UNIVERSIDADE ESTADUAL DE SANTA CRUZ
PRO-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM MODELAGEM COMPUTACIONAL EM CIÊNCIA E TECNOLOGIA

DENILSON OLIVEIRA MELO

UM MÉTODO BASEADO NO ALGORITMO FIREFLY PARA BICLUSTERING DE DADOS DE EXPRESSÃO GÊNICA

ILHÉUS-BA
2019

DENILSON OLIVEIRA MELO

UM MÉTODO BASEADO NO ALGORITMO FIREFLY PARA BICLUSTERING DE DADOS DE EXPRESSÃO GÊNICA

Dissertação apresentada ao Programa de Pós-Graduação em Modelagem Computacional em Ciência e Tecnologia da Universidade Estadual de Santa Cruz, como parte das exigências para obtenção do título de Mestre em Modelagem Computacional em Ciência e Tecnologia.

Orientador: Prof. Dr. Paulo Eduardo Ambrósio

ILHÉUS-BA
2019

M528 Melo, Denilson Oliveira.
Um método baseado no algoritmo firefly para biclustering de dados de expressão gênica / Denilson Oliveira Melo. – Ilhéus, BA: UESC, 2019.
45 f. : il.

Orientador: Paulo Eduardo Ambrósio.
Dissertação (mestrado) – Universidade Estadual de Santa Cruz. Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia.
Inclui referências.

1. Algorítmos. 2. Cluster (Sistema de computador). 3. Expressão gênica. 4. Programação heurística. I. Título.

CDD 511.8

DENILSON OLIVEIRA MELO

UM MÉTODO BASEADO NO ALGORITMO FIREFLY PARA BICLUSTERING DE DADOS DE EXPRESSÃO GÊNICA

Ilhéus-BA, 25/02/2019

Comissão Examinadora

Prof. Dr. Paulo Eduardo Ambrósio
UESC
(Orientador)

Prof. Dr. Esbel Tomás Varelo Orellana
UESC

Prof. Dr. Rodrigo Antonio Faccioli
Centro Universitário Barão de Mauá

Agradecimentos

- Aos meus pais Jonatas e Dennise e ao meu irmão Jonatas, pelo exemplo, criação, e por sempre me apoiarem em todos os aspectos.
- À Ellen Dias por sempre me incentivar em momentos difíceis, e por tornar os momentos de felicidade ainda mais felizes.
- Aos amigos por todo apoio, em especial a Jackson Farias com quem sempre posso contar.
- Ao professor doutor Paulo Eduardo Ambrósio pela orientação e pelas contribuições no desenvolvimento deste trabalho, e à equipe do PPGMC por fornecer todo o suporte necessário par ao desenvolvimento da pesquisa.
- Aos colegas pelos conhecimentos compartilhados durante o curso
- À UESC, pela formação acadêmica na graduação e agora no mestrado.
- À FAPESB pelo incentivo financeiro.

Um Método Baseado no Algoritmo Firefly para Biclustering de Dados de Expressão Gênica

Resumo

Dados de expressão gênica são frequentemente estudados através da análise de agrupamento por *biclustering*, onde genes e condições experimentais são agrupadas simultaneamente com o objetivo de detectar e inferir funções de módulos gênicos e redes de transcrição. Porém a tarefa de encontrar *biclusters* é um problema NP-Difícil, onde o número de operações aumenta exponencialmente com o tamanho dos dados. Uma maneira de resolver problemas dessa complexidade é por meio de heurísticas. O Algoritmo *Firefly* é uma meta-heurística populacional inspirada nos padrões de emissão de luz dos vaga-lumes, que possui algumas características desejáveis para o problema em questão. No presente trabalho foi desenvolvido um método que utiliza exclusivamente o Algoritmo *Firefly* para *biclustering* de dados de expressão gênica. O método proposto foi avaliado em dados sintéticos e mostrou-se capaz de efetuar o *biclustering*, porém com algumas melhorias necessárias. Foi esperado que o método encontrasse múltiplos *biclusters* simultaneamente devido a dinâmica de subdivisão de enxame do Algoritmo *Firefly*, o que não ocorreu, porém o método mostrou-se robusto ao ser capaz de encontrar múltiplos *biclusters* utilizando uma abordagem sequencial.

Palavras-chave: algoritmo firefly. biclustering. expressão gênica. meta-heurística.

A Method Based on the Firefly Algorithm for Biclustering of Gene Expression Data

Abstract

Gene expression data are often studied through cluster analysis via biclustering, where genes and experimental conditions are grouped simultaneously with the goal to detect and infer gene modules and transcription networks. However, the task of finding biclusters is an NP-Hard problem, where the number of operations increases exponentially with the size of the data. One way to solve problems with this complexity is through heuristics. The Firefly Algorithm is a populational metaheuristic, inspired by the light emission patterns of fireflies, that presents has some desirable characteristics for the problem in question. In the present work a method was developed that exclusively employs the Firefly Algorithm to find biclusters in gene expression data. The proposed method was evaluated with synthetic data and is valid, with some necessary improvements. It was also expected that the method would find multiple biclusters simultaneously, due to the dynamics of swarm subdivision of the Firefly Algorithm, which did not occur, however, the method proved to be robust as being able to find multiple biclusters using a sequential approach.

Keywords: firefly algorithm. biclustering. gene expression. metaheuristic

Sumário

1 Introdução

As grandes quantidades de dados obtidos através dos microarranjos de DNA permitem análises robustas de dados de expressão gênica, como a análise de agrupamento, ou *clustering*, que visa agrupar genes que apresentam padrões de expressão similares. Dentre as diversas aplicações de *clustering* em dados de expressão estão: a descoberta de drogas, a exploração de subtipos de doenças, a identificação de genes e módulos, a elucidação de redes biológicas, a análise de modularidade, a anotação funcional de genes, entre outros (XIE et al., 2018). Apesar de sua popularidade e das mais diversas possibilidades de aplicações, o *clustering* deve ser aplicado cuidadosamente, pois possui limitações quanto a características dos dados a serem analisados.

Uma das características do *clustering*, que em certos casos pode ser considerada como limitação, é que o agrupamento é feito considerando todos os atributos dos objetos, isto implica que, ao se considerar todos os atributos, agrupar objetos parcialmente similares é impossível: ou os objetos são similares em todos os atributos ou não são. Com dados de expressão gênica, isto impõe que o *clustering* idealmente seja aplicado em dados com condições amostrais homogêneas, isto é, em dados que já se assume algum tipo de correlação prévia (ASYALI et al., 2006). Para superar essa limitação do *clustering* o *biclustering* pode ser utilizado.

No *biclustering* as linhas e colunas são agrupadas simultaneamente, permitindo que sejam formados grupos que considerem também subconjuntos de atributos, podendo então ser aplicado em dados com condições experimentais heterogêneas, ou até, dado tratamento estatístico adequado, em dados de diferentes experimentos. Porém o biclustering é um problema NP-Difícil, e portanto não possui algoritmo ótimo que o resolva.

Pontes et al. (2015a) apresenta uma visão geral das técnicas de biclustering mais relevantes, onde são descritos, classificados e avaliados um total de 47 algoritmos. Algumas das conclusões do trabalho citado foram: a inexistência de algoritmo superior, a dificuldade de escolher algum método adequado na hora da análise, devido à grande variedade de propostas e da ausência de um método eficaz para compará-los, e que a maioria dos algoritmos propostos são baseados em procedimentos de otimização inspirados na natureza. Entre os algoritmos inspirados na natureza, 9 das 12 propostas citadas são baseadas em algoritmos evolutivos. Esta popularidade da aplicação de algoritmos inspirados na natureza pode ser atribuída à sua flexibilidade. Geralmente esses algoritmos não fazem suposições sobre o problema a ser resolvido, tornando-os métodos genéricos que podem ser adaptados para resolver os mais diversos problemas.

O Algoritmo *Firefly* (FA ou FFA) proposto por Yang (2009), é uma meta-heurística populacional inspirada nos padrões de emissão de luz de vaga-lumes. Ele pode ser classificado como um algoritmo baseado em inteligência de enxames (*Swam Intelligente*) (YANG; HE, 2018), em que a busca inteligente por soluções se dá através de interações básicas entre indivíduos. Na inteligência de enxames não há rotinas de coordenação centralizada, cada indivíduo possui um comportamento bem definido, e o aspecto inteligente da busca surge das interações básicas entre os indivíduos (ZOMAYA, 2006).

No Algoritmo Firefly é definido a partir de três princípios: (a) todos os vaga-lumes são unissex; (b) dado dois vaga-lumes, o de menor brilho será atraído pelo de maior brilho; (c) a intensidade da atração varia com a distância. A ideia é que cada vaga-lume represente uma solução, e que a intensidade do seu brilho seja proporcional à qualidade da sua solução, de forma que, os vaga-lumes com as melhores soluções atraiam outros próximos, que por sua vez irão melhorar suas próprias soluções ao incorporar elementos da solução do vaga-lume pelo qual fora atraído.

Considerando os recentes avanços na aplicação do Algoritmo *Firefly* (YANG; HE, 2018), e suas características, o presente estudo tem como objetivo avaliar a capacidade do Algoritmo *Firefly* para encontrar *biclusters* em dados de expressão gênica.

O presente trabalho está organizado da seguinte forma. Os capítulos 2 e 3 são destinados a fundamentação teórica dos conceitos envolvidos neste trabalho, sendo o capítulo 2 destinado a apresentar o conceito de *biclustering* no contexto de expressão gênica, e o capítulo 3 destinado a descrever o Algoritmo *Firefly*. No capítulo 4, fazendo uso dos conceitos apresentados nos capítulos anteriores, será apresentado a modelagem do problema de *biclustering* para ser solucionado com o Algoritmo *Firefly*. Em seguida no capítulo 5 é descrito como o método proposto foi avaliado. Logo após, no capítulo 6, serão apresentados os resultados obtidos, junto com discussões relevantes observadas. E por fim no capítulo 7 serão apresentadas as conclusões e os comentários finais.

2 Biclustering em Dados de Expressão Gênica

A análise de agrupamento, ou *clustering*, é uma tarefa de análise exploratória de dados estatísticos, cujo objetivo é agrupar objetos, tais que, objetos em um mesmo grupo sejam similares entre si, e objetos em grupos distintos sejam dissimilares. A análise de agrupamento é aplicada em diversas áreas do conhecimento, como computação, reconhecimento de padrões, aprendizado de máquina, recuperação da informação, entre outras. Os métodos para análise de agrupamento em sua maioria são os mesmos nas diferentes áreas, as particularidades se dão nas finalidades de análise e na interpretação dos resultados. Porém, algumas aplicações possuem necessidades de agrupamento mais robustas, como é o caso da análise de dados de expressão gênica.

Por apresentar o maior interesse em *biclustering*, atestado pelo número de publicações (XIE et al., 2018), os conceitos abordados neste capítulo serão contextualizados no âmbito de dados de expressão gênica. Cada região codificadora de DNA pode ser vista como um receita para se produzir respectiva proteína. Desconsiderando a água, as proteínas compõem a maior parte massa de uma célula, e são responsáveis por praticamente todo o seu funcionamento (ALBERTS et al., 2010, p. 5), como por exemplo, existem proteínas responsáveis apenas pela movimentação, outras que fazem a manutenção da estrutura, e outras que quebram nutrientes para serem absorvidos (ALBERTS et al., 2010, p. 6).

Um dos fundamentos da biologia de sistemas é o fato de que genes e proteínas não trabalham isoladamente, mas sim em redes complexas de módulos (De Smet; MARCHAL, 2010). Uma das razões pela qual a análise de agrupamento é utilizada é a possibilidade e a relevância de quantificar a expressão gênica do DNA da célula. A expressão gênica é o processo em que a informação codificada no gene é convertida em um produto gênico funcional, geralmente proteína (AL-AKWAA, 2012). As condições bioquímicas da célula e do meio que a envolve determinam quais genes deverão ser expressos ou não, e em qual quantidade, de forma que o produto gênico a ser gerado contribua para sua aptidão (BROWN; BOTSTEIN, 1999).

Uma das técnicas disponíveis para quantificar os níveis de expressão gênica é o microarranjo de DNA, que permite ao pesquisador captar o estado atual (*snapshot*), dos níveis de expressão dos genes de sua escolha, sob também quais condições experimentais desejar. O microarranjo de DNA, ou *microarray*, é uma tecnologia que permite medir os níveis de expressão de milhares de genes simultaneamente em um único experimento (HARRINGTON et al., 2000). Ele consiste de uma superfície sólida com milhares de pontos (*spots*), cada ponto contendo fitas simples acopladas de um gene específico.

Para quantificar os níveis de expressão gênica, os microarranjos fazem proveito do processo de hibridização, no qual fitas simples de sequências de ácidos nucleicos (DNA, RNAm, cDNA entre outros) são complementadas por outra fita correspondente. A expressão é quantificada com base na abundância de transcritos hibridizados detectados através fluorescência (MELOUANE et al., 2018). Um microarranjo é montado de acordo com genes conhecidos *a priori*, e os pontos são organizados em um padrão reprodutível para que uma mesma configuração seja utilizada com outras amostras. O resultado do experimento de microarray é uma matriz de dados de expressão, onde cada linha representa um gene, e cada coluna representa uma condição experimental, de forma que cada célula da matriz representa o nível expressão de determinado gene sob determinada condição (MADEIRA; OLIVEIRA, 2004).

Com dados de expressão gênica, a análise de agrupamento pode ser aplicada com o objetivo de inferir funções de módulos e das redes de transcrição que os regulam (De Smet; MARCHAL, 2010). A ideia é reconstruir ou modelar esses módulos através de agrupamentos para facilitar a compreensão dos processos celulares (BONNEAU, 2008). Madeira e Oliveira (2004) apresenta alguns dos objetivos que são buscados nesses dados, sendo eles: agrupar os genes com base na expressão sob as condições, ou agrupar as condições com base na expressão dos genes; classificar um novo gene, visto que seu padrão de expressão corresponde com outro já classificado, ou classificar amostras caso os padrões detectados já foram classificados. Os microarranjos também são utilizados por pesquisadores e médicos para finalidades como: diagnose e prognose, estudar alterações provocadas ao decorrer de determinada doença (fisiopatologia), ou escolher o melhor tipo de tratamento para determinada patologia (ASYALI et al., 2006).

Nas próximas seções serão apresentados conceitos e definições relevantes para *biclustering*. Este capítulo será finalizado com uma descrição do estado da arte referente à *biclustering* em dados de expressão.

2.1 Definição

Seja M uma matriz composta por um conjunto de linhas L e de colunas C, um *bicluster* B nada mais é que uma submatriz de M, composta também por um subconjunto de linhas $I \subseteq L$ e outro subconjunto de colunas $J \subseteq C$, podendo ser representado como como:

$$B = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1|J|} \\ b_{21} & b_{22} & \dots & b_{2|J|} \\ \vdots & \vdots & \ddots & \vdots \\ b_{|I|1} & b_{|I|2} & \dots & b_{|I||J|} \end{bmatrix}. \tag{1}$$

O que diferencia o conceito *bicluster* do de submatriz é que um *bicluster* possui valores que seguem algum padrão, ou entre as linhas, ou entre as colunas, ou em ambas. Vale notar que em aplicações reais, dificilmente os valores de um *bicluster* irão assumir precisamente os valores do padrão. O padrão representa a tendência do *bicluster*, e no geral quanto mais próximo dessa tendência estiverem os valores, melhor será a coerência do *bicluster*.

Na literatura os *biclusters* são caracterizados de acordo com o padrão apresentado em seus valores (AGUILAR-RUIZ, 2005). Estes padrões estão são ilustrados na Figura 1, sendo eles: valores constantes, valores constantes nas linhas ou nas colunas, valores coerentes aditivos e multiplicativos.

