João Luís Baptista de Almeida

ASA_{Clu}: Selecionando Clusters Diversos e Relevantes

Sorocaba, SP

23 de Janeiro de 2017

João Luís Baptista de Almeida

ASA_{Clu}: Selecionando Clusters Diversos e Relevantes

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação (PPGCC-So) da Universidade Federal de São Carlos como parte dos requisitos exigidos para a obtenção do título de Mestre em Ciência da Computação. Linha de pesquisa: Análise de Agrupamentos.

Universidade Federal de São Carlos – UFSCar

Centro de Ciências em Gestão e Tecnologia – CCGT

Programa de Pós-Graduação em Ciência da Computação – PPGCC-So

Orientador: Prof. Dr. Katti Faceli

Coorientador: Prof. Dr. Tiemi C. Sakata

Sorocaba, SP

23 de Janeiro de 2017

Baptista de Almeida, João Luís
ASA_{Clu}: Selecionando Clusters Diversos e Relevantes / João Luís Baptista de Almeida. – 2016
48 f. : 30 cm.

Dissertação (Mestrado) – Universidade Federal de São Carlos, campus Sorocaba, Sorocaba
Orientador: Prof. Dr. Katti Faceli
Banca examinadora: Prof. Dr. Tiemi C. Sakata, Prof. Dr. Murilo Coelho Naldi, Prof. Dr. Tiago A. Almeida
Bibliografia

1. Seleção de *clusters*. 2. Agrupamento de dados. 3. Múltiplas soluções em agrupamento. I. Orientador. II. Universidade Federal de São Carlos. III. Título.

Ficha catalográfica elaborada pelo Programa de Geração Automática da Secretaria Geral de Informática (SIn). DADOS FORNECIDOS PELO(A) AUTOR(A)

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências em Gestão e Tecnologia

Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a defesa de dissertação de mestrado do candidato João Luís Baptista de Almeida, realizada em 12/12/2016:

Profa. Dra. Tiemi Christine Sakata
UFSCar

Prof. Dr. Murilo Coelho Naldi
UFV

Prof. Dr. Tiago Agostinho de Almeida
UFSCar

Certifico que a sessão de defesa foi realizada com a participação à distância do membro Prof. Dr. Murilo Coelho Naldi e, depois das arguições e deliberações realizadas, o participante à distância está de acordo com o conteúdo do parecer da comissão examinadora redigido no relatório de defesa do aluno João Luís Baptista de Almeida.

Profa. Dra. Tiemi Christine Sakata
Presidente da Comissão Examinadora
UFSCar

Ao meu filho Joaquim, à minha esposa Gracyellen e à toda minha família, que sempre estão presentes me incentivando e me apoiando.

Agradecimentos

Agradeço,

à minha família pelo apoio em todos os momentos.

à minha orientadora Katti pelo excelente trabalho e comprometimento como pesquisadora e orientadora.

à minha coorientadora Tiemi pela ajuda de extrema importância na orientação desse trabalho.

ao professor Tiago pela ajuda na escolha do orientador e pela dedicação nas aulas de aprendizado de máquina.

"Uma paixão forte por qualquer objeto assegurará o sucesso, porque o desejo pelo objetivo mostrará os meios.
(Fruto do Esforço, William Hazlitt)

Resumo

Nenhum algoritmo de agrupamento garante encontrar grupos reais em qualquer conjunto de dados. Para lidar com esse problema, muitas técnicas aplicam vários algoritmos de agrupamento a um conjunto de dados, gerando um conjunto de partições e avaliando-as para selecionar as mais apropriadas. O problema na seleção de partições é que a redundância pode ser vista dentro de partições, como o mesmo *cluster* pode aparecer em diferentes partições. Além disso, pode-se subestimar a qualidade de um *clusters*, avaliando apenas a qualidade de uma partição. Neste trabalho, é proposta uma nova estratégia de seleção chamada ASA_{Clu}, que visa selecionar um subconjunto relevante e diverso de *cluster* em vez de partições, dada uma coleção inicial.

Palavras-chaves: Seleção de *clusters*. Agrupamento de dados. Múltiplas soluções em agrupamento.

Abstract

No clustering algorithm is guaranteed to find actual groups in any dataset. To deal with this problem, many techniques apply various clustering algorithms to a dataset, generating a set of partitions and assessing them to select the most appropriated ones. The problem in selecting partitions is that redundancy can be seen inside partitions, as the same cluster can appear in different partitions. Also, one can underestimate the quality of a cluster, assessing only the quality of a partition. For these reasons, a new selection strategy named ASA_{Clu} is aimed at selecting a relevant and diverse subset of clusters instead of partitions, given an initial collection.

Key-words: Clustering. Clusters selection. Multiple solutions in clustering.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ASA_{Clu}	*Automatic cluster Selection Algorithm*
ASA	*Automatic Selection Algorithm*
MOC	*MultiObjective Clustering*
$MBCS$	*Multiplicity Based Cluster Selection*
EM	Algoritmo *Expectation Maximization*
SL	Algoritmo hierárquico *Single Link*
AL	Algoritmo hierárquico *Average Link*
CoL	Algoritmo hierárquico *Complete Link*
CeL	Algoritmo hierárquico *Centroid Link*
SNN	Algoritmo *Shared Nearest Neighbors*
KM	Algoritmo *K-Means*
JI	Índice Jaccard
JI_m	A média dos índices Jaccard
ARI	Índice Rand ajustado

Lista de símbolos

k	Número de *clusters* de uma partição qualquer
k^{max}	Número máximo de *clusters*
C_C	Multiconjunto completo de *clusters*
C_u	Conjunto subjacente de *clusters* de C_C
c^i	i-ésimo *cluster*
n_i	A multiplicidade do *cluster* c^i
C_R	Conjunto reduzido de *clusters* de C_C
C_I	Conjunto de *clusters* iniciais, produzido na inicialização do ASA_{Clu}
n_r	Parâmetro do limiar da multiplicidade de *clusters*, usado na inicialização do ASA_{Clu}
t	*Threshold*
r_t	Taxa de redução
PR	Número de *clusters* parcialmente recuperados
CR	Número de *clusters* completamente recuperados
Π_C	Multiconjunto inicial de partições
π^i	i-ésima partição
p_i	A multiplicidade da partição π^i
Π^u	Conjunto subjacente de partições de Π_C
Π_R	Conjunto reduzido de partições de Π_C
p	Parâmetro do limiar da multiplicidade de partições, usado na inicialização do ASA
c^{known}	*Cluster* real pertencente as estruturas conhecidas
C_{known}	Conjunto de *clusters* das estruturas conhecidas

Sumário

1 Introdução

1.1 Contextualização

Uma das formas mais comuns na análise exploratória de dados envolve a tarefa de sumarizar dados em forma de *clusters*, onde objetos similares pertencem a um mesmo *cluster*. Essa forma de organizar o conhecimento acompanha a evolução da humanidade. Para aprender um novo objeto ou entender um fenômeno, pessoas sempre tentam encontrar as características que descrevem um objeto ou fenômeno e utilizam formas de comparações de similaridade entre esses objetos e fenômenos (XU; WUNSCH, 2005). A análise de agrupamentos tem uma ampla aplicação, tendo seus principais propósitos em detectar anomalias ou características salientes nos dados, identificar o grau de similaridade entre formas de organismos (relação filogenética), gerar hipóteses pela observação dos padrões dos dados e organizar os dados de forma resumida em grupos (JAIN, 2008).

A análise de agrupamentos tradicionalmente aplica uma única estrutura de agrupamento em um conjunto de dados subjacente. Essa estrutura é construída a partir de um critério de agrupamento, uma definição formal do que é um *cluster* (HANDL; KNOWLES; KELL, 2005). Muitas vezes, um conjunto de dados pode apresentar mais de uma estrutura de agrupamento de interesse, e cada estrutura representa os dados em diferentes aspectos e características. Em tais cenários, um objeto pode pertencer a diferentes *clusters*, assumindo papéis diferentes (MüLLER et al., 2015). Isso fornece mais visões de diferentes perspectivas dos dados do que uma única estrutura poderia fornecer. Dessa forma, a diversidade de *clusters* pode conter conhecimento adicional (MüLLER et al., 2012). Por exemplo, numa aplicação real, artigos científicos são agrupados pela similaridade de seus conteúdos (Figura 1). Pode-se rotular alguns grupos com temas de estudo científico já conhecidos, tais como aprendizado de máquina (AM), banco de dados (BD) e mineração de dados (MD). Por outro lado, alguns artigos também fazem parte de um outro tema de estudo científico ainda menos conhecido (múltiplas soluções em agrupamento). Analisando essas estruturas, podemos extrair conhecimento de um novo tema de pesquisa. Em termos de agrupamento, haveria uma partição com os grupos (AM), (BD) e (MD), e uma outra partição alternativa criaria outros grupos com outros significados (múltiplas soluções em agrupamento).

Para lidar com o problema de múltiplas soluções de agrupamento, pode-se aplicar diversos algoritmos de agrupamento em um conjunto de dados. Cada algoritmo de agrupamento está condicionado a formar estruturas a partir de um critério de agrupamento. Um critério de agrupamento pode fornecer características peculiares para a formação de um *cluster*. Por exemplo, o algoritmo K-means produz *clusters* esféricos (Figura 2), enquanto o algoritmo Single-Link produz cluters encadeados (Figura 3).

