UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Alexandre Maciel Guerra

Extensões de coclusterização para conjuntos de dados numéricos e extração de regras de associação

Campinas

2017

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Alexandre Maciel Guerra

Extensões de coclusterização para conjuntos de dados numéricos e extração de regras de associação

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica, na Área de Engenharia de Computação.

Orientador: Prof. Dr. Fernando José Von Zuben

Co-orientadores: Dra. Rosana Veroneze e Dr. Marcelo Falsarella Carazzolle

Este exemplar corresponde à versão da tese defendida pelo aluno Alexandre Maciel Guerra, e orientada pelo Prof. Dr. Fernando José Von Zuben

---

Campinas

2017

Agência(s) de fomento e nº(s) de processo(s): CAPES
ORCID: http://orcid.org/http://orcid.org/00

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Luciana Pietrosanto Milla - CRB 8/8129

Guerra, Alexandre Maciel, 1990-
G937e Extensões de coclusterização para conjuntos de dados numéricos e extração de regras de associação / Alexandre Maciel Guerra. – Campinas, SP : [s.n.], 2017.

Orientador: Fernando José Von Zuben.
Coorientador: Marcelo Falsarella Carazzolle.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Análise de dados. 2. Interface gráfica com o usuário (Sistemas de computação). I. Von Zuben, Fernando José,1968-. II. Carazzolle, Marcelo Falsarella,1975-. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Coclustering extensions in numerical datasets and association rules extraction
Palavras-chave em inglês:
Data analysis
Graphical user interface (Computer systems)
Área de concentração: Automação
Titulação: Mestre em Engenharia Elétrica
Banca examinadora:
Fernando José Von Zuben [Orientador]
Jacques Wainer
Levy Boccato
Data de defesa: 11-09-2017
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - DISSERTAÇÃO DE MESTRADO

Candidato: Alexandre Maciel Guerra RA: 104518

Data da Defesa: 11 de setembro de 2017

Título da Tese: "Extensões de coclusterização para conjuntos de dados numéricos e extração de regras de associação".

Prof. Dr. Fernando José Von Zuben (Presidente, FEEC/UNICAMP)
Prof. Dr. Jacques Wainer (IC/UNICAMP)
Prof. Dr. Levy Boccato (FEEC/UNICAMP)

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no processo de vida acadêmica do aluno.

Dedico esta tese a meus pais, minha irmã, minha noiva, minha família, meus orientadores, professores e amigos

Agradecimentos

"Aqueles que passam por nós, não vão sós, não nos deixam sós. Deixam um pouco de si, levam um pouco de nós"
(Antoine de Saint-Exupéry)

Este ano em que fiz esta pesquisa foi uma curta mas árdua jornada de desafios, construções e amadurecimento. Neste período, aprendi que uma tese é a extensão da vida de seu autor. Que para construir algo não é suficiente apenas toda minha força de vontade, mas sim todo o suporte das pessoas que estão ao meu lado. Por este motivo, agradeço sincera e profundamente a todas as pessoas que passaram pela minha vida e que me encorajaram e me auxiliaram a me tornar uma pessoa melhor.

"E aprendi que se depende sempre
De tanta, muita, diferente gente
Toda pessoa sempre é as marcas
das lições diárias de outras tantas pessoas.
É tão bonito quando a gente entende
Que a gente é tanta gente
Onde quer que a gente vá.
É tão bonito quando a gente sente
Que nunca está sozinho
Por mais que pense estar..."
(Caminhos do coração – Gonzaguinha.)

Primeiramente, eu gostaria de agradecer aos meus pais, minha irmã e minha família por estarem sempre ao meu lado, me ajudando, incentivando e mantendo meus pés no chão.

Obrigado também à minha noiva, que nesse um ano e meio juntos me acompanhou por toda a trajetória deste mestrado, sempre me ajudando, acalmando e incentivando. Obrigado por permanecer ao meu lado nos bons e nem tão bons momentos. Por sempre acreditar em mim, por me fazer uma pessoa melhor e por ser um modelo de dedicação e boa vontade.

Muito obrigado professor Von Zuben, por toda sua dedicação, paciência e por acreditar em mim. Por ter me aceitado para este projeto mesmo conhecendo todas as dificuldades por conta do curto espaço de tempo. Com certeza o senhor é um modelo de

pesquisador e professor a quem eu tenho o prazer de seguir e em quem vou procurar me espelhar.

Obrigado aos meus co-orientadores Rosana e Marcelo, por todo o auxílio e suporte que me deram durante esta jornada, desde a graduação até agora.

Obrigado a todos os integrantes do LBiC e do LGE com quem eu convivi neste período: Fernando, João, Pedro, Saullo, Marcos, Sheila, Carol por toda ajuda e por todas as conversas. Manifesto aqui também o meu enorme agradecimento a todos os professores e funcionários que estiveram presentes durante minha graduação e no mestrado, em particular os professores Eduardo Tavares, Gilmar Barreto, Gonçalo Pereira e Marcos Alcocer.

Mais uma etapa foi vencida! Sei que é apenas o começo... e que logo mais uma nova longa etapa se iniciará. E todos os aprendizados que obtive me auxiliarão e me darão força e confiança para seguir nesse caminho.

Resumo

Coclusterização é uma técnica de modelagem local de dados capaz de superar a clusterização convencional em tarefas de análise de dados, tendo como objetivo explicitar coerências internas em bases de dados multidimensionais, por meio de múltiplas clusterizações simultâneas. A primeira proposta da literatura de algoritmos enumerativos eficientes para bases de dados contendo informações numéricas mais genéricas, não necessariamente binárias, é de autoria do grupo de pesquisa em que se insere este projeto, na forma de uma família de algoritmos denominada RIn-Close. Visando explorar mais amplamente o potencial de aplicação da coclusterização para bases de dados numéricas, este projeto contribuiu propondo: (1) Concepção e implementação de uma interface gráfica para o RIn-Close (e extensível a qualquer outra técnica de coclusterização), de apoio ao usuário para o auxílio na visualização, interpretação e seleção (usando métricas de validação intra-coclusters) de múltiplos coclusters; (2) Construção de regras de associação de classes a partir de coclusters em bases de dados numéricas rotuladas, permitindo ao usuário selecionar e interpretar essas regras usando métricas específicas voltadas para regras de associação de classes. Cabe enfatizar que regras de associação para bases de dados binárias já se encontravam bem difundidas e vinculadas a soluções de alto desempenho, mas não havia, até então, contribuições relevantes na área de regras de associação de classes para o caso de bases de dados numéricas mais gerais vinculadas à coclusterização.

Palavras-chaves: coclusterização; regras de associação de classes; interface gráfica, métricas de validação.

Abstract

Coclustering is a local data modeling method capable of overcoming the conventional clustering technique in data analysis tasks. The goal is to highlight consistencies inside a multidimensional database, using multiple and simultaneous clustering. The first attempt in the literature toward an efficient enumerative algorithms for more generic numerical datasets, not necessarily binary, comes from the research group in which this work has been developed, being denoted as the RIn-Close family of algorithms. In order to increase the potential of coclustering in numerical datasets, this project contributed proposing: (1) Conception and implementation of a graphical user interface for the RIn-Close (and extensible to any other coclustering technique) to aid in the visualization, interpretation and selection (using intra-cocluster validation metrics) of multiple coclusters; (2) Creation of class association rules using coclusters mined from numerical datasets, allowing the user to select and interpret these rules using specific metrics. Association rules are well described in the literature devoted to binary datasets and generally yield high performance solutions. On the other hand, there is no major contribution for class association rules in the literature regarding generic numerical datasets related to coclustering.

Keywords: coclustering; class association rule; graphical interface; validation metrics.

"So the problem is not so much to see what nobody has yet seen, as to think what nobody has yet thought concerning that which everybody sees."

(Arthur Schopenhauer)

Lista de ilustrações

Figura 2.1 – Diferença entre clusterização e coclusterização (adaptado de Madeira [21]). 23
Figura 2.2 – Tipos de cocluster: (a) com valor constante, (b) com valores constantes nas colunas, (c) com valores constantes nas linhas, (d) com coerência aditiva, (e) com coerência multiplicativa, (f) com evolução coerente (adaptado de Madeira [21]). 24
Figura 2.3 – Gráficos dos diferentes tipos de coclusters exemplificados na Figura 2.2. 25
Figura 2.4 – Possíveis estruturas dos coclusters em uma base de dados. (a) cocluster único; (b) coclusters exclusivos em linha e coluna; (c) estrutura de tabuleiro; (d) coclusters exclusivos em linha; (e) coclusters exclusivos em coluna; (f) coclusters não sobrepostos em árvore; (g) coclusters não-sobrepostos e não-exclusivos; (h) coclusters com estrutura hierárquica, (i) coclusters possivelmente sobrepostos e posicionados arbitrariamente (adaptado de Madeira [21]). 27
Figura 2.5 – Exemplos de cocluster perturbados: (a) com valor constante, (b) com valores constantes nas linhas, (c) com valores constantes nas colunas, (d) com coerência aditiva, (e) com coerência multiplicativa (reproduzido de Veroneze [38]). 31
Figura 2.6 – Gráficos dos diferentes tipos de coclusters exemplificados na Figura 2.5. 31
Figura 3.1 – Definição e exemplo de conceito como unidade de conhecimento. 34
Figura 3.2 – Relação de conceitos: exemplo de super-conceito e sub-conceito. 35
Figura 3.3 – Diagrama *line* dos conceitos da Tabela 3.2 (adaptado de Kropat [18]). . 36
Figura 3.4 – Um diagrama *line* (esquerda) com sua correspondente árvore de conceitos Andrews & Hirsch [4]. 36
Figura 3.5 – Ávore correspondente ao diagrama *line* da Figura 3.3. 37
Figura 4.1 – Exemplo de descendentes gerados pelo (a) In-Close2, (b) RIn-Close_CVC_P e (c) RIn-Close_CVC (reproduzido de Veroneze[38]) 44
Figura 4.2 – Todos os coclusters com valores constantes nas colunas minerados a partir da Tabela 4.8 (utilizando $minRow = 2$ e $\epsilon = 1$) (reproduzido de Veroneze [38]) 46
Figura 4.3 – Diagrama que indica os diferentes passos para se obter um cocluster com valores coerentes a partir de coclusters com valores constantes nas colunas (reproduzido de Veroneze [38]) 47

Figura 6.1 – Um cocluster com coerência aditiva visualizado por meio de coordenadas paralelas 52
Figura 6.2 – Exemplo de execução da interface BiVisu [7] após a mineração de coclusters de uma matrz artificial com coclusters com valores coerentes aditivos. 53
Figura 6.3 – (**A**) Três coclusters e suas representações. c_1 e g_1 aparecem nos coclusters B_1 e B_2, então a aresta entre eles é menor e eles aparecem na área de intersecção na visualização final. Áreas de intersecção são mais opacas para realçar a sobreposição. (**B**) Método de visualização principal do BicOverlapper, onde as áreas $(1, 2, 3)$ indicam o agrupamento de coclusters principalmente por conta das condições e as áreas $(4, 5, 6)$ por conta dos genes (reproduzido de Santamaria *et al.* [31]). 55
Figura 6.4 – Exemplo de execução da interface BicOverlapper2 (reproduzido de Santamaria *et al.* [33]). 55
Figura 6.5 – Exemplo de execução da interface FCA Tree Tool, indicando a expansão do segundo conceito da base de dados *Live-In Water* (reproduzido de Andrews & Hirsch [4]). 56
Figura 6.6 – Exemplo de execução da interface SiBIC (reproduzido de Takahashi *et al.* [14]). 57
Figura 7.1 – Indicação dos botões para zoom e barra de menus principal. 61
Figura 7.2 – Menu para importar uma base de dados, o rótulo dos seus atributos, o rótudo dos seus objetos, o rótulo das suas classes e os coclusters minerados desta base; importar os resultados criados por essa interface; e salvar os resultados. 61
Figura 7.3 – Menu para gerar as métricas de validação intra-cocluster e as métricas de regras de associação de classes. 62
Figura 7.4 – Menu para atualizar os coclusters após a seleção das métricas no painel Bic Info; e criação dos gráficos para visualização dos coclusters. 62
Figura 7.5 – Painel inicial da interface que mostra as métricas de validação intra-coclusters. 64
Figura 7.6 – Painel que indica as métricas para a regra de associação de classes. . . 65
Figura 7.7 – Painel para visualização dos métodos diagrama *line*, árvore de conceitos e coordenadas paralelas. 67
Figura 7.8 – Painel para visualização e para percorrer a árvore de conceitos como mostrado por Andrews & Hirsch [4]. 69
Figura 7.9 – Painel para visualização da rede de um conjunto de objetos, inspirada em Takahashi *et al.* [14] e Santamaria *et al.* [31]. 71

Figura 8.1 – Diagrama *line* criado a partir da Tabela 8.2 usando a interface desenvolvida ao longo da pesquisa. 73
Figura 8.2 – Árvore de conceitos criada a partir da Tabela 8.2 usando a interface desenvolvida ao longo da pesquisa 73
Figura 8.3 – Diagrama *line* criado a partir da Tabela 8.4, usando o software Conexp [28]. 76
Figura 8.4 – Árvore de conceitos criada a partir da Tabela 8.4, usando o software FCA Tree Tool [4]. 76
Figura 8.5 – Diagrama *line* criado a partir da Tabela 8.4 pela interface desenvolvida ao longo da pesquisa. 77
Figura 8.6 – Árvore de conceitos totalmente expandida criada a partir da Tabela 8.4 pela interface desenvolvida ao longo da pesquisa 78
Figura 8.7 – Rede de objetos obtida a partir das Tabelas 8.3 e 8.4. 79
Figura 8.8 – Parte do dogma central da Biologia (transcrição), indicando as áreas de genômica e transcriptômica (adaptado de McGrath [23]) 80
Figura 8.9 – Diagrama simplificado indicando as principais etapas do RNA Seq (adaptado de Jabob [15]) 80
Figura 8.10–Rede de objetos formada pelos 150 coclusters filtrados utilizando as métricas: confiança, E_{AColV}, E_{MSR}, E_{SMSR}, E_{ACV}, E_{ASR} e $\overline{\Gamma}$. Os nós em azul escuro indicam as amostras não-SRBCTs, em azul claro os tumores da família EWing, em marrom rhabdomyosarcomas, em laranja neuroblastomas e em verde não-Hodgkin linfomas. 83
Figura 8.11–Análise de componentes principais da base de dados de expressão gênica de gramíneas da espécie *Brachiaria*. 87
Figura 8.12–Rede de objetos formada pelos 21 coclusters filtrados utilizando as métricas: confiança e número de linhas. Os nós em azul escuro indicam os experimentos da classe "Susceptível", em marrom a classe "Resistente" e em verde a classe "Controle". 88
Figura 8.13–*Heatmaps* dos coclusters 1, 2 e 3 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário. 90
Figura 8.14–*Heatmaps* dos coclusters 4, 5 e 6 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário. 91

Figura 8.15–*Heatmaps* dos coclusters 7, 8 e 9 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário. 92
Figura 8.16–*Heatmaps* dos coclusters 10, 11 e 12 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário. 93
Figura 8.17–*Heatmaps* dos coclusters 13, 14 e 15 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário. 94
Figura 8.18–*Heatmaps* dos coclusters 16, 17 e 18 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário. 95
Figura 8.19–*Heatmaps* dos coclusters 19, 20 e 21 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário. 96
Figura 8.20–*Heatmaps* de todos os experimentos da Tabela 8.7 e todos os transcritos indicados na Tabela 8.8 com os valores originais de FPKM. 97
Figura 8.21–*Heatmaps* de todos os experimentos da Tabela 8.7 e todos os transcritos indicados na Tabela 8.8. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário. 98

Lista de tabelas

Tabela 2.1 – Resultado das métricas de validação intra-cocluster aplicadas aos coclusters das Figuras 2.2 e 2.5. 32
Tabela 3.1 – Exemplo de contexto formal - Características de planetas do Sistema Solar [18]. 33
Tabela 3.2 – Conceitos extraídos da Tabela 3.1 [18]. 35
Tabela 4.1 – Variáveis do pseudo-código 4.1 38
Tabela 4.2 – Fechamento do *supremum* (A_1, B_1) da Tabela 3.1 40
Tabela 4.3 – Fechamento do conceito (A_2, B_2) da Tabela 3.1 40
Tabela 4.4 – Conceitos formais da matriz binária da Tabela 4.5 (utilizando $minRow = 2$ e $minCol = 1$). 41
Tabela 4.5 – Características de planetas do Sistema Solar [18] - indicação dos coclusters da Tabela 4.4 41
Tabela 4.6 – Propriedades dos algoritmos da Família RIn-Close [38] 42
Tabela 4.7 – Exemplo de uma matriz de dados numéricos (reproduzido de Veroneze [38]) 46
Tabela 4.8 – Matriz aumentada da matriz descrita na Tabela 4.7 (reproduzido de Veroneze [38]) 46
Tabela 5.1 – Base de dados de transações em um supermercado. 48
Tabela 7.1 – Formato dos dados em cada arquivo que será inserido na interface. . . 61
Tabela 8.1 – Características de planetas do Sistema Solar [18]. 72
Tabela 8.2 – Conceitos extraídos da Tabela 8.1 usando o algoritmo InClose2 [3] com $minRow = 1$ e $minCol = 1$. 73
Tabela 8.3 – Base de dados *Live in Water* [4, 28]. 74
Tabela 8.4 – Coclusters extraídos usando o algortimo InClose2 [3] com $minRow = 1$ e $minCol = 1$ da base de dados *Live in Water*. 75
Tabela 8.5 – Biomarcadores propostos por Khan *et al.* [17] e Pal *et al.* [26] para identificar câncer do tipo SRBCT [38]. Os valores em negrito indicam os biomarcadores em comum às duas propostas 82
Tabela 8.6 – Biomarcadores únicos para os tipos de câncer família Ewing, não-Hodgkin Linfomas, neuroblastomas e rhabdomyosarcomas encontrados a partir da seleção de 150 coclusters utilizando as métricas confiança, E_{AColV}, E_{MSR}, E_{SMSR}, E_{ACV}, E_{ASR} e $\overline{\Gamma}$. Em negrito, estão destacados os genes em comum com a proposta de Khan *et al.* [17]. 84

Tabela 8.7 – Informações sobre cada experimento realizado pelo LGE para analisar duas espécies de gramíneas do gênero *Brachiaria* na presença ou não de cigarrinhas-das-pastagens. 86
Tabela 8.8 – Coclusters extraídos usando o algortimo RIn-Close_CVC com $minRow = 2$ e $minCol = 1$ e as métricas $confiança = 1$ e $númerodelinhas > 4$ . . 87

Sumário

1 Introdução 19
1.1 Motivação 20
1.2 Objetivos 21
1.3 Organização da Dissertação 22
2 Coclusterização 23
2.1 Métricas de Validação 27
2.2 Visão geral do capítulo 32
3 Análise de Conceitos Formais 33
3.1 Visão geral do capítulo 37
4 Algoritmos de Coclusterização 38
4.1 Algoritmo In-Close2 38
4.2 Família RIn-Close 40
4.2.1 Coclusters com valores constantes nas colunas: RIn-Close_CVC . . 42
4.2.2 Coclusters com valores coerentes: RIn-Close_CHV 43
4.3 Visão geral do capítulo 47
5 Regras de Associação 48
5.1 Métricas de Validação 49
5.2 Visão geral do capítulo 51
6 Interfaces de Coclusterização 52
6.1 BiVisu 53
6.2 BicOverlapper 54
6.3 BicOverlapper2 55
6.4 FCA Tree Tool 56
6.5 SiBIC 57
6.6 Visão geral do capítulo 58
7 Interface proposta para visualização, seleção e análise de coclusters 59
7.1 Aspectos técnicos de projeto da interface 59
7.2 Informações relevantes ao usuário da interface proposta 60
7.2.1 Painel Bic Info 62
7.2.2 Painel Bic View 66
7.2.3 Painel Bic Tree 68
7.2.4 Painel Network 70
7.3 Visão geral do capítulo 70
8 Aplicação da interface proposta 72

8.1 Planetas do Sistema Solar 72
8.2 Live In Water 74
8.3 RNA-Seq 79
8.4 Expressão gênica associada a tumores 81
8.5 Expressão gênica de gramíneas do gênero *Brachiaria* 85
9 Conclusão 99
9.1 Perspectivas Futuras 100

Referências 102

1 Introdução

"It is not enough to have a good mind. The main thing is to use it well."
(René Descartes)

A coclusterização tem se mostrado uma técnica poderosa para análise de dados, já incluindo uma ampla gama de cenários de aplicação [11]. Os coclusters são modelos locais fundamentados na clusterização simultânea das múltiplas dimensões que compõem os dados em análise. A coclusterização também pode ser chamada de clusterização simultânea ou biclusterização, no caso de dados bidimensionais, (termo utilizado primeiramente por Cheng e Church [8] na análise de dados de expressão gênica).

Considerando o caso de uma matriz de dados bidimensional, onde as linhas representam objetos e as colunas representam atributos, os coclusters permitem que um mesmo objeto participe de mais de um grupo, motivado principalmente por um elenco distinto de atributos em cada grupo. Além disso, objetos que compõem o mesmo grupo, por causa de um certo subconjunto de atributos, podem compor grupos distintos para outros subconjuntos de atributos. Essas condições não são admissíveis com a clusterização convencional [22].

Para o caso de bases de dados binárias, já existem várias soluções capazes de enumerar de forma eficiente todos os coclusters maximais cujos elementos são todos iguais a 1. Em análise de conceitos formais (FCA, do inglês *formal concept analysis*), mineração de padrões frequentes (FPM, do inglês *frequent patter mining*) e análise de grafos, existem diversos algoritmos capazes de enumerar todos os coclusters maximais com valores constantes (no caso, coclusters maximais de elementos iguais a 1), partindo de uma matriz binária. Esses coclusters são chamados de conceitos formais em FCA, *closed frequent itemsets* [1] em FPM e bicliques maximais em teoria de grafos. Os algoritmos dessas áreas são caracterizados por terem quatro propriedades fundamentais: eficiência, completude, exatidão e não-redundância. Contudo, segundo Veroneze [38], até o desenvolvimento da família de algoritmos RIn-Close em sua dissertação de doutorado, defendida em junho de 2016, não existiam na literatura algoritmos capazes de enumerar coclusters com valores constantes nas linhas ou valores constantes nas colunas, ou então valores coerentes, em matrizes de dados numéricas (não apenas binárias), preservando as quatro propriedades mencionadas.

[1] sendo mais específico, os closed itemsets correspondem ao conjunto de atributos de um cocluster/conceito formal.

Veroneze [38] realizou uma análise minuciosa dos algoritmos concorrentes e mostrou que todos eles carecem de ao menos uma das quatro propriedades citadas, propondo em seguida a família RIn-Close que é capaz de preservar essas propriedades quanto à mineração de coclusters com valores constantes nas linhas e nas colunas (com ou sem ruído) e coclusters com valores coerentes perfeitos (sem ruído). No caso de coclusters com valores coerentes com ruído, três das quatro propriedades são preservadas: completude, exatidão e não-redundância. Neste tipo de cenário, o algoritmo não possui tempo polinomial por cocluster, havendo assim um comprometimento no custo computacional do mesmo.

Uma abordagem alternativa existente na literatura é utilizar algoritmos tradicionais de FCA e FPM por meio da discretização da matriz de dados numérica, seguida da itemização da matriz discretizada [13, 37]. No entanto, este processo envolve perda de informação, o que sustenta ainda mais a família RIn-Close.

1.1 Motivação

Métodos enumerativos de coclusterização tendem a gerar um número elevado de coclusters dependendo do tamanho da base de dados e do número mínimo de linhas e colunas que cada cocluster terá. Portanto, analisar esses resultados se torna uma tarefa árdua e dificulta um uso mais frequente desta técnica em análise de dados [34], principalmente por grupos de pesquisa em áreas não tradicionalmente ligadas a clusterização e aprendizado não-supervisionado.. Algumas das dificuldades na análise de coclusters são:

- Não existe uma métrica de validação que possa ser considerada de consenso para comparar coclusters.
- A interpretação biológica ou científica de alguns coclusters encontrados não é clara.
- Existem poucas técnicas específicas de visualização de coclusters.

Portanto, é nosso objetivo desenvolver uma interface gráfica para auxilar na análise de coclusters obtidos, em especial, pela família RIn-Close, mas também por qualquer outro algoritmo de coclusterização. Assim podemos conciliar melhor teoria e prática, permitindo que essa técnica de análise de dados seja mais utilizada em diversas áreas da ciência.

1.2 Objetivos

Este projeto de pesquisa realizou as seguintes extensões junto à abordagem de Veroneze e colaboradores [38, 39]:

1. Desenvolvimento de uma interface gráfica e de apoio ao usuário, que visa facilitar, agilizar e tornar mais amigável a análise de dados com coclusterização em matrizes numéricas, usando como base a família de algoritmos RIn-Close. Nosso objetivo é criar uma interface gráfica onde se possa recorrer a diferentes métodos de visualização de coclusters (coordenadas paralelas, árvore e diagrama de conceitos formais e grafos), assim como outras informações pertinentes à execução do algoritmo e ao conjunto de coclusters criados pelo RIn-Close (métricas de validação e regras de associação).

2. Construção de regras de associação de classes a partir dos coclusters enumerados, para extrair mais informações relevantes sobre a base de dados estudada. Clusterização e coclusterização são estratégias muito usadas para estudar microarrays. Porém, determinar as interações que existem entre diferentes genes não é algo que possamos inferir facilmente a partir apenas das soluções dessas técnicas. Então, podemos determinar essas relações a partir de regras de associação de classes. Alguns estudos já foram propostos na literatura para utilizar regras de associação juntamente com coclusters em bases de dados binárias [24] e bases de dados reais, porém discretizadas [13]. Veroneze [38] utilizou o conceito de regras de associação de classes (CAR, do inglês *class association rules*) para definir regras de associação para uma base de dados numérica (CAR será definido na Seção 2.4). Para utilizar essas regras, é necessário que a base de dados tenha objetos rotulados em diferentes classes. Em sua dissertação de doutorado, Veroneze [38] utilizou como métricas: suporte, confiança e completude para selecionar os coclusters. Este projeto visou, portanto, ampliar o número de métricas utilizadas, permitindo ao usuário encontrar regras de associação promissoras.

3. Uso de métricas de validação intra-coclusters para medir a qualidade dos coclusters extraídos pelo RIn-Close. Obter uma métrica para avaliar a qualidade de um cocluster é um aspecto importante, não só para guiar uma pesquisa (selecionando coclusters importantes para uma determinada base de dados), como também para estabelecer um critério de comparação entre os resultados obtidos por diferentes técnicas de coclusterização. A interface proposta irá permitir ao usuário selecionar diferentes pontos de corte para diferentes métricas, assim como combiná-las para filtrar e selecionar coclusters.

1.3 Organização da Dissertação

O conteúdo desta dissertação está organizado da seguinte maneira:

Capítulo 2 introduz definições e formulações matemáticas sobre coclusterização. Ele faz uma revisão sobre os principais tipos de cocluster e métodos de validação intra-cocluster.

Capítulo 3 revisa os principais conceitos de Análise de Conceitos Formais, assim como seus principais métodos de visualização.

Capítulo 4 revisa dois algoritmos de coclusterização: o InClose2 e a família RIn-Close, que serão utilizados para minerar coclusters nos casos de estudo desta pesquisa.

Capítulo 5 apresenta a definição de regras de associação e regras de associação de classes, assim como as métricas de validação que serão utilizadas.

Capítulo 6 apresenta as principais interfaces na literatura para análise e visualização de coclusters.

Capítulo 7 apresenta a principal contribuição desta dissertação, mais especificamente, a interface criada, juntamente com seus métodos de visualização e validação de coclusters.

Capítulo 8 apresenta e discute os resultados experimentais de quatro casos de estudo: dois utilizando bases de dados binárias e dois utilizando bases de dados reais de expressão gênica.

Capítulo 9 apresenta as principais conclusões desta pesquisa, assim como aponta sugestões para futuras contribuições.

2 Coclusterização

"The Milky Way is nothing else but a mass of innumerable stars planted together in clusters."
(Galileo Galilei)

A clusterização é uma técnica de aprendizagem não-supervisionada para descobrir padrões em bases de dados. Ela consiste em agrupar dados em clusters, onde as informações internas a cada cluster exibem similaridades ou coerências, ao mesmo tempo em que diferem de informações encontradas em outros clusters. Essas similaridades são medidas utilizando técnicas como distância euclidiana ou distância de Manhattan.

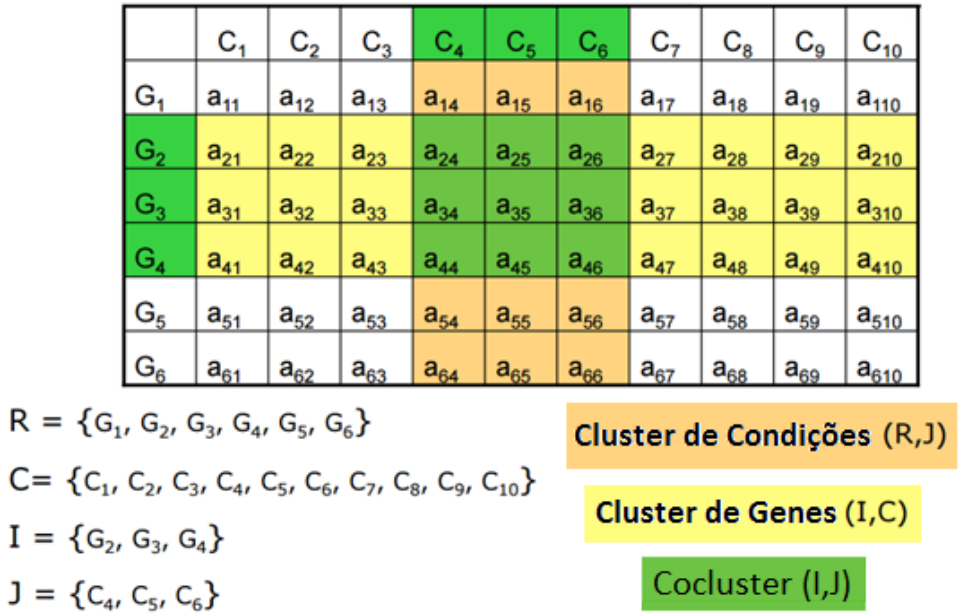

Figura 2.1 – Diferença entre clusterização e coclusterização (adaptado de Madeira [21]).

A coclusterização, por sua vez, é uma técnica local de clusterização que procura por subconjuntos de elementos junto a cada dimensão dos dados, de modo a produzir uma elevada coerência interna do cocluster [13]. Por exemplo, no caso de dados bidimensionais, coclusters são sub-matrizes compostas por subconjuntos de linhas e colunas cujos elementos são coerentes entre si. Para exemplificar a diferença, vide a Figura 2.1. Vamos supor uma matriz $A \in \mathbb{R}^{n \times m}$, cujas linhas representam $n = 6$ genes (objetos) e as colunas representam $m = 10$ condições (atributos). Os valores a_{ij} representam o valor da condição j sobre o gene i. Digamos que os genes G_2, G_3 e G_4 possuam grande correlação quando analisadas as condições C_4, C_5 e C_6. A clusterização poderá gerar dois clusters: (1) Cluster de condições (indicado pelas colunas completas C_4, C_5 e C_6); (2) Cluster de

genes (indicado pelas linhas completas G_2, G_3 e G_4). No mesmo exemplo, o processo de coclusterização agrupará somente a sub-matriz indicada em verde (contém o conjunto formado pela intersecção das linhas G_2, G_3 e G_4 e das colunas C_4, C_5 e C_6). Além disso, as condições e os genes não precisam ser contíguos na matriz, embora assim tenha sido considerado na Figura 2.1. Logo, ocorrem duas situações novas: (1) fica explícito o subconjunto de condições (atributos) responsáveis pela composição desse grupo de genes; e (2) cada gene pode ser alocado a múltiplos agrupamentos, cada um associado a um subconjunto específico de condições. Sendo assim, um gene G_i poderia estar no mesmo grupo do gene G_k em relação a um subconjunto de condições, mas não em relação a outro subconjunto de condições. Portanto, este tipo de situação só pode ser apropriadamente modelado com o emprego da coclusterização.