Suponha um bicluster $B = (I, J)$ com um valor típico π, o bicluster B será do tipo valores constantes (Figura 1a) caso, para todo $i = \{1, 2, ..., |I|\}$ e $j = \{1, 2, ..., |I|\}$:

$$b_{ij} = \pi. \tag{2}$$

Considere B_i um fator constante a ser adicionado para cada linha do bicluster, e α_i outro coeficiente a ser multiplicado para cada linha do bicluster. Um *bicluster* seguirá o padrão de linhas constantes (Figura 1b) caso seus valores sejam descritos por:

$$\begin{aligned} b_{ij} &= \pi + B_i, \\ b_{ij} &= \pi \times \alpha_i. \end{aligned} \tag{3}$$

Analogamente está definido o padrão de colunas constantes (Figura 1c), porém com os coeficientes B_j e α_j para as colunas:

$$\begin{aligned} b_{ij} &= \pi + B_j, \\ b_{ij} &= \pi \times \alpha_j. \end{aligned} \tag{4}$$

Outro padrão é caracterizado pela presença simultânea dos coeficientes para as linhas e para as colunas, nomeado como padrão de valores coerentes, podendo ser aditivo (Figura 1d) ou multiplicativo ser aditivo (Figura 1e); é dado por:

$$\begin{aligned} b_{ij} &= \pi + B_i + B_j, \\ b_{ij} &= \pi \times \alpha_i \times \alpha_j. \end{aligned} \tag{5}$$

Aguilar-Ruiz (2005) propõe não utilizar um valor π típico do bicluster, mas sim que cada linha ou coluna possua seu valor típico π_i ou π_j. Com essa mudança, é caracterizado apenas dois padrões de dados de expressão: deslocamento (*shifting*) e escala (*scale*). Um *bicluster* seguindo o padrão de deslocamento possui a seguinte forma:

$$\begin{aligned} b_{ij} &= \pi_j + B_i, \\ b_{ij} &= \pi_i + B_j. \end{aligned} \tag{6}$$

Figura 1 – Exemplos de padrões de *biclusters*: (a) constante, (b) constante nas linhas, (c) constante nas colunas, (d) coerente aditivo, (e) coerente multiplicativo, e (f) coerência linear (GAN et al., 2008).

x	y	z	w
1.2	1.2	1.2	1.2
1.2	1.2	1.2	1.2
1.2	1.2	1.2	1.2
1.2	1.2	1.2	1.2

(a)

x	y	z	w
1.2	1.2	1.2	1.2
2.0	2.0	2.0	2.0
1.5	1.5	1.5	1.5
3.0	3.0	3.0	3.0

(b)

x	y	z	w
1.2	2.0	1.5	3.0
1.2	2.0	1.5	3.0
1.2	2.0	1.5	3.0
1.2	2.0	1.5	3.0

(c)

x	y	z	w
1.2	2.2	0.2	3.2
2.0	3.0	1.0	4.0
1.4	2.4	0.4	3.4
2.4	3.4	1.4	4.4

(d)

x	y	z	w
1.0	2.0	0.5	1.5
2.0	4.0	1.0	3.0
1.4	2.8	0.7	2.1
2.4	4.8	1.2	3.6

(e)

x	y	z	w
1.0	2.1	0.6	1.7
2.0	4.1	1.1	3.2
1.4	2.9	0.8	2.3
2.4	4.9	1.3	3.8

(f)

Fonte: Gan et al. (2008)

Já o padrão de escala:

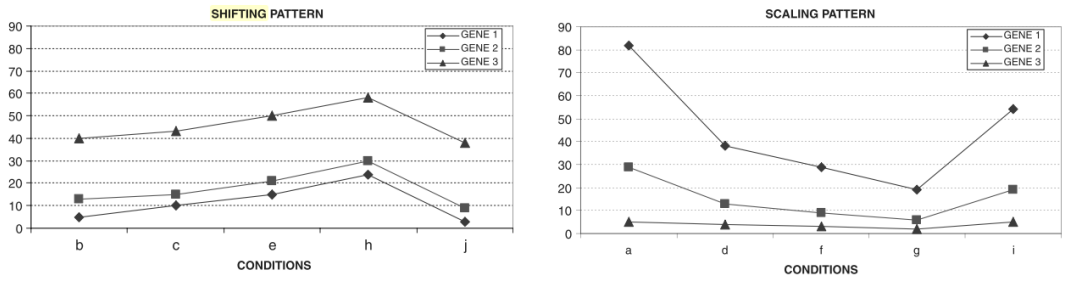
$$\begin{aligned} b_{ij} &= \pi_j \times \alpha_i, \\ b_{ij} &= \pi_i \times \alpha_j. \end{aligned} \tag{7}$$

Por fim, ambos os padrões de deslocamento e escala podem ocorrer simultaneamente (*shifing-scale*):

$$\begin{aligned} b_{ij} &= \pi_j \times \alpha_i + B_i, \\ b_{ij} &= \pi_i \times \alpha_j + B_j. \end{aligned} \tag{8}$$

Os modelos de deslocamento e/ou escala representam os padrões mais comuns no comportamento dos genes. Na Figura 2 é ilustrado visualmente como esses padrões se comportam.

Figura 2 – Representação visual dos padrões de deslocamento (esquerda) e escala (direita)



Fonte: Aguilar-Ruiz (2005)

2.2 Método Cheng Church

O *biclustering* foi comprovado como de classe pelo menos NP-Difícil (TANAY et al., 2002). Um problem dessa classe não possui algoritmo determinístico que o resolva em tempo polinomial, somente exponencial, o que torna exaustar as combinações um processo inviável (HOOS; STÜTZLE, 2004). Para resolver problemas dessa natureza, geralmente são empregada meta-heurísticas de otimização (HOOS; STÜTZLE, 2004), ou outras técnicas para reduzir o tamanho do espaço de busca e exaustar as possibilidades, além de técnicas que dependem de conhecimentos específico sobre o problema.

Apesar do conceito de *biclustering* ter sido apresentado por Hartigan (1972) (*direct clustering*), Cheng e Church (2000) foi o pioneiro em aplicar *biclustering* para dados de expressão. Sua técnica consiste em otimizar uma métrica de homogeneidade de bicluster, a média do quadrado do erro, ou *Mean Squared Residue* (MSR). O MSR é uma medida de resíduo, calculada a partir da variância de cada elemento do *bicluster* em relação à média da sua linha, coluna, e da média geral do *bicluster*. Suponha um *bicluster* B formado por um par de subconjuntos, um de linhas I e outro de colunas J, tal que $B = (I, J)$; considere também dois contadores para os subconjuntos: $i = \{1, 2, ..., |I|\}$ e $j = \{1, 2, ..., |J|\}$; O resíduo de cada elemento é dado por:

$$b_{ij} - b_{Ij} - b_{iJ} + b_{JJ}, \tag{9}$$

onde b_{ij} é o elemento atual, b_{Ij} é a média da coluna atual, b_{iJ} é a média da linha atual e b_{IJ} a média do bicluster. Com esse resíduo, o MSR é calculado como a média da variância, dado por:

$$MSR(I, J) = \frac{1}{|I||J|} \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} (b_{ij} - b_{Ij} - b_{iJ} + b_{JJ})^2. \tag{10}$$

Por ser uma métrica baseada em erro, o MSR mede quão próximo de um mesmo valor o bicluster está, portanto biclusters com MSR 0 são considerados ideais. Vale notar que a medida de qualidade é independente do processo de biclustering em si, portanto pode ser utilizada por outros métodos.

O método de biclustering apresentado por Cheng e Church consiste em começar a busca à partir de um *bicluster* contendo todas as linhas e colunas da matriz original, e a cada iteração remover elementos, efetuando os seguintes passos:

1. Calcula o MSR de cada *bicluster* possível ao se remover ou adicionar cada linha ou coluna.

2. Escolhe a possibilidade com o melhor (menor) MSR.

Esses passos são realizados até se obter um $MSR <= \delta$, sendo δ o resíduo considerado aceitável, definido pelo usuário. Esse método faz uso do fato de que é sempre possível diminuir o resíduo do *bicluster* ao se remover uma linha ou coluna, portanto sempre que $MSR > \delta$ o método irá remover linhas ou colunas. Após encontrar o *bicluster*, o mesmo é mascarado na matriz de dados original com valores aleatórios, para que não seja encontrado novamente, e processo pode ser repetido até número de vezes indicado pelo usuário.

Os passos descritos apresentam uma versão idealizada do método, segundo Cheng e Church (2000) avaliar todas as possibilidades de *bicluster* representa um alto custo computacional. Para evitar tal custo, foram aproveitadas algumas características do MSR para desenvolver um método otimizado. Esse método otimizado se baseia em verificar apenas a contribuição de cada linha e coluna para o resíduo do *bicluster*, ao invés de avaliar as possibilidades de *biclusters*. O método inicial e o otimizado não são equivalentes, porém representam uma mesma estratégia de busca de *biclusters*. O método otimizado apesar de ser computacionalmente superior, ele é exclusivo apenas para a métrica MSR, enquanto que o método inicial pode ser utilizado para qualquer métrica de qualidade.

Por sempre escolher a melhor alternativa local, o algoritmo de Cheng e Church é classificado como guloso, é uma característica de métodos gulosos que eventualmente o método fique preso em um mínimo local, sem nenhuma garantia de encontrar outros mínimos locais melhores ou algum mínimo global. Como os *biclusters* não são conhecidos *a priori*, o mínimo local é um indicativo natural de que a busca foi encerrada, um mínimo local neste contexto indica que o método não consegue mais melhorar tal solução. Computacionalmente um mínimo local é identificado ao detectar que o método removeu uma linha ou a coluna e em seguida adicionou a mesma.

Tradicionalmente um método guloso ficar preso em um mínimo local é uma característica indesejada caso o objetivo seja a otimização global, porém para *biclustering* a otimização local é desejada, portanto ficar preso em um mínimo local é um mecanismo que irá preservar uma possível boa solução e que indica a conclusão da busca. Porém em alguns casos isso ainda pode ser um problema, mas não em relação a algum mínimo global, mas sim caso o método fique preso em um mínimo local prematuro, impedindo que outros mínimos locais melhores sejam encontrados. Alguns mínimos locais poderão não ser encontrados pois a busca gulosa irá ignorar certas regiões do espaço de busca caso para alcançar tais regiões passos sub-ótimas devem ser tomados.

O método de Cheng e Church foi um método pioneiro em aplicar *biclustering* em dados de expressão, revelando uma aplicação promissora, porém o método apresentado possui várias limitações. Esse desempenho inferior pode ser atribuído parte para a natureza gulosa do método, e parte para as limitações da métrica MSR. Em diversos

trabalhos o desempenho do algoritmo Cheng e Church é utilizado como comparação, porém em quase todos demonstra uma performance aceitável apenas para os casos mais básicos (PRELIĆ et al., 2006), (EREN et al., 2013), (DALE et al., 2018).

A métrica MSR não é capaz de encontrar padrões de escala, e consequentemente deslocamento e escala simultaneamente. O valor δ depende dos dados e não possui valor superior. Um valor de msr igual a 1 pode representar um bicluster de alta qualidade em um conjunto de dados, já em outro pode significar um resultado inaceitável. Para detectar padrões de escala foi proposto o SMSR, porém este detecta apenas padrões de escala, e não de deslocamento. Pontes et al. (2015b) apresenta uma avaliação de diversas métricas de qualidade utilizadas para *biclustering*, mostrando suas vantagens e desvantagens.

2.3 Estado da arte

Para análise de dados de expressão o processo de *biclustering* é um dos passos intermediários do *pipeline* de análise de dados de expressão, após feito o biclustering os *biclusters* obtidos devem ser interpretados de acordo com os objetivo do estudo (XIE et al., 2018). Com o objetivo de anotação funcional por exemplo, os *biclusters* podem ser enriquecidos com bases de dados de anotação funcional como o *DAVID — Database for Annotation, Visualization and Integrated Discovery* (HUANG et al., 2009) — ou o *g:Profiler* (REIMAND et al., 2016). Para inferir os mecanismos responsáveis por co-regular os genes bases de dados como o *STRING database* (SZKLARCZYK et al., 2017) ou algoritmos como o *Inferelator* (BONNEAU et al., 2006) podem ser utilizados.

Como exemplos de aplicações que fazem uso dessas ferramentas, Yoon et al. (2013) em seu trabalho fez uso de algumas dessas ferramentas para conduzir sua pesquisa, onde para biclusterin utilizou o método *cMonkey* (REISS et al., 2006), fez a anotação das redes regulatórias utilizando *Inferelator* (BONNEAU et al., 2006), e utilizou o *Cytoscape* (SHANNON et al., 2003) para visualizar as redes resultantes. Guttman et al. (2009) utilizou o algoritmo de biclustering *Statistical-Algorithmic Method for Bicluster Analysis* (SAMBA) (TANAY et al., 2002) para indentificar funções de *Long intervening noncoding RNAs* (lincRNAs) em ratos.

Atualmente os *Next-Generation Sequencers* (NGS), ou sequenciadores de nova geração, ganharam notoriedade para *profiling* de expressão gênica, onde diversos autores relatam diversas vantagens desses em relação aos microarranjos de DNA (WANG et al., 2009), (THERMES, 2014), (HURD; NELSON, 2009). Essas vantagens se dão pela fundamentação da tecnologia, enquanto que os microarranjos de DNA medem a abundância de RNA através da fluorescência, os NGS lêem diretamente milhões de sequências de transcritos (THERMES, 2014). Entre essas vantagens está o fato de que os NGS, por

fazerem o sequenciamento, permitem quantificar transcritos sem conhecimento à priori do DNA ou dos possíveis transcritos da amostra, enquanto que os microarranjos de DNA precisam ser montados com os genes que se queira medir (HURD; NELSON, 2009). Outra vantagem é que com o NGS os transcritos são quantificados de maneira absoluta, enquanto que nos microarranjos os níveis de expressão são relativos. (WANG et al., 2009).

Os NGS também possuem suas próprias limitações, e nem todas a vantagens ainda foram alcançadas de forma plena. Uma das qualidades propostas e desejáveis dos NGS é a possibilidade de quantificar transcritos de maneira absoluta, porém quantificação absoluta ainda não se mostrou confiável, entretando a quantificação relativa é válida (SU et al., 2014). Limitações referentes à capacidade de leitura dificultam a quantificação de transcritos não abundantes o suficiente (ZHAO et al., 2014), isso por exemplo dificulta a quantificação e a categorização de transcritos alternativos (SU et al., 2014). Uma barreira a nível de análise é quantidade de dados crus gerados; Zhao et al. (2014) reporta que para os experimentos que fez, o tamanho típico dos dados crus gerados por *microarrays* foi de 5 *megabytes*, enquanto que para RNA-Seq teve-se em média 23 *gigabytes*.

Mesmo com os avanços em NGS, os microarranjos de DNA ainda são extensivamente utilizados, tanto por seu baixo custo, quanto pela maturidade da técnica. As possibilidades com microarranjos ainda não foram exaustadas. Para *biclustering*, os dados crus de microarray estão mais próximos da matriz de expressão do que os dados de NGS, porém dado um tratamento correto é possível chegar em algo equivalente à matriz de expressão com dados de sequenciamento.

3 Algoritmo Firefly

Os vaga-lumes são insetos da família *Lampyridae*, da ordem *Coleoptera* (besouros), caracterizados pela capacidade de emitir luz. Eles possuem um órgão luminescente localizado na região inferior do abdômen, onde produzem uma luz fosforescente fria, em que quase 100% da energia resulta em luz, com uma quantidade minúscula de calor. Essa luz é emitida em um padrão rítmico característico da espécie, e possui diferentes funções que ainda são debatidas (DURKOTA, 2011). A emissão de luz possui diversas funções, como atrair parceiros sexuais, atrair presas, ou afastar predadores. O Algoritmo *Firefly* proposto por Yang (2009) se baseia nos comportamentos de emissão de luz dos vaga-lumes para resolver problemas de otimização.

Neste capítulo serão apresentados os conceitos relevantes que definem o Algoritmo *Firefly*. Na primeira secção serão introduzidos os conceitos de meta-heurística e otimização; em seguida será apresentado o precursor do Algoritmo *Firefly*: a Otimização por Enxame de Partículas; na terceira secção o Algoritmo *Firefly* será definido.

3.1 Otimização e Meta-heurísticas

Uma tarefa comum em muitas áreas do conhecimento é, dado um problema cujas soluções possam ser avaliadas objetivamente, encontrar a melhor solução para esse problema. Esse tipo tarefa é chamada de problema de otimização. Em um problema de otimização, é dado um conjunto de variáveis a serem otimizadas, um conjunto de restrições que devem ser mantidas para essas variáveis, e uma função objetivo que quantifique o conjunto de variáveis. A função objetivo irá avaliar o custo, ou a utilidade de determinado conjunto de variáveis.

