solução é criada pela Equação 3.6, baseada nas duas soluções selecionadas e na máscara M_R. Após a criação, é calculado o valor de *fitness* para a nova solução. Na Figura 3.8 é ilustrado o processo de criação da nova solução.

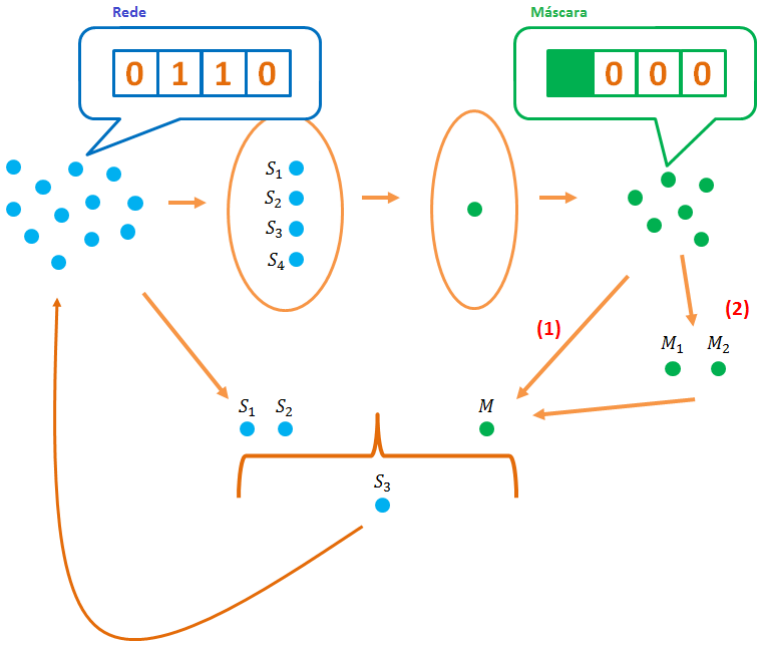

Figura 3.6: Visão geral do Algoritmo2. O algoritmo trabalha com duas populações: uma das soluções e outra de máscaras. As máscaras são criadas a partir da seleção de N_S soluções. E, as soluções são geradas, pela combinação de duas soluções de acordo com apenas uma ou uma combinação de máscaras presente na população. Tanto a solução quanto a máscara são avaliadas para verificar se serão descartadas ou não. Esse processo ocorre enquanto o critério de parada não seja atingido.

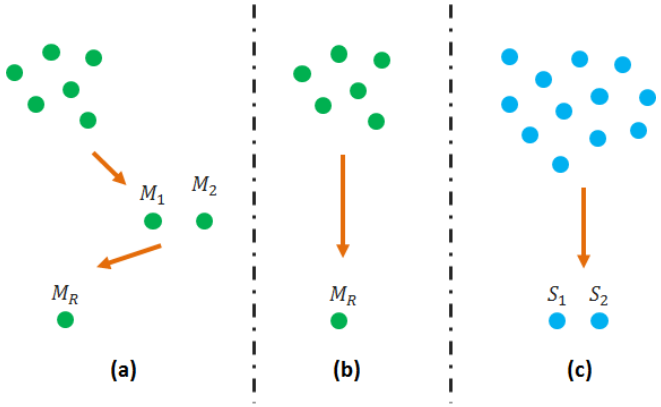

Figura 3.7: Seleção de máscara(s) e soluções no Algoritmo2. (a) A máscara M_R é criada a partir das máscaras M_1 e M_2, (b) A máscara selecionada da população é definida como M_R e (c) Duas soluções S_1 e S_2 são selecionadas da população.

$$S = S_1 \text{ x } M_R + S_2 \text{ x } (1 - M_R) \tag{3.6}$$

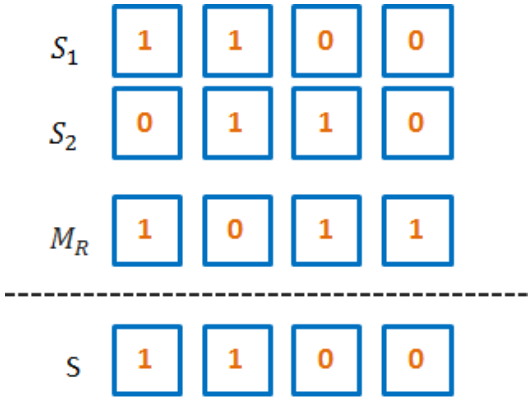

Figura 3.8: Criação de nova solução no Algoritmo2. S_1 e S_2 são as soluções selecionadas, M_R é a máscara e S é a nova solução.

Uma vez criada a nova solução, o algoritmo passa para o segundo passo: a busca por uma solução similar. Em vez de verificar, pela comparação do valor de *fitness*, se a nova solução é melhor que a pior presente na população, o algoritmo realiza uma busca para encontrar a solução mais parecida. A Figura 3.9 ilustra como é realizada a pesquisa por similaridade.