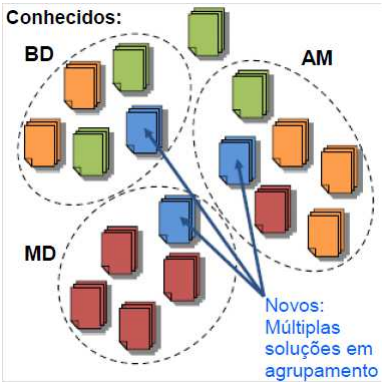

Figura 1 – Aplicação real de artigos científicos agrupados por conteúdo em mais de uma estrutura de agrupamento (MüLLER et al., 2012).

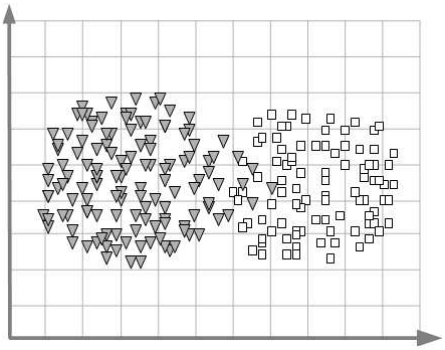

Figura 2 – *Clusters* esféricos produzidos pelo algoritmo K-means (FACELI et al., 2011).

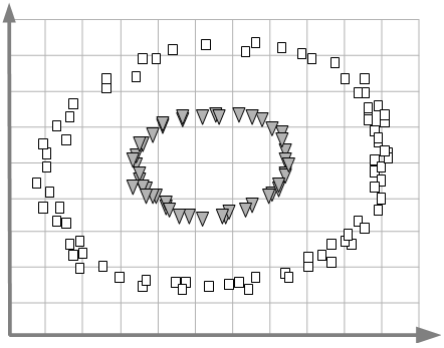

Figura 3 – *Clusters* encadeados produzidos pelo algoritmo Single-Link (FACELI et al., 2011).

Para aumentar ainda mais a diversidade de *clusters* encontrados em um conjunto de dados, pode-se variar o número de *clusters* a serem produzidos pelo algoritmo. Dessa forma, é possível obter *clusters* de diferentes tamanhos (Figura 4).

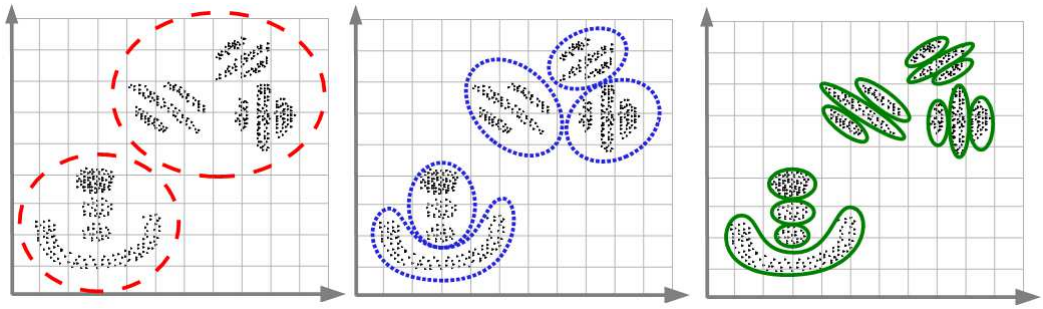

Figura 4 – *Clusters* de diferentes tamanhos (FACELI et al., 2011).

1.2 Motivação

Ao gerar *clusters* de diferentes tamanhos, formas e que se sobrepõem dentro de um conjunto de dados, um multiconjunto relativamente grande de soluções é obtido. Esse multiconjunto de soluções requer um passo adicional para selecionar as soluções mais relevantes (SAKATA et al., 2010). O algoritmo de seleção automática *Automatic Selection Algorithm* (*ASA*) foi proposto para resolver o problema da dificuldade de analisar um grande número de soluções (SAKATA et al., 2010). O *ASA* seleciona as partições mais relevantes e mantém a diversidade dentro de um multiconjunto de partições geradas pela aplicação de diversos algoritmos de agrupamento.

Como o *ASA*, muitas técnicas de agrupamento baseadas em múltiplas soluções estão focadas em encontrar partições. Porém, redundância pode ser encontrada dentro de partições, pois um *cluster* pode aparecer muitas vezes em diferentes partições. No contexto de múltiplas soluções em agrupamento, pode-se subestimar a qualidade de um *cluster* ao avaliar somente a qualidade da partição que ele está inserido (FACELI; SAKATA, 2016). A técnica *Multiplicity Based Cluster Selection* (*MBCS*) seleciona *clusters* ao invés de partições. Dado um multiconjunto inicial de *clusters* obtidos por diversas partições geradas por algoritmos de agrupamento aplicados a um conjunto de dados, o *MBCS* se baseia na multiplicidade de um *cluster* para compor o conjunto reduzido de *clusters* como solução. Um *cluster* que aparece pelo menos duas vezes no multiconjunto inicial é selecionado para compor o conjunto reduzido (FACELI; SAKATA, 2016).

No mesmo contexto de selecionar *clusters* ao invés de partições, *MOC* (*MultiObjective Clustering*) é um framework que busca recuperar *clusters* interessantes em relação a dois ou mais objetivos (critério) (JIAMTHAPTHAKSIN; EICK; VILALTA, 2009). *MOC* produz um repositório de *clusters* potencialmente interessantes de acordo com múltiplos objetivos e tem uma unidade de sumarização de *clusters* que permite a seleção de subconjuntos desses *clusters*, de acordo com preferências do usuário. Esse repositório contem somente *clusters* que satisfazem pelo menos dois objetivos. O passo de sumarização permite que usuários filtrem o repositório de *clusters*, com diferentes objetivos e limiares. O resultado é um "agrupamento final de um ponto de vista de um único ou pequeno conjunto

de objetivos que são de interesse particular para o usuário" (JIAMTHAPTHAKSIN; EICK; VILALTA, 2009).

1.3 Abordagem Proposta

O ASA_{Clu} parte do princípio do ASA em selecionar um conjunto reduzido de soluções relevantes e diversas, dado um multiconjunto inicial de partições obtidas pela aplicação de diversos algoritmos de agrupamento. Porém, ele busca *clusters* ao invés de partições. O objetivo principal desse trabalho é fazer uma adaptação do algoritmo ASA para selecionar *clusters*. Além disso, evidenciar as vantagens de selecionar *clusters* em vez de partições. A ideia é que dentro do contexto de múltiplas soluções em agrupamento, a solução final seja um conjunto de *clusters* e não de partições. No Capítulo 5 foram feitos experimentos comprovando a eficiência em selecionar *clusters*, e os resultados do ASA_{Clu} foram comparados aos resultados do ASA.

O MOC e o $MBCS$ seguem a mesma ideia do ASA_{Clu} em selecionar *clusters*. A diferença para o $MBCS$ é que o ASA_{Clu} complementa o conjunto reduzido com mais *clusters* de qualidade, num passo posterior ao passo de selecionar de acordo com a multiplicidade, e ele também permite ajustar o limiar de multiplicidade, podendo selecionar mais ou menos *clusters* de acordo com o número de vezes que aparecem na coleção inicial. O ASA_{Clu} difere do MOC no aspecto do critério para selecionar um *cluster*. No caso do MOC, os *clusters* selecionados devem ser multi-objetivos e os objetivos devem ser escolhidos pelo usuário. Já no ASA_{Clu}, os *clusters* são selecionados por sua relevância ao serem mais similares a outros *clusters* das partições iniciais, e por sua diversidade ao descartar *clusters* altamente similares àqueles já adicionados na solução final.

1.4 Organização do Trabalho

O Capítulo 2 apresenta os trabalhos relacionados ao ASA_{Clu}. Esses trabalhos foram utilizados como base para comparação com os resultados do ASA_{Clu} e como base para o desenvolvimento do algoritmo. O Capítulo 3 apresenta o algoritmo. O Capítulo 4 apresenta os experimentos e os métodos utilizados, para ajustes de parâmetro e dos algoritmos tradicionais de agrupamento para compor o conjunto inicial. O Capítulo 5 apresenta os resultados da aplicação do ASA_{Clu} e faz uma comparação com os resultados do ASA e do $MBCS$.

2 Trabalhos relacionados

2.1 Considerações Iniciais

Muitas abordagens recentes e avançadas em agrupamentos dependem de algoritmos tradicionais e tem como objetivo encontrar múltiplas soluções alternativas. Nesta seção, serão brevemente descritas as abordagens usadas neste trabalho de pesquisa e aquelas que motivaram as ideias presentes aqui. Também, serão descritas as notações usadas nos algoritmos desta seção e das próximas.

- $C_C = \{(c^i, n_i) \mid c^i \in C_u, n_i \in Z^+\}$ é um multiconjunto de $|C_C|$ *clusters*, onde C_u é o conjunto subjacente de C_C, com $|C_u|$ *clusters*, e n_i é a multiplicidade do *cluster* c^i.
- $C_I = \left\{c^1, c^2, ..., c^I\right\}$ é um **conjunto reduzido**, onde $C_I \subset C_C$ e $|C_I|$ deve ser muito menor do que $|C_C|$.
- $\Pi_C = \{(\pi^i, p_i) \mid \pi^i \in \Pi^u, p_i \in Z^+\}$ é um multiconjunto de $|\Pi_C|$ partições, onde Π^u é o conjunto subjacente de Π_C, e p_i é a multiplicidade da partição π^i.
- $\Pi_R = \left\{\pi^1, \pi^2, ..., \pi^{|\Pi_R|}\right\}$ é um **conjunto reduzido**, onde $\Pi_R \subset \Pi_C$ e $|\Pi_R|$ deve ser muito menor do que $|\Pi_C|$.