Matematicamente um problema de otimização possui a seguinte forma:

$$\begin{aligned} \textit{minimizar} \quad & f_0(x) \\ \textit{sujeito a} \quad & f_i(x) \leq b_i, i = 1, ..., m. \end{aligned} \tag{11}$$

Onde o vetor $x \in R^n$ é a variável a ser otimizada no problema; a função $f_0 : R^n \to R$ é a função objetivo; $f_i : R^n \to R$ um conjunto de restrições, e b_i os limites dessas restrições. Um vetor x' que possua o menor valor para a função objetivo será considerado um vetor ótimo, ou a melhor solução para o problema.

Problemas de otimização podem ser classificados com base em diferentes características, sendo algumas delas: o tipo da variável x (real ou discreta); presença ou não de restrições f_i; número de objetivos (mono ou multi-objetivo); convexidade (ou concavidade) ou não da função objetivo; entre outros. Assim como existem diversas

classificações, também existem diferentes técnicas técnicas para resolver os diversos tipos de problemas de otimização (BOYD; VANDENBERGHE, 2006). No geral a maioria das técnicas de otimização pode ser dividida em métodos iterativos e heurísticas. Os métodos iterativos buscam aprimorar a qualidade da solução através de passos determinísticos, exemplos desses são o método *Simplex* e *Hill Climbling*. Já nos métodos baseados em heurísticas não há garantia de convergência.

As meta-heurísticas são normalmente utilizadas quando não há método específico para resolver o problema em questão, ou em casos em que o espaço de busca é grande demais para que outros métodos encontrem boas soluções em tempo hábil. Em problemas classificados como discretos finitos ou combinatoriais, as meta-heurísticas são uma das primeiras alternativas consideradas. Muitos problemas com classificação NP-Difícil são resolvidos através de meta-heurísticas.

Uma meta-heurística é um algoritmo, ou conjunto de regras, para resolver problemas de otimização (DORIGO et al., 2017). As meta-heurísticas são métodos generalizados, sem suposições sobre os problemas que resolve, e por isso podem ser aplicadas em diversos domínios. As meta-heurísticas em sua maioria são métodos iterativos, onde a partir de uma configuração inicial de variáveis, geralmente inicializada de forma aleatória, a cada iteração busca-se aprimorar o valor da função objetivo. Elas podem ser classificadas de acordo com a quantidade de soluções que são buscadas simultaneamente, podendo ser individual, onde apenas uma solução é aprimorada, ou populacional, onde várias são aprimoradas simultaneamente (GENDREAU; POTVIN, 2005).

Com as meta-heurísticas não há garantia que uma boa solução ou a solução ótima será encontrada, pois a busca é feita apenas em regiões do espaço de busca julgadas interessantes pela função objetivo, diferente dos métodos exaustivos onde todas as possibilidades relevantes são avaliadas; portanto as meta-heurísticas são métodos considerados incompletos. A maioria das meta-heurísticas explora o espaço de busca através de movimentações estocásticas. A condição de parada das meta-heurísticas são subjetivas, porém a grande maioria utiliza um número fixo de iterações para sinalizar o término da execução.

3.2 Otimização por Enxame de Partículas

A Otimização por Enxame de Partículas, ou *Particle Swarm Optimization* (PSO) (KENNEDY; EBERHART, 1995), é uma meta-heurística populacional que consiste em explorar o espaço de busca através de um conjunto de indivíduos. Cada indivíduo, geralmente inicializado em uma posição aleatória do espaço de busca, se movimenta no hiperespaço em busca de melhorar a qualidade da sua solução, avaliada pelo valor

função objetivo. O PSO é um método também da classe de Inteligência de Enxames, ou *Swarm Intelligence* (SI) (MILLONAS, 1994), os métodos dessa classe são caracterizados pelas interações e dinâmicas entre os indivíduos, sem nenhum mecanismo de coordenação externo. Os métodos de SI não estão restringidos a problemas de otimização, essa classificação se refere apenas ao mecanismo utilizado, e não ao objetivo, portanto os métodos de SI podem ser utilizados para outros propósitos como simulações.

No PSO cada partícula armazena uma velocidade e e três posições, uma sendo a posição atual e as outras duas são posições consideradas interessantes (KENNEDY; EBERHART, 1997). A velocidade e a posição atual são inicializadas de maneira aleatória, e a cada iteração a posição é atualizada com base na velocidade, que por sua vez é atualizada com base nas posições interessantes. Essa posições de interesse são conhecidas como $pbest$ e $gbest$, onde $pbest$ é a posição da melhor solução obtida pela partícula em questão (*personal best*), e $gbest$ a posição da melhor solução obtida entre todas as partículas (*global best*).

Considere um problema de otimização a ser resolvido com o PSO, o problema com a variável a ser otimizada contendo d dimensões, e o PSO com i partículas. Cada posição (ou solução) é representada por $x_i = \{x_{i1}, x_{i2}, ..., x_{id}\}$, e a velocidade $v_i = \{v_{i1}, v_{i2}, ..., v_{id}\}$. Na versão original do PSO as partículas se movem de acordo com a seguinte expressão (KENNEDY; EBERHART, 1997):

$$\begin{aligned} v_{id} &= v_{id} + \varphi(pbest_{id} - p_{id}) + \varphi(gbest_d - p_{id}), \\ x_{id} &= x_{id} + v_{id}. \end{aligned} \tag{12}$$

Onde φ é um número aleatório positivo, o valor máximo assumido por φ é o primeiro parâmetro configurável do PSO apresentado. Essa expressão representa a mesma movimentação descrita anteriormente, porém com as influências de $pbest$ e $gbest$ reguladas por φ. Versões mais robustas do PSO (SHI; EBERHART, 1998) fazem uso de constantes definidas pelo usuário para regular a influência de cada um dos termos da expressão, essas costantes agem como pesos. A movimentação do PSO com pesos é dada por:

$$v_{id} = wv_{id} + c_1\varphi(pbest_{id} - p_{id}) + c_2\varphi(gbest_d - p_{id}), \tag{13}$$

onde w representa a inércia, e c_1 e c_2 os pesos de cada fator.

Os passos do método PSO serão apresentados em duas etapas, a primeira de inicialização e outra que é executada a cada iteração. A inicialização é feita nos seguintes passos:

1. Inicializa um conjunto de partículas com posições e velocidades aleatórias.
2. Avalia o valor da função objetivo de cada partícula.

3. Para cada partícula é atribuído o valor da função objetivo atual como $pbest$.
4. A posição partícula com melhor valor para função objetivo é atribuída como $gbest$.

Após a inicialização, então a cada iteração é feito para cada partícula:

1. Efetua a movimentação
2. Avalia o valor da função objetivo
3. Caso o valor seja maior que $pbest$, essa nova posição torna-se $pbest$
4. Caso o valor seja maior que $gbest$, essa nova posição torna-se $gbest$

O PSO compartilha algumas similaridades com algoritmos da família de computação evolutiva, porém possui seu funcionamento distinto (EBERHART; SHI, 1998). Similar aos algoritmos genéticos/evolutivos, uma população inicial de indivíduos é gerada afim de se buscar por soluções, porém diferente dos algoritmos genéticos, o enxame de partículas não simula reprodução ou morte; todas as partículas iniciais estarão presentes no fim da execução do algoritmo.

3.3 Definindo o Algoritmo *Firefly*

O Algoritmo *Firefly* é uma meta-heurística similar ao PSO, porém inspirada no comportamento de vaga-lumes (YANG, 2009). O termo inspirado indica que comportamento dos vaga-lumes não é simulado, mas sim utilizado como base para o método. Portanto, algumas características identificadas no comportamento dos vaga-lumes deram origem aos mecanismos do Algoritmo *Firefly*.

O Algoritmo *Firefly* se baseia em duas características específicas dos vaga-lumes, a primeira delas é a atração entre vaga-lumes, que é determinada por alguns fatores como o ritmo e a frequência das emissões. Tal característica é representada como um único valor chamado de brilho, sendo a função objetivo a ser otimizada. A outra característica é a absorção de luz pelo ambiente, que acaba afetando a atração. Portanto ao captar o brilho, a atração dos vaga-lumes deverá variar com a distância. Com essas características são então definidas as três regras básicas do Algoritmo *Firefly*:

1. Todos os vaga-lumes são unissex, para que quaisquer dois vaga-lumes possam se atrair.
2. Vaga-lumes são atraídos por outros de maior brilho.
3. A atração entre vaga-lumes varia com a distância.

Com base nessas regras, os passos do Algoritmo Firefly são apresentados no Algoritmo 1.

Algoritmo 1: Algoritmo Firefly

```
Avalia e atualiza o Brilho de cada vaga-lume
Ordena os vaga-lumes em ordem crescente do brilho
while condição de parada do
    for i = 1 : n em todos n vaga-lumes do
        for j = i : n em todos n vaga-lumes do
            if Brilho(j) > Brilho(i) then
                Calcula a atração
                Move o vaga-lume i em direção ao j com base na atração
                Avalia e atualiza o Brilho de cada vaga-lume
            end
        end
    end
    Ordena os vaga-lumes em ordem crescente
end
```

A avaliação do brilho será dada pela função objetivo escolhida, e irá depender das particularidades do problema. A movimentação e a atração também podem ser definidas pelo problema, porém o autor dá a sugestão de como modelar a movimentação e a atração para problemas no espaço contínuo. A movimentação no Algoritmo Firefly é dada pela seguinte expressão:

$$x_i = x_i + B(x_j - x_i) + \alpha(rand - 0.5). \tag{14}$$

Na qual i e j são vaga-lumes e x é o eixo na qual está sendo feita a movimentação. Portanto $(x_j - x_i)$ é uma movimentação de i em direção a j no eixo x. Essa movimentação é regulada pela atração B. Já $\alpha(rand - 0.5)$ é uma movimentação aleatória, na qual $rand$ é um gerador de números pseudo aleatórios entre $[0, 1]$, fazendo com que $(rand - 0.5)$ seja uma movimentação aleatória no eixo x entre $[-0.5, 0.5]$. Essa por sua vez é regulada pelo parâmetro α.

A atração B deve variar com a distância, para isso o autor do Algoritmo Firefly propõe utilizar o modelo de decaimento exponencial dado por:

$$B = B_0 e^{\gamma r^2}, \tag{15}$$

na qual r é a distância, e B_0 é a atração quando a distância é 0, para grande parte dos problemas esse valor pode ser considerado $B_0 = 1$. Tornando assim a expressão final da movimentação:

$$x_i = x_i + B_0 e^{\gamma r^2}(x_j - x_i) + \alpha(rand - 0.5). \tag{16}$$

O controle do balanço entre intensificação e exploração se dá pelos parâmetros α e γ. Quanto maior α mais movimentações aleatórias serão feitas. Quanto menor γ

maior será a visibilidade dos vaga-lumes, e consequentemente mais forte será a atração. Estes dois parâmetros controlam a convergência, se o balanço não estiver correto o algoritmo não irá convergir para algum bom resultado. Caso haja muita aleatoriedade, caracterizada por valores α e/ou γ elevados o algoritmo não irá convergir pois as soluções encontradas serão aleatórias. Já no caso oposto, caso γ e/ou α sejam muito pequenos, o algoritmo irá convergir abruptamente para alguma solução sub-ótima.

Uma dinâmica interessante do Algoritmo *Firefly* é a subdivisão do enxame de partículas inicial em múltiplos enxames, permitindo que o Algoritmo *Firefly* realize otimização multimodal (YANG; HE, 2018). O conceito de multimodalidade se refere a problemas caracterizados por múltiplos pontos ótimos locais e/ou globais julgados interessantes. Para tais problemas, um bom resultado não é apenas uma solução global ótima, mas sim múltiplas soluções ótimas, locais e globais (YU; GEN, 2010). Uma maneira simples de alcançar otimização multimodal é aplicar algum método de otimização local, como a busca local por exemplo, em regiões diferentes do espaço de busca.

A capacidade de otimização multimodal com o Algoritmo *Firefly* se dá através do mecanismo de atração, onde expressão da atração dada pelo decaimento exponencial torna a atração à curta distância maior que atração a longa distância (YANG; HE, 2018). Como as partículas são inicializadas aleatoriamente, o enxame inicial não estará convergindo para algum ponto em particular, portanto diferentes vaga-lumes estarão destacados em diferentes regiões do espaço de busca; consequentemente os demais vaga-lumes irão se mover em direção aos vaga-lumes de destaque mais próximos. Essa dinâmica faz com que múltiplos sub-enxames se formem em diferentes regiões do espaço de busca, no qual cada sub-enxame irá efetuar uma otimização local, caracterizando assim a capacidade de realizar otimização multimodal do Algoritmo *Firefly* através de múltiplas otimizações locais.

4 Modelando o *Biclustering* com o Algoritmo *Firefly*

Com base nos conceitos apresentados até então, neste capítulo será apresentada a modelagem do *biclustering* com o Algoritmo Firefly. O primeiro passo nessa modelagem é definir como *bicluster* será representado computacionalmente, para isso foi adotado o modelo de *string* binária, comumente utilizado na literatura (SERIDI et al., 2015), (GENDREAU; POTVIN, 2005), (DIVINA; AGUILAR-RUIZ, 2006). Computacionalmente o termo *string* denota uma estrutura de dados que armazena uma sequência de caracteres, normalmente utilizada para representar dados textuais. Uma *string* é uma sequência de *bytes*, onde normalmente cada *byte* representa um caractere (para linguagens ocidentais, para orientais outra codificação é utilizada).

Uma *string* binária possui a mesma estrutura de uma *string* tradicional, o que muda é o seu propósito, para armazenar dados não textuais porém ainda sequênciais como imagens. Para entender o propósito da *string* binária, considere os algoritmos genéticos ou evolutivos, onde se considera que cada indivíduo (solução) possui um genótipo e um fenótipo; o fenótipo é a solução em si, enquanto que o genótipo é a representação da solução. Ao se utilizar a *string* binária para representar o genótipo, operadores como *crossover* e mutação podem ser implementados de acordo com os processos naturais em que se baseiam (GENDREAU; POTVIN, 2005).

Suponha um *bicluster* B formado pelo conjunto de linhas L e conjunto de colunas C, para representar B utilizando uma *string* binária usa-se uma *string* tamanho $|L| + |C|$, onde os $|L|$ primeiro elementos serão referentes aos índices linhas e os $|C|$ elementos seguintes serão os índices das colunas, de forma que, cada elemento dessa *string* possa ter valor 0 ou 1 como falso ou verdadeira respectivamente, que representam a presença ou não do elemento no bicluster. Essa representação segue a mesma analogia de genótipo de fenótipo dos algoritmos genéticos, nesse caso o genótipo também seria uma *string* binária indicando quais linhas e colunas fazem parte do bicluster, e o fenótipo será a submatriz coletada com base nos elementos indicados na *string* binária. Essa escolha de representação possui alguns benefícios computacionais devido as submatrizes em si não serem armazenadas mas apenas os índices dos seus elementos, isso facilita a manipulação de biclusters para buscar soluções, uma vez que adicionar ou remover alguma linha ou coluna é feito apenas trocando um único valor, e os custos computacionais de armazenamento e de manipulação de memória da solução são limitados apenas para *strings* binárias.

Por se um problema NP-Difícil, o *biclustering* pode ser visto como um problema de busca (otimização) cujo objetivo é encontrar *biclusters*. Todo processo de busca se supõe um ponto de partida, como neste trabalho trata-se de utilizar o Algoritmo *Firefly*,

um método de enxame de partículas, as partículas serão inicializadas em posições aleatórias. Existem outras formas válidas de se inicializar a busca, porém esta é a mais comum, pois atesta que a natureza estocástica do método ser capaz de encontrar boas soluções independente do estado inicial da busca, e também permite variações no processo de busca.

Portanto, para inicializar o enxame, cada partícula irá conter linhas e colunas aleatórias, o que significa preencher cada posição da *string* binária com valor 0 ou 1 aleatoriamente. Vários biclusters poderão estar presentes no espaço de busca, e à depender de onde se inicializa a busca ela irá convergir para um determinado bicluster ao invés de outro. Vale notar que essa abordagem de inicialização não é estritamente requerida, outras maneiras de fazer inicialização também são válidas, como por exemplo, inicializar biclusters com probabilidades diferentes entre 0 e 1, ou até utilizar resultados de outros métodos como soluções iniciais.

