UNIVERSIDADE DE CAXIAS DO SUL
CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

THIAGO ANGELO BASSO

CLUSTERIZAÇÃO APLICADA NA ANÁLISE GENÔMICA

CAXIAS DO SUL
2015

THIAGO ANGELO BASSO

CLUSTERIZAÇÃO APLICADA NA ANÁLISE GENÔMICA

Trabalho de conclusão de curso para graduação em Ciência da Computação pela Universidade de Caxias do Sul.
Área de concentração: biotecnologia

Orientador Prof. Esp. Daniel Antônio Faccin

CAXIAS DO SUL
2015

RESUMO

A necessidade pelo desenvolvimento de metodologias computacionais que auxiliem na análise dos dados de origem biológica cresce constantemente. A bioinformática é área de conhecimento onde ferramentas dos campos da computação e matemática são aplicadas na geração e manipulação de informações biológicas. Um esforço muito grande vem sendo realizado nesta área, especialmente na análise genômica, desde a década de 90, quando grandes projetos foram iniciados com o objetivo de sequenciar o DNA dos organismos vivos conhecidos. Neste trabalho são analisadas sequências do DNA da bactéria E.coli. Os dados são obtidos de uma base de dados pública e são submetidos a aplicação de uma técnica de mineração de dados, a clusterização. O foco está na etapa de seleção e extração de atributos, onde diferentes abordagens são aplicadas a fim de reduzir a quantidade de ruídos e informações irrelevantes contidas nas amostras e aprimorar a análise da região promotora da expressão gênica, bem como a identificação de padrões expressos na mesma.

Palavras-chave: bioinformática, análise genômica, clusterização, região promotora, seleção de atributos, extração de atributos

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE EQUAÇÕES

LISTA DE SIGLAS

ATE	*Automatic Term Expansion*
CFS	*Correlation based Feature Selection*
CHI	*Chi Square*
CURE	*Clustering Using Representatives*
DFT	*Discrete Fourier Transform*
DNA	*Deoxyribonucleic acid*
DWT	*Discrete Wavelet Transform*
FGKA	*Fast Genetic k-means Algorithm*
FT	*Fourier Transform*
GKA	*Genetic k-means Algorithm*
HGKA	*Hibrid Genetic k-means Algorithm*
IDF	*Inverse Document Frequency*
IG	*Information Gain*
IGKA	*Incremental Genetic k-means Algorithm*
IR	*Information Retrieval*
KNN	*K-Nearest Neighbors*
LDA	*Linear Discriminant Analysis*
MUSA	*Motif Finding Using na Unsupervised Approach*
ORF	*Open Reading Frame*
PAM	*Partitioning Around Medoids*
PCA	*Principal Component Analysis*
QE	*Query Expansion*

RNA	*Ribonucleic acid*
RNAp	*RNA* polimerase
SOM	*Self Organized Maps*
SVM	*Support Vector Machine*
TF	*Term Frequency*
VSM	*Vector Space Model*

SUMÁRIO

1 INTRODUÇÃO

A biologia tem sido tradicionalmente uma ciência observacional ao invés de dedutiva. Apesar dos recentes desenvolvimentos não terem alterado essa orientação básica, a natureza dos dados tem mudado radicalmente. Durante muito tempo, todas as observações biológicas eram admitidas com variados níveis de precisão. Entretanto, nas últimas duas décadas, os dados se tornaram mais quantitativos e precisos, especialmente no caso das sequências de aminoácidos e nucleotídeos, os quais tornou possível a determinação da sequência genômica por completa do organismo de qualquer indivíduo (LESK, 2008).

A partir dessa evolução, nasceu a bioinformática, que tem como uma das suas principais aplicações, o emprego da tecnologia da informação na manipulação de dados biológicos. No mesmo período, o armazenamento destes dados em bases de dados públicas se tornou crescentemente comum, e elas têm crescido exponencialmente. A pesquisa na bioinformática e biologia computacional pode abranger qualquer informação sobre a abstração de propriedades de um sistema biológico em um modelo físico, ou matemático, para a implementação de novos algoritmos para a análise dos dados.

Dados biológicos advindos do conhecimento genômico são relativamente complexos em comparação aos provenientes de outras áreas científicas, dada a sua diversidade e ao seu inter-relacionamento (SANTOS & ORTEGA, 2002).

Tecnologias para geração de sequenciamentos do ácido desoxirribonucleico (DNA) estão se desenvolvendo rapidamente. Elas permitem uma visão global e simultânea dos níveis de transcrição de milhares de genes. Para muitos organismos que tiveram seus genomas completamente sequenciados, o conjunto inteiro de genes já pode ser monitorado dessa forma hoje. O potencial das tais tecnologias é grande: a informação obtida através do monitoramento dos níveis de expressão gênica em diferentes estágios de desenvolvimento, condições e organismos pode ajudar no entendimento da função dos genes e redes genéticas, e auxiliar no diagnóstico de doenças e nos efeitos de tratamentos medicinais (SHAMIR & SHARAN, 2001).

Devido a necessidade de se analisar essa grande quantidade de dados biológicos, diversas técnicas de mineração de dados tiveram de ser adaptadas e

aperfeiçoadas para a área da análise genômica, dentre elas a clusterização. Um passo chave na análise dos dados de expressões gênicas é a identificação dos grupos de genes que manifestam padrões similares de expressão. Essa tarefa é transferida para os algoritmos de mineração, a fim de organizar os dados da expressão gênica.

Um dos problemas de clusterização consiste de um, ou mais elementos, e um vetor de características para cada elemento. Uma medida de similaridade é definida entre pares de vetores. Analogamente, na expressão gênica, os elementos são os genes, o vetor de cada gene contém seus níveis de expressão, que são suas características, e a similaridade pode ser medida, por exemplo, pelo coeficiente de correlação entre os vetores. O objetivo é dividir os elementos em subconjuntos, que são chamados de *clusters*, para que dois critérios sejam satisfeitos: homogeneidade - elementos no mesmo *cluster* são altamente similares aos outros; e separação - elementos de diferentes *clusters* têm baixa similaridade aos outros.

1.1 PROBLEMA DE PESQUISA

O estudo da informação contida nos genomas é chamado de genômica. Muito embora saibamos que trincas codificam os aminoácidos nos segmentos codificantes de proteínas, grande parte da informação contida em um genoma não é decifrável por mera inspeção (GRIFFITHS *et al*., 2010).

Atualmente a bioinformática é imprescindível para a manipulação dos dados biológicos. Ela pode ser definida como uma modalidade que abrange todos os aspectos de aquisição, processamento, armazenamento, distribuição, análise e interpretação da informação biológica. Através da combinação de procedimentos e técnicas da matemática, estatística e ciência da computação, são elaboradas várias ferramentas que nos auxiliam a compreender o significado biológico representado nos dados genômicos (BORÉM & SANTOS, 2001).

A sequência do DNA é uma estrutura complexa e, para analisá-la, o uso de metodologias que automatizam e facilitam essa análise é imprescindível.

1.2 QUESTÃO DE PESQUISA

Baseado no problema de pesquisa descrito na seção anterior, foi criada a seguinte questão de pesquisa:

É possível analisar sequências de DNA provenientes de bases de dados públicas e aplicar métodos de análise alternativos para encontrar resultados mais satisfatórios do que os atingidos com os modelos já implementados? É possível aplicar técnicas de clusterização para classificar sequências de um mesmo contexto biológico e agrupá-las em clusters distintos?

O objetivo deste trabalho é identificar padrões expressos nas sequências de DNA, em especial na região promotora do DNA, através da aplicação da técnica de clusterização.

1.3 ESTRUTURA DO TRABALHO

A estrutura deste tralho está dividida em 6 capítulos, no capítulo 1 são descritos os conceitos iniciais, bem como a questão e o problema de pesquisa que são a motivação deste trabalho, no capítulo 2, nomeado referencial teórico, são apresentados os fundamentos biológicos referentes a área de análise genômica e uma descrição da bactéria *Escherichia Coli (E. coli)*, que será o organismo a ser estudado. Neste capítulo também é abordada uma visão geral sobre o conceito de bioinformática, a fim de embasar o conteúdo dos capítulos seguintes. No capítulo 3, clusterização e transformação de atributos, uma técnica de mineração de dados é apresentada, bem como suas etapas, seus principais algoritmos e aplicações na área da bioinformática. As etapas de seleção e extração de atributos são abordadas mais especificamente, detalhando 3 metodologias distintas, que serão aplicadas para atingir o objetivo principal deste trabalho. No capítulo 4, metodologia, o fluxo do trabalho que realizado, as etapas da construção e o desenvolvimento da solução para o problema de pesquisa são demonstrados, bem como as ferramentas e dados que serão utilizados. No capítulo 5, resultados e discussões, são apresentados os resultados dos testes realizados com a ferramenta desenvolvida e, por fim, no capítulo 6, é descrita a conclusão deste trabalho.

2 REFERENCIAL TEÓRICO

Neste capítulo são apresentados alguns conceitos biológicos que serão de grande importância para a compreensão deste trabalho e como a informática está inserida neste contexto.

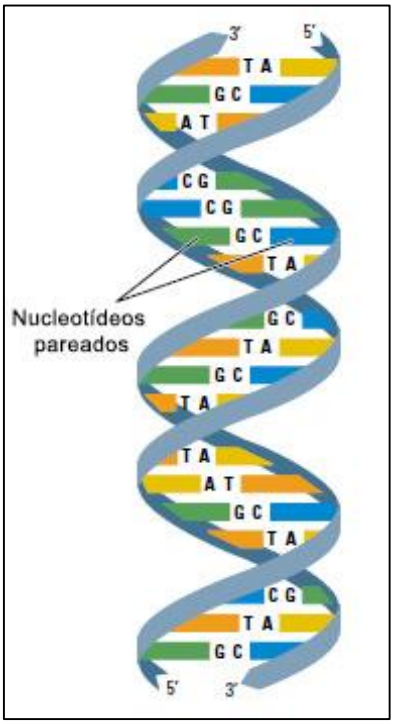
Cada espécie de organismo vivo possui um único conjunto de características herdadas que a torna diferente de outras espécies. Cada espécie tem seu próprio plano de desenvolvimento, geralmente descrito como um tipo de “diagrama” para a construção do organismo, que é codificado nas moléculas do DNA presente nas suas células (HARTL & JONES, 2004). Dentro dessas moléculas, numa concepção simplista podemos admitir que um gene é o segmento que contém um código para a produção dos aminoácidos da cadeia polipeptídica e as sequências reguladoras para a expressão.

2.1 ESTRUTURA DO DNA

No início da década de cinquenta, um número de pesquisadores começou a tentar entender a estrutura detalhada do DNA. Em 1953, James Watson e Francis Crick, da Universidade de Cambridge, propuseram a primeira estrutura tridimensional essencialmente correta de uma molécula de DNA. A estrutura era deslumbrante em sua elegância e revolucionária ao sugerir como o DNA duplica a si mesmo, controla os traços de hereditariedade, e passa por mutações. Na estrutura Watson-Crick, o DNA consiste de duas longas correntes de subunidades, cada uma disposta ao entorno da outra para formar uma dupla hélice do DNA, figura 1. A dupla hélice é destra, que significa que cada cadeia segue um caminho no sentido horário em seu progresso. As subunidades de cada fita são chamadas de nucleotídeos, onde cada um possui algum dos quatro constituintes químicos, chamados bases. As quatro bases no DNA são: adenina(A), timina(T), guanina(G) e citosina(C). As bases na dupla hélice são pareadas como é mostrado na figura 1. Em qualquer posição nas fitas pareadas de uma molécula de DNA, se uma fita contém um A, então a parceira contém um T, se uma fita contém um G, então a parceira possui um C. O pareamento entre A e T e entre G e C é dito como sendo complementar. O pareamento complementar significa que cada base ao longo da fita do DNA é encaixada em uma base na posição oposta na outra fita. Nada restringe a sequência de bases em uma única fita, então qualquer sequência pode aparecer ao longo de uma fita. Este

princípio explica como somente 4 bases de DNA podem codificar a imensa quantidade de informação necessária para criar um organismo (HARTL & JONES, 2004).

Figura 1 – Diagrama realçando as fitas helicoidais ao redor da molécula e os pares AT e GC na parte interior.



Fonte: HARTL e JONES (2004).

2.2 O GENOMA DAS BACTÉRIAS

Genes de bactérias são regiões contínuas de DNA. Entretanto, a unidade funcional de informação de uma sequência genética de uma bactéria é uma *string* de $3n$ nucleotídeos codificando uma *string* de n aminoácidos, ou uma string de n nucleotídeos codificando uma molécula estrutural RNA de n resíduos (LESK, 2008).

O genoma de uma típica bactéria é constituído por uma simples molécula de DNA, que, se estendida, teria em torno de 2mm. O DNA de organismos maiores é organizado em cromossomos, células de humanos normais contém 23 pares de cromossomos. A quantidade total de informação genética por célula, a sequência de nucleotídeos de DNA, é constante para todos os mesmos de uma espécie, mas varia imensamente entre espécies, como é mostrada na tabela 1 (LESK, 2008).

Tabela 1 – Organismos e tamanho de genoma

Organismo	Tamanho do genoma (pares base)
Vírus (*Epstein-Barr*)	$0{,}172 * 10^6$
Bactéria (*E. coli*)	$4{,}6 * 10^6$
Levedura (*S. cerevisiae*)	$12{,}1 * 10^6$
Nematódeo (*C. elegants*)	$95{,}5 * 10^6$
Planta (*A. thaliana*)	$117 * 10^6$
Inseto (*D. melanogaster*)	$180 * 10^6$
Humano (*Homo sapiens*)	$3200 * 10^6$

Fonte: LESK (2008).

A bactéria *E. coli* foi descoberta no cólon humano em 1885 pelo bacteriologista alemão Theodor Escherich. Escherich também mostrou que certas cepas da bactéria eram responsáveis por diarreia e gastroenterite, uma importante descoberta na saúde pública. Entretanto, a *E. coli* foi inicialmente chamada de *Escherichia coli* para honrar o seu descobridor, ela é comumente referenciada como o organismo vivo mais estudado.

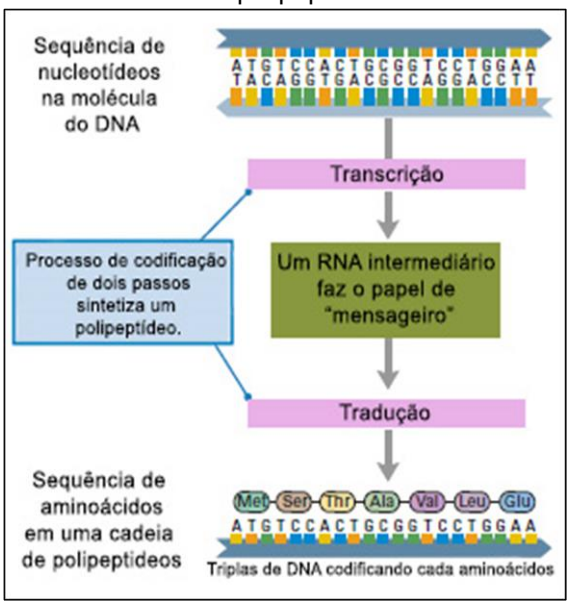
2.3 EXPRESSÃO GÊNICA

A expressão gênica é o processo em que a informação contida no gene é transformada em uma proteína, ou uma molécula de RNA. A expressão gênica inicia com a ligação de múltiplos fatores de proteína, conhecidos como fatores de transcrição. Fatores de transcrição regulam a expressão gênica ativando ou inibindo a transcrição. Entender os mecanismos que regulam a expressão gênica é um grande desafio na área da biologia. Identificar elementos regulatórios, os locais de ligação para fatores de transcrição, bem como o descobrimento de padrões nas sequências de DNA são um dos maiores e mais desafiantes problemas da biologia molecular e ciência da computação (DAS & DAI, 2007).

A expressão gênica é a sequência de bases ao longo do DNA que codifica a informação genética, e a sequência é completamente irrestrita. Watson e Crick estavam corretos ao propor que a informação genética no DNA está contida na sequência de bases de maneira análoga as letras impressas em uma fita de papel. Em uma região do DNA que direciona a síntese da proteína, o código genético para a proteína está contido somente em uma única fita, e é decodificado em uma ordem linear. Uma típica proteína é feita de uma ou mais cadeias de polipeptídios, cada cadeia consiste de uma sequência linear de aminoácidos conectados ponta a ponta. Existem mais de 20 aminoácidos diferentes. Apenas quatro bases de código para

esses 20 aminoácidos, onde cada “palavra” no código genético consiste de 3 bases adjacentes. Um exemplo de relacionamento entre a sequência de bases em um DNA e a sequência de aminoácidos da proteína correspondente é demonstrada na figura 2, onde a sequência de DNA especifica a sequência de aminoácidos através de uma molécula de RNA que serve como um "mensageiro" intermediário. Apesar do processo de decodificação ser indireto, o resultado é que cada aminoácido na cadeia de polipeptídios está especificado por um grupo de três bases adjacentes no DNA (HARTL & JONES, 2004).

Figura 2 – Sequência de DNA codificando para os primeiros sete aminoácidos em uma cadeia de polipeptídios



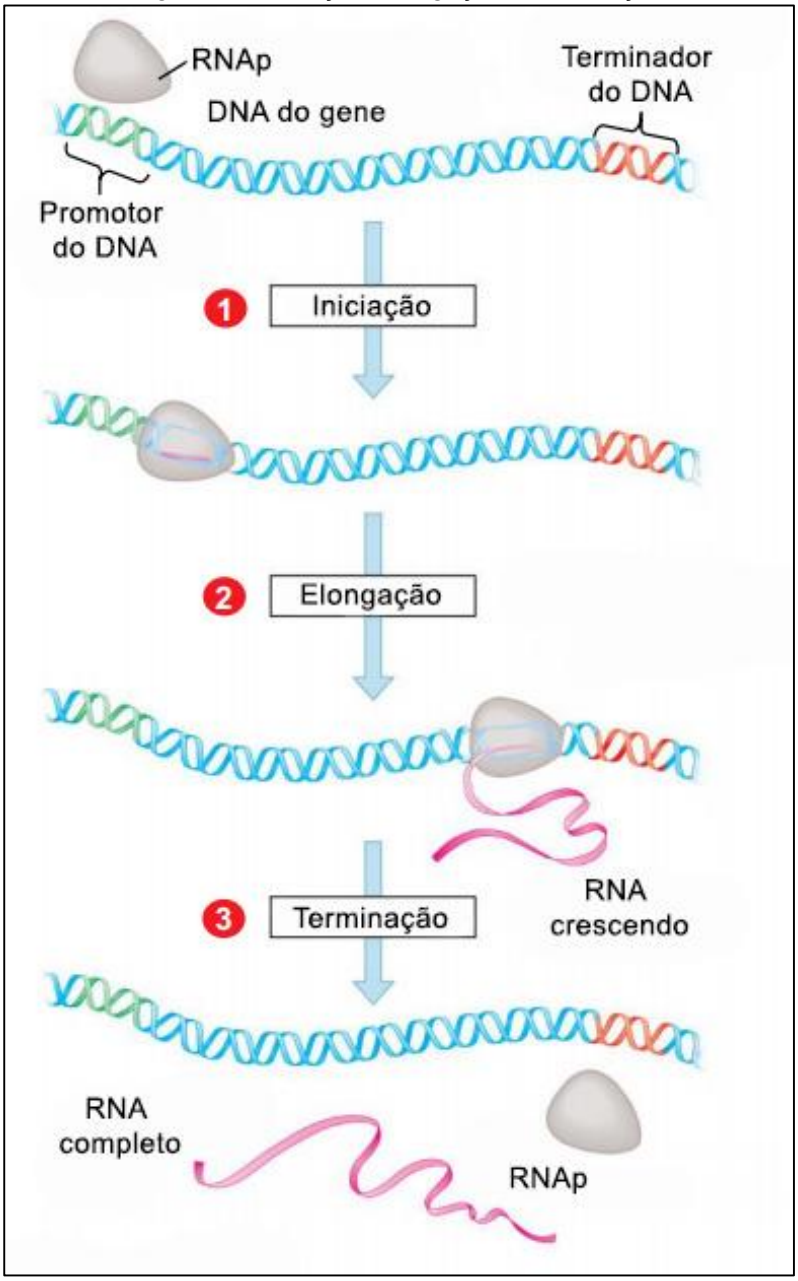
Fonte: HARTL e JONES (2004).

O esquema demonstrado na figura 2 indica que o DNA codifica proteínas não diretamente, mas indiretamente através do processo de transcrição e tradução. O processo de transformação do DNA para o ácido ribonucleico (RNA) é chamado de transcrição e a transformação do RNA para proteína é chamada de tradução. A rota indireta da transferência de informação do DNA para o RNA e do RNA para a proteína é conhecida como o dogma central da genética molecular. O conceito principal do dogma é que o DNA não codifica a proteína diretamente, mas age através de uma molécula intermediaria de RNA. A estrutura do RNA é similar, mas não é idêntica ao DNA (HARTL & JONES, 2004).

A sequência de nucleotídeos no DNA fornece o código para a construção de uma proteína, então a transcrição reescreve o código do DNA para o RNA, utilizando a mesma “linguagem”. O fluxo de informação do gene para a proteína é baseado em uma tripla de código: as instruções genéticas para a sequência de aminoácidos de uma cadeia de polipeptídios estão escritas no DNA e no RNA como séries de 3 palavras bases não sobrepostas chamadas de códons. A tradução envolve a troca da “linguagem” do nucleotídeo para a “linguagem” do aminoácido. Cada aminoácido é especificado por um códon. O código genético ditará como os códons serão traduzidos em aminoácidos. O código genético é redundante, com mais de um códon para alguns aminoácidos, ele não é ambíguo, onde qualquer códon para um aminoácido não codifica para nenhum outro aminoácido, é quase universal, pois são compartilhados por organismos simples como uma bactéria até os animais mais complexos e não possuem pontuação no sentido de que os códons são adjacentes aos outros sem intervalos entre eles (REECE *et al*., 2011).