2.2 *Automatic Selection Algorithm* (ASA)

Uma consideração importante na busca de múltiplas soluções em agrupamento é que as soluções alternativas devem ser bastante diferentes para que cada alternativa seja capaz de contribuir com conhecimento adicional (MüLLER et al., 2012). Por outro lado, quando uma solução pode ser encontrada por maneiras diferentes (diferentes algoritmos com diferentes critérios), isso é um sinal claro e evidente de relevância (SAKATA et al., 2010). O ASA é um algoritmo de seleção de partições, que considera a evidência de partições (dada a facilidade de identificação por diferentes critérios), e a diversidade da coleção de partições (SAKATA et al., 2010).

Essa estratégia de seleção usa o índice ARI (*Adjusted Rand Index*) (HUBERT; ARABIE, 1985) como medida de similaridade, para selecionar um subconjunto das partições mais diversas. O tamanho do conjunto de soluções é controlado por um limiar t do valor desse índice, que limita o grau de similaridade considerado para a exclusão de uma partição do conjunto de soluções. A redução do valor do limiar t reduz o número de soluções. O ASA automaticamente ajusta esse limiar t, até que nenhuma redução significativa no

número de soluções seja atingida. Assim, o ASA garante a diversidade das partições no conjunto reduzido, e não requer nenhum parâmetro do usuário.

O ASA é aplicado a um multiconjunto de partições Π_C gerado com algoritmos de agrupamento tradicionais. Foi observado em (FACELI et al., 2010) que aplicando a seleção diretamente a tal multiconjunto pode ser tão eficiente como sua aplicação aos resultados de técnicas de agrupamento mais complexas. Por isso, foi decidido focar na alternativa com o menor custo computacional de considerar Π_C como um *multiconjunto elementar*, isto é, um multiconjunto que contém partições geradas diretamente pela aplicação de diversos algoritmos tradicionais de agrupamento, sem nenhum tipo de processamento posterior, como por exemplo, a otimização multi-objetivo. Qualquer estratégia de agrupamento poderia ser usada para produzir Π_C. Todavia, para se ter vantagem do aspecto de seleção de partições evidentes, é necessário empregar diferentes algoritmos para produzir Π_C.

Dado um multiconjunto inicial de partições Π_C, o ASA produz um conjunto de partições Π_R, da seguinte maneira. Primeiro, ele inicializa Π_R com as partições mais evidentes, que foram encontradas simultaneamente por vários algoritmos de agrupamento. Para isso, foi incluído o passo de inicialização que adiciona diretamente em Π_R as partições que aparecem repetidamente em Π_C. Como Π_C é um multiconjunto elementar, o ASA adiciona uma partição π^i diretamente se sua multiplicidade $p_i \geq p$, onde p é o número de algoritmos de agrupamento distintos. Após π^i ser adicionada a Π_R, ela e todas a partições idênticas são removidas de Π_C. Então, ASA começa um processo iterativo:

1. Descarta partições de Π_C que são altamente similares aquelas já selecionadas. Uma partição π^j é considerada altamente similar a π^i quando $ARI(\pi^i, \pi^j) \geq t$, onde t é um limiar ajustado automaticamente pelo algoritmo.

2. Seleciona de Π_C uma nova partição π^i com maior ARI_m, e adiciona π^i a Π_R. $ARI_m\left(\pi^i\right)$ é a média de similaridade de uma partição π^i com todas as partições $\pi^j \in \Pi_C$, dada pela Equação 2.1.

$$ARI_m\left(\pi^i\right) = \frac{1}{|\Pi_C|} \sum_{\pi^j \in \Pi_C} ARI\left(\pi^i, \pi^j\right) \tag{2.1}$$

A ideia por trás do ASA é começar a seleção com um valor alto para t, decrementando este valor até que a redução no número de soluções torne-se pequena. Para isso, é calculada a taxa entre o número de partições selecionadas e o número inicial de partições. A diferença entre as taxas obtidas por dois valores consecutivos de t, é usada como o critério de parada para o algoritmo. Em (SAKATA et al., 2010), foi determinado empiricamente que se a diferença das taxas for menor ou igual a 0.12, o algoritmo não atinge mais nenhuma redução significante. Sendo assim, o valor de 0.12 é usado para

testar quando a redução deve parar. Observa-se que essa estratégia não requer nenhuma configuração do usuário. Ele escolhe automaticamente um limiar t, resultando em um conjunto de partições que é sempre muito menor do que o multiconjunto elementar. O Algoritmo 1 demonstra os passos do *ASA*.

Algoritmo 1 *ASA*

Entrada: Π_C, p // p é o número de algoritmos tradicionais distintos
Saída: Π_R

para todos $\pi^i \in \Pi_C$ **faça**
  se $p_i \geq p$ **então**
    $\Pi_R \leftarrow \Pi_R \bigcup \{\pi^i\}$
    para todos $\pi^j \in \Pi_C$ **faça**
      se $\pi^j = \pi^i$ **então**
        $\Pi_C \leftarrow \Pi_C - \{\pi^j\}$
$nInitial \leftarrow n^C + n^R$
$t \leftarrow 0.9$ // valor inicial do limiar
$r_{current} \leftarrow 1.0$ // valor atual da taxa
$\Pi_{current} \leftarrow \Pi_C$
repita
  $r_{previous} \leftarrow r_{current}$
  $\Pi_{previous} \leftarrow \Pi_{current}$
  $\Pi_{current} \leftarrow \emptyset$
  para todos $\pi^i \in \Pi_R$ **faça**
    para todos $\pi^j \in \Pi_C$ **faça**
      se $ARI\left(\pi^i, \pi^j\right) \geq t$ **então**
        $\Pi_C \leftarrow \Pi_C - \{\pi^j\}$
  para todos $\pi^i \in \Pi_C$ **faça**
    Calcular o $ARI_m\left(\pi^i\right)$.
  repita
    $\pi^d \leftarrow \pi^i \in \Pi_C$
    para todos $\pi^j \in \Pi_C$ **faça**
      se $ARI_m\left(\pi^d\right) < ARI_m\left(\pi^j\right)$ **então**
        $\pi^d \leftarrow \pi^j$
    $\Pi_C \leftarrow \Pi_C - \pi^d$
    $\Pi_{current} \leftarrow \Pi_{current} \bigcup \{\pi^d\}$
    para todos $\pi^j \in \Pi_C$ **faça**
      se $ARI\left(\pi^d, \pi^j\right) \geq t$ **então**
        $\Pi_C \leftarrow \Pi_C - \{\pi^j\}$
  até que Π_C esteja vazio
  $t \leftarrow t - 0.1$
  $\Pi_C \leftarrow \Pi_{current}$
  $r_{current} \leftarrow nCurrent/nInitial$ // onde $nCurrent$ é o número de partições em $\Pi_{current}$
até que $(r_{previous} - r_{current}) \leq 0.12$ *or* $t < 0.1$
$\Pi_R \leftarrow \Pi_R \cup \Pi_{previous}$

Em (SAKATA et al., 2010), pode-se perceber que o *ASA* foi aplicado diretamente no multiconjunto de partições geradas por algoritmos tradicionais de agrupamento, sendo

eficiente em reduzir o número de soluções enquanto mantém partições de alta qualidade no conjunto reduzido. Isso significa que nenhuma técnica de alto custo computacional, como agrupamento multi-objetivo baseado em Pareto, foi necessária para alcançar tais bons resultados.

2.3 *Multiplicity Based Cluster Selection* ($MBCS$)

Como a multiplicidade de soluções em uma coleção de partições tem sido usada com sucesso no ASA, foi investigado o potencial da mesma informação no contexto de *clusters* (FACELI; SAKATA, 2016). Para tal, será considerada a seleção de *clusters* com multiplicidade maior ou igual a dois, isto é, *clusters* que aparecem mais de uma vez na coleção inicial são selecionados para o conjunto final de soluções. Essa abordagem para produzir uma coleção de *clusters* ao invés de partições é referida como *Multiplicity Based Cluster Selection* ($MBCS$). Ele utiliza o mesmo princípio do ASA em usar um multiconjunto elementar de partições geradas pela aplicação de diversos algoritmos de agrupamento. A diferença é que ele quebra essas partições em seus *clusters* componentes, formando o multiconjunto C_C. Então, ele parte da suposição de que se um *cluster* aparece pelo menos duas vezes em C_C, ele é uma solução evidente.

Dado um multiconjunto de *clusters* C_C gerado pela aplicação de algoritmos de agrupamento, selecionar um conjunto reduzido de *clusters* C_I com *clusters* $c^i \in C_C$ tendo $n_i \geq 2$, como descrito no Algoritmo 2.

Algoritmo 2 $MBCS$

Entrada: C_C
Saída: C_I

```
1: n_r ← 2 // limiar da multiplicidade de clusters
2: para todos c^i ∈ C_C faça
3:    se n_i ≥ n_r então
4:       C_I ← C_I ⋃ {c^i}
5:       para todos c^j ∈ C_C faça
6:          se c^j = c^i então
7:             C_C ← C_C − {c^j}
```

Em (FACELI; SAKATA, 2016), foi fornecida uma metodologia para avaliar um conjunto de múltiplas soluções em agrupamento ao considerar os *clusters* como soluções em vez de partições. Então, foi feita uma comparação das 2 formas de analisar múltiplas soluções: a forma tradicional que compara partições e a forma de avaliar *clusters* obtidos independentemente das partições que eles pertencem. Sendo assim, mostrou-se que (i) mesmo um conjunto diverso de partições pode ter uma grande quantidade de informação redundante em seus *clusters*, e mais importante, que (ii) a qualidade da informação extraída é bastante subestimada ao avaliá-la pela análise de partições.