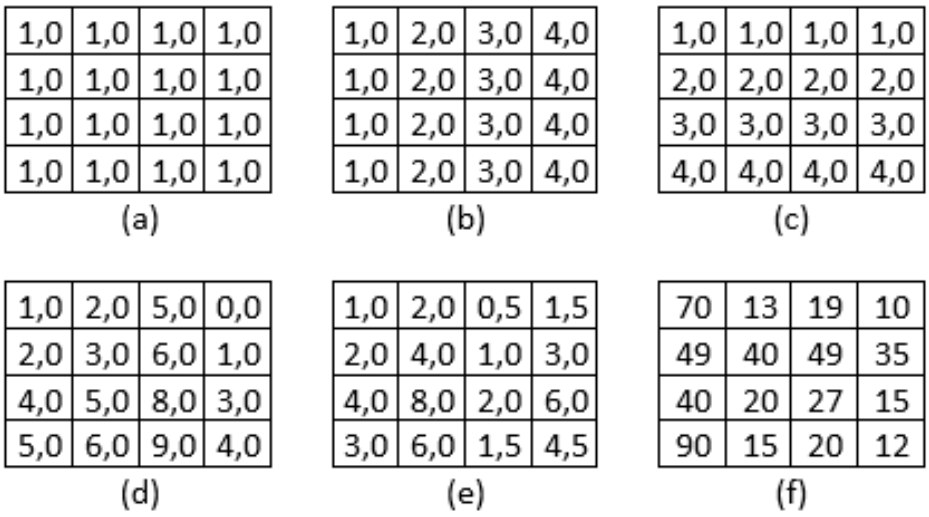

Figura 2.2 – Tipos de cocluster: (a) com valor constante, (b) com valores constantes nas colunas, (c) com valores constantes nas linhas, (d) com coerência aditiva, (e) com coerência multiplicativa, (f) com evolução coerente (adaptado de Madeira [21]).

Para a definição formal de um cocluster, vamos considerar o caso bidimensional e tomar uma matriz $A \in \mathbb{R}^{n \times m}$, que pode ser definida por seu conjunto de linhas $X = \{1, 2, \cdots, n\}$ e seu conjunto de colunas $Y = \{1, 2, \cdots, m\}$. Generalizações para mais dimensões são imediatas. Considere $I \subseteq X$e $J \subseteq Y$como sendo subconjuntos das linhas e colunas da matriz A, respectivamente. O cocluster $A_{IJ} = (I, J)$ representa uma submatriz de A, contendo apenas os elementos a_{ij} pertencentes ao subconjunto de linhas I e ao subconjunto de colunas J de A. Esses elementos a_{ij} compõem um cocluster por apresentarem um comportamento coerente. De acordo com Madeira e Oliveira [22], existem seis tipos de comportamento coerente que um cocluster pode apresentar, conforme ilustrado nas Figuras 2.2 e 2.3.

Embora coclusters perfeitos possam existir em algumas bases de dados, eles

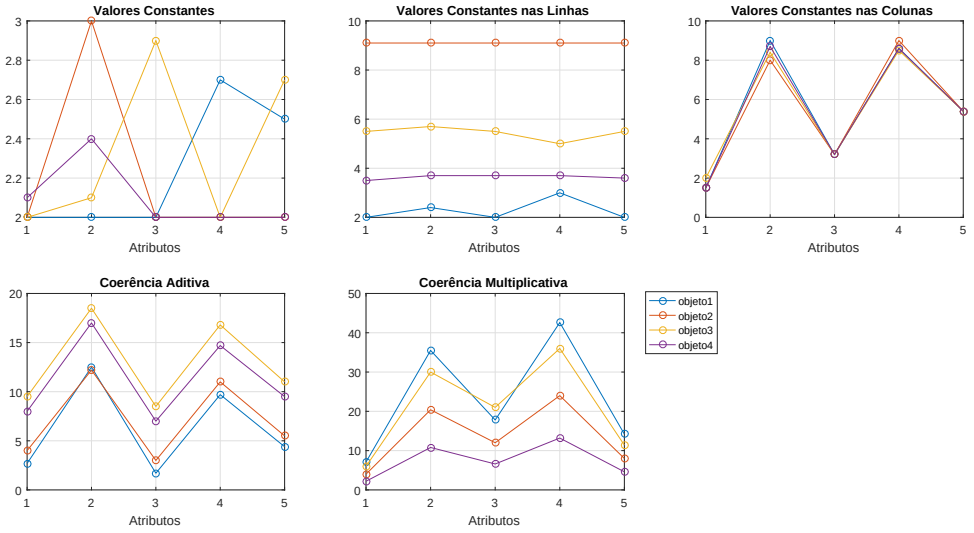

Figura 2.3 – Gráficos dos diferentes tipos de coclusters exemplificados na Figura 2.2.

normalmente estão mascarados por ruídos em bases de dados reais. Portanto, iremos definir todos os seis tipos de coclusters nos casos perfeito e perturbado usando as definições de Veroneze [38]. O parâmetro $\varepsilon \geqslant 0$ definido pelo usuário determina o ruído máximo permitido em um cocluster.

Seja $|\zeta|$ (i) o valor absoluto (ou módulo) de ζ se ζ for um escalar; ou (ii) a cardinalidade (número de elementos) de ζ se ζ for um conjunto.

Um cocluster com valores constantes, Figura 2.2 (a), perfeito é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que $a_{ij} = a_{kl}$, $\forall i, k \in I$ e $\forall j, l \in J$. Um cocluster com valores constantes perturbado é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que $|a_{ij} - a_{kl}| \leqslant \varepsilon$, $\forall i, k \in I$ e $\forall j, l \in J$, ou seja,

$$\max_{i\in I} \max_{j\in J}(a_{ij}) - \min_{i\in I} \min_{j\in J}(a_{ij}) \leqslant \varepsilon. \tag{2.1}$$

Um cocluster com valores constantes nas linhas, Figura 2.2 (c), é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que $a_{ij} = a_{il}$, $\forall i \in I$ e $\forall j, l \in J$. Um cocluster com valores perturbados nas linhas é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que $|a_{ij} - a_{il}| \leqslant \varepsilon$, $\forall i \in I$ e $\forall j, l \in J$, ou seja,

$$\max_{j\in J}(a_{ij}) - \min_{j\in J}(a_{ij}) \leqslant \varepsilon, \forall i \in I. \tag{2.2}$$

O mesmo raciocínio se aplica ao cocluster com valores constantes nas colunas, Figura 2.2 (b). Logo, um cocluster com valores contantes nas colunas perfeito é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que $a_{ij} = a_{kj}$, $\forall i, k \in I$ e $\forall j \in J$. Um cocluster

com valores perturbados nas colunas é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que $|a_{ij} - a_{kj}| \leqslant \varepsilon$, $\forall i, k \in I$ e $\forall j \in J$, ou seja,

$$\max_{i\in I}(a_{ij}) - \min_{i\in I}(a_{ij}) \leqslant \varepsilon, \forall j \in J. \tag{2.3}$$

Seja $Z^{jl} = \{a_{ij} - a_{il}\}_{i\in I}$, $j, l \in J$. Então, o cocluster perfeito com valor coerente no modelo aditivo, Figura 2.2 (d), é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que todos os elementos do conjunto Z^{jl},$\forall j, l \in J$ são iguais, ou seja, satisfazem $z = w$, $\forall z, w \in Z^{jl}$, $\forall j, l \in J$. Um cocluster perturbado com valor coerente no modelo aditivo é uma submatriz (I, J) tal que $|z - w| \leqslant \varepsilon$, $\forall z, w \in Z^{jl}$, $\forall j, l \in J$, ou seja,

$$\max(Z^{jl}) - \min(Z^{jl}) \leqslant \varepsilon, \forall j, l \in J. \tag{2.4}$$

Seja $Z^{jl} = \{a_{ij}/a_{il}\}_{i\in I}$, $j, l \in J$. Um cocluster perfeito com valores coerentes no modelo multiplicativo, Figura 2.2 (e), é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que todos os elementos de Z^{jl}, $\forall j, l \in J$ são iguais, ou seja, $z = w$, $\forall z, w \in Z^{jl}$, $\forall j, l \in J$. Um cocluster perturbado com valores coerentes no modelo multiplicativo é uma sub-matriz (I, J) tal que

$$\left|\frac{\max(Z^{jl})}{\min(Z^{jl})} - 1\right| \leqslant \varepsilon, \forall j, l \in J. \tag{2.5}$$

Os coclusters com evolução coerente são definidos através do comportamento dos valores dentro do cocluster, e não dependem diretamente dos valores em si. Há vários subtipos de coclusters com evolução coerente, sendo o mais comum o OPSM (do inglês, *order-preserving submatrix*). Um cocluster OPSM é uma sub-matriz (I, J) da matriz $A_{n\times m}$ tal que existe uma permutação $P = \{p_1, p_2, \cdots, p_s\}$ de um conjunto de colunas J onde $a_{ip_1} \leqslant a_{ip_2} \leqslant \cdots \leqslant a_{ip_s}$ $\forall i \in I$. A Figura 2.2 (f) é um exemplo de um cocluster OPSM, no qual temos: $a_{i4} \leqslant a_{i2} \leqslant a_{i3} \leqslant a_{i1}$, para $i = 1, 2, 3, 4$.

Além de ser possível definir diversos tipos de coclusters, os vários coclusters podem assumir estruturas diferentes numa mesma base de dados, sendo que cada uma dessas estruturas apresenta um desafio diferente em sua obtenção. A Figura 2.4 exibe as estruturas exploradas até hoje na literatura [22].

A família de algoritmos RIn-Close opera com a versão mais genérica e mais desafiadora de coclusters, que são representados na Figura 2.4 (i). Embora a Figura 2.4 (i) assim ilustre, não há a necessidade de que os coclusters sejam contíguos, ou seja, com linhas e colunas em sequência.

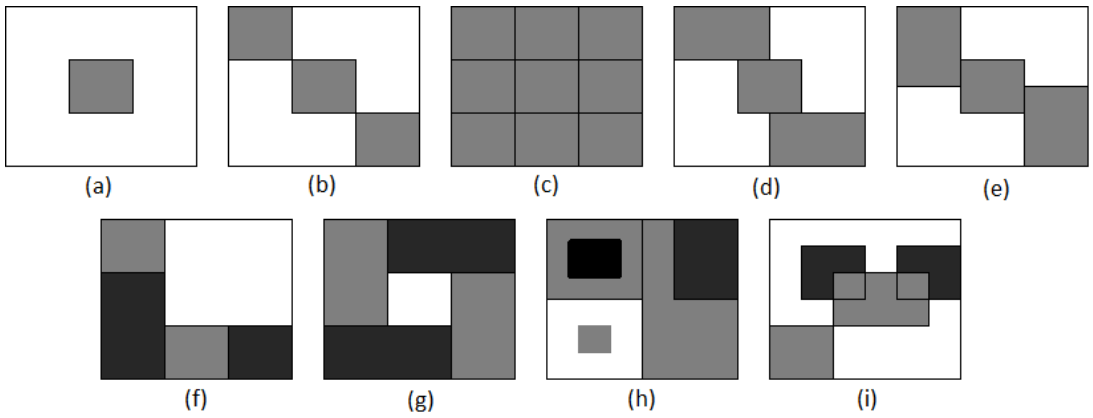

Figura 2.4 – Possíveis estruturas dos coclusters em uma base de dados. (a) cocluster único; (b) coclusters exclusivos em linha e coluna; (c) estrutura de tabuleiro; (d) coclusters exclusivos em linha; (e) coclusters exclusivos em coluna; (f) coclusters não sobrepostos em árvore; (g) coclusters não-sobrepostos e não-exclusivos; (h) coclusters com estrutura hierárquica, (i) coclusters possivelmente sobrepostos e posicionados arbitrariamente (adaptado de Madeira [21]).

2.1 Métricas de Validação

Uma métrica de validação intra-cocluster é um indicador da qualidade de um cocluster, ou seja, ela quantifica o grau de coerência interna em um cocluster [29]. Ao contrário de métricas de validação de clusters que já estão estabelecidas na literatura, existem poucas métricas para coclusters e, além disso, não existe um consenso sobre qual a métrica mais eficaz [20]. Por esses motivos, adotamos 11 métricas para selecionar coclusters, sendo 8 métricas de validação intra-cocluster (*Average Column Variance, Average Row Variance, Mean Squared Residue, Scaling Mean Squared Residue, Average Correlation Value, Average Spearman's Rho, Hubert Statistics* e *p-value*) e 3 métricas para seleção segundo o tamanho dos coclusters (número de linhas, número de colunas e volume). Sejam (I, J) um cocluster, m_{ij} uma célula deste cocluster, m_{iJ} o valor médio sobre a linha i, m_{Ij} o valor médio sobre a coluna j e m_{IJ} o valor médio sobre o cocluster todo, definimos as seguintes métricas:

1. **Average Column Variance**: coclusters que possuem grande variação nas suas colunas são caracterizados por terem uma grande variância média nas colunas. Com essa métrica, então, podemos escolher coclusters que possuem colunas com valores coerentes sobre um determinado conjunto de linhas, o que auxilia na escolha de coclusters do tipo CVC (do inglês, *constant value on columns*).

$$E_{AColV}(I, J) = \frac{\sum_{i \in I} \sum_{j \in J} (m_{ij} - m_{Ij})^2}{|I||J|} \tag{2.6}$$

2. **Average Row Variance** [29]: coclusters que possuem grande variação nas suas linhas são caracterizados por terem uma grande variância média nas linhas. Com essa métrica, então, podemos escolher coclusters que possuem linhas com valores coerentes sobre um determinado conjunto de colunas, o que auxilia na escolha de coclusters do tipo CVR (do inglês, *constant value on rows*).

$$E_{ARV}(I,J) = \frac{\sum\limits_{i\in I}\sum\limits_{j\in J}(m_{ij} - m_{iJ})^2}{|I||J|} \tag{2.7}$$

3. **Mean Squared Residue** [27, 29]: essa métrica representa a variação associada com a interação entre linhas e colunas num cocluster. Portanto, uma resposta baixa (alta), ou seja próxima de 0 (maior que um limiar δ), indica que um cocluster é fortemente (fracamente) coerente. Essa função é interessante para coclusters do tipo CHV (do inglês, *coherent values*) tomando um modelo aditivo, porém não se aplica no caso de modelo multiplicativo.

$$E_{MSR}(I,J) = \frac{\sum\limits_{i\in I}\sum\limits_{j\in J}(m_{ij} - m_{iJ} - m_{Ij} + m_{IJ})^2}{|I||J|} \tag{2.8}$$

4. **Scaling Mean Squared Residue** [27]: essa métrica permite encontrar padrões de escalonamento, ou seja, coerência multiplicativa (cocluster do tipo CHV tomando um modelo multiplicativo), sendo uma alternativa, portanto, à metrica anterior.

$$E_{SMSR}(I,J) = \frac{1}{|I||J|}\sum_{i\in I}\sum_{j\in J}\frac{(m_{iJ}\times m_{Ij} - m_{ij}\times m_{IJ})^2}{m_{iJ}^2 \times m_{Ij}^2} \tag{2.9}$$

5. **Average Correlation Value** [27, 29]: essa métrica usa o coeficiente de correlação de Pearson associado a cada linha do cocluster para avaliar a homogeneidade deste. Os valores desta métrica pertencem ao intervalo $[0; +1]$. Então, valores próximos de 1 indicam que o cocluster é altamente coerente. Contudo, este valor pode decrescer se existir ruído na base de dados.

$$E_{ACV}(I,J) = \max\left\{\frac{\sum\limits_{i_1=1}^{|I|}\sum\limits_{i_2=1}^{|I|}|r_{i_1i_2}| - |I|}{|I|^2 - |I|}, \frac{\sum\limits_{j_1=1}^{|J|}\sum\limits_{j_2=1}^{|J|}|r_{j_1j_2}| - |J|}{|J|^2 - |J|}\right\} \tag{2.10}$$

onde $r_{i_1i_2}$ e $r_{j_1j_2}$ se referem à correlação entre pares de linhas i_1i_2 e colunas j_1j_2, respectivamente, de acordo com o coeficiente de Pearson.

6. **Average Spearman's Rho** [29, 27]: essa métrica utiliza o rank de correlação de Spearman associado às linhas do cocluster. Os valores pertencem ao intervalo:

$[-1; +1]$. Então, um valor próximo de 1 indica que o cocluster é altamente coerente. Esta métrica também é sensível à presença de ruído na base de dados.

$$E_{ASR}(I, J) = 2 \max \left\{ \frac{\sum_{i \in I} \sum_{j \in I, j \geqslant i+1} \rho_{ij}}{|I|(|I| - 1)}, \frac{\sum_{k \in J} \sum_{l \in J, , l \geqslant k+1} \rho_{kl}}{|J|(|J| - 1)} \right\} \tag{2.11}$$

onde ρ_{ij} e ρ_{kl} se referem à correlação de Spearman entre duas linhas e duas colunas, respectivamente.

7. **Hubert Statistics** [30]: A estatística de Hubert é uma métrica conhecida para validação de clusters. Santamaria [34] adaptou essa métrica para validar coclusters. Na clusterização, ela mede a correlação entre as entradas de duas matrizes $n \times n$, $\mathcal{X}$ e $\mathcal{Y}$, sobre os n objetos. $\mathcal{X}_{ij}$ denota a proximidade observada entre os objetos i e j, dada uma métrica de distância. Na clusterização, $\mathcal{Y}_{ij}$ é normalmente 0 se os objetos i e j estão na mesma categoria (ou cluster) e 1, caso contrário. A estatística normalizada ($\overline{\Gamma}$) é dada por:

$$\overline{\Gamma}(\mathcal{X}, \mathcal{Y}) = \frac{\frac{1}{k} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (\mathcal{X}_{ij} - \mu_{\mathcal{X}})(\mathcal{Y}_{ij} - \mu_{\mathcal{Y}})}{\sigma_{\mathcal{X}} \sigma_{\mathcal{Y}}} \tag{2.12}$$

onde $k = \dfrac{n(n-1)}{2}$, $\mu_{\mathcal{X}}$ e $\mu_{\mathcal{Y}}$ são as médias das matrizes e $\sigma_{\mathcal{X}}$ e $\sigma_{\mathcal{Y}}$ são as variâncias das matrizes. A adaptação feita por Santamaria [34] leva em conta dois aspectos da coclusterização e de coclusters: a bi-dimensionalidade e a sobreposição. Sobre a bi-dimensionalidade, com o intuito de capturar a correlação entre os objetos e entre os atributos, dois pares de matrizes são definidos, (P^O,C^O) para os objetos (linhas) e (P^F,C^F) para os atributos (colunas), compondo assim dois índices ($\overline{\Gamma}_O$,$\overline{\Gamma}_F$). Assim, para compor a estatística de Hubert ($\overline{\Gamma}$), ambos os índices são combinados por meio de uma média ponderada:

$$\overline{\Gamma} = \frac{n\overline{\Gamma}_O + m\overline{\Gamma}_F}{n + m} \tag{2.13}$$

onde n é o número de objetos e m é o número de atributos. As matrizes de proximidade, P^O e P^F, são criadas usando a distância euclidiana a partir da matriz de dados A, assim como na clusterização:

$$P^O = (p^O_{ij}) = \frac{1}{n} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \sqrt{\sum_{k=1}^{m} (a_{ik} - a_{jk})^2} \tag{2.14}$$

$$P^F = (p^F_{ij}) = \frac{1}{m} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sqrt{\sum_{k=1}^{n} (a_{ki} - a_{kj})^2} \tag{2.15}$$

Para resolver o problema de sobreposição, Santamaria [34] substituiu a matriz de clusterização $\mathcal{Y}$ pelas matrizes de coclusterização C^O e C^F:

$$C = (c_{ij}), \text{onde } c_{ij} = \frac{1}{1 + k_{ij}} \tag{2.16}$$

onde k_{ij} é o número de coclusters em que o elemento i (um objeto em C^O ou um atributo em C^F) é agrupado juntamente com o elemento j. $c_{ij} \in (0, 1]$, sendo 1 se eles nunca aparecem juntos e se encaminhando até 0 se eles aparecem juntos em vários coclusters.

Após a definição das matrizes P e C, podemos definir as estatísticas de Hubert para os objetos $\overline{\Gamma}_O$ e para os atributos $\overline{\Gamma}_F$:

$$\overline{\Gamma}_O = \frac{2}{n(n-1)} \frac{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (p_{ij} - \mu_p)(c_{ij} - \mu_c)}{\sigma_p \sigma_c} \tag{2.17}$$

$$\overline{\Gamma}_F = \frac{2}{m(m-1)} \frac{\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} (p_{ij} - \mu_p)(c_{ij} - \mu_c)}{\sigma_p \sigma_c} \tag{2.18}$$

onde $\mu_p(\mu_c)$ e $\sigma_p(\sigma_c)$ são a média e o desvio padrão de $P(C)$, respectivamente.

8. **p-value** [14]: cômputo empírico de p-values em termos de quão correlacionadas as linhas de um cocluster são. Para um cocluster com N linhas e M colunas, matrizes aleatórias de mesmo tamanho, a partir da matriz original (ou seja, permutando os elementos da matriz original), são geradas. Takahashi *et al.* [14] geravam 100.000 matrizes aleatórias. Essa métrica, então, calcula o seguinte teste estatístico T sobre cada matriz gerada:

$$T = \frac{1}{N} \sum_{i=1}^{N} corr(v_i, \overline{v}), \text{onde } \overline{v} = \frac{1}{N} \sum_{i=1}^{N} v_i \tag{2.19}$$

onde v_i é um vetor linha M-dimensional representando cada objeto, $\overline{v}$ é a média aritmética dos objetos do cocluster sobre todos os atributos e $corr(v_i, \overline{v})$ é o coeficiente de correlação de Pearson entre os vetores v_i e $\overline{v}$. Depois, computa-se a razão de quantas matrizes aleatórias possuíam valores menores que T em relação ao valor de T do cocluster da matriz original, resultando no valor empírico do p-value de um cocluster.

9. **Número de linhas** [38]: essa métrica permite ao usuário selecionar o número mínimo de linhas que um cocluster deve possuir.

10. **Número de colunas** [38]: essa métrica permite ao usuário selecionar o número mínimo de colunas que um cocluster deve possuir.

11. **Volume** [29]: quando consideramos o caso bi-dimensional, essa métrica calcula a área de um cocluster. A denominação de volume é usual na literatura e será mantida aqui, embora se refira à área de um cocluster.

$$E_V(I, J) = |I||J| \tag{2.20}$$

2.0	2.0	2.0	2.7	2.5
2.0	3.0	2.0	2.0	2.0
2.0	2.1	2.9	2.0	2.7
2.1	2.4	2.0	2.0	2.0

(a)

2.0	2.4	2.0	3.0	2.0
9.1	9.1	9.1	9.1	9.1
5.5	5.7	5.5	5.0	5.5
3.5	3.7	3.7	3.7	3.6

(b)

1.5	9.0	3.2	8.6	5.4
1.5	8.0	3.2	9.0	5.4
2.0	8.4	3.2	8.5	5.4
1.5	8.7	3.2	8.6	5.4

(c)

2.7	12.2	1.7	9.7	4.4
4.0	12.5	3.0	11.0	5.5
9.5	18.5	8.5	16.8	11.0
8.0	17.0	7.0	14.7	9.5

(d)

7.1	35.5	17.8	42.6	14.2
4.0	20.4	12.0	24.0	8.0
6.0	30.0	21.0	36.0	11.4
2.2	10.8	6.6	13.2	4.6

(e)

Figura 2.5 – Exemplos de cocluster perturbados: (a) com valor constante, (b) com valores constantes nas linhas, (c) com valores constantes nas colunas, (d) com coerência aditiva, (e) com coerência multiplicativa (reproduzido de Veroneze [38]).

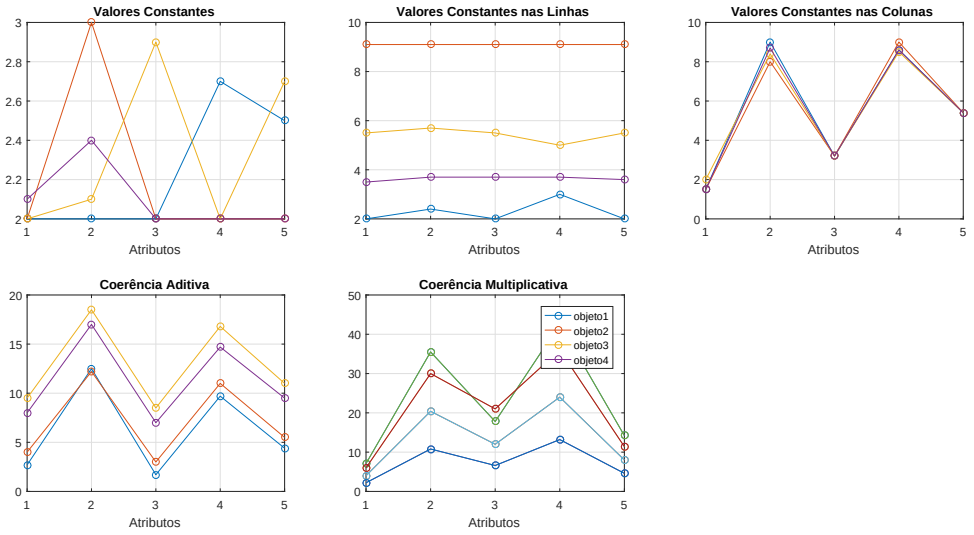

Figura 2.6 – Gráficos dos diferentes tipos de coclusters exemplificados na Figura 2.5.

Para exemplificar a utilização das métricas apresentadas acima, elas foram aplicadas em diferentes tipos de coclusters sem e com ruído, Figuras 2.2 e 2.5, respectivamente, resultando na Tabela 2.1. A Figura 2.6 apresenta gráficos dos diferentes tipos de coclusters exemplificados na Figura 2.5.

A partir da Tabela 2.1, percebemos a importância de se ter diversas métricas de validação para analisar diferentes tipos de coclusters. Por exemplo, a métrica *Mean Squared Residue* para analisar o cocluster da Figura 2.2 (d), onde temos coerência aditiva e a métrica *Scaling MSR* para analisar o cocluster da Figura 2.2 (e), onde temos coerência multiplicativa. Note que o cocluster com coerência aditiva possui $E_{msr} = 0$, enquanto que o cocluster com coerência multiplicativa possui $E_{smsr} = 0$.

Tabela 2.1 – Resultado das métricas de validação intra-cocluster aplicadas aos coclusters das Figuras 2.2 e 2.5.

	E_{AColV}	E_{ARV}	E_{MSR}	E_{SMSR}	E_{ACV}	E_{ASR}
Fig. 2.2 (a)	0,0000	0,0000	0,0000	0,0000	NaN	NaN
Fig. 2.2 (b)	0,0000	1,2500	0,0000	0,0000	1,0000	1,0000
Fig. 2.2 (c)	1,2500	0,0000	0,0000	0,0000	1,0000	1,0000
Fig. 2.2 (d)	2,5000	3,5000	0,0000	0,1021	1,0000	1,0000
Fig. 2.2 (e)	2,3438	2,3438	0,3906	0,0000	1,0000	1,0000
Fig. 2.2 (f)	183,8438	441,7813	136,9375	0,0999	0,8367	0,9743
Fig. 2.5 (a)	0,0985	0,1052	0,0911	0,0171	0,4560	-0,1434
Fig. 2.5 (b)	6,5966	0,0536	0,0470	0,0069	0,9931	1,0000
Fig. 2.5 (c)	0,0441	7,9514	0,0422	0,0039	0,9949	0,9333
Fig. 2.5 (d)	7,7310	15,7808	0,0564	0,0330	0,9974	1,0000
Fig. 2.5 (e)	52,5852	94,0478	13,4370	0,0027	0,9916	1,0000

2.2 Visão geral do capítulo

Este capítulo apresentou as principais definições de coclusterização, assim como os diferentes tipos e estruturas de coclusters. Apresentamos também o conjunto de métricas de validação intra-coclusters que será utilizado para selecionar coclusters na interface proposta nesta dissertação, descrita no capítulo 7.

O próximo capítulo irá apresentar os principais aspectos da área de Análise de Conceitos Formais que será base de dois métodos de visualização utilizados na interface proposta: diagrama *line* e árvores de conceitos.

3 Análise de Conceitos Formais

Este capítulo fornece um pequeno resumo sobre os principais aspectos de Análise de Conceitos Formais, com o intuito de ampliar os conhecimentos sobre os métodos de visualização que serão utilizados na interface proposta nesta dissertação. Este capítulo introduz as definições dos principais termos de Análise de Conceitos Formais, essencialmente, contextos formais, conceitos formais e hierarquia de conceitos, assim como explicações sobre os principais métodos de visualização, diagrama line e árvore de conceitos.

A *análise de conceitos formais* estuda como objetos podem ser agrupados hierarquicamente de acordo com atributos em comum [18].

Definição 3.1 *Um contexto formal é um trio* (G, M, R) *de dois conjuntos* G *e* M*, e uma relação* $R \subseteq G \times M$*. Cada* $g \in G$ *é interpretado como um objeto, e cada* $m \in M$ *como um atributo. Para expressar que um objeto* g *possui uma relação* R *com o atributo* m*, escrevemos* $(g, m) \in R$ *ou* gRm*, ou seja, "o objeto* g *possui o atributo* m*"[10].*

Tabela 3.1 – Exemplo de contexto formal - Características de planetas do Sistema Solar [18].

	Gas giant	Terrestrial	Moon	Habital Zone
Earth		×	×	×
Jupiter	×		×	
Mercury		×		
Mars		×	×	

A Tabela 3.1 mostra um contexto formal em que os objetos G se referem a planetas do Sistema Solar e os atributos M se referem as suas características. A relação binária R é dada pelos símbolos × na Tabela 3.1 e descrevem quais características cada planeta apresenta.

Um conceito formal é análogo à definição de um cocluster de valores constantes, se considerada uma matriz binária. Seja G o conjunto de objetos, M o conjunto de atributos e representando a relação $R \subseteq G \times M$, temos uma matriz binária A, em que $a_{ij} = 1 \Leftrightarrow iAj$. Para um subconjunto de objetos $I \subseteq G$ (ou de atributos $J \subseteq M$), define-se o conjunto $I' \subseteq M$ de atributos comuns entre os objetos de I (ou conjunto $J' \subseteq G$

de objetos comuns entre os atributos de J), produzindo: $I' = \{j \in M | \forall i \in I : iAj\}$ e $J' = \{i \in G | \forall j \in J : iAj\}$.

Definição 3.2 *Um conceito formal de um contexto formal* (G, M, R) *é um par* (I, J) *com* $I \subseteq G$, $J \subseteq M$, $I' = J$ *e* $J' = I$.

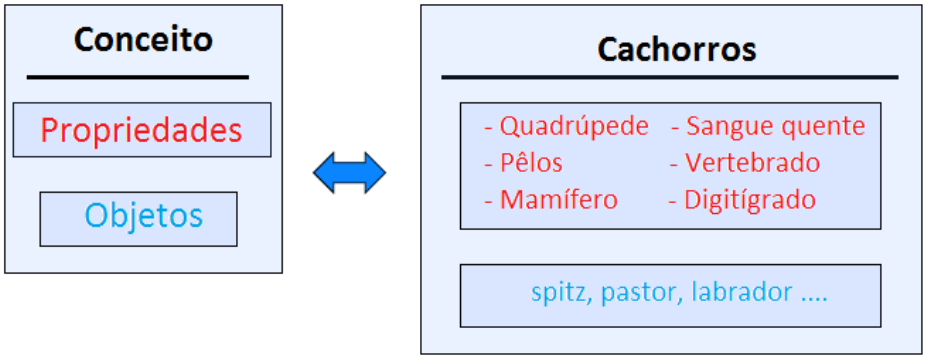

Figura 3.1 – Definição e exemplo de conceito como unidade de conhecimento.

I é chamado de *extent* e J é chamado de *intent* de um conceito formal (I, J). Considerando a matriz A, o conceito formal (I, J) equivale ao cocluster $A_{IJ} = \{i \in I, j \in J | \forall i, j : a_{ij} = 1\}$. Note que, por definição, um conceito formal é um cocluster maximal de 1s. Cada combinação de linhas ou colunas pode gerar um conceito formal, de modo que o número de conceitos formais pode crescer exponencialmente com o tamanho da matriz [10].

Por simplicidade, iremos usar, a partir deste momento, apenas conceito e contexto para nos referir a conceito formal e contexto formal, respectivamente.

Muitos algoritmos de busca dos conceitos se baseiam na ideia de ordem lexicográfica para evitar conceitos repetidos [10]. Na matemática, combinações possuem uma ordem lexicográfica, ou cânon, na qual $\{1, 2, 3\}$ vem antes de $\{1, 2, 4\}$, e $\{1, 2, 6\}$ vem antes de $\{1, 3\}$, por exemplo. Se os conceitos forem gerados nessa ordem (para objetos em G ou atributos em M), é possível saber se um conceito é novo observando sua canonicidade [19].