Caso não fosse encontrado nenhuma nenhuma solução similar, então a nova solução seria comparada com a pior solução presente na população. Se o *fitness* da nova solução for melhor, então a substituição ocorre.

Agora, a próxima etapa consiste na criação de uma nova máscara. Primeiro, é necessário selecionar N_S soluções da população. A seguir, é calculado a média para cada coluna entre as N_S soluções. Uma vez calculada a média, é criada a máscara baseada na Equação 3.7.

$$M_j = \begin{cases} 1, & m_j = 0 \text{ ou } m_j = 1 \\ 0, & \text{caso contrário} \end{cases} \tag{3.7}$$

em que M é a máscara criada, j é a posição do vetor e m é a média das N_S soluções.

Na Figura 3.10 é ilustrado o processo de criação da nova máscara. No exemplo, foi definido $N_S = 4$.

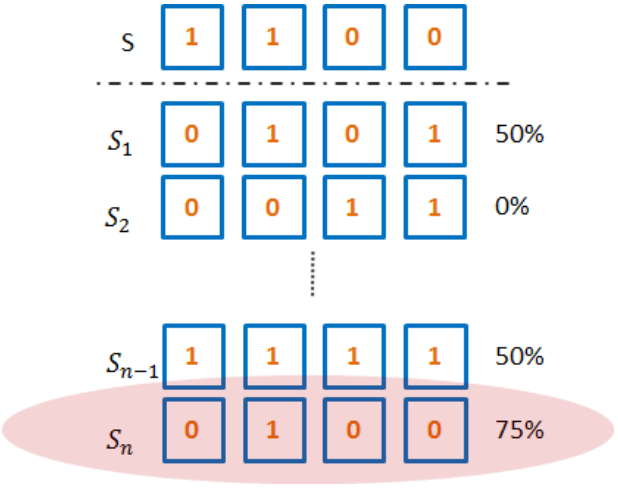

Figura 3.9: Similaridade no Algoritmo2. S_n é a solução mais similar a solução criada. Então, em vez de comparar com a solução com o pior *fitness* da população, será comparada com a solução mais similar (S_n).

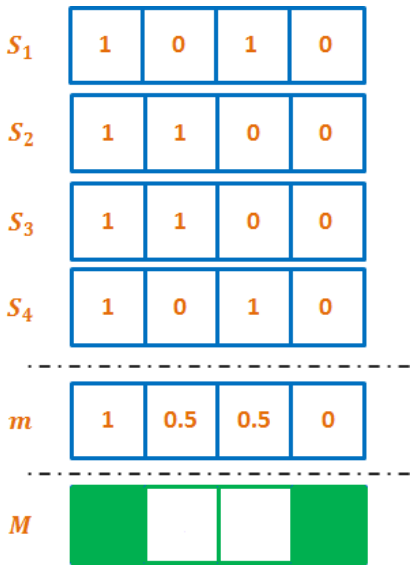

Figura 3.10: Criação de nova máscara no Algoritmo2. S_1, S_2, S_3 e S_4 são as N_S soluções selecionadas, m é a média da soluções para cada posição j e M é a máscara criada de acordo com a Equação 3.7.

O cálculo (Equação 3.8) do *fitness* da máscara é média dos *fitness* de cada uma das N_S soluções selecionadas.

$$fit_M = \frac{fit_{S_1} + fit_{S_2} + ... + fit_{S_{N_S}}}{N_S} \tag{3.8}$$

Uma vez criada a nova máscara, o algoritmo busca por uma máscara similar a máscara criada. O processo é o mesmo que ocorre quando uma nova solução é criada. Na Figura 3.11 é apresentado o fluxograma que descreve essa etapa.

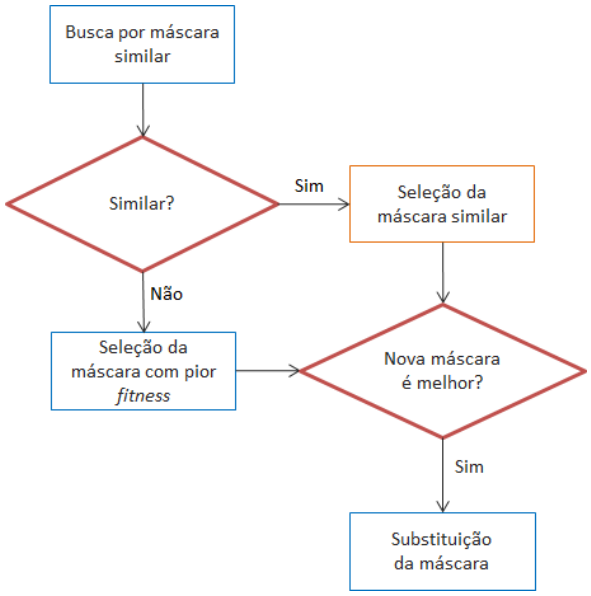

Figura 3.11: Fluxograma da etapa de similaridade no Algoritmo2. .