Por essa análise, foi constatado que a simples seleção dos *clusters* identificados repetidamente por algoritmos tradicionais ($MBCS$) pode levar a identificação de informação relevante. Além disso, ao comparar a quantidade de informação irrelevante obtida, foi possível perceber que o multiconjunto C_C obteve aproximadamente 18 vezes mais *clusters* distintos do que os *clusters* reais existentes. Ao selecionar os *clusters* de C_C com multiplicidade de pelo menos 2 ($MBCS$), foi obtido 8 vezes mais *clusters* do que os *clusters* reais, o que foi a menor quantidade de *clusters* irrelevantes obtidos, comparado com as outras técnicas.

Em (FACELI; SAKATA, 2016) concluiu-se, que o ASA e o $MBCS$ não foram capazes de manter toda a informação presente na coleção inicial de partições. Por outro lado, ao custo de uma pequena perda de informação relevante, uma redução significante em informação irrelevante produzida foi alcançada junto com uma redução no custo computacional exigido para produzir os resultados, em comparação ao custo computacional exigido por algoritmos multi-objetivos.

2.4 *Multi-Objective Clustering* (MOC)

O MOC é um *framework* multi-objetivo que também seleciona *clusters* ao invés de partições. Ele aplica algoritmos de agrupamento que suportam funções de aptidão. Essas funções de aptidão otimizam diferentes objetivos. Diferentemente de outras técnicas multi-objetivo, o MOC procura *clusters* que individualmente satisfaçam mais de um objetivo. O usuário fornece limiares de satisfação para cada objetivo. Além de satisfazer mais de um objetivo de acordo com os limiares fornecidos pelo usuário, os *clusters* selecionados fazem parte do Pareto-ótimo, não sendo dominados por nenhum outro *cluster* em relação a todos os objetivos. No final, o usuário pode filtrar os *clusters* selecionados, fornecendo um ou mais objetivos e limiares de interesse. O resultado é um "agrupamento final de um ponto de vista de um único ou pequeno conjunto de objetivos que são de interesse particular para o usuário" (JIAMTHAPTHAKSIN; EICK; VILALTA, 2009).

O principal objetivo do agrupamento multi-objetivo (MOC) é encontrar *clusters* individuais que são bons em relação a múltiplos objetivos. Devido à natureza do MOC, apenas os *clusters* que são bons em relação a pelo menos dois objetivos são relatados. Essa estratégia assume que $Q = \{q_1, q_2, ..., q_z\}$ é o conjunto de objetivos que o agrupamento multi-objetivo maximiza. Para cada objetivo $q \in Q$, uma função de aptidão $Reward_q$ deve ser fornecida, que mede em que extensão o objetivo q é satisfeito. Quanto maior a aptidão, melhor é o *cluster* em relação ao objetivo q. Além disso, limiares $\theta_{q1}, ..., \theta_{qz}$ são associados a cada função de aptidão. Se $Reward_q(c^i) > \theta_q$, então o *cluster* c^i satisfaz o objetivo q. No geral, o objetivo do MOC é encontrar *clusters* que satisfaçam um grande número de objetivos em Q, mas raramente todos os objetivos, já que diferentes *clusters* satisfazem

diferentes objetivos.

Essa estratégia emprega algoritmos de agrupamento que suportam funções de aptidão. Esses algoritmos de agrupamento são executados múltiplas vezes com as mesmas ou diferentes funções de aptidão e armazenam os *clusters* potencialmente interessantes em uma lista de *clusters* M. Cada vez que um novo agrupamento π é obtido, M é atualizado. Alguns *clusters* em π podem ser inseridos em M, e alguns *clusters* em M podem ser excluídos devido à chegada de melhores *clusters* em π. Somente *clusters* multi-objetivo e não dominados são armazenados em M. Será definido mais formalmente quais *clusters* M pode conter. Para um *cluster* ser considerado multi-objetivo, ele deve satisfazer pelo menos dois objetivos, como definido na Equação 2.2.

$$MO\_Cluster(c,Q) \iff \exists q \in Q\ \exists q^{'} \in Q(q \neq q^{'} \wedge Reward_q(c) \geq \theta_q \wedge Reward_{q^{'}}(c) \geq \theta_{q^{'}}) \tag{2.2}$$

$$\begin{aligned} Dominates(c^i,c^j,Q) \iff \forall q \in Q((Reward_q(c^i) \geq Reward_q(c^j) \vee \\ Reward_q(c^i) < \theta_q \wedge Reward_q(c^j) < \theta_q) \wedge JI(c^i,c^j) \geq \theta_{sim}) \end{aligned} \tag{2.3}$$

A Equação 2.3 introduz a relação de dominância entre *clusters*. No geral, c^i e c^j são comparados somente nos objetivos que pelo menos um deles satisfaz. Além disso, a Equação 2.3 assume que *clusters* c^i e c^j são comparáveis se a similaridade entre eles for maior ou igual ao limiar θ_{sim}, fornecido pelo usuário. A similaridade entre dois *clusters* é medida pelo índice Jaccard $JI(c^i,c^j) = \frac{c^i \cap c^j}{c^i \cup c^j}$, que é a razão do número de objetos comuns entre c^i e c^j sobre o número total de objetos em c^i e c^j. O mesmo índice JI é utilizado no algoritmo ASA_{Clu} para medir a similaridade entre *clusters*.

Por último, uma unidade de sumarização de *clusters* recupera um subconjunto $M^{'}$ de M baseando-se em preferências do usuário. O usuário fornece um subconjunto $Q^{'} \subset Q$ de objetivos, limiares θ_q para cada $q \in Q^{'}$ e um limiar de similaride de exclusão de *clusters* θ_{rem}. Então, o algoritmo retorna *clusters* que são bons em relação a $Q^{'}$ e remove *clusters* que são muito similares, com base em θ_{rem} (basicamente, se dois *clusters* são muito similares, um não é incluído no resultado final).

A Figura 5 mostra a arquitetura do *framework* MOC, que consite de 4 componentes principais: um algoritmo de agrupamento, unidade de armazenamento, gerador de função de aptidão orientada por objetivo e unidade de sumarização de *clusters*. O MOC funciona da seguinte forma: No passo S1, o gerador de função de aptidão seleciona uma nova função de aptidão para o algoritmo de agrupamento, o qual gera um novo agrupamento π no segundo passo (S2). No passo S3, a unidade de armazenamento atualiza a lista de *clusters* M, usando os *clusters* em π. O algoritmo itera nesses 3 passos até que um número grande

de *clusters* tenha sido obtido. No passo S4, a unidade de sumarização de *clusters* produz os *clusters* finais baseando-se em preferências do usuário, que é um subconjunto dos *clusters* em M.

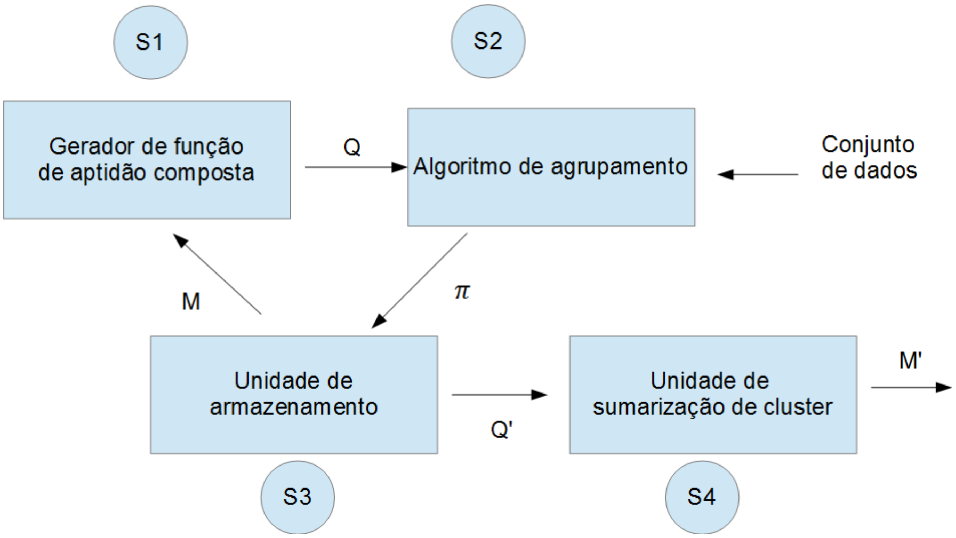

Figura 5 – Arquitetura do MOC.

Essa estratégia tem o foco em agrupamento multi-objetivo. Em particular, o interesse é apoiar aplicações em que um grande número de diversos, muitas vezes contraditórios, objetivos são dados e a meta é encontrar *clusters* que satisfaçam grandes subconjuntos desses objetivos. Aplicações que precisam de tais capacidades incluem sistemas de recomendação, problemas de satisfação de restrições que envolvem muitas restrições, problemas de desenho complexo e análise de associação. O MOC é de natureza orientada ao domínio, em que usuários podem expressar critérios de agrupamento baseados em necessidades específicas do domínio e não baseados em funções objetivo altamente genéricas e independentes de domínio, que é a abordagem da maioria dos algoritmos tradicionais de agrupamento.

2.5 Considerações Finais

As ideias do ASA, $MBCS$ e MOC motivaram a técnica do ASA_{Clu}, em investigar a qualidade dos *clusters* dentro de um multiconjunto de partições produzidas com algoritmos tradicionais de agrupamento. Desse modo, os resultados dos algoritmos são analisados como se fossem repositórios de *clusters* e então, é analisada a extensão que eles envolvem os *clusters* reais escondidos entre diversas estruturas subjacentes de um dado conjunto de dados.