Definição 3.3 *Se* (I_1, J_1) *e* (I_2, J_2) *são conceitos de um contexto, então* (I_1, J_1) *é dito um* subconceito *de* (I_2, J_2) *se* $I_1 \subseteq I_2$ *(o que é equivalente a* $J_2 \subseteq J_1$*). Neste caso,* (I_2, J_2) *é um* superconceito *de* (I_1, J_1) *e escrevemos* $(I_1, J_1) \leq (I_2, J_2)$*. A relação* $\leq$ *é chamada de ordem hierárquica (ou simplesmente ordem) de conceitos. Por exemplo, gato* $\leq$ *mamífero. O conjunto de todos os conceitos de* (G, M, R)*, ordenados desta maneira, é denotado como* $\beta(G, M, R)$ *e é chamado de* lattice *de conceitos do contexto* (G, M, R) *[10].*

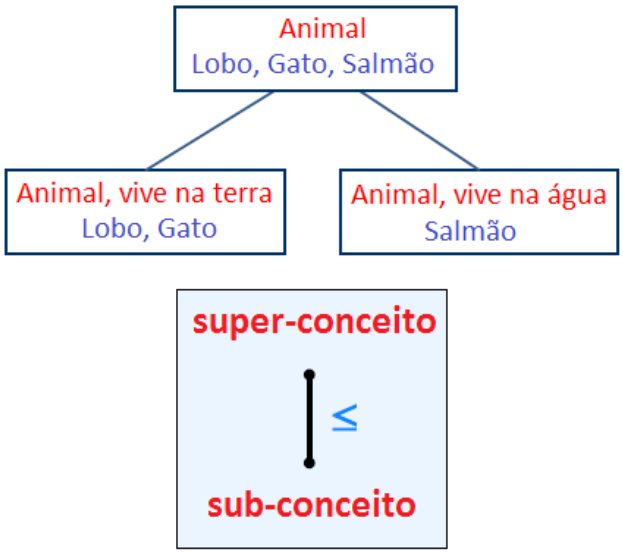

Figura 3.2 – Relação de conceitos: exemplo de super-conceito e sub-conceito.

Tabela 3.2 – Conceitos extraídos da Tabela 3.1 [18].

Item	Extent	Intent
c_1	{Jupiter}	{Gas giant, Moon}
c_2	{Earth, Mercury, Mars}	{Terrestrial}
c_3	{Earth, Jupiter, Mars}	{Moon}
c_4	{Earth}	{Terrestrial, Moon, Habital Zone}
c_5	$\varnothing$	{Gas giant, Terrestrial, Moon, Habital Zone}
c_6	{Earth, Mars}	{Terrestrial, Moon}
c_7	{Earth, Jupiter, Mercury, Mars}	$\varnothing$

A Tabela 3.2 mostra os conceitos do contexto formal da Tabela 3.1. A partir deles, é possível criar a Figura 3.3, que mostra os conceitos em termos de um diagrama *line*. Neste diagrama, cada nó representa um conceito. Um conceito c_1 é um sub-conceito de um conceito c_2 se, e somente se, existe um caminho descendente por uma aresta que liga o nó do conceito c_2 com o nó do conceito c_1 (ver também Figura 3.2).

Este diagrama se inicia no *supremum* e termina no *infimum*. O primeiro nó (parte superior) representa o *supremum*, ou seja, ele possui todas as linhas da matriz. Os outros nós são gerados a partir da inclusão de atributos (colunas) possivelmente diminuindo o conjunto de objeto (linhas) e deixando o nó, portanto, mais específico. O último nó (parte inferior) representa o *infimum*, ou seja, ele é formado por todas as colunas.

Um diagrama *line* pode ser visto como uma coleção de árvores sobrepostas, onde os nós com múltiplos pais têm suas arestas reduzidas para possuir, no fim, um único pai. Em termos das conexões entre conceitos formais em um diagrama *line*, onde uma aresta é removida para formar a árvore correspondente, qualquer atributo ou objeto

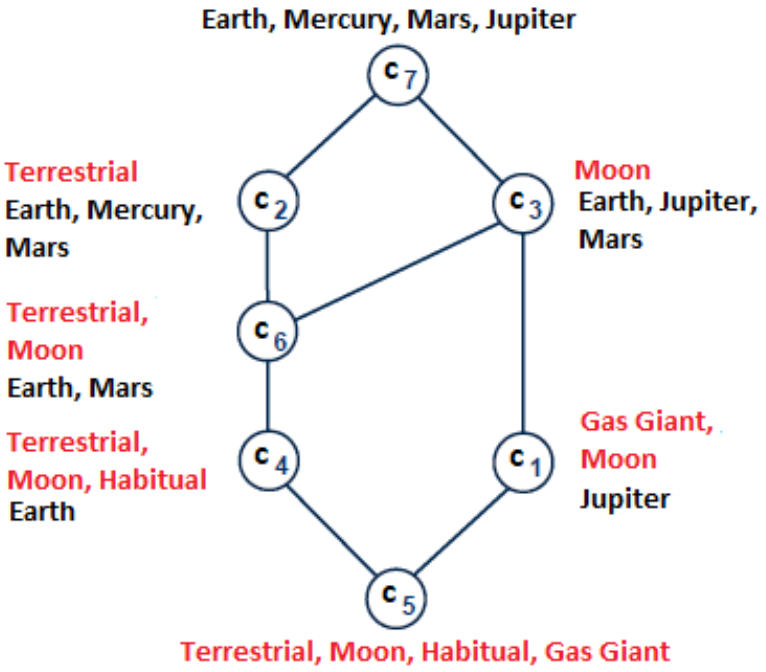

Figura 3.3 – Diagrama *line* dos conceitos da Tabela 3.2 (adaptado de Kropat [18]).

herdado e que é perdido na remoção dessa aresta deve ser restaurado para os conceitos pai e filho cujas arestas foram removidas. Assim, os conceitos em um *lattice* e numa árvore são exatamente os mesmos – apenas a quantidade de conexões foi removida. A Figura 3.4 ilustra esta relação. Nesse caso, os atributos são $a1$, $a2$, $a3$ e $a4$ e os objetos são $o1$, $o2$, $o3$ e $o4$. Note que o atributo $a3$ e o objeto $o4$ estão presentes duas vezes, para assim manter o *intent* e o *extent* de um conceito cujas conexões foram perdidas na criação da árvore [4].

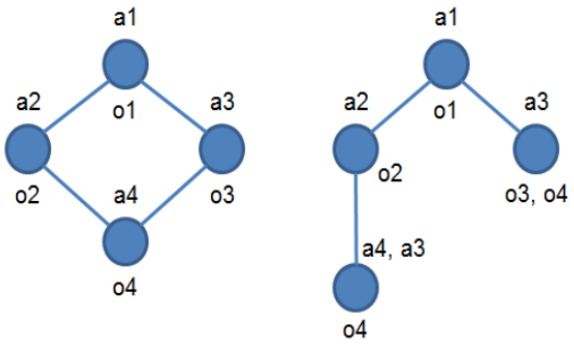

Figura 3.4 – Um diagrama *line* (esquerda) com sua correspondente árvore de conceitos Andrews & Hirsch [4].

Portanto, a partir desta definição de árvore de conceitos, podemos criar uma árvore (Figura 3.5) para o contexto formal da Tabela 3.1. A vantagem de se utilizar árvores de conceitos é sua facilidade de entendimento e são particularmente apropriadas em situações em que a informação admite uma representação hierárquica.

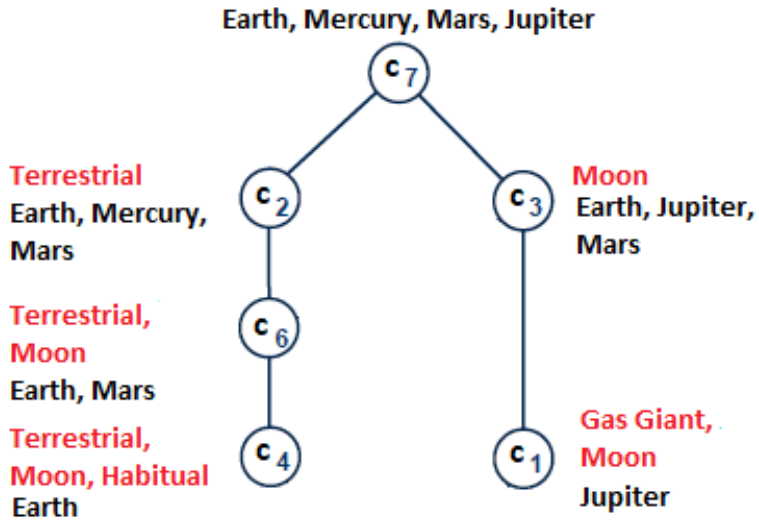

Figura 3.5 – Ávore correspondente ao diagrama *line* da Figura 3.3.

Para ilustrar a facilidade de entendimento de uma árvore de conceitos, utilizamos a Figura 3.5. O nó raiz (primeiro nó no topo) indica o início da árvore, onde temos a indicação de todos os objetos e nenhum atributo. Este nó, então, divide-se em dois nós devido à adição de dois atributos: *terrestrial* e *moon*. A cada adição de um novo atributo, um novo nó é formado. Portanto, estamos aumentado as características presentes nos objetos, deixando cada nó mais específico. Percebemos então uma diminuição no número de objetos em cada nó, sendo que os nós folha contêm todas as características dos objetos ali representados (normalmente um único objeto).

3.1 Visão geral do capítulo

Este capítulo apresentou os aspectos centrais da área de Análise de Conceitos Formais. As definições de contexto e conceitos formais auxiliam no entendimento de hierarquia de conceitos, que serão a base dos métodos de visualização diagrama *line* e árvore de conceitos, implementados na interface proposta nesta dissertação. Na literatura existem diversos algoritmos capazes de extrair conceitos formais de contextos formais. Alguns exemplos são: Close-by-One [19], InClose [2] e InClose2 [3]. Como a família RInClose é baseada no algoritmo InClose2, o capítulo 4 irá formalizar as explicações desses algoritmos.

4 Algoritmos de Coclusterização

Neste capítulo, iremos apresentar os principais aspectos do algoritmo InClose2 [3] e da família RIn-Close [38]. Esta família de algoritmos, desenvolvida por Veroneze [38], é conceitualmente baseada no algoritmo InClose2 [3]. Os algoritmos apresentados neste capítulo serão posteriormente utilizados para analisar as diferentes bases de dados apresentadas no capítulo 8.

4.1 Algoritmo In-Close2

O algoritmo In-Close2 [3] e seu predecessor In-Close [2] são baseados no algoritmo Close-By-One [19]. Esses algoritmos usam uma função recursiva para gerar combinações de objetos ou atributos na ordem lexicográfica. Esses algoritmos enumeram todos os coclusters constantes maximais numa matriz binária (conceitos formais de um contexto formal), de forma eficiente [3]. O algoritmo In-Close2 é computacionalmente mais rápido que o algoritmo In-Close porque o In-Close2 (i) permite que todos os elementos de um *intent* sejam herdados e (ii) implementa algumas otimizações e técnicas de pré-processamento para um uso eficiente da memória cache.

O Algoritmo 4.1 é uma adaptação do pseudo-código do In-Close2 [3] feita na dissertação de doutorado de Veroneze [38] para facilitar a leitura e o entendimento do In-Close2. A Tabela 4.1 descreve algumas variáveis utilizadas no pseudo-código.

Tabela 4.1 – Variáveis do pseudo-código 4.1

Variável	Definição
A_r	*extent* do r-ésimo conceito
B_r	*intent* do r-ésimo conceito
J_k	k-ésimo elemento do conjunto J
R_k	k-ésimo elemento do conjunto R

Na função principal, inicializamos $(A_1, B_1) \leftarrow (\{1, ..., n\}, \{\})$ (que é chamado de *supremum*), e $r_{new} \leftarrow 1$. Então, chamamos a função In-Close2 que fecha incrementalmente o conceito (A_1, B_1), começando do atributo com índice 1. Portanto, todos os conceitos serão descobertos de maneira recursiva. Durante o fechamento de um conceito, o In-Close2 itera sobre os atributos (linha 3). Se o atributo atual j não é um atributo herdado (linha 4), o In-Close2 computa o candidato para novo *extent* RW (linha 5). O

tamanho de RW deve ser maior or igual que o valor do parâmetro $minRow$ (linha 6). Se o candidato a *extent* RW é igual ao atual *extent* A_r (linha 7), então o atributo j é adicionado ao *intent* atual B_r (linha 8). Se o *extent* RW não for o mesmo que o *extent* atual A_r, o In-Close2 testa se RW é canônico (único) (linha 9). Se sim, o conceito atual (A_r, B_r) gerará um conceito filho (linhas 10 a 13). Após o fechamento do conceito atual (A_r, B_r), o In-Close2 começa o fechamento dos seus filhos (linhas 19 a 22) [38].

Pseudocódigo 4.1 In-Close2 (reproduzido de Veroneze [38])

Input: Matrix binária $\mathbf{D}_{n\times m}$, número mínimo de linhas $minRow$, índice do conceito a ser fechado r, índice do atributo inicial y

1: $J \leftarrow \{\}$
2: $R \leftarrow \{\}$
3: **for** $j \leftarrow y$ to m **do**
4: **if** $j \notin B_r$ **then**
5: $RW \leftarrow A_r \cap \{j\}'$
6: **if** $|RW| \geq minRow$ **then**
7: **if** $|RW| = |A_r|$ **then**
8: $B_r \leftarrow B_r \cup \{j\}$
9: **else if** RW is canonical **then**
10: $r_{new} \leftarrow r_{new} + 1$ // global variable
11: $J \leftarrow J \cup \{j\}$
12: $R \leftarrow R \cup \{r_{new}\}$
13: $A_{r_{new}} \leftarrow RW$
14: **end if**
15: **end if**
16: **end if**
17: **end for**
18: Store (A_r, B_r) in the solution $\mathfrak{B}$
19: **for** $k \leftarrow 1$ to $|J|$ **do**
20: $B_{R_k} \leftarrow B_r \cup \{J_k\}$ // R_k and J_k are the k-th elements of R and J, respectively
21: In-Close2$(\mathbf{D}, minRow, R_k, J_k + 1)$
22: **end for**

Além do número mínimo de linhas $minRow$, é possível adicionar um número mínimo de colunas $minCol$ ao algoritmo In-Close2. Enquanto o In-Close2 itera sobre os atributos, um conceito (A_r, B_r) pode ser descartado se, mesmo que adicionando todos os atributos restantes ao seu *intent*, ele não terá atingido o número mínimo de colunas $minCol$ (portanto, seus próximos descendentes também não iriam ter pelo menos $minCol$ colunas). Embora essa restrição consiga ser verificada apenas durante o fechamento de um conceito, ela irá reduzir o espaço de busca e diminuir o custo computacional porque (i) irá parar a construção de um conceito que será descartado posteriormente, dado que ele não atinge o número mínimo de colunas e (ii) previne que sejam gerados descendentes que não irão atingir a restrição $minCol$ também.

Para ilustrar como o In-Close2 opera, nós utilizaremos como exemplo o fechamento do supremum da Tabela 3.1. Portanto, seja $\mathbf{D}$ a matriz binária da Tabela 3.1. O

supremum é inicializado como $(A_1 = \{1,2,3,4\}, B_1 = \{\})$. Suponha que $minRow = 2$, a Tabela 4.2 mostra todos os *extents* RW computados durante o fechamento do *supremum* (A_1, B_1) (linhas 3 a 17 do pseudo-código 4.1). Assim, os descendentes de (A_1, B_1) são:

- $(A_2 = \{1,3,4\}, B_2 = \{2\})$,
- $(A_3 = \{1,2,4\}, B_3 = \{3\})$

O próximo passo do In-Close2 é fechar os descendentes de (A_1, B_1) recursivamente (linhas 19 a 22 do pseudo-código 4.1).

Tabela 4.2 – Fechamento do *supremum* (A_1, B_1) da Tabela 3.1

	RW	Situação
$j = 1$	$\{2\}$	$\|RW\| < minRow$
$j = 2$	$\{1,3,4\}$	é canônico
$j = 3$	$\{1,2,4\}$	é canônico
$j = 4$	$\{1\}$	$\|RW\| < minRow$

Para ilustrar, vamos calcular o fechamento do conceito (A_2, B_2). A Tabela 4.3 mostra todos os *extents* RW computados durante o fechamento. Assim os descendente de (A_2, B_2) são:

- $(A_4 = \{1,4\}, B_4 = \{2,3\})$.

O próximo passo do In-Close2 é fechar os descendentes de (A_2, B_2).

Tabela 4.3 – Fechamento do conceito (A_2, B_2) da Tabela 3.1

	RW	Situação
$j = 3$	$\{1,4\}$	é canônico
$j = 4$	$\{1\}$	$\|RW\| < minRow$

Este processo se repete até que o In-Close2 compute o fechamento de todos os conceitos que satisfaçam $minRow$. O resultado final é apresentado na Tabela 4.4.

4.2 Família RIn-Close

Veroneze [38] propôs generalizações do algoritmo In-Close2 para enumerar diversos tipos de coclusters (veja Figura 2.2) numa matriz numérica (não somente binária, mas também com valores pertencentes ao conjunto dos números reais ou inteiros). A família de algoritmos gerados foi denominada RIn-Close. Embora sejam extensíveis para

Tabela 4.5 – Características de planetas do Sistema Solar [18] - indicação dos coclusters da Tabela 4.4

	Gas giant	Terrestrial	Moon	Habital Zone
Earth	0	1	1	1
Jupiter	1	0	1	0
Mercury	0	1	0	0
Mars	0	1	1	0

Tabela 4.4 – Conceitos formais da matriz binária da Tabela 4.5 (utilizando $minRow = 2$ e $minCol = 1$).

#	Extent	Intent
1	$\{1, 3, 4\}$	$\{2\}$
2	$\{1, 2, 4\}$	$\{3\}$
3	$\{1, 4\}$	$\{2, 3\}$

mais dimensões, todos esses algoritmos foram concebidos para lidar com bases de dados bidimensionais, representáveis, portanto, por uma matriz de linhas e colunas.

Veroneze [38] decidiu adaptar o algoritmo In-Close2 porque: (1) trata-se de um algoritmo de fácil compreensão; (2) é considerado um algoritmo rápido na análise de conceitos formais; (3) possui suporte para determinar previamente o número mínimo de linhas em um cocluster; (4) permitiu incorporar um suporte para determinar a priori o número mínimo de colunas; e (5) o In-Close2 começa com todos os objetos no extent do conceito formal que dará origem a todos os outros, recursivamente. Este último aspecto é muito importante porque, por exemplo, quando o algoritmo RIn-Close procura por coclusters com valores constantes nas colunas, dado um atributo, é possível procurar por um subconjunto de linhas do extent que atendem uma restrição de similaridade ε [38].

Cabe mencionar que a família de algoritmos RIn-Close, proposta por Veroneze [38], é a única opção da literatura capaz de enumerar coclusters com valores constantes (perfeito ou perturbado) nas linhas e nas colunas, sendo a enumeração (1) eficiente, (2) completa, (3) não-redundante, portanto todos os coclusters são maximais, e (4) correta. No caso de coclusters com valores coerentes (ver Figuras 2.5 (d) e (e)) e com evolução coerente (mais especificamente OPSM, ver Figura 2.2 (f)), o RIn-Close não apresenta a propriedade (1), por minerar os coclusters com valores coerentes a partir de coclusters com valores constantes nas colunas minerados de uma matriz aumentada. Contudo, no caso de coclusters com valores coerentes perfeitos (Figuras 2.2 (d) e (e)), o RIn-Close é eficiente. A família RIn-Close não conta com um algoritmo para enumerar coclusters com valores constantes, visto que esse tipo de cocluster não é geralmente requisitado em aplicações práticas envolvendo matrizes numéricas reais.

Tabela 4.6 – Propriedades dos algoritmos da Família RIn-Close [38]

Tipo de Cocluster	Eficiente	Completo	Não-Redundante	Correto
Valores Constantes nas Linhas [1]	√	√	√	√
Valores Constantes nas Colunas [1]	√	√	√	√
Valores Coerentes Perfeitos	√	√	√	√
Valores Coerentes Perturbados	×	√	√	√
Evolução Coerente (OPSM)	×	√	√	√

Aqui explicaremos os princípios básicos de cada um dos algoritmos pertencentes à família RIn-Close. Para mais detalhes, referir-se à dissertação de doutorado de Veroneze [38].

4.2.1 Coclusters com valores constantes nas colunas: RIn-Close_CVC

A família RIn-Close conta com dois algoritmos para enumerar biclutsers com valores constantes nas colunas: (*i*) RIn-Close_CVC_P para o caso perfeito, e (*ii*) RIn-Close_CVC para o caso perturbado [38].

Segundo Veroneze [38], a adaptação do In-Close2 para minerar coclusters perfeitos com valores constantes nas colunas é direta, contendo apenas uma grande diferença: no In-Close2, cada cocluster (A_r, B_r) pode gerar apenas um descendente por atributo, enquanto que cada cocluster no RIn-Close_CVC_P pode gerar múltiplos descendentes por atributo. Isso acontece porque o In-Close2 busca por blocos de 1s, enquanto que o RIn-Close_CVC_P busca por blocos com valores constantes nas colunas [38].

Na Figura 4.1a, o In-Close2 está fechando o cocluster ($A_r = \{g_2, g_3, g_4, g_8, g_9, g_{10}, g_{11}, g_{15}\}, B_r = \{m_1, m_3, m_7\}$). Quando ele tenta adicionar o atributo m_8, o cocluster (A_r, B_r) gera um novo cocluster ($A = \{g_3, g_4, g_9, g_{15}\}, B = \{m_1, m_3, m_7, m_8\}$). Na figura 4.1b, o RIn-Close_CVC_P tenta fechar o cocluster (A_r, B_r). Porém, quando ele tenta adicionar o atributo m_8, o cocluster (A_r, B_r) gera quatro novos coclusters perfeitos com valores constantes nas colunas *sem sobreposição entre os seus extents*:

- (d1) ($A = \{g_{11}\}, B = \{m_1, m_3, m_7, m_8\}$),
- (d2) ($A = \{g_2, g_{15}\}, B = \{m_1, m_3, m_7, m_8\}$),
- (d3) ($A = \{g_8, g_9, g_{10}\}, B = \{m_1, m_3, m_7, m_8\}$), e
- (d4) ($A = \{g_3, g_4\}, B = \{m_1, m_3, m_7, m_8\}$).

[1] Perfeitos e Perturbados

Para criar o algoritmo RIn-Close_CVC (que enumera coclusters perturbados) é necessário uma adaptação mais elaborada do In-Close2, do que foi feito com o RIn-Close_CVC_P. Isso se deve ao fato de que, além de um cocluster poder gerar múltiplos descendentes por atributo, pode haver sobreposição entre os *extents* desses descendentes. Isso pode ser verificado na Figura 4.1 (c), onde o RIn-Close_CVC está fechando o cocluster ($A_r = \{g_2, g_3, g_4, g_8, g_9, g_{10}, g_{11}, g_{15}, g_{16}, g_{19}, g_{20}, g_{22}, g_{23}\}, B_r = \{m_1, m_3, m_7\}$). Quando ele tenta adicionar o atributo m_8, o cocluster (A_r, B_r) gera três novos cocluster *com sobreposição entre seus extents* (considerando $\epsilon = 1$):

- (d1) $(A = \{g_{11}, g_2, g_{15}\}, B = \{m_1, m_3, m_7, m_8\})$,
- (d2) $(A = \{g_2, g_{15}, g_8, g_9, g_{10}, g_{16}, g_{19}, g_{22}\}, B = \{m_1, m_3, m_7, m_8\})$, e
- (d3) $(A = \{g_8, g_9, g_{10}, g_{16}, g_{19}, g_{22}, g_3, g_4, g_{20}, g_{23}\}, B = \{m_1, m_3, m_7, m_8\})$.

4.2.2 Coclusters com valores coerentes: RIn-Close_CHV

Primeiramente, mostraremos como enumerar coclusters perfeitos com valores coerentes usando o algoritmo RIn-Close_CHV_P. Depois, iremos mostrar como o algoritmo RIn-Close_CHV enumera cocluters não-perfeitos com valores coerentes.

Para encontrar coclusters com valores coerentes, precisamos verificar se ocorre alguma coerência (aditiva ou multiplicativa) entre as colunas de uma matriz. Para isso, no RIn-Close_CHV_P, um cocluster começa com uma coluna no seu *intent*. Então, o RIn-Close_CHV_P minera todos os coclusters que possuem essa coluna nos seus *intents* [38]. Ou seja,

1ª Iteração o RIn-Close_CHV_P irá procurar todos os coclusters que possuem a coluna 1 nos seus *intents*;

2ª Iteração O RIn-Close_CHV_P irá procurar todos os que possuem a coluna 2 e não possuem a coluna 1 nos seus *intents*;

3ª Iteração o RIn-Close_CHV_P irá minerar todos os coclusters que possuem a coluna 3 e não possuem as colunas 1 e 2 nos seus *intents*;

4ª Iteração E assim sucessivamente.

O algoritmo RIn-Close_CHV utiliza os seguintes passos para minerar coclusters com valores coerentes, dada uma restrição de similaridade $\epsilon > 0$ que é definida pelo usuário [38].

	m_1	m_3	m_7		m_8
g_2	1	1	1		0
g_3	1	1	1		1
g_4	1	1	1		1
g_8	1	1	1	+	0
g_9	1	1	1		1
g_{10}	1	1	1		0
g_{11}	1	1	1		0
g_{15}	1	1	1		1

	m_1	m_3	m_7	m_8
g_3	1	1	1	1
g_4	1	1	1	1
g_9	1	1	1	1
g_{15}	1	1	1	1

(a) Geração de um único descendente pelo In-Close2.

	m_1	m_3	m_7		m_8
g_2	3	1	4		2
g_3	3	1	4		4
g_4	3	1	4		4
g_8	3	1	4	+	3
g_9	3	1	4		3
g_{10}	3	1	4		3
g_{11}	3	1	4		1
g_{15}	3	1	4		2

	m_1	m_3	m_7	m_8	
g_{11}	3	1	4	1	**d1**
g_2	3	1	4	2	**d2**
g_{15}	3	1	4	2	
g_8	3	1	4	3	
g_9	3	1	4	3	**d3**
g_{10}	3	1	4	3	
g_3	3	1	4	4	**d4**
g_4	3	1	4	4	

(b) Geração de múltiplos descendentes, d1, d2, d3 e d4, pelo RIn-Close_CVC_P.

	m_1	m_3	m_7		m_8
g_2	1	2	1		2
g_3	2	2	1		4
g_4	2	1	1		4
g_8	1	2	1	+	3
g_9	2	1	1		3
g_{10}	1	1	2		3
g_{11}	1	1	2		1
g_{15}	2	1	1		2
g_{16}	2	1	1		3
g_{19}	2	1	2		3
g_{20}	1	2	2		4
g_{22}	2	2	1		3
g_{23}	2	2	2		4

	m_1	m_3	m_7	m_8	
g_{11}	1	1	2	1	**d1**
g_2	1	2	1	2	**d2**
g_{15}	2	1	1	2	
g_8	1	2	1	3	**d**
g_9	2	1	1	3	
g_{10}	1	1	2	3	
g_{16}	2	1	1	3	
g_{19}	2	1	2	3	
g_{22}	2	2	1	3	
g_3	2	2	1	4	
g_4	2	1	1	4	
g_{20}	1	2	2	4	
g_{23}	2	2	2	4	

(c) Geração de múltiplos descendentes, d1, d2 e d3, pelo RIn-Close_CVC (considerando $\epsilon = 1$)

Figura 4.1 – Exemplo de descendentes gerados pelo (a) In-Close2, (b) RIn-Close_CVC_P e (c) RIn-Close_CVC (reproduzido de Veroneze[38])

- Cria-se uma matriz aumentada a partir da matriz original de dados $\mathbf{D}$, denominada $\mathbf{D}_a$, a qual contém a diferença entre todos os pares de colunas de $\mathbf{D}$. Por exemplo, a matriz aumentada $\mathbf{D}_a$ da matriz de dados $\mathbf{D}$ da Tabela 4.7 é ilustrada na Tabela 4.8, onde:

  $$d_1 = m_1 - m_2$$
  $$d_2 = m_1 - m_3$$
  $$d_3 = m_1 - m_4$$
  $$\vdots$$
  $$d_{10} = m_4 - m_5$$

- Aplica-se o algoritmo RIn-Close_CVC na matriz aumentada $\mathbf{D}_a$. A Figura 4.2 mostra todos os coclusters maximais com valores constantes nas colunas minerados pelo RIn-Close_CVC quando aplicado na matriz da Tabela 4.8 (utilizando $minRow = 2$ e $\epsilon = 1$).

- Mineram-se todos os coclusters maximais com valores coerentes a partir dos coclusters maximais com valores constantes nas colunas minerados pelo RIn-Close_CVC.

Para minerar os coclusters com valores coerentes a partir de coclusters com valores constantes nas colunas, temos que realizar os seguintes passos:

1. Encontrar as colunas da matriz D correspondentes ao *intent* do cocluster na matriz aumentada D_a. Por exemplo, no cocluster D6 na Figura 4.2, a coluna 1 de D_a corresponde à diferença entre as colunas 1 e 2 de D. Portanto, as colunas 1 e 2 são coerentes considerando $\{1, 3\}$ como *extent*.

2. Criar um grafo não-direcionado onde os nós representam as colunas de D e as arestas representam a conexão das colunas de D, ou seja, as colunas que são coerentes dado o *extent* do cocluster. No caso do cocluster D6, ele possui as colunas: 3 - 5, 1 - 2, 1 - 5, 2 - 3 e 2 - 5.

3. Encontrar cliques maximais deste grafo, pois todos os pares de colunas do cocluster têm que ser coerentes entre si. Um clique é um subgrafo em que todos seus pares de nós são adjacentes. Então, cada clique indica um coclusters com valores coerentes. Portanto, o cocluster D6 irá gerar dois coclusters com valores coerentes (Figura 4.3)

Tabela 4.7 – Exemplo de uma matriz de dados numéricos (reproduzido de Veroneze [38])

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	2	1	1	0	6
g_3	2	2	1	7	6
g_4	8	9	2	6	7

Tabela 4.8 – Matriz aumentada da matriz descrita na Tabela 4.7 (reproduzido de Veroneze [38])

	d_1 $m_1 - m_2$	d_2 $m_1 - m_3$	d_3 $m_1 - m_4$	d_4 $m_1 - m_5$	d_5 $m_2 - m_3$	d_6 $m_2 - m_4$	d_7 $m_2 - m_5$	d_8 $m_3 - m_4$	d_9 $m_3 - m_5$	d_{10} $m_4 - m_5$
1	-1	-1	0	-5	0	1	-4	1	-4	-5
2	1	1	2	-4	0	1	-5	1	-5	-6
3	0	1	-5	-4	1	-5	-4	-6	-5	1
4	-1	6	2	1	7	3	2	-4	-5	-1

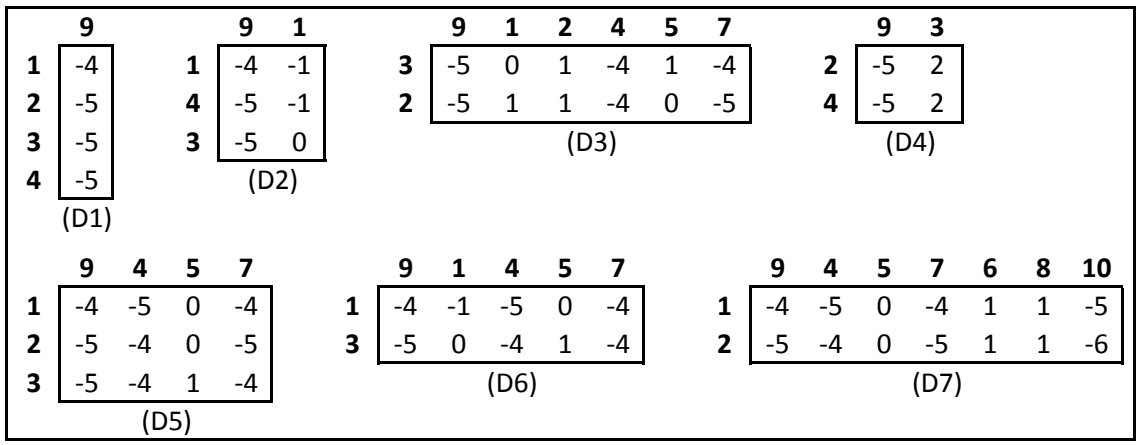

Figura 4.2 – Todos os coclusters com valores constantes nas colunas minerados a partir da Tabela 4.8 (utilizando $minRow = 2$ e $\epsilon = 1$) (reproduzido de Veroneze [38])

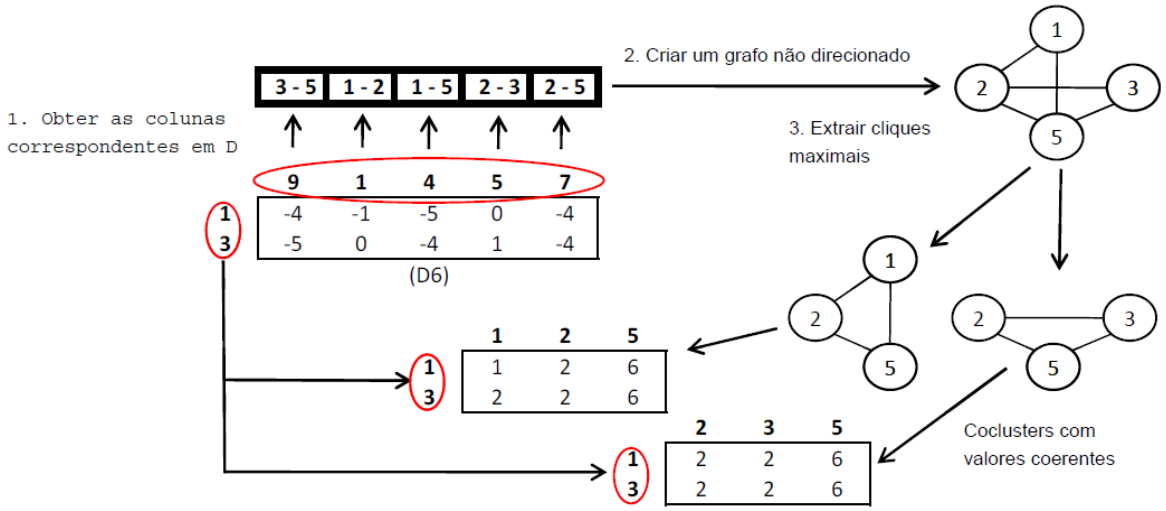

Figura 4.3 – Diagrama que indica os diferentes passos para se obter um cocluster com valores coerentes a partir de coclusters com valores constantes nas colunas (reproduzido de Veroneze [38])

4.3 Visão geral do capítulo

Neste capítulo, apresentamos os principais detalhes e aspectos do algoritmo Inclose2 e da família RIn-Close. Esses algoritmos serão utilizados para extrair coclusters das bases de dados do capítulo 8.