Como já mencionado na presente seção, o Algoritmo2 também é executado enquanto o critério de parada não é atingido. Para o Algoritmo2, é adotado apenas um critério: ocorrência de N iterações sem nenhuma solução gerada entrar na população. Diferente do Algoritmo1, a população nunca irá convergir para uma única solução, uma vez que o algoritmo2, quando utilizando o processo de similaridade, compara soluções parecidas e não com a pior presente na população.

Capítulo 4

Experimentos

Uma série de experimentos foram realizados a fim de validar empiricamente e avaliar a qualidade dos algoritmos Algoritmo1 e Algoritmo2. Esta série contou com três tipos distintos de dados: (i) dados gerados a partir de uma distribuição Gaussiana multivariada, (ii) dados gerados com a ferramenta GeneNetWeaver e (iii) dados reais obtidos em experimentos de citometria de fluxo. Para cada um dos três tipos de dados, foram gerados cinco conjuntos de dados com 100 medições. Os dados obtidos a partir da ferramenta GeneNetWeaver e os dados reais foram pré-processados antes de serem analisados.

Dados Gaussianos: Em uma distribuição Gaussiana, a variável aleatória X_i é distribuída de acordo com $X_i \sim N(\sum_k w_{ik}x_k, \sigma^2)$ em que $N(.)$ define a distribuição normal. O somatório abrange todos os pais do nó i e x_k representa o valor do nó k. Já w_{ik} indica se existe interação entre as variáveis, isto é, se X_k é pai de X_i. O valor σ^2 pode ser intepretado como ruído. Valores baixos de σ^2 indicam que o valor de X_i é totalmente determinado pelo valor de seus pais. Para gerar os dados, foi definido $w_{ik} = 1$ para indicar a existência de interação entre as variáveis e $w_{ik} = 0$ para o contrário. Também foi definido $\sigma^2 = 0{,}01$. Vale ressaltar que os dados Gaussianos são os melhores exemplos para a função objetivo do algoritmo. Uma vez que a métrica BGe assume que cada variável da rede está associada a uma distribuição Gaussiana.

Dados do GeneNetWeaver: Os dados gerados usando a ferramenta GeneNetWeaver são obtidos a partir de um sistema de equações diferenciais com ruído adicio-

nado. Este tipo de dado é mais similar com dados reais que os dados Gaussianos, uma vez que apresentam não-linearidades que são comuns em sistemas biológicos. Entretanto, em virtude dos dados serem simulados a partir de uma estrutura conhecida, o resultado já é conhecido.

Para obter os conjuntos de dados Gaussianos e GNW, foi utilizada a estrutura presente na Figura 4.1.

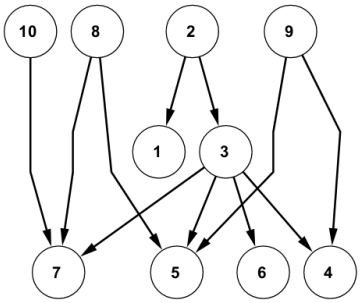

Figura 4.1: Sub-rede do Escherichia Coli. O grafo mostra uma sub-rede extraída do Escherichia Colinetwork. Ela faz parte do DREAM challenge 3 como apresentado em [Sch11]

.

Dados reias de citometria de fluxo: Citometria de fluxo é uma técnica que pode medir diferentes parametros nas células (ou outras partículas) utilizando os princípios da dispersão da luz e fluorescência. Envolve o uso de um feixe de laser que é projetado em um meio aquoso que contém as partículas. As partículas são projetadas ao feixe de laser, e, quando atingidas pela luz, dão sinais que são identificados pelos detectores. A partir desses sinais, é possível reunir diversos tipos de informações sobre a partícula. Em [SPP$^+$05] os autores utilizaram experimentos de citometria de fluxo para medir os níveis de concentração das 11 proteínas que compõem a rede ilustrada na Figura 4.2. Essa rede está envolvida na regulação da proliferação celular em células do sistema imunológico humano. Os dados obtidos com este método são considerados como dados reais neste trabalho.

Para o conjunto de dados reais, a estrutura utilizada está ilustrada na Figura 4.2.

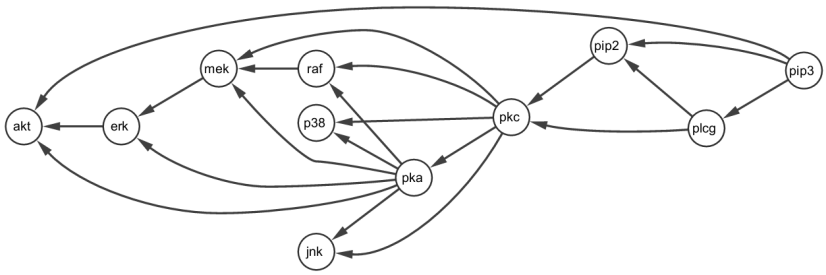

Figura 4.2: Rede de sinalização. No grafo, os nós representam as proteínas e as arestas representam as interações e as setas indicam a direção da transdução de sinal.