A transcrição produz mensagens gênicas em formato de RNA, o funcionamento é ilustrado na figura 3. Uma molécula de RNA é transcrita de um padrão de DNA através de um processo que se assemelha a síntese de uma fita de DNA durante a replicação do DNA. Os nucleotídeos de RNA são interligados pela enzima da transcrição chamada RNA polimerase (RNAp). Sequências específicas de nucleotídeos ao longo do DNA marcam onde a transcrição inicia e termina. O sinal para início da transcrição é uma sequência de nucleotídeos chamada de promotora. A transcrição inicia quando a enzima RNAp se encaixa ao promotor. Durante a fase de alongamento, o RNA cresce mais. Quando o RNA se separa, as fitas de DNA se juntam novamente. Finalmente, na última fase, terminação, a enzima RNAp alcança a sequência de bases no padrão do DNA, chamada de terminador, que simboliza o final do gene (REECE *et al*., 2011).

Figura 3 – Iniciação, elongação e terminação



Fonte: REECE *et al*. (2011).

Os fatores sigma (σ) são reguladores transcricionais que compreendem uma classe de proteínas dissociáveis do cerne da RNA polimerase (RNAp) bacteriana. Cada um dos vários fatores sigma é requerido para a transcrição direta ou indireta de um subconjunto específico de genes, denominados *regulon* (MOONEY *et al*., 2005; CASES & DE LORENZO, 2005). Os fatores sigma podem ser classificados em duas famílias; σ70, devido a sua similaridade com o fator sigma primário de 70 kDa de *Escherichia* coli, e σ54, devido a sua similaridade com o fator sigma de 54 kDa, responsável pela regulação do metabolismo de nitrogênio em *E. coli.* (SACHDEVA *et al*., 2009; KAZMIERCZAK *et al*., 2005).

2.4 PROPRIEDADES ESTRUTURAIS DAS REGIÕES PROMOTORAS

Assume-se que a coexpressão de genes inicia principalmente a partir da co-regulação transcricional. Como genes co-regulados são conhecidos por compartilhar similaridades no seu mecanismo regulatório, possivelmente em um nível transcricional, suas regiões promotoras podem conter motivos comuns que são locais de ligação para fatores de transcrição (OLIVARES-ZAVALETA *et al*., 2006). De acordo com diversos autores, não somente as sequências regulatórias contém elementos específicos de sequência que servem como alvos para proteínas interativas, mas também apresentam diferentes propriedades, tais como: curvatura, flexibilidade e propriedades físicas (por exemplo, energia empilhada, estabilidade e estabilização). Diversos estudos têm reportado que sequências promotoras eucariontes [1] e procariontes[2] possuem uma estabilidade mais baixa, alta curvatura e flexibilidade mais reduzida que as sequências de codificação (GABRIELIAN & BOLSHOY, 1999; KANNHERE & BANSAL, 2005a; DAS & DAI, 2007; SILVA & ECHEVERRIGARAY, 2011)

A estabilidade do DNA é uma propriedade sequência-dependente baseada na soma das interações entre os dinucleotídios de uma dada sequência. É possível calcular a estabilidade da dupla fita do DNA e prever o comportamento da fusão se a contribuição de cada interação com vizinho mais próximo é conhecida (SANTALUCIA & HICKS, 2004; SILVA & ECHEVERRIGARAY, 2011). A curvatura do DNA tem mostrado ser importante como base física em diversos processos biológicos, em particular naqueles que possuem interação do DNA com as proteínas que se ligam ao DNA, tal como a iniciação e a terminação (GABRIELIAN & BOLSHOW, 1999; JAUREGUI *et al*., 2003; NICKERSON & ACHBERGER, 1995; THIYAGARAJAN *et al*., 2006). A curvatura do DNA nos procariontes está usualmente presente antes da

[1] A maioria dos animais e plantas são dotados deste tipo de sequências.

[2] A este grupo pertencem seres unicelulares ou coloniais, tais como bactérias e cianofitas (algas cianofíceas, algas azuis ou ainda Cyanobacteria).

sequência do promotor, mas algumas vezes está dentro da mesma (JAUREGUI *et al*., 2003; KOSOBAY-AVRAHAM *et al*.,2006; SILVA & ECHEVERRIGARAY, 2011).

2.5 BIOINFORMÁTICA

O sequenciamento do DNA é efetuado através de variados métodos desenvolvidos nas últimas décadas. Os equipamentos sequenciadores de DNA são capazes de ler uma amostra de DNA e gerar um arquivo eletrônico com símbolos que representam a sequência das bases nitrogenadas do DNA, A, C, G, T, contidas na amostra. O formato de arquivo mais conhecido e utilizado para representação do sequenciamento do DNA é o FASTA, desenvolvido por David J. Lipman e William R. Pearson em 1985. Maiores detalhes sobre a formatação e funcionamento do FASTA são encontrados em PEARSON *et al*. (1988).

Em consequência da grande quantidade de informações de sequências de nucleotídeos e de aminoácidos que são produzidas atualmente, o uso dos bancos de dados vem assumindo uma importância crescente na bioinformática. Um banco de dados pode ser considerado uma coleção de dados inter-relacionados, projetado para suprir as necessidades de um grupo específico de aplicações e usuários. Os bancos de dados envolvendo sequências de nucleotídeos, de aminoácidos, ou estruturas de proteínas podem ser classificados em bancos de sequências primários e secundários. Os primários são formados pela deposição direta de sequências de nucleotídeos, aminoácidos ou estruturas proteicas, sem qualquer processamento ou análise. Os principais bancos de dados primários são o *GenBank*, o Instituto Europeu de Bioinformática(EBI), o Banco de dados de DNA do Japão (DDBJ) e o Banco de dados de Proteínas (PDB). Eles trocam informações entre si diariamente, de modo que todos os três possuem informações atualizadas de todas as sequências de DNA depositadas em todo o mundo. Os bancos de dados secundários, como a Base de informação de proteína (PIR) e o SWISSPROT, são aqueles que derivam dos primários, ou seja, foram formados usando as informações depositadas nos bancos primários (PROSDOCIMI, 2002). Existem também bancos de dados que carregam informações sobre um único organismo, como é o caso do *RegulonDB*, especializado no mapeamento do genoma da bactéria *E.Coli*.

2.6 LOCALIZADORES DE MOTIVOS

A partir do sequenciamento do DNA, o desafio do descobrimento e identificação de motivos presentes nas sequências do DNA incentivou o desenvolvimento de diversas metodologias e algoritmos de domínio da bioinformática. Um motivo é um padrão de sequência de nucleotídeos ou aminoácidos que é generalizado e tem algum significado biológico. O mesmo é definido como sendo um local de ligação do DNA para uma proteína regulatória, ou seja, um fator de transcrição. As sequências podem conter zero, um, ou diversas cópias de um motivo (DAS & DAI, 2007).

Os métodos disponíveis atualmente podem ser classificados em duas classes principais: probabilística e combinatorial. Métodos probabilísticos tem a vantagem de requerer poucos parâmetros de pesquisa, mas dependem de modelos probabilísticos das regiões regulatórias, que podem ser muito sensíveis as pequenas mudanças nos dados de entrada. Os mais populares métodos probabilísticos incluem abordagens de EM (*Expectation-Maximization*) (SEGAL & SHARAN, 2005), *Projection* (BUHLER & TOMPA, 2002) e MEME (BAILEY & ELKAN, 1994). Estes métodos utilizam um procedimento iterativo de duas fases, onde no primeiro passo as ocorrências mais comuns dos motivos são identificadas, baseadas em um modelo computado na iteração anterior. O segundo passo ajusta o modelo para o motivo, que normalmente é referenciado como uma matriz de pesos, baseado nas ocorrências determinadas no passo anterior. Na primeira iteração os parâmetros do modelo inicial são geralmente inicializados randomicamente. A maior desvantagem destes algoritmos é a sua sensibilidade a ruídos nos dados e o fato de que eles não são garantidos para convergir a um máximo global. A maioria deles assume que somente haverá um motivo ocorrendo nas sequências de entrada e na maioria, um a cada sequência.

Métodos combinatórios, que tipicamente extraem motivos consistindo de sequências simples de nucleotídeos, normalmente envolvem a enumeração de todos os possíveis padrões, explicitamente ou implicitamente. Considerando um conjunto de sequências $S = \{S1, S2, \dots, St\}$. Se faz necessário encontrar motivos em um alcance de tamanhos $l_{min}, \dots, l_{max}$, que ocorrem em $q \leq t$ das sequências com, no máximo e posições nas quais elas diferem entre si. Métodos combinatórios utilizam uma ou duas possibilidades de abordagem. A primeira abordagem enumera todos os possíveis padrões de um tamanho fixo l, e verifica sua ocorrência nas sequências de entradas

com no máximo e diferenças. A segunda abordagem pega cada elemento de tamanho l ocorrendo na sequência de entrada e gera sua vizinhança de diferenças e. Apesar dos métodos combinatórios de motivo serem capazes de encontrar todos os motivos que se encaixem nas especificações, eles possuem a desvantagem de não serem bons em discriminar os motivos relevantes extraídos dos numerosos potenciais falsos motivos. O problema de determinar qual porção da saída corresponde a um resultado biológico significante tem sido endereçada na sua maioria ao uso de técnicas estatísticas e raciocínio biológico. Uma característica chave nos modernos localizadores de motivos é a habilidade de extrair motivos complexos, por exemplo, motivos com intervalos, também conhecidos como motivos estruturados. Em primeiro lugar, motivos complexos podem ser modelos melhores para regiões promotoras. Em segundo lugar, muitos autores concordam que motivos compostos podem ser muito fracos para serem extraídos em isolação, pois podem ser pouco distinguíveis do ruído presente ao seu redor nas sequências. Entretanto, impondo uma distância específica entre motivos compostos, um padrão incomum pode ser identificado (MENDES *et al*., 2006).

2.7 CONSIDERAÇÕES FINAIS

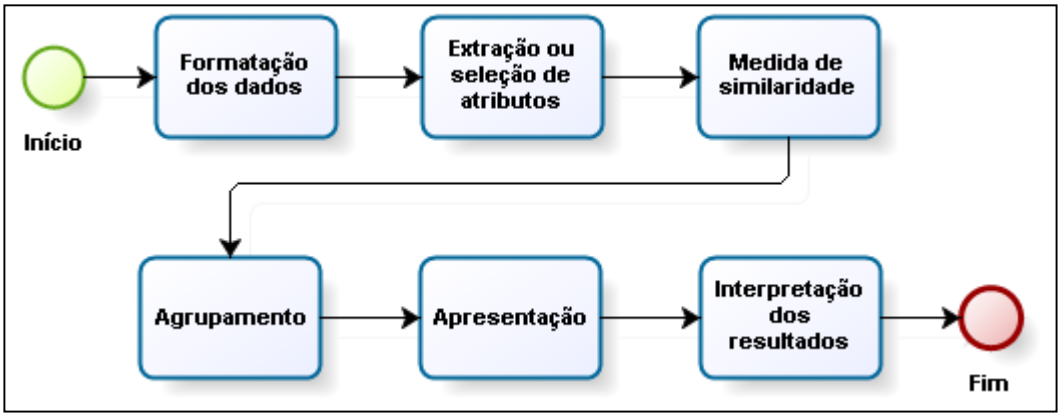
Tendo em vista os conceitos apresentados neste capítulo, pode-se concluir que a sequência genômica é um código altamente complexo, contendo a informação para construir e manter um organismo funcional. Os principais fundamentos e conceitos que serão utilizados no desenvolvimento deste trabalho foram demonstrados. Nos próximos capítulos serão abordadas algumas das técnicas que a bioinformática propõe para aprimorar e facilitar a complexa tarefa de compreender e interpretar o genoma de qualquer organismo vivo.

3 CLUSTERIZAÇÃO E TRANSFORMAÇÃO DE ATRIBUTOS

A mineração de dados é o processo de descoberta de informações acionáveis em grandes conjuntos de dados. A mineração de dados usa análise matemática para derivar padrões e tendências que existem nos dados. Normalmente, esses padrões não podem ser descobertos com a exploração de dados tradicional pelo fato de as relações serem muito complexas ou por haver muitos dados (MITRA *et al*., 2003).

A clusterização é uma metodologia da área de mineração de dados, cuja principal função é o agrupamento de objetos de dados em um conjunto de classes distintas, chamadas *clusters*, para que os objetos de uma mesma classe tenham grande similaridades entre eles, enquanto os objetos em classes separadas são mais dissimilares. Clusterização é um exemplo de classificação não supervisionada, onde o termo classificação, refere-se ao procedimento que atribui objetos de dados para um conjunto de classes e o termo não supervisionado, significa que a clusterização não depende de classes predefinidas e exemplos treinados enquanto classificam-se os objetos de dados. Entretanto, clusterização é diferente de reconhecimento de padrão, ou das áreas de estatísticas conhecidas como análise discriminativa e análise de decisão, que visam encontrar regras para classificar objetos de um determinado conjuntos de objetos pré-classificados (JIANG *et al*., 2004). A figura 4 representa as etapas de um processo de clusterização.

Figura 4 – Etapas de um processo de clusterização



Fonte: O Autor (2015).

O processo de clusterização normalmente envolve os seguintes estágios:

- Formatação dos dados: envolve definição do número, tipo e modo de apresentação dos atributos que descrevem a natureza dos dados;

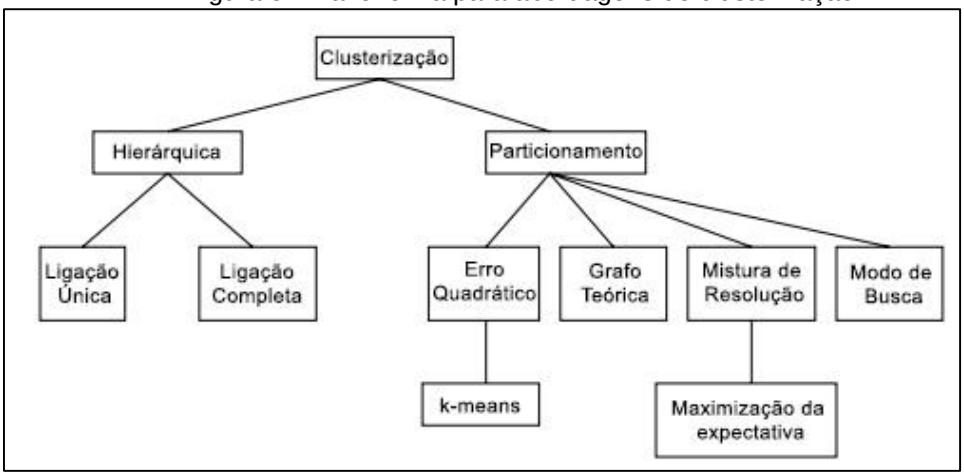
- Seleção ou extração de atributos: a seleção de atributos, ou características, é o processo de identificação do subconjunto mais efetivo dos atributos disponíveis para descrever cada objeto, enquanto a extração de atributos é a utilização de uma ou mais transformações junto aos atributos de entrada de modo a salientar uma ou mais característica dentre aquelas que estão presentes nos dados. Ambos os processos serão abordados detalhadamente nas seções 3.4 e 3.5.
- Medida de similaridade: envolve o cálculo da medida de similaridade, de um modo geral, uma medida de similaridade mede o grau em que um par de objetos são semelhantes. Cada métrica de distância quantifica uma noção diferente do que significa para dois vetores estarem próximos um ao outro. Existem diversas características que podem ser avaliadas para calcular se dois vetores estão próximos. As principais são a correlação, a distância euclidiana e o ângulo cosseno.
- Agrupamento: os objetos são atribuídos aos seus respectivos grupos(*clusters*). Os grupos podem ser definidos como conjuntos *crisp*, quando um padrão pertence ou não pertence a um determinado grupo, ou *fuzzy*, quando um padrão pode apresentar graus de pertinência aos grupos. O processo de agrupamento pode ser hierárquico, ou não-hierárquico.
- Apresentação: resultado final da clusterização, onde os dados obtidos no processo são apresentados em um formato interpretável pelo usuário, seja em arquivos de texto, ou gráficos.
- Interpretação dos resultados: os resultados são analisados e são elaborados comparativos com diversas configurações, normalmente por um especialista na área de conhecimento em que processo foi executado (JAIN *et al*., 1999).

3.1 ALGORITMOS DE CLUSTERIZAÇÃO

A maioria dos algoritmos de clusterização atribuem objetos a *clusters* baseado na similaridade de seus padrões de atividade. Objetos com atividades similares são agrupados juntamente, enquanto objetos com diferentes deveriam ser distribuídos em diferentes *clusters*. Os algoritmos de clusterização podem ser divididos inicialmente

em métodos hierárquicos e particionais, uma taxonomia das abordagens mais utilizadas é demonstrada na figura 5.

Figura 5 – Taxonomia para abordagens de clusterização

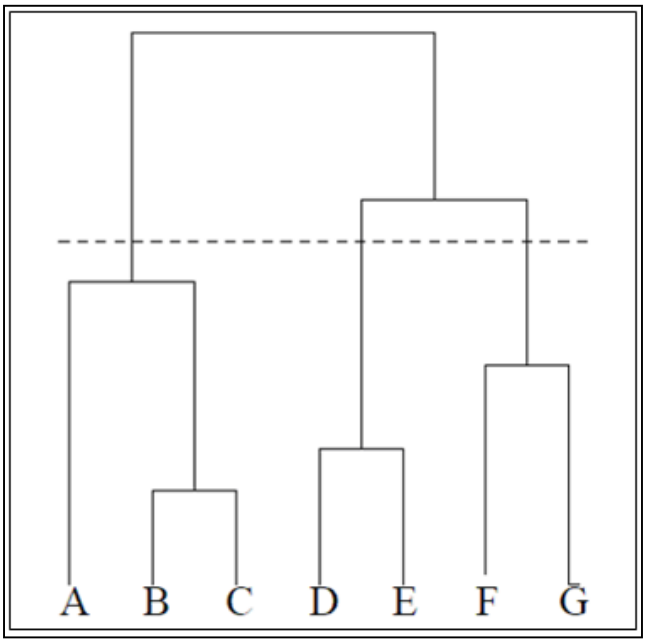


Fonte: JAIN *et al*. (1999).

3.1.1 Clusterização hierárquica

A clusterização hierárquica constrói uma hierarquia de *clusters*, ou, em outras palavras, uma árvore de *clusters*. Cada nodo de *cluster* contem *clusters* filhos, *clusters* irmãos particionam os pontos cobertos pelos seus pais comuns. Esta abordagem permite a exploração dos dados em diferentes níveis de granularidade (JAIN & DUBES 1988; KAUFMAN & ROUSSEEUW 1990). As soluções de clusterização hierárquica são comumente representadas através de um dendograma, vide figura 6. Algoritmos para gerar tais soluções normalmente trabalham tanto de uma maneira *top-down* (de cima para baixo), particionando repetidamente um conjunto de elementos, quanto *bottom-up* (de baixo para cima). Os algoritmos hierárquicos iniciam de uma partição inicial até atingir um *cluster* único, através de sucessivas fusões entre *clusters*, até que todos os elementos pertençam ao mesmo *cluster*.

Figura 6 - Representação em dendograma da clusterização hierárquica



Fonte: JAIN *et al.* (1999).

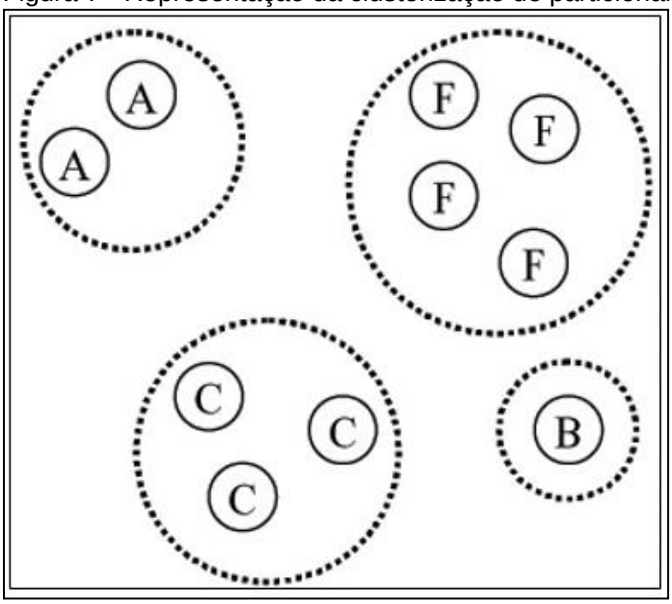
As principais vantagens da clusterização hierárquica incluem a alta flexibilidade, devido ao nível de granularidade, a facilidade de lidar com qualquer forma de similaridade ou distância, e consequentemente, a aplicabilidade para qualquer tipo de atributo. As principais desvantagens são a imprecisão de um critério de terminação e o fato de que a maioria dos algoritmos hierárquicos não revisitam os *clusters* construídos com o propósito de melhorá-los (BERKHIN, 2002).

3.1.2 Clusterização de particionamento

A clusterização baseada no particionamento divide os dados em diversos subconjuntos. A tarefa de checar todos os subconjuntos possíveis é computacionalmente inviável, portanto certas heurísticas gulosas são utilizadas no formato de iteração para otimização, ou seja, diferentes esquemas de realocação iterativamente atribuem os pontos entres os k *clusters*, sendo k o número total de *clusters*. Ao contrário dos métodos hierárquicos, onde os *clusters* não são revisitados após a sua construção, os algoritmos de realocação gradualmente melhoram os *clusters*. Uma abordagem para o particionamento dos dados é tomar um ponto de vista conceitual que identifica o *cluster* com um certo modelo cujos parâmetros desconhecidos precisam ser encontrados, mais especificamente, modelos probabilísticos assumem que os dados são originados de uma mistura de diversas populações cujas distribuições precisam ser descobertas. Uma vantagem desta abordagem é a facilidade para a interpretação dos *clusters* construídos. Outra

abordagem inicia com a definição da função objetivo dependendo de uma partição. A distância ou a similaridade entre pares podem ser usadas para computar medidas de relações de dentro dos *clusters* e entre eles. Nos melhoramentos iterativos, a computação das medidas entre pares seria muito alta, uma opção para reduzir o custo de computação é a utilização de representantes únicos de *clusters*, tornando a computação da função objetivo em linear, conforme o número de *clusters*.