O próximo capítulo irá abordar o tema regras de associação, mais especificamente, regras de associação de classes. A partir de coclusters extraídos de algoritmos como a família RIn-Close, por exemplo, e de uma base de dados rotulados, podemos definir regras de associação de classes para analisar e selecionar coclusters.

5 Regras de Associação

Regras de associação são um dos objetos de estudo desta dissertação. Essas regras indicam combinações de itens que ocorrem em uma determinada frequência numa base de dados. Métodos de construção de regras de associação têm como premissa encontrar padrões frequentes em conjuntos de dados. Por exemplo, a partir de uma base de dados de uma padaria, ao usar regras de associação para analisar os dados, poderíamos gerar a seguinte regra: {leite, pão} $\rightarrow$ {manteiga}, ou seja, se um cliente comprar leite e pão, ele tende a comprar, com certo grau de certeza, também manteiga. Este grau de certeza é definido pelo fator de suporte e pelo fator de confiança [1].

Tabela 5.1 – Base de dados de transações em um supermercado.

Identificador	Itens
1	arroz, feijão, carne
2	chá, biscoito, pão
3	café, pão
4	manteiga, pão
5	café, kiwi, pão
6	arroz, feijão, pão

A Tabela 5.1 ilustra uma base de dados de transações hipotéticas de compras de um supermercado. Cada transação é constituída por um identificador único e por uma lista de itens. As regras de associação foram introduzidas em Agrawal *et al.* [1]. Seja então $\mathcal{I} = \{i_1, i_2, \cdots, i_m\}$ um conjunto de m itens distintos e $\mathcal{D}$ uma base de dados de transações, onde cada transação T é formada por um conjunto de itens (denominado em inglês *itemset*), tal que $T \subseteq \mathcal{I}$. Definimos uma regra de associação como:

Definição 5.1 *Regra de Associação*

É uma expressão na forma $A \Rightarrow B$, onde A e B são itemsets, $A \subset \mathcal{I}$, $B \subset \mathcal{I}$, $A \neq \emptyset$, $B \neq \emptyset$ e $A \cap B = \emptyset$. A é denominado antecedente e B é denominado consequente da regra.

Neste projeto, usaremos bases de dados multi-classes para gerar regras de associação. Nossas regras são, então, baseadas em regras de associação de classes construídas a partir dos coclusters extraídos de bases de dados numéricas usando o RIn-Close. Seja então A uma matriz numérica $n \times m$, onde n é o número de linhas e m o número de colunas. Seja (I, J) um cocluster de A, $|I|$ a quantidade de linhas do cocluster, e $|J|$ a

quantidade de colunas. Por fim, seja $\mathfrak{C} = \{1, \cdots, k\}$ o conjunto dos k rótulos que os objetos de uma base de dados podem assumir.

Definição 5.2 *Regra de Associação de Classes - CAR*

Uma regra de associação $X \Rightarrow Y$ é uma regra de associação de classes (do inglês, class association rule - CAR) se o seu consequente Y é um 1-itemset $Y = \{c\}$, onde $\{c\}$ é um rótulo pertencente ao conjunto $\mathfrak{C}$. Por simplicidade, usaremos $X \Rightarrow c$ para indicar uma regra que aponta para uma classe do conjunto $\mathfrak{C}$.

Essa definição de CAR considera um relação binária $R \subseteq G \times M$ entre os objetos G e os atributos M de uma base de dados. Contudo, estamos interessados, principalmente, em extrair regras de associação de classes em matrizes numéricas rotuladas. Então, CAR precisa de uma informação adicional no seu antecedente: uma intervalo de valores associado a cada atributo do antecedente. Portanto, o antecedente de uma regra $X \Rightarrow c$ será um conjunto de atributos, cada qual associado a um intervalo: $X = \{x_1[l_1, u_1], x_2[l_2, u_2], \cdots\}$. Por exemplo, uma regra $x_1[l_1, u_1], x_2[l_2, u_2] \Rightarrow 1$ significa que se um objeto possuir os atributos $x_1 \in [l_1, u_1]$ e $x_2 \in [l_2, u_2]$, então esse objeto está contido na classe com rótulo 1 com um certo grau de certeza..

5.1 Métricas de Validação

Diferentes métricas para regras de associação tentam quantificar a dependência entre o consequente e o antecedente de uma regra. Neste trabalho, procuramos coclusters que consigam definir determinadas classes. Por exemplo, se tivermos uma base que contenha níveis de expressão gênica de pacientes saudáveis e com diferentes tipos de câncer, estamos interessados em descobrir se algum (ou alguns) coclusters são capazes de definir quais genes causam estes diferentes tipos de câncer. Assim utilizamos 9 métricas comumente usadas na literatura para analisar as regras extraídas.

1. **Suporte e Suporte Relativo do *intent* de um Cocluster [38]**: definimos o suporte como:

$$sup(X) = |I| \tag{5.1}$$

e o suporte relativo ou frequência como:

$$rsup(X) = \frac{sup(X)}{n}, \tag{5.2}$$

onde n é o número de linhas da base de dados.

2. **Suporte e Suporte Relativo de uma Classe [38]**: seja c o rótulo de uma classe, i um objeto (linha da matriz) e $label(\dot{})$ uma função que dado um objeto retorna o rótulo dele, então definimos o suporte da classe c como:

$$sup(c) = |\{i \mid label(i) = c\}| \tag{5.3}$$

Definimos também o suporte relativo ou frequência de uma classe como:

$$rsup(c) = \frac{sup(c)}{n} \tag{5.4}$$

3. **Suporte e Suporte Relativo de uma Regra [38]**: o suporte da regra $X \Rightarrow c$ está definido na Equação 5.5, ou seja, o suporte de uma regra é a quantidade de vezes que a classe c aparece no cocluster com *intent* X

$$sup(X \Rightarrow c) = |I \cap \{i \mid label(i) = c\}| \tag{5.5}$$

definimos o suporte relativo ou frequência de uma regra $X \Rightarrow c$ como:

$$rsup(X \Rightarrow c) = \frac{sup(X \Rightarrow c)}{n} \tag{5.6}$$

4. **Confiança [38, 41]**: é a fração de predições corretas feita pela regra para uma classe c. Por exemplo, se a regra $\{G_1, G_4, G_{10}\} \Rightarrow linfoma$, onde G são genes, tiver uma confiança de $0.2/0.4 = 0.5$, significa que 50% dos pacientes (*extent*) do cocluster analisado possuem linfoma como tipo de câncer.

$$conf(X \Rightarrow c) = \frac{sup(X \Rightarrow c)}{sup(X)} \tag{5.7}$$

5. **Lift [41]**: é a razão do suporte observado desta regra de associação, para o suporte se X e c fossem estatisticamente independentes. Esta métrica é utilizada para avaliar a dependência entre o antecedente e o consequente.

$$lift(X \Rightarrow c) = \frac{conf(X \Rightarrow c)}{rsup(c)} = \frac{rsup(X \Rightarrow c)}{rsup(X) \times rsup(c)} \tag{5.8}$$

Se $lift(X \Rightarrow c) = 1$, então X e c_i são independentes. Se $lift(X \Rightarrow c) > 1$, então X e c são positivamente dependentes. Se $lift(X \Rightarrow c) < 1$, então X e c são negativamente dependentes. Quanto mais distante de 1 for o *lift* de uma regra, menos usual ela é.

6. **Leverage [41]**: indica o valor da diferença do suporte relativo real da regra e o suporte relativo esperado se X e c fossem estatisticamente independentes.

$$leverage(X \Rightarrow c) = rsup(X \Rightarrow c) - rsup(X) \times rsup(c) \tag{5.9}$$

   Se $leverage(X \Rightarrow c) = 0$, então X e c são independentes. Se $leverage(X \Rightarrow c) > 0$, então X e c são positivamente dependentes. Se $leverage(X \Rightarrow c) < 0$, então X e c são negativamente dependentes. Quanto mais distante de 0 for o *leverage* de uma regra, mais interessante, inovadora e não usual é a regra. Essa métrica deve ser usada conjuntamente com o lift. Essa métrica possui um intervalo: $]-0,25; +0,25[$ [5].

7. **Jaccard [41]**: coeficiente que mede a similaridade entre dois conjuntos, ou seja, mede o grau de sobreposição (*overlap*) entre as instâncias de X e as instâncias de c.

$$Jaccard(X \Rightarrow c) = \frac{sup(X \Rightarrow c)}{sup(X) + sup(c) - sup(X \Rightarrow c)} \tag{5.10}$$

8. **Completeness [38]**: é a proporção de instâncias que são previstas pela regra.

$$comp(X \Rightarrow c) = \frac{sup(X \Rightarrow c)}{sup(c)} \tag{5.11}$$

   Se $comp(X \Rightarrow c) = 1$, então o cocluster com *intent* X tem como *extent* todos os objetos da classe c.

Como um cocluster pode conter objetos (linhas) de classes distintas, calculamos essas métricas para cada uma das classes presentes no cocluster. Porém, os resultados serão apresentados apenas considerando a classe que o cocluster mais representa, ou seja, aquela classe com maior confiança. Cabe ressaltar que o antecedente da regra é formado pelos atributos do cocluster e os intervalos associados a cada atributo são obtidos com base nos objetos dos coclusters.

5.2 Visão geral do capítulo

Neste capítulo, definimos regras de associação e regras de associação de classes. A partir da definição de regras de associação de classes para coclusters, definimos diversas métricas de validação que serão usadas na interface proposta no capítulo 7 para analisar e selecionar coclusters relevantes.

O próximo capítulo irá indicar as principais interfaces descritas na literatura para visualizar, analisar e selecionar coclusters.

6 Interfaces de Coclusterização

Na literatura, existem alguns softwares para detecção e visualização de coclusters, como por exemplo: BiVisu [7], BicOverlapper [31], BicOverlapper2 [33], FCA Tree Tool [4] e SiBIC [14]. Neste capítulo, iremos explorar cada uma dessas interfaces indicando suas principais características e métodos de visualização e análise.

Dois métodos de visualização que comumente são utilizados para se analisar coclusters são: coordenadas paralelas e *heatmaps*.

O método mais utilizado para representar coclusters únicos são mapas de calor (do inglês *heatmaps*). Esses mapas são muito utilizados para representar o resultado de experimentos de expressão gênica. Cada ponto deste mapa representa a expressão de um gene utiilizando uma cor diferente. Contudo, eles são mais apropriados para uma estrutura *checkboard* ou tabuleiro de xadrex, própria de um microarranjo, o que faz com que eles possuam algumas limitações geométricas para representar diversos coclusters simultaneamente, especialmente se houver sobreposição [32]. Por conta desta desvantagem, os mapas de calor não serão utilizados na interface proposta.

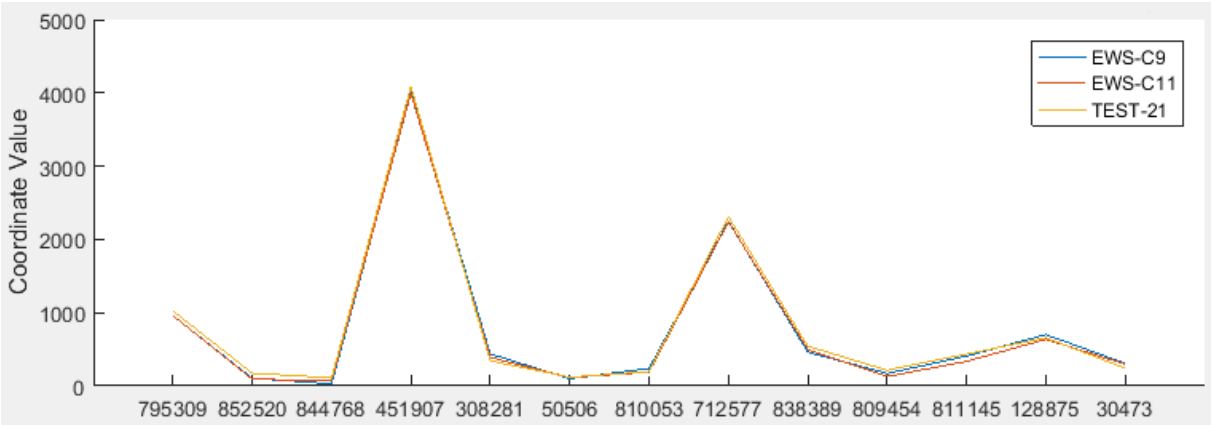

Figura 6.1 – Um cocluster com coerência aditiva visualizado por meio de coordenadas paralelas

Outro método muito utilizado para visualizar coclusters são coordenadas paralelas. Esse método é apropriado para visualizar um único cocluster por vez e não um conjunto de coclusters. Os atributos são visualizados no eixo vertical enquanto que os objetos são as linhas. Neste caso, um cocluster com n objetos e m atributos é representado por n linhas e um eixo horizontal com m condições. Quando se busca representar diversos coclusters com esse método, novamente temos limitações na visualização, por conta do grande número de linhas e a sobreposição de diversas delas [32].

6.1 BiVisu

BiVisu (veja exemplo da interface na Figura 6.2) é um software de acesso aberto em Matlab capaz de detectar e visualizar coclusters numa matriz de expressão gênica [7]. As principais caracterísitcas do BiVisu estão listadas abaixo [7]:

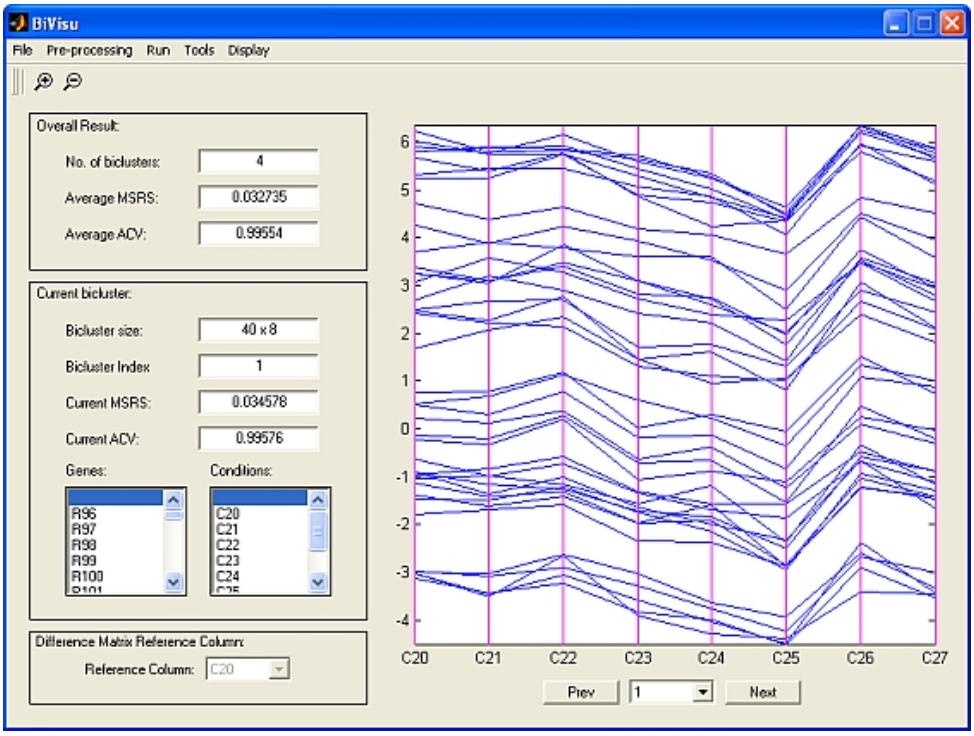

Figura 6.2 – Exemplo de execução da interface BiVisu [7] após a mineração de coclusters de uma matrz artificial com coclusters com valores coerentes aditivos.

- Detecção de coclusters (detalhes sobre a técnica e sobre garantia de enumeração são apresentadas a seguir) com valores constantes, valores constantes nas colunas, nas linhas e valores coerentes (do tipo aditivo e multiplicativo);
- Pré-processamento dos valores de expressão gênica utilizando logaritmo;
- Seleção de coclusters de acordo com o número mínimo de linhas, de colunas, número máximo de coclusters e máxima sobreposição permitida;
- Visualização de coclusters a partir de coordenadas paralelas e *heatmaps*;
- Coordenadas paralelas da diferença e razão de matrizes de coclusters com coerência aditiva e multiplicativa, respectivamente; e
- Indicação de duas métricas para avaliação dos coclusters encontrados: *mean square residue score* (MSRS) e *average correlation value* (ACV).

BiVisu utiliza o algoritmo criado por Cheng *et al.* [25]. Este algoritmo é uma versão *greedy* do algoritmo pCluster, que utiliza uma matriz aumentada, onde cada coluna desta matriz é formada pela diferença de pares de colunas da matriz original. Assim, os coclusters são minerados analisando a distribuição dos dados em cada coluna da matriz aumentada. Este algoritmo possui como parâmetros do usuário: número mínimo de linhas e colunas, sobreposição máxima permitida (em porcentagem) e um limite para ruído (ϵ).

O BiVisu permite visualizar coclusters utilizando *heatmaps* e coordenadas paralelas. Com relação ao último, o software é capaz de criar coordenadas paralelas de uma matriz de diferenças e de uma matriz de razão. No primeiro caso, a matriz de diferença é formada pela diferença, elemento a elemento, de todas as colunas do cocluster que será visualizado, com relação a uma coluna de referência (este método é indicado para visualizar coclusters com valores coerentes no modelo aditivo). No segundo caso, a matriz de razão é formada pela razão, elemento a elemento, de todas as colunas do coclusters que será visualizado, com relação a uma coluna de referência (este método é indicado para coclusters com valores coerentes no modelo multiplicativo) [7].

Apesar de conseguir minerar os mesmos tipos de coclusters que o RIn-Close, o BiVisu não assegura a maximalidade desses coclusters e também não assegura encontrar todos os coclusters, sendo estas diferenças fundamentais.

6.2 BicOverlapper

BicOverlapper é um software capaz de visualizar coclusters extraídos de uma matriz de expressão gênica. O método de visualização deste software é um grafo não-direcionado, onde coclusters são representados como grupos sobrepostos de genes e condições. Neste grafo, os nós representam genes ou condições e as arestas unem nós que estão presentes em um ou mais coclusters. Portanto, cada cocluster é um subgrafo não-direcionado completo (Figura 6.3 A) [31].

Para evitar um grande número de arestas sobrepostas, as arestas não são desenhadas. Como alternativa, os coclusters assumem uma forma circular que tem como posição o nó mais externo do cocluster (Figura 6.3 B). Os nós então são posicionados considerando um grafo com layout de força. Neste caso, cada par de nós é afetado por pelo menos duas forças. Se os nós estão conectados, uma força age para mantê-los juntos; se eles não estão conectados, uma força age para repelí-los. Neste caso, nós de um mesmo cocluster tendem a se manter juntos, enquanto que nós em coclusters diferentes tendem a se separar [31].

Ao contrário da interface BiVisu, a BicOverlapper possui duas grandes diferen-

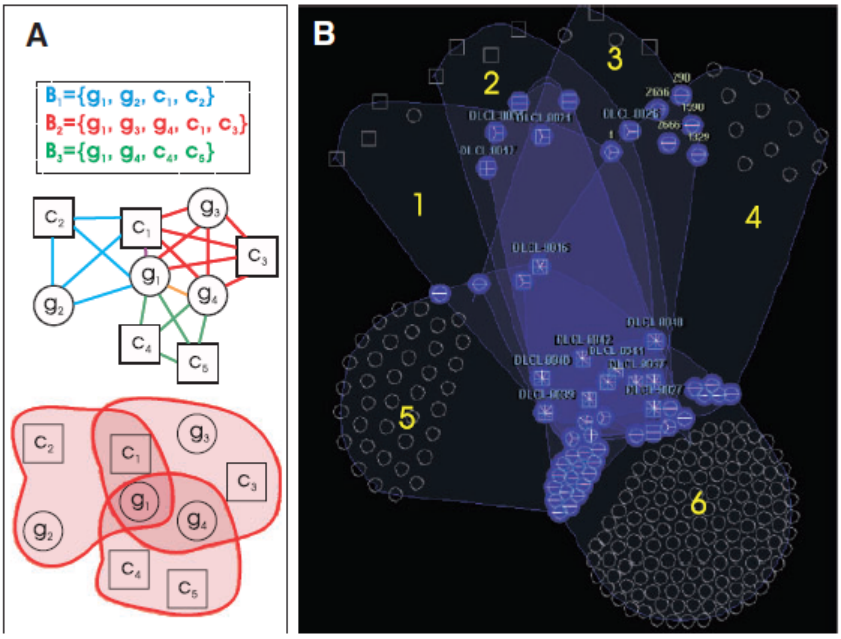

Figura 6.3 – (**A**) Três coclusters e suas representações. c_1 e g_1 aparecem nos coclusters B_1 e B_2, então a aresta entre eles é menor e eles aparecem na área de intersecção na visualização final. Áreas de intersecção são mais opacas para realçar a sobreposição. (**B**) Método de visualização principal do BicOverlapper, onde as áreas $(1, 2, 3)$ indicam o agrupamento de coclusters principalmente por conta das condições e as áreas $(4, 5, 6)$ por conta dos genes (reproduzido de Santamaria *et al.* [31]).

ças: (*i*) ela não é capaz de minerar coclusters, ela é apenas uma ferramenta de visualização; (*ii*) ela não é capaz de selecionar coclusters e não possui métricas de validação.

6.3 BicOverlapper2

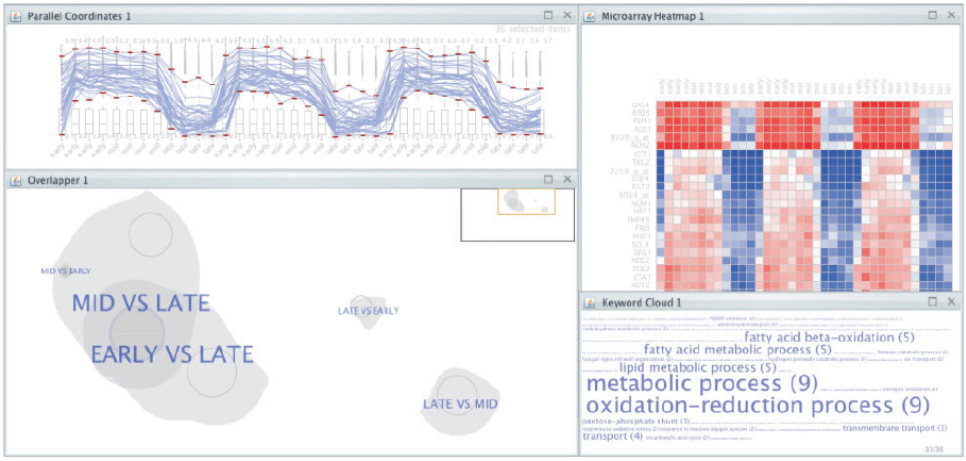

Figura 6.4 – Exemplo de execução da interface BicOverlapper2 (reproduzido de Santamaria *et al.* [33]).

O BicOverlapper2 (Figura 6.4) foi criado com o auxílio de sugestões de usuários

que utilizaram o BicOverlapper para fazer a análise de coclusters. Durante o desenvolvimento do BicOverlapper2, Santamaria *et al.* [33] focaram num software com uma interface mais simples e com mais métodos de visualização para auxiliar na análise dos coclusters sob diferentes pontos de vista, que é uma demanda essencial em biologia de sistemas.

O BicOverlapper2 mantém o método de visualização do BicOverlapper e adiciona coordenadas paralelas e *heatmaps* para indicar os níveis de expressão gênica de um cocluster. Contudo, esta interface continua sendo apenas uma ferramenta de visualização, sem ser capaz de minerar coclusters, filtrá-los ou computar métricas de validação.

Por ser uma interface voltada para matrizes de expressão gênica, ela utiliza bases de dados de *Gene Ontology* para buscar pelas anotações de genes, descobrindo assim os termos mais frequentes dentro de um cocluster.

6.4 FCA Tree Tool

Andrews & Hirsch [4] criaram a primeira interface de propósito geral para visualizar e explorar árvores de conceitos formais.

A *FCA Tree Tool* cria uma árvore de conceitos a partir de contextos formais. A árvore é criada como saída de uma versão modificada do algoritmo In-Close2 [3]. Com controles simples para expandir e colapsar os nós da árvore (Figura 6.5), essa interface se torna fácil de usar e as árvores produzidas são mais simples de se entender do que diagramas *line*, que são o método mais natural de visualização de conceitos hierárquicos.

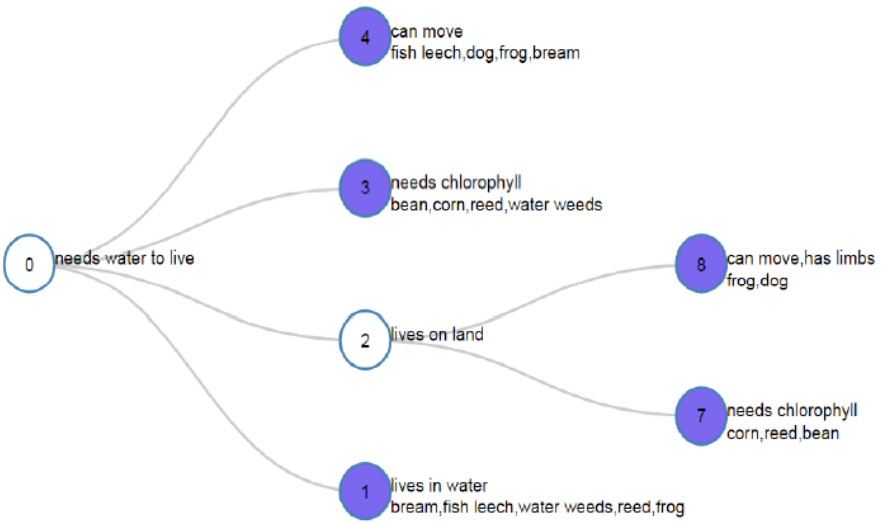

Figura 6.5 – Exemplo de execução da interface FCA Tree Tool, indicando a expansão do segundo conceito da base de dados *Live-In Water* (reproduzido de Andrews & Hirsch [4]).

Assim como os softwares BicOverlapper e BicOverlapper2, o FCA Tree Tool é uma ferramenta de visualização, não existindo a possibilidade de filtrar coclusters e verificar métricas de validação. Contudo, diferentemente dessas duas interfaces, o FCA Tree Tool possui seu próprio mecanismo para minerar coclusters, o In-Close2, se restringindo, consequentemente, a matrizes binárias ou discretizadas e itemizadas.

6.5 SiBIC

SiBIC é um servidor da *Web* (Figura 6.6) que, dada uma base de dados de expressão gênica, inicialmente irá enumerar exaustivamente os coclusters que depois serão combinados para formar um conjunto menor de coclusters; e finalmente, esse conjunto menor será resumido numa rede de conjuntos de genes, onde cada nó possui um conjunto de genes coexpressos sobre uma condição experimental em particular [14]. Utilizar uma rede de conjunto de genes como método de visualização é mais vantajoso que um diagrama *line* ou uma árvore de conceitos, pois permite que o número de nós do grafo se mantenha pequeno, já que na rede este número é igual ou menor ao número de genes na base de dados. Já em um diagrama *line* e numa árvore de conceitos, o número de nós é igual ao número de coclusters.

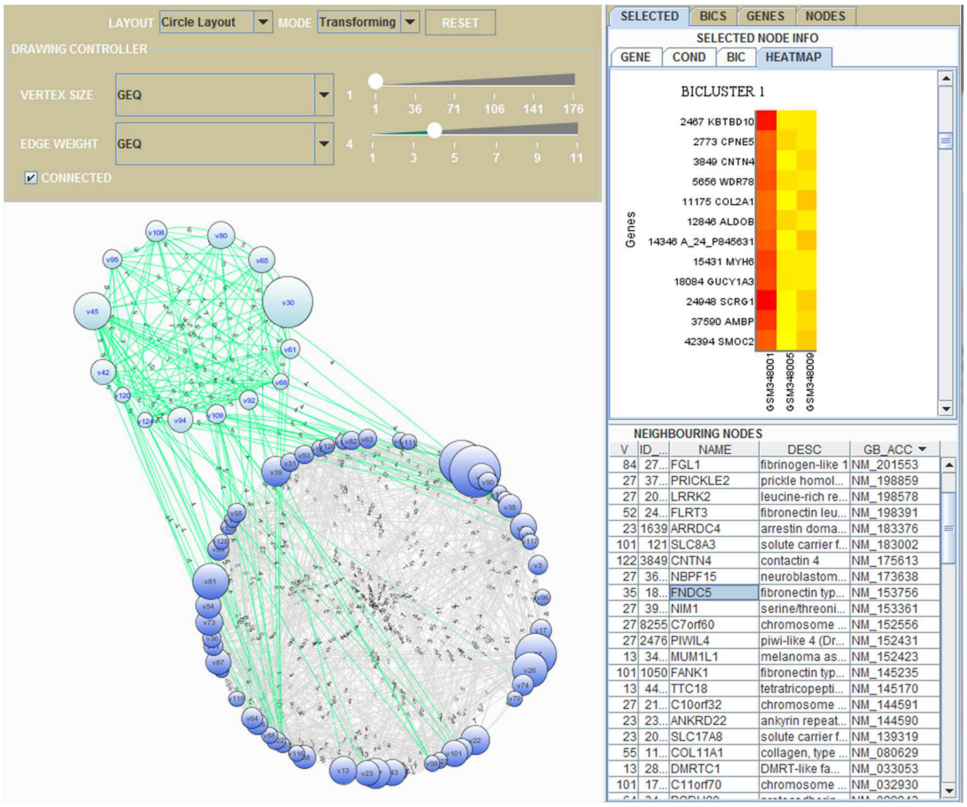

Figura 6.6 – Exemplo de execução da interface SiBIC (reproduzido de Takahashi *et al.* [14]).

SiBIC enumera apenas coclusters com valores constantes nas colunas. Para isso, ele utiliza o algoritmo MAFIA [6] da área de *frequent itemset mining* (FIM) para minerar coclusters maximais. Para fazer essa mineração, o SiBIC inicialmente pré-processa os dados para discretiza-los, ou seja, para cada condição experimental, o SiBIC primeiro gera itens, onde cada um possui um ou mais genes com os mesmo valores de expressão; este processo transforma a matriz de expressão gênica numa nova matriz, onde cada elemento é um conjunto de itens [14].

Um problema do algoritmo MAFIA [6] é que os coclusters encontrados são muitas vezes redundantes e sobrepostos. Portanto, o SiBIC combina dois coclusters se sua significância é mantida igual ou aumentada. Esse software usa o método de *p-values*, de acordo com a correlação de valores de expressão, para determinar a significância de um cocluster.

SiBIC, BicOverlapper e BicOverlapper2 possuem o mesmo método de visualização: uma rede. Enquanto o BicOverlapper e o BicOverlapper2 possuem grafos com nós representando um gene ou uma condição, o SiBIC utiliza como nós um conjunto de genes, que compartilham um mesmo padrão de expressão gênica. Isso faz com que o grafo se torne mais compacto que aquele produzido por BicOverlapper e BicOverlapper2, porque há uma redução no número de nós.

6.6 Visão geral do capítulo

Neste capítulo, apresentamos as principais interfaces encontradas na literatura para visualização e análise de coclusters. Com base nas vantagens de cada uma dessas interfaces, o capítulo 7 irá explicar com detalhes a interface proposta nesta dissertação, que aglutina todas essas vantagens e insere novos métodos de visualização, assim como métricas de validação, tornando seu elenco de funcionalidades o mais completo e diversificado da literatura.