4.1 Configuração da Simulação

Para cada tipo de dados, aplicamos os algoritmos em cada um dos cinco conjuntos de dados. Como os dados são escassos e, portanto, a informação sobre as relações variáveis presentes nos dados é difusa, é muito provável que uma única rede resultante não possa representar adequadamente a verdadeira rede. Portanto, a simulação para cada conjunto de dados foi realizada 30 vezes e a média das redes resultantes foi definida como sendo o resultado final.

O Algoritmo1 será executado enquanto o critério de parada não for alcançado. Como já dito no Capítulo 3, pode acontecer de duas maneiras. O número de grupos utilizados em que a população será agrupada, é um parâmetro do algoritmo e foi utilizado o valor de 4. O tamanho da população foi definido com o valor de 200. A probabilidade de utilizar o operador *cg-recombination* foi ajustada em 50%.

O Algoritmo2, como mencionado no Capítulo 3, também é executado enquanto o critério de parada não for atingido. Existem duas populações distintas. Consequentemente, é necessário definir os tamanhos de cada uma delas. O tamanho da população de soluções, inicialmente foi definido com o valor de 200, mas obteve melhor desempenho com o valor de 80. O tamanho da população de máscara foi ajustado para o valor 50. A quantidade de soluções utilizadas para gerar uma máscara, também é um parâmetro do algoritmo e foi definido com o valor de 4. A probabilidade de cruzar duas máscaras foi definida com o valor de 50%.

Tanto no Algoritmo1 quanto no Algortimo2, cada solução para o problema está co-

dificada como uma matriz de adjacência que representa uma estrutura de BN, M. Em cada execução do EDA a população inicial é gerada respeitando os seguintes critérios para cada solução: (i) cada variável pode ter apenas três reguladores (pais), (ii) a rede não pode ter ciclos e (iii) os zeros (0) e uns (1) na matriz de adjacência são amostrados, respectivamente, com uma probabilidade de 65% e 35%.

A limitação do número de reguladores, conhecido como o restrição *fan-in*, é típica e têm sido aplicadas por exemplo, em [FLNP00, DMR+05]. A BN é representada por um DAG e, portanto, ciclos não são permitidos. Além disso, as GRNs são pouco conectadas e por isso são geradas entradas de zeros e uns com probabilidades diferentes.

Para cada uma das soluções propostas M é calculado um *fitness*. O *fitness* para a rede, $P(D|H)$, é obtido como proposto em [Hec96] e é conhecido como *BGe score*.

O resultado de uma execução do algoritmo é uma matriz de adjacência. Esta execução é repetida 30 vezes e a solução considerada, para cada conjunto de dados é definida como a média das 30 matrizes resultantes de adjacência R.

4.2 Critérios de Avaliação

O resultado das simulações dos algoritmos é uma coleção de redes representadas por uma matriz de adjacência. Se a rede S possui n nós, então a dimensão da matriz de adjacência M será n x n e cada entrada $m_{ij} \in \{0,1\}$indica a existência ou não de relacionamento entre s_i e s_j. Dessa coleção de matrizes, nós obtemos uma matriz média R, no qual cada entrada $r_{ij} \in \{0,1\}$indica a força do relacionamento entre os nós s_i e s_j. Em outras palavras, indica a média de cada aresta no conjunto de redes inferidas.

Para avaliar o desempenho dos algoritmos, foram comparados os seus resultados com uma rede conhecida T. Nessa rede, as entradas em $t_{ij} \in \{0,1\}$, indicam a presença ou ausência da ligação entre s_i e s_j. Para realizar a comparação, a rede T é transformada em uma matriz de adjacência $A_R(\epsilon)$, através da imposição de um limite ϵ. Cada entrada de A_R é definida pela Equação 4.1.

$$a_{ij} = \begin{cases} 1, & r_{ij} \geq \epsilon \\ 0, & r_{ij} < \epsilon \end{cases} \tag{4.1}$$

Considerando as duas matrizes, T e $A_R(\epsilon)$, é possível classificar as arestas em categorias: Uma aresta pode ser classificada como: verdadeiro positivo (*true positive* - TP), falso positivo (*false positive* - FP), verdadeiro negativo (*true negative* - TN) ou falso negativo (*false negative* - FN). A Tabela 4.1 ilustra como as arestas são classificadas nessas categorias.

Tabela 4.1: **Classificação das arestas.** Essa tabela mostra como uma aresta é classificada de acordo com os valores presente na matriz verdadeira (t_{ij}) e na matriz de adjacência (a_{ij}).

t_{ij}	a_{ij}	Categoria
0	0	TN
0	1	FP
1	0	FN
1	1	TP

Por exemplo, caso R e T sejam definidos como segue:

$$R = \begin{bmatrix} 0.2 & 0.5 \\ 0.3 & 0.4 \end{bmatrix} \quad T = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{4.2}$$

Para cada entrada ϵ, se $r_{ij} \geq \epsilon$ então a_{ij} será 1. Assim:

$$A_R(0.2) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad A_R(0.3) = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \quad A_R(0.4) = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} \quad A_R(0.5) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{4.3}$$

A Tabela 4.2 mostra os valores de TN, FP, FN e TP de acordo com a Tabela 4.1.