3 O algoritmo ASA_{Clu}

3.1 Considerações Iniciais

Nesta seção, será descrita a técnica de seleção de *clusters* ASA_{Clu}. Também serão introduzidas as fórmulas e notações usadas no algoritmo.

- $C_R = \left\{c^1, c^2, ..., c^{|C_R|}\right\}$ é um **conjunto reduzido**, onde $C_R \subset C_C$ e $|C_R|$ deve ser muito menor do que $|C_C|$.
- $JI\left(c^i, c^j\right) = \frac{c^i \cap c^j}{c^i \cup c^j}$ é o índice Jaccard entre dois *clusters* c^i e c^j, utilizado para medir a similaridade entre dois *clusters*.
- $JI_m\left(c^i\right) = \frac{1}{|C_C|} \sum_{c^j \in C_C} JI\left(c^i, c^j\right)$ é a média da similaridade de um *cluster* c^i com todos os *clusters* $c^j \in C_C$.

3.2 Descrição da Proposta

Ao perceber a eficiência do ASA em selecionar soluções relevantes com um baixo custo computacional e as vantagens de analisar *clusters* em vez de partições em (FACELI; SAKATA, 2016), neste trabalho de pesquisa foi proposta uma nova estratégia de seleção de *clusters* denominada ASA_{Clu}. Ela se baseia no algoritmo ASA para a seleção das soluções e no algoritmo $MBCS$ para avaliar *clusters* independentemente da partição que estão inseridos.

A Figura 6 mostra o procedimento completo de aplicação do algoritmo ASA_{Clu} para um conjunto de dados. No Passo 1 (P1), os algoritmos de agrupamento são aplicados ao conjunto de dados, variando seus parâmetros. No Passo 2 (P2), as partições geradas pelos algoritmos de agrupamento são quebradas em seus *clusters* componentes, formando uma coleção de *clusters* C_C. No Passo 3 (P3), o ASA_{Clu} é aplicado, recebendo como entrada a coleção de *clusters* C_C e o parâmetro n_r. No final de todo esse procedimento, o conjunto reduzido de *clusters* C_R é obtido. O Passo 1 é idêntico ao procedimento do ASA na geração das partições iniciais. Esse procedimento será explicado com mais detalhes no Capítulo 4. O Passo 2 é idêntico ao procedimento do $MBCS$ na quebra dos *clusters*. Todos os *clusters* gerados pelas partições são inseridos numa coleção C_C.

Essa nova estratégia de seleção utiliza o índice Jaccard (JI) (JACCARD, 1901) para medir a similaridade entre dois *clusters* c^i e c^j. Ele mede a proporção dos objetos que se encontram tanto em c^i como em c^j. O seu valor varia de 0 a 1, sendo que 0 significa que

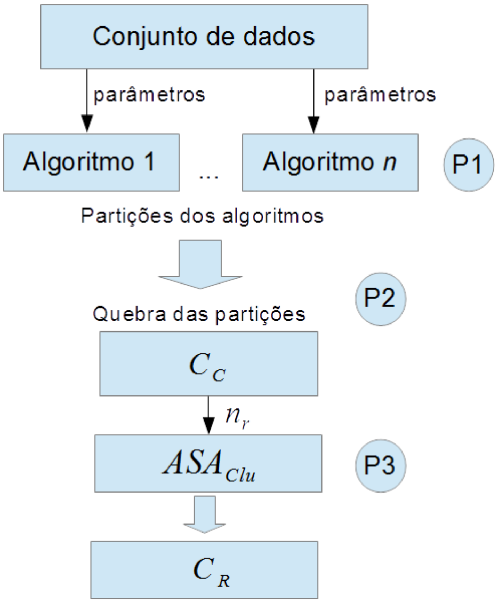

Figura 6 – Procedimento de aplicação do algoritmo ASA_{Clu}.

nenhum objeto está presente ao mesmo tempo nos dois *clusters* e 1 significa que os dois *clusters* são identicos. A Equação 3.1 mostra o índice Jaccard entre dois *clusters* c^i e c^j.

$$JI\left(c^i, c^j\right) = \frac{c^i \cap c^j}{c^i \cup c^j} \tag{3.1}$$

Dado um multiconjunto de *clusters* C_C gerado por um conjunto de algoritmos tradicionais de agrupamento, ASA_{Clu} seleciona um conjunto reduzido C_R de *clusters* diversos e relevantes. Um *cluster* c^i é considerado relevante pela sua multiplicidade n_i ou pela sua similaridade média JI_m. A diversidade é mantida em C_R, descartando *clusters* similares àqueles já adicionados em C_R. A Figura 7 mostra os passos gerais de execução do algoritmo ASA_{Clu}.

O ASA_{Clu} começa com o passo de inicialização (P1 em 7), adicionando diretamente em C_R um *cluster* c^i com multiplicidade $n_i \geq n_r$, onde n_r é um parâmetro do algoritmo, e removendo de C_C todos os *clusters* idênticos a c^i. Então, ele itera nos seguintes passo gerais (P2, P3 em 7).

1. Descarta todos os *clusters* de C_C que são altamente similares aqueles já selecionados. Um *cluster* c^j é considerado altamente similar a c^i quando $JI(c^i, c^j) \geq t$, onde t é um limiar ajustado automaticamente pelo algoritmo.

2. Seleciona de C_C e adiciona ao C_R, o *cluster* c^i com maior JI_m.

O ASA_{Clu} para de reduzir, quando alcança o critério de parada (P4 em 7), um valor que determina a diferença mínima entre duas taxas de redução, em iterações consecutivas

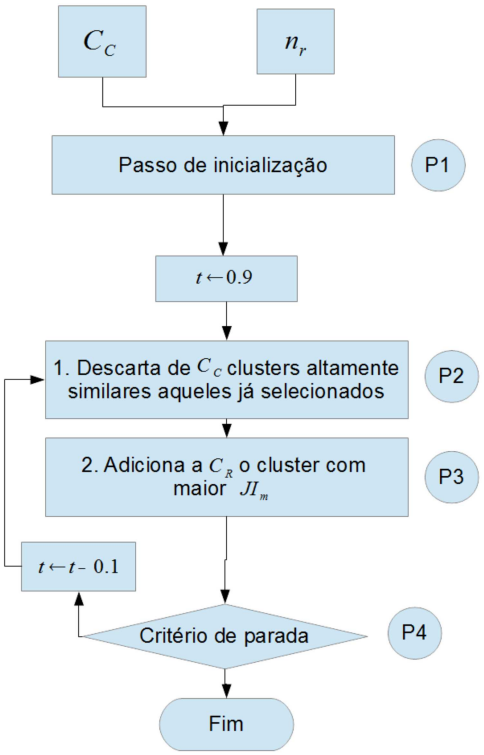

Figura 7 – Passos gerais de execução do algoritmo ASA_{Clu}.

do algoritmo. A ideia é a mesma do ASA: começar com um alto valor para t, diminuindo esse valor até que a redução no número de *clusters* se torne pequena. Para isso, a taxa de redução é obtida entre o número de *clusters* selecionados e o número inicial de *clusters*. A diferença das taxas é obtida para dois valores consecutivos de t, e é comparada ao critério de parada. Se a diferença for igual ou menor do que o critério de parada então a redução deve parar. O valor do critério de parada de 0.13 foi determinado empiricamente, conforme será descrito no Capítulo 4.

O Algoritmo 3 mostra de forma mais detalhada todos os passos do ASA_{Clu}. O algoritmo recebe como entrada C_C e n_r. A diferença para o ASA é que ele recebe uma coleção de partições Π_C no lugar de C_C. O ASA avalia as partições geradas pelos algoritmos e não seus *clusters* individualmente. O parâmetro p do ASA é o limiar da multiplicidade de partições, e o n_r do ASA_{Clu} é o limiar da multiplicidade de *clusters*. Os passo do Algoritmo 3 de 1 a 6 se referem ao passo de inicialização, onde os *clusters* com multiplicidade maior ou igual a n_r são adicionados diretamente no conjunto reduzido C_R. A diferença para o ASA é que ele adiciona diretamente no conjunto reduzido Π_R, as partições com multiplicidade maior ou igual a p. O passo 7 inicializa $nInitial$ com o número de *clusters* em C_C mais o número de *clusters* em C_R. Esse valor é usado como a razão para calcular a taxa de redução de *clusters* do algoritmo. Já no ASA, esse valor é utilizado para calcular a taxa de redução de partições, e ele é inicializado com o número de partições de Π_C mais o número de partições de Π_R. Os passo 8 e 9 em 3 são idênticos aos passos 8 e 9 do ASA

(1). O passo 8 inicializa o limiar t, que é responsável por determinar se uma solução é altamente similar aquelas já adicionadas ao conjunto reduzido, a fim de descartá-la. O passo 9 inicializa a taxa de redução atual, que é subtraída da taxa de redução anterior para a verificação do critério de parada (passo 35). O passo 10 do Algoritmo 3 inicializa a coleção atual de *clusters* $C_{current}$ com todos os *clusters* de C_C. Essa coleção atual é guardada na coleção anterior $C_{previous}$ (passo 13) para ser adicionada ao conjunto reduzido C_R depois que o algoritmo atinge o critério de parada (passo 36). No caso do ASA, o passo 10 em 1, $\Pi_{current}$ guarda partições em vez de *clusters* e é atribuído ao $\Pi_{previous}$ no final de cada iteração (passo 13), para ser adicionado a Π_R depois que o critério de parada é atingido (passo 36).