7 Interface proposta para visualização, seleção e análise de coclusters

"A picture is worth a thousand words. An interface is worth a thousand pictures."
(Ben Shneiderman)

Quanto mais complexo e abstrato um conjunto de dados, mais importantes e relevantes os métodos de visualização se tornam. A tecnologia atual permite gerar enormes quantidades de dados, com estruturas complexas ou desconhecidas, em quase todas as áreas de pesquisa. Disciplinas como astronomia, biologia, física, química e sociologia necessitam da análise de grandes quantidades de dados, que podem ser estudados por métodos de coclusterização. Porém, a quantidade de coclusters gerados torna difícil a interpretação e representação dos resultados [34].

Análises numéricas exploram nossa inteligência abstrata (lógica e matemática) para compreender um problema. Contudo, ao utilizar recursos de visualização, é possível ampliar o entendimento e interpretação de problemas, principalmente os mais complexos e/ou caracterizados por processos e interações desconhecidos [34].

Neste capítulo, iremos mostrar e explicar a interface desenvolvida ao longo da pesquisa para compreender, visualizar, interpretar e analisar coclusters obtidos com o auxílio de diferentes algoritmos de coclusterização.

7.1 Aspectos técnicos de projeto da interface

Nesta seção, iremos rever alguns aspectos gerais sobre *design* de interfaces e iremos resumir a interface desenvolvida ao longo da pesquisa.

A diretriz mais usada para o *design* de interfaces é o *Visual Information-Seeking Mantra* [36]: *"Overview first, zoom and filter, then details on demand"*. Esse *mantra* pode ser aplicado em diversos tipos de dados, e é dividido nas seguintes tarefas:

Overview: obter uma visão geral dos dados.

Zoom: focar num subconjunto de interesse dos dados.

Filter: elimina dados redundantes e/ou irrelevantes.

Details-on-demand: seleciona um subconjunto para análises mais aprofundadas.

Relate: verifica relações entre conjuntos de dados.

History: armazena as ações do usuário para desfazer, refazer e refinar o processo de análise.

Extract: permite a exportação dos resultados para análises futuras.

A interface criada neste projeto é baseada neste *mantra* e em diversas interfaces já descritas na literatura [4, 7, 14, 31, 33]. Ela foi toda desenvolvida em ambiente MATLAB R2016a (The MathWorks, Inc., Natick, Massachusetts, United States), pois este é um software de alta performance voltado para o cálculo numérico usando matrizes. Sua linguagem fácil, sua depuração interativa e outras conveniências o transformam numa linguagem mais acessível do que C ou C++.

7.2 Informações relevantes ao usuário da interface proposta

A interface desenvolvida nesta dissertação é o resultado do nosso esforço para utilizar métodos de visualização, métricas de validação e regras de associação de classes para analisar resultados de coclusterização, podendo assim extrair as informações mais significantes de um conjunto potencialmente grande de coclusters.

Para facilitar a análise, a interface possui botões para aumentar e diminuir o zoom em cada gráfico (lupas indicadas com + e −, respectivamente, na Figura 7.1), assim como um botão para percorrer o gráfico após a aplicação do zoom (mão indicada na Figura 7.1). Além disso, funcionalidades como inserir variáveis (Figura 7.2), cálculo de métricas (Figura 7.3), criação e atualização de gráficos (Figura 7.4) também estão disponíveis na forma de menus. Eles são acessíveis a partir da barra de menus inicial (Figura 7.1)

Os arquivos que serão inseridos na interface utilizando o menu da Figura 7.2 devem ter extensão *.mat*. A Tabela 7.1 indica o formato que os dados devem ter para serem inseridos na interface.

Tabela 7.1 – Formato dos dados em cada arquivo que será inserido na interface.

Item do Menu	Formato dos dados
Load Data	matriz *double* com n linhas e m colunas
Load Attributes	vetor *cell* m-dimensional contendo o nome de cada atributo
Load Objects	vetor *cell* n-dimensional contendo o nome de cada objeto
Load Labels	vetor *cell* n-dimensional contendo o rótulo de cada classe
Load Coclusters	*struct* k-dimensional com dois campos, A e B, que indicam o *extent* e o *intent*, respectivamente, de cada um dos k coclusters.

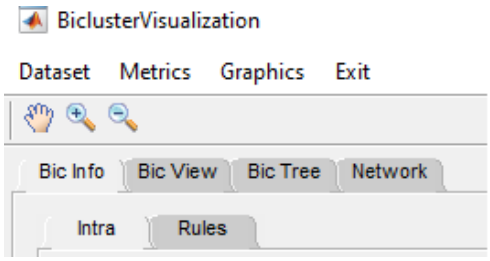

Figura 7.1 – Indicação dos botões para zoom e barra de menus principal.

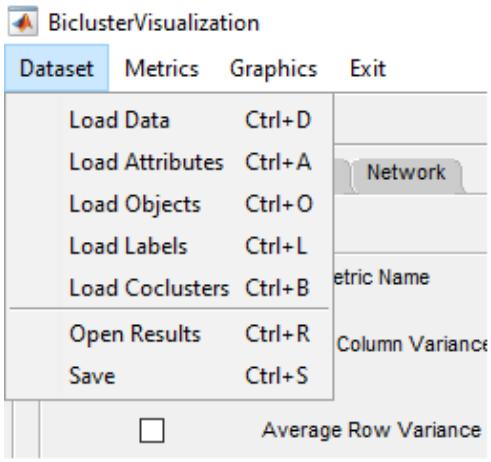

Figura 7.2 – Menu para importar uma base de dados, o rótulo dos seus atributos, o rótudo dos seus objetos, o rótulo das suas classes e os coclusters minerados desta base; importar os resultados criados por essa interface; e salvar os resultados.

Ela possui 4 painéis, onde o primeiro possui a função de análise utilizando métricas de validação intra-coclusters e regras de associação, e os outros três possuem a função de visualização.

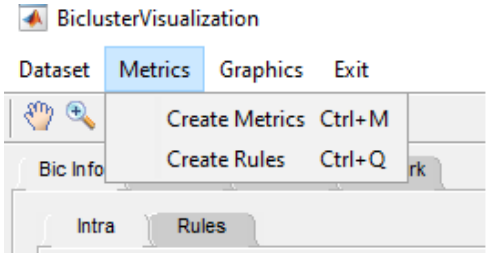

Figura 7.3 – Menu para gerar as métricas de validação intra-cocluster e as métricas de regras de associação de classes.

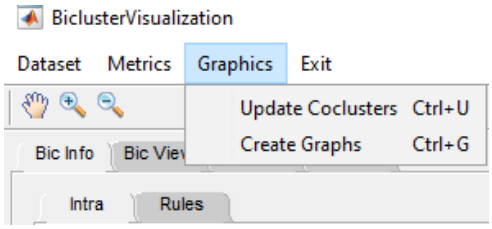

Figura 7.4 – Menu para atualizar os coclusters após a seleção das métricas no painel Bic Info; e criação dos gráficos para visualização dos coclusters.

7.2.1 Painel Bic Info

O painel Bic Info é formado por dois painéis internos que permitem ao usuário visualizar os resultados das métricas de validação e inserir intervalos numéricos para as métricas de validação intra-cocluster e de regras de associação de classes. Desta forma, é possível filtrar os coclusters de acordo com as métricas selecionadas.

Métricas de validação intra-cocluster: este painel (Figura 7.5), inspirado na interface BiVisu [7], permite a visualização (histogramas) das métricas de validação intra-coclusters (descritas na seção 2.1) e permite ao usuário inserir os intervalos numéricos para cada métrica desejada. Para criar as métricas de validação intra-cocluster, é necessário acessar o menu *Metrics* e clicar no item *Create Metrics*. Após clicar neste item, a interface irá requisitar que o usuário diga quantas matrizes aleatórias devem ser geradas para o cômputo da métrica *p-value*.

Métricas de Regras de Associação de Classes: este painel (Figura 7.6) permite a visualização (histogramas) das métricas de validação para as regras de associação de classes (descritas na seção 5.1). Da mesma maneira que o painel anterior, permite ao usuário inserir os intervalos numéricos para as métricas desejadas. Para criar as métricas de validação CAR, é necessário acessar o menu *Metrics* e clicar no item *Create Rules*.

Em ambos os painéis descritos acima, para selecionar uma métrica é necessário clicar no respectivo *checkbox*, inserir o valor da métrica na respectiva caixa *Value* e

selecionar o modo de filtro. Por exemplo, na Figura 7.5 a métrica *p-value* varia entre 0 (valor mínimo, destacado na caixa *Min*) e 1 (valor máximo, destacado na caixa *Max*). Logo, devemos inserir na respectiva caixa *Value* um valor α entre 0 e 1. Depois, devemos escolher o modo de seleção, entre os três permitidos, sendo = o modo padrão:

- $p\text{-}value = \alpha$
- $p\text{-}value > \alpha$, resultando em $\alpha < p\text{-}value \leqslant 1$
- $p\text{-}value < \alpha$, resultando em $0 \leqslant p\text{-}value < \alpha$

Após a seleção das métricas desejadas, é necessário acessar o menu *Graphics* e clicar no item *Update Coclusters*. A interface então procurará a intersecção dos coclusters que satisfazem as métricas selecionadas com seus respectivos valores. Para gerar os métodos de visualização, é necessário acessar o menu *Graphics* e clicar no item *Create Graphics*. Por padrão, ao atualizarmos os coclusters filtrados, a interface cria o método de visualização rede de objetos (Seção 7.2.4)

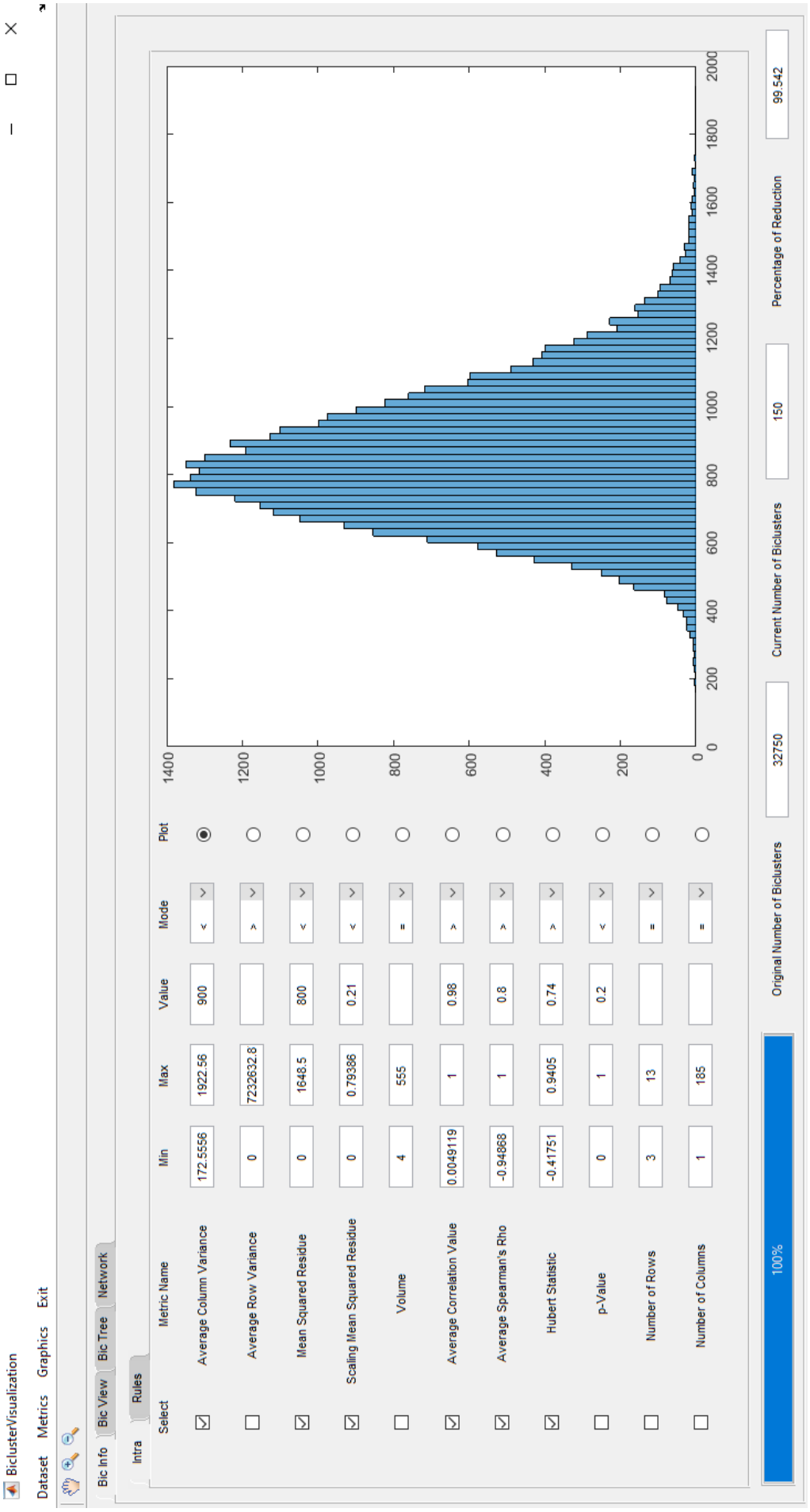

Figura 7.5 – Painel inicial da interface que mostra as métricas de validação intra-coclusters.

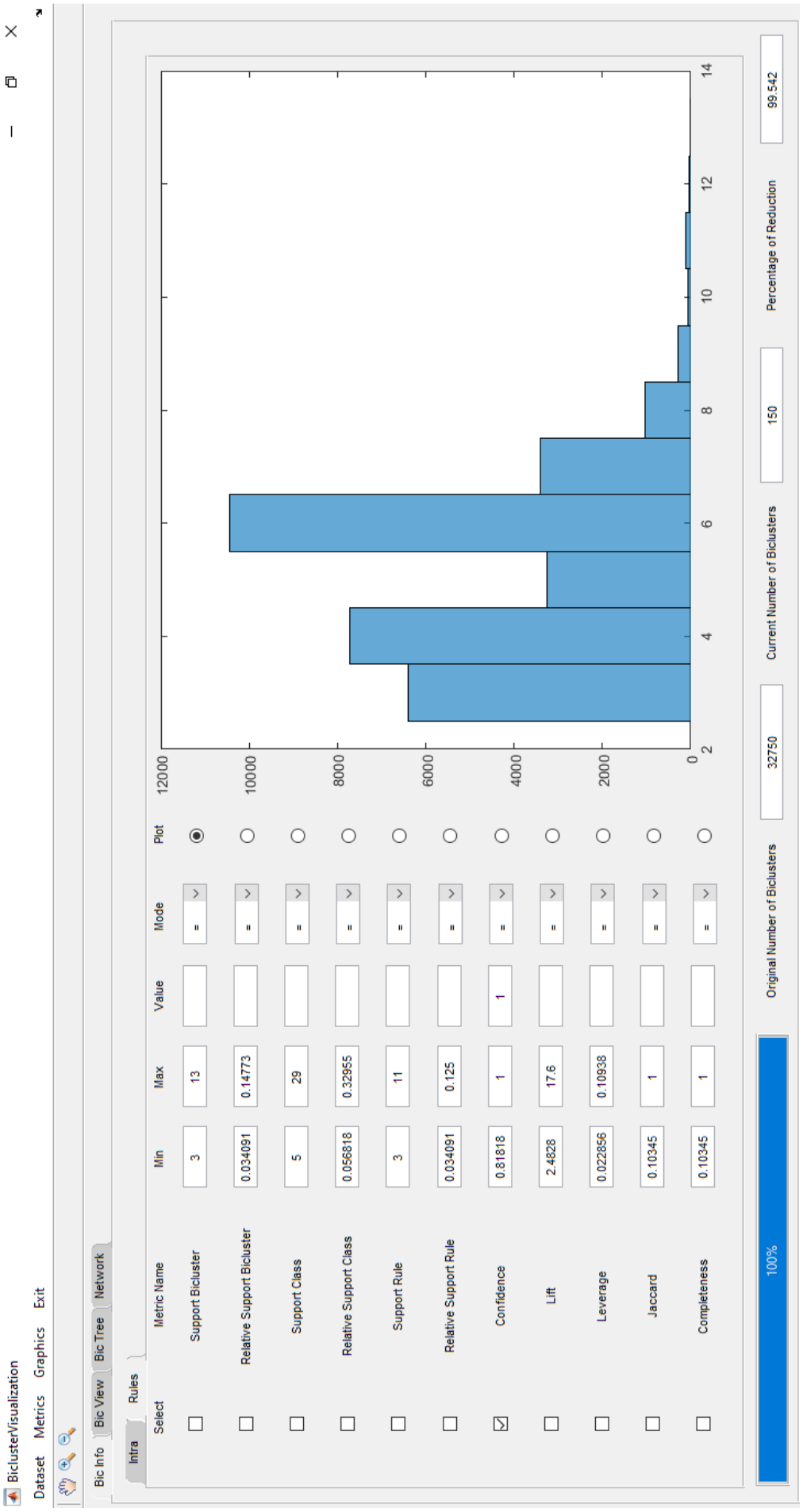

Figura 7.6 – Painel que indica as métricas para a regra de associação de classes.

7.2.2 Painel Bic View

O painel Bic View (Figura 7.7) possui três métodos de visualização de coclusters: coordenadas paralelas, diagrama *line* e árvore de conceitos. É possível selecionar coclusters a partir de suas tabelas, clicando no *checkbox* do atributo ou objeto desejado nas tabelas, ou clicando nos nós de cada grafo, que será destacado (tamanho maior). Ao selecionar um cocluster, sua posição é indicada na caixa de texto, seus valores são indicados no gráfico de coordenadas paralelas, e seus elementos (objetos e atributos) nas tabelas.

Este painel permite visualizar o diagrama *line* e a árvore de conceitos de objetos e atributos. A diferença entre essas visualizações é a maneira como os gráficos são construídos. No diagrama *line* e na árvore de conceitos de atributos, o primeiro nó de ambos é formado por todos os objetos e nenhum atributo, e os outros nós são construídos verificando-se a relação entre os atributos. No diagrama *line* e na árvore de conceitos de objetos, o primeiro nó de ambos é formado por todos os atributos e nenhum objeto, e os outros nós são construídos verificando-se a relação entre os objetos.

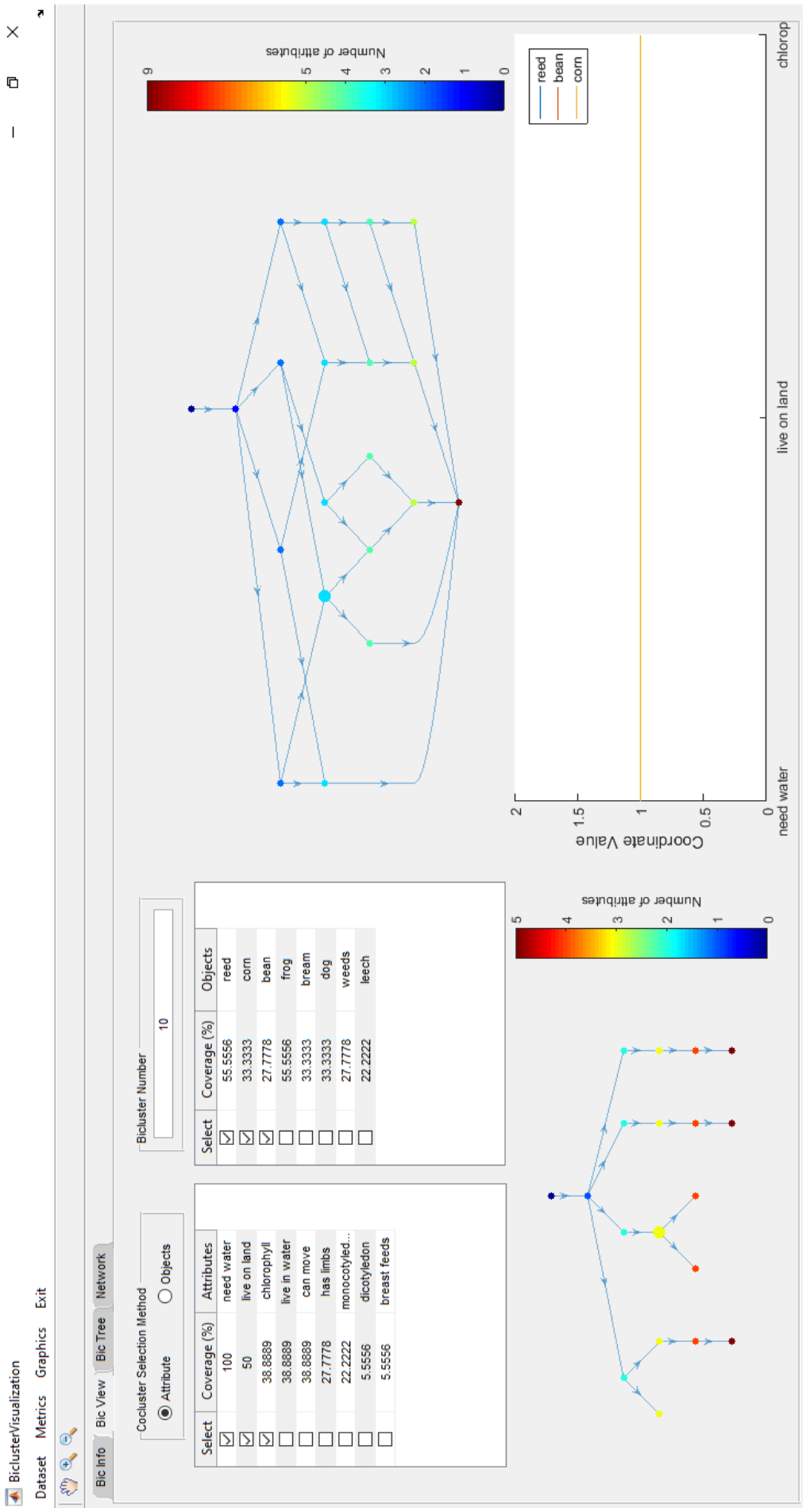

Figura 7.7 – Painel para visualização dos métodos diagrama *line*, árvore de conceitos e coordenadas paralelas.

7.2.3 Painel Bic Tree

O painel Bic Tree (Figura 7.8) é inspirado na interface criada por Andrews & Hirsch [4]. Nele, podemos visualizar e caminhar na árvore de conceitos. Com controles simples para expandir e colapsar nós da árvore, é possível explorar a árvore de uma maneira simples e direta.

Para expandir um nó da árvore, é necessário clicar com o botão direito do mouse no nó desejado. Para colapsar um nó da árvore, é necessário clicar com o botão esquerdo do mouse no nó desejado. Os nós indicados em vermelho são passíveis de colapso, enquanto que os nós indicados em azul são passíveis de expansão.

O botão *Fully Expanded* permite visualizar a árvore com todos os seus nós (como indicado na Figura 7.8). Já o botão *One Level* irá indicar apenas o nó raiz.

Se os *checkboxes Show Attributes* e *Show Objects* estiverem selecionados, cada nó receberá um rótulo com seu respectivo atributo e objeto. O *checkbox Count* indica em porcentagem quantos atributos cada nó possui.

Devido à possível grande quantidade de atributos e/ou objetos, o usuário pode alterar o tamanho da fonte da letra dos rótulos, assim como rotacioná-los, inserindo valores nas caixas *Font Size* e *Rotation*, respectivamente.

Figura 7.8 – Painel para visualização e para percorrer a árvore de conceitos como mostrado por Andrews & Hirsch [4].

7.2.4 Painel Network

O painel Network (Figura 7.9) é inspirado nas interfaces criadas em Takahashi *et al.* e Santamaria *et al.* [14, 31]. Nele, mostramos um grafo onde nós representam as linhas (objetos) da base de dados e as arestas ligam nós que estão agrupados por um ou mais coclusters, onde o peso de cada aresta é o número de coclusters.

Este painel permite ao usuário escolher:

- Se os nós terão cores diferentes para indicar as diferentes classes dos objetos (alterar entre *Yes* e *No* no painel *Show Class*);
- Se os nós terão um rótulo indicando seus respectivos nomes (alterar entre *Yes* e *No* no painel *Node Name*);
- Se as arestas receberão um rótulo para indicar seu peso e serão realçadas para indicar a quantidade de coclusters a que elas se referem (alterar entre *Yes* e *No* no painel *Edge Weight*);
- O método de *layout* do grafo que será visualizado: automático, círculo (*Circle*), força (*Force*) ou em camadas (*Layered*).

Neste painel, é possível também selecionar um ou mais nós (alterar entre *Single* e *Multiple* no painel *Selection Mode*) ao clicar no botão *Select*. O(s) nó(s) selecionado(s) serão indicados na caixa de texto *Nodes Selected* e seus vizinhos na caixa *Neighbours Nodes*. Para desfazer a seleção, é necessário clicar no botão *Reset*.

7.3 Visão geral do capítulo

Neste capítulo foi apresentada a principal contribuição desta dissertação, ou seja, a interface proposta para análise, visualização e seleção de coclusters. Tendo em mãos esta ferramenta de visualização e análise de coclusters, no capítulo 8 utilizaremos bases de dados binárias para validar os métodos de visualização e bases de dados reais para estudar os métodos de validação intra-coclusters e de regras de associação de classes.

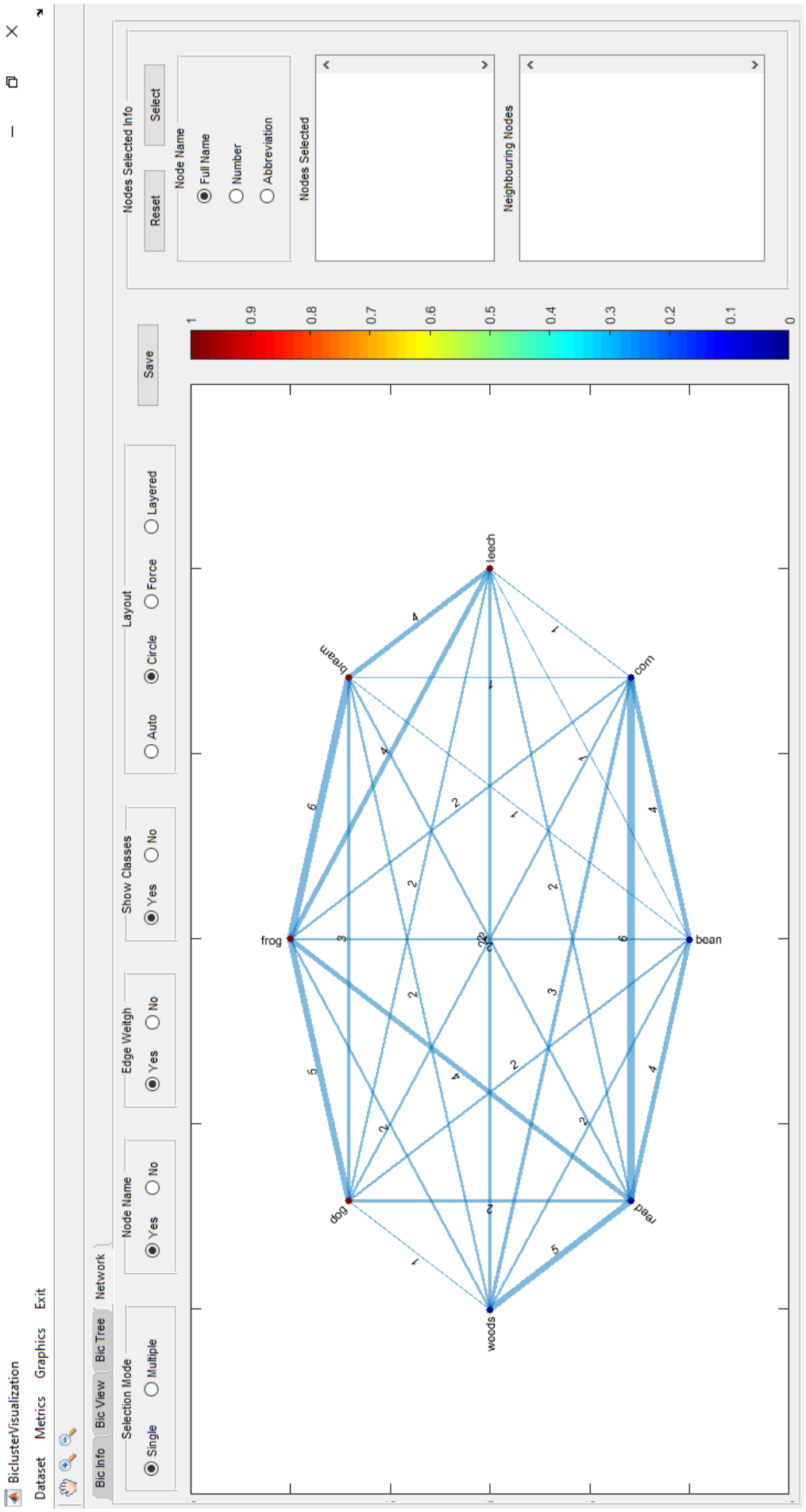

Figura 7.9 – Painel para visualização da rede de um conjunto de objetos, inspirada em Takahashi *et al.* [14] e Santamaria *et al.* [31].

8 Aplicação da interface proposta

A interface descrita no Capítulo 7 permite ao usuário analisar coclusters obtidos através de diferentes métodos de coclusterização. Assim, o usuário tem a possibilidade de obter uma visão geral do resultado da coclusterização. Esta interface possui como métodos de visualização: diagrama *line*, árvore de conceitos, coordenadas paralelas e redes de objetos. Aliado a isso, uma ampla gama de filtros podem ser definidos pelo usuário para obter subconjuntos mais representativos de coclusters, a partir do conjunto inicial fornecido como entrada, empregando para tal fim métricas de validação intra-coclusters e/ou métricas de regras de associação de classes..

Neste capítulo, iremos discutir quatro casos de estudo. Os dois primeiros são casos de controle, com matrizes binárias, para demonstrar a importância e a necessidade de se ter diferentes métodos de visualização para auxiliar na análise dos dados. Após a análise desses casos envolvendo bases binárias, iremos explicar resumidamente o significado de sequenciamento de RNA, tendo em vista que as bases de dados dos dois últimos casos foram obtidas usando esta técnica. Nos dois últimos casos, usaremos bases reais de dados de expressão gênica para selecionar coclusters usando métricas de validação, sendo que no último caso de estudo a análise empregando coclusterização é inédita..

8.1 Planetas do Sistema Solar

A Tabela 8.1 mostra uma matriz binária em que os objetos se referem a planetas do Sistema Solar e os atributos se referem as suas características. Usando o algoritmo InClose2 [3] com $minRow = 1$ e $minCol = 1$, mineramos os coclusters desta matriz, que estão descritos na Tabela 8.2.

Tabela 8.1 – Características de planetas do Sistema Solar [18].

	Gas giant	Terrestrial	Moon	Habital Zone
Earth	0	1	1	1
Jupiter	1	0	1	0
Mercury	0	1	0	0
Mars	0	1	1	0

Tabela 8.2 – Conceitos extraídos da Tabela 8.1 usando o algoritmo InClose2 [3] com $minRow = 1$ e $minCol = 1$.

Cocluster	Extent	Intent
1	{Jupiter}	{Gas giant, Moon}
2	{Earth, Mercury, Mars}	{Terrestrial}
3	{Earth, Jupiter, Mars}	{Moon}
4	{Earth}	{Terrestrial, Moon, Habital Zone}
5	{Earth, Mars}	{Terrestrial, Moon}

Ao utilizarmos a interface desenvolvida ao longo da pesquisa, obtemos como diagrama *line* a Figura 8.1 e como respectiva árvore de conceitos a Figura 8.2. Em cada nó desses grafos, o texto superior indica os objetos e o texto inferior indica os atributos de cada cocluster. Ao compararmos com as Figuras 3.3 e 3.5, percebemos que elas são idênticas, indicando que a interface desenvolvida é capaz de reproduzir os resultados obtidos na literatura.

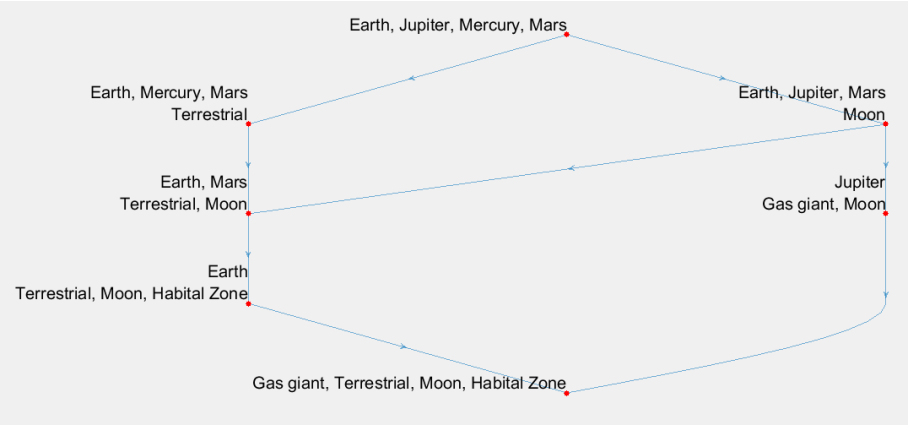

Figura 8.1 – Diagrama *line* criado a partir da Tabela 8.2 usando a interface desenvolvida ao longo da pesquisa.