Na avaliação do resultado foi aplicado o critério *directed graph evaluation* (DGE). O DGE mantém a direção das arestas. Em outras palavras, caso a solução aponte uma relação entre dois nós da rede, a orientação dessa ligação é considerada.

A curva Característica de Operação do Receptor (*Receiver Operating Characteristic* - ROC) é uma representação gráfica e é obtida através da variação do limiar e plotando o número relativo de arestas TP dado por:

$$\mathrm{TP} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{4.4}$$

Tabela 4.2: **Quantitativo de TN, FP, FN e TP para o exemplo.** Essa tabela mostra os valores de TN, FP, FN e TP para o exemplo.

ϵ	TN	FP	FN	TP
0.2	0	3	0	1
0.3	1	2	0	1
0.4	2	1	0	1
0.5	3	0	0	1

contra o número de arestas FP dado por:

$$\text{FP} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{4.5}$$

para cada limiar. O ideal seria comparar todas as curvas ROC, mas isso é impraticável. Portanto, utilizamos a área sob essa curva (*Area Under the ROC Curve* - AUC). O desempenho do algoritmo é proporcional ao valor de AUC. O resultado ótimo teria um valor de 1,00 para a área. Já um resultado aleatório teria um valor em torno de 0,50. Em suma, valores altos para AUC representam resultados melhores.

Capítulo 5

Resultados

A Figura 5.1 apresenta as pontuações médias de AUC sobre os cinco conjuntos de dados para cada um dos três tipos de dados apresentados: Gaussianos, dados da ferramenta GNW e dados reais. Os valores apresentados nesse gráfico são do Algoritmo1. Os melhores resultados são observados a partir dos dados Gaussianos. Esse desempenho já era esperado visto que o modelo utilizado para a inferência é o mesmo aplicado para gerar os dados, ou seja, uma distribuição Gaussiana multivariada. Os altos valores de AUC obtidos neste conjunto de dados indica que o método de inferência proposto apresenta um bom desempenho em relação a reconstrução das estruturas de redes regulatórias.

Considerando os dados da ferramenta GNW, a inferência de redes de regulação apresenta resultados piores. Embora a estrutura utilizada para gerar os dados seja conhecida, o processo de geração dos dados produz interações não-lineares e são incompatíveis com o modelo de inferência.

Os piores resultados são obtidos quando inferimos redes a partir dos dados reais. Esse resultado é esperado visto que é muito improvável que o processo de geração dos dados de um sistema biológico real resultasse em uma distribuição Gaussiana multivariada. Além disso, a quantidade de ruído no próprio processo biológico e nas técnicas de medição não são conhecidas.

A Figura 5.2 mostra o resultado obtido com o Algoritmo2 sem utilizar a similaridade de soluções. Desse modo, o algoritmo realiza o mesmo processo do Algoritmo1. Cada solução nova gerada é comparada sempre com o pior elemento da população.

Os resultados obtidos com esse algoritmo não foram bons para todos os conjuntos de

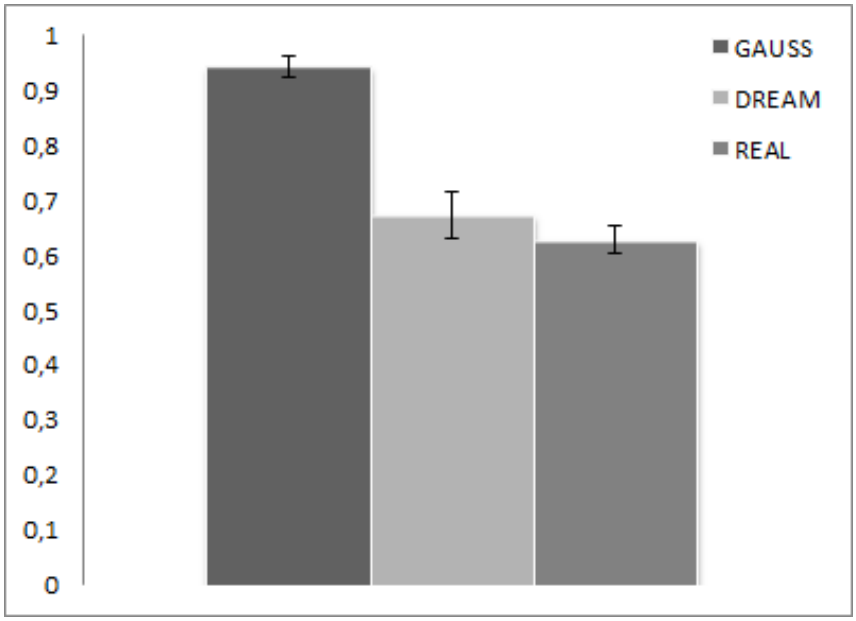

Figura 5.1: **Resultados da inferência aplicando o Algoritmo1. O melhor desempenho é a partir dos dados Gaussianos, seguido pelos dados da ferramenta GNW. Por fim, a inferência a partir dos dados reais teve o pior resultado.**

dados. A população de soluções acabou convergindo prematuramente.