Figura 7 - Representação da clusterização de particionamento.

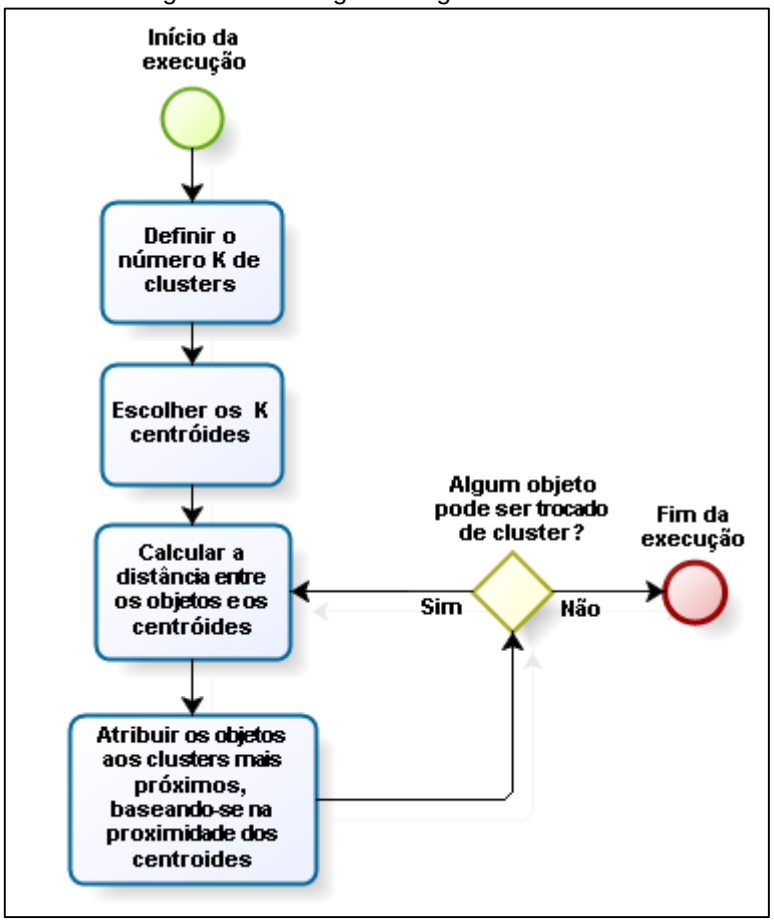


Fonte: JAIN *et al.* (1999).

Os algoritmos de iteração e otimização do particionamento são subdivididos nos métodos *k-medoids* e *k-means*. Os métodos *k-medoids* não apresentam limitação nos tipos de atributos e a escolha dos *medoids* (objetos representativos utilizados em conjuntos de dados cuja média de dissimilaridade para todos os objetos é mínima) é determinada pela localização de uma fração predominante de pontos dentro de um *cluster* e, por este motivo, é menos sensível a presença de objetos que são raras exceções, conhecidos como *outliers*. No método *k-means* um *cluster* é representado pelo seu centroide, que é uma média para os pontos em um *cluster*. No *k-means*, deve ser selecionado o número k de *clusters* inicialmente e cada objeto é atribuído para um *cluster*, ele inicia com uma partição inicial e através de iterações, os objetos similares vão sendo agrupados dentro de um mesmo *cluster*. O cálculo da medida de similaridade mais frequentemente utilizado é o euclidiano. Um dos problemas do *k-means* é que o sucesso do seu agrupamento está diretamente relacionado a escolha do melhor k, ou seja, dos centroides. Por este motivo ele não possuí uma única solução, a clusterização depende da partição inicial, além de ser bastante sensível a

ruídos e *outliers*. A figura 6 demonstra as etapas do algoritmo *k-means* no formato de fluxograma.

Figura 8 – Fluxograma algoritmo *k-means*



Fonte: O autor (2015).

Além das abordagens demonstradas anteriormente, diversas outras técnicas já foram desenvolvidas, bem como o algoritmo de particionamento sobre *medoids* (PAM), que funciona como uma extensão do *K-means*, melhorando a eficiência no tratamento dos *outliers*, e o CURE (*Clustering using representatives*), para a abordagem hierárquica. Uma descrição detalhada do funcionamento das abordagens apresentadas na figura 5 pode ser encontrada no trabalho de ANDRITSOS (2002), que traz uma pesquisa sobre as técnicas de clusterização existentes.

3.2 CLUSTERIZAÇÃO APLICADA NA BIOINFORMÁTICA

Técnicas de clusterização têm provado serem úteis para entender a função do gene, a regulação, os processos celulares, e subtipos de células. Essa abordagem fornece entendimento das funções de muitos genes os quais a informação não tem estado disponível previamente. Genes coexpressos no mesmo *cluster* geralmente

estão envolvidos nos mesmos processos celulares, e uma grande correlação de padrões de expressão entre estes genes indicam co-regulação. Procurar por sequências de DNA comuns nas regiões promotoras dos genes do mesmo *cluster* permite a localização de motivos regulatórios específicos para cada *cluster* e possivelmente propor elementos que regulam a transcrição (JIANG *et al*., 2004).

Diversas técnicas de algoritmos foram previamente utilizadas na clusterização dos dados das expressões gênicas, incluindo clusterização de particionamento e hierárquica, mapas auto organizados (SOM – *Self Organized Maps*), e abordagens que utilizam a teoria dos grafos. SHAMIR e SHARAN (2001) desenvolveram um trabalho onde são demonstradas abordagens de clusterização que já foram aplicadas na análise da expressão gênica.

No trabalho de TAMAYO *et al*. (1999), foi utilizada a técnica de mapas auto organizados para a interpretação de padrões na expressão gênica. SOMs impõem estrutura aos dados, com os nodos vizinhos tendendo na definição dos *clusters* relacionados. Um SOM baseado em uma grade retangular é análogo a uma gaveta de espécies de um entomologista, com compartimentos adjacentes guardando insetos similares. Estruturas alternativas podem ser impostas nos dados através de diferentes geometrias iniciais, como grades, anéis e linha, com diferentes números de nodos. O sucesso da metodologia SOM na identificação de padrões predominantes na expressão gênica neste sistema de modelos sugere que a análise da expressão gênica junto com as ferramentas computacionais apropriadas, provê uma valorosa compreensão dos processos biológicos que não podem ser entendidos no nível molecular (TAMAYO *et al*., 1999).

KRISHNA *et al*. (1999), propuseram uma adaptação do algoritmo *k-means*, o algoritmo genético *k-means* (GKA – *Genetic k-means algorithm*), que mistura características da natureza de algoritmos genéticos com a alta performance do *k-means*. Como resultado, o GKA sempre irá convergir para um ótimo global mais rápido do que outros algoritmos. LU *et al*. (2004), propuseram três algoritmos baseados no GKA, o algoritmo genético *k-means* rápido (FGKA – *Fast Genetic K-means Algorithm*), o algoritmo genético *k-means* incremental (IGKA - *Incremental Genetic K-means Algorithm*) e o algoritmo genético *k-means* híbrido (HGKA - *Hibrid Genetic K-means Algorithm*) e aplicaram eles na análise da expressão gênica, mais especificamente,

agrupando os genes de diferentes categorias funcionais e identificando o número de sequências de leitura aberta (ORF – *Open Reading Frame*) em cada categoria. Como resultado, o algoritmo *k-means* foi o mais rápido na sua execução, porém sofre com a convergência para um ótimo local e a dependência da inicialização, os algoritmos FGKA, o IGKA e o HGKA, mostraram ser mais lentos, porém foram encontrados melhores resultados no agrupamento dos genes, principalmente devido a convergência para um ótimo global e a independência da inicialização. Estas adaptações do algoritmo *k-means* podem ser objetos de estudos em trabalhos futuros, pois apresentaram resultados significativos sendo aplicadas em objetivos similares ao deste trabalho. Nas próximas seções, ainda serão abordados mais alguns trabalhos que envolvem clusterização aplicada na análise genômica, com enfoque nas etapas de extração e seleção de atributos.

3.3 FORMATOS DE REPRESENTAÇÃO DOS DADOS DO SEQUENCIAMENTO DO DNA

Neste trabalho serão analisadas as representações em linguagem natural e em séries temporais. Inicialmente, o resultado do sequenciamento do DNA é uma representação no formato da linguagem natural, vide seção 2.6, após este processo, ferramentas fazem a conversão para o formato de séries temporais, este último possibilita diferentes análises e abordagens sobre as propriedades estruturais do DNA.

3.3.1 Linguagem natural

A linguagem natural é conhecida por dificultar o entendimento pela máquina, essa situação propõe um desafio significativo para a análise das funções gênicas. Por exemplo, dada uma nova biblioteca de sequência, constantemente é desejável que sejam identificados todos os genes associados com alguma função biológica específica, ou para rapidamente examinar sua distribuição de acordo com categorias predefinidas. É conhecido que a mensuração do padrão de proximidade tem um papel crucial na pesquisa de inteligência computacional, principalmente nos campos de recuperação de informação, reconhecimento de padrões e análise de *cluster*. A avaliação da correlação entre dois termos biológicos e suas anotações funcionais primeiramente afetam a performance da seleção automática de genes e processos de

categorização (HE, 2008). Um exemplo deste formato de representação é demonstrado na figura 9.

Figura 9 - Exemplo de arquivo de promotor

```
# Colunas:
# (1) Identificador do promotor atribuído por RegulonDB
# (2) Nome do promotor
# (3) Fita do DNA onde o promotor está localizado
# (4) Posição no genoma do local de início da transcrição
# (5) Fator sigma que reconhece o promotor
# (6) Sequência do promotor
# (7) Evidência que suporta a existência do promotor
ECK125153230    TSS_1   forward 38  unknown aaacgccttagtaagtatttttcagcttttcattctgactgcaacgggcaatatgtctctGtgtggattaaaaaaagagtg
ECK125153239    TSS_10  forward 2811    unknown tgtctttgctgatctgctacgtaccctctcatggaagttaggagtctgacatggttaaagTttatgccccggcttccagtg
ECK125153329    TSS_100 forward 57261   unknown ctgtccgaatagcgtcagtggtgttaggcacggcattgaatgacaggtatgataatgcaaAttataggcgatgtcccacaa
ECK125154229    TSS_1000    reverse 951381  unknown gcaatcaacggcggcgttgacgaaaaactgaaaatgcaggttggtccgaagtctgaaccgAtcaaaggcgatgtcctgaac
ECK125154230    TSS_1001    reverse 951386  unknown tgtacgcaatcaacggcggcgttgacgaaaaactgaaaatgcaggttggtccgaagtctgAaccgatcaaaggcgatgtcc
ECK125154231    TSS_1002    reverse 951390  unknown atgctgtacgcaatcaacggcggcgttgacgaaaaactgaaaatgcaggttggtccgaagTctgaaccgatcaaaggcgat
```

Fonte: RegulonDB (2015).

3.3.2 Séries temporais

Dados representados em séries temporais consistem de uma sequência de N pares $(y_i, t_i), i = 1, \dots, N$, onde y_i é o valor medido de uma quantidade no tempo t_i. Realizar experimentos com séries temporais é um método popular para estudar uma grande quantidade de sistemas biológicos. Algoritmos especificamente desenvolvidos para séries temporais são necessários para que se possa obter vantagem de suas características únicas, como a habilidade de inferir casualmente a partir da resposta do padrão temporal, e endereçar os únicos problemas que eles levantam, por exemplo, lidar com taxas não uniformes (BAR-JOSEPH, 2004).

A expressão gênica é um processo temporal, onde diferentes proteínas são requeridas e sintetizadas para diferentes funções sobre diferentes condições. Para determinar o conjunto completo de genes que são expressos sob essas condições, e para determinar a interação entre os genes, é necessário que seja medido o curso do tempo dos experimentos da expressão. Isso nos permite determinar não somente o estado estável seguindo a nova condição, mas também o caminho e as redes que foram ativadas em ordem para chegar neste novo estado. Neste trabalho, as séries temporais serão utilizadas para representar a propriedades estruturais das regiões promotoras do DNA, especificamente a curvatura e a estabilidade. A tabela 2 demonstra o formato de representação de uma série temporal (BAR-JOSEPH, 2004).

Tabela 2 – Exemplo de representação da sequência do DNA em séries temporais

Identificador do Gene	Sequência do gene
ECK120001251	-1,362667;-1,353467;-1,336533;-1,314133;-1,2956;-1,287067;-1,2868;...;-1,3812;-1,4132;-1,42386
ECK120000987	-1,382;-1,370;-1,370;-1,342;-1,277;-1,227;-1,224;-1,253;-1,289;-1,332;...;-1,236;-1,232;-1,206;-1,226;-1,316

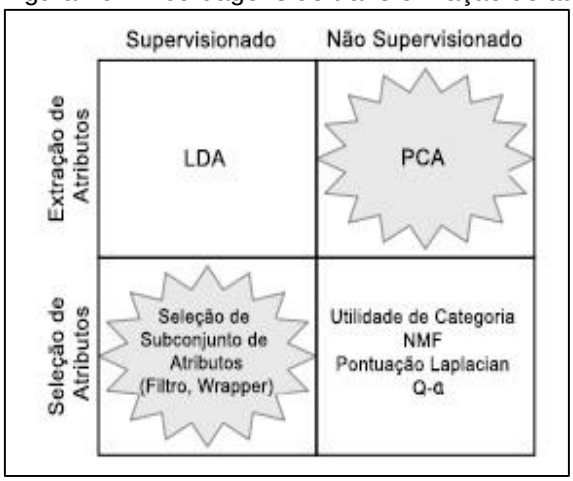
Fonte: O autor (2015).

No capítulo 2, concluímos que os dados advindos do sequenciamento do DNA são complexos e bastante ruidosos. Para contornar este problema e atingir resultados satisfatórios no processo de clusterização, surge a necessidade de metodologias que auxiliem na transformação destes dados em atributos importantes e que realmente exerçam impactos significativos nas funções biológicas.

O termo transformação de atributos refere-se a uma família de técnicas de pré-processamento de dados que transformam os atributos originais de um conjunto de dados em um alternativo, e mais compacto, conjunto de dimensões, tentando conter a maior quantidade de informação possível. Essas técnicas podem ser subdivididas em extração de atributos e seleção de atributos. A principal diferença entre os métodos de extração e seleção de atributos é que o primeiro se caracteriza quando é aplicada algum tipo de transformação nos dados, a fim de projetá-los em um novo espaço de atributos, com dimensões menores. O segundo método escolhe atributos do conjunto original baseando-se em algum critério, por exemplo, a correlação, ou seja, são utilizados critérios para filtrar atributos redundantes, ou menos importantes.

A figura 10 representa um esquema das técnicas de transformação de atributos. As duas principais distinções da área são as abordagens supervisionadas e não supervisionadas. As técnicas mais utilizadas são a seleção de subconjunto de atributos e análise de componentes principais (PCA – *Principal Component Analysis*). As outras técnicas demonstradas na figura, como a análise discriminante linear (LDA - *Linear Discriminant Analysis*) e a fatoração não negativa de matriz (NMF - *Non-negative Matrix Factorisation*), não serão contempladas neste trabalho. Maiores informações sobre estas e outras técnicas podem ser encontradas no trabalho de CUNNINGHAM (2007).

Figura 10 – Abordagens de transformação de atributos



Fonte: CUNNINGHAM (2007).

3.4 SELEÇÃO DE ATRIBUTOS

Uma funcionalidade, um atributo, ou uma variável referem-se a um aspecto do dado. Atributos podem ser discretos, contínuos, ou nominais. Seleção de atributos é uma técnica para escolher um subconjunto de atributos relevantes, entre todos os atributos, que mostraram a melhor pontuação de classificação. O propósito da seleção de atributos é determinar quais atributos são os mais relevantes para a tarefa atual de classificação, este processo consiste de quatro componentes, nomeados, geração de subconjunto de atributos, avaliação de subconjuntos, critério de parada e avaliação de resultado e envolve a descoberta de informações faltantes entre as características no conjunto de dados original e a construção de características adicionais que dão ênfase para as novas informações descobertas (COSTA *et al*., 2004).

A necessidade da seleção de atributos é bem documentada na literatura de máquinas de aprendizagem, onde a performance dos algoritmos de classificação degrada rapidamente se atributos irrelevantes são selecionados, e até mesmo se atributos relevantes forem selecionados, no caso deles estarem correlacionados com os atributos atuais. Essa degradação é ainda mais drástica em cenários realísticos, onde a distribuição subjacente não é conhecida, e o algoritmo de classificação necessita estimar os parâmetros a partir dos dados (DOUGHERTY *et al*., 2005).

Diferentes abordagens usadas na seleção de atributos podem ser agrupadas em duas categorias: abordagem de filtro e abordagem *wrapper*. A abordagem de filtro é independente de classificador, enquanto a *wrapper* é dependente. Os métodos *wrapper* tentam otimizar a performance de um classificador principal e selecionar os

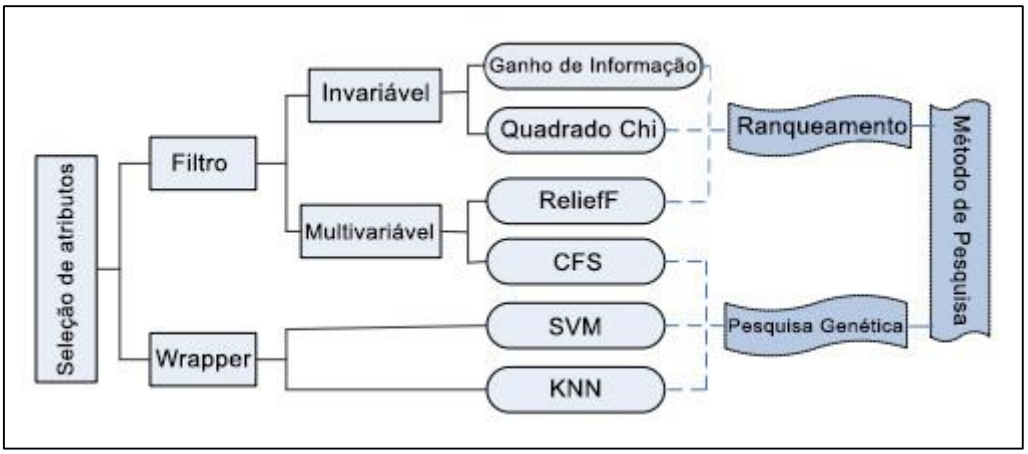
atributos produzindo a melhor performance de generalização. Na generalização, recentemente, algumas novas técnicas baseadas em maximizar a incerteza ou combinar múltiplos redutos de conjuntos irregulares têm sido propostas para melhorar a generalização do classificador (ALOK *et al*., 2012).

Um método *wrapper* depende de um classificador adicional para identificar o conjunto de características ótimas. O processo de seleção de atributos consiste em uma pesquisa no espaço de características guiado pela performance do classificador adicional. Apesar de trazerem uma boa pontuação para o classificador principal, métodos *wrapper* são computacionalmente custosos. Métodos de filtro são independentes de qualquer classificador adicional. Estes métodos selecionam as características olhando somente para as propriedades intrínsecas dos dados, como a separabilidade de classe, ou correlação entre características e classes. Primeiramente, uma pontuação de relevância da característica é calculada e as características com menos pontuação são filtradas. Após, as características selecionadas são apresentadas como dados de entrada para o algoritmo de classificação.

GAN *et al*. (2012) desenvolveram um comparativo entre as metodologias de seleção de atributos apresentadas na figura 11, com o intuito de atingir melhores resultados na predição de promotores. Para os métodos de filtro, quatro diferentes critérios de avaliação foram adotados para avaliar as características, atribuindo uma pontuação para cada característica que sugere quão valiosa a característica é para a classificação. Os quatro critérios de avaliação incluem dois critérios invariáveis: ganho de informação (IG – *Information Gain*) e *Chi Square* (CHI), e dois critérios multivariáveis, *RelieF* e seleção de características baseada em correlação (CFS – *Correlation-based Feature Selection*). Todas as características foram ranqueadas pelas suas pontuações e as mais relevantes foram obtidas de acordo com um critério de parada. Os subconjuntos de características selecionados por diferentes métodos de filtro foram avaliados por performance de classificação. O cálculo dos coeficientes de correlação entre as características é custoso e é ainda mais custoso ranquear cada subconjunto de características, então os autores adotaram a estratégia de pesquisa genética para CFS, que é diferente dos outros critérios. Para métodos *wrapper*, o classificador alvo e a estratégia de pesquisa são elementos críticos. Os autores utilizaram duas técnicas de reconhecimento de padrões, máquinas de vetores de

suporte (SVM – *Support vector machine*) e *k-Nearest Neighbors* (kNN) como classificadores alvos, utilizando a pontuação da classificação para medir a qualidade das características selecionadas. Os resultados deste trabalho demonstraram que as propriedades estruturais estão especialmente correlacionadas com os promotores. Especificamente a rigidez da ligação, a desnaturação e a energia livre da fita dupla do DNA estão altamente correlacionadas com os promotores. O poder de predição de sequências de promotores se diferencia vastamente entre diferentes propriedades estruturais. Selecionar as características relevantes pode melhorar significativamente a acurácia da predição de promotores (GAN *et al*., 2012).

Figura 11 – Framework do processo de seleção de características



Fonte: (Gan *et al*., 2012)

No trabalho de FONTANA (2013), os atributos dos dados representados em linguagem natural foram selecionados com base na metodologia TF*IDF (*Term Frequency - Inverse document frequency* - TF*IDF), juntamente com o padrão de medição de proximidade VSM (*Vector Space Model* – VSM). Essa metodologia tem mostrado boa performance em diversas aplicações da vida real, como na clusterização e classificação de documentos de texto de larga escala, porém na aplicação dessa metodologia para a análise genômica, ocorre um grande número de ruídos nos vetores de funções TF*IDF. Para resolver este problema, HE (2008) propõe duas abordagens: primeiramente, ele investiga técnicas de melhoria sobre a representação convencional de funções TF*IDF, mais especificamente o método IDF baseado na pesagem de termos, para que os termos que refletem melhor as funções biológicas dos genes possam receber pesos relativamente maiores e diminuir a quantidade de ruídos causada por termos irrelevantes que seriam correspondentemente suprimidos. Segundamente, ao invés de assumir uma completa

independência entre os termos das funções, o autor estuda as suas correlações herdadas, e levar essas informações em conta para uma estimação mais precisa do padrão de proximidade (HE, 2008).