Figura 8.2 – Árvore de conceitos criada a partir da Tabela 8.2 usando a interface desenvolvida ao longo da pesquisa

8.2 Live In Water

O segundo caso de estudo é um experimento controlado, pois utilizamos uma base de dados previamente estudada e com um número pequeno de coclusters para validar a interface criada. A Tabela 8.3 é um contexto formal que mostra os objetos (animais e plantas), seus atributos e a relação entre eles.

Tabela 8.3 – Base de dados *Live in Water* [4, 28].

	needs water to live	lives in water	lives on land	needs chlorophyll	dicotyledon	monocotyledon	can move	has limbs	breast feeds
fish leech	1	1	0	0	0	0	1	0	0
bream	1	1	0	0	0	0	1	1	0
frog	1	1	1	0	0	0	1	1	0
dog	1	0	1	0	0	0	1	1	1
water weeds	1	1	0	1	0	1	0	0	0
reed	1	1	1	1	0	1	0	0	0
bean	1	0	1	1	1	0	0	0	0
corn	1	0	1	1	0	1	0	0	0

Utilizando a base de dados *Live in Water* descrita na Tabela 8.3, obtivemos todos os coclusters desta base usando o algoritmo InClose2 com $minRow = 1$ e $minCol = 1$ [3], que estão descritos na Tabela 8.4.

A partir dos coclusters obtidos, a interface cria um diagrama *line*, que está representado na Figura 8.5, e uma árvore de conceitos (Figura 8.6), para auxiliar na visualização e no entendimento destes coclusters. As Figuras 8.3 e 8.4 foram obtidas da literatura e indicam um diagrama *line* e uma árvore de conceitos para a base *Live in Water*. No diagrama *line* da Figura 8.3, o atributo principal de cada nó está destacado em um semi-círculo azul e o objeto principal de cada nó em um semi-círculo preto.

Comparando essas quatro figuras, podemos validar os métodos de visualização desenvolvidos por nossa interface. Percebemos que os diagramas *line* são praticamente iguais, tendo como diferença a posição dos nós e a inserção de um nó a mais na interface proposta aqui. Este nó a mais é o primeiro, que contém todos os objetos e nenhum atributo.

Nas árvores de conceitos (Figuras 8.4 e 8.6), percebemos uma diferença maior na localização de cada cocluster (nó). Isso ocorre devido à maneira com que as árvores são

Tabela 8.4 – Coclusters extraídos usando o algortimo InClose2 [3] com $minRow = 1$ e $minCol = 1$ da base de dados *Live in Water*.

Cocluster	**Linhas**	**Colunas**
1	1; 2; 3; 4; 5; 6; 7; 8	1
2	1; 2; 3; 5; 6	1; 2
3	3; 6	1; 2; 3
4	6	1; 2; 3; 4; 6
5	3	1; 2; 3; 7; 8
6	5; 6	1; 2; 4; 6
7	1; 2; 3	1; 2; 7
8	2; 3	1; 2; 7; 8
9	3; 4; 6; 7; 8	1; 3
10	6; 7; 8	1; 3; 4
11	7	1; 3; 4; 5
12	6; 8	1; 3; 4; 6
13	3; 4	1; 3; 7; 8
14	4	1; 3; 7; 8; 9
15	5; 6; 7; 8	1; 4
16	5; 6; 8	1; 4; 6
17	1; 2; 3; 4	1; 7
18	2; 3; 4	1; 7; 8

construídas. Na interface FCA Tree Tool [4], o InClose2 simplesmente armazena apenas a primeira ocorrência de cada conceito filho, não permitindo assim que um conceito filho esteja conectado a mais que um conceito pai (como ocorre no diagrama *line*). Por outro lado, na nossa interface, a árvore é montada verificando a relação entre os atributos de cada conceito, ou seja, o primeiro nó não terá nenhum atributo, e a cada nó iremos adicionando um atributo, até varrer todos os coclusters. Decidimos montar a árvore de conceitos desta maneira para generalizar sua montagem e não ter a necessidade de vincular a sua criação a um algoritmo em particular.

Para ilustrar a facilidade de entendimento de uma árvore de conceitos, utilizamos a Figura 8.6, que foi obtida a partir dos coclusters da Tabela 8.4. O nó raiz (primeiro à esquerda) indica o início da árvore, onde temos a indicação de todos os objetos e nenhum atributo. No segundo nó, temos a adição de um atributo (*need water*). Como todos os objetos são dos reinos *Animalea* ou *Plantae*, temos novamente o conjunto completo de objetos presentes neste nó. Este nó então se divide em 4 ramos com a adição de mais um atributo (*can move* ou *chlorophyll* ou *live on land* ou *live in water*) e a cada adição de um novo atributo, um novo nó é criado. Com a inserção de um nó, estamos aumentando as características presentes nos objetos. Percebemos, então, uma diminuição no número de objetos em cada nó. É possível observar também que alguns nós-folha contêm um único objeto, com todas as características (atributos) pertencentes a esse objeto.

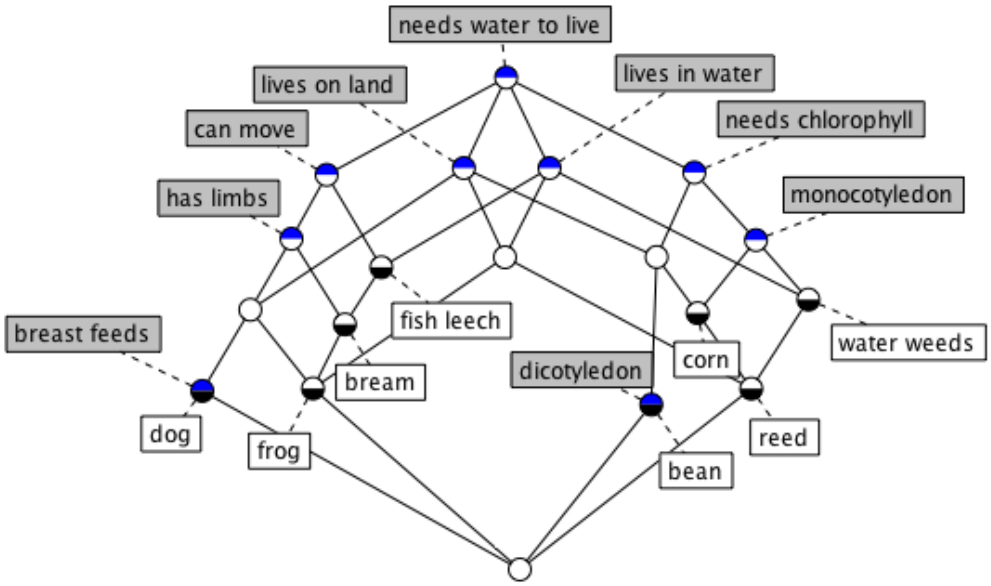

Figura 8.3 – Diagrama *line* criado a partir da Tabela 8.4, usando o software Conexp [28].

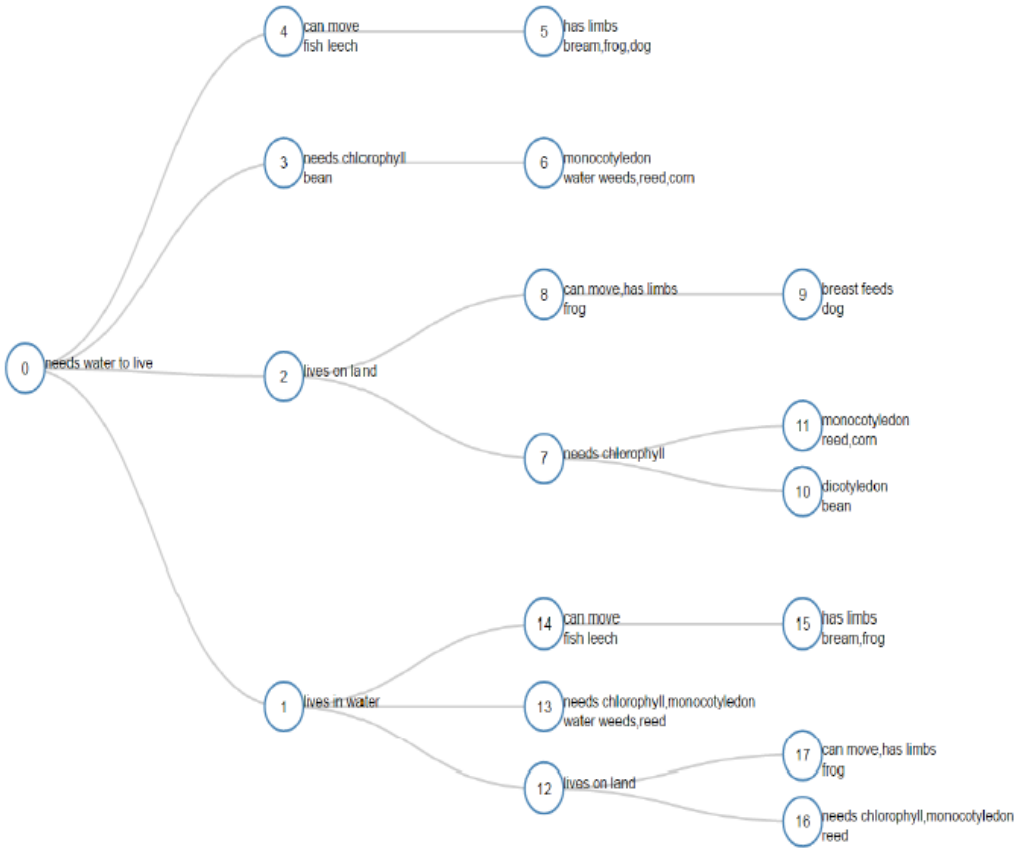

Figura 8.4 – Árvore de conceitos criada a partir da Tabela 8.4, usando o software FCA Tree Tool [4].

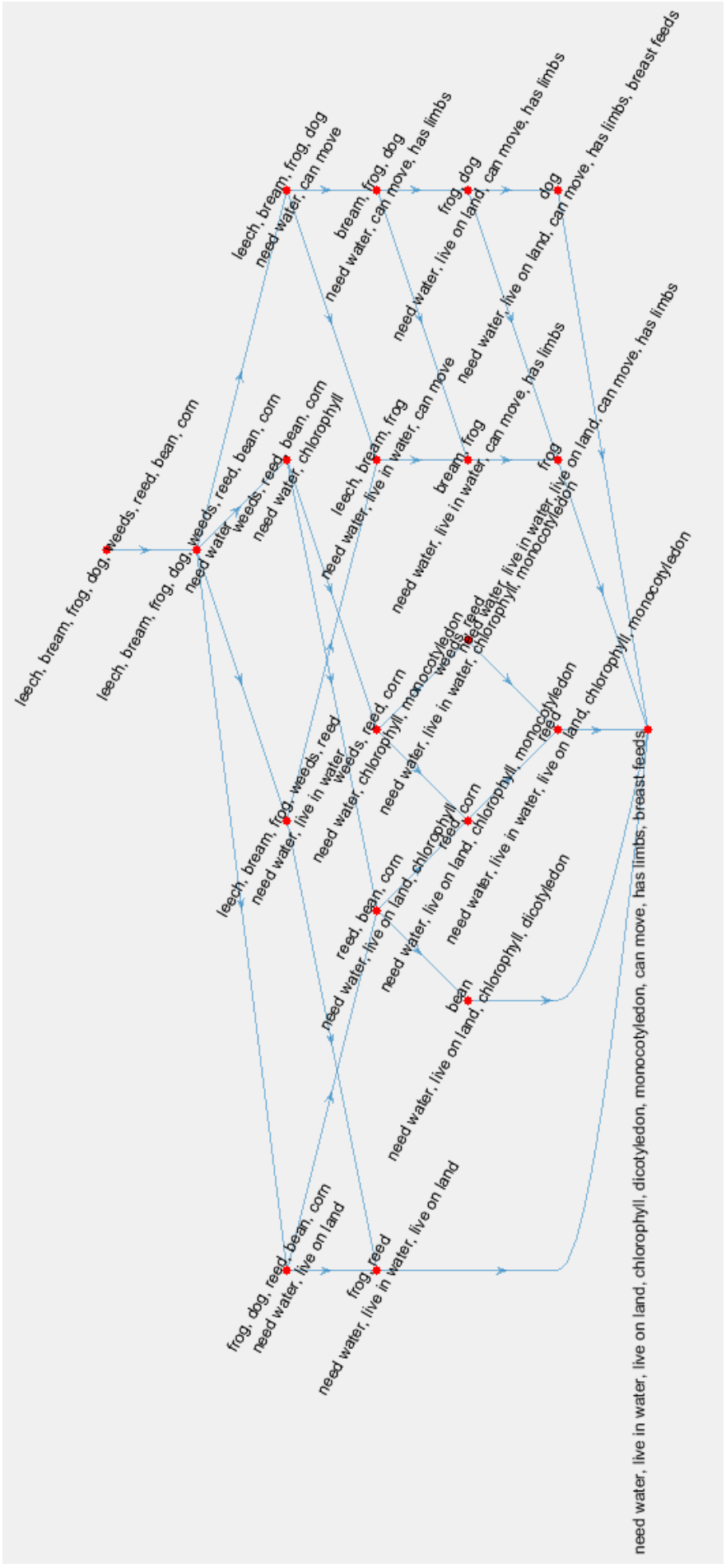

Figura 8.5 – Diagrama *line* criado a partir da Tabela 8.4 pela interface desenvolvida ao longo da pesquisa.

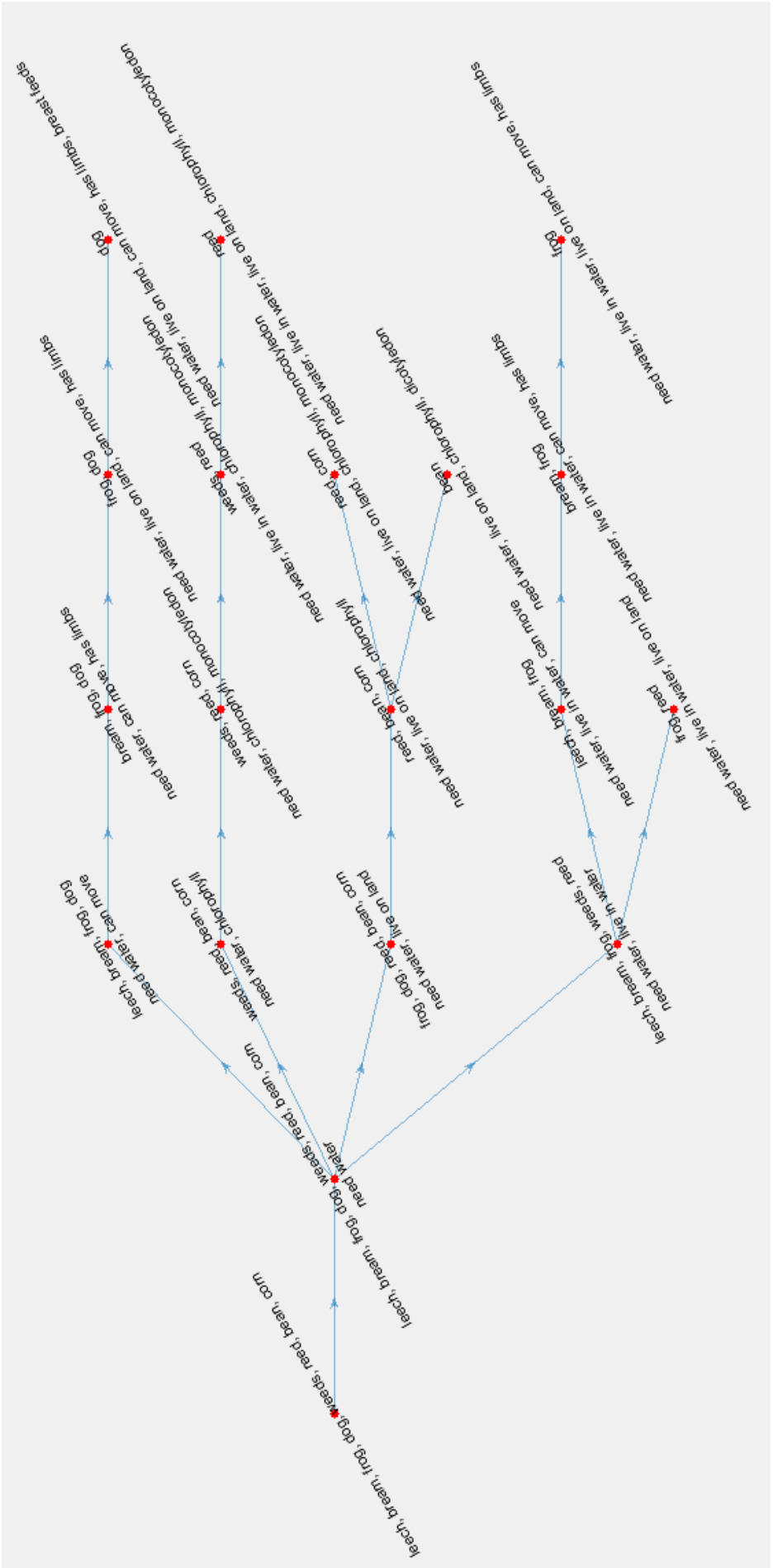

Figura 8.6 – Árvore de conceitos totalmente expandida criada a partir da Tabela 8.4 pela interface desenvolvida ao longo da pesquisa

Takahashi *et al.* [14] e Santamaria *et al.* [31] utilizam em suas interfaces um grafo onde os nós representam os objetos (linhas) da base de dados, as arestas unem os nós que estão agrupados por um ou mais coclusters e o peso da aresta é a quantidade de coclusters. Portanto, cada cocluster é representado por um subgrafo completo. Esta visualização é mais simples e reduz o número de nós e arestas significativamente, quando comparado a um diagrama *line* ou a uma árvore de conceitos. A Figura 8.7 mostra a rede de objetos obtida a partir das Tabelas 8.3 e 8.4, empregando a interface desenvolvida ao longo desta pesquisa. O resultado é equivalente àquele que seria produzido pelas interfaces de Takahashi *et al.* [14] e de Santamaria *et al.* [31].

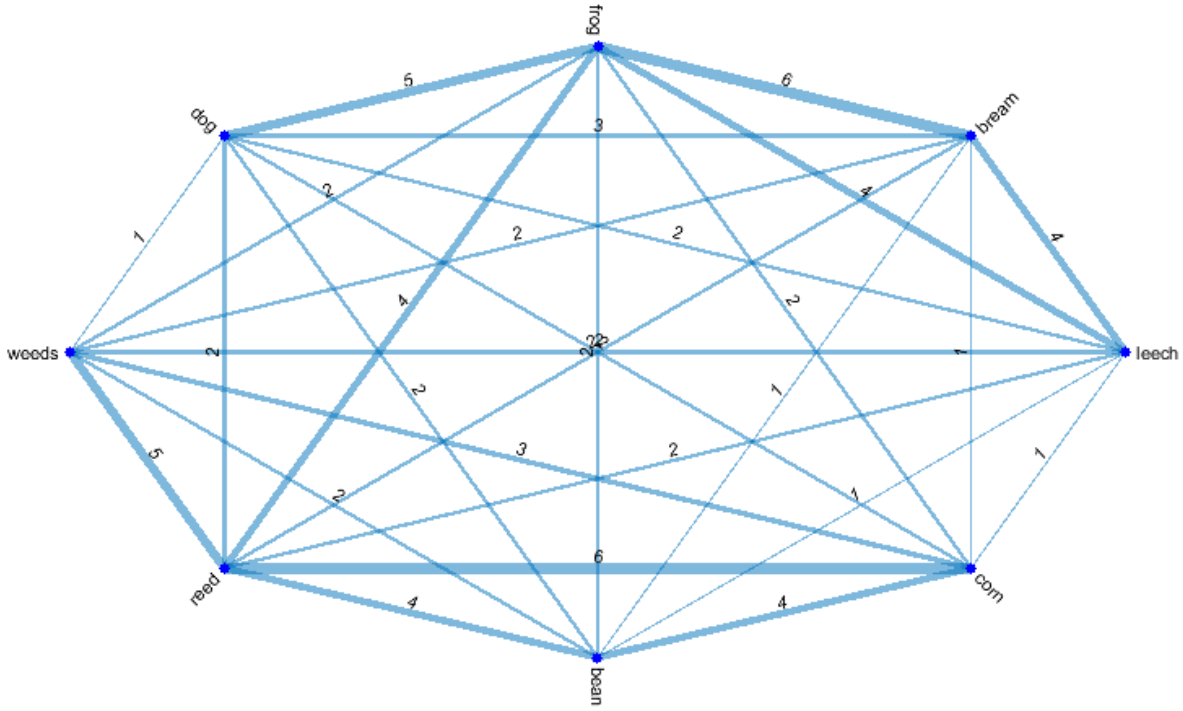

Figura 8.7 – Rede de objetos obtida a partir das Tabelas 8.3 e 8.4.

8.3 RNA-Seq

A identificação de transcritos e quantificação da expressão gênica são duas áreas importantes da biologia molecular. Um transcriptoma é um conjunto completo dos transcritos numa célula (Figura 8.8) e sua quantidade , para um estágio específico do desenvolvimento ou para uma condição fisiológica específica. Compreender o transcriptoma é essencial para interpretar os elementos do genoma, para assim entender o desenvolvimento da célula e de doenças. Os objetivos da transcriptômica são: catalogar todos os transcritos (incluindo mRNAs, *non-coding* RNAs e *small* RNAs), determinar a estrutura de um gene e quantificar as variações dos níveis de expressão durante o desenvolvimento e sobre diferentes condições [9].

Figura 8.8 – Parte do dogma central da Biologia (transcrição), indicando as áreas de genômica e transcriptômica (adaptado de McGrath [23])

RNA-Seq (do inglês, RNA sequencing) é uma abordagem experimental desenvolvida para analisar transcriptomas (Figura 8.9), ou seja, a caracterização da expressão gênica global das células. Esta técnica utiliza metodologias de sequenciamento de última geração para ler e quantificar a variação das concentrações de RNA mensageiros (mRNA) em função das condições experimentais de interesse, ajudando, assim, os pesquisadores no entendimento de quais genes estão sendo expressos e/ou tendo seus níveis de expressão modulados em determinadas células e/ou condições e num determinado momento [40].

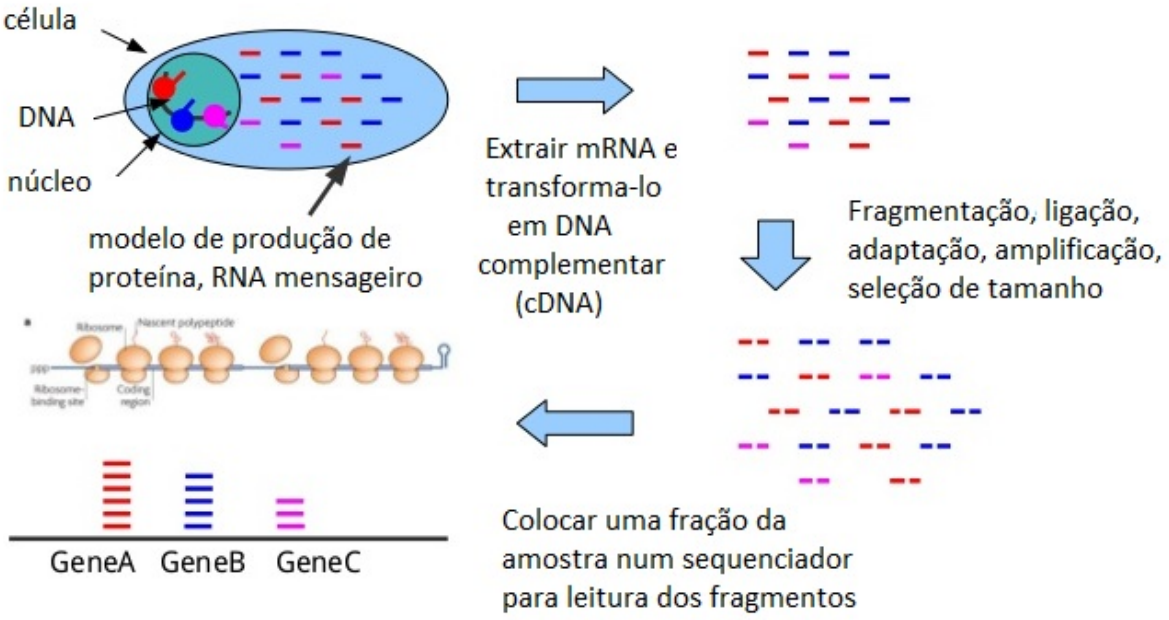

Figura 8.9 – Diagrama simplificado indicando as principais etapas do RNA Seq (adaptado de Jabob [15])

A análise computacional dos dados de RNA-Seq tipicamente produz uma matriz de expressão contendo milhares de genes nas linhas e seus respectivos valores de expressão, nas diferentes condições experimentais, descritos nas colunas [9]. Dessa forma, o método de clusterização é uma das abordagens mais comuns para detectar conjuntos de genes funcionais nessa base de dados. Contudo, essa abordagem possui algumas desvantagens. Primeiramente, a clusterização assume que um cluster de genes irá se comportar de forma similar em todas as condições. Entretanto, um processo celular pode afetar um subconjunto de genes, dados apenas um subconjunto de condições. Segundo, a clusterização irá colocar um gene em apenas um cluster. Porém, alguns genes podem estar inativos em várias condições e outros podem estar ativos em múltiplas condições [35].

Sendo assim, a coclusterização se credencia como uma abordagem mais flexível e apropriada para a análise dos dados de expressão gênica, uma vez que é capaz de encontrar subconjuntos de genes que se comportam de forma semelhante em um subconjunto específico de condições. As propriedades únicas da família de algoritmos RIn-Close serão devidamente exploradas na análise de dados de RNA-Seq.

8.4 Expressão gênica associada a tumores

Para analisar as métricas de validação intra-coclusters e as métricas para regras de associação de classes, utilizamos uma base de dados numérica que consiste nos níveis de expressão de 2308 genes, que foram obtidos de microarranjos de cDNA [17]. A base de dados possui 88 observações (linhas), contendo 5 classes: 29 tumores da família Ewing, 11 não Hodgkin linfomas, 18 neuroblastomas, 25 rhabdomyosarcomas e 5 amostras não SRBCTs (do inglês, *the small, round blue cell tumors*) [17].

Os coclusters desta base de dados numérica foram obtidos usando o algoritmo RIn-Close com $minrow = 3$, $mincol = 1$ e $\epsilon = 100$. Para utilizar o mesmo ϵ em todas as colunas, escalonamos cada coluna para o intervalo $[0, 10000]$. Com esses parâmetros, obtivemos 934618 coclusters. Como primeiro filtro, selecionamos os coclusters com um valor mínimo de confiança maior ou igual a $0,8$, ou seja, selecionamos coclusters que são formados por mais de 80% de uma única classe, o que resulta num número de coclusters igual a 32750. Assim, calculamos os valores para as métricas de validação intra-coclusters e para as métricas de regras de associação de classes.

Khan *et al.* [17] utilizaram uma rede neural artificial para classificar diagnósticos de câncer a partir de dados de expressão gênica. Os autores identificaram um subconjunto de 96 genes, a partir dos 2308 genes iniciais, que conseguiram classificar todas as amostras corretamente com relação a seus respectivos diagnósticos. Os 96 genes propostos como biomarcadores estão descritos na Tabela 8.5.

Em um outro estudo, Pal *et al.* [26] utilizaram a mesma base de dados de Khan *et al.* [17]. Utilizando uma rede neural com múltiplas camadas do tipo *perceptron* com seleção de atributos em tempo real, os autores propuseram um conjunto de 7 biomarcadores para identificar os genes causadores de câncer do tipo SRBCT (Tabela 8.5). Existem apenas 4 genes em comum nessas duas propostas: 325182, 784224, 812105 e 814260.

Tabela 8.5 – Biomarcadores propostos por Khan *et al.* [17] e Pal *et al.* [26] para identificar câncer do tipo SRBCT [38]. Os valores em negrito indicam os biomarcadores em comum às duas propostas

Ref.	Biomarcadores Propostos
[17]	21652 39093 41591 42558 43733 44563 45291 45542 52076 80109 80338 82225 122159 135688 183337 200814 204545 207274 208718 212542 233721 241412 244618 245330 246377 289645 291756 292522 293500 295985 296448 297392 298062 308163 308231 308497 323371 324494 **325182** 357031 364934 365826 377048 377461 377671 377731 383188 395708 416959 417226 461425 486110 486787 504791 563673 609663 629896 714453 745343 755599 755750 756556 767183 768370 769959 770394 782503 **784224** 784257 784593 788107 796258 809901 809910 811000 **812105** 812965 813266 **814260** 814526 824602 839736 840942 841620 841641 854899 859359 866702 868304 878280 897788 1409509 1416782 1435862 1469292 1473131
[26]	143306 **325182** 745019 770868 **784224 812105 814260**

Utilizando a interface desenvolvida ao longo da pesquisa, filtramos os 32750 coclusters utilizando como métricas:

- $Confiança = 1$;
- $E_{AColV} < 900$;
- $E_{MSR} < 800$;
- $E_{SMSR} < 0,21$;
- $E_{ACV} > 0,98$;
- $E_{ASR} > 0,8$ e
- $\overline{\Gamma} > 0,74$

Ao utilizarmos a intersecção dessas métricas, obtivemos 150 coclusters, que formam a rede de objetos da Figura 8.10.

A partir dos 150 coclusters, verificamos quais genes estavam relacionados a cada classe e obtivemos 252 genes relacionados a tumores da família Ewing; 108 genes à não-Hodgkin linfomas; 141 à neuroblastomas; e 76 à rhabdomyosarcomas. Posteriormente,

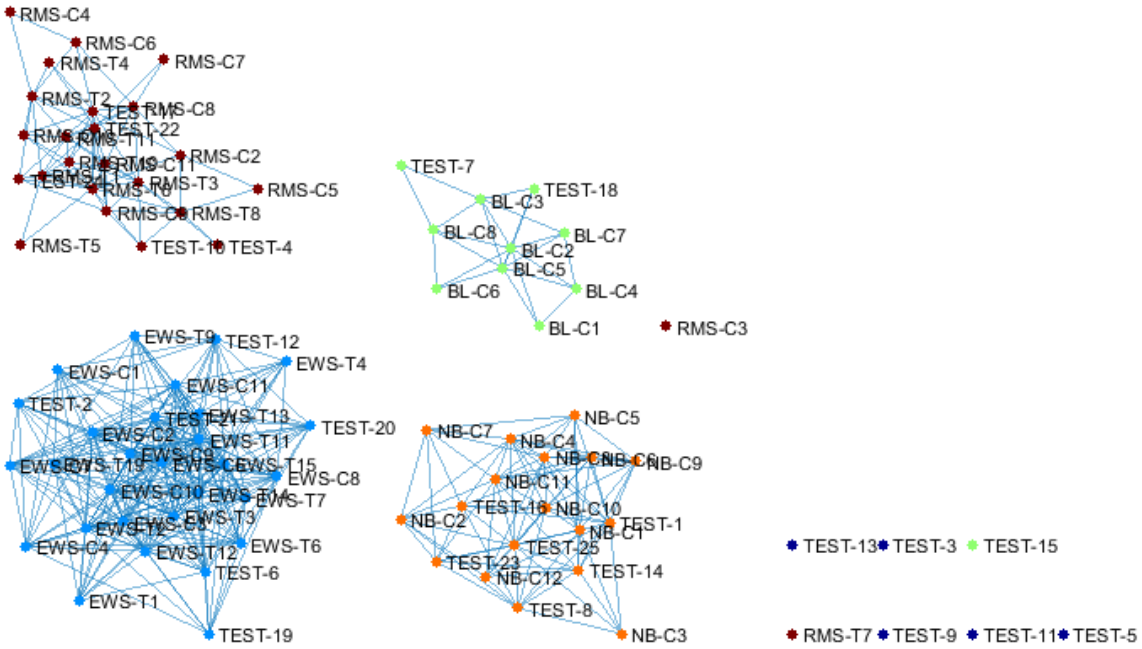

Figura 8.10 – Rede de objetos formada pelos 150 coclusters filtrados utilizando as métricas: confiança, E_{AColV}, E_{MSR}, E_{SMSR}, E_{ACV}, E_{ASR} e $\overline{\Gamma}$. Os nós em azul escuro indicam as amostras não-SRBCTs, em azul claro os tumores da família EWing, em marrom rhabdomyosarcomas, em laranja neuroblastomas e em verde não-Hodgkin linfomas.

selecionamos apenas os genes únicos a cada classe que estão descritos na Tabela 8.6, onde 16 genes são em comum com a proposta de Khan *et al.* [17] e um gene é comum à proposta de Pal *et al.* [26].