Basicamente, a diferença principal do ASA para o ASA_{Clu} é que o ASA é aplicado a partições e o ASA_{Clu} é aplicado a *clusters*. Além disso, o ASA utiliza o índice ARI para medir a similaridade entre duas partições, enquanto que o ASA_{Clu} utiliza o índice JI para medir a similaridade entre dois *clusters*. Já o valor do critério de parada é de 0.13 para o ASA_{Clu} e 0.12 e para o ASA. O valor de 0.12 foi determinado empiricamente em (SAKATA et al., 2010). A determinação do valor de 0.13 para o ASA_{Clu} será apresentada detalhadamente no Capítulo 4. Esse valor determina se a redução deve parar, partindo do pressuposto de que se a diferença entre duas taxas de redução consecutivas for menor ou a 0.13, o algoritmo não atingiria mais nenhuma redução significativa, e então deve parar.

Os passo 11 ao 35 fazem parte da iteração principal do Algoritmo 3, que é responsável por decrementar o valor do limiar t (passo 32). O valor de t começa com 0.9 e é decrementado em passos de 0.1 a cada iteração do algoritmo. Quanto menor o valor de t, mais *clusters* são descartados da solução final. Os passos 15 ao 18 descartam de C_C *clusters* que são altamente similares a *clusters* já adicionados em C_R, e os passos 28 a 30 descartam de C_C *clusters* que são altamente similares ao *cluster* c^d com maior similaridade média JI_m entre os *clusters* de C_C. Esses *clusters* são descartados para manter a diversidade na solução final, e o *cluster* c^d com maior JI_m é incluído na solução final pela suposição de sua evidência, já que é similar a maioria dos *clusters* de C_C.

Os passos 21 a 31 fazem parte de uma iteração dentro da iteração principal (passos 11 a 35 em 3) e são responsáveis por selecionar os *clusters* mais relevantes de acordo com JI_m e ao mesmo tempo descartar aqueles altamente similares para um valor de t. Para isso, o *cluster* mais relevante $c^d \in C_C$ é adicionado a $C_{current}$ (passos 22 a 27) e todos os outros *cluster* de C_C altamente similares a c^d são descartados (passos 28 a 30). Essa iteração continua até que C_C esteja vazio (passo 31).

A similaridade média JI_m é calculada para cada *cluster* $c^i \in C_C$ nos passos 19 e 20. Esses passos estão dentro da iteração principal do algoritmo (passos 11 a 35) e todas as similaridades são recalculadas a cada iteração.

Essa estratégia requer que o usuário forneça somente um parâmetro, n_r. Esse

Algoritmo 3 ASA_{Clu}

Entrada: C_C, n_r
Saída: C_R

para todos $c^i \in C_C$ **faça**
  se $n_i \geq n_r$ **então**
    $C_R \leftarrow C_R \bigcup \{c^i\}$
    para todos $c^j \in C_C$ **faça**
      se $c^j = c^i$ **então**
        $C_C \leftarrow C_C - \{c^j\}$
$nInitial \leftarrow C + R$
$t \leftarrow 0.9$ // valor inicial do threshold
$r_{current} \leftarrow 1.0$
$C_{current} \leftarrow C_C$
repita
  $r_{previous} \leftarrow r_{current}$
  $C_{previous} \leftarrow C_{current}$
  $C_{current} \leftarrow \emptyset$
  para todos $c^i \in C_R$ **faça**
    para todos $c^j \in C_C$ **faça**
      se $JI\left(c^i, c^j\right) \geq t$ **então**
        $C_C \leftarrow C_C - \{c^j\}$
  para todos $c^i \in C_C$ **faça**
    Calculate the $JI_m\left(c^i\right)$.
  repita
    $c^d \leftarrow c^i \in C_C$
    para todos $c^j \in C_C$ **faça**
      se $JI_m\left(c^d\right) < JI_m\left(c^j\right)$ **então**
        $c^d \leftarrow c^j$
    $C_C \leftarrow C_C - c^d$
    $C_{current} \leftarrow C_{current} \bigcup \{c^d\}$
    para todos $c^j \in C_C$ **faça**
      se $JI\left(c^d, c^j\right) \geq t$ **então**
        $C_C \leftarrow C_C - \{c^j\}$
  até que C_C esteja vazio
  $t \leftarrow t - 0.1$
  $C_C \leftarrow C_{current}$
  $r_{current} \leftarrow nCurrent/nInitial$ // onde $nCurrent$ é o número de *clusters* em $C_{current}$
até que $(r_{previous} - r_{current}) \leq 0.13$ *or* $t < 0.1$
$C_R \leftarrow C_R \cup C_{previous}$

parâmetro é sensível à multiplicidade dos *clusters* em C_C. Se há um grande número de *clusters* redundantes em C_C, escolhendo um n_r pequeno levará à seleção de um grande número de *clusters* no passo de inicialização. Por outro lado, um n_r muito grande implicaria na seleção de poucos ou nenhum *cluster* no passo de inicialização, o que significa que o algoritmo não está tendo vantagem da característica da evidência de um *cluster* como um sinal de sua relevância.

3.3 Considerações Finais

Foi apresentado neste capítulo, a descrição mais detalhada do algoritmo ASA_{Clu} e suas semelhanças com o algoritmo ASA apresentado no Capítulo 2. O ASA_{Clu} é uma adaptação do ASA para selecionar *clusters*. Sua principal proposta é utilizar a vantagem em analisar *clusters* independentemente da partição que estão inseridos. No Capítulo 5, essa vantagem será validada através de experimentos e comparações entre as duas abordagens de seleção de *clusters* e seleção de partições.

O Capítulo 4 fará um estudo mais aprofundado de cada parte do algoritmo ASA_{Clu}, mostrando através de experimentos os resultados obtidos em relação ao conjunto ótimo de algoritmos tradicionais de agrupamento para a formação do multiconjunto C_C, a escolha do critério de parada e um estudo do comportamento do parâmetro n_r. Todos esses resultados serão analisados para validar algumas hipóteses e ajustar a configuração ideal para o algoritmo.

4 Materiais e Métodos

4.1 Considerações Iniciais

Neste capítulo, serão descritos os dados e os procedimentos empregados para o ajuste de parâmetros do ASA_{Clu} e validar o seu uso com conjuntos de dados independentes, não aqueles usados para ajuste de parâmetros. Primeiramente, serão descritos os conjuntos de dados empregados nos experimentos bem como o procedimento para a criação dos multiconjuntos de *clusters* e o procedimento usado para avaliar os resultados. A seguir será detalhado o processo de ajuste do parâmetro do critério de parada (Seção 4.4) e a escolha da melhor combinação de algoritmos de agrupamento para a criação do multiconjunto completo que serve como entrada para o ASA_{Clu} (Seção 4.5).

4.2 Conjuntos de Dados

Para os experimentos, foram usados 37 conjuntos de dados (15 artificiais e 22 reais) com diferentes dimensões, estruturas conhecidas e tamanhos, obtidos do *Cluster Evaluation Benchmark* (CEB). O CEB é composto por conjuntos de dados provenientes de outras fontes, comumente usados como *benchmarks* e foi proposto em (FACELI; SAKATA, 2016) como um *benchmark* para sua metodologia de validação de *clusters*. A Tabela 1 mostra as características dos 37 conjuntos de dados. Os 10 conjuntos de dados marcados com * são **conjuntos de dados de ajustes**, usados para ajustes no algoritmo. Eles foram escolhidos para manter a diversidade de características, como dimensão, número de objetos, dados reais ou artificiais, número de estruturas conhecidas, definição de *clusters* e área de domínio. Os 27 conjuntos de dados restantes foram usados como amostras independentes para avaliar o ASA_{Clu}. A coluna n é o número de elementos do conjunto, a coluna d é o número de atributos, a coluna n^E é o número de estruturas conhecidas, a coluna K é o número de *clusters* de cada estrutura conhecida e a coluna K_C é o número de *clusters* distintos das estruturas conhecidas.

Os **conjuntos de dados de ajuste** foram escolhidos pela diversidade de suas características. Assim os valores ajustados para o algoritmo ASA_{Clu} não ficam tendenciosos para um tipo de dados em específico. Desses 10 conjuntos, 5 são conjuntos reais que representam problemas de bioinformática. As estruturas conhecidas dos conjuntos reais correspondem a diferentes classificações conhecidas dos dados. Assim, é assumido que as divisões dos dados nas classes estão em concordância com algum dos critérios de agrupamentos empregados, sendo as classes conhecidas referidas como *clusters*. Algumas das classificações podem não ter relação com um critério de agrupamento, resultando em