A representação da metodologia TF*IDF pode ser formulada como:

$$X = \{x_1, x_2, \dots, x_M\} = \{tf_1 * idf_1, tf_2 * idf_2, \dots, tf_M * idf_M\}, \quad (1)$$

onde tf_i é a TF de um termo característico t_i, no documento X, e idf_i é a frequência inversa do t_i, em referência a um conjunto de dados inteiro, definido como:

$$idf = -log \frac{df_i}{N}, \quad (2)$$

onde df_i é a frequência do documento de t_i, por exemplo, o número de documentos que contém t_i, e N é o número total de documentos no conjunto de dados (HE, 2008).

A ideia de expansão automática de termos (ATE – *Automatic Term Expansion*) origina da teoria da expansão de consulta (QE – *Query Expansion*) da área de recuperação de informação (IR – *Information Retrieval*). Dada uma consulta do usuário e um número inicial de resultados recuperados, QE refere-se ao processo de interativamente, ou automaticamente reformular a pesquisa do usuário para aumentar a performance de recuperação em termos de precisão (MITRA *et al*., 1998).

O primeiro método proposto por HE (2008) utiliza o conceito de QE, onde o autor considerou a descrição funcional de cada gene como sendo uma consulta individual e seus resultados recuperados em referência a um conjunto inteiro são ranqueadas de acordo com uma medida de proximidade, utilizando a equação euclidiana. Então ao aplicar um algoritmo de QE apropriado, pode-se polir a consulta através da modificação do seu vetor de características e identificação de outros genes com funções biológicas mais relevantes, em outras palavras, aumentar a medição do padrão de proximidade. O autor propôs dois métodos de expansão baseados na coocorrência de termos literal(ATE-L) e conceitual(ATE-C).

3.4.1 ATE utilizando estatísticas de coocorrência de termos literais

O primeiro método é derivado da ideia de MITRA *et al*. (1998), onde assume-se que dois termos correlacionados próximos irão ocorrer juntos em vários documentos. Por outro lado, se dois termos se referem a diferentes conceitos, sua coocorrência não irá ser fortemente correlacionada. Nos trabalhos de MITRA *et al*.

(1998), termos com grandes correlações a aqueles presentes nos resultados recuperados tiveram seus pesos baixados a fim de maximizar a variedade de futuros resultados. Com uma abordagem diferente, o estudo de HE (2008) focou em expandir o conjunto de termos correlacionados a fim de reduzir a escassez do espaço vetorial, e aumentar o peso deles, a fim de aumentar a representação da consulta original. Especificamente, a correlação termo-a-termo entre o t_i e t_j baseada na sua coocorrência literal em todo o conjunto de dados S, notada como $Cor(t_i, t_j)$ é calculada por:

$$Cor(t_i, t_j) = min\{P(t_i|t_j), P(t_j|t_i)\}, \quad (3)$$

onde a probabilidade posterior $P(t_j|t_i)$ é estimada como:

$$P(t_i|t_j) = \frac{número de documentos em S contendo t_i e t_j}{número de documentos em S contendo t_j} \quad (4)$$

Então dado um vetor de características gênicas na equação 1, em respeito ao conjunto de termos de características $T = \{t_1, t_2, \dots, t_M\}$, cada valor de atributo x_i, é reformulado como:

$$x_i' = \sum_{j=1}^{M} U(t_i, t_j).Cor(t_i, t_j).x_j = \sum_{j=1}^{M} U(t_i, t_j).Cor(t_i, t_j).tf_j.idf_j, \quad (5)$$

onde $U(t_i, t_J)$ é um interruptor de utilização em resposta a correlação termo-a-termo entre t_i e t_J, definida por:

$$U(t_i, t_J) = \begin{cases} 1 \; se \; Cor(t_i, t_j) \geq \alpha \\ 0 \; caso \; contrário \end{cases}, \quad (6)$$

onde $\alpha \in [0,1]$ é um limiar positivo.

Já que $Cor(t_i, t_j) = 1.0$, a equação 5 essencialmente aumenta o peso de todos os termos. O grau dos aumentos de peso depende do número de termos que são altamente correlacionados a eles e aos níveis de suas correlações termo-a-termo. Compreensivelmente, termos comuns têm baixa correlação a todos os outros termos de acordo com a equação 4, devido a sua grande TF. Por outro lado, termos especializados que frequentemente coocorrem nas mesmas anotações funcionais terão seus pesos aumentados significativamente. Por fim, este processo otimiza os “melhores” termos e reduz a quantidade de ruído causada por termos comuns durante a representação das funções (HE, 2008).

O segundo método de expansão automática de termo (ATE-C) proposto por HE (2008) é a expansão automática de termos literais baseados em anotações de ontologias de genes (GO) e não será objeto de estudo deste trabalho.

Ao aplicar os dois métodos de ATE propostos em seu trabalho na medição de padrão de similaridades entre genes (baseado nas suas funções biológicas), HE (2008) obteve um ganho de desempenho considerável sobre o método convencional TF*IDF. O autor concluiu que com um controle apropriado do grau da expansão (α), a ATE baseada na coocorrência literal pode auxiliar a identificar mais genes do interesse do usuário baseado na consulta de pesquisa.

3.4.2 MUSA

SAGOT *et al*. (2000), apontaram a necessidade de distinguir entre um motivo e suas ocorrências nas sequências de entrada. De fato, os autores evitaram o uso do termo motivo, introduzindo a noção de modelo. O modelo corresponde a uma descrição do que constitui um modelo de ocorrência. Um modelo é definido como uma sequência sobre $\sum^{+}$, onde $\sum$ é o alfabeto. Um modelo M é dito por possuir uma e-ocorrência, ou simplesmente uma ocorrência nas sequências de entrada, se existe uma palavra U na sequência de entrada em uma distância de M não maior que e. O modelo é dito por ser válido se ele possuir ocorrências no mínimo em $q \leq t$ das sequências de entrada, onde q é o quórum. Um modelo estruturado (motivo complexo) é definido como um par (m, d), onde:

$m = (m_i)_{1 \leq i \leq p}$ é uma tupla p de modelos $(m_i \in \sum^{+})$, denotando p componentes;

$d = (d_{min}, d_{max}, \delta_i)_{1 \leq i \leq p-1}$ é uma tupla $(p-1)$ de triplas, denotando $p-1$ intervalos entre os componentes.

Além disso, considerando um conjunto de sequências de entrada $S = \{S1, \ldots, St\}$, um modelo estruturado é dito por ser válido se, para cada $1 \leq i \leq p-1$ e para todas as ocorrências u_i de m_i, existem ocorrências u_1,..., u_p, de simples motivos $m_1, \ldots, m_p$ tal que:

- u_1,..., u_p pertencem a mesma sequência de entrada;

- Existe d_i, com $d_{min} + \delta_i \leq d_{min} \leq d_{max} - \delta_i$, tal que a distância entre a posição final de u_i e a posição inicial de u_{i+1} na sequência está em $[d_i - \delta_i, d_i + \delta_i]$;
- d_i é o mesmo para a tupla p de ocorrências presentes em no mínimo $q \leq t$ das sequências de entrada distintas.

MENDES *et al*. (2006), desenvolveram um algoritmo que possibilita a identificação de motivos simples e complexos utilizando uma abordagem não supervisionada, a MUSA (*Motif Finding Using an Unsupervised Approach*). As definições listadas anteriormente servem o propósito de um localizador de motivos, que precisa restringir o espaço de pesquisa e decidir o que é um padrão suficientemente comum para que ele possa ser reportado. O algoritmo não faz suposições sobre o número de componentes dos motivos complexos que está procurando, nem sobre as distâncias entre cada componente. Os autores definiram um motivo complexo como um número p de componentes de motivos simples, cada um separado por distâncias $(d_i, \varepsilon)_{1 \leq i \leq p-1}$, onde p é o número de componentes. Uma ocorrência de um motivo complexo é, similarmente, um conjunto de ocorrências exatas de cada componente de motivo simples, $u_1, \ldots, u_p$ na mesma sequência de entrada, onde cada ocorrência é separada por um intervalo cujo tamanho está em $[d_i - \varepsilon, d_i + \varepsilon]$.

Tanto p como as distâncias d_i são determinadas pelos algoritmos. O único parâmetro especificado pelos autores é o ε.

Figura 12 – Conjunto de sequências de entrada

S_1 : TAACCTGGTACA
S_2 : CGAATCTTGGTC
S_3 : GGAACTGCGGTG
S_4 : CTAATCCTAGGC
S_5 : GTAACTTCCGGT
S_6 : TCAAGCCTAGGC

Fonte: MENDES *et al*. (2006).

Para evitar os parâmetros de extração definidos arbitrariamente, os autores tentaram realçar certas características das sequências de entrada. Para este efeito, foi iniciada a construção de uma matriz de coocorrência M. Para construir essa matriz, foi necessário identificar todas as ocorrências das sequências de algum tamanho

menor γ, por exemplo, todos os elementos de tamanho γ nas sequências de entrada $S = \{S_1, \dots, S_t\}$.

Tendo $L(S) = \{m_1, \dots, m_z\}$ como sendo a lista de todos os elementos de tamanho γ, notando que $z \leq |\sum|^{\gamma}$. A figura 12 mostra um conjunto de sequências de entrada que será utilizado para ilustrar as definições dadas abaixo. Neste exemplo foi utilizado γ = 2.

Definição 1: lista de elementos de tamanho γ, tendo S como sendo um conjunto de sequências de entrada e $m \in L(S)$. $Occ_{i,s}(m)$ denota o conjunto de coordenadas de todas as ocorrências de m em $S_I \in S$.

O conjunto $Occ_{i,s}(m)$, é uma lista de inteiros denotando as posições em que pode-se encontrar m em uma sequência $S_i \in S$. Sempre que o conjunto de sequências S, estiver claro, escreve-se $Occ_i(m)$. Considerando o exemplo na figura 1, pode-se visualizar que $Occ_3(GG) = \{1,9\}$, $Occ_1(AA) = \{2\}$, ou que $Occ_6(TT) = \emptyset$.

Definição 2: configuração de um par de elementos de tamanho γ, tendo S como sendo um conjunto de sequências de entrada. A configuração de um par de elementos de tamanho γ é uma tupla (m_r, m_s, d), com $m_r, m_s \in L(S)$ e $d \in Z \backslash \{0\}$.

Nessa definição os autores introduziram um objeto matemático chamado de configuração. Este objeto foi associado a um conjunto de sequências de entrada e foi usado para denotar a coocorrência de um par de elementos de tamanho γ em uma posição relativa específica.

Definição 3: configurações de um par de elementos de tamanho γ sobre uma sequência, tendo S como sendo um conjunto de sequências de entrada e $m_r, m_s \in L(S)$. $\Delta_s(m_r, m_s)$ denota o conjunto de todas as configurações de m_r, m_s sobre $S_i \in S$.

$$\Delta_{i,S}(m_r, m_s) = \{(m_r, m_s, d) : d = c_s - c_r\}, \quad (7)$$

tal que, $c_r \in Occ_{i,S}(m_r)$, $c_s \in Occ_{i,S}(m_s)$ e $c_r \neq c_s$

Considerando a configuração (m_r, m_s, d), e se $d < \gamma$ então a configuração representa a ocorrência de um elemento de tamanho $(\gamma + d)$. Nota-se que $\Delta_{i,S}(m_s, m_r) = \{(m_s, m_r, d) : (m_r, m_s, -d) \in \Delta_{i,S}(m_r, m_s)$. Novamente, foi utilizado

$\Delta_{i,}(m_r, m_s)$ toda vez que o conjunto de sequências de entradas estiver claro. Considerando o exemplo anterior, podemos observar que $\Delta_{3,}(AA, GG) = \{(AA, GG, -2), (AA, GG, 6)\}$ ou que $\Delta_{6,}(AA, TT) = \emptyset$.

Definição 4: pontuação de uma configuração de um par de elementos de tamanho γ, tendo S como sendo um conjunto de sequências de entrada e $m_r, m_s \epsilon L(S)$ e $d \in N$. $\mu_{i,S} : \sum^{\gamma} \times \sum^{\gamma} \times Z \rightarrow \{0,1\}$ é a função de filiação de uma configuração com respeito ao conjunto de todas as configurações de dois elementos de tamanho γ em uma sequência de entrada $S_i \in S$, definida como:

$$\mu_{i,S}(m_r, m_s, d) = \begin{cases} 1\ se\ (m_r, m_s, d) \in \Delta_{i,S}(m_r, m_s) \\ 0, caso\ contrário \end{cases} \quad (8)$$

$\sigma_s: \sum^{\gamma} \times \sum^{\gamma} \times Z \rightarrow \{0, \ldots, |S|\}$ é a função da pontuação para uma configuração e é definida como:

$$\sigma_s(m_r, m_s, d) = \sum_{i=1}^{|S|} \mu_{i,S}(m_r, m_s, d) \quad (9)$$

A pontuação de uma configuração de um par de elementos de tamanho γ é simplesmente a quantidade de sequências onde aquela configuração particular pode ser observada. Partindo das definições anteriores, $\sigma(AA, GG, 7) = 3$, uma vez que GG ocorre sete posições depois de AA nas sequências S_4, S_5 e S_6, produzindo $\mu_{4,S}(AA, GG, 7) = \mu_{5,S}(AA, GG, 7) = \mu_{6,S}(AA, GG, 7) = 1$.

Definição 5: pontuação ε-tolerante de uma configuração de um par de elementos de tamanho γ, tendo S como sendo um conjunto de sequências de entrada e $m_r, m_s \epsilon L(S)$ e $\varepsilon \in N_0$. A pontuação ε-tolerante de uma configuração $\sigma_s^{\varepsilon}: \sum^{\gamma} \times \sum^{\gamma} \times Z \rightarrow N_0$ é definida como:

$$\sigma_s^{\varepsilon}(m_r, m_s, d) = \sum_{i=1}^{|S|} max_{k=-\varepsilon,\ldots,\varepsilon}\ \mu_{i,S}(m_r, m_s, d + k), \quad (10)$$

sendo $(d \neq 0)$, portanto, $\sigma_s^{\varepsilon}(m_r, m_s, 0) = 0$

O conceito da pontuação ε-tolerante de uma configuração de um par de elementos de tamanho γ endereça a necessidade de habilitar que uma configuração tenha pequenas variações. Isso remove a rigidez de requerer que um par de elementos de tamanho γ ocorram em posições relativas fixadas a fim de obter uma pontuação maior. Essa situação ocorre no exemplo mostrado na figura 1, onde

$\sigma(AA, GG, 7) = 3$, $\sigma(AA, GG, 6) = 2$ e $\sigma(AA, GG, 5) = 1$, mas $\sigma^1(AA, GG, 7) = 5$, $\sigma^1(AA, GG, 6) = 6$ e $\sigma^1(AA, GG, 5) = 3$. Uma pontuação l -tolerante é capaz de compreender o fato de que o elemento AA coocorre com GG em todas as sequências de entrada a uma distância de 6 ± 1 posições. Incidentalmente, $\sigma^1(AA, GG, 4) = 1$, desconsiderando o fato que $\sigma(AA, GG, 4) = 0$. Isso pode ser útil para descrever padrões de coocorrência que possuem uma pontuação ε-tolerante alta, mesmo eles nunca realmente ocorrendo nas sequências de entrada.

Definição 6: configuração mais comum de um par de elementos de tamanho γ, tendo S como sendo um conjunto de sequências de entrada e $m_r, m_s \, \epsilon \, L(S)$. Uma configuração (m_r, m_s, d^*) é tida como a configuração mais comum de dois elementos de tamanho γ se, para cada configuração (m_r, m_s, d), $\sigma_s \, (m_r, m_s, d^*) \geq \sigma_s(m_r, m_s, d)$. A configuração ε-tolerante é a mais comum se a mesma asserção mantém para a pontuação ε-tolerante.

A noção da configuração mais comum foi utilizada para encontrar as configurações com a maior pontuação para um par de elementos de tamanho γ. Para o exemplo mostrado na figura 9, pode-se visualizar que a configuração mais comum para o par (AA, GG) é $(AA, GG, 7)$. A configuração l -tolerante mais comum é, entretanto, $(AA, GG, 6)$.

A partir das definições anteriores, o autor definiu a matriz de coocorrências, que reúne a informação sobre a pontuação ε-tolerante da configuração mais comum de cada par de elementos de tamanho γ.

Definição 7: matriz de coocorrências, tendo S como sendo um conjunto de sequências de entrada e $m_r, m_s \, \epsilon \, L(S)$. Uma matriz de coocorrências sobre S com ε-tolerância, M_S^{ε} , é a matriz onde cada um de seus elementos a_{ij} é definida por:

$$a_{ij} = \sigma_S^{\varepsilon}(m_i, m_j, d^*), \tag{11}$$

onde (m_i, m_j, d^*) é a configuração ε-tolerante mais comum de $m_i, m_j \in L(S)$ e $\mathrm{i, j} = 1, \ldots, |\mathrm{L(S)}|$.

A matriz de coocorrências M^1, derivada da sequência de entrada da figura 10 é mostrada na figura 13. Ao inspecionar a matriz e as sequências de entrada, verificou-se que existem duas configurações com a pontuação máxima l-tolerante: $(AA, CT, 3)$

e $(AA, GG, 6)$. Neste caso, pode-se enxergar que AA coocorre com CT em todas as sequências em uma distância relativa de 3 ± 1 e com GG, também em todas as sequências a uma distância relativa de 6 ± 1. A matriz de coocorrências da figura 11 é simétrica. Os autores constataram que a matriz de coocorrência, M_S^{ε} , é simétrica, ou seja, $a_{ij} = a_{ij}$ para cada $i, j = 1, \ldots, |L(S)|$.

Figura 13 – Matriz de coocorrências

	AA	AC	AG	AT	CA	CC	CG	CT	GA	GC	GG	GT	TA	TC	TG	TT
AA	0	3	2	2	1	3	2	**6**	2	2	**6**	4	3	3	3	2
AC	3	1	0	0	1	1	2	3	1	1	3	3	2	1	2	1
AG	2	0	1	1	1	2	0	2	0	2	2	0	2	2	0	0
AT	2	0	1	0	0	1	1	2	1	1	2	1	1	2	1	1
CA	1	1	1	0	0	1	0	1	0	1	1	1	1	1	1	0
CC	3	1	2	1	1	0	1	3	0	2	4	2	2	2	1	1
CG	2	2	0	1	0	1	0	2	1	1	2	2	1	1	1	1
CT	**6**	3	2	2	1	3	2	1	2	3	5	3	3	2	3	2
GA	2	1	0	1	0	0	1	2	0	1	2	2	0	1	2	1
GC	2	1	2	1	1	2	1	3	1	1	2	1	2	2	1	0
GG	**6**	3	2	2	1	4	2	5	2	2	1	4	3	3	3	2
GT	4	3	0	1	1	2	2	3	2	1	4	1	2	2	3	2
TA	3	2	2	1	1	2	1	3	0	2	3	2	2	2	1	1
TC	3	1	2	2	1	2	1	2	1	2	3	2	2	1	1	1
TG	3	2	0	1	1	1	1	3	2	1	3	3	1	1	1	1
TT	2	1	0	1	0	1	1	2	1	0	2	2	1	1	1	0

Fonte: MENDES *et al*. (2006).

Após encontrar a matriz de coocorrências, os autores aplicaram uma abordagem de biclusterização e, ainda, uma etapa de pós-processamento que classificou os motivos em famílias de acordo com sua relevância.

Segundo os autores, o resultado dos experimentos realizados com a MUSA, utilizando sequências promotoras de bactérias, levaram a descoberta de padrões interessantes que são biologicamente relevantes e que conduziram os resultados pesquisas anteriores. A informação que emergiu da aplicação da MUSA é de interesse para guiar pesquisas experimentais no campo da regulação da expressão gênica das bactérias

3.5 EXTRAÇÃO DE ATRIBUTOS

Métodos de extração de atributos procuram extrair os atributos mais representativos com menores dimensões dos dados originais. Este processo envolve a produção de um novo conjunto de atributos a partir dos atributos originais dos dados,

através da aplicação de algum mapeamento. Os métodos não supervisionados mais conhecidos de extração de atributos são o *Principal Component Analysis* (PCA) e clusterização espectral. Os atributos que são importantes para manter os conceitos dos dados originais são extraídos do conjunto inteiro de atributos.

O método PCA é um procedimento matemático que transforma um número de variáveis possivelmente correlacionadas em um número menor de variáveis não-correlacionadas, chamadas de componentes principais. O objetivo do PCA é reduzir a dimensionalidade do conjunto de dados e ao mesmo tempo manter a maior parte da variabilidade original presente nos dados. Uma matriz de transformação A, utilizada para o cálculo de componentes principais, pode ser computada resolvendo os vetores e valores característicos de covariância dos vetores de atributos. Então os k (a dimensão do vetor de atributos) vetores característicos correspondentes aos k maiores valores característicos para formar a matriz Ak são escolhidos. Utilizando a matriz de transformação Ak, um vetor de atributos comprimido de dimensão k pode ser derivado através de combinações lineares dos vetores originais. A redução da dimensionalidade dos atributos torna o classificador mais eficiente e mais estável (ZHENG & ESSOCK, 2003).

Para dados em séries temporais, conforme contexto apresentado na seção 3.3.2, os atributos extraídos podem ser ordenados por importância através da utilização de uma função de mapeamento. Por este motivo, a extração de atributos é muito mais popular que a seleção de atributos na mineração de dados de séries temporais (ZHANG *et al*., 2005). O objetivo é encontrar uma representação em uma dimensionalidade mais baixa que preserve a informação original e descreva o formato original da série temporal da forma mais próxima possível. Diversas abordagens têm sido sugeridas na literatura, incluindo a transformada discreta de Fourier (DFT – *Discrete* Fourier *Transform*) (AGRAWAL *et al*., 1993), decomposição de Valor Singular (SVD – *Singular Value Decomposition*) (KORN *et al*., 1997), aproximação agregada por partes (PAA – *Piecewise Agregation Aproximation*) (YI & FALOUTSOS, 2000), e a transformada discreta de *wavelet* (VLACHOS *et al*., 2003).