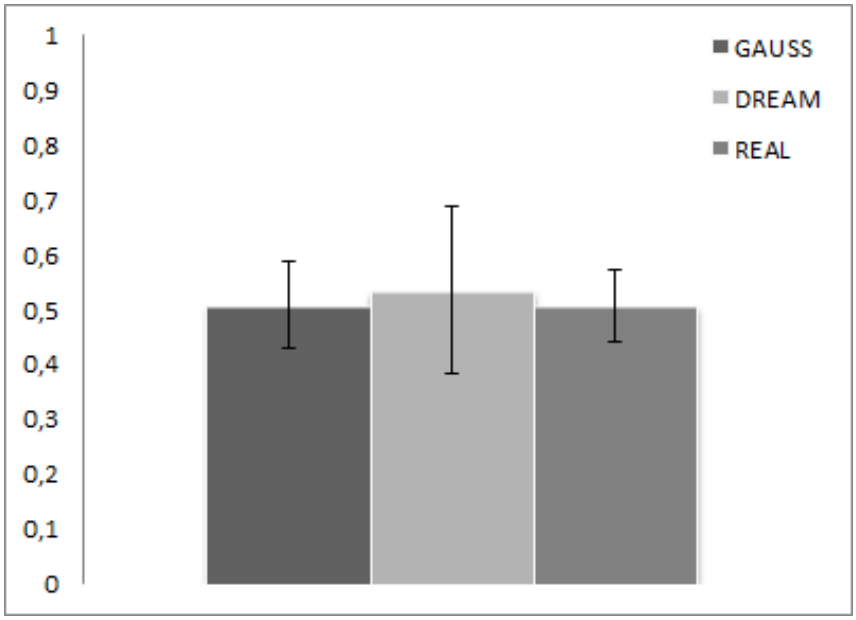

Figura 5.2: **Resultados da inferência aplicando o Algoritmo2 sem similaridade. Os resultados foram ruins para todos os três conjuntos de dados.**

Já utlizando o Algoritmo2 com similaridade no qual a solução criada é comparada com a solução mais similar dentro da população, os resultados são muito semelhantes aos alcançados com o Algoritmo1. Na Figura 5.3 é possível observar o resultado atingido.

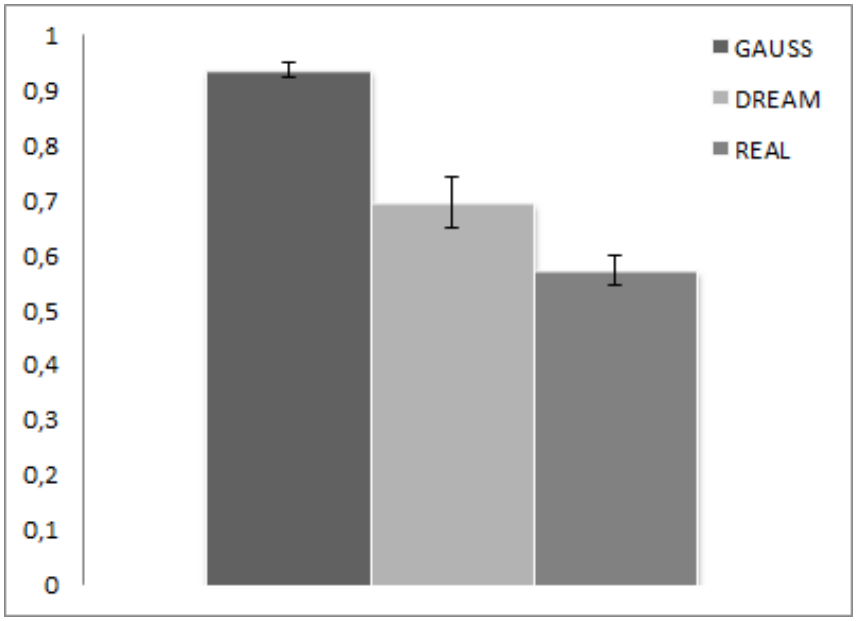

Figura 5.3: **Resultados da inferência aplicando o Algoritmo2 com similaridade. Os resultados foram semelhantes aos do Algoritmo1.**

Capítulo 6

Conclusões

6.1 Considerações Finais

O trabalho mostrou que o método proposto para inferir redes regulatórias genéticas utilizando um EDA é viável. Os algoritmos Algoritmo1 e Algoritmo2 obtiverem resultados semelhantes. Entretanto, o Algoritmo2 mostrou-se mais rápido uma vez que não utiliza o algoritmo de agrupamento. Para o Algoritmo2 foram realizados dois testes, utilizando similaridade e sem utilizá-la. Os resultados mostraram a importância de implementar essa característica no algoritmo. Realizar a comparação diretamente com o pior elemento da população levou a uma convergência prematura e, consequentemente, a piores resultados. Uma vez implementada a funcionalidade, o algoritmo foi capaz de encontrar soluções semelhantes já existentes na população. E, ao comparar as novas soluções com suas semelhantes, o algoritmo conseguiu manter a diversidade na população.