Para efetuar a busca incremental por biclusters, as operações mecânicas de adição ou remoção de elementos já estão definidas pela representação da *string* binária, onde para incluir um elemento na solução basta atribuído o valor 1, ou 0 caso contrário. Resta agora definir o mecanismo de tomada de decisões que irá movimentar as partículas no processo de busca, e esse será o papel Algoritmo *Firefly*. Porém o Algoritmo *Firefly* foi definido para o espaço contínuo, e portanto deverá ser adaptado para o problema atual.

As subsecções deste capítulo foram dedicadas a descrever como foram modelados os diferentes aspectos do Algoritmo *Firefly* para efetuar o *biclustering*. Essas secções foram decididas com base na análise dos diferentes mecanismos presentes na expressão de movimentação do Algoritmo *Firefly*.

Considere dois vaga-lumes i e j, uma possível movimentação de i em direção a j é dada por:

$$x_i = x_i + B_0 e^{\gamma r^2}(x_j - x_i) + \alpha(rand - 0.5). \tag{17}$$

Onde x é uma dimensão do vetor no qual a movimentação está acontecendo, e o primeiro termo da equação: x_i indica que a movimentação será relativa à posição atual do vaga-lume i. Essa movimentação só irá ocorrer caso o vaga-lume j possua brilho maior que o vaga-lume i, portanto a primeira subsecção foi dedicada à definição do brilho. O segundo termo da equação: $B_0 e^{\gamma r^2}(x_j - x_i)$ representa o deslocamento de x_i em direção a x_j com base na atração $B_0 e^{\gamma r^2}$, esta atração será definida em detalhes na subsecção subsequente à do brilho. O último termo se refere a um deslocamento aleatório, onde $rand$ é gerador de numeros pseudo aleatórios entre 0 e 1 e α é um termo para regular a intensidade de tal movimentação aleatória. Por fim na última subsecção desse capítulo será definido como todos os deslocamentos anteriores irão ocorrer no espaço de busca definido, caracterizando assim a movimentação.

4.1 Brilho

O brilho, ou a função objetivo, é o mecanismo que guiará a busca por melhores soluções. O papel da função objetivo é avaliar de forma objetiva e quantificável a qualidade de uma solução, de forma que outros mecanismos da meta-heurística irão se valer desta objetivação para tomar decisões em prol da busca. No caso do Algoritmo Firefly, os vaga-lumes são atraídas pelo brilho, fazendo com que vaga-lumes de menor brilho busquem se aproximar de vaga-lumes de maior brilho, na esperança de melhorar sua própria solução. A escolha da função objetivo é uma peça chave em muitas problemas de otimização, pois ela irá caracterizar quais soluções serão as ideais, e que consequentemente serão buscadas. Em alguns domínios onde existe apenas uma maneira de se avaliar tal qualidade, a escolha desta é clara, porém em outros como no caso deste trabalho, existem várias formas de se avaliar um bicluster, e nenhuma delas é claramente superior.

Na literatura, uma heurística aceita para a de buscar *biclusters* com alta coerência e de maior volume possível (PONTES et al., 2015a), (LIU et al., 2009), porém quanto maior um *bicluster* mais provável é que sua coerência seja baixa e vice-versa, portanto esses objetivos são conflitantes. Para então otimizar objetivos conflitates há o conceito de otimização multi-objetivo (DEB et al., 2002), (DEB, 2012), onde os múltiplos objetivos são otimizados individualmente. Porém boa parte dos algoritmos está focada em encontrar uma fronteira de Pareto ótima, fazendo efetivamente uma otimização global, porém para o biclustering é interessante que seja feita a otimização local (COELHO et al., 2009).

Uma forma de representar esses objetivos conflitantes é condensar os objetivos em uma única expressão que irá produzir um único valor a ser otimizado, podendo nessa expressão serem atribuídos pesos diferentes aos diferentes objetivos. A função objetivo utilizada neste trabalho foi inspirada no trabalho de Divina e Aguilar-Ruiz (2006). Nesse trabalho a função objetiva utilizada pelo algoritmo genético proposto é uma expressão que combina multiplos objetivos em um único valor, sendo os objetivos: o volume, coerência, variância entre as linhas e uma penalidade para *biclusters já encontrados*. A função objetivo citada é dada por:

$$\frac{residue(X)}{\delta} + \frac{1}{row\_variance(x)} + wd + penalty. \tag{18}$$

Cada um dos termos dessa equação representa alguma característica desejável para o bicluster ideal, definindo assim o objetivo a ser alcançado. O volume e a coerência já foram abordados, são considerados essenciais para *biclustering*, porém a variância das linhas e a penalidade serão apresentados. No trabalho citado e outros a variância entre as linhas é utilizada para buscar biclusters relevantes no contexto de expressão gênica (PONTES et al., 2015a), com este objetivo busca-se maximizar a variância entre as linhas, isto se remeta ao fato biológico de que os genes trabalham em conjunto, caso a variância

seja baixa, isto indica que não há alteração entre os valores, tratando-se de um bicluster constante, e no contexto desses trabalhos tais biclusters não são importantes. Portanto a variância entre linhas é um fator importante de qualidade para esses trabalhos.

O método citado foi projetado para ser executado múltiplas vezes, onde ao final de cada execução é retornado um *bicluster* caso encontrado, ou nenhum caso contrário. Porém ao se executar o mesmo algoritmo várias vezes, não há garantia de que novos *biclusters* serão encontrados, pode ser retornado o mesmo bicluster várias vezes. Com isso Divina e Aguilar-Ruiz (2006) apresenta a penalidade como uma solução. Essa penalidade consiste em atribuir pesos negativos para elementos que já foram retornados como resultado anterior, isso fará com que em execuções subsequentes o algoritmo favoreça outros *biclusters*.

O resíduo na métrica proposta por Divina e Aguilar-Ruiz (2006) é referente ao MSR, consequentemente o fator δ é baseado no mesmo. Já o volume nesta função objetivo é tratado de uma forma especial. Ao invés de se calcular o volume de forma simples, dado por exemplo por $Volume = |L| * |C|$, pesos diferentes são atribuídos para as linhas e para as colunas, isto pois pode ocorrer maior tendência em crescer linhas ou colunas devido ao tamanho original da matriz e a natureza dos dados. Em dados de expressão por exemplo, geralmente a quantidade de genes é bem maior que a quantidade de condições, possuindo então muito mais linhas do que colunas, neste caso a tendência será para acrescentar o número de linhas, visto que provavelmente adicionar uma nova linha irá conter mais elementos do que adicionar uma nova coluna. Levando isto em conta, a métrica utilizada possibilita pesos diferentes para linhas e colunas, permitindo que o usuário encontra e defina manualmente os valores adequados. Com base nas colocações anteriores, o cálculo do volume em (DIVINA; AGUILAR-RUIZ, 2006) é dado por: $w_v(w_r\frac{\delta}{row_x} + w_c\frac{\delta}{col_x})$. Vale notar que se deseja que o volume dos *biclusters* seja o maior possível e a heurística está sendo apresentada como um problema de minimização, portanto quanto maior o volume, menor deverá ser o seu número no score final, com base nisso o número de linhas e colunas foram atribuídos como denominadores em função de δ, dessa quanto maior for o número de linhas ou colunas, seu respectivo fator será menor (melhor).

Por fim, a função objetivo utilizada como brilho neste trabalho é dada por:

$$\frac{MSR(X)}{\delta} + w_v(w_l\frac{\delta}{|L|} + w_c\frac{\delta}{|C|}) \tag{19}$$

4.2 Atração

A atração é o mecanismo de intensificação do Algoritmo Firefly. A cada iteração, dado vaga-lume irá se mover em direção a outro com maior brilho, indicando que tal vaga-lume irá incorporar elementos do qual está atraído a fim de melhorar sua própria

solução. Tal movimentação pode resultar em uma melhoria ou não da qualidade da solução, porém qualquer um dos casos será aceito igualmente. Caso tal movimentação resulte em melhoria da solução, essa terá sua função objetivo elevada, e pode vir a subir no rank da população. Quanto maior for a colocação de um vaga-lume no rank menos movimentações ele irá fazer, fazendo com que os melhores indivíduos preservem suas características, enquanto que os demais explorem mais alternativas à fim de melhorarem seu próprio brilho.

Apesar de todos os vaga-lumes serem atraídos pelo menos pelo melhor vaga-lume, nem todos irão convergir para a sua posição, isso porque a atração também se baseia na distância. Cada vaga-lume será atraído por todos os outros cujo brilho seja maior, porém a intensidade dessa atração deverá variar com a distância. Tal relativização da atração foi adotada pelo autor inspirado no fenômeno de absorção da luz pelo ambiente, no qual quanto menor o coeficiente de absorção mais o brilho de um vaga-lume chegará a outro, e consequentemente maior será a atração.

A intensidade da luz será diferente à depender da distância dá fonte, quanto maior a distância menor será a intensidade da luz, isso é chamado de lei do quadrado inverso, dada matematicamente por:

$$I = \frac{I_f}{r^2}, \tag{20}$$

onde r é a distância e I_f é a intensidade na fonte. Caso seja considerado que a luz irá permear algum meio com coeficiente de absorção γ, a variação de intensidade da luz segue o decaimento exponencial:

$$I = I_f e^{-\gamma r}. \tag{21}$$

Yang (2009) combina as duas equações anteriores em:

$$I = I_f e^{-\gamma r^2}, \tag{22}$$

e utiliza esse modelo de decaimento exponencial da intensidade da luz como mecanismo de atração, esse mecanismo é regulado pelo parâmetro γ. Vale notar que na descrição original do Algoritmo Firefly, o coeficiente γ faz referencia ao coeficiente de absorção da luz, e como brilho decai com a distância, porém esta parte não tem relação com a função objetivo. No Algoritmo em si, não há descrições de que o brilho varia, muito pelo contrário, a comparação do brilho entre os vaga-lumes é feita diretamente com o valor da função objetivo, porém a atração varia sim. A metáfora do coeficiente de absorção de luz aplica-se somente à atração, e não ao valor da função objetivo. A ideia na forma que foi escrita originalmente pode levar a várias interpretações, porém a que é utilizada neste trabalho é apresentada no pseudocódigo do algoritmo. Esclarecendo, dado vaga-lume será definitivamente atraído por outros de maior brilho, independente

da distância, porém a intensidade dessa atração irá variar com a distância de acordo com o parâmetro γ. Portanto a atração foi desvinculada do brilho, é dada por:

$$B_0 e^{\gamma r^2}, \tag{23}$$

onde B_0 é a atração quando a distância é 0, e para esse valor assumiu-se que $B_0 = 1$.

Note que foi citado recorrentemente o conceito de distância, porém ele ainda não está definido propriamente para o contexto de biclustering. Na definição original do Algoritmo Firefly a distância utilizada é a Euclidiana, porém para esta aplicação ela não faz sentido, pois a distância Euclidiana representa o comprimento do segmento de linha que conecta duas coordenadas, porém não estamos trabalhando com coordenadas e sim com uma string *binária*.

Com isso a distancia de *Hamming* pode ser adotada como medida (BOOKSTEIN et al., 2002). Essa medida compara duas *strings* de mesmo tamanho, e corresponde ao número de posições em que ambas são diferentes. Para ilustrar, nos casos das *strings* binárias $"000"$ e $"111"$ a distância *Hamming* será 3; já entre $"0101"$ e $"0100"$ a distância será 1. A intenção é utilizar a distância de *Hamming* para calcular a distância entre as *strings* binárias dos biclusters. Porém a distãncia de *Hamming* máxima irá depender to tamanho da matriz de dados, e consequentemente irá afetar o coeficiente γ. Na tentativa de eliminar, ou ao menos reduzir tal dependência, a distância r será normalizada de acordo com o tamanho da *string* binária, de maneira que $0 \geq r \leq 1$.

Com isso o mecanismo de atração está definido, resta agora caracterizar como serão realizadas as movimentações.

4.3 Movimentação

A movimentação no Algoritmo Firefly acontece em dois passos: primeiro em direção à atração e outro passo aleatório. Esta é a heurística principal do Algoritmo Firefly, onde a intensificação é dada pela movimentação em direção à vaga-lumes mais atraentes, com a intensão de melhorar a própria qualidade, e a exploração é dada pelo passo aleatório. Vale ressaltar que também há um pouco de exploração durante a intensificação, visto que a adoção de elementos da melhor solução é dada por probabilidade, ou seja, não se sabe quais elementos serão adicionados, e não há garantia de que tal movimentação irá melhorar a solução. Como boa parte das meta-heurísticas, o Algoritmo Firefly aceita iterações que não melhore a solução como mecanismo de explorar adequadamente o espaço de busca e evitar mínimos locais.

Tal movimentação irá ocorrer somente caso o vaga-lume j possua brilho I maior

que i, caso contrário nenhuma movimentação será feita, portanto:

$$x_i = \begin{cases} x_i + B_0 e^{\gamma r^2}(x_j - x_i) + \alpha(rand - 0.5), & \text{se } I_j > I_i \\ x_i, & \text{senão.} \end{cases} \tag{24}$$

Note que essa movimentação é uma expressão no espaço contínuo, que visa movimentar dimensões em um vetor, porém não estamos trabalho com vetores, mas sim com a *string* binária, portanto a operação em si deverá ser modelada de acordo com as nossas necessidades.

Num espaço binário, tal movimentação pode ser vista como a alteração dos *bits* da solução, onde um indivíduo se mantém parado caso nenhum dos seus *bits* sejam alterados, ou se moveu em velocidade máxima caso todos os seus *bits* sejam invertidos. Tal característica pode ser obtida ao se representar a movimentação em termos probabilísticos, Tilahun e Ngnotchouye (2017) classifica a estratégias com isso em duas formas, a primeira ele chama de atualizar no espaço contínuo, a segunda ele nomeia de atualizar no espaço discreto.

Na estratégia de atualizar no espaço contínuo, o Algoritmo Firefly serial executado de maneira tradicional no espaço contínuo, porém a posição final é limitada para um valor probabilístico entre 0 e 1, onde geralmente é utilizado a função sigmoidal:

$$\frac{1}{1 + e^{-x}}. \tag{25}$$

Um dos pioneiros nesta técnica foi Kennedy e Eberhart (1997), qual descreve uma versão binária do algoritmo PSO. Nesse trabalho o autor define velocidade e trajetórias como a probabilidade de que um *bit* assuma um valor ou outro, onde por exemplo, uma velocidade $v_i = 0.20$ indica uma probabilidade de 20% que $x_i = 1$ e de 80% que $x_i = 0$. Tal técnica foi testada em funções do espaço contínuo, porém com as variáveis de ponto flutuante sendo representadas de forma binária. A técnica proposta mostrou-se válida em todos os casos teste.

Já com a estratégia de atualizar no espaço discreto, a própria expressão de movimentação é convertida para funcionar de maneira probabilística. Essa estratégia é pouco explorada, porém vantajosa. Durkota (2011) propõe uma versão discreta do Algoritmo Firefly para resolver o Problema de Alocação Quadrática, ou *Quadratic Assignment Problem* (QAP), e adapta todos os mecanismos do Algoritmo Firefly para o problema em questão, mantendo os princípios. A abordagem tomada neste trabalho foi inspirada na abordagem do trabalho citado, porém adequada para *biclustering*.

Adotando a estratégia de atualizar no espaço discreto então, a movimentação deve ser feita através de probabilidades. A cada iteração a probabilidade em relação a B deve ser calculada, com isso em mente, suponha um valor p tal que $0 \geq p \leq 1$, gerado

aleatoriamente, a movimentação então pode ser representada como:

$$x_i = \begin{cases} x_j, & p < B \\ x_i, & p >= B \end{cases} \tag{26}$$

Onde x representa uma posição na string binária.