Ao utilizarmos o RIn-Close e a interface proposta, fomos capazes de selecionar genes específicos para cada classe, o que pode auxiliar num diagnóstico mais preciso para cada tipo de tumor e, consequentemente, num tratamento mais direcionado.

Tabela 8.6 – Biomarcadores únicos para os tipos de câncer família Ewing, não-Hodgkin Linfomas, neuroblastomas e rhabdomyosarcomas encontrados a partir da seleção de 150 coclusters utilizando as métricas confiança, E_{AColV}, E_{MSR}, E_{SMSR}, E_{ACV}, E_{ASR} e $\overline{\Gamma}$. Em negrito, estão destacados os genes em comum com a proposta de Khan *et al.* [17].

Ref.	Biomarcadores Propostos
Família Ewing	22260 33525 34355 323603 795309 250883 289857 51373 842825 724932 868380 809720 **377461** 854760 882548 786680 772304 740907 295551 882510 25807 627533 1455641 1472775 244146 306358 345077 379309 432194 451098 506032 684655 815861 489506 23353 753157 754649 1323203 1493527 51328 754085 163174 823886 47559 182661 321529 22918 210405 130884 261204 810934 769921 345858 128783 548957 49860 361048 347434 347373 811842 740027 365060 234237 813751 184365 701112 491066 898237 786067 843094 897670 897982 **813266** 824041 838389 724387 726236 825433 47908 898317 489489 1374571 502518 878652 753418 712341 **788107** 739993 796253 897971 824922 840978 194161 135247 773220 811168 284479 810059 811150 119133 415899 **866702** 811145 857874 251936 740604 525799 866874 235938 854576 770588 773137 590500 624429 586854 433666 60565 127841 845363 594517 869538 342349 509800 321389 785967 767277 **43733** 767236 179500 343072 129506 753215 713922 28475 813254 241365 755506 773246 783696 773332 753620 79712 85259 841370 740554 151261 66552 283897 125799 137971 296155 134495 197054 **292522** 292424 193182 753381 796198 130610 240208 180902 380620 366238 341588 323577 810133
não-Hodgkin Linfomas	773301 452374 742115 361688 25517 781506 324885 267725 379708 454333 **207274** 1486260 1344137 47542 810600 811108 **45542** **245330** 754538 156045 782800 231355 78869 815235 789014 755474 712460 773217 66711 309591 884718 278483 796278 741885 770901 770704 786673 810331 545189 509484 840384 950680 247818 129293 234150 232789 325155 771172 811028
Neuroblastomas	37904 44537 809507 431296 586803 878406 46367 154472 197657 1422723 725672 825740 154651 1323432 1412412 771220 144816 70002 204335 254321 726086 897567 310105 758662 786084 713886 79022 429182 530814 233071 490556 432564 306575 510002 769571 588559 757248 151449 460403 **364934** 362624 770794 756452 **80338** 843098 812955 756666 171936 35185 35828 245853 109708 826077 898123 700527 38465 357970 470348 244227 130892 141192 143661 309864 136814 782335 33616 795277
Rhabdomyosarcomas	214816 364510 855910 264556 68950 **769959** 1455976 271748 1472753 824552 511850 127928 773599 789204 843352 814409 1471841 826173 563621 257422 346696 365973 471598 741958 743804 302632 768260 469412 222181 71672 111981 144924 139705 **246377**

8.5 Expressão gênica de gramíneas do gênero *Brachiaria*

Gramíneas do gênero *Brachiaria* são muito utilizadas no Brasil como forrageiras pelos pecuaristas, e é um dos gêneros mais importantes para a pecuária mundial, principalmente nas regiões equatoriais e subtropicais. A área cultivada deste gênero no Brasil em 2005/2006 era quatro vezes a área de soja plantada no território nacional, correspondendo a mais de 90 milhões de hectares [16].

No Brasil, este gênero de gramínea sofre com constantes ataques de insetos conhecidos como cigarrinhas-das-pastagens. Atualmente, diversos estudos, principalmente na Embrapa, têm como foco a procura de espécies e híbridos do gênero *Brachiaria* que possuam resistência natural a este inseto. Devido a este fato, o Laboratório de Genômica e Expressão (LGE) do Instituto de Biologia da Universida Estadual de Campinas em parceria com a Embrapa, cultivaram duas espécies de gramíneas do gênero *Brachiaria*, uma conhecidamente resistente à cigarrinhas-das-pastagens e outra susceptível a estes insetos. O objetivo desta parceria é identificar os possíveis transcritos que viabilizam a resistência às cigarrinhas.

O LGE, então, utilizou 11 plantas da espécie resistente e 11 plantas da espécie susceptível para realizar o experimento descrito na Tabela 8.7. A partir da análise do RNA de cada planta, foi possível sequenciá-lo criando assim uma matriz de expressão gênica baseada em FPKM (do inglês *fragments per kilobase of exon model per million mapped reads*).

A base de dados utilizada neste estudo de caso contêm, então, 22 experimentos e 224627 transcritos. Devido ao elevado número de transcritos, inicialmente removemos os transcritos que possuem uma variação menor que 1, ou seja, a diferença entre seu valor máximo e mínimo é menor que 1. Assim, reduzimos o número de transcritos para 98193. Para analisar essa base de dados, utilizamos o algoritmo RIn-Close_CVC com $minRow = 2$, $minCol = 1$, $eps = \frac{max_col - min_col}{100}$, onde max_col é o valor máximo de cada coluna (transcrito) e min_col é o valor mínimo de cada coluna. Com esses parâmetros, obtivemos 344 coclusters. Para filtrá-los, utilizamos como métricas na interface: $confiança = 1$ e $número de linhas > 4$, obtendo assim 21 coclusters, que estão descritos na Tabela 8.8.

As Figuras 8.13 a 8.19 mostram *heatmaps* dos coclusters indicados na Tabela 8.8. As linhas desses *heatmaps* indicam todos os experimentos, as colunas indicam os transcritos (colunas) de cada cocluster, e o título indica as linhas de cada cocluster. Para facilitar a visualização, as colunas foram normalizadas para ter valor médio nulo e desvio padrão unitário.

Os *heatmaps* mostram a variação dos transcritos para todos os experimentos,

Tabela 8.7 – Informações sobre cada experimento realizado pelo LGE para analisar duas espécies de gramíneas do gênero *Brachiaria* na presença ou não de cigarrinhas-das-pastagens.

Experimento	**Tipo de espécie**	**Presença de cigarrinhas**	**Classe**
BZ1	Susceptível	Sim	Susceptível
BZ2	Susceptível	Sim	Susceptível
BZ3	Susceptível	Sim	Susceptível
BZ5	Susceptível	Sim	Susceptível
BZ6	Susceptível	Sim	Susceptível
BZ7	Susceptível	Sim	Susceptível
BZ8	Susceptível	Sim	Susceptível
BZ9	Susceptível	Sim	Susceptível
BZ10	Susceptível	Não	Controle
BZ11	Susceptível	Não	Controle
BZ12	Susceptível	Não	Controle
BZ13	Resistente	Sim	Resistente
BZ14	Resistente	Sim	Resistente
BZ15	Resistente	Sim	Resistente
BZ16	Resistente	Sim	Resistente
BZ17	Resistente	Sim	Resistente
BZ18	Resistente	Sim	Resistente
BZ19	Resistente	Sim	Resistente
BZ21	Resistente	Sim	Resistente
BZ22	Resistente	Não	Controle
BZ23	Resistente	Não	Controle
BZ24	Resistente	Não	Controle

com o intuito de verificar se um transcrito está:

- ***Up-regulated*:** regulado para cima, ou seja, se a expressão dos transcritos nas espécies resistentes é maior que nas espécies susceptíveis;
- ***Down-regulated*:** regulado para baixo, ou seja, se a expressão dos transcritos nas espécies resistentes é menor que nas espécies susceptíveis;
- ***Not-regulated*:** se ele é constante em todos os experimentos.

A Figura 8.12 indica a rede de objetos formada pelos coclusters indicados na Tabela 8.8. Percebemos que o nó "BZ1" não se encontra conectado aos outros nós da sua classe, pois ele não está presente em nenhum dos coclusters selecionados. Ao verificar a variação de todos os transcritos presentes nos coclusters da Tabela 8.8 nas Figuras 8.20 (valores de FPKM originais) e 8.21 (valores normalizados), percebemos que o experimento "BZ1" possui transcritos com expressão contrária aos demais experimentos que compõem a sua classe, o que o torna um *outlier*. Isso pode ser comprovado na Figura 8.11 que indica

Tabela 8.8 – Coclusters extraídos usando o algortimo RIn-Close_CVC com $minRow = 2$ e $minCol = 1$ e as métricas $confiança = 1$ e $númerodelinhas > 4$

Cocluster	Linhas	Colunas
1	14; 16; 17; 18; 19	15795; 46581; 87440; 95134
2	2; 3; 4; 5; 8	19099; 23905; 28069; 36778; 57282; 62151; 65760; 68037; 76700; 96709
3	2; 3; 6; 7; 8	19099; 29073; 36778; 54901; 57282; 65760; 72982; 76700
4	3; 5; 6; 7; 8	19099; 29506; 36778; 54901; 57282; 65760; 76700
5	4; 5; 6; 7; 8	19099; 29506; 36778; 57282; 65760; 72982; 76700
6	2; 3; 5; 6; 7; 8	19099; 36778; 54901; 57282; 65760; 76700
7	2; 3; 5; 6; 8	19099; 36778; 54901; 55817; 57282; 65760; 76700
8	2; 3; 5; 7; 8	19099; 36778; 54901; 57282; 62185; 65760; 76700; 77511
9	2; 3; 5; 6; 7	19099; 36778; 54901; 57282; 65760; 76434; 76700
10	2; 3; 4; 7; 8	19099; 36778; 57282; 65760; 67521; 72982; 76700; 87931
11	12; 14; 15; 16; 17	19099; 46581; 87440; 95134
12	13; 14; 15; 16; 17; 18	22848; 46581; 87440; 95134
13	13; 14; 15; 17; 18	22848; 46581; 47685; 87440; 95134
14	13; 15; 16; 17; 18	22848; 46581; 49348; 87440; 95134
15	13; 14; 15; 16; 17	22848; 46581; 78484; 87440; 95134
16	12; 13; 15; 17; 18	39778; 46581; 62909; 87440; 95134
17	13; 15; 16; 18; 19	41164; 46581; 49348; 87440; 95134
18	12; 14; 15; 17; 18	46581; 47685; 87440; 95134
19	13; 15; 16; 17; 18; 19	46581; 49348; 87440; 95134
20	12; 13; 14; 16; 18	46581; 65680; 87440; 95134
21	13; 14; 16; 18; 19	46581; 76360; 87440; 95134

a análise de componentes principais (PCA, do inglês *Principal Component Analysis*) onde o experimento "BZ1" é claramente um *outlier*.

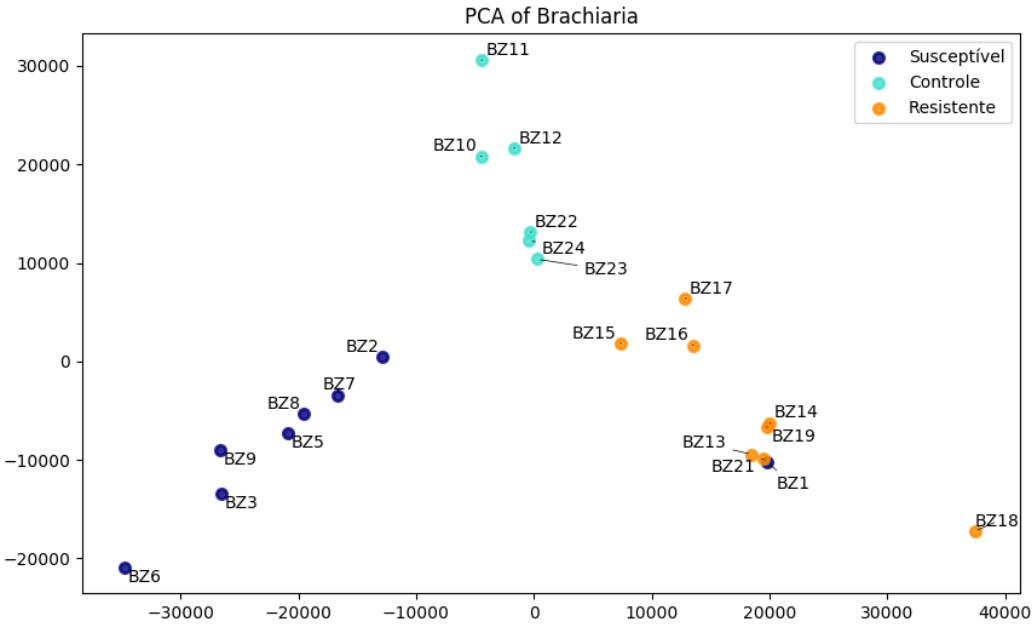

Figura 8.11 – Análise de componentes principais da base de dados de expressão gênica de gramíneas da espécie *Brachiaria*.

Analisando a variação de expressão gênica dos transcritos na Figura 8.21, percebemos que os transcritos:

- "c128942_g1", "c132433_g3", "c135526_g1", "c132089_g5", "c130608_g2", "c116087_g1", "c110615_g1", "c132325_g3", "c119078_g1" e "c129745_g3" estão regulados para cima quando comparamos as gramíneas do tipo susceptível e resistente, indicando que esses transcritos estão ativos nas *Brachiarias* resistentes;
- "c126114_g2", "c137934_g3", "c137279_g1", "110679_g1", "c130475_g1", "c113880_g1" e "c118333_g1" estão regulados para baixo quando comparamos as gramíneas do tipo susceptível e resistente, indicando que esses transcritos estão pouco expressos nas *Brachiarias* resistentes;
- "c113788_g1", "c119048_g1", "c36796_g1" e "c130608_g1" estão regulados para cima quando comparamos as classes susceptível e resistente (excluindo os controles), indicando que esses transcritos estão ativos nas *Brachiarias* resistentes e possivelmente foram ativados devido à presença das cigarrinhas;
- "c141202_g1" está regulado para baixo quando comparamos as classes susceptível e resistente (excluindo os controles), indicando que esse transcrito foi muito expresso nas *Brachiarias* susceptíveis, e possivelmente foi ativado devido à presença das cigarrinhas;

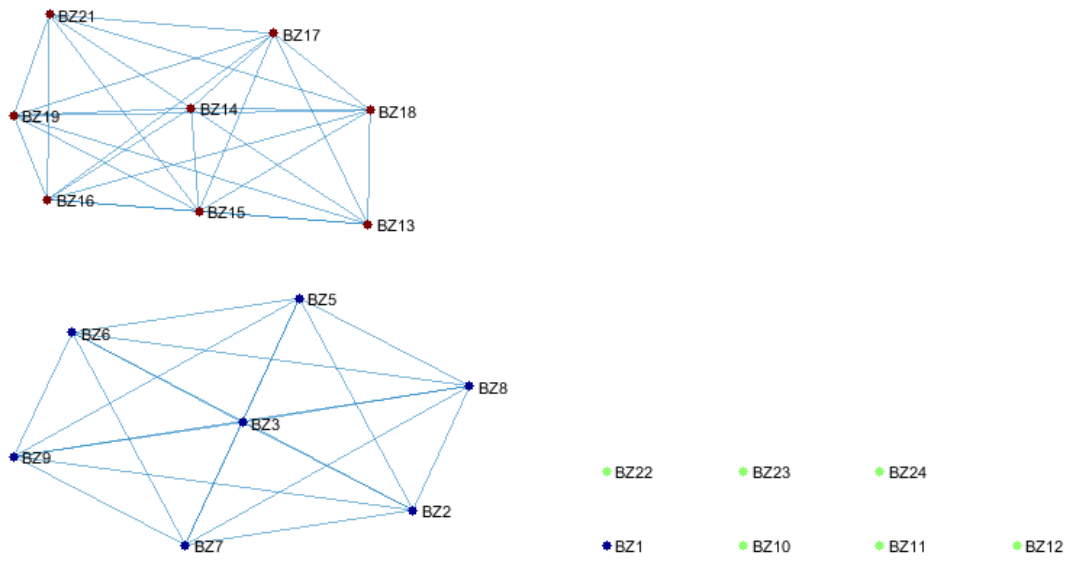

Figura 8.12 – Rede de objetos formada pelos 21 coclusters filtrados utilizando as métricas: confiança e número de linhas. Os nós em azul escuro indicam os experimentos da classe "Susceptível", em marrom a classe "Resistente" e em verde a classe "Controle".

Portanto, uma vez que são inéditas as análises desta base de dados empregando coclusterização, entende-se que se chegou a resultados promissores e que suscitam investigações subsequentes centradas nesses transcritos, os quais são candidatos a conferir resistência às cigarrinhas- das-pastagens nas gramíneas do gênero Brachiaria.

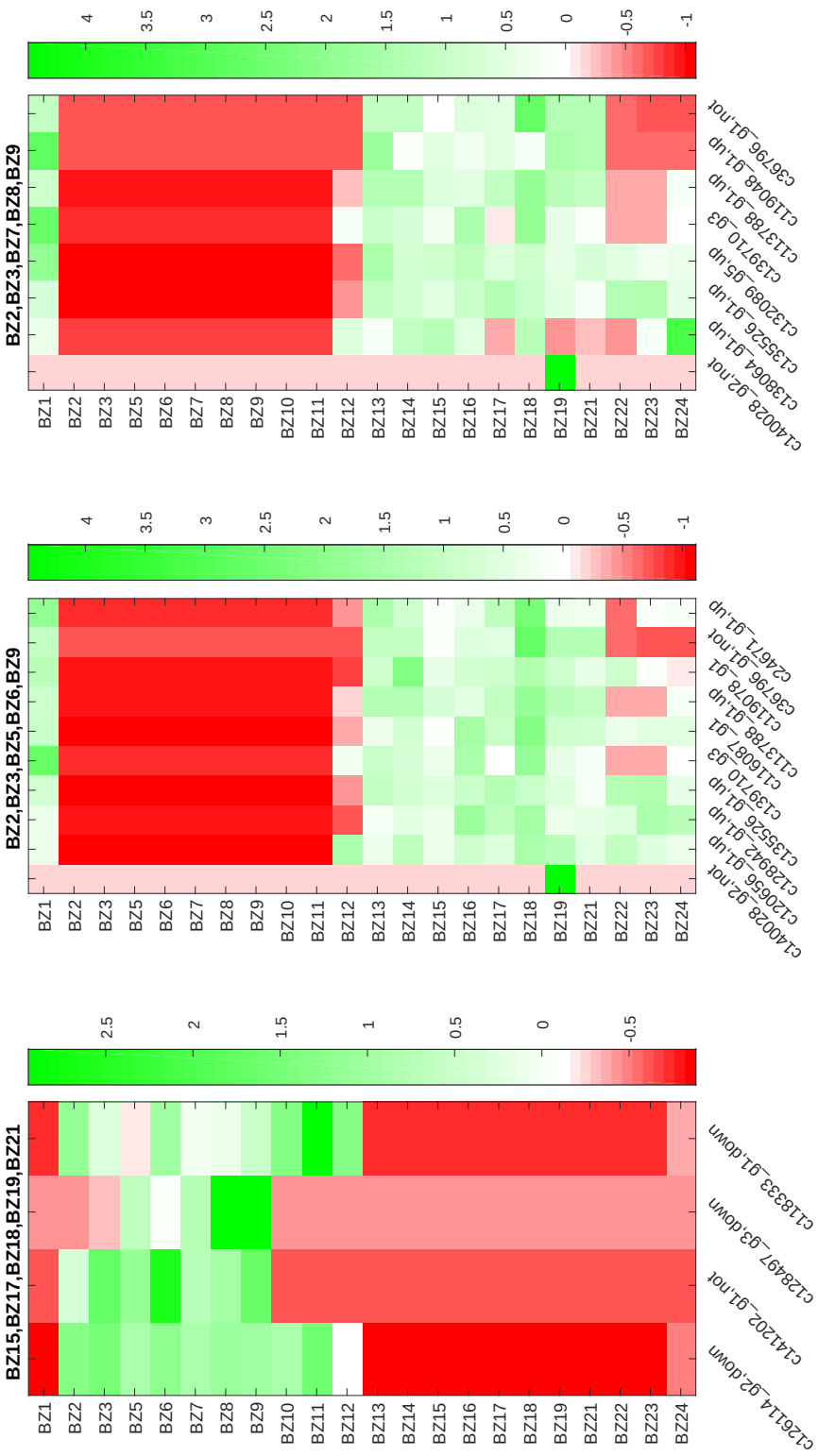

Figura 8.13 – *Heatmaps* dos coclusters 1, 2 e 3 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário.

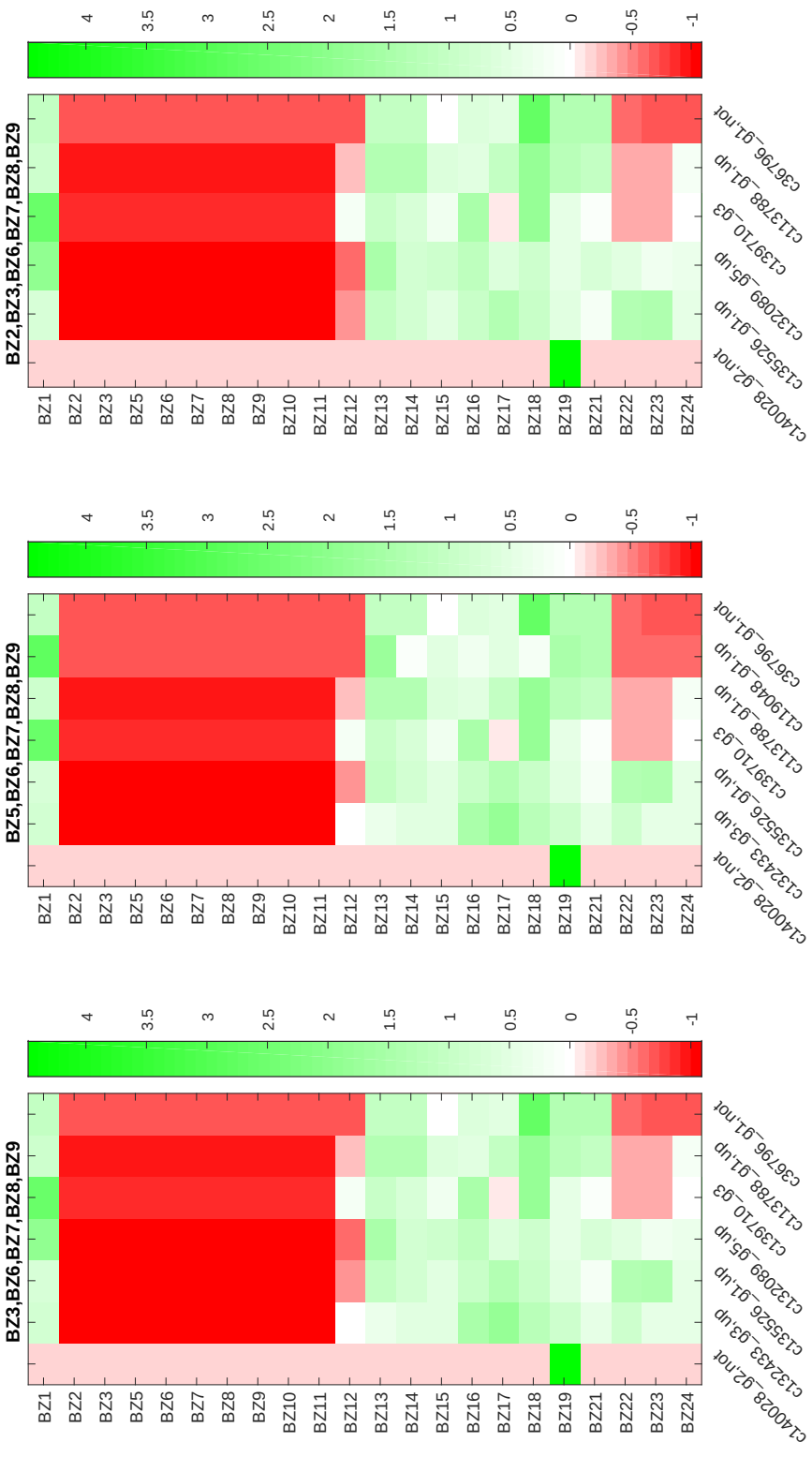

Figura 8.14 – *Heatmaps* dos coclusters 4, 5 e 6 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário.

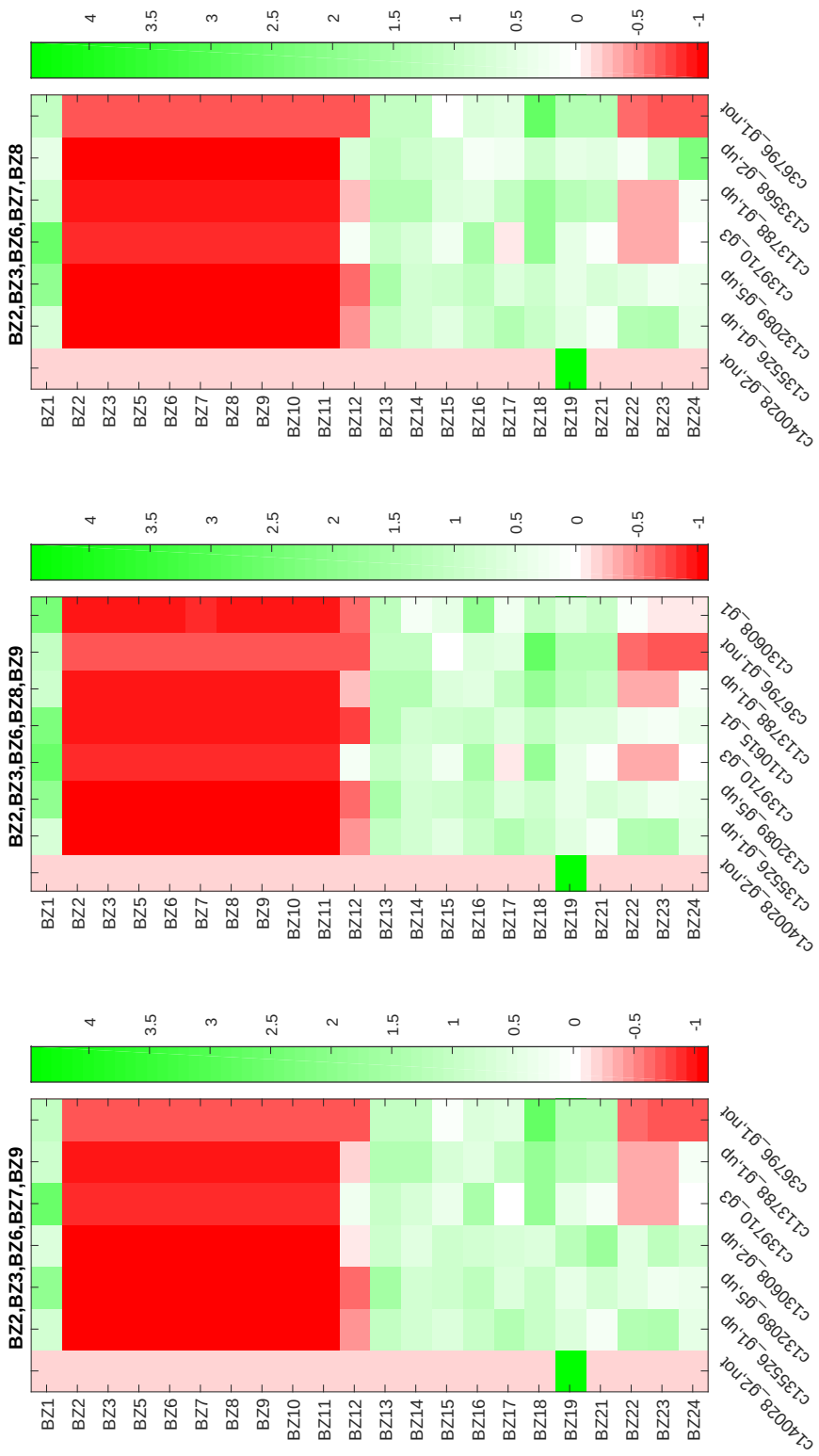

Figura 8.15 – *Heatmaps* dos coclusters 7, 8 e 9 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário.

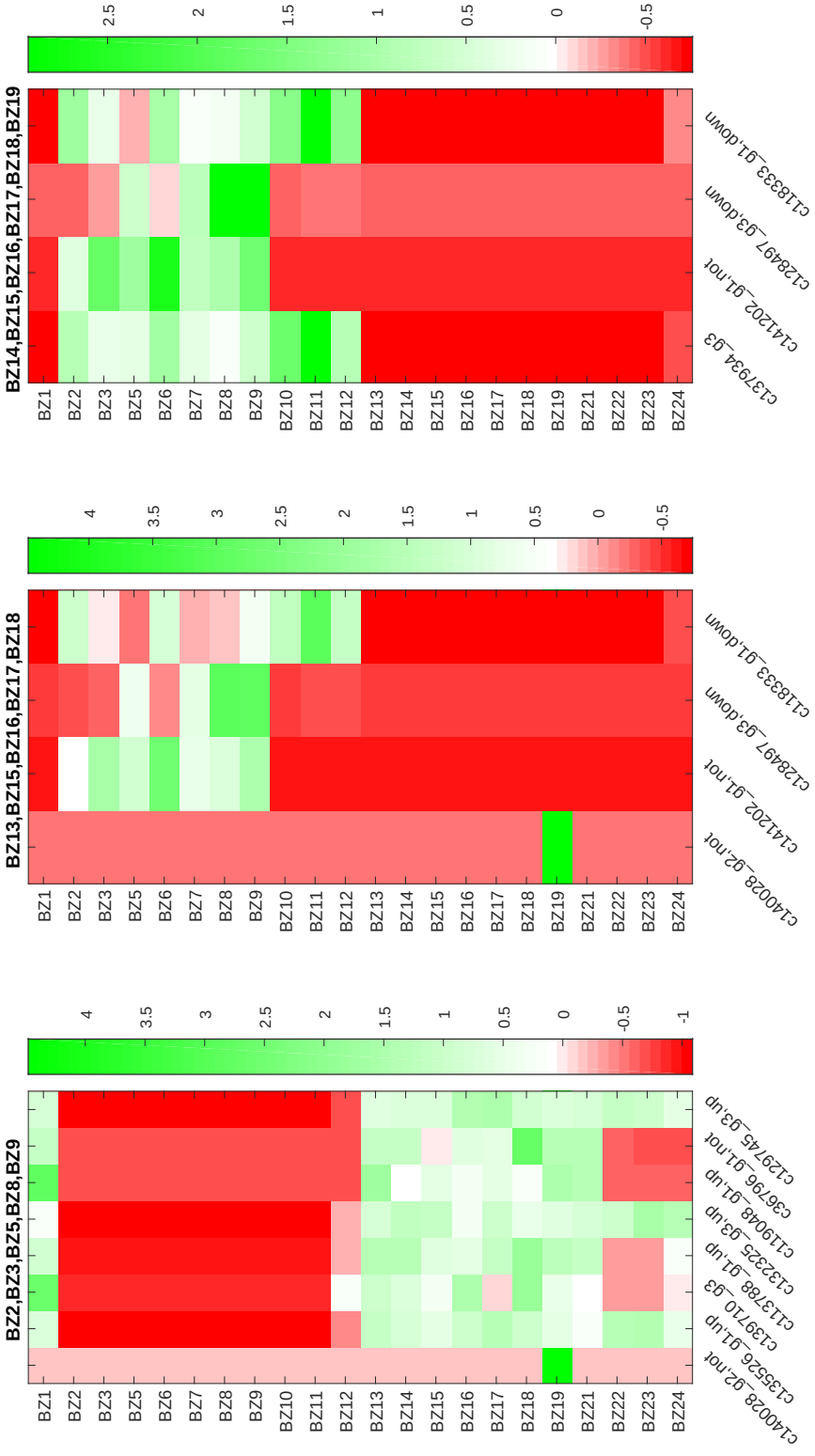

Figura 8.16 – *Heatmaps* dos coclusters 10, 11 e 12 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário.