Os resultados obtidos neste trabalho são semelhantes aos obtidos em [Wer07] no que diz respeito a reconstrução da rede. Naquele trabalho foi aplicado o tradicional Markov Chain Monte Carlo (MCMC) e utilizado o mesmo conjunto de dados. Assim, uma possível comparação entre as duas abordagens deve ser realizada a fim de avaliar corretamente os resultados.

Uma vez observado que a utilização de EDAs na tarefa de inferência de GRNs são viáveis, uma comparação entre outras abordagens para inferência é desejável. Além disso, neste trabalho foi utilizado Redes Bayesianas na tarefa de modelar as estruturas das GRNs. Seria possível comparar o resultado utilizando outros modelos junto ao EDA.

Referências Bibliográficas

[AC98] Alan Agresti and Brent A. Coull. Approximate Is Better than "Exact"for Interval Estimation of Binomial Proportions. *The American Statistician*, 52(2):119–126, May 1998.

[ADHR04] Gautam Altekar, Sandhya Dwarkadas, John P. Huelsenbeck, and Fredrik Ronquist. Parallel metropolis coupled markov chain monte carlo for bayesian phylogenetic inference. *Bioinformatics*, 20(3):407–415, 2004.

[AZ12] Henrique Bunselmeyer Ferreira e Luciane M. P. Passaglia Arnaldo Zaha. *Biologia Molecular Básica*. Editora Artmed, Porto Alegre, 4nd edition, 2012.

[Bal94] S. Baluja. Population-based incremental learning. Technical Report CMU-CS-94-163, Computer Science Dept., Carnegie Mellon University, 1994.

[BIV97] J. S. De Bonet, C. L. Isbell, and P. Viola. Mimic: Finding optima by estimating probability densities. In M. Jordan, M. Mozer, and M. Perrone, editors, *Advances in Neural Information Processing Systems*, volume 9, pages 424–430. MIT Press, Cambridge, MA, 1997.

[BKK00] Atul J. Butte, Isaac S. Kohane, and I. S. Kohane. Mutual information relevance networks: Functional genomic clustering using pairwise entropy measurements. *Pacific Symposium on Biocomputing*, 5:415–426, 2000.

[Bro02] Terence A. Brown. Genomes. Oxford: Wiley-Liss, 2002.

[CLA03] Mark Cumiskey, John Levine, and Douglas Armstrong. Gene network reconstruction using a distributed genetic algorithm with a backprop local search. In Günther R. Raidl, Jean-Arcady Meyer, Martin Middendorf,

Stefano Cagnoni, Juan J. Romero Cardalda, David Corne, Jens Gottlieb, Agnès Guillot, Emma Hart, Colin G. Johnson, and Elena Marchiori, editors, *EvoWorkshops*, volume 2611 of *Lecture Notes in Computer Science*, pages 33–43. Springer, 2003.

[Cri70] Francis Crick. Central dogma of molecular biology. In *Nature*, volume 8, pages 561–563, 1970.

[Dav10] Carl Davidson. Identifying gene regulatory networks using evolutionary algorithms. *J. Comput. Sci. Coll.*, 25(5):231–237, May 2010.

[DMR+05] M. K. Dougherty, Jürgen Müller, Daniel A. Ritt, Ming Zhou, Xiao Zhen Zhou, Terry D. Copeland, Thomas P. Conrads, Timothy D. Veenstra, Kun Ping Lu, and Deborah K. Morrison. Regulation of Raf-1 by direct feedback phosphorylation. *Molecular Cell*, 17:215–224, January 2005.

[EL99] R. Etxeberria and Pedro Larrañaga. Global Optimization Using Bayesian Networks. In A. A. O. Rodriguez, M. R. S. Ortiz, and R. S. Hermida, editors, *CIMAF 99, Second Symposium on Artificial Intelligence*, Adaptive Systems, pages 332–339, La Habana, 1999.

[Emm07] Leonardo Ramos Emmendorfer. Aprendizado da ligação entre genes em computação evolutiva: Uma nova abordagem baseada em estatísticas de baixa ordem. Master's thesis, Programa de Pós-Graduação em Métodos Numéricos - UFPR, 2007.

[FK00] Nir Friedman and Daphne Koller. Being bayesian about network structure. In Craig Boutilier and Moisés Goldszmidt, editors, *UAI*, pages 201–210. Morgan Kaufmann, 2000.

[FLNP00] Nir Friedman, Michal Linial, Iftach Nachman, and Dana Pe'er. Using bayesian networks to analyze expression data. In *Proceedings of the fourth annual international conference on Computational molecular biology*, RECOMB '00, pages 127–135, New York, NY, USA, 2000. ACM.