Em toda transformação aplicada sobre um sinal, as funções de base utilizadas pelo método em questão determinam o tipo de informação adicional que poderá ser extraída do processo em análise. No domínio de frequência, também chamado de

análise espectral, a ferramenta matemática mais difundida entre os profissionais para processamento de sinais é a transformada de Fourier, que faz o uso de funções harmônicas (senos e cossenos) como funções base. Este tipo de abordagem constitui um método não paramétrico de análise de processos estocásticos, cuja descrição obtida para o sinal se dá pela caracterização do seu conteúdo na frequência e nenhuma informação temporal.

> Ferramentas de domínio de frequência, tais como as séries de Fourier, transformadas de Fourier e suas variantes discretas constituem uma das pedras angulares do processamento de sinais. Essas transformadas decompõem um sinal em uma sequência ou contínuo de componentes senoidais que identificam o conteúdo de domínio de frequência do sinal (HANSELMAN & LITTLEFIELD, 2003, p 251).

3.5.1 Transformada discreta de Fourier

A DFT é utilizada para produzir análise de frequência de sinais discretos não-periódicos. A equação da DFT envolve diversas adições e multiplicações de números complexos. Para amostras pequenas, este número é aceitável, porém para grandes quantidades de dados, o cálculo se torna muito custoso, atingindo proporções inimagináveis. A DFT é a projeção de um sinal a partir do domínio tempo para o domínio de frequência por:

$$\mathrm{c_f} = \frac{1}{\sqrt{\mathrm{n}}}\sum_{\mathrm{t=1}}^{\mathrm{n}} \mathrm{f(t)}\exp\left(\frac{-2\pi \mathrm{ift}}{\mathrm{n}}\right), \qquad (12)$$

onde $f = 1, \dots, n$ e $i = \sqrt{-1}$. Os c_f são números complexos que representam as amplitudes e deslocamentos de uma decomposição do sinal em funções sinusoidais. Para sinais reais, c_i é a conjugada complexa de $c_{n-i+1}\left(i = 2, \dots, \frac{n}{2}\right)$ (MORCHEN, 2005).

3.5.2 Transformada discreta cosseno

A transformada discreta cosseno (DCT – *Discrete Cosine Transform*) é uma transformada similar a DFT, mas utiliza somente funções cosseno ao invés de cossenos e senos. Ela transforma o sinal do domínio temporal para o domínio de frequência, que realça a periodicidade do sinal (BATAL & HAUSKRECHT, 2009).

A DCT para uma série temporal X de um tamanho n é definida como:

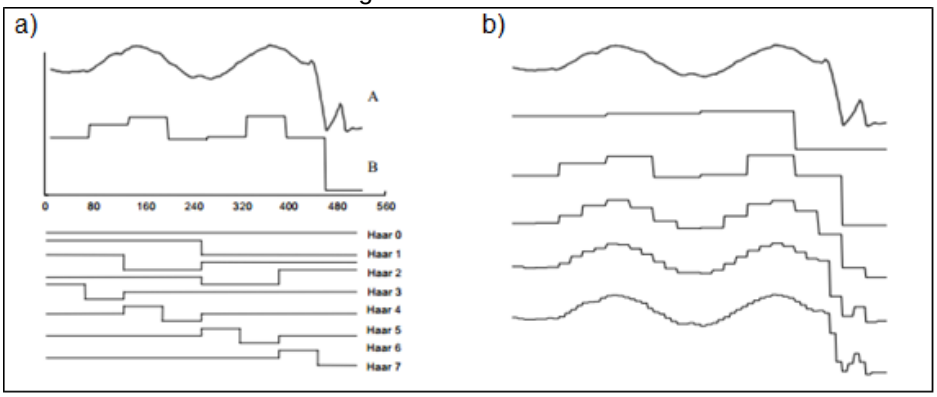
$$X_f = K(f)\sum_{i=1}^{n} x_i cos\frac{\pi f(i-0,5)}{n}, \quad (13)$$

sendo $f = 0, \ldots, n-1$, onde $K(f) = \frac{1}{\sqrt{n}}$ quando $f = 0$ e $K(f) = \sqrt{\frac{2}{n}}$ quando $1 \leq f \leq n-1$. A multiplicação por $K(f)$ torna a DCT uma função de transformação ortonormal.

3.5.3 Transformada discreta de *wavelet*

Enquanto as abordagens mencionadas anteriormente têm compartilhado a habilidade de produzir uma aproximação de alta qualidade na redução da dimensionalidade das séries temporais, as *wavelets* são únicas no sentido de que a sua representação dos dados é intrinsecamente multiresolução. *Wavelets* são funções matemáticas que representam os dados ou outras funções em termos de médias e diferenças de uma função protótipo, também chamada de mãe *wavelet*. Neste sentido, elas são similares a transformada de Fourier. Uma das vantagens principais das *wavelets* é a sua habilidade de se adaptar as características de uma função, como descontinuidades e variação no comportamento da frequência. Uma diferença fundamental é que as *wavelets* estão localizadas no tempo. Em outras palavras, alguns dos coeficientes de *wavelets* representam pequenas subseções locais dos dados estudados, que é o oposto dos coeficientes de Fourier, que sempre representam contribuições globais para os dados. Essa propriedade é muito útil para a análise de multirresolução dos dados. Os primeiros coeficientes contêm uma visão geral dos dados, enquanto os coeficientes adicionais podem ser percebidos ao dar zoom nas áreas de grande detalhe. A figura 14a é uma representação da *wavelet* de Haar, que pode ser visualizada como uma tentativa de aproximar uma série temporal com uma combinação linear de funções base, onde a série temporal A é transformada na B através da decomposição Haar *wavelet* e a dimensionalidade foi reduzira de 512 para 8. A figura 14b demonstra os diferentes níveis de resolução encontrados através da Haar *wavelet*. (VLACHOS *et al*., 2003; BENNET *et al*., 2014).

Figura 14 – *Wavelet* de Haar



Fonte: VLACHOS *et al*., (2003)

A transformada discreta *wavelet* (DWT – *Discrete Wavelet Transform*) é utilizada na análise da expressão gênica devido a sua abordagem multirresolução no processamento de sinais. Na aplicação de BENNET *et al*. (2014), os dados do *microarray* foram transformados no domínio de escala temporal e foram utilizados como atributos de classificação. Os dados da expressão gênica foram representados por uma matriz em que as linhas representaram os genes e as colunas representaram as amostras. Já que cada amostra continha milhares de genes, o número de genes pôde ser visualizado como o tamanho dos sinais (BENNET *et al*., 2014).

A decomposição Haar *wavelet* age através do cálculo da média entre dois valores adjacentes nas funções de séries temporais em uma determinada resolução para formar uma suavidade, um sinal menos dimensionado, e os coeficientes resultantes são simplesmente as diferenças entre os valores e suas médias. No trabalho de VLACHOS *et al*. (2003), foi proposta a utilização da decomposição *wavelet* para executar a clusterização nos maiores níveis de decomposição, onde a Haar *wavelet* é calculada para todas as séries temporais do conjunto de dados. A competição entre DFT e Haar DWT foi resolvida por WU *et al*. (2000), concluindo que elas são comparáveis na preservação da energia, mas a DWT é mais rápida para calcular e oferece a decomposição multirresolução.

No trabalho de MORCHEN (2005), a utilização do mesmo subconjunto de coeficientes do tamanho k, escolhidas por uma função de agregação, medindo a importância de cada posição do coeficiente é proposta. A função do critério é avaliada com os valores em uma posição de coeficiente de todas as séries temporais. Desta forma, o autor atinge a mesma redução de dimensionalidade para k valores por série e mantém a computação da distância de uma forma simples. A restrição para utilizar

o mesmo subconjunto aumenta a comparabilidade de duas séries temporais e também a intepretação de qualquer resultado. Este método também oferece maior poder de semântica no sentido de que formatos mais complexos podem ser representados por um vetor de caraterísticas. Utilizar posições de coeficiente arbitrárias não somente diminui as frequências, mas qualquer frequência pode ser incluída se for de importância para o conjunto de dados (MORCHEN, 2005).

Tendo $\big(f(1),\dots,f(n)\big)$ como sendo uma amostra uniforme de um sinal real $f(t)$. Tanto a DFT quanto DWT expressam o sinal como coeficientes em uma função espaço gerados por um conjunto de funções base. As funções base da transformada de Fourier contêm somente a função exponencial complexa, vide seção 3.5.1, representando funções sinusoidais no domínio real. As funções base da transformada *wavelet* consistem de diversas versões escaladas e deslocadas de uma função mãe *wavelet* (MORCHEN, 2005).

A DWT mede a frequência em diferentes resoluções de tempo e locais e o sinal é projetado no plano de tempo-frequência. As funções base são:

$$\omega_{j,k}(t) = 2^{\frac{j}{2}}\omega\big(2^j t - k\big), \tag{14}$$

onde ω é a função *wavelet* mãe. Qualquer função real integrável quadrada $f(t)$ pode ser representada em termos dessas funções bases como:

$$f(t) = \sum_{j,k} c_{j,k}\,\omega_{j,k}(t), \tag{15}$$

e os $c_{j,k} = \Big(\omega_{j,k}(t), f(t)\Big)$ são os coeficientes da DWT. Uma *wavelet* simples e comumente usada é a Haar (figura 12) como a função mãe:

$$\omega_{\mathrm{Haar}}(\mathrm{t}) = \begin{cases} 1, \text{se } 0 < \mathrm{t} < 0{,}5 \\ -1, \text{se } 0{,}5 < \mathrm{t} < 1 \\ 0, \text{caso contrário} \end{cases} \tag{16}$$

MORCHEN (2005), utilizou as seguintes definições: para um conjunto de l séries temporais, tenhamos $C = \big(c_{i,j}\big) i = 1,\dots,lj = 1,\dots,m$ como sendo a matriz $l \times m$ dos coeficientes obtidos com a DFT, ou DWT. Para DFT e Haar DWT, $m = n$. Tendo c_j como sendo a j-ésima coluna de$C$, $I: R^l \to R$ como sendo a função medindo a importância da posição do coeficiente j baseado em todos os valores de l para este coeficiente e $J_k(I, C)$ como sendo a função para escolher um subconjunto de $M =$

$\{1, \ldots, m\}$ de tamanho k baseado nos valores agregados calculados com a função I em todas as colunas de C. Uma escolha sensível para J_k, é a escolha das colunas com os maiores k valores de l, dependendo da semântica do I:

$$J_k^1(I, A) = \{J | I(c_j) \geq I(c_{j'}) \forall_j \in J, j' \in M \backslash J\}, \quad (17)$$

onde JCM com $|J| - k$.

A energia de um sinal amostrado para as transformadas ortogonais, como a DFT e DWT, é definida como:

$$E(f(t)) = \sum_{j=1}^{n} a_j \, c_j^2, \quad (18)$$

com os coeficientes de escala apropriados a_j para os coeficientes c_j obtidos do sinal correspondente $f(t)$ (MORCHEN, 2005).

A utilização de posições de coeficientes arbitrárias, não somente diminui as frequências, mas qualquer frequência pode ser incluída se for de importância do conjunto de dados. MORCHEN (2005) definiu o seguinte teorema:

Na escolha das colunas k de C, a função $J_k^1(média(c_j^2), C)$ é ótima na preservação da energia.

Para provar este teorema, o autor quis otimizar a energia presente nos coeficientes das colunas indexadas por JCM, definida como:

$$E = \sum_{i-1}^{1} \sum_{j \in J} c_{ij}^2, \quad (19)$$

ao trocar as somas e fatorar a definição anterior, ele conseguiu um termo proporcional a soma da potência da média por posição de coeficiente:

$$= \sum_{j \in J} \sum_{i=1}^{l} c_{1,j}^2 = \sum_{j \in J} (l \frac{1}{l} \sum_{i=1}^{l} c_{1,j}^2) = l \sum_{j \in J} média(c_j^2) \propto \sum_{j \in J} média(c_j^2), \quad (20)$$

Onde c_j^2 é o quadrado do elemento c_j . Assim escolhendo $I(c_j) = média(c_j^2)$ e utilizando J_k^1 resultará na melhor preservação de energia com k possíveis coeficientes, sob a restrição de escolher o mesmo subconjunto para todas as séries temporais (MORCHEN, 2005).

MORCHEN (2005) aplicou essas definições para a extração de atributos de diversos conjuntos de séries temporais em diferentes domínios de problemas. Os

dados resultantes da extração foram submetidos a clusterização através da execução do algoritmo *k-means*. O autor concluiu que a solução proposta para extração de atributos funcionou conforme o esperado, a classificação perfeita pôde ser atingida com a DWT e a utilização de apenas 4 coeficientes, além disso, concluiu-se que quanto mais coeficientes fossem utilizados, mais ruídos seriam introduzidos e piores resultados seriam encontrados.

3.6 CONSIDERAÇÕES FINAIS

Neste capítulo foram demonstradas metodologias de clusterização, extração e seleção de atributos, bem como algumas de suas aplicações na área da bioinformática. Pôde-se observar que a área de mineração de dados, aplicada a bioinformática é alvo de constantes pesquisas e melhoramentos quando se trata da análise genômica. Devido a complexidade dos dados de natureza biológica, a extração e seleção de atributos acaba se tornando uma etapa crucial dentre os estágios da clusterização, pois dependendo da escolha dos atributos, os resultados podem ser afetados diretamente. No próximo capítulo será proposto um projeto desenvolvimento da solução para o problema deste projeto de pesquisa, fundamentado nos conceitos apresentados nos capítulos anteriores.

4 METODOLOGIA

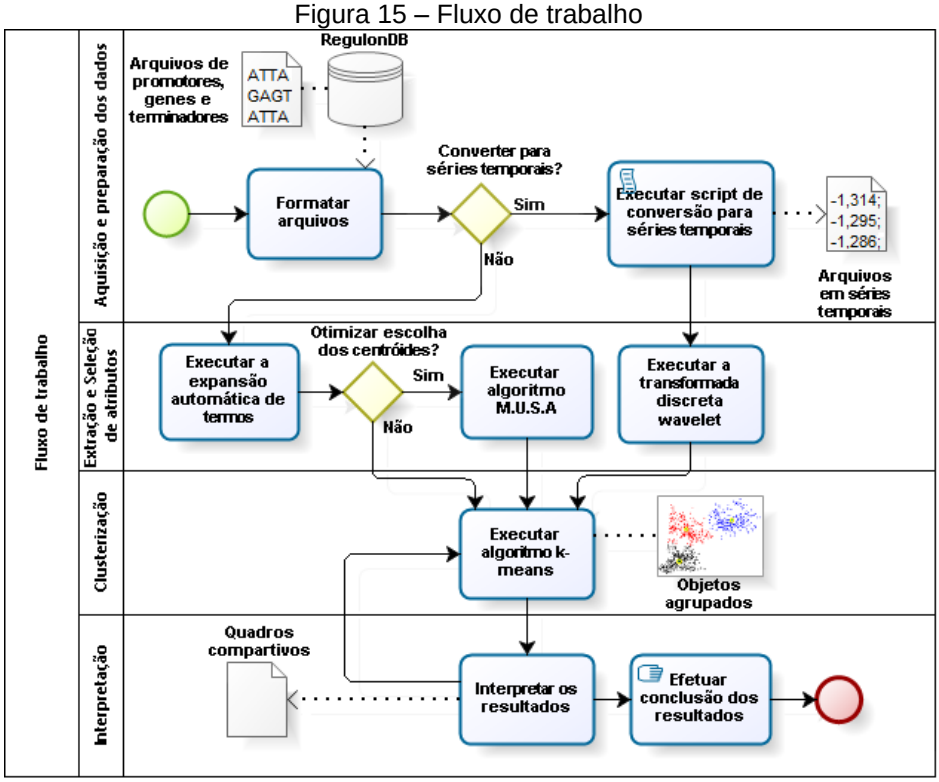
Neste capítulo é apresentada a metodologia para o desenvolvimento da solução, contemplando o fluxo de trabalho, as ferramentas utilizadas, a base de dados e os algoritmos implementados.

4.1 FLUXO DE TRABALHO

O desenvolvimento deste trabalho consiste no fluxo de trabalho demonstrado na figura 15 e ocorrerá através da implementação e execução de uma ferramenta que comporte as funcionalidades abaixo, apresentadas em ordem de utilização:

- Importação de arquivos contendo sequências do DNA da bactéria *E.coli* que serão extraídos da base de dados pública no formato de linguagem natural, conforme descrito na seção 3.3.1;
- Formatação dos arquivos de entrada para um padrão que seja interpretável pela ferramenta;
- Conversão das sequências em linguagem natural para o formato de séries temporais, descrito na seção 3.3.2. Possibilidade de conversão baseando-se em duas propriedades estruturais do DNA, a estabilidade e curvatura, descritas na seção 2.4;
- Extração e seleção dos atributos das sequências nos formatos de linguagem natural, através da aplicação da ATE e a MUSA e também das séries temporais, com a aplicação da DWT;
- Aplicação do algoritmo de clusterização em ambos os tipos de sequência, em linguagem natural e em séries temporais e em diferentes configurações de extração de atributos e quantidade de *clusters*;
- Interpretação dos resultados gerados pela etapa de clusterização através da elaboração de quadros comparativos.

Figura 15 – Fluxo de trabalho



Fonte: O autor (2015).

A ferramenta desenvolvida no trabalho de FONTANA (2013) contempla as funcionalidades de importação e formatação dos dados, extração de atributos através da coocorrência e modelo de espaço vetorial, bem como a clusterização com os algoritmos K-means e CURE. Mais detalhes sobre o desenvolvimento dessas funcionalidades são encontrados em seu trabalho. Neste projeto será desenvolvida a conversão das sequências para séries temporais, e a seleção e extração de atributos utilizando ATE, MUSA e a DWT.

4.2 DADOS

Os dados analisados neste trabalho são provenientes da base de dados pública RegulonDB, acessada através do endereço http://regulondb.ccg.unam.mx/. Esta base é a referência primária na regulação transcricional da bactéria *E.* coli, contendo conhecimento manualmente extraído de publicações cientificas originais, complementado com uma grande quantidade de conjuntos de dados e predições computacionais compreensivas. Devido à grande quantidade de informações e mapeamentos do gene da *E. coli*, esta será o organismo que será estudado neste trabalho. Para a conversão das sequências no formato de séries temporais baseando-se na propriedade de curvatura será utilizada a ferramenta desenvolvida pelo

Christoph Gohlke, do *Laboratory for Fluorescence Dynamics,* departamento de engenharia biomédica, da Universidade da Califórnia.

4.3 FERRAMENTAS

A linguagem de programação escolhida para o desenvolvimento deste trabalho é a C#, o ambiente de desenvolvimento (IDE - *Integrated Development Environment*) é o Microsoft Visual Studio 2012. Tanto a linguagem, quanto o IDE foram escolhidos por possuírem uma vasta base de conhecimento e por serem amplamente utilizados atualmente. O arquivo de saída que será gerado com os resultados da ferramenta será analisado utilizando o software Excel, do pacote Microsoft Office 2013.

4.4 ALGORITMOS

Para a seleção dos atributos dos dados representados na linguagem natural, serão aplicadas duas abordagens, a ATE e a MUSA, descritas nas seções 3.4.1 e 3.6.1, respectivamente, sendo que a MUSA será aplicada somente para a definição dos centroides iniciais do algoritmo *k-means*, com o intuito de eliminar a possibilidade de que os resultados dos testes realizados no trabalho anterior foram influenciados pela escolha dos centroides iniciais. Para a extração dos atributos dos dados representados no formato de séries temporais, será aplicado o algoritmo da DWT, descrita na seção 3.5.1. A clusterização de ambas as representações será realizada através do algoritmo *k-means*.

A primeira etapa do desenvolvimento consistiu em desenvolver a expansão automática de termos para aumentar os pesos dos atributos mais significativos. O algoritmo foi implementado, conforme passos descritos na sessão 3.4.1. O resultado da ATE é um vetor de atributos com valores reformulados, que servirão como base para a VSM, desta forma, após rodar a ATE, o programa automaticamente executa o algoritmo do VSM.

A segunda etapa consistiu na implementação da MUSA. Inicialmente a estruturação do código seguiu fielmente as 7 definições apresentadas no capítulo 3.4.2, porém ao rodar o programa, percebeu-se que o mesmo não estava sendo executado de maneira eficiente. O tempo de execução estava demasiadamente alto.

Para minimizar o problema da lentidão, foi necessário alterar a estrutura das configurações de pares de elementos. Ao invés de gerar uma nova configuração para

cada comparação entre os elementos, optou-se por verificar se a configuração já havia sido encontrada em algum momento, e se fosse o caso, a nova ocorrência seria guardada em uma lista na primeira configuração. Esta alteração melhorou a performance do algoritmo, pois o consumo de memória foi diminuído.

O resultado da MUSA originalmente era uma lista com as configurações de pares de elementos ordenadas pelo número de ocorrências. Durante a implementação e testes iniciais do código, percebeu-se que a informação da configuração mais comum não era de grande valia para a escolha dos centroides iniciais da etapa de clusterização. A fim de otimizar o aproveitamento do resultado do, foi implementada a noção de “sequências mais comuns”, ao invés de configurações mais comuns, onde são calculadas quantas configurações mais comuns ocorrem em cada sequência de entrada. Essa noção proporcionou que, após a execução da MUSA, as sequências fossem ordenadas por ordem decrescente de ocorrência, sendo a primeira sequência a mais comum, e a última a menos comum.