O passo aleatório, pode ser modelado de forma mais direta. Na expressão original ele é representado como $\alpha(rand - 0.5)$ para representar que a cada iteração será feito uma movimentação em uma direção aleatória. Com uma variável binária há somente duas movimentações possíveis, portanto esse mecanismo pode ser implementado como probabilidade, de x_i ter seu valor ou 0 ou 1, de forma aleatória. Essa probabilidade pode ser ditada diretamente pelo parâmetro α, de forma que, um $\alpha = 0.1$ por exemplo, indica 10% de chance de se realizar um passo aleatória. Portanto, no passo aleatório:

$$x_i = \begin{cases} x_\alpha, & p < \alpha \\ x_i, & p >= \alpha. \end{cases} \tag{27}$$

Onde x_α é 0 ou 1 escolhido aleatoriamente a cada avaliação.

Com isto é finalizada a modelagem do Algoritmo Firefly do espaço contínuo para um espaço discreto combinatorial. Ao final tem-se que a movimentação resulta na avaliação da probabilidade de dois eventos, um para o passo em direção a atração, e outro referente ao passo aleatório.

5 Materiais e Métodos

Devido à críticas relevantes referentes ao estado atual da área de estudo de meta-heurísticas (CHUANG et al., 2008), (WEYLAND, 2015), o método proposto será avaliado de acordo com seus próprios méritos, sem comparação direta com outros métodos já publicados. Uma dessas críticas é a proliferação de trabalhos que sempre são apresentados como superiores aos anteriores, o que chamam de *One-upmanship-game*, onde centenas de meta-heurísticas 'originais' propostas sempre são avaliadas como superiores.

Uma abordagem de design de experimentos tradicional não foi abordada neste trabalho principalmente por dois motivos. O primeiro motivo é a falta de consenso nos métodos de avaliação de resultados de *biclustering*, e o segundo motivo é que o método proposto possui muitos parâmetros a serem configurados.

O método proposto contém um total de 8 parâmetros, 4 referentes ao Algoritmo Firefly: número de iterações, número de soluções iniciais, γ, α; e outras 4 da função objetivo: δ, w_v, w_r e w_c. Esses parâmetros são valores inteiros ou reais, e não há indicação na literatura de quais valores devam assumir idealmente, além de que a parametrização irá depender do conjunto de dados a ser analisado.

Para evitar os problemas citados, foi escolhida então uma estratégia exploratória para avaliar o método proposto. Essa estratégia consiste em avaliar o algoritmo em dados sintéticos progressivamente mais complexos, a fim de identificar em quais casos o método seria adequado. A utilização dos dados sintéticos será descrita em detalhes na próxima subsecção.

A configuração de parâmetros foi feita com base em observações do pesquisador sobre a convergência do método, onde caso uma configuração não apresente convergência, o pesquisador observa o comportamento do método e propõe uma nova configuração de parâmetros com o objetivo de superar o resultado anterior.

A linguagem de programação R foi escolhida como ambiente computacional. de acordo com alguns critérios. O primeiro foi a disponibilidade de pacotes atualizados para análise de dados de expressão gênica. Outro critério foi a facilidade de manipulação das estruturas de dados utilizadas, como a *string* binária. Por fim o último critério foi a disponibilidade de um ambiente interativo robusto, que permita fazer testes e ajustes rápidos no código.

5.1 Validação dos resultados

O *biclustering* pode ser aplicado e analisado de diversas formas, a depender da aplicação, e ainda não há consenso sobre as formas de se validar a qualidade de resultados de *biclustering*, e mais pesquisa deve ser feita nessa área (PONTES et al., 2015a). A maneira que vêm sendo feita para dados de expressão gênica é enriquecer os *biclusters* encontrados com dados disponíveis em bases de dados de anotação gênica, como o *Gene Ontology* (GO) (PONTES et al., 2015a). Tal validação consiste em coletar todos os termos anotados na ontologia referentes a cada um dos genes no *bicluster* e aplicar o teste de significância estatística para verificar a relevância de tais termos (PONTES et al., 2015a).

Porém a validação com base na ontologia está sujeita a falhas, pois o conhecimento biológico em anotações ainda não é completo, portanto *biclusters* que possuam genes não anotados terão sua qualidade prejudicada pela presença desses genes. A presença de termos não anotados pode ajudar na inferência na função de determinado gene não anotado, e portanto outros estudos devem ser feitos para averiguar tal inferência. Outra desvantagem é que a ontologia classifica genes hierarquicamente, com funções mais genéricas no topo e as mais específicas nos níveis mais baixo, isso cria um viés em favor de *biclusters* maiores, pois um maior número de termos anotados estará presente (PONTES et al., 2015a).

Uma alternativa para validação é o uso de dados sintéticos (PONTES et al., 2015a). Buscando simular matrizes de dados de expressão, esse dados sintéticos são matrizes geradas com dados artificiais, contendo biclusters conhecidos *à priori*. Esses dados podem ser utilizados para avaliar diferentes aspectos que afetam o *biclustering*, como o padrão dos *biclusters*, a intensidade de ruído ou *biclusters* com sobreposição (EREN et al., 2013). A grande vantagem de utilizar dados desse tipo é por justamente já se conhecer o resultado, tornando a validação bem mais fácil, porém os dados são artificiais, e podem não ser condizentes com as aplicações reais.

Fazendo uso de dados sintéticos então é possível avaliar os resultados por meio das métricas de recuperação e relevância (PRELIĆ et al., 2006), (EREN et al., 2013). Essas são operações entre dois conjuntos, os conjuntos serão *biclusters*, onde cada *bicluster* é um conjunto de linhas e colunas que o compõe. Para calcular tais métricas, primeiro é preciso definir uma outra operação que irá verificar o grau de similaridade entre conjuntos, no qual conjuntos similares apresentem valores maiores que conjuntos dissimilares. Para isso foi escolhido o coeficiente de *Jaccard* aplicado aos biclusters. Sejam dois *biclusters* b_1 e b_2, o coeficiente de *Jaccard* J é dado por:

$$J(b_1, b_2) = \frac{|b_1 \cap b_2|}{|b_1 \cup b_2|}, \tag{28}$$

no qual $|b_1 \cap b_2|$ é a quantidade de elementos na intersecção, e $|b_1 \cup b_2|$ a quantidade de

elementos na união, e o valor resultante pode ser interpretado como a porcentagem de elementos compartilhados entre ambos biclusters (EREN et al., 2013). Dessa forma caso b_1 e b_2 sejam idênticos $J(b_1, b_2) = 1$, e caso sejam disjuntos $J(b_1, b_2) = 0$.

Considere a expressão S como:

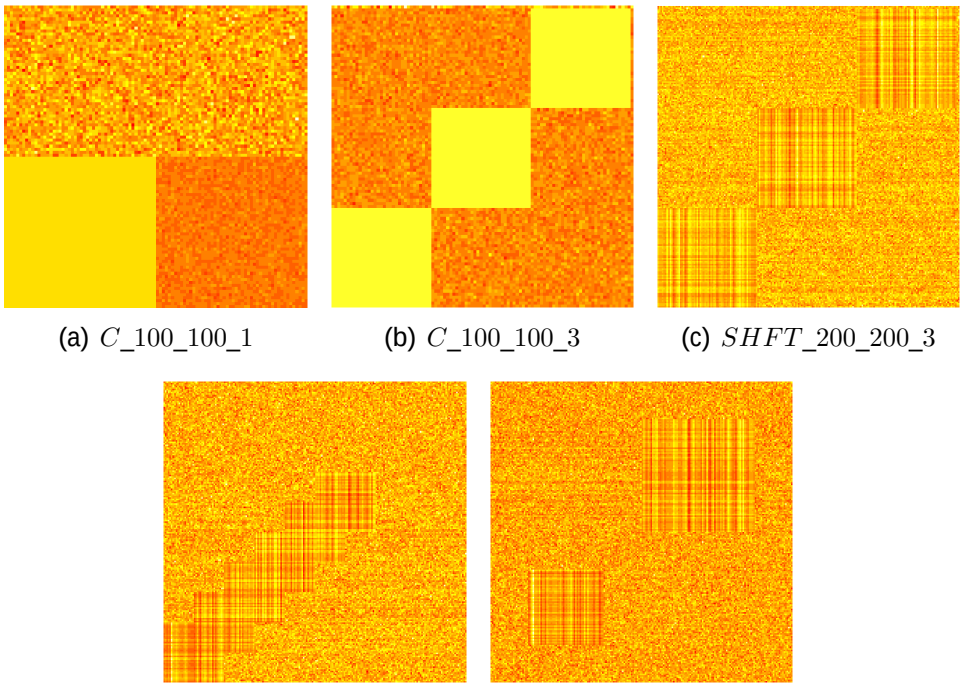
$$S(M_1, M_2) = \frac{1}{|M_1|} \sum_{b_1 \in M_1} \max_{b_2 \in M_2} \frac{|b_1 \cap b_2|}{|b_1 \cup b_2|}. \tag{29}$$

Seja V o conjunto de biclusters embutidos, 'verdadeiros', e E o conjunto de biclusters encontrados pelo método. A recuperação é dada por $S(V, E)$, que significa a fração dos documentos encontrados que são verdadeiros, já a relevância é dada por $S(E, V)$, significando fração dos documentos verdadeiros que foram encontrados.

Foram explorados 5 matrizes de dados sintéticas no total, ilustradas na Figura 5.1. O primeiro conjunto de dados é o teste mais simples, uma matriz quadrada de ordem 100, contendo um *bicluster* de padrão constante, rotulado $C\_100\_100\_1$ (Figura 4(a)). O segundo conjunto de dados rotulado $C\_100\_100\_3$, também é uma matriz quadrada de ordem 100 porém contendo 3 *biclusters* embutidos (Figura 4(b)). A terceira matriz também é quadrada, porém de tamanho 200×200 e contendo 3 *biclusters* seguindo um padrão de deslocamento, portanto sendo rotulada como $SHFT\_200\_200\_3$ (Figura 4(c)). O quarto conjunto de dados consiste de outra matriz quadrada de ordem 200, porém com 6 *biclusters* com sobreposição, rotulada $SHFT\_200\_200\_6ov$ (Figura 4(d)). Por fim o último conjunto de dados gerado também de tamanho 200×200, porém contendo apenas 2 *biclusters*, um de tamanho 50×50 e outro com tamanho 75×75; rotulado $SHFT\_200\_200\_2$ (Figura 4(e)).

Os dados sintéticos foram geradas com a ferramenta BiBench (*Biclustering Benchmark*) (EREN, 2012), (EREN et al., 2013). Essa biblioteca possui diversas funcionalidades para a avaliação de algoritmos de *biclustering*, permitindo a geração de dados sintéticos, a avaliação por recuperação e relevância, e caso queira o enriquecimento com dados biológicos, entre diversas outras funcionalidades relacionadas. Para este trabalho foram utilizadas apenas as funcionalidades para gerar e avaliar dados sintéticos.

Figura 3 – Mapa de calor que ilustram os dos dados sintéticos gerados



(a) C_100_100_1 (b) C_100_100_3 (c) $SHFT$_200_200_3

(d) $SHFT$_200_200_6ov (e) $SHFT$_200_200_2

6 Resultados e discussões

Dando início a estratégia de explorar o desempenho algoritmo em dados sintéticos progressivamente mais difíceis, o primeiro conjunto de dados utilizado foi uma matriz quadrada de ordem 100 contendo apenas um *bicluster* do tipo constante de tamanho 50×50. Para esse caso os parâmetros foram escolhidos empiricamente, tendo $n = 10$ indicando que foram geradas 10 soluções aleatórias, junto com os parâmetros do Algoritmo *Firefly* $\gamma = 1$ e $\alpha = 0.01$, caracterizando assim a configuração A. Os resultados da execução do método com esses parametros estão apresentados na Figura 4. Já na primeira configuração o algoritmo indicou convergência para a função objetivo. Para conferir a validade do resultado, o conjunto de biclusters encontrado então foi submetido ao teste de recuperação e relevância, os resultados estão apresentados na Tabela 1, e indica uma recuperação perfeita, com o valor máximo de recuperação e relevância. Vale notar que no conjunto de dados há apenas um bicluster, porém o algoritmo inicializado com $n = 10$ irá retonar dez soluções, e devido ao mecanismo de aproximação entre partículas espera-se que muitas soluções encontradas sejam redudantes. O cálculo da recuperação e relevância não é prejudicado por redundâncias.

Figura 4 – Gráfico de convergência da execução do método na matriz de dados sintéticos de tamanho 100×100, contendo 1 *bicluster* de tamanho 50×50 com valores constantes.

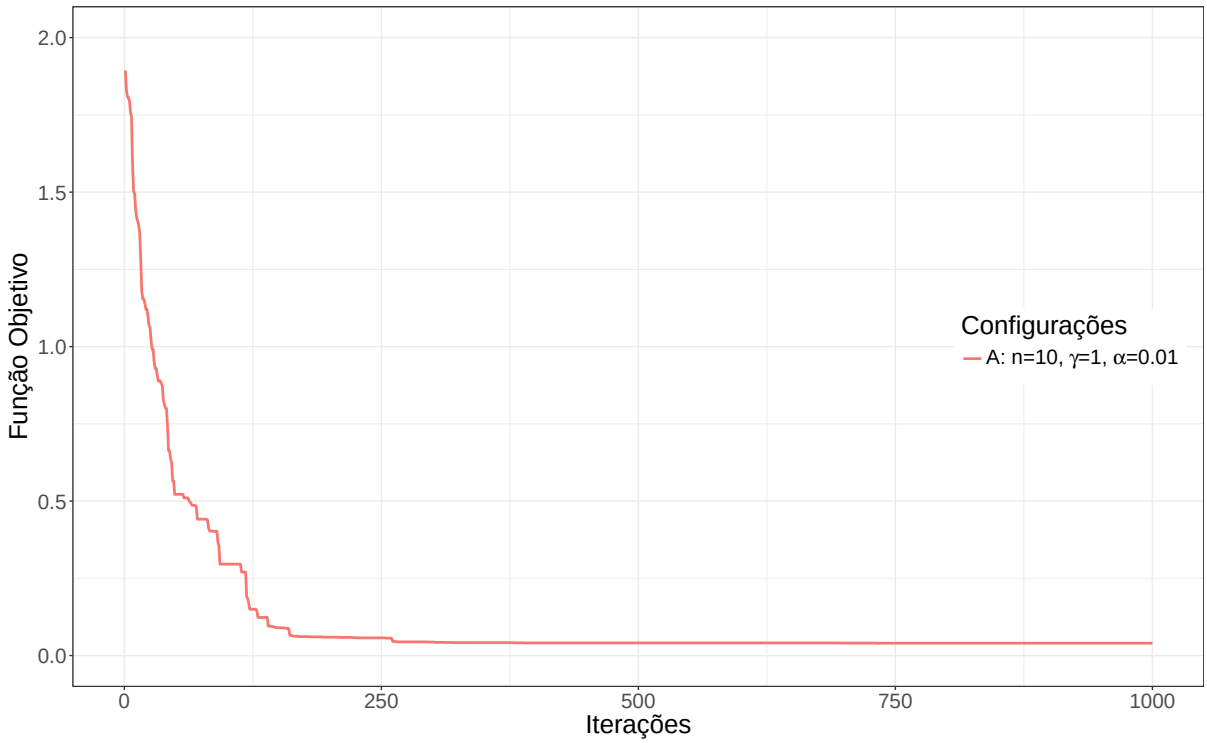
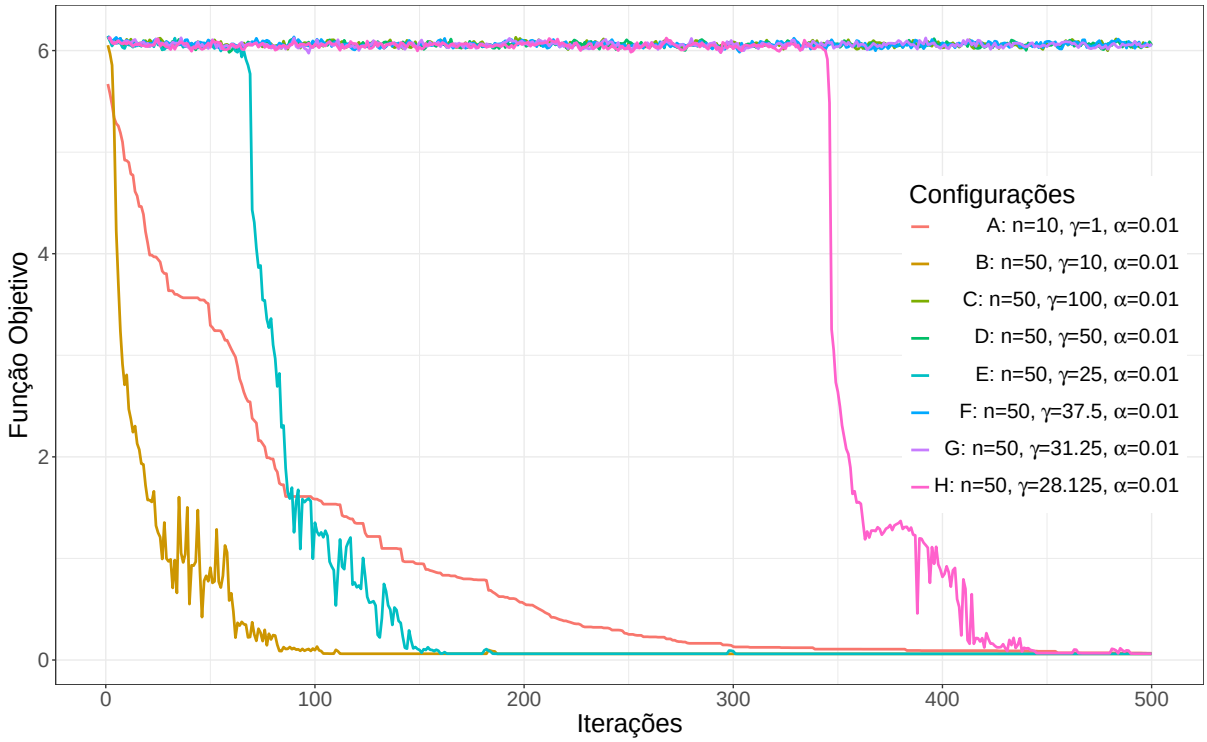


Tabela 1 – Valores de Recuperação e Relevância obtidos ao se executar o método na matriz sintética de tamanho 100×100, com 1 *bicluster* embutido de tamanho 50×50 com valores constante.