Tipo	Conjunto de dados	n	d	n^E	K	K_C
Artificial	`ds2c2sc13` * (FACELI; SAKATA, 2016)	588	2	3	2, 5, 13	19
	`monkey` * (FACELI; SAKATA, 2016)	4000	2	4	8,5,3,2	14
	`spiralsquare` * (FACELI; SAKATA, 2016)	2000	2	2	2, 6	8
	`twoDiamonds` * (ULTSCH et al., 2015)	800	2	1	2	2
	`wingNut` * (ULTSCH et al., 2015)	1016	2	1	2	2
	`atom` (ULTSCH et al., 2015)	800	3	1	2	2
	`ds3c3sc6` (FACELI; SAKATA, 2016)	905	2	2	3, 6	8
	`ds4c2sc8` (FACELI; SAKATA, 2016)	485	2	2	2, 8	10
	`engyTime` (ULTSCH et al., 2015)	4096	2	1	2	2
	`gaussian3` (MONTI et al., 2015)	60	600	1	3	3
	`hepta` (ULTSCH et al., 2015)	212	3	1	7	7
	`lsun` (ULTSCH et al., 2015)	400	2	1	3	3
	`simulated6` (MONTI et al., 2015)	60	600	1	6	6
	`target` (ULTSCH et al., 2015)	770	2	1	6	6
	`tetra` (ULTSCH et al., 2015)	400	3	1	4	4
Real	`armstrong` * (ARMSTRONG et al., 2002)	72	1081	2	2,3	4
	`golub` * (GOLUB et al., 1999)	72	3571	4	2, 3, 2, 4	10
	`laryngeal2` * (KUNCHEVA et al., 2015)	692	16	1	2	2
	`miRNAcancer` * (LU et al., 2005)	218	217	6	3, 20, 4, 9, 2, 2	40
	`yeoh` * (YEOH et al., 2002)	248	2526	2	2, 6	7
	`chowdary` (CHOWDARY et al., 2006)	104	182	1	2	2
	`contractions` (KUNCHEVA et al., 2015)	98	27	1	2	2
	`dyrskjot` (DYRSKJØT et al., 2003)	40	1203	1	3	3
	`eTongueSugar` (FACELI; SAKATA, 2016)	375	6	2	2,3	5
	`glass` (AHA et al., 2015)	214	9	3	2, 5, 6	9
	`gordon` (GORDON et al., 2002)	181	1626	1	2	2
	`iris` (AHA et al., 2015)	150	4	1	3	3
	`laryngeal1` (KUNCHEVA et al., 2015)	213	16	1	2	2
	`laryngeal3` (KUNCHEVA et al., 2015)	353	16	2	2,3	4
	`libras` (AHA et al., 2015)	360	90	2	8,15	21
	`lung` (BHATTACHARJEE et al., 2001)	197	1000	1	4	4
	`respiratory` (KUNCHEVA et al., 2015)	85	17	1	2	2
	`segmentation` (AHA et al., 2015)	2310	19	1	7	7
	`su` (SU et al., 2001)	174	1571	1	10	10
	`voice3` (KUNCHEVA et al., 2015)	238	10	2	2,3	4
	`voice9` (KUNCHEVA et al., 2015)	428	10	2	2,9	10
	`weaning` (KUNCHEVA et al., 2015)	302	17	1	2	2

Tabela 1 – Conjuntos de dados (FACELI; SAKATA, 2016) com diferentes características. Os 10 conjuntos de dados marcados com * são **conjuntos de dados de ajustes**.

um baixo desempenho de todas as técnicas de agrupamento utilizadas. Já, os conjuntos de dador artificias foram construídos com o propósito de conter *clusters* bem definidos em relação a um ou mais critérios de agrupamento. As estruturas conhecidas dos 5 conjuntos de dados artificais estão descritas a seguir.

A Figura 8 mostra as estruturas conhecidas do conjunto de dados `monkey`. Essas estruturas contêm diferentes níveis de refinamento, isto é, estruturas com diferentes números de *clusters*, e diferentes tipos de *clusters*, como *clusters* encadeados, *clusters* globulares e

clusters baseados em densidade.

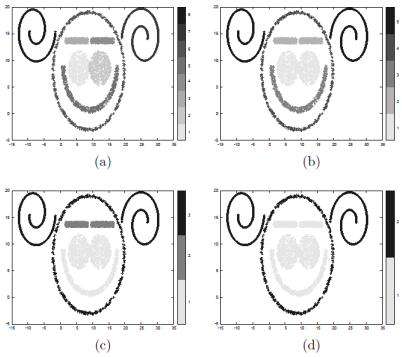

Figura 8 – Estruturas conhecidas do conjunto de dados `monkey` (FACELI; SAKATA, 2016).

Os conjuntos de dados `twoDiamonds` e `wingnut` foram retirados de (ULTSCH et al., 2015). A Figura 9 mostra a única estrutura conhecida do conjunto de dados `twoDiamonds`. Essa estrutura é composta por dois *clusters* em formato de diamante com as bordas definidas pela densidade dos objetos. A Figura 10 mostra a única estrutura conhecida do conjunto de dados `wingnut`. Essa estrutura é composta por dois *clusters* retangulares separados por uma distância.

Figura 9 – Estrutura conhecida do conjunto de dados `twoDiamonds` (ULTSCH et al., 2015).

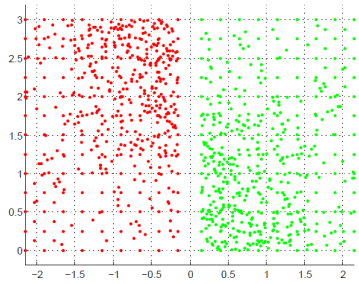

Figura 10 – Estrutura conhecida do conjunto de dados `wingnut` (ULTSCH et al., 2015).

O conjunto de dados `ds2c2sc13` foi especialmente projetado para conter três estruturas diferentes: E1, E2 e E3. A Figura 11 mostra esse conjunto de dados e suas estruturas. Como pode ser observado, E1 é a estrutura mais geral e contém dois *clusters*,

E2 é um refinamento de E1 e contém cinco *clusters*, e E3 é um refinamento de E2, com 13 *clusters*. Também pode ser notado que os *clusters* nesse conjunto de dados têm tamanhos e formas variados. E1 é a estrutura que mais se destaca, sendo obtida facilmente por qualquer técnica de agrupamento. Nessa estrutura, os *clusters* têm um formato aproximadamente esférico e estão muito bem separados uns dos outros. Já nas estruturas E2 e E3, os *clusters* são bastante heterogêneos. Em E2, há um *cluster* em formato de sorriso, um *cluster* alongado e três *clusters* aproximadamente globulares. Já na estrutura E3, o mesmo *cluster* em forma de sorriso aparece, mas o *cluster* alongado da estrutura E2 pode ser visto como três *clusters* esféricos. Além disso, cada um dos *clusters* globulares de E2 pode ser visto como três *clusters* alongados em E3.

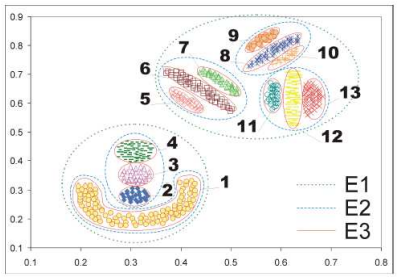

Figura 11 – Estruturas conhecidas do conjunto de dados `ds2c2sc13` (FACELI, 2007).

O conjunto de dados `spiralsquare` foi construído a partir de dois conjuntos de dados descritos em (HANDL; KNOWLES, 2004). Esse conjunto de dados contém duas estruturas: E1 com 2 *clusters* e E2 com 6 *clusters* (também um refinamento de E1), como pode ser observado na Figura 12. A estrutura mais facilmente distinguida é E1, que apresenta dois *clusters* esféricos e bem separados um do outro. Em E2, um dos *clusters* de E1 é subdividido em dois *clusters* em espiral e o outro é subdividido em outros quatro *clusters* globulares bastante próximos um do outro.

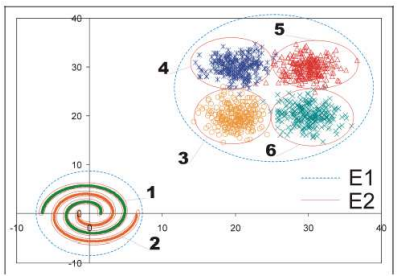

Figura 12 – Estruturas conhecidas do conjunto de dados `spiralsquare` (FACELI, 2007).

Muitos desses conjuntos de dados têm mais de uma estrutura conhecida e todos eles foram usados aqui para avaliar a qualidade dos resultados do ASA_{Clu}, da mesma forma feita em (FACELI; SAKATA, 2016), como descrito na seção a seguir.

4.3 Procedimentos de Obtenção dos *Clusters* e Avaliação da Qualidade dos Restultados

Para a criação dos multiconjuntos de *clusters*, foram usadas as partições fornecidas pelo CEB junto com partições produzidas com o algoritmo *Expectation Maximization* (EM) (MOON, 1996). Em CEB, as partições foram produzidas com diversos algoritmos de agrupamento, baseados em critérios diferentes, a saber: k-means (KM), average-link (AL), single-link (SL), complete-link (CoL), centroid-link (CeL) (JAIN; DUBES, 1988) e Shared Nearest Neighbors (SNN) (ERTOZ; STEINBACH; KUMAR, 2002). Uma descrição detalhada pode ser encontrada em (FACELI; SAKATA, 2016).

O software *Weka* (FRANK; HALL; WITTEN, 2016) foi utilizado para a geração das partições pelo algoritmo EM. Ele foi aplicado aos 10 conjuntos de dados de ajuste. O procedimento da geração das partições pelo algoritmo EM aplicado a cada conjunto de dados, levou em consideração o parâmetro k do algoritmo, e a inicialização dos centroides. O parâmetro k foi variado de 2 até duas vezes o número máximo de *clusters* das estruturas conhecidas. Os centroides foram gerados aleatoriamente. Por exemplo, o conjunto de dados `ds2c2sc13` tem 3 estruturas conhecidas (coluna n^E em 1), e a estrutura com o maior número de *clusters* tem 13 *clusters* no total (coluna K em 1). Com isso, foram geradas partições com o k de 2 a 26 (25 partições no total).

Como dito anteriormente, as partições geradas pelos outros algoritmos foram obtidas do CEB. Essas partições não foram geradas nesse trabalho de pesquisa. O procedimento para a geração dessas partições foi o mesmo utilizado para a geração das partições pelo algoritmo EM, descrito anteriormente. A distância Euclidiana foi utilizada como a medida de similaridade entre objetos para todos os algoritmos. Todas essas partições geradas foram quebradas em seus *clusters* componentes para a criação dos multiconjuntos de *clusters*, utilizados nos experimentos desse trabalho de pesquisa.