[GB90] D. E. Goldberg and C. L. Bridges. An analysis of a reordering operator with tournament selectionon a ga-hard problem. In *Biological Cybernetics*, pages 397–405, 1990.

[GC03] Paolo Giudici and Robert Castelo. Improving markov chain monte carlo model search for data mining. *Machine Learning*, 50(1-2):127–158, 2003.

[Gol98] D. E. Goldberg. The race, the hurdle, and the sweet spot: Lessons from genetic algorithms for the automation of design innovation and creativity. Technical report, University of Illinois at Urbana-Champaign, 1998.

[GRHG99] Fernando G. Lobo Georges R. Harik and David E. Goldberg. The compact genetic algorithm. *IEEE Trans. Evolutionary Computation*, 3(4):287–297, 1999.

[Har97] Georges Raif Harik. Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms. Master's thesis, 1997.

[Har99] George Harik. Linkage learning via probabilistic modeling in the ECGA. IlliGAL report 99010, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, 1999.

[Hec94] D. Heckerman. Learning Gaussian networks. Technical Report MSR-TR-94-10, Microsoft Research, Redmond, Washington, July 1994.

[Hec96] David Heckerman. A tutorial on learning with bayesian networks. Technical report, Microsoft Research, 1996.

[HJ98] D.L. Hartl and E.W. Jones. *Genetics: principles and analysis*. Life Science Series. Jones and Bartlett Publishers, 1998.

[HLT+09] Michael Hecker, Sandro Lambeck, Susanne Töpfer, Eugene P. van Someren, and Reinhard Guthke. Gene regulatory network inference: Data integration in dynamic models - a review. *Biosystems*, 96(1):86–103, 2009.

[HMOR99] Thilo Mahnig Heinz Mühlenbein and Alberto Ochoa-Rodríguez. Schemata, distributions and graphical models in evolutionary optimization. *J. Heuristics*, 5(2):215–247, 1999.

[Hol75] J.H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, 1975.

[JBS93] Ari Juels, Shumeet Baluja, and Alistair Sinclair. The equilibrium genetic algorithm and the role of crossover. *Unpublished manuscript*, 1993.

[Kau69] S. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437–467, 1969.

[Mac67] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[Müh97] H. Mühlenbein. The equation for the response to selection and its use for prediction, 1997.

[MP96] Heinz Mühlenbein and Gerhard PaaSS. From recombination of genes to the estimation of distributions i. binary parameters. pages 178–187. Springer-Verlag, 1996.

[MPL99] David E. Goldberg Martin Pelikan and Fernando Lobo. A survey of optimization by building and using probabilistic models. Technical report, Computational Optimization and Applications, 1999.

[MS58] Mathew Meselson and Franklin W. Stahl. The replication of dna in escherichia coli. *Proc Natl Acad Sci*, 44:671–682, 1958.

[Mur01] Kevin P. Murphy. An introduction to graphical models. Technical report, May 2001.

[MY93] David Madigan and Jeremy York. Bayesian graphical models for discrete data, 1993.

[PGCP00] Martin Pelikan, David E. Goldberg, and Erick Cantú-Paz. Linkage problem, distribution estimation, and bayesian networks. *Evolutionary Computation*, 8(3):311–340, 2000.

[PLnS13] Concha Bielza Pedro LarrañAga, Hossein Karshenas and Roberto Santana. A review on evolutionary algorithms in bayesian network learning and inference tasks. *Inf. Sci.*, 233:109–125, June 2013.

[PM99] Martin Pelikan and Heinz Mühlenbein. The bivariate marginal distribution algorithm, 1999.

[RN99] Miguel Rocha and José Neves. Preventing premature convergence to local optima in genetic algorithms via random offspring generation. In *Proceedings of the 12th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems: Multiple Approaches to Intelligent Systems*, IEA/AIE '99, pages 127–136, Secaucus, NJ, USA, 1999. Springer-Verlag New York, Inc.

[Sch11] Marbach D. Floreano D. Schaffter, T. Genenetweaver: In silico benchmark generation and performance profiling of network inference methods. *Bioinformatics 27*, 16:2263–2270, June 2011.

[SPP+05] Karen Sachs, Omar Perez, Dana Pe'er, Douglas A. Lauffenburger, and Garry P. Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, April 2005.

[Wer07] Adriano Velasque Werhli. Reconstruction of gene regulatory networks from postgenomic data. Master's thesis, School of Informatics of University of Edinburgh, 2007.

[YpTlKG07] Chen Ying-ping, Yu Tian-li, Sastry Kumara, and David E. Goldberg. A survey of linkage learning techniques in genetic and evolutionary algorithms, 2007.

[YSG05] Tian-Li Yu, Kumara Sastry, and David E. Goldberg. Linkage learning, overlapping building blocks, and systematic strategy for scalable recombination. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, GECCO '05, pages 1217–1224, New York, NY, USA, 2005. ACM.