Por realizar milhões de cálculos para encontrar a distância mais comum entre pares de elementos, o programa não estava sendo executado de forma eficiente para as sequências com maior quantidade de informações, como é o caso dos genes. O tempo de execução para a análise de quatro mil sequências superou 48 horas, quando decidiu-se parar a execução e revisar as estruturas de dados onde as configurações estavam sendo armazenadas. Inicialmente optou-se por armazená-las em uma lista, porém essa estrutura não demonstrou eficiência na localização dos objetos, possui complexidade de O(n). Para contornar esta situação, a estrutura foi alterada para o formato de dicionário, onde cada configuração possui uma chave exclusiva, fazendo com que a busca por elas fosse realizada a uma complexidade de O(1). Esta estrutura fornece maior eficiência, porém gera um custo maior na utilização de memória, onde além de armazenar os próprios objetos, também armazena um índice para os mesmos.

Ambos os resultados da expansão automática de termos e da MUSA são gerados em memória. A cada vez que o usuário quisesse utilizar a MUSA na escolha dos centroides, ele teria que executar novamente o programa, tornando o processo demasiadamente lento e repetitivo. Para contornar este problema, foi desenvolvida uma função que gera o resultado da MUSA e da ATE em um arquivo temporário para

cada arquivo de entrada. Isto permite que as próximas execuções ocorram rapidamente, já que o programa somente lê o arquivo temporário com os resultados encontrados previamente e os mantém em memória. Para o arquivo temporário da MUSA, além de serem armazenadas as pontuações de cada sequência, também são mantidas as configurações mais comuns para cada par de elementos. No arquivo temporário da ATE, além do vetor com o peso dos elementos reformulado, também são informados quais os elementos que tiveram o peso aumentado, bem como o peso anterior e o novo peso. Estas informações são importantes para o domínio biológico e servirá para análises por profissionais especializados em análise genômica.

Na terceira etapa, foi desenvolvido o processo de extração de atributos das séries temporais. Surgiu a necessidade da criação de uma rotina que converta os dados em linguagem natural para séries temporais de uma forma automática, sem a necessidade de utilizar ferramentas externas. Por este motivo foram desenvolvidas as opções de conversão baseadas nas propriedades de estabilidade e curvatura. Na conversão pela estabilidade, cada combinação de bases (AA, AC, TG e etc.) recebe um valor de estabilidade previamente mensurado, que são somados conforme o tamanho da janela escolhida, resultando em uma série temporal. Na conversão pela curvatura, foi necessária a chamada externa da ferramenta citada na seção 4.1. Essa ferramenta faz a conversão de acordo com o modelo selecionado (*Straight, Trifonov, AAWedge, Calladine, Desantis e Reversed,* maiores informações são encontradas em http://www.lfd.uci.edu/~gohlke/code/dnacurve.py.html) dos dados originais para um arquivo no formato “.csv” contendo os valores de curvatura para cada elemento da sequência. Posteriormente este arquivo é lido e convertido para o formato de entrada padrão da ferramenta desenvolvida. Após realizar algumas simulações com a ferramenta da curvatura para os três tipos de sequência, percebeu-se uma baixa performance e um tempo de execução elevado para realização da conversão, principalmente para as sequências gênicas (devido ao tamanho e complexidade maiores), onde foram necessárias aproximadamente 40 horas para converter 1000 sequências.

A DWT foi desenvolvida através de uma função recursiva que retorna em um vetor todos os coeficientes encontrados em cada nível de resolução. Foi necessário que todas as séries temporais fossem equiparadas em tamanho (com a adição de zeros), sendo ele expoente de 2, para que a DWT fosse executada. Para aplicar o

método de escolha dos melhores coeficientes proposto por MORCHEN (2005) foi calculada a média da somatória de todas as posições de coeficientes, conforme a equação 17. Foram selecionadas as posições com os maiores valores, restringindo o mesmo subconjunto de coeficientes para todas as sequências.

Na etapa da clusterização pelo *K-means*, foram incluídas três opções para a escolha dos centroides iniciais. A primeira e a segunda opção permitem que o centroide inicial seja a sequência mais comum, ou a menos comum, respectivamente, ambas resultantes da MUSA. A terceira opção permite que os primeiros centroides sejam escolhidos aleatoriamente dentre as 20 sequências mais comuns de cada arquivo (genes, promotores e terminadores). Nesta etapa também foi incluída a opção de clusterizar arquivos convertidos, gerados através da etapa da conversão, sem que seja necessário submetê-los a fase de extração de atributos.

Por fim, ainda na etapa de clusterização, foi identificado que o algoritmo implementado anteriormente para o cálculo da distância entre os objetos requeria que ambos tivessem a mesma quantidade de atributos. Está é uma característica das sequências em linguagem natural, porém as sequências em séries temporais possuem tamanhos distintos. Portanto, foi necessário que as séries fossem preenchidas e equiparadas com zeros para que o cálculo da distância entre os objetos deste tipo fosse efetuado com sucesso.

4.5 CONSIDERAÇÕES FINAIS

O fluxo de trabalho proposto para o desenvolvimento da ferramenta foi concluído com sucesso. Imagens da interface da ferramenta desenvolvida podem ser visualizadas no anexo A deste documento. Todas as funcionalidades foram implementadas, porém algumas dificuldades foram encontradas. Foi necessário alterar a abordagem MUSA, conforme descrito na seção anterior, visando a melhora no desempenho da ferramenta e diminuição do consumo de memória. Mesmo com as alterações efetuadas, a funcionalidade ainda apresenta demasiado consumo de memória para as sequências gênicas, devido sua natureza bastante ruidosa. A conversão para séries temporais através da propriedade curvatura também apresentou demasiada lentidão. Neste caso o motivo não foi averiguado, pois ocorreu durante a execução do script de curvatura (ferramenta de terceiro, conforme seção 4.1). Foi necessário dividir o arquivo original das sequências gênicas em 5 partes,

após convertê-los um de cada vez e juntá-los novamente para poder utilizá-las na etapa de clusterização.

Algumas funcionalidades que não estavam no escopo inicial foram adicionadas a ferramenta a fim de facilitar a experiência do usuário e eliminar o retrabalho, como é o caso da possibilidade de salvar o resultado da execução da ATE e da MUSA em um arquivo adicional para que possam ser reutilizados, eliminando a necessidade de aguardar o tempo de execução a cada novo teste. Estes resultando também podem ser analisados posteriormente por um profissional da área.

5 RESULTADOS E DISCUSSÕES

Neste capítulo serão apresentados os resultados proporcionados pela ferramenta desenvolvida. A fim de comprovar a eficácia dos métodos implementados e a usabilidade da ferramenta, uma bateria de testes foi realizada, tendo como objetivo a separação das sequências promotoras, terminadoras e de genes em *clusters* diferentes. Os resultados foram obtidos através da realização de simulações para cada uma das abordagens implementadas. Na etapa de clusterização, a equação Euclidiana foi adotada em todos os testes para o cálculo da distância entre os objetos, pois conforme resultados do trabalho de FONTANA (2013), independente da equação utilizada, os resultados eram semelhantes. A configuração da máquina utilizada foi: processador *Intel(R) Core(TM) i7 @ 4.00GHz*, memória: *DDR3, 16 GBytes*. Os tempos de execução foram eliminados de algumas tabelas para melhorar a visualização dos resultados.

5.1 SIMULAÇÕES UTILIZANDO ATE

Inicialmente foram realizados 12 testes utilizando a ATE na extração de atributos, variando o grau de expansão entre 0,1 e 0,3 para todas as sequências com um tamanho de 2n para os nucleotídeos. Após analisar os resultados, pôde-se constatar que os *clusters* formados não estavam de acordo com o esperado, conforme demonstrado na tabela 3.

Tabela 3 – Resultados dos primeiros testes utilizando ATE

(continua)

					Quantidade de objetos por cluster								
Teste	**Grau de Expansão**	**Configuração Cluster**	**Qtd. Clusters**	**Sequência**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
1	0,1	Primeiro em cada arquivo	3	Gene	458	2689	1488						
				Promotor			8457						
				Terminador			261						
2		Randômico entre todos os arquivos	3	Gene	458	2689	1488						
				Promotor			8457						
				Terminador			261						
3		Randômico entre todos os arquivos	5	Gene	1763	1498	114	829	431				
				Promotor				8457					
				Terminador				261					
4		Randômico entre todos os arquivos	9	Gene	44	157	621	311	817	866	724	278	817
				Promotor				8457					
				Terminador				261					
5	0,2	Primeiro em cada arquivo	3	Gene	468	2710	1457						
				Promotor			8457						
				Terminador			261						
6		Randômico entre todos os arquivos	3	Gene	2710	1457	468						
				Promotor		8457							
				Terminador		261							

(conclusão)

					Quantidade de objetos por cluster								
Teste	**Grau de Expansão**	**Configuração Cluster**	**Qtd. Clusters**	**Sequência**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
7	0,2	Randômico entre todos os arquivos	5	Gene	774	1516	470	125	1750				
				Promotor	8457								
				Terminador	261								
8		Randômico entre todos os arquivos	9	Gene	817	863	724	44	157	821	309	278	621
				Promotor							8456		
				Terminador							260		
9	0,3	Primeiro em cada arquivo	3	Gene	482	1393	2760						
				Promotor		8457							
				Terminador		261							
10		Randômico entre todos os arquivos	3	Gene	2760	1393	482						
				Promotor		8457							
				Terminador		261							
11		Randômico entre todos os arquivos	5	Gene	1581	198	807	1498	551				
				Promotor					8457				
				Terminador					261				
12		Randômico entre todos os arquivos	9	Gene	44	863	765	157	280	809	806	630	280
				Promotor					8456				
				Terminador					260				

Fonte: O autor (2015).

Na tentativa de melhorar o agrupamento dos *clusters*, foi efetuada uma nova etapa de testes, na qual foram configurados diferentes graus de expansão para cada tipo de sequência, mantendo a mesma quantidade de *clusters*. Ao analisar os novos resultados, foi constatado que o grau de expansão ideal varia de acordo com o tipo da sequência, pois a frequência dos nucleotídeos é diferente entre cada uma delas. Os agrupamentos encontrados variaram consideravelmente, conforme o grau de expansão era alterado. A tabela 4 demonstra os resultados da clusterização após efetuar os novos testes. Se o grau de expansão for configurado acima de 0,47 para as sequências promotoras, 0,53 para as terminadoras 0,94 para as gênicas, os resultados são estabilizados, como pode-se verificar nos testes 8, 9 e 10. Quanto maior o grau de expansão, menor é a quantidade de pesos de elementos que são aumentados, sendo que após chegar ao limite (diferente para cada tipo de sequência), o efeito da ATE é nulo. Portanto, quanto menor o grau da expansão, maior é o efeito da ATE sobre as sequências, pois diminui o limite de corte para que a expansão seja realizada.

Tabela 4 – Resultados da segunda etapa de testes utilizando ATE

(continua)

				Quantidade de objetos por cluster		
Teste	**Conf. Cluster**	**Grau de Expansão**	**Sequência**	**C1**	**C2**	**C3**
1	Primeiro em cada arquivo	0,4	Gene	456	2685	1494
		0,1	Promotor			8457

(conclusão)

				Quantidade de objetos por cluster		
Teste	**Conf. Cluster**	**Grau de Expansão**	**Sequência**	**C1**	**C2**	**C3**
1	Primeiro em cada arquivo	0,25	Terminador			261
2		0,92	Gene	1496		3139
		0,1	Promotor		8457	
		0,25	Terminador		8	253
3		0,3	Gene	456	2685	1494
		0,1	Promotor			8457
		0,25	Terminador			261
4		0,93	Gene	2680	1954	
		0,44	Promotor		8457	
		-	Terminador			261
5		0,91	Gene	2512	430	1693
		0,1	Promotor			8457
		0,25	Terminador			261
6		0,92	Gene	1359	7	3269
		0,1	Promotor		8457	
		0,1	Terminador		261	
7		0,95	Gene	4635		
		0,45	Promotor	149	4069	4239
		0,3	Terminador	242	2	17
8		0,94	Gene	820	1231	2584
		0,47	Promotor		8457	
		0,53	Terminador		261	
9		0,95	Gene	4284	350	1
		0,48	Promotor	2633	5824	
		0,54	Terminador	15	7	239
10		0,96	Gene	4284	350	1
		0,49	Promotor	2633	5824	
		0,55	Terminador	15	7	239

Fonte: O autor (2015).

Adicionalmente, foram realizados testes com os tamanhos 3n e 4n. Notou-se que quanto maior for o tamanho dos nucleotídeos, menor é o grau de correlação entre os termos, e consequentemente, menor é a quantidade de termos que possuem seus pesos elevados. A tabela 5 demonstra essa realidade. Foram selecionados os 10 termos que tiveram a maior elevação de peso com o grau de expansão em 0,01 (menor possível) e em 0,1. Na primeira execução, todas as sequências geraram ao menos 10 termos com pesos elevados. Na segunda, com o grau de expansão em 0,1 (apenas 0,9 graus a mais), somente as sequências de genes geraram termos com pesos elevados em todas as configurações. As sequências promotoras não geraram

termos elevados para os tamanhos 3n e 4n e as terminadoras não geraram para o tamanho 4n.

Tabela 5 – Resultados da ATE para tamanhos de 2, 3 e 4

			Promotor			Gene			Terminador		
Grau de expansão	Nucleotídeos	Tempo de execução	Termo	Peso original	Peso Aumentado	Termo	Peso original	Peso Aumentado	Termo	Peso original	Peso Aumentado
0,01	2n	2m	AA	8398	53747	AA	4625	68432	AA	254	1152
			AC	8358	37416	AC	4631	63907	AC	239	646
			AT	8431	48369	AT	4634	66561	AT	238	744
			AG	8284	34933	AG	4629	62083	AG	238	727
			CA	8404	41390	CA	4633	66396	CA	238	774
			CC	7934	29289	CC	4627	60964	CC	254	836
			CT	8355	37933	CT	4634	64455	CT	229	652
			CG	8240	39298	CG	4632	67779	CG	246	866
			TA	8338	40651	TA	4631	59423	TA	237	684
			TC	8386	36601	TC	4634	62487	TC	230	660
	3n	1h	AAA	7009	26640	CTG	4591	78433	AAA	204	698
			TTT	6421	23727	GCG	4537	78214	TTT	177	530
			ATT	7031	21062	CGC	4537	74135	GCC	195	341
			AAT	6904	20314	AAA	4503	74131	CGG	164	320
			TGA	7054	19656	TGG	4576	72435	CCG	172	318
			CTG	6715	19104	GGC	4555	65295	AAG	174	314
			TAA	6681	19053	GCA	4571	65248	GCG	161	314
			TTA	6687	18998	TGC	4538	63770	GGC	181	314
			TAT	6580	18865	GAA	4572	63672	CGC	159	305
			GAA	6657	18308	GCT	4554	61842	TAA	166	302
	4n	12h	AAAA	3879	6222	CTGG	4368	40608	AAAA	149	261
			TTTT	3314	5083	GCTG	4322	37116	TTTT	129	225
			AAAT	4081	4673	TGGC	4322	35074	AAAG	109	131
			ATTT	3866	4393	GGCG	4207	33783	TAAA	96	119
			TGAA	3875	4247	AAAA	4264	33226	GAAA	89	112
			TTAA	3454	3992	GCGC	4153	30748	CTTT	84	106
			TAAA	3776	3989	TGAA	4363	26864	AAAC	85	104
			TTAT	3580	3983	GAAA	4337	26756	GCCG	87	104
			GAAA	3695	3898	GCCG	4130	26477	CCGG	72	103
			TTTA	3593	3863	TGGT	4243	26266	TTTA	86	100
0,1	2n	2m	AA	8398	53747	GC	4634	68972	AA	254	1129
			TT	8373	50756	TG	4631	68788	TT	249	990
			AT	8431	48369	AA	4625	68432	GC	251	881
			TG	8446	46300	CG	4632	67779	CG	246	842
			CA	8404	41390	AT	4634	66561	CC	254	825
			GC	8343	41292	TT	4626	66544	CA	238	774
			TA	8338	40651	CA	4633	66396	GG	250	733
			CG	8240	39298	GA	4635	65791	TG	239	719
			GA	8381	38908	GT	4633	65056	AT	238	665
			GT	8398	38314	CT	4634	64455	GA	237	659
	3n	1h	-	-	-	GCG	4537	77842	AAA	204	275
			-	-	-	CTG	4591	77606	TTT	177	238
			-	-	-	CGC	4537	73801	GTT	137	151
			-	-	-	TGG	4576	71330	ATA	109	120
			-	-	-	AAA	4503	73752	-	-	-
			-	-	-	GCA	4571	64496	-	-	-
			-	-	-	GGC	4555	64217	-	-	-
			-	-	-	TGC	4538	63523	-	-	-
			-	-	-	GAA	4572	62410	-	-	-
			-	-	-	GCT	4554	60589	-	-	-
	4n	12h	-	-	-	TGGC	4322	4775	-	-	-
			-	-	-	GGCG	4207	4648	-	-	-

Fonte: O autor (2015).

5.2 SIMULAÇÕES UTILIZANDO M.U.S.A

No início dos testes com a MUSA foi identificado um elevado consumo de memória durante a análise das sequências gênicas. Este fato pode ser explicado devido a alta complexidade destas sequências, tanto em tamanho, quanto em estrutura. A análise do arquivo de sequências gênicas com elementos de tamanho 2 consumiu aproximadamente 6GB de memória RAM, sendo necessário que o programa fosse executado em um ambiente com arquitetura de 64 bits, devido a limitação da linguagem C# em ambientes 32 bits, onde a quantidade máxima de memória alocada é de 2GB.

Ao analisar os resultados advindos das sequências gênicas, ficou ainda mais evidente a complexidade das mesmas. Para um total de 4635 sequências, 816 delas possuíram a pontuação máxima (136) para o tamanho 2. Isto significa que 816 sequências continham todas as configurações mais comuns para todas as combinações de elementos possíveis, sendo 16 (2^4) elementos de tamanho 2 multiplicados entre si, totalizando em 256 (16*16) combinações. A fim de eliminar otimizar o desempenho e reduzir cálculos computacionais desnecessários, foram removidas as 120 combinações inversas, pois não são relevantes para a identificação dos motivos, totalizando em uma redução de 256 para 136 possíveis combinações. A tabela 6 demonstra os resultados desta execução. Devido a grande complexidade das sequências gênicas, os testes somente foram efetuados com elementos de tamanho 2, pois o tempo de execução estendeu-se além do considerado viável.

Tabela 6 – Resultados do MUSA com elementos de tamanho 2

(continua)

Tipo	Sequências com maior pontuação		Combinações com maior número de ocorrências			
	Sequência	Pontuação	Primeiro Elemento	Segundo Elemento	Ocorrências	Distância
Gene	ECK120000987	136	CT	TG	4591	1
	ECK120002703	136	TT	GT	4585	-1
	ECK120001067	136	CA	TC	4580	-1
	ECK120000485	136	TG	GG	4576	1
	ECK120000130	136	AG	CA	4573	-1
	ECK120001511	136	AA	GA	4572	-1
	ECK120001512	136	CA	GC	4571	-1
	ECK120001519	136	AG	GC	4566	1
	ECK120000514	136	TG	GA	4565	1
	ECK120001521	136	AT	GA	4558	-1
	ECK125155591	82	TG	GA	7054	1

(conclusão)

Tipo	Sequências com maior pontuação		Combinações com maior número de ocorrências			
	Sequência	Pontuação	Primeiro Elemento	Segundo Elemento	Ocorrências	Distância
Promotor	ECK125156794	82	AT	TT	7031	1
	ECK125154049	82	CT	TG	6715	1
	ECK125155757	81	AA	TA	6681	-1
	ECK125156791	80	AA	GA	6657	-1
	ECK125156792	80	TT	GT	6640	-1
	ECK125156793	80	AT	TA	6580	-1
	ECK125137238	80	AT	TG	6490	1
	ECK125156789	79	AT	GA	6482	-1
	ECK120010228	79	TG	GC	6454	1
Terminador	ECK120033263	87	CC	GC	195	-1
	ECK120014414	81	AA	TA	166	-1
	ECK120033264	81	CG	GG	164	1
	ECK125160654	66	CG	GC	161	-1
	ECK120029600	66	GC	GG	154	-1
	ECK120010816	64	AA	GA	152	-1
	ECK125108723	63	AA	AA	149	2
	ECK120010857	60	AT	TT	148	1
	ECK120015457	60	AG	GC	147	1
	ECK120033739	59	CT	TG	147	1

Fonte: O autor (2015).

Na etapa de clusterização, foram realizados os mesmos testes demonstrados na tabela 4, porém com a utilização do resultado da MUSA para a escolha dos centroides iniciais, ou seja, os centroides iniciais serão a sequência mais comum de cada tipo de sequência. Os resultados são demonstrados na tabela 7. Comparando os resultados de ambas as tabelas, não são observadas grandes diferenças, somente o teste número 3 resultou em um agrupamento diferente, juntando as sequências terminadores com as promotoras no teste com o MUSA.

Tabela 7 – Resultados da clusterização realizadas com o MUSA na escolha dos centroides iniciais

(continua)

				Quantidade de objetos por cluster		
Teste	Conf. Cluster	Grau de Expansão	Sequência	C1	C2	C3
1	Sequência mais comum (MUSA) em cada arquivo	0,4	Gene	456	2685	1494
		0,1	Promotor			8457
		0,25	Terminador			261
2		0,92	Gene	1359	67	3209
		0,1	Promotor		8457	
		0,25	Terminador		261	

(conclusão)

				Quantidade de objetos por cluster		
Teste	**Conf. Cluster**	**Grau de Expansão**	**Sequência**	**C1**	**C2**	**C3**
3		0,3	Gene	456	2685	1494
		0,1	Promotor			8457
		0,25	Terminador			261
4		0,93	Gene	820	2584	1231
		0,44	Promotor			8457
		-	Terminador			261
5		0,91	Gene	430	2513	1692
		0,1	Promotor			8457
		0,25	Terminador			261
6		0,92	Gene	1359	3269	7
		0,1	Promotor			8457
		0,1	Terminador			261
7		0,95	Gene	4635		
		0,45	Promotor	149	4069	4239
		0,3	Terminador	242	2	17
8		0,94	Gene	820	2584	1231
		0,47	Promotor			8457
		0,53	Terminador			261
9		0,95	Gene	4284	350	1
		0,48	Promotor	2635	5822	
		0,54	Terminador	15	7	239
10		0,96	Gene	4284	350	1
		0,49	Promotor	2635	5822	
		0,55	Terminador	15	7	239

Fonte: O autor (2015).