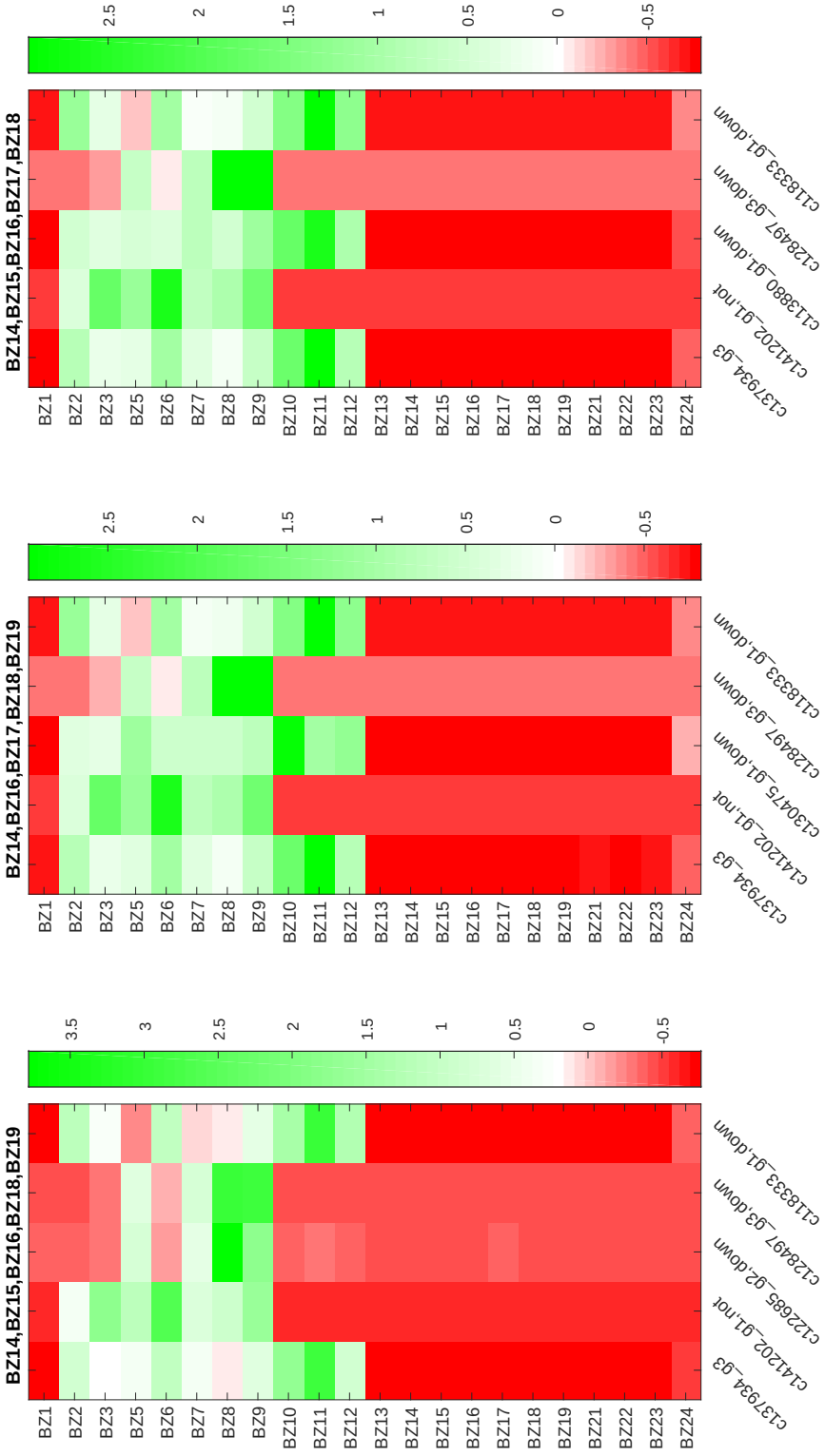

Figura 8.17 – *Heatmaps* dos coclusters 13, 14 e 15 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário.

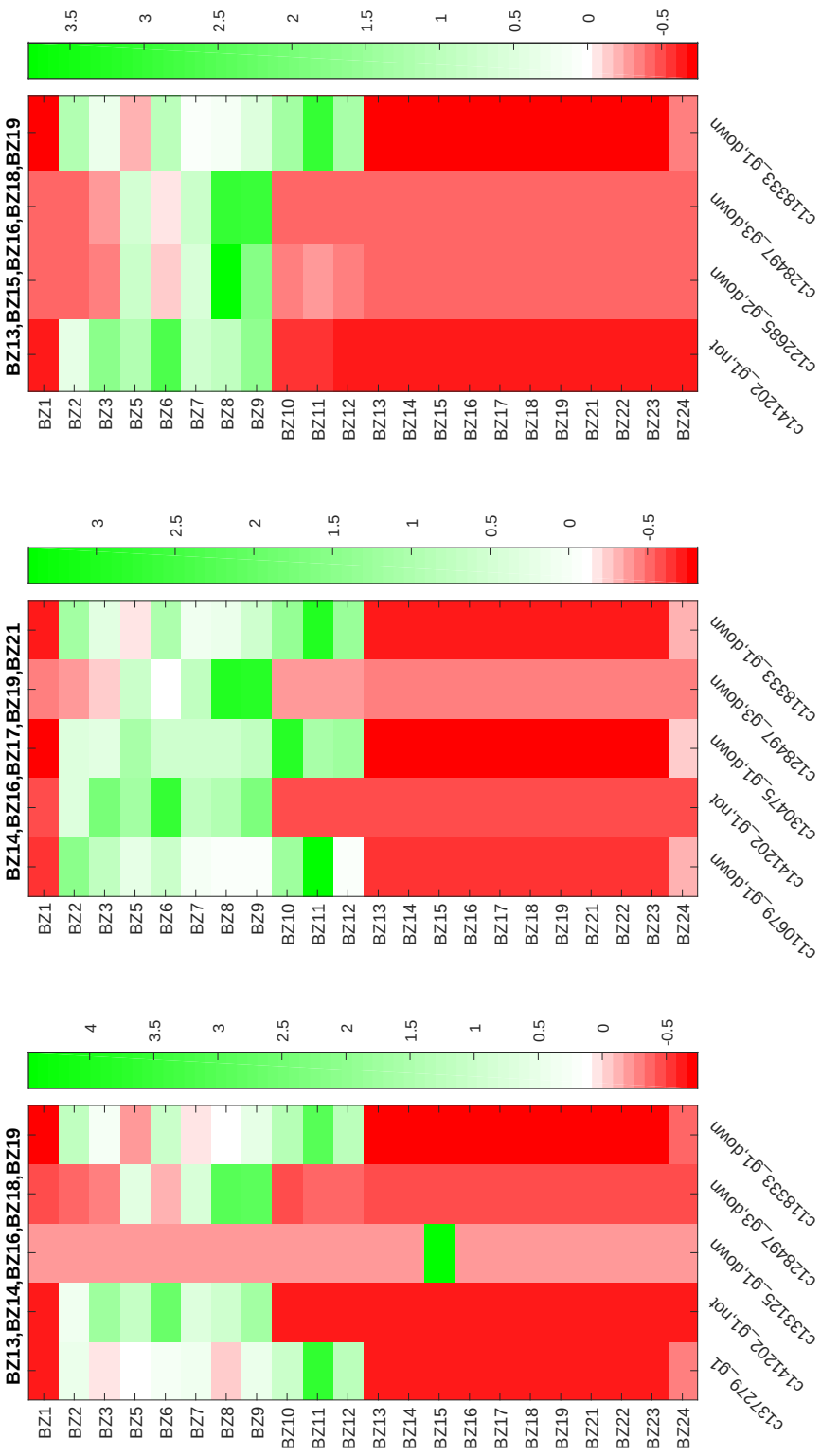

Figura 8.18 – *Heatmaps* dos coclusters 16, 17 e 18 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário.

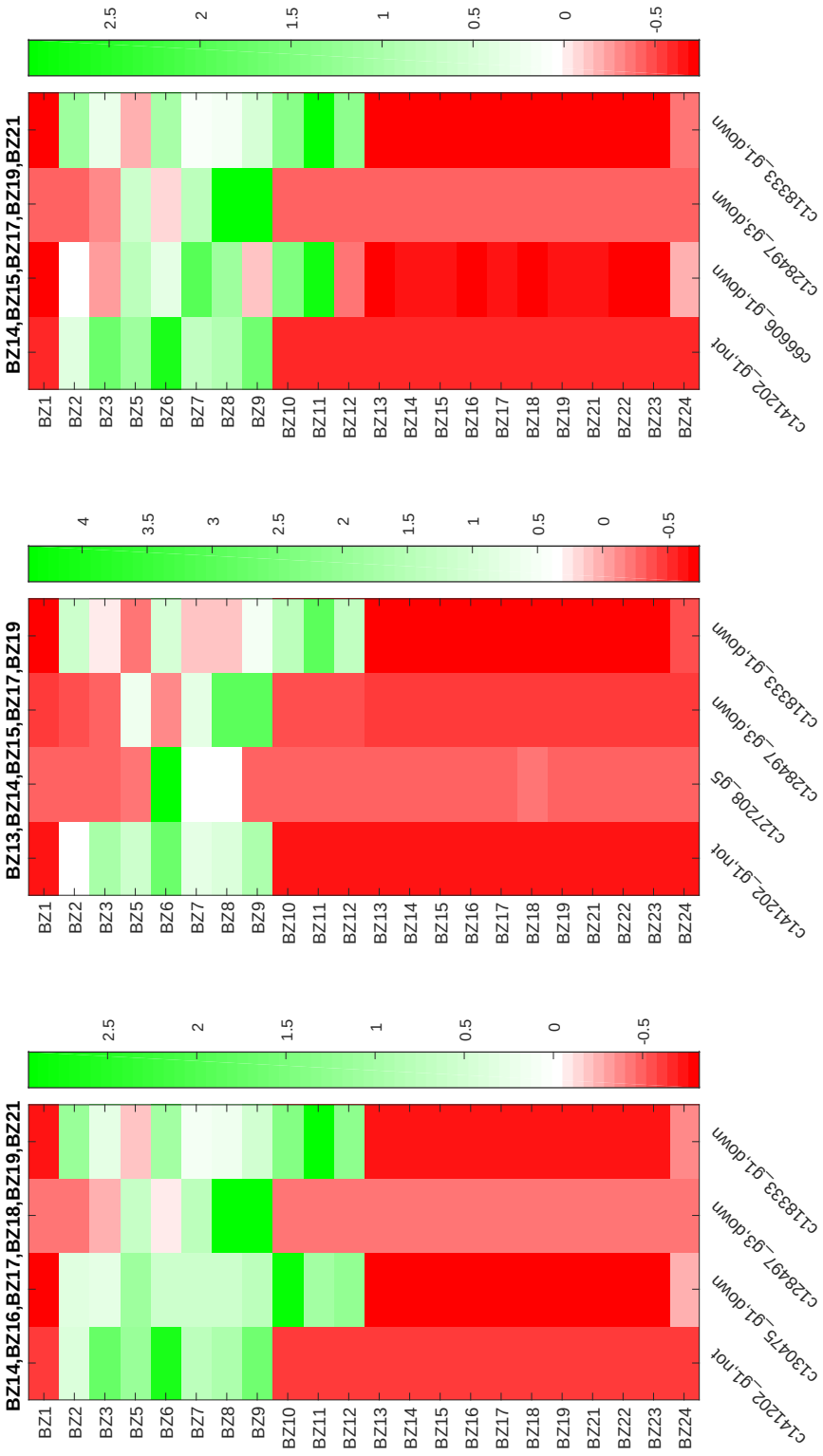

Figura 8.19 – *Heatmaps* dos coclusters 19, 20 e 21 indicados na Tabela 8.8. O título de cada *heatmap* indica as linhas do cocluster. No *heatmap* indicamos as colunas do cocluster (transcritos) e todas os experimentos da base de dados. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário.

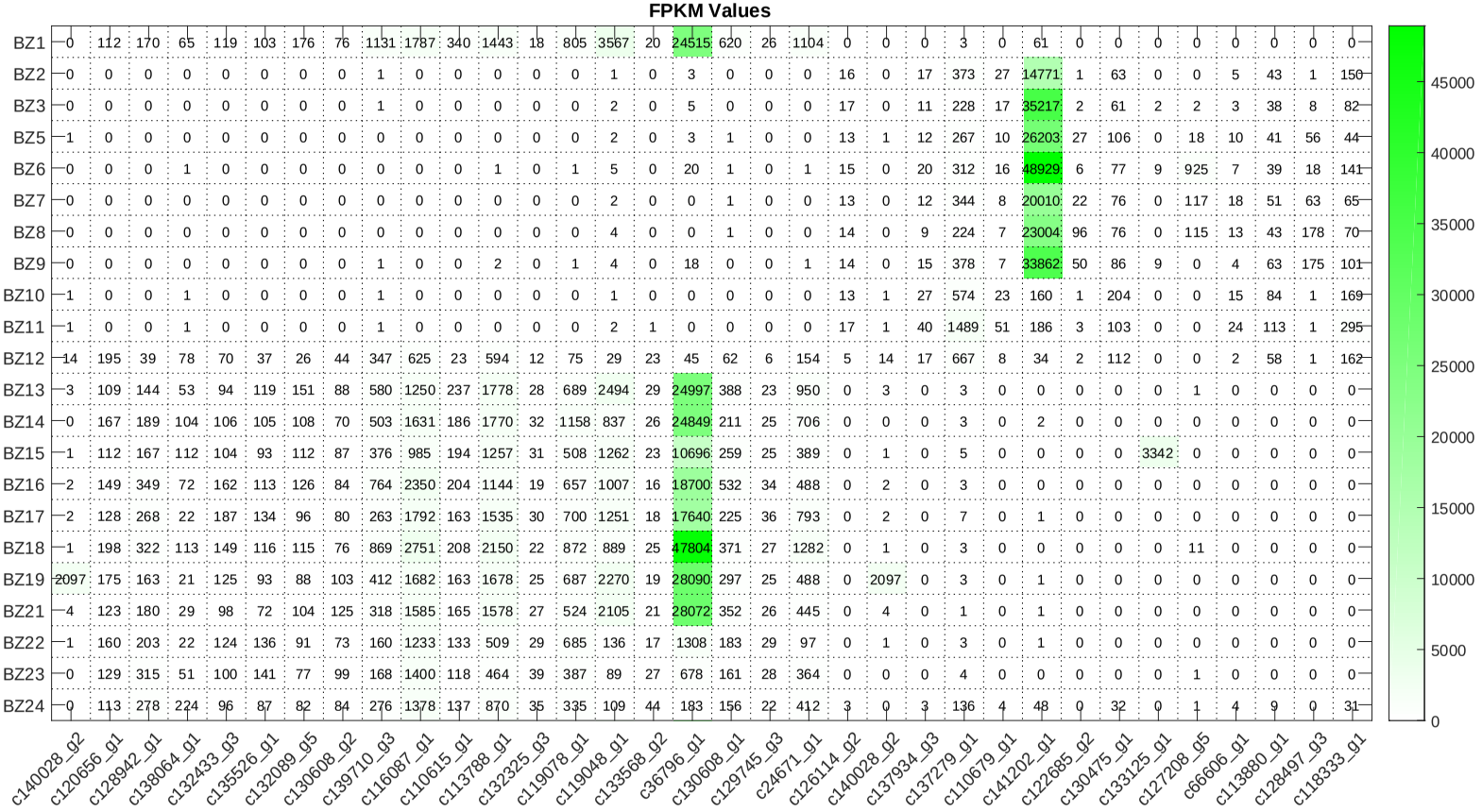

Figura 8.20 – *Heatmaps* de todos os experimentos da Tabela 8.7 e todos os transcritos indicados na Tabela 8.8 com os valores originais de FPKM.

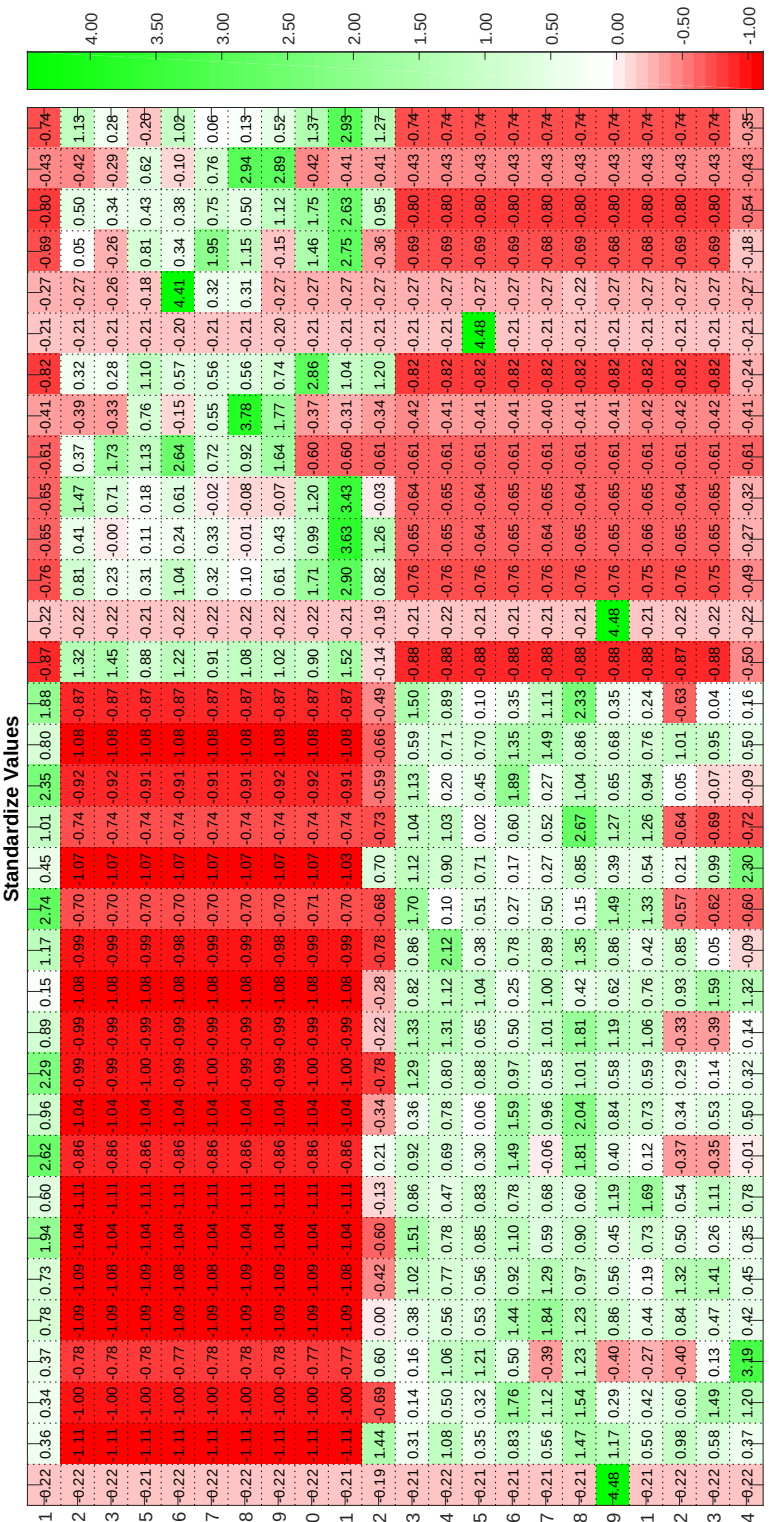

Figura 8.21 – *Heatmaps* de todos os experimentos da Tabela 8.7 e todos os transcritos indicados na Tabela 8.8. Para facilitar a visualização, os transcritos (colunas) foram normalizados para ter média nula e desvio padrão unitário.

9 Conclusão

Esta pesquisa envolveu o estudo de coclusterização, análise de conceitos formais, regras de associação de classes, o desenvolvimento de uma interface para visualização e a análise de resultados de coclusterização.

A interface proposta aqui é fruto da junção de diversas vantagens que interfaces descritas na literatura possuem, assim como a inserção de métodos que facilitam e auxiliam na análise dos resultados de coclusterização. Nesta interface, implementamos os seguintes métodos de visualização: coordenadas paralelas, diagrama *line*, árvore de conceitos e rede de objetos. Trata-se, portanto, da única interface disponível na literatura a unir métodos tradicionais de visualização de coclusters com métodos de análise de conceitos formais. Inserimos também na interface uma gama maior de métricas de validação intra-cocluster, permitindo assim que o usuário tenha liberdade para decidir que métrica ele usará, dependendo dos dados que estiver analisando. A possibilidade de analisar bases de dados com objetos rotulados utilizando regras de associação de classes também torna única essa interface, quando comparada a interfaces tradicionais de coclusterização.

O uso da interface proposta aqui permite visualizar, usando diferentes métodos, resultados de coclusters advindos de bases de dados binárias (utilizando o InClose2, por exemplo) e numéricas (utilizando o RIn-Close, por exemplo). Ao visualizarmos os métodos de diagrama *line* e árvore de conceitos, percebemos como os coclusters interagem entre si e como coclusters menores geram coclusters maiores. O método de coordenadas paralelas nos permite visualizar as variações dentro de um cocluster, enquanto que o método de rede de objetos permite visualizar como os objetos presentes em diferentes coclusters estão interagindo entre si.

A interface criada nos permite também analisar e selecionar coclusters com base em métricas de validação intra-coclusters (baseadas em variâncias, resíduos, correlações e testes estatísticos). Apesar de não encontrarmos na literatura nenhuma referência a qual métrica é mais interessante, implantamos a interface de maneira a permitir que o usuário possa usar diferentes métricas ao mesmo tempo para reduzir o número de coclusters, buscando assim selecionar os coclusters mais relevantes para a base de dados.

No painel de regras de associação de classes, introduzimos nessa interface a possibilidade de descobrir quais coclusters são os mais relevantes para descrever classes da base de dados. A maioria das regras geradas, normalmente, são redundantes ou insignificantes. Assim, é necessário elimina-las, para que permaneçam apenas regras relevantes.

Utilizamos quatro estudos de caso para verificar e testar as diversas funcio-

nalidades apresentadas na interface. Nos dois primeiros casos, utilizamos o InClose2 [3] para minerar coclusters de bases de dados binárias (Planetas do Sistema Solar e *Live in Water*). Nesses dois casos, verificamos como os métodos de visualização ajudam na análise dos resultados dos coclusters, mostrando a relação que cada cocluster possui. Nos dois últimos casos, utilizamos bases de dados numéricas com coclusters minerados pela família RIn-Close para verificar a utilidade e a necessidade de se utilizar métricas de validação para selecionar, filtrar e analisar coclusters.

Na base de dados relativa a expressão gênica de diversos tipos de câncer, conseguimos reduzir drasticamente o número de genes da matriz original e, mais importante, verificar, em cada tipo de câncer, quais genes eram relevantes. Na base de dados sobre resistência a cigarrinhas em gramíneas do gênero *Brachiaria*, conseguimos reduzir drasticamente o número de genes também, e encontramos 22 genes diferencialmente expressos que podem ser importantes para entender o motivo de uma espécie possuir resistência a cigarrinhas e outra não. Conseguimos também encontrar um experimento que pode ser considerado um *outlier*.

9.1 Perspectivas Futuras

Embora tenhamos proposto uma interface capaz de visualizar o resultados de coclusters utilizando métodos das áreas de clusterização (coordenadas paralelas), análise de conceitos formais (diagrama *line* e árvores de conceitos) e grafos (redes de objetos), e selecionar coclusters utilizando métricas de validação intra-cocluster e regras de associação de classes, ainda existem benefícios que podem ser inseridos nesta interface para facilitar e beneficiar o usuário.

A primeira é inserir novos métodos de validação, como por exemplo o BSig, recentemente proposto por Henriques & Madeira [12], que avalia estatisticamente a significância de coclusters. A segunda é permitir ao usuário inserir arquivos de entrada com outros formatos.

Esperamos também fixar novas parcerias com grupos de pesquisa diversos para analisar bases de dados utilizando o RIn-Close e a interface criada ao longo desta pesquisa, permitindo assim receber contribuições de diversos usuários com o intuito de melhorar a interface proposta, tanto nas estratégias de interação com o usuário como na disponibilização de novas funcionalidades.

Em sintonia com tendências verificadas em aplicativos de última geração, haverá um investimento na autonomia da interface, no sentido de que ao menos parte das decisões de análise sejam realizadas automaticamente pela própria interface, incluindo

ajuste ótimo de parâmetros livres, quando pertinente.

Referências

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. *ACM Sigmod Conference On Manegement Of Data*, pages 207–216, 1993. Citado na página 48.

[2] S. Andrews. In-close, a fast algorithm for computing formal concepts. *International Conference on Conceptual Structures*, 2009. Citado 2 vezes nas páginas 37 e 38.

[3] S. Andrews. In-close2, a high performance formal concept miner. *International Conference on Conceptual Structures*, pages 50–62, 2011. Citado 9 vezes nas páginas 14, 37, 38, 56, 72, 73, 74, 75 e 100.

[4] S. Andrews and L. Hirsch. A tool for creating and visualising formal concept trees. *International Conference on Conceptual Structures*, 2016. Citado 13 vezes nas páginas 10, 11, 12, 14, 36, 52, 56, 60, 68, 69, 74, 75 e 76.

[5] P. J. Azevedo and A. M. Jorge. *Comparing Rule Measures for Predictive Association Rules*, pages 510–517. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. Citado na página 51.

[6] D. Burdick, M. Calimlim, J. Flannick, J. Gehrke, and T. Yiu. Mafia: A maximal frequent itemset algorithm. *IEEE Trans on Knowl and Data Eng*, 17:1490—-1504, 2005. Citado na página 58.

[7] K. O. Cheng, N. F. Law, W. C. Siu, and T. H. Lau. Bivisu: software tool for bicluster detection and visualization. *Bioinformatics Applications Note*, 23:2342–2344, 2007. Citado 6 vezes nas páginas 11, 52, 53, 54, 60 e 62.

[8] Y. Cheng and G. M. Church. Biclustering of expression data. *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103, 2000. Citado na página 19.

[9] A. Conesa, P. Madrigal, S. Tarazona, D. Gomez-Cabrero, A. Cervera, A. McPherson, M. W. Szcześniak, D. J. Gaffney, L. L. Elo, X. Zhang, and A. Mortazavi. A survey of best practices for rna-seq data analysis. *Genome Biology*, 17(1):13–32, 2016. Citado 2 vezes nas páginas 79 e 81.

[10] B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag, first edition, 1997. Citado 2 vezes nas páginas 33 e 34.

[11] G. Govaert and M. Nadif. *Co-clustering: models, algorithms and applications*. Wiley-ISTE, 2013. Citado na página 19.

[12] R. Henriques and S. C. Madeira. Bsig: evaluating the statistical significance of biclustering solutions. *Data Mining and Knowledge Discovery*, Jun 2017. Citado na página 100.

[13] R. Henriques and S.C. Madeira. Bicpam: Pattern-based biclustering for biomedical data analysis. *Algorithms for Molecular Biology*, 9(1):27, 2014. Citado 3 vezes nas páginas 20, 21 e 23.

[14] K i. Takahashi, I. Takigawa, and H. Mamitsuka. Sibic: A web server for generating gene set networks based on biclusters obtained by maximal frequent itemset mining. *PLoS ONE*, 8:e82890, 2013. Citado 9 vezes nas páginas 11, 30, 52, 57, 58, 60, 70, 71 e 79.

[15] J. Jacob. Part 1 of rna-seq for de analysis: Defining the goal. `http://www.slideshare.net/jakonix/part-1-of-rnaseq-for-de-defining-the-goal`, 2014. Online; accessed 01-August-2017. Citado 2 vezes nas páginas 12 e 80.

[16] C. T. Karia, J. B. Duarte, and A. C. G. Araújo. Desenvolvimento de cultivares do gênero Brachiaria (trin.) griseb. no Brasil. Technical report, Embrapa - Empresa Brasileira de Pesquisa Agropecuária, 08 2006. Citado na página 85.

[17] J. Khan, J.S. Wei, M. Ringner, L.H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C.R. Antonescu, C. Peterson, and P.S. Meltzer. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7:673–679, 2001. Citado 5 vezes nas páginas 14, 81, 82, 83 e 84.

[18] E. Kropat. Formal concept analysis. Achievements and Applications of Contemporary Informatics, Mathematics and Physics (AACIMP), 2011. Citado 7 vezes nas páginas 10, 14, 33, 35, 36, 41 e 72.

[19] S. O. Kuznetsov. Learning of simple conceptual graphs from positive and negative examples. *Principles of Data Mining and Knowledge Discovery, Lecture Notes in Computer Science*, 1704:384–391, 1999. Citado 3 vezes nas páginas 34, 37 e 38.

[20] Y. Lee, J. Lee, and C. Jun. Validation measures of bicluster solutions. In *Industrial Engineering and Management Systems*, volume 8, pages 101–108, 2009. Citado na página 27.

[21] S. C. Madeira. (Clustering and) Biclustering gene expression data. `http://cw.fel.cvut.cz/wiki/_media/courses/a6m33bin/biclustering.pdf`, 2011. Online; accessed 19-July-2016. Citado 4 vezes nas páginas 10, 23, 24 e 27.

[22] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pages 24–45, 2004. Citado 3 vezes nas páginas 19, 24 e 26.

[23] S. McGrath. The power and pitfalls of omics part 2: epigenomics, transcriptomics and me/cfs. `http://phoenixrising.me/archives/27961`, 2016. Online; accessed 01-August-2017. Citado 2 vezes nas páginas 12 e 80.

[24] A. Mukhopadhyay, U. Maulik, and S. Bandyopadhyay. A novel biclustering approach to association rule mining for predicting hiv-1–human protein interactions. *PLoS ONE*, 7:e32289, 2012. Citado na página 21.

[25] K o. Cheng, N f. Law, W c. Siu, and A. W c. Liew. Identification of coherent patterns in gene expression data using an efficient biclustering algorithm and parallel coordinate visualization. *BMC Bioinformatics*, 8:210, 2008. Citado na página 54.

[26] N.R. Pal, K. Aguan, A. Sharma, and S. Amari. Discovering biomarkers from gene expression data for predicting cancer subgroups using neural networks and relational fuzzy clustering. *BMC Bioinformatics*, 8:5, 2007. Citado 3 vezes nas páginas 14, 82 e 83.

[27] B. Pontes, R. Girldez, and J. S. Aguilar-Ruiz. Quality measures for gene expression biclusters. *PLoS ONE*, 10:e0115497, 2015. Citado na página 28.

[28] Uta Priss. Formal concept analysis examples. `http://www.upriss.org.uk/fca/examples.html`, 2007. Online; accessed 15-December-2016. Citado 4 vezes nas páginas 12, 14, 74 e 76.

[29] H. B. Saber and M. Elloumi. A new study on biclustering tools, biclusters validation and evaluation functions. *International Journal of Computer Science & Engineering Survey (IJCSES)*, 6, 2015. Citado 3 vezes nas páginas 27, 28 e 31.

[30] R. Santamaría, R. Théron, and L. Quintales. Methods to bicluster validation and comparison in microarray data. In Yin H., Tino P., Corchado E., Byrne W., and Yao X., editors, *Intelligent Data Engineering and Automated Learning - IDEAL 2007*, volume 4881. Springer, 2007. Citado na página 29.

[31] R. Santamaría, R. Théron, and L. Quintales. Bicoverlapper: A tool for bicluster visualization. *Bioinformatics Applications Note*, 24:1212–1213, 2008. Citado 8 vezes nas páginas 11, 52, 54, 55, 60, 70, 71 e 79.

[32] R. Santamaría, R. Théron, and L. Quintales. A visual analytics approach for understanding biclustering results from microarray data. *BMC Bioinformatics*, 9:247, 2008. Citado na página 52.

[33] R. Santamaría, R. Théron, and L. Quintales. Bicoverlapper 2.0: visual analysis for gene expression. *Bioinformatics Applications Note*, 30:1785–1786, 2014. Citado 5 vezes nas páginas 11, 52, 55, 56 e 60.

[34] R. Santamaria. *Visual Analysis of Gene Expression Data By Means of Biclustering*. PhD thesis, Department of Computer and Automation, Faculty of Sciences, University of Salamanca, Salamanca, 2009. Citado 4 vezes nas páginas 20, 29, 30 e 59.

[35] A. Serin and M. Vingron. Debi: Discovering differentially expressed biclusters using a frequent itemset approach. *Algorithms for Molecular Biology*, 6:18–30, 2011. Citado na página 81.

[36] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *IEEE Visual Languages, UMCPCSD CS-TR-3665*, pages 336–343, 1996. Citado na página 59.

[37] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *Acm Sigmod Record*, volume 25, pages 1–12. ACM, 1996. Citado na página 20.

[38] R. Veroneze. *Enumerating all maximal biclusters in numerical datasets*. PhD thesis, Faculdade de Engenharia Elétrica e Computação, Universidade Estadual de Campinas (Unicamp), Campinas, 2016. Citado 21 vezes nas páginas 10, 14, 19, 20, 21, 25, 30, 31, 38, 39, 40, 41, 42, 43, 44, 46, 47, 49, 50, 51 e 82.

[39] R. Veroneze, A. Banerjee, and F. J. Von Zuben. Enumerating all maximal biclusters in numerical datasets. *Information Sciences*, 379:288–309, 2017. Citado na página 21.

[40] Z. Wang, M. Gerstain, and M. Synder. Rna-seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10:57–64, 2009. Citado na página 80.

[41] M. J. Zaki and W. Meira Jr. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Pres, 2014. Citado 2 vezes nas páginas 50 e 51.