Configuração	Recuperação	Relevância
A	1	1

No próximo caso de teste buscou-se avaliar a capacidade do algoritmo para encontrar múltiplos *biclusters*. Foi então gerada uma nova matriz quadrada também de ordem 100, porém com 3 *biclusters* embutidos. Como o teste anterior apresentou resultados válidos, a mesma configuração foi mantida como ponto de partida. A Figura 5 apresenta o gráfico de convergência contendo todas as configurações avaliadas. Para este caso a configuração A também convergiu, porém ao submeter os *biclusters* encontrados para validação, foram obtidos 30% de recuperação e 90% de relevância. A baixa recuperação indica que boa parte dos biclusters não foram encontrados, porém a alta relevância indica que o que foi encontrado estava próximo dos verdadeiros, essas frações indicam que aproximadamente um dos biclusters foi encontrado.

Figura 5 – Gráfico de convergência da execução do método na matriz de dados sintéticos de tamanho 100×100, com 3 *biclusters* de tamanho 33×33 com valores constante.



Na tentativa de identificar qual o motivo de tal resultado, foi verifica a aproximação entre partículas. Foram calculados os coeficientes de *Jaccard* entre todas as partículas par a par, resultando em uma matriz de ordem da quantidade de partículas, onde

cada célula representa a fração de sobreposição entre duas partículas. Naturalmente a diagonal principal contém apenas valores 1, porem as demais apresentaram somente valores entre $0,97$ e 1, indicando uma extrema taxa de sobreposição entre as partículas. Portanto foi constatado que todas as partículas convergiram para um mesmo *bicluster*, ignorando os demais.

Para então tentar favorecer a busca por mais *biclusters*, foi definida uma nova configuração B com o objetivo de aumentar a capacidade de exploração através dos parâmetros $\gamma = 100$ e $n = 50$. O aumento no número de vaga-lumes n irá fazer com que mais soluções iniciais sejam geradas e exploradas simultaneamente, enquanto que γ irá afetar o raio de efeito dos vaga-lumes. Com o resultado da convergência também apresentado na Figura 5, apresenta que essa nova configuração não convergiu. A mesma verificação de sobreposição de partículas foi feita, e apresentou valores de sobreposição apenas entre 0.11 e 0.16, indicando não haver aproximação ou sobreposição entre as partículas.

A não convergência combinada com a não sobreposição acusa que as partículas estão fazendo apenas movimentos aleatórios, causados ou por um valor α elevado ou por um γ muito elevado que prejudica a visibilidade das partículas. Como o mesmo valor para parâmetro α obteve sucesso no caso anterior, foi suposto que o valor γ estivesse muito elevado. Portanto foi suposto que algum γ ideal estaria entre $1 \geq \gamma \leq 100$, e para novas configurações foram geradas em busca de um valor adequado para coeficiente de absorção γ.

A busca por um valor γ adequado foi realizada com uma estratégia similar à busca binária. Dado um intervalo inicial, neste caso $[1, 100]$, o valor no meio desse intervalo (50) é avaliado. Essa avaliação consiste executar o método com uma nova configuração contendo o valor γ em questão, e verificar os *biclusters* desejados foram encontrados ou não. Caso a configuração seja adequado a busca é finalizada e o valor encontrado é o ideal, caso contrário, se a configuração com $\gamma = 50$ não convergir, a busca irá continuar no sub-intervalo $[1, 50]$, ou caso essa configuração convirja porém ainda não seja o valor adequado, a busca irá continuar no sub-intervalo $[50, 100]$. O mesmo procedimento será repetido com o novo sub-intervalo e assim sucessivamente, até que o valor adequado seja encontrado ou a busca seja interrompida.

Também com a convergência apresentadas Figura 5, a busca foi realizada da configuração C até configuração H onde foi interrompida. As validações por recuperação e relevância estão apresentado na Tabela 2. A busca foi interrompida pois os resultados obtidos apresentaram apenas os mesmos comportamentos detectados até então: ou todas as partículas convergiam para um mesmo lugar, ou sequer convergiam, sem qualquer indicativo que um γ ideal fosse encontrado. Outro motivo para interrupção é que o valor γ caso fosse encontrado só seria adequado para esse caso de teste, não

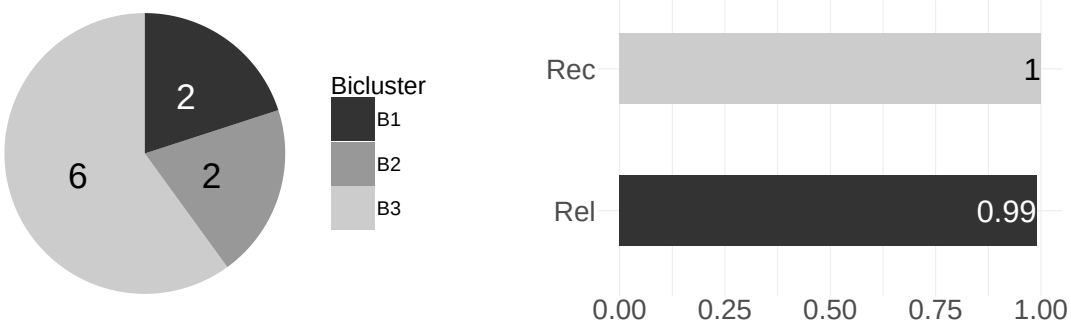
havendo garantia de convergência para outros conjuntos de dados ou outros valores para os parâmetros. Sendo assim, o prospecto de buscar um valor γ para cada caso torna essa abordagem em geral pouco atraente.

Tabela 2 – Valores de recuperação e relevância obtidos com cada configuração executada na matriz de dados sintéticos de tamanho 100×100, com 3 *biclusters* de tamanho 33×33 com valores constante.

Configuração	Recuperação	Relevância
A	28%	82%
B	33%	100%
C	12%	10%
D	13%	10%
E	33%	100%
F	12%	10%
G	12%	10%
H	31%	94%

Foi então proposta uma nova abordagem, ao invés de inicializar um único enxame para buscar múltiplos *biclusters* simultaneamente, os *biclusters* serão buscados um por vez. Primeiro foi verificado se o algoritmo, sem alterações, é capaz de encontrar *biclusters* diferentes apenas com variações no conjunto de soluções iniciais. Para isso foi retomada a configuração inicial, $n = 10$, $\gamma = 1$, e o método foi então executado 10 vezes consecutivas, no qual foram geradas soluções aleatórias diferentes para cada execução. Com os resultados de cada execução foi verificado qual *bicluster* verdadeiro o algoritmo encontrou. O sumário dos resultados estão apresentados na Figura 6. O bicluster $B1$ foi encontrado duas vezes, $B2$ também duas vezes, e $B3$ seis vezes. Para compor o resultado final, foi coletado a melhor partícula de cada execução e combinado em um único resultado composto por dez *biclusters*, com esse conjunto de *biclusters* foi feito o teste de recuperação e relevância, e mais mostrou-se satisfatório, obtendo 100% de recuperação e 99% de relevância.

Figura 6 – Sumário dos resultados encontrados na matriz de dados sintéticos de tamanho 100×100, com 3 *biclusters* de tamanho 33×33 com valores constantes.

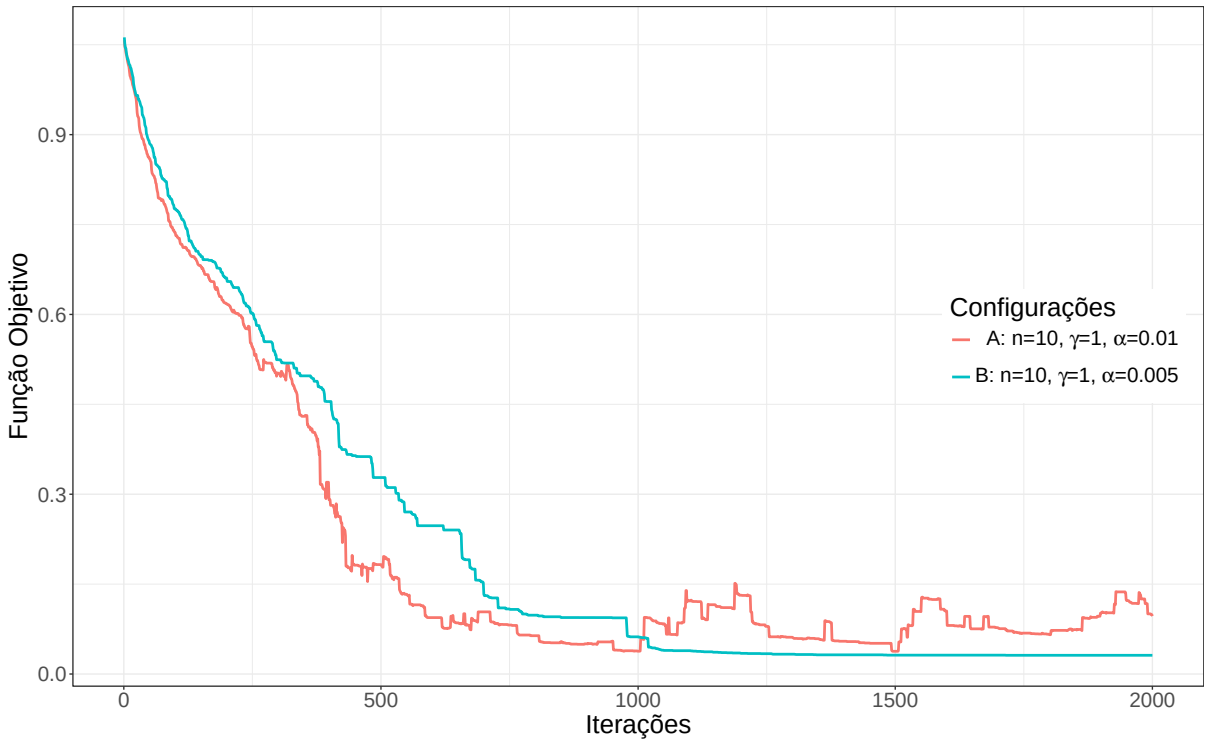


(a) Biclusters encontrados

(b) Recuperação e Relevância

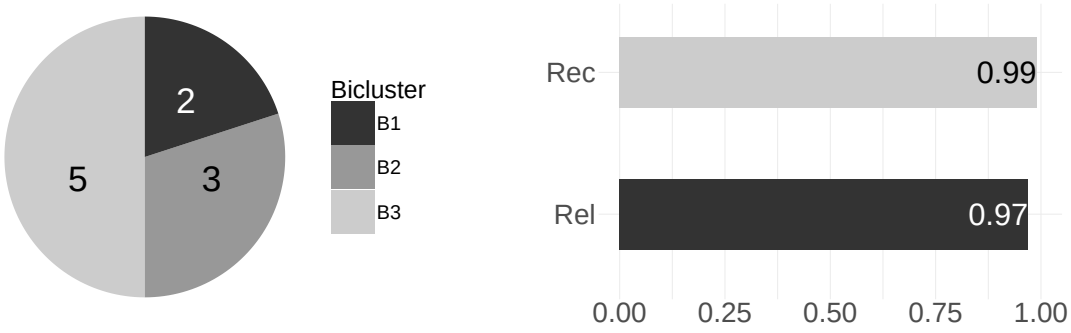
Para o próximo teste, também foi aumentada a complexidade dos dados, dessa vez foi gerada uma matriz quadrada de ordem 200, com 3 *biclusters* embutidos com o padrão de deslocamento. A mesma configuração de parâmetros foi mantida como ponto de partida. O gráfico de convergência está apresentado na Figura 7. Analisando o gráfico, a configuração A apresentou tendência de convergência, porém ao chegar próximo do valor ideal a solução não se manteve estável, caracterizado pelas oscilações no valor da função objetivo. Em prol da exploração o algoritmo irá aceitar soluções piores, porém foi notado uma perda significativa e recorrente na qualidade da solução, e desse comportamento foi suposto que essas oscilações ocorreram devido ao parâmetro $\alpha = 0.01$ estar elevado. Portanto foi proposta uma nova configuração B mantendo os demais parâmetros porém com $\alpha = 0.005$, e essa por sua vez convergiu. Como consequência da redução de α, o algoritmo demorou um pouco mais a convergir, visto que as soluções serão mais conservadas. A conservação da solução está mais clara entre as iterações 850 e 1000, onde se apresenta uma reta, ilustrando que não houve melhoria na solução.

Figura 7 – Gráfico de convergência da execução do método na matriz de dados sintéticos de tamanho 200×200, com 3 *biclusters* de tamanho 66×66 com padrão de deslocamento.



Similar ao caso anterior, a convergência ocorreu apenas para um *bicluster*, para encontrar os três *biclusters* presentes nos dados foi então utilizada a mesma estratégia do caso de testes anterior. O sumário dos *biclusters* encontrados e da validação por recuperação e relevância estão apresentados na Figura 8. O *bicluster* $B3$ foi encontrado

Figura 8 – Sumário dos resultados encontrados na matriz de dados sintéticos de tamanho 200×200, com 3 *biclusters* de tamanho 66×66 com padrão de deslocamento.



(a) Biclusters encontrados

(b) Recuperação e Relevância

com maior frequência que os demais, e a recuperação perfeita (100%) e relevância quase perfeita (99%). Esse resultado atesta a capacidade do método de encontrar *biclusters* do tipo deslocamento.

No próximo caso de testes, foi avaliado a capacidade para encontrar *biclusters* com sobreposição. Para esta avaliação foi gerada uma matriz quadrada de ordem 200, contendo 6 *biclusters* de tamanho 40×40 seguindo o padrão de deslocamento e com sobreposição. Quatro dos seis biclusters fazem sobreposição com outros dois biclusters vizinhos, enquanto que dois biclusters apresentados na extremidades fazem sobreposição com apenas um. O gráfico de convergência está representado na Figura 9. Como a primeira configuração apresentou convergência, foi feita a validação dos resultados. O sumário da validação está apresentando na Figura 10. Para este foi realizado o mesmo procedimento dos testes anteriores, porém como o número de *biclusters* é maior, foram realizadas 20 execuções ao invés de 10. O sumério dos *biclusters* encontrado apresenta um resultado interessante, onde dois *biclusters* não foram encontrados, e justamente os localizados nas extremidades. Também foi dado preferência para o *bicluster* $B4$.