Foram realizados outros testes com a aplicação da MUSA sem a ATE, porém os resultados foram os mesmos dos encontrados sem a MUSA. Além disso, foram efetuados testes com o MUSA para as sequencias de tamanho 3 e 4 para as sequências promotoras e terminadoras, porém os resultados também não demonstraram melhoras significativas.

5.3 SIMULAÇÕES UTILIZANDO SÉRIES TEMPORAIS

Inicialmente foram realizados testes com a propriedade estabilidade, convertendo as sequências com nucleotídeos de tamanhos 2, 3 e 4. Os resultados destas configurações estão demonstrados na tabela 8. Pode-se observar que a formação dos *clusters* não atingiu o resultado esperado. Apenas as sequencias de terminadores foram divididas em outros *clusters*, enquanto as promotoras e gênicas permaneceram agrupadas em um mesmo *cluster*. Segundo BEYER (2003) e

MORCHEN (2005), este fato pode ser explicado devido a utilização da equação Euclidiana para cálculo da distância entre duas séries temporais, onde a grande dimensionalidade produzida utilizando todos os pontos da série temporal, faz com que o conceito de vizinho mais próximo seja insignificante. Em outras palavras, o contraste nas distâncias entre diferentes pontos se torna inexistente, ao menos que sejam utilizadas séries temporais muito pequenas.

Tabela 8 - Resultados da clusterização com a propriedade estabilidade

						Quantidade de objetos por cluster								
Teste	**Nucleotídeos**	**Tempo de Execução**	**Conf. Cluster**	**Qtd. Clusters**	**Sequências**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
1	2	3m	Primeiro em cada arquivo	3	Gene			4635						
					Promotor			8457						
					Terminador	41	11	209						
2		3m	Randômico em cada arquivo	3	Gene			4635						
					Promotor			8457						
					Terminador	45	16	200						
3		5m	Randômico entre todos os arquivos	3	Gene			4634						
					Promotor			8456						
					Terminador	45	16	199						
4		8m	Randômico entre todos os arquivos	9	Gene									4635
					Promotor									8457
					Terminador	3	33	15	11	4	4	22	13	156
5	3	3m	Primeiro em cada arquivo	3	Gene			4635						
					Promotor			8457						
					Terminador	42	16	203						
6		3m	Randômico em cada arquivo	3	Gene			4635						
					Promotor			8457						
					Terminador	42	16	203						
7		5m	Randômico entre todos os arquivos	3	Gene				4635					
					Promotor				8457					
					Terminador	6	38	18	199					
8		8m	Randômico entre todos os arquivos	9	Gene									4635
					Promotor									8457
					Terminador	8	4	1	4	28	15	19	10	172
9	4	3m	Primeiro em cada arquivo	3	Gene			4635						
					Promotor			8457						
					Terminador	44	11	206						
10		3m	Randômico em cada arquivo	3	Gene			4635						
					Promotor			8457						
					Terminador	46	13	202						
11		5m	Randômico entre todos os arquivos	3	Gene			4634						
					Promotor			8456						
					Terminador	13	46	201						
12		8m	Randômico entre todos os arquivos	9	Gene									4635
					Promotor									8457
					Terminador	9	3	13	5	29	16	8	5	173

Fonte: O autor (2015).

Após foram realizados testes com a propriedade curvatura com diversas configurações de *clusters*, no modelo *Trifonov*, conforme demonstrado na tabela 9. Pode-se observar que a formação dos *clusters* foi realizada de forma semelhante aos

testes anteriores, com a propriedade estabilidade. Praticamente todas as sequências foram agrupadas em um mesmo *cluster*, com exceção das sequências terminadoras, que foram agrupadas em diferentes *clusters* a medida que a quantidade de *clusters* aumentava.

Tabela 9 - Resultados da clusterização com a propriedade curvatura

						Quantidade de objetos por cluster								
Teste	**Modelo**	**Tempo de Execução**	**Conf. Cluster**	**Qtd. Clusters**	**Sequencias**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
1	Trifonov	3m	Primeiro em cada Arquivo	3	Gene	3	4	4626						
					Promotor			8457						
					Terminador	52	55	154						
2		3m	Randômico em cada arquivo	3	Gene	5	3	4625						
					Promotor			8457						
					Terminador	86	53	122						
3		3m	Randômico entre todos os arquivos	3	Gene	5	3	4625						
					Promotor			8457						
					Terminador	86	53	122						
4		3m	Randômico entre todos os arquivos	5	Gene	2	4	4	3	4619				
					Promotor					8456				
					Terminador	18	36	53	42	111				
5		6m	Randômico entre todos os arquivos	7	Gene	1	4	4		4		4619		
					Promotor							8456		
					Terminador	34	53	12	17	35	33	76		
6		7m	Randômico entre todos os arquivos	9	Gene	3	3		1	2	1	1	4	4617
					Promotor									8456
					Terminador	14	22	17	29	32	32	18	24	72

Fonte: O autor (2015).

Por fim, foram realizados testes com a extração de atributos de séries temporais, especificamente com a aplicação da DWT. A quantidade de energia preservada pela DWT está totalmente ligada a quantidade de coeficientes que serão mantidos na etapa de extração. Portanto, a fim de simular a aplicação da wavelet em diferentes níveis de resolução, optou-se por utilizar coeficientes expoentes de 2. Foram realizados testes com 8, 16, 32 e 64 coeficientes. A tabela 10 demonstra os resultados com a propriedade estabilidade. Pode-se observar que, em todos os testes realizados, as sequências foram separadas de forma semelhantes entre os três *clusters*, eliminando características específicas de cada sequência. Estes resultados podem ser uma indicação de que a propriedade estabilidade é expressa de forma semelhante nos três tipos de sequência.

Tabela 10 - Resultados da clusterização com a propriedade estabilidade e DWT aplicada

						Quantidade de objetos por cluster								
Teste	**Coef.**	**Tempo de Execução**	**Config. Cluster**	**Quant. Clusters**	**Sequencias**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
1	8	3m	Primeiro em cada arquivo	3	Gene	1866	308	2461						
					Promotor	2883	2795	2779						
					Terminador	118	59	84						
2		3m	Randômico em cada arquivo	3	Gene	2460	1866	309						
					Promotor	2780	2886	2791						
					Terminador	84	118	59						
3		3m	Randômico entre todos os arquivos	5	Gene	309	2460	1866						
					Promotor	2791	2780	2886						
					Terminador	59	84	118						
4		3m		9	Gene	539	960	351	241	30	350	93	1095	976
					Promotor	997	802	1071	906	1054	941	1109	695	882
					Terminador	17	13	10	6	68	26	16	41	64
5	16	5m	Primeiro em cada arquivo	3	Gene	2463	1509	662						
					Promotor	2245	2606	3605						
					Terminador	83	107	70						
6		5m	Randômico em cada arquivo	3	Gene	773	2568	1293						
					Promotor	3643	2046	2767						
					Terminador	73	91	96						
7		6m	Randômico entre todos os arquivos	5	Gene	2568	1292	774						
					Promotor	2044	2771	3641						
					Terminador	89	98	73						
8		9m		9	Gene	599	417	653	761	757	169	144	897	237
					Promotor	802	960	854	663	730	1220	1401	646	1180
					Terminador	34	18	55	44	28	23	22	16	20
9	32	9m	Primeiro em cada arquivo	3	Gene	799	2883	952						
					Promotor	2886	1962	3608						
					Terminador	46	97	117						
10		10m	Randômico em cada arquivo	3	Gene	929	2914	791						
					Promotor	3635	1938	2883						
					Terminador	113	96	51						
11		5m	Randômico entre todos os arquivos	5	Gene	922	2910	802						
					Promotor	3605	1930	2921						
					Terminador	113	94	53						
12		13m		9	Gene	266	257	452	263	575	1240	910	318	353
					Promotor	1136	1120	1087	1175	811	466	570	1127	964
					Terminador	6	13	31	35	49	48	23	24	31
13	64	5m	Primeiro em cada arquivo	3	Gene	1849	844	1942						
					Promotor	2820	2686	2951						
					Terminador	131	9	121						
14		6m	Randômico em cada arquivo	3	Gene	826	1924	1885						
					Promotor	2650	2832	2975						
					Terminador	7	139	115						
15		7m	Randômico entre todos os arquivos	5	Gene	1114	1317	2204						
					Promotor	2975	2760	2722						
					Terminador	87	53	121						
16		16m		9	Gene	412	730	342	210	241	328	349	264	1758
					Promotor	856	666	952	1238	1216	1143	1011	1065	309
					Terminador	59	27	6		68	58	3	34	5

Fonte: O autor (2015).

A tabela 11 demonstra os testes realizados com a propriedade curvatura, com as mesmas configurações dos testes anteriores. Observando os resultados, pôde-se identificar que os *clusters* foram agrupados em um padrão diferentes dos testes anteriores, onde a maioria das sequências promotoras foram alocadas em *clusters* separados das gênicas e terminadoras. Esse comportamento indica que a propriedade curvatura nas sequências promotoras é expressada de forma diferenciada, reforçando o que foi identificado no trabalho de SILVA *et al*, 2011.

Tabela 11 - Resultados da clusterização com a propriedade curvatura e a DWT aplicada

(continua)

						Quantidade de objetos por cluster								
Teste	**Coef.**	**Tempo de Execução**	**Conf. Cluster**	**Qtd. Clusters**	**Sequências**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
1	8	1m20s	Primeiro em cada arquivo	3	Gene	4626		6						
					Promotor	110	2767	5579						
					Terminador	206	34	20						
2		1m20s	Randômico em cada arquivo	3	Gene	6		4626						
					Promotor	5579	2767	110						
					Terminador	20	34	206						
3		2m22s	Randômico em cada arquivo	3	Gene	6		4626						
					Promotor	5579	2767	110						
					Terminador	20	34	206						
4		3m	Randômico em cada arquivo	9	Gene	1421			1421	1423			2	365
					Promotor		2084	1388		10	1092	1444	2432	6
					Terminador	25	42	5	28	116			3	41
5	16	1m20s	Primeiro em cada arquivo	3	Gene	4606	2	24						
					Promotor	155	2835	5466						
					Terminador	240		20						
6		1m20s	Randômico em cada arquivo	3	Gene	4605	2	25						
					Promotor	154	2831	5471						
					Terminador	240		20						
7		2m	Randômico em cada arquivo	3	Gene	25	4605	2						
					Promotor	5472	154	2830						
					Terminador	20	240							
8		3m	Randômico em cada arquivo	9	Gene	1211			820		1221		39	1341
					Promotor	4	1126	1543	3	1580		1722	2456	22
					Terminador	30			8	21	30	2	2	167
9	32	3m	Primeiro em cada arquivo	3	Gene	10		4622						
					Promotor	5537	2770	149						
					Terminador	8	1	251						
10		4m	Randômico em cada arquivo	3	Gene	4622		10						
					Promotor	149	2770	5537						
					Terminador	251	1	8						
11		4m	Randômico em cada arquivo	3	Gene		4622	10						
					Promotor	2770	149	5537						
					Terminador	1	251	8						
12		4m	Randômico em cada arquivo	9	Gene			4596			1	34	1	
					Promotor	899	812	35	954	1141	1085	1463	1261	806
					Terminador	1		232	4	3		7	13	

(conclusão)

						Quantidade de objetos por cluster								
Teste	**oef.**	**Tempo de Execução**	**Conf. Cluster**	**Qtd. Clusters**	**Sequências**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
13	64	3m44s	Primeiro em cada arquivo	3	Gene		4570	62						
					Promotor	2788	86	5582						
					Terminador		197	63						
14		3m33	Randômico em cada arquivo	3	Gene	2176	15	2441						
					Promotor	1	8131	324						
					Terminador	32	22	206						
15		4m	Randômico em cada arquivo	3	Gene	1	4606	25						
					Promotor		244	8212						
					Terminador		238	22						
16		6m	Randômico em cada arquivo	9	Gene	131		1270			1249			1982
					Promotor	1471	1242		1403	1534	3	1351	1446	6
					Terminador	190		7	3		9	2	1	48

Fonte: O autor (2015).

Poderão ser realizados mais testes com a propriedade de curvatura, utilizando outros modelos de cálculo que estão disponíveis na ferramenta e com variações na quantidade de coeficientes a fim de obter novos resultados, ou confirmar os anteriores.

5.4 CONSIDERAÇÕES FINAIS

Ao término dos testes, foi possível identificar que o tempo de execução dos métodos aumenta exponencialmente a medida que a quantidade de nucleotídeos e *clusters* é aumentada. Os resultados encontrados nos testes realizados não atingiram os objetivos esperados. Sobre as sequências em linguagem temporal, embora a ATE e a MUSA cumpriram com sua finalidade, a primeira em aumentar o peso dos termos com maior correlação e a segunda em encontrar as sequências mais comuns, ambas as técnicas não apresentaram melhorias na performance do algoritmo de clusterização. Não foram gerados *clusters* distintos para cada tipo de sequência. Ainda, entende-se que mais testes possam ser realizados com a abordagem ATE, pois houve dificuldade em determinar qual é a configuração ideal para cada sequência devido a grande sensibilidade ao grau de expansão.

Sobre as sequências no formato de séries temporais, a aplicação da DWT atingiu o objetivo de suavizar as sequências e trouxe resultados interessantes na etapa de clusterização, porém também não agrupou os tipos de sequências em *clusters* distintos. Pode-se constatar que a técnica de clusterização é uma ferramenta com grande potencial na identificação de padrões para sequências em séries temporais.

6 CONCLUSÃO

O estudo realizado neste projeto de pesquisa contribuiu para o aumento do conhecimento na área da bioinformática, desde os conceitos biológicos até as técnicas de mineração de dados advindos do sequenciamento de DNA. Pôde-se constatar que os dados de natureza biológica são extremamente complexos e analisá-los manualmente é uma tarefa praticamente impossível. Novas ferramentas computacionais projetadas para cumprir este desafio são desenvolvidas e otimizadas de uma maneira extremamente rápida e cada vez mais as facilidades apresentadas pela bioinformática vêm proporcionando resultados satisfatórios.

A grande quantidade de dados gerados diariamente pelos modernos sequenciadores de DNA resulta na criação e no crescimento de bases de dados públicas, por este motivo, a mineração de dados é um tema que vem crescendo nos últimos anos. Metodologias para gerir toda essa informação da maneira mais eficiente possível são necessárias, fazendo com que as técnicas de clusterização sejam otimizadas constantemente. Devido a complexidade dos dados de natureza biológica, métodos de extração e seleção de atributos são necessários para transformar os dados originais em representações que possam ser mensuradas e analisadas com as ferramentas disponíveis, visando o desempenho e a confiabilidade dos resultados. A aplicação das técnicas abordadas nos experimentos dos autores citados neste trabalho trouxe resultados satisfatórios na análise de padrões e identificação de motivos das regiões promotoras da expressão gênica, essas tarefas são cruciais na análise genômica e a aplicação de diferentes abordagens abre as portas para novas descobertas no entendimento das funções dos genes, bem como o descobrimento das causas das enfermidades e o auxílio no diagnóstico e tratamento das mesmas.

As abordagens de seleção e extração de atributos foram implementadas e as novas funcionalidades foram agregadas a ferramenta. Porém, os resultados encontrados ainda não podem ser considerados satisfatórios. As sequências foram submetidas a uma bateria de testes de transformação de atributos e clusterização, em diversos tipos de configurações. Ao analisar os resultados, pôde-se observar que as sequências gênicas possuem uma grande quantidade de informações ruidosas, fazendo com que a técnica de clusterização não seja tão eficiente quando é utilizada para analisar este tipo de sequência. Além disso, ficou evidente que o número de

clusters (k) é um fator com grande relevância sobre o resultado final, porém o valor ideal ainda não pôde ser definido.

Constatou-se que a abordagem ATE cumpre com a sua proposta, elevando os pesos dos atributos com maior grau de correlação entre as sequências analisadas, porém essa funcionalidade não acarretou em uma melhora nos resultados da etapa de clusterização. A abordagem MUSA mostrou ser eficiente na identificação dos motivos, porém devido ao seu demasiado esforço computacional, mostrou ser de pouca valia na aplicação de seu resultado para a escolha dos centroides iniciais. A utilização da mesma que não trouxe melhoras consideráveis nos testes realizados. Por fim, a extração de atributos em séries temporais com a utilização da DWT também demonstrou cumprir com o seu objetivo. Foi possível suavizar as séries temporais e melhorar consideravelmente o agrupamento dos *clusters* na etapa da clusterização. O número de coeficientes ideal não pôde ser alcançado. Mais testes deverão ser realizados a fim de identificar quanta energia (quantidade de coeficientes) é necessária armazenar a fim de obter os melhores resultados.

Os trabalhos futuros poderão contemplar a otimização do desempenho da conversão das sequências em linguagem natural para séries temporais a partir da propriedade curvatura com a aplicação de técnicas avançadas de processamento, como o paralelismo. A MUSA também poderá ser otimizada seguindo o mesmo princípio. Outras técnicas de extração de atributos em séries temporais podem ser implementadas, tais como a transformada de Fourier. Abordagens para otimização da escolha dos centroides iniciais também poderão ser exploradas, como por exemplo, o algoritmo HGKA (citado na seção 3.3) que combina elementos de algoritmos genéticos com a clusterização a fim de eliminar a dependência da inicialização, inclusive, esta técnica pode ser útil para ambos os tipos de representação.

A fim de melhorar a aplicabilidade da ferramenta, poderá ser desenvolvido um configurador de *layout* para os arquivos de entrada, fazendo com que arquivos de diferentes bases de dados possam ser importados e analisados nesta ferramenta. Com o mesmo objetivo, poderá ser implementada uma funcionalidade para plotar os resultados da etapa da clusterização em gráficos, proporcionando uma melhor interpretação dos mesmos.

7 REFERÊNCIAS

AGRAWAL, R.; FALOUTSOS, C.; SWAMI, A. Efficient similarity search in sequence databases. **In proceedings of the 4th Int'l Conference on Foundations of Data Organization and Algorithms**, 1993. 69-84.

ALOK, S. et al. **Null space based feature selection method for gene expression data**. Sharma International Journal of Machine Learning and Cybernetics, v. 3, 2012.

ANDRITSOS, P. **Data Clustering Techniques**. Toronto: Universidade de Toronto, 2002.

BAR-JOSEPH, Z. Analyzing time series gene expression data. **Bioinformatics**, 20, Abril 2004. 2493-2503.

BATAL, I.; HAUSKRECHT, M. A Supervised Time Series Feature Extraction Technique using DCT and DWT. **International Conference on Machine Learning and Applications**, 2009. 735-739.

BENNET, J.; GANAPRAKASAM, C. A.; ARPUTHARAJ, K. A Discrete Wavelet Based Feature Extraction and Hybrid Classification Technique for Microarray Data Analysis. **The Scientific World Journal**, 6 Agosto 2014. 1-9.

BERGERON, B. P. **Bioinformatics computing**. New Jersey: Prentice Hall Professional, 2003.

BERKHIN, P. **Survey of Clustering Data Mining Techniques**. 2002.

BORÉM, A.; F. R. SANTOS. **Biotecnologia Simplificada**. Viçosa, MG: Suprema, 2001.

CLAVERIE, J.-M.; NOTREDAME, C. **Bioinformatics For Dummies**. New Jersey: Wiley Publishing, v. 2, 2007.

COSTA, I. G.; CARVALHO, F. D. A. T. D.; P, M. C. Comparative analysis of clustering methods for gene expression time course. **Genetics and Molecular Biology**, 20 Agosto 2004. 623-631.

CUNNINGHAM, P. Dimension Reduction. **Technical Report UCD-CSI-2007-7**, 08 Agosto 2007.

DAS, M. K.; DAI, H.-K. A survey of DNA motif finding algorithms. **Fourth Annual MCBIOS Conference. Computational Frontiers in Biomedicine**, Novembro 2007.

DOUGHERTY, E. R. et al. **Genomic Signal processing and Statistics**. New York: Hindawi Publishing Corporation, 2005.

EISENSTEIN; BARRY; ZALEZNIK. **Pinciples and Practice of Infectious Diseases**. 2000.

FENG, P. **Bacteriological Analytical Manual**., v. 8, 2002. Disponivel em: <http://www.fda.gov/Food/FoodScienceResearch/LaboratoryMethods/BacteriologicalAnalyticalManualBAM/ucm064948.htm>.

FONTANA, E. **Algoritmos de clusterização aplicados na análise genômica da bactéria Escherichia Coli**. Caxias do Sul: Universidade de Caxias do Sul, 2013. 74 p.

GAN, Y.; GUAN, J.; ZHOU, S. **A comparison study on feature selection of DNA structural properties for promoter prediction**. China: BioMed Central, 2012.

GRIFFITHS, A. J. F. et al. **Introdução a Genética**. New York: GUANABARA KOOGAN, v. 10, 2010. 395 p.

HANSELMAN, D.; LITTLEFIELD, B. **MATLAB 6:** curso completo. São Paulo: Pearson Prentice Hall, 2003.

HARTL, D. L.; JONES, E. W. **Genetics:** Analysis of Genes and Genomes. Jones and Bartlett, 2004. 1-133 p.

HE, J. Improving feature representation of natural language gene functional annotations using automatic term expansion. **Computer Inteligence in Bioinformatics and Computational Biology**, Setembro 2008. 173-179.

JAIN, A. K.; M.N. MURTY; P.J. FLYNN. **Data Clustering:** A Review. ACM Computing Surveys, v. 31, 1999.

JEYACHIDRA, J. **A comparative analysis of feature selection algorithms on classification of gene microarray dataset**. Thanjavur: IEEE, 2013. 1088-1093 p.

JIANG, D.; AIDONG ZHANG; CHUN TANG. Cluster Analysis for Gene Expression Data: A Survey. **Knowledge and Data Engineering**, 04 Outubro 2004. 1370 - 1386.

KORN, F.; JAGADISH, H.; FALOUTSOS, C. Efficiently supporting ad hoc queries in large datasets of time sequences. **In proceedings of the ACM SIGMOD Int'l Conference on Management of Data**, 1997. 289-300.

KRISHNA, K.; MURTY, M. Genetic K-means algorithm. **IEEE Transactions on Systems, Man and Cybernetics**, 1999. 433-439.

LESK, A. M. **Introduction to Bioinformatics**. New York: Oxford University Press, 2008. 60-63.

MARSAN, L.; SAGOT, M.-F. Algorithms for extracting structured motifs using a suffix tree with an application to promoter and regulatory site consensus identification. **J Comput Biol**, 2000. 345-362.

MENDES, N. D. et al. MUSA: a parameter free algorithm for the identification of biologically significant motifs. **Bioinformatics**, 13 Outubro 2006. 2996–3002.

MITRA, M.; AMIT SINGHAL; CHRIS BUCKLEY. Improving Automatic Query Expansion. **ACM SIGIR conference on Research and development in information retrieval**, 1998. 206-214.

MITRA, S.; ACHARYA, T. **Data Mining:** Multimedia, Soft Computing, and Bioinformatics. Hoboken: John Wiley & Sons, 2003.

MORCHEN, F. **Time series feature extraction for data mining using DWT and DFT**, Novembro 2005.

PABLO TAMAYO et al. **Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation**, 7 Março 1999. 2907–2912.

PEARSON, W. R.; LIPMAN, D. J. Improved tools for biological sequence comparison. **Proceedings of the National Academy of Sciences of the United States of America**, Abril 1988. 2444-2448.

POPIVANOV, I.; MILLER, R. J. Similarity search over time series data using wavelets. **In proceedings of the 18th Int'l Conference on Data Engineering**, 2002. 212-221.

PROSDOCIMI, F. Bioinformática: manual do usuário. In: PROSDOCIMI, F. **Biotecnologia Ciência & Desenvolvimento**, 2002. p. 12-25.

REECE, J. B. et al. **Campbell Biology:** Concepts & Connections. São Francisco: Benjamin Cummings, v. 7, 2011. 80-100 p.

SANTOS, F.; ORTEGA, J. **Bioinformática aplicada à Genômica**. Belo Horizonte: Universidade Federal de Minas Gerais, 2002.

SHAMIR, R.; SHARAN, R. **Algorithmic Approaches to Clustering Gene Expression Data**. Current Topics In Computational Biology. 2001.

VLACHOS, M. et al. A Wavelet-Based Anytime Algorithm for K-Means Clustering of Time Series. **Proceedings of the Workshop on Clustering High Dimensionality Data and Its Applications**, 2003. 22-30.

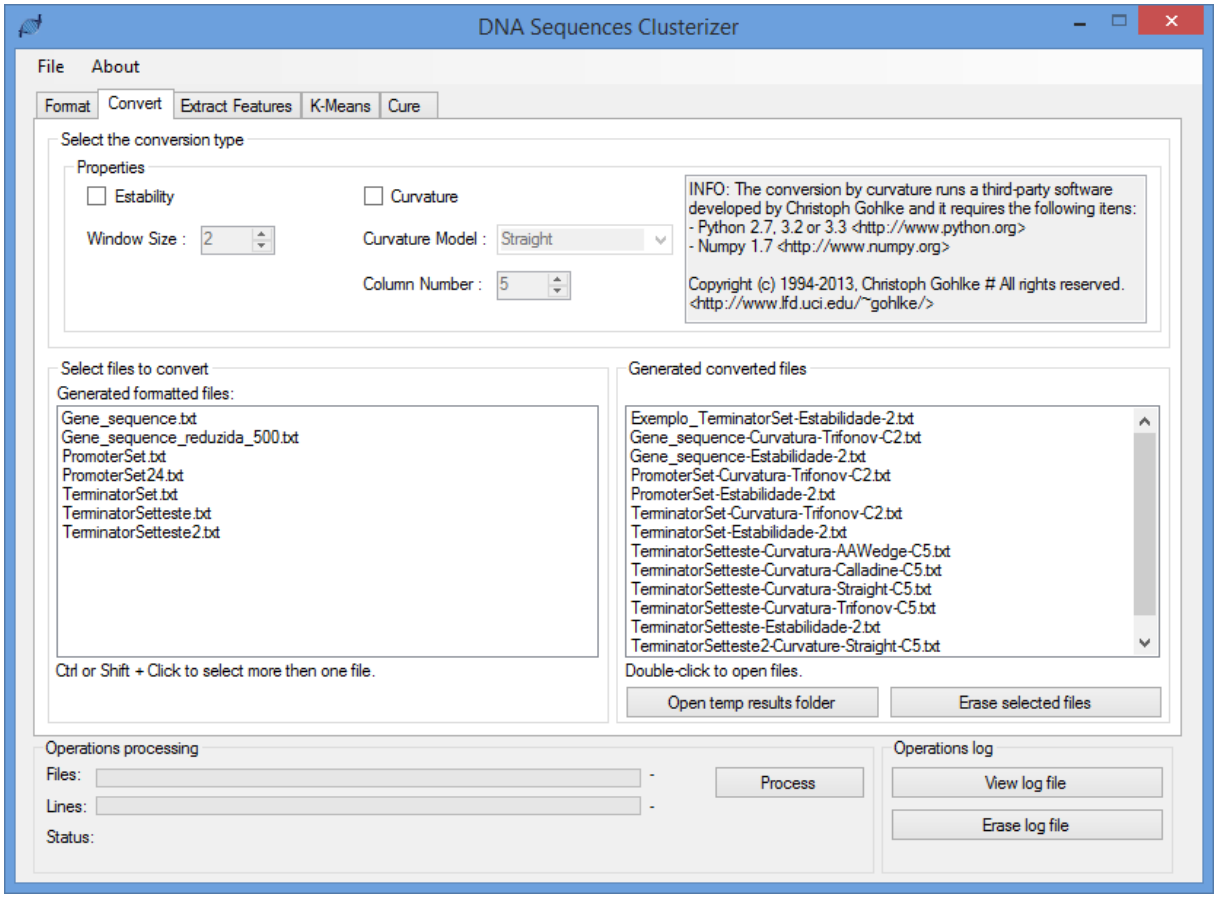
WU, Y.-L.; AGRAWAL, D.; ABBADI, A. E. A comparison of DFT and DWT based similarity search in time-series databases. **CIKM**, 2000. 488-495.

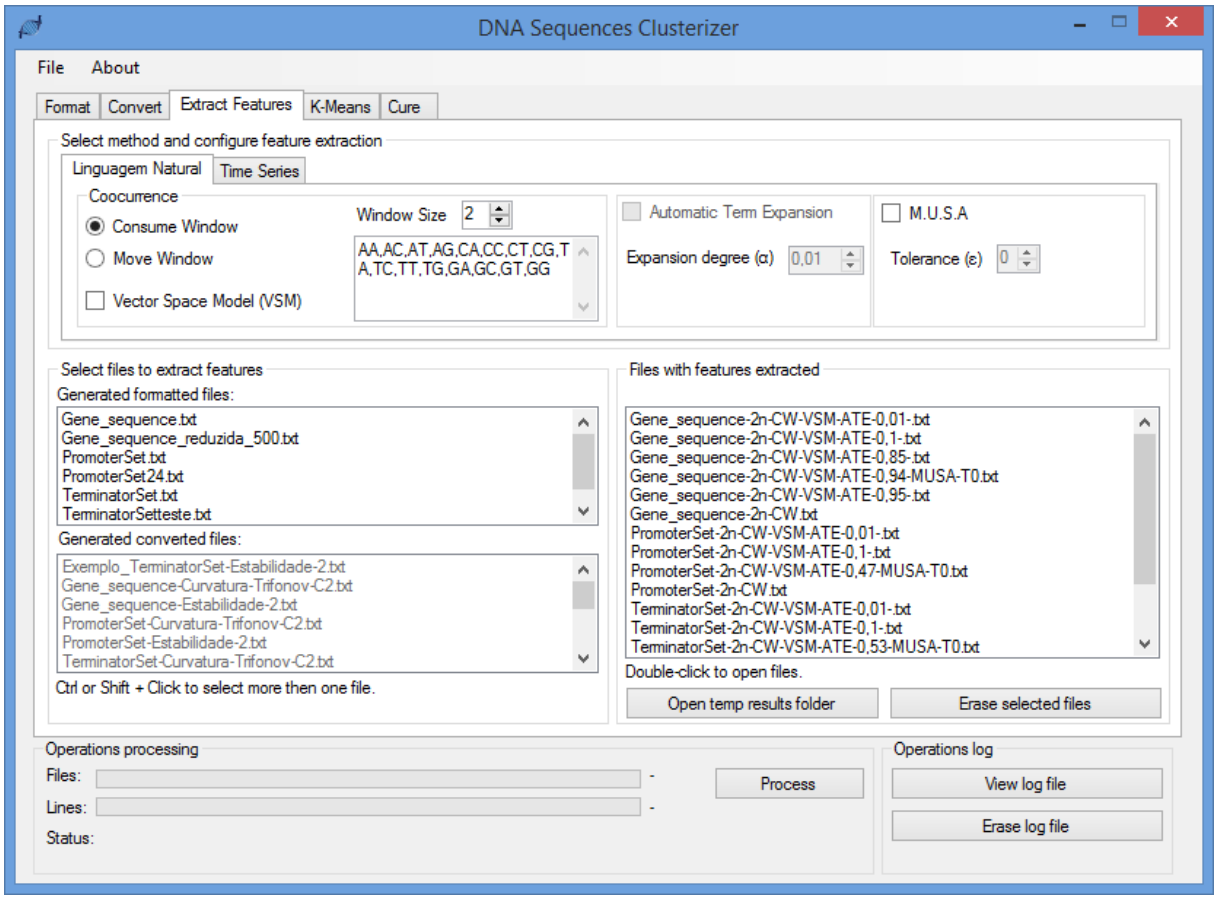
YI, B.; FALOUTSOS, C. Fast time sequence indexing for arbitrary lp norms. **In proceedings of the 26th Int'l Conference on Very Large Databases**, 2000. 385-394.

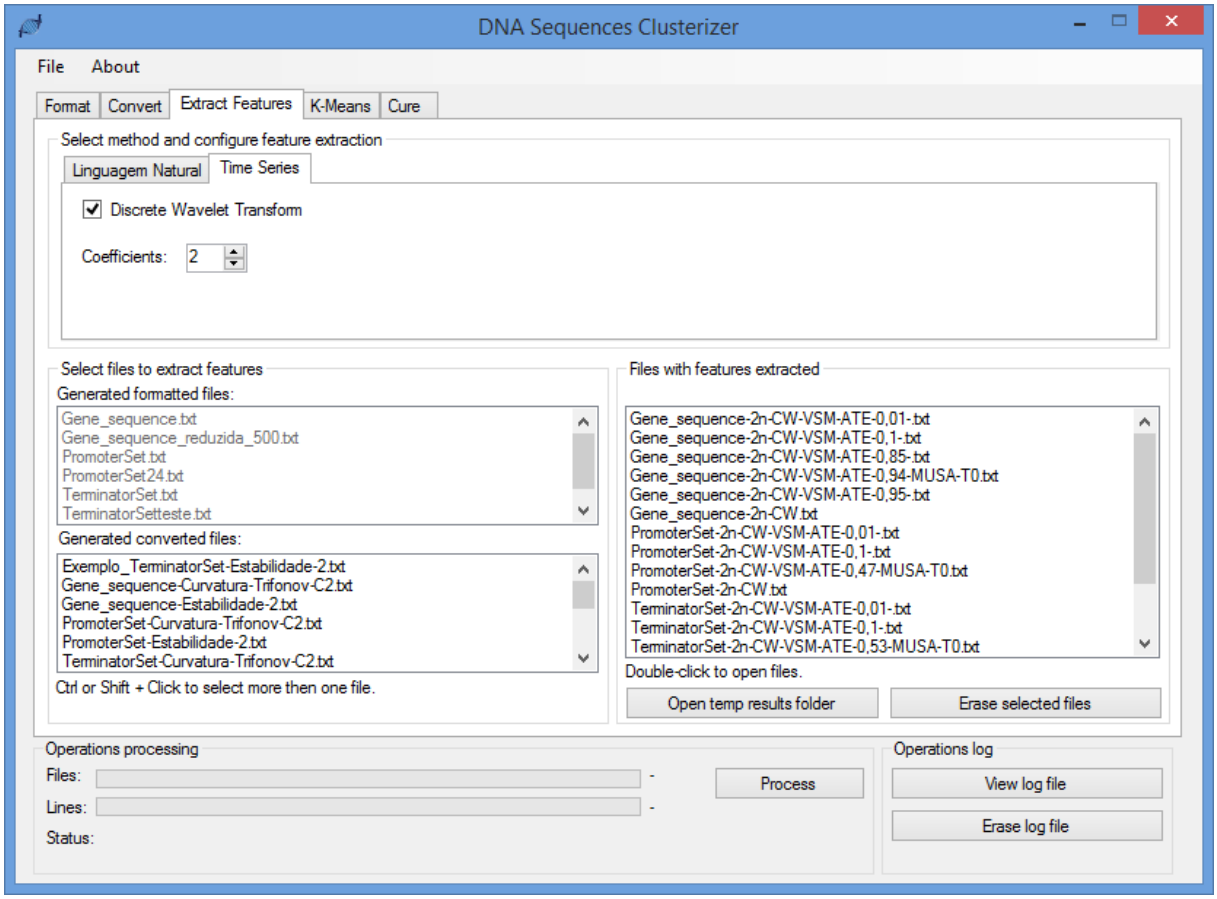
ZHANG, H. et al. Unsupervised Feature Extraction for Time Series Clustering Using Orthogonal Wavelet Transform. **Informatica**, Setembro 2005. 305-319.

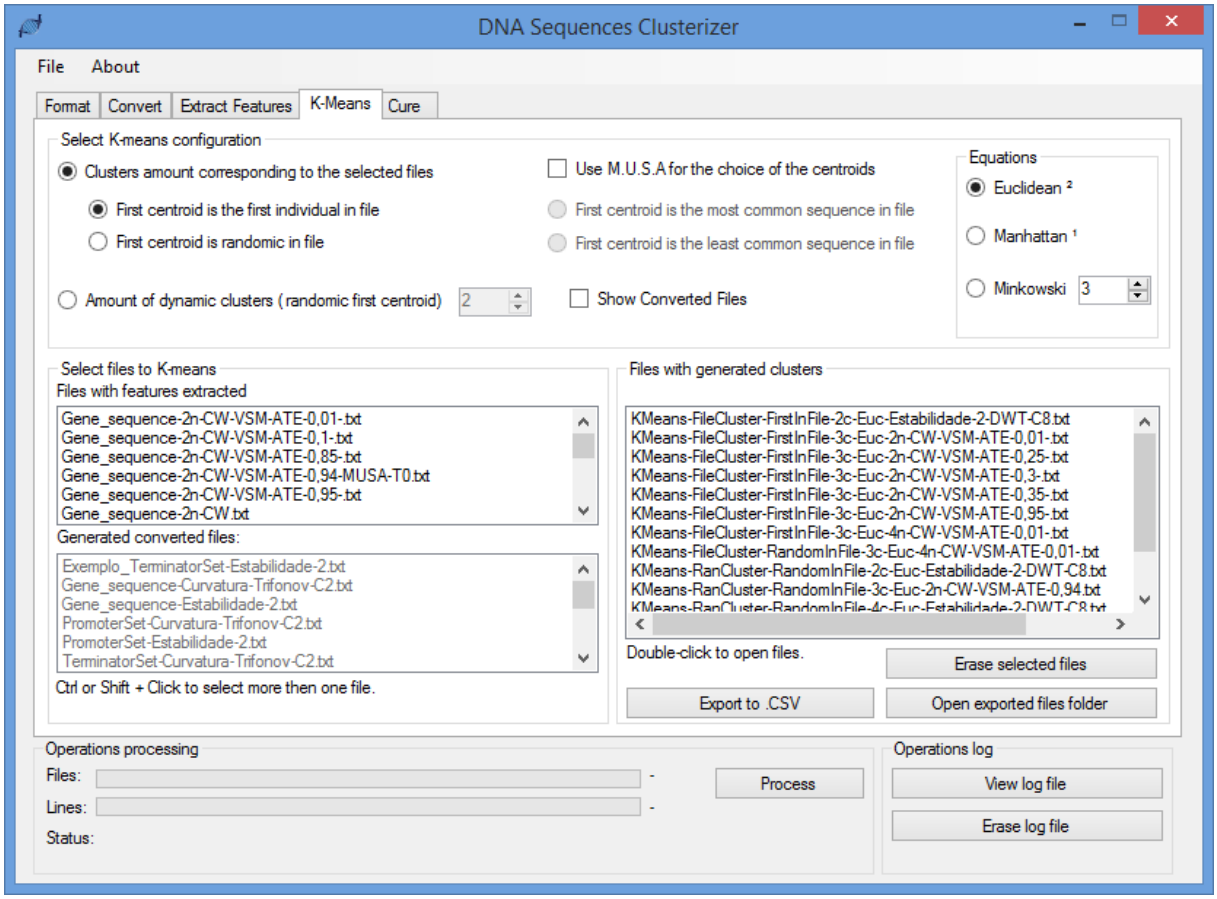
ZHENG, Y.; ESSOCK, E. A. A Novel Feature Extraction Method – Wavelet-Fourier Analysis and Its Application to Glaucoma Classification. **Proceedings of 7th Joint Conference on Information Sciences**, 2003. 672-675.

8 ANEXO A – TELAS DA FERRAMENTA DESENVOLVIDA






DNA Sequences Clusterizer
File
About
Format
Convert
Extract Features
K-Means
Cure
Select method and configure feature extraction
Linguagem Natural
Time Series
Discrete Wavelet Transform
Coefficients:
2
Select files to extract features
Generated formatted files:
Gene_sequence.txt
Gene_sequence_reduzida_500.txt
PromoterSet.txt
PromoterSet24.txt
TerminatorSet.txt
TerminatorSetteste.txt
Generated converted files:
Exemplo_TerminatorSet-Estabilidade-2.txt
Gene_sequence-Curvatura-Trifonov-C2.txt
Gene_sequence-Estabilidade-2.txt
PromoterSet-Curvatura-Trifonov-C2.txt
PromoterSet-Estabilidade-2.txt
TerminatorSet-Curvatura-Trifonov-C2.txt
Ctrl or Shift + Click to select more then one file.
Files with features extracted
Gene_sequence-2n-CW-VSM-ATE-0,01-.txt
Gene_sequence-2n-CW-VSM-ATE-0,1-.txt
Gene_sequence-2n-CW-VSM-ATE-0,85-.txt
Gene_sequence-2n-CW-VSM-ATE-0,94-MUSA-T0.txt
Gene_sequence-2n-CW-VSM-ATE-0,95-.txt
Gene_sequence-2n-CW.txt
PromoterSet-2n-CW-VSM-ATE-0,01-.txt
PromoterSet-2n-CW-VSM-ATE-0,1-.txt
PromoterSet-2n-CW-VSM-ATE-0,47-MUSA-T0.txt
PromoterSet-2n-CW.txt
TerminatorSet-2n-CW-VSM-ATE-0,01-.txt
TerminatorSet-2n-CW-VSM-ATE-0,1-.txt
TerminatorSet-2n-CW-VSM-ATE-0,53-MUSA-T0.txt
Double-click to open files.
Open temp results folder
Erase selected files
Operations processing
Files:
Lines:
Status:
Process
Operations log
View log file
Erase log file


DNA Sequences Clusterizer
File
About
Format
Convert
Extract Features
K-Means
Cure
Select K-means configuration
Clusters amount corresponding to the selected files
First centroid is the first individual in file
First centroid is randomic in file
Amount of dynamic clusters (randomic first centroid)
2
Use M.U.S.A for the choice of the centroids
First centroid is the most common sequence in file
First centroid is the least common sequence in file
Show Converted Files
Equations
Euclidean ²
Manhattan ¹
Minkowski
3
Select files to K-means
Files with features extracted
Gene_sequence-2n-CW-VSM-ATE-0,01-.txt
Gene_sequence-2n-CW-VSM-ATE-0,1-.txt
Gene_sequence-2n-CW-VSM-ATE-0,85-.txt
Gene_sequence-2n-CW-VSM-ATE-0,94-MUSA-T0.txt
Gene_sequence-2n-CW-VSM-ATE-0,95-.txt
Gene_sequence-2n-CW.txt
Generated converted files:
Exemplo_TerminatorSet-Estabilidade-2.txt
Gene_sequence-Curvatura-Trifonov-C2.txt
Gene_sequence-Estabilidade-2.txt
PromoterSet-Curvatura-Trifonov-C2.txt
PromoterSet-Estabilidade-2.txt
TerminatorSet-Curvatura-Trifonov-C2.txt
Ctrl or Shift + Click to select more then one file.
Files with generated clusters
KMeans-FileCluster-FirstInFile-2c-Euc-Estabilidade-2-DWT-C8.txt
KMeans-FileCluster-FirstInFile-3c-Euc-2n-CW-VSM-ATE-0,01-.txt
KMeans-FileCluster-FirstInFile-3c-Euc-2n-CW-VSM-ATE-0,25-.txt
KMeans-FileCluster-FirstInFile-3c-Euc-2n-CW-VSM-ATE-0,3-.txt
KMeans-FileCluster-FirstInFile-3c-Euc-2n-CW-VSM-ATE-0,35-.txt
KMeans-FileCluster-FirstInFile-3c-Euc-2n-CW-VSM-ATE-0,95-.txt
KMeans-FileCluster-FirstInFile-3c-Euc-4n-CW-VSM-ATE-0,01-.txt
KMeans-FileCluster-RandomInFile-3c-Euc-4n-CW-VSM-ATE-0,01-.txt
KMeans-RanCluster-RandomInFile-2c-Euc-Estabilidade-2-DWT-C8.txt
KMeans-RanCluster-RandomInFile-3c-Euc-2n-CW-VSM-ATE-0,94.txt
Double-click to open files.
Erase selected files
Export to .CSV
Open exported files folder
Operations processing
Files:
Lines:
Status:
Process
Operations log
View log file
Erase log file