Para verificar, foi gerado mais um conjunto de dados, uma matriz quadrada de ordem 200, dessa vez contendo apenas 2 *biclusters*, um bicluster $B1$ de tamanho 505×50 e outro *bicluster* $B2$ de tamanho 75×75, ambos seguindo um padrão de deslocamento. Com essa configuração, é esperado que o *bicluster* $B2$ seja encontrado mais vezes. Como $B2$ é maior, haverá maior probabilidade de que durante as iterações iniciais do enxame o encontre. Levando em consideração a função objetivo, os *biclusters* $B1$ e $B2$ seguem um padrão perfeito de deslocamento, porém o $B2$ é maior que $B1$ e o volume é um dos critérios de qualidade, portanto $B2$ será considerado mais atraente pela função objetivo. Os valores da função objetivo para os *biclusters* são: $f(B1) = 0.04$ e $f(B2) = 0.026$,

Continuando com a mesma configuração o método foi executado. A convergência apresentada na Figura 11 apresentou comportamento normal. Já o sumário dos

Figura 9 – Gráfico de convergência da execução do método na matriz de dados sintéticos de tamanho 200×200, com 6 *biclusters* com sobreposição

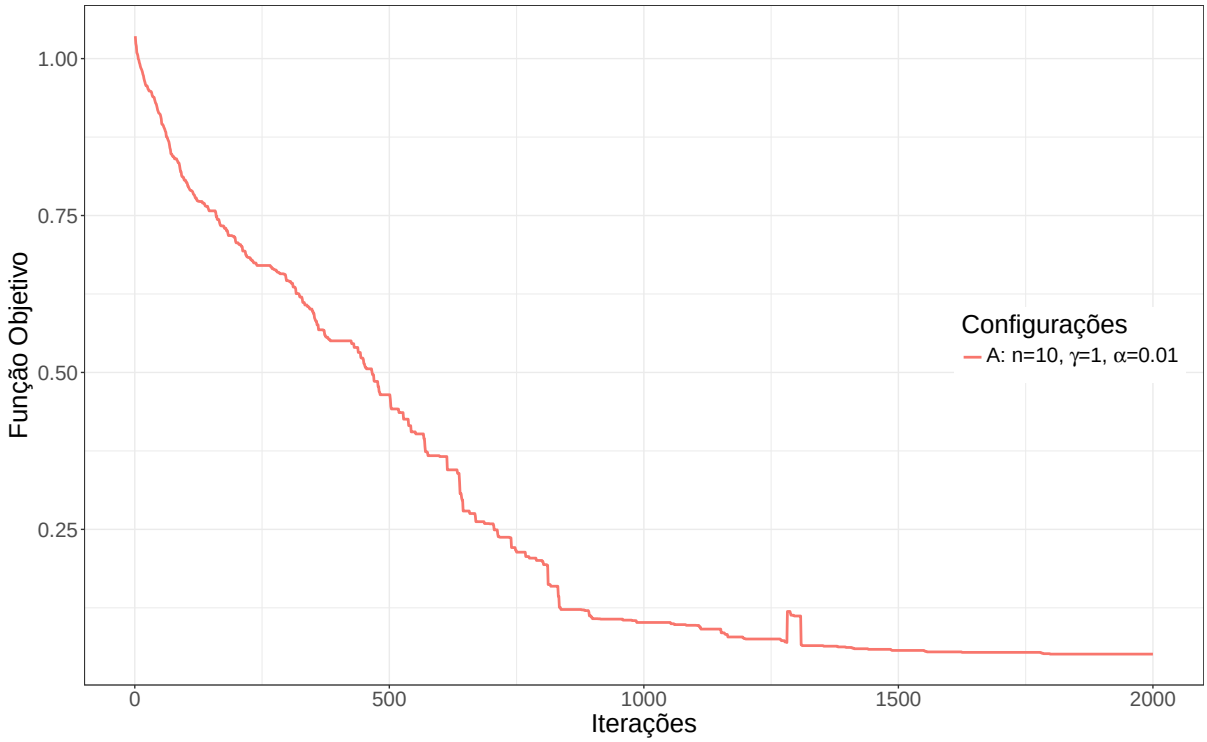
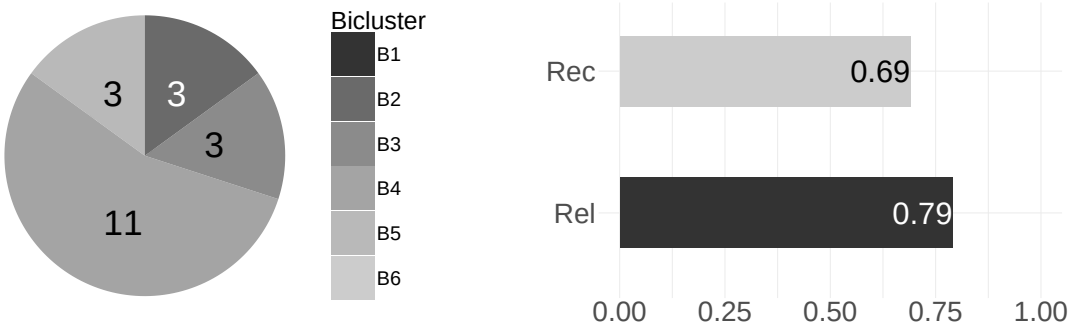


Figura 10 – Sumário dos resultados encontrados na matriz de dados sintéticos de tamanho 200×200, com 6 *biclusters* com sobreposição



(a) Biclusters encontrados

(b) Recuperação e Relevância

biclusters encontrados (Figura 12) mostrou que apenas o *bicluster* mais atraente ($B2$) foi encontrado.

Os resultados dos últimos dois casos testes atestam uma falha no método, onde alguns *biclusters* não são encontrados, ou pela estratégia de inicialização que irá naturalmente favorecer alguns *biclusters*, ou pelo viés a favor de *biclusters* de maior qualidade. Se supõe que essa falha pode ser corrigida pela adoção do mecanismo de penalidade presente na métrica proposta por Divina e Aguilar-Ruiz (2006). Como foi descrito na secção de modelagem, a função objetivo utilizada neste trabalho se baseia na métrica

Figura 11 – Gráfico de convergência da execução do método na matriz de dados sintéticos de tamanho 200×200, com 2 *biclusters* $B1$ e $B2$ seguindo padrão de deslocamento, $B1$ de tamanho 50×50 e $B2$ de tamanho 75×75.

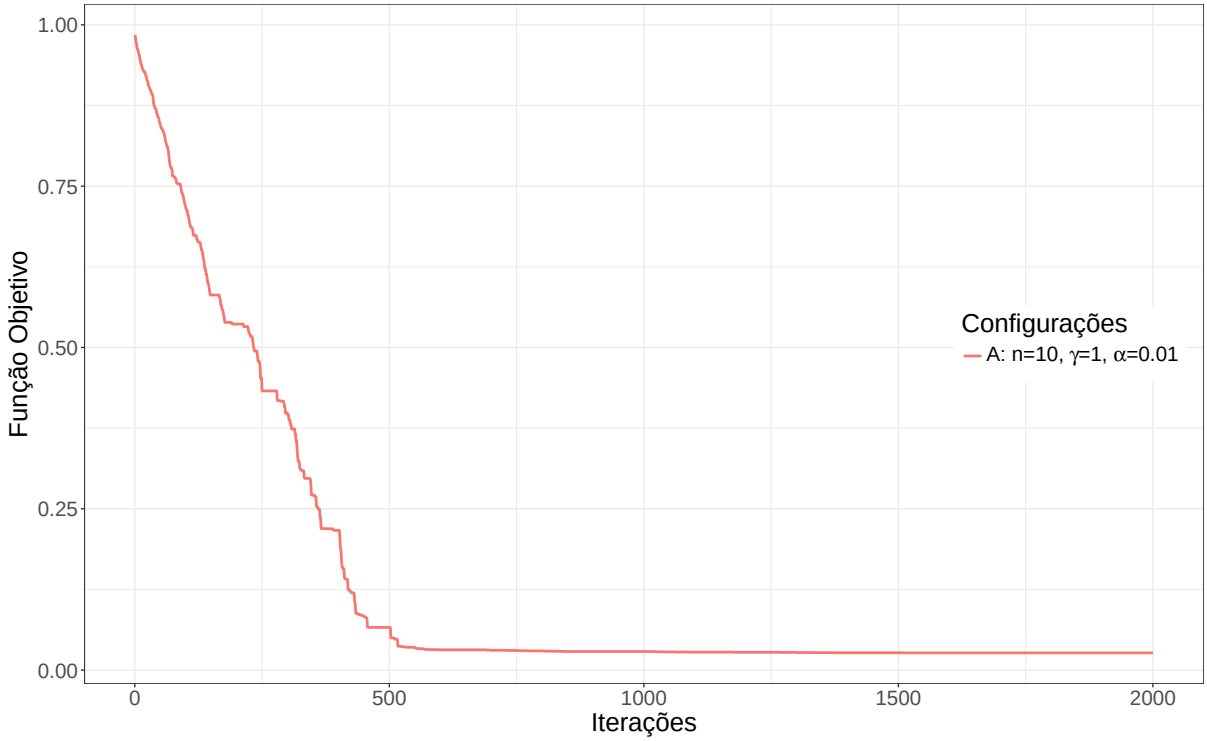
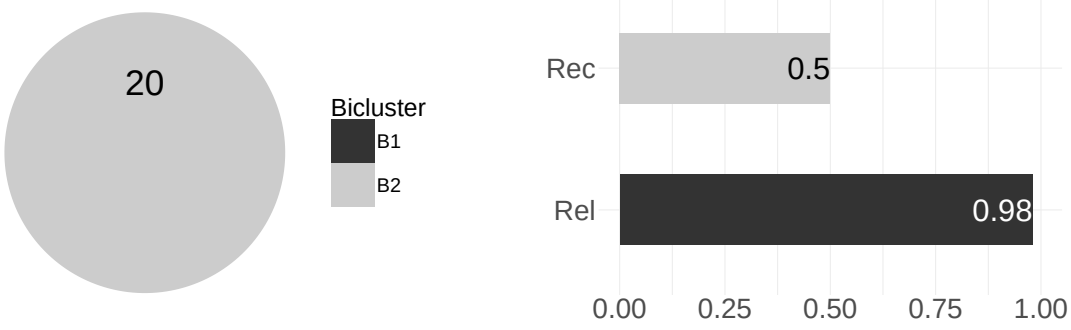


Figura 12 – Sumário dos resultados encontrados na matriz de dados sintéticos de tamanho 200×200, com 2 *biclusters* $B1$ e $B2$ seguindo padrão de deslocamento, $B1$ de tamanho 50×50 e $B2$ de tamanho 75×75.



(a) Biclusters encontrados

(b) Recuperação e Relevância

citada, porém com alguns ajustes, como a remoção de termos julgados desnecessários. A expressão referente a penalidade foi dada como desnecessária pois se esperava que o método proposto encontrasse múltiplos biclusters simultaneamente, através da divisão do enxame inicial em múltiplos sub-enxames. Porém como ilustrado no segundo caso de testes, tal dinâmica deveria ocorrer com um valor adequado para o coeficiente de absorção γ, porém o mesmo não foi encontrado, e todas as partículas convergiram para um mesmo ponto, encontrando apenas um *bicluster*. Portanto com a estratégia de

encontrar um bicluster por vez, é necessário haver algum mecanismo que permita que outros biclusters sejam encontrados.

6.1 Expressão de penalidade

Como sugestão de solução, pode-se aderir o mecanismo de penalidade de Divina e Aguilar-Ruiz (2006), que atribui pesos às células da matriz com base nos *biclusters* já encontrados. Suponha um conjunto de biclusters encontrados R e uma célula b_{ij}, quanto mais *biclusters* em R possuírem b_{ij}, maior será o peso atrelado a b_{ij}. O método de Divina e Aguilar-Ruiz (2006) retorna apenas um ou nenhum *bicluster* por execução, portanto inicialmente o conjunto R é vazio e todos os pesos são 0, então a cada execução é adicionado um *bicluster* (ou nenhum caso não encontre) ao conjunto R e os pesos são atualizados. Em cada execução do método deste trabalho são encontrados n (número de vaga-lumes) *biclusters* redundantes, portanto para aproveitar o mecanismo de penalidade da maneira que foi proposto, apenas o melhor dos *biclusters* redundantes deverá ser incluído no conjunto de resultados R.

Para calcular os pesos de cada célula b_{lc} tais que $l = \{1, 2, ..., |L|\}$ e $c = \{1, 2, ..., |C|\}$, primeiro são contadas as ocorrências de determinada célula b_{lc} no conjunto de resultados $R = B_1, B_2, ..., B_n$. Para isso é calculada a cobertura da célula em questão, a cobertura é o conjunto dos *biclusters* em R que a possuam, representada por $Cob(l, c)$. Note que aqui há uma distinção entre a célula b_{lc} e o valor contido na mesma, portanto $Cob(l, c)$ considera as ocorrências da célula não do valor. Com a contagem das ocorrências dos elementos, o cálculo dos peso é dado por:

$$w_p(l, c) = \frac{\sum_{i=1}^{|L|} \sum_{j=1}^{|C|} e^{-|Cob(i,j)|}}{e^{-|Cob(l,c)|}}. \tag{30}$$

Essa expressão do peso é inspirada em uma expressão de peso para operadores de seleção de indivíduos em algoritmos genéticos e evolutivos (DIVINA; MARCHIORI, 2005).

Com os pesos das células da matriz, a penalidade então será parte da função objetivo, sendo calculada como o somatório dos pesos das células contidas no *bicluster*, dada por:

$$\sum_{i=1}^{|I|} \sum_{j=1}^{|J|} w_p(i, j). \tag{31}$$

Portanto a função objetiva sugerida para superar tal falha é:

$$f(B) = \frac{MSR(B)}{\delta} + w_v(w_l \frac{\delta}{|I|} + w_c \frac{\delta}{|J|}) + \sum_{i=1}^{|I|} \sum_{j=1}^{|J|} w_p(i, j) \tag{32}$$

Para ilustrar o uso da expressão, considere o exemplo apresentado na Tabela 3. Nesse exemplo se considera que três *biclusters* foram encontrados: $B1$, $B2$, $B3$. Considere uma matriz hipotética de tamanho 5×5 tal que $i \in \{1, 2, ...5\}$, $j \in \{1, 2, ...5\}$. Logo abaixo serão apresentadas duas matrizes: C e W, a matriz C apresenta o $Cob(i, j)$ de cada célula da matriz original, e a W contém o peso calculado para cada célula:

$$
\begin{aligned}
C = & \begin{bmatrix} \text{\{B1, B2, B3\}} & \text{\{B1\}} & \text{\{B2\}} & \text{\{B3\}} & \text{\{B3\}} \\ \text{\{B1\}} & \text{\{B1\}} & \text{\{\}} & \text{\{\}} & \text{\{\}} \\ \text{\{B1, B2\}} & \text{\{B1\}} & \text{\{B2\}} & \text{\{\}} & \text{\{\}} \\ \text{\{B3\}} & \text{\{\}} & \text{\{\}} & \text{\{B3\}} & \text{\{B3\}} \\ \text{\{B3\}} & \text{\{\}} & \text{\{\}} & \text{\{B3\}} & \text{\{B3\}} \end{bmatrix}, \\
W = & \begin{bmatrix} 287{,}93 & 38{,}97 & 38{,}97 & 38{,}97 & 38{,}97 \\ 38{,}97 & 38{,}97 & 0 & 0 & 0 \\ 105{,}92 & 38{,}97 & 38{,}97 & 0 & 0 \\ 38{,}97 & 0 & 0 & 38{,}97 & 38{,}97 \\ 38{,}97 & 0 & 0 & 38{,}97 & 38{,}97 \end{bmatrix}.
\end{aligned}
\tag{33}
$$

Nesse exemplo, caso o *bicluster* $B2$ fosse avaliado durante a busca, seria adicionado ao valor final da função objetivo o peso das células presentes, neste caso $w_p(b_{11}) = 287,93$, $w_p(b_{13}) = 38,97$, $w_p(b_{33}) = 105,92$, $w_p(b_{33}) = 38,97$, totalizando $471,79$ de penalidade. Esse valor adicionado à função objetivo irá criar um viés em favor de *biclusters* com menos elementos que já foram encontrados.

Tabela 3 – Exemplo de *biclusters* encontrados em uma busca hipotética, no formato para se calcular os pesos

Bicluster	Linhas	Colunas	Células
B1	$I = \{1, 2, 3\}$	$J = \{1, 2\}$	$\{b_{11}, b_{12}, b_{21}, b_{22}, b_{31}, b_{32}\}$
B2	$I = \{1, 3\}$	$J = \{1, 3\}$	$\{b_{11}, b_{13}, b_{31}, b_{33}\}$
B3	$I = \{1, 4, 5\}$	$J = \{1, 4, 5\}$	$\{b_{11}, b_{14}, b_{15}, b_{41}, b_{44}, b_{45}, b_{51}, b_{54}, b_{55}\}$

7 Conclusões

A grande contribuição deste trabalho é a descrição do método do Algoritmo *Firefly* para *biclustering* através de uma abordagem de otimização combinatorial. A abordagem utilizada mostrou-se viável, uma vez que encontrou resultados bons, bem próximos do resultado ideal, em três dos cinco casos de teste.

As causas da falha puderam ser identificadas nos outros dois casos teste graças ao uso de dados sintéticos, caso o método fosse executado em dados reais, não haveriam pontos de referência a qual comparar os resultados. Nesses casos os resultados obtidos estavam incompletos, devido ao método dar preferência para alguns *biclusters* com base no tamanho, qualidade e localização. Foi proposto como solução incorporar um mecanismo de penalidade para *biclusters* já encontrados.

Um resultado inesperado foi que a dinâmica de subdivisão de enxames do Algoritmo *Firefly* não mostrou-se presente, fazendo com que todas as partículas convergissem para um mesmo *bicluster*. Essa era uma das características desejáveis, visto que *biclustering* trata-se de um problema multimodal. Com a ausência dessa dinâmica, os *biclusters* tiveram que ser buscados individualmente, sendo realizadas múltiplas buscas com diferentes soluções iniciais. O método desenvolvido, sem alterações em sua formulação, foi capaz de encontrar *biclusters* diferentes apenas com variações nas soluções iniciais. Se supõe que tal dinâmica não ocorreu devido ao calculo de distância utilizado.

O método proposto foi desenvolvido utilizando as técnicas consideradas mais básicas e fundamentais apresentadas na literatura, consequentemente diversas melhorias podem ser feitas. Uma delas é em relação à medida de qualidade, o *MSR* foi utilizado mesmo com suas limitações em encontrar *biclusters* seguindo padrões de escala. Da forma que é estruturado, a função objetivo pode ser atualizada sem ser necessário fazer alterações nos procedimentos do método.

Outro resultado interessante é que o método funcionou bem mesmo com a inicialização totalmente aleatória, diferente de relatos encontrados na literatura que apontam essa estratégia de inicialização não leva a bons resultados. Mitra e Banka (2006) reportaram que o seu método só foi capaz de encontrar bons resultados caso fosse feita uma busca local nas soluções aleatórias, e Divina e Aguilar-Ruiz (2006) que teve que inicializar *biclusters* aleatórios porém de tamanho um.

Nos resultados obtidos foi constado que o método desenvolvido neste trabalho não é prejudicado em relação as soluções iniciais. Isso possibilita que diferentes estratégias de inicialização sejam empregadas, como uma inicialização menos aleatória

porém melhor distribuída no espaço de busca, ou até a inicialização com resultados encontrados por outros métodos.

Por fim o método desenvolvido não é uma tentativa de desenvolver um método de *biclustering* superior aos demais, mas sim uma forma de explorar os fundamentos do Algoritmo *Firefly* para *biclustering* de dados de expressão. As técnicas aqui empregadas podem ser utilizadas por futuros pesquisadores, para implementar suas próprisa versões de *bicluster* com o Algoritmo *Firefly*, ou desenvolver métodos híbridos com outras heurísticas promissoras.

Referências

AGUILAR-RUIZ, J. S. Shifting and scaling patterns from gene expression data. **Bioinformatics**, v. 21, n. 20, p. 3840–3845, 2005. ISSN 13674803.

AL-AKWAA, F. M. Analysis of Gene Expression Data Using Biclustering Algorithms. In: **Functional Genomics**. [S.l.]: InTech, 2012.

ALBERTS, B. et al. **Biologia Molecular da Célula**. [S.l.: s.n.], 2010. ISSN 02140659. ISBN 9788578110796.

ASYALI, M. et al. Gene Expression Profile Classification: A Review. **Current Bioinformatics**, v. 1, n. 1, p. 55–73, 2006. ISSN 15748936.

BONNEAU, R. Learning biological networks: From modules to dynamics. **Nature Chemical Biology**, v. 4, n. 11, p. 658–664, 2008. ISSN 15524469.

BONNEAU, R. et al. The inferelator: An algorithn for learning parsimonious regulatory networks from systems-biology data sets de novo. **Genome Biology**, v. 7, n. 5, 2006. ISSN 1474760X.

BOOKSTEIN, A.; KULYUKIN, V. A.; RAITA, T. Generalized hamming distance. **Information Retrieval**, v. 5, n. 4, p. 353–375, 2002. ISSN 13864564.

BOYD, S.; VANDENBERGHE, L. **Convex Optimization**. [S.l.: s.n.], 2006. v. 51. 1859–1859 p. ISSN 0018-9286. ISBN 978-0-521-83378-3.

BROWN, P. O.; BOTSTEIN, D. Exploring the new world of the genome with DNA microarrays. **Nature Genetics**, v. 21, n. 1 Suppl, p. 33–37, 1999. ISSN 1061-4036.

CHENG, Y.; CHURCH, G. M. Biclustering of Expression Data. **Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology**, v. 3721, p. 93–103, 2000. ISSN 1553-0833.

CHUANG, L.-Y. et al. Improved binary PSO for feature selection using gene expression data. **Computational Biology and Chemistry**, v. 32, n. 1, p. 29–38, 2008. ISSN 14769271.

COELHO, G. P.; De França, F. O.; Von Zuben, F. J. Multi-objective biclustering: When non-dominated solutions are not enough. **Journal of Mathematical Modelling and Algorithms**, v. 8, n. 2, p. 175–202, jun 2009. ISSN 15701166.

DALE, J.; NISHIMOTO, A.; OBAFEMI-AJAYI, T. Performance Evaluation and Enhancement of Biclustering Algorithms. **Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods**, n. Icpram, p. 202–213, 2018.

De Smet, R.; MARCHAL, K. Advantages and limitations of current network inference methods. **Nature Reviews Microbiology**, Nature Publishing Group, v. 8, n. 10, p. 717–729, 2010. ISSN 17401526.

DEB, K. Multi-objective optimization using evolutionary algorithms: An Introduction. **IEEE Wireless Communications and Networking Conference, WCNC**, n. 2002, p. 3030–3034, 2012. ISSN 15253511.

DEB, K. et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. **IEEE Transactions on Evolutionary Computation**, v. 6, n. 2, p. 182–197, 2002. ISSN 1089778X.

DIVINA, F.; AGUILAR-RUIZ, J. S. Biclustering of expression data with evolutionary computation. **IEEE Transactions on Knowledge and Data Engineering**, v. 18, n. 5, p. 590–602, 2006. ISSN 10414347.

DIVINA, F.; MARCHIORI, E. Knowledge-Based Evolutionary Search for Inductive Concept Learning. In: **Knowledge Incorporation in Evolutionary Computation**. [S.l.]: Springer, Berlin, Heidelberg, 2005. p. 237–253. ISBN 978-3-540-44511-1.

DORIGO, M.; BIRATTARI, M.; STÜTZLE, T. Metaheuristic. In: **Encyclopedia of Machine Learning and Data Mining**. Boston, MA: Springer US, 2017. p. 817–818.

DURKOTA, K. **Implementation of a discrete firefly algorithm for the QAP problem within the Seage framework**. Tese (Doutorado) — Czech technical university in Prague, 2011.

EBERHART, R. C.; SHI, Y. Comparison between genetic algorithms and particle swarm optimization. In: **Electronic Theses and Dissertations**. [S.l.: s.n.], 1998. p. 611–616. ISBN 978-3-540-64891-8.

EREN, K. **Application of Biclustering Algorithms to Biological Data**. 91 p. Tese (Masters) — The Ohio State University, 2012.

EREN, K. et al. A comparative analysis of biclustering algorithms for gene expression data. **Briefings in Bioinformatics**, Oxford University Press, v. 14, n. 3, p. 279–292, may 2013. ISSN 1467-5463.

GAN, X.; LIEW, A. W. C.; YAN, H. Discovering biclusters in gene expression data based on high-dimensional linear geometries. **BMC Bioinformatics**, v. 9, p. 1–15, 2008. ISSN 14712105.

GENDREAU, M.; POTVIN, J. Y. Metaheuristics in combinatorial optimization. **Annals of Operations Research**, v. 140, n. 1, p. 189–213, 2005. ISSN 02545330.

GUTTMAN, M. et al. Chromatin signature reveals over a thousand highly conserved large non-coding RNAs in mammals. **Nature**, Nature Publishing Group, v. 458, n. 7235, p. 223–227, 2009. ISSN 00280836.

HARRINGTON, C. A.; ROSENOW, C.; RETIEF, J. Monitoring gene expression using DNA microarrays. **Current opinion in microbiology**, v. 3, n. 3, p. 285–91, jun 2000. ISSN 1369-5274.

HARTIGAN, J. A. Direct Clustering of a Data Matrix. **Journal of the American Statistical Association**, v. 67, n. 337, p. 123–129, mar 1972. ISSN 0162-1459.

HOOS, H. H.; STÜTZLE, T. **Stochastic Local Search: Foundations and Applications**. [S.l.]: Elsevier, 2004. 658 p. ISBN 1-55860-872-9.

HUANG, D. W.; SHERMAN, B. T.; LEMPICKI, R. A. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. **Nature Protocols**, v. 4, n. 1, p. 44–57, jan 2009. ISSN 1754-2189.

HURD, P. J.; NELSON, C. J. Advantages of next-generation sequencing versus the microarray in epigenetic research. **Briefings in Functional Genomics and Proteomics**, v. 8, n. 3, p. 174–183, may 2009. ISSN 1473-9550.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: **Proceedings of ICNN'95 - International Conference on Neural Networks**. [S.l.]: IEEE, 1995. v. 4, p. 1942–1948. ISBN 0-7803-2768-3.

KENNEDY, J.; EBERHART, R. A discrete binary version of the particle swarm algorithm. **1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation**, v. 5, p. 4104–4108, 1997. ISSN 1062-922X.

LIU, J. et al. Biclustering of microarray data with MOSPO based on crowding distance. **BMC Bioinformatics**, v. 10, n. SUPPL. 4, p. 1–10, 2009. ISSN 1471-2105.

MADEIRA, S. C.; OLIVEIRA, A. L. Biclustering algorithms for biological data analysis: a survey. **Transactions on computational biology and bioinformatics**, v. 1, n. 1, p. 24–45, 2004. ISSN 1545-5963.

MELOUANE, A. et al. Differential gene expression analysis in ageing muscle and drug discovery perspectives. **Ageing Research Reviews**, Elsevier B.V., v. 41, p. 53–63, 2018. ISSN 18729649.

MILLONAS, M. M. Swarms, Phase Transitions, and Collective Intelligence. In: ____. **Computational Intelligence: A Dynamic System Perspective**. [S.l.: s.n.], 1994. cap. Swarms, Ph, p. 137–151.

MITRA, S.; BANKA, H. Multi-objective evolutionary biclustering of gene expression data. **Pattern Recognition**, v. 39, n. 12, p. 2464–2477, 2006. ISSN 00313203.

PONTES, B.; GIRÁLDEZ, R.; AGUILAR-RUIZ, J. S. Biclustering on expression data: A review. **Journal of Biomedical Informatics**, Elsevier Inc., v. 57, p. 163–180, 2015. ISSN 15320464.

PONTES, B.; GIRLDEZ, R.; AGUILAR-RUIZ, J. S. Quality measures for gene expression biclusters. **PLoS ONE**, v. 10, n. 3, p. 1–24, 2015. ISSN 19326203.

PRELIĆ, A. et al. A systematic comparison and evaluation of biclustering methods for gene expression data. **Bioinformatics**, v. 22, n. 9, p. 1122–1129, 2006. ISSN 13674803.

REIMAND, J. et al. g:Profiler—a web server for functional interpretation of gene lists (2016 update). **Nucleic Acids Research**, v. 44, n. W1, p. W83–W89, jul 2016. ISSN 0305-1048.

REISS, D. J.; BALIGA, N. S.; BONNEAU, R. Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks. **BMC Bioinformatics**, v. 7, p. 1–22, 2006. ISSN 14712105.

SERIDI, K.; JOURDAN, L.; TALBI, E. G. Using multiobjective optimization for biclustering microarray data. **Applied Soft Computing Journal**, Elsevier B.V., v. 33, p. 239–249, 2015. ISSN 15684946.

SHANNON, P. et al. Cytoscape: a software environment for integrated models of biomolecular interaction networks. **Genome research**, v. 13, n. 11, p. 2498–504, nov 2003. ISSN 1088-9051.

SHI, Y.; EBERHART, R. A modified particle swarm optimizer. In: **1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)**. [S.l.: s.n.], 1998. p. 69–73.

SU, Z. et al. A comprehensive assessment of RNA-seq accuracy, reproducibility and information content by the Sequencing Quality Control Consortium. **Nature Biotechnology**, v. 32, n. 9, p. 903–914, 2014. ISSN 15461696.

SZKLARCZYK, D. et al. The STRING database in 2017: Quality-controlled protein-protein association networks, made broadly accessible. **Nucleic Acids Research**, v. 45, n. D1, p. D362–D368, 2017. ISSN 13624962.

TANAY, A.; SHARAN, R.; SHAMIR, R. Discovering statistically significant bicluster in gene expression data. **Bioinformatics (Oxford, England)**, v. 18, n. Suppl. 1, p. S136 – S144, 2002. ISSN 1367-4803.

THERMES, C. Ten years of next-generation sequencing technology. **Trends in genetics : TIG**, v. 30, n. 9, p. 418–426, 2014. ISSN 01689525.

TILAHUN, S. L.; NGNOTCHOUYE, J. M. T. Firefly algorithm for discrete optimization problems: A survey. **KSCE Journal of Civil Engineering**, v. 21, n. 2, p. 535–545, 2017. ISSN 19763808.

WANG, Z.; GERSTEIN, M.; SNYDER, M. RNA-Seq: a revolutionary tool for transcriptomics. **Nature Reviews Genetics**, v. 10, n. 1, p. 57–63, jan 2009. ISSN 1471-0056.

WEYLAND, D. A critical analysis of the harmony search algorithm — How not to solve sudoku. **Operations Research Perspectives**, Elsevier Ltd, v. 2, p. 97–105, 2015. ISSN 2214-7160.

XIE, J. et al. It is time to apply biclustering: a comprehensive review of biclustering applications in biological and biomedical data. **Briefings in Bioinformatics**, n. June, p. 1–16, 2018. ISSN 1467-5463.

YANG, X. S. Firefly algorithms for multimodal optimization. **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**, v. 5792 LNCS, p. 169–178, 2009. ISSN 03029743.

YANG, X. S.; HE, X. S. Why the firefly algorithm works? **Studies in Computational Intelligence**, v. 744, p. 245–259, 2018. ISSN 1860949X.

YOON, S. H. et al. A systems level predictive model for global gene regulation of methanogenesis in a hydrogenotrophic methanogen. **Genome Research**, v. 23, n. 11, p. 1839–1851, nov 2013. ISSN 1088-9051.

YU, X.; GEN, M. Multimodal Optimization. In: **Introduction to Evolutionary Algorithms**. 0. ed. [S.l.]: Springer, London, 2010. ISBN 978-1-84996-129-5.

ZHAO, S. et al. Comparison of RNA-Seq and microarray in transcriptome profiling of activated T cells. **PLoS ONE**, v. 9, n. 1, 2014. ISSN 19326203.

ZOMAYA, A. Y. **Handbook of Nature-Inspired and Innovative Computing**. Boston: Kluwer Academic Publishers, 2006. ISBN 0-387-40532-1.