Na avaliação dos resultados, os *clusters* das estruturas conhecidas dos conjuntos de dados foram utilizados como comparação. Para os algoritmos terem um bom resultado, eles devem recuperar os *clusters* das estruturas conhecidas. Para isso, os *clusters* em C_C obtidos pelos algoritmos são comparados com os *clusters* das estruturas conhecidas C_{known}. Essa comparação é feita usando o índice Jaccard JI para medir a similaridade entre um *cluster* $c^i \in C_C$ e um *clusters* $c^{known} \in C_{known}$. Um *cluster* integralmente recuperado CR é um *cluster* c^{known} que corresponde exatamente a um ou mais *clusters* em C_C, e um *cluster* parcialmente recuperado PR é um *cluster* c^{known} onde $JI(c^{known}, c^j) > 0.7$ para pelo menos um $c^j \in C_C$.

4.4 Critério de Parada

Como mencionado no Capítulo 3, o ASA_{Clu} reduz iterativamente o número de *cluster* do C_C, variando o valor do limiar t até alcançar o critério de parada. O valor do critério de parada foi determinado empiricamente como segue, usando os 10 conjuntos de dados de ajuste. O multiconjunto C_C para cada conjunto de dados foi criado com todos os *clusters* de todas as partições geradas pelos 7 algoritmos de agrupamento $(KM, SL, EM, AL, CeL, CoL, SNN)$. O procedimento para a geração das partições foi descrito na seção anterior.

Para esse experimento, não será considerado o passo de inicialização do ASA_{Clu}, e será usado o conjunto de *clusters* distintos C_U subjacente ao multiconjunto C_C, já que o processo de redução acontece depois do passo de inicialização. O valor de t foi variado de 1.0 a 0.1 em passos de 0.1 e a taxa de redução foi extraída para cada valor de t, $r_t = R_t/|C_U|$, onde R_t é o número de *clusters* selecionados ao final de cada iteração. A Tabela 2 mostra as médias das taxas de redução r_t para os 10 conjuntos de dados e as diferenças entre duas taxas consecutivas $r_{t+0.1} - r_t$.

t	r_t	$r_{t+0.1} - r_t$
1.0	1.00	-
0.9	0.72	0.27
0.8	0.59	**0.13**
0.7	0.50	0.09
0.6	0.45	0.05
0.5	0.38	0.06
0.4	0.30	0.07
0.3	0.24	0.06
0.2	0.21	0.03
0.1	0.15	0.05

Tabela 2 – Médias da taxa de redução para cada valor de t.

No geral, as diferenças mostram uma redução muito baixa para $t < 0.8$. Sendo assim, o valor escolhido para o critério de parada foi a diferença entre $t = 0.9$ e $t = 0.8$, isso é 0.13. A ideia é parar quando nenhuma redução expressiva pudesse ser alcançada.

4.5 Combinação de Algoritmos de Agrupamento

A qualidade do conjunto de *clusters* selecionados pelo ASA_{Clu} é extremamente dependente da qualidade do C_C, já que o ASA_{Clu} não produz novos *clusters*, somente seleciona. Desse modo, é importante a utilização de um conjunto de algoritmos capaz de produzir a diversidade e a qualidade exigidas para lidar com conjuntos de dados tendo diferentes tipos de *clusters*. Ao mesmo tempo, o tamanho de C_C (e assim o número de

algoritmos e seus parâmetros de configuração) devem ser mantidos o menor possível, para o melhor desempenho. Para encontrar um bom equilíbrio e assim dar uma indicação de um bom conjunto de algoritmos para produzir o C_C, foram comparados os multiconjuntos produzidos pelas combinações de algoritmos descritas, considerando os conjuntos de dados de ajuste. Deve ser observado que o usuário pode aplicar o ASA_{Clu} em qualquer multiconjunto de *clusters*, independentemente desta sugestão.

Foram criados diferentes multiconjuntos de *clusters* (C_C), um para cada combinação de algoritmos, e o C_R de cada C_C foi comparado, a fim de buscar as combinações que encontram mais *clusters* de qualidade. O procedimento para a geração das partições que constroem o C_C e o procedimento de avaliação da qualidade dos *clusters* foram descritos na Seção 4.3. Foi comparado o CR de todas as combinações de dois algoritmos, e a combinação que recuperou mais *clusters* integralmente foi $KM + SL$. O mesmo procedimento foi feito para as combinações de 3 algoritmos, e resultou que a melhor delas em relação ao CR foi $KM + SL + CoL$. Para as combinações de 4 algoritmos, a melhor em relação ao CR foi $KM + SL + SNN + CoL$. A combinação de todos os 7 algoritmos obteve o mesmo CR que a combinação de 4 algoritmos $KM + SL + SNN + CoL$. Então, as combinações de 5 e 6 algoritmos não obtiveram nenhuma melhora em relação ao CR. A Tabela 3 apresenta a soma de $|C_C|$, PR e CR dos conjuntos de dados de ajuste, para as melhores combinações em relação ao valor de CR, considerando cada número de algoritmos 2, 3, 4 e 7. As combinações de um único algoritmo não foram avaliadas, pois a ideia é conseguir encontrar *clusters* de diferentes tipos (critérios de agrupamento).

Combinação	$\|C_C\|$	PR	CR
(1) $KM + SL$	3080	77	47
(2) $KM + SL + CoL$	4620	80	47
(3) $KM + SL + SNN + CoL$	5272	81	48
(4) $KM + SL + SNN + CoL$ $CeL + EM + AL$	2*9892	2*81	2*48

Tabela 3 – Resumo das melhores combinações.

As combinações (3) e (4) têm os mesmos valores para PR e CR. Isso indica que a combinação (3) de 4 algoritmos é melhor, já que o seu valor de $|C_C|$ é menor e menos algoritmos foram necessários para alcançar o mesmo resultado. Para as outras combinações (1) e (2), o valor de $|C_C|$ é menor, mas com um custo de recuperar menos *clusters*. A combinação (1), apesar de recuperar um *cluster* a menos que as outras combinações, obteve o menor número de *clusters* e um melhor custo computacional, já que apenas dois algoritmos foram aplicados.

Com as combinações da Tabela 3, foi feita uma análise detalhada com o ASA_{Clu}, incluindo o passo de inicialização que gera um conjunto de *clusters* C_I com I *clusters*. Para cada combinação aplicada a cada conjunto de dados, foi aplicado o ASA_{Clu} com

o parâmetro $n_r = 2$, isto é, *clusters* com multiplicidade maior ou igual a dois em C_C são inseridos diretamente em C_R e removidos de C_C. Esse valor torna a inicialização do ASA_{Clu} equivalente ao $MBCS$ e conduz a uma seleção mais ampla nesse passo inicial. Então, foi feita uma avaliação mais detalhada da capacidade do ASA_{Clu} em manter *clusters* de qualidade em C_R enquanto reduz o número de *clusters*. Para isso, como mencionado anteriormente, foi comparado cada *cluster* de C_R produzido com cada *cluster* C_{known}, para obter o número de *clusters* recuperados das estruturas conhecidas. Foi também comparado o número total de *clusters* selecionados pelo ASA_{Clu} e pelo passo de inicialização com o número inicialmente presente em C_C.

A Tabela 4 mostra o número de *clusters* em C_C, C_R e o número de *clusters* selecionados pelo passo de inicialização ($|C_I|$) para cada C_C produzido pelas combinações de algoritmos da Tabela 3 aplicados a cada conjunto de dados de ajuste. A coluna *total* é a soma do número de *clusters* de todos os conjuntos de dados de ajuste.

		armstrong-2002	ds2c2sc13	golub	laryngeal2	miRNAcancer	spiralsquare	twoDiamonds	wingNut	yeoh-2002-v1	monkey	total
$\|C_C\|$	(1)	40	700	70	18	1638	154	18	18	154	270	3080
	(2)	60	1050	105	27	2457	231	27	27	231	405	4620
	(3)	71	1223	158	27	2756	317	52	32	231	405	5272
	(4)	131	2273	263	54	5213	548	79	59	462	810	9892
$\|C_R\|$	(1)	21	166	35	13	378	50	10	9	56	101	**839**
	(2)	20	237	44	13	427	68	14	14	73	130	1040
	(3)	33	271	57	13	482	82	22	17	73	130	1180
	(4)	55	371	96	31	707	145	29	34	110	215	**1793**
$\|C_I\|$	(1)	7	101	9	2	132	22	3	3	12	32	**323**
	(2)	12	135	17	4	186	38	6	6	20	56	480
	(3)	14	142	21	4	228	42	11	6	20	56	544
	(4)	20	233	31	6	435	79	17	11	41	117	**990**

Tabela 4 – Número de *clusters*.

Pela Tabela 4, pode-se observar que o ASA_{Clu} e o passo de inicialização desempenharam uma boa redução no número de *clusters* inicialmente presentes em C_C para todas as combinações de algoritmos e todos os conjuntos de dados de ajuste. Por exemplo, se considerarmos a combinação de todos os algoritmos (4) com o número total de *clusters* selecionados para todos os conjuntos de dados, ASA_{Clu} selecionou 18% dos *clusters* em C_C (1793 de 9892), sendo 10% dos *clusters* de C_C selecionados somente pelo passo de inicialização (990 de 9892).

Naturalmente, C_I será menor que C_R, já que o ASA_{Clu} complementa C_I com outros *clusters* de C_C. O custo do ASA_{Clu} em selecionar mais *clusters* pode ser recompensado

quando mais *clusters* de qualidade são inseridos em C_R.

O menor número de *clusters* selecionados pelo C_R foi obtido pela combinação de somente dois algoritmos (1), no total 839 de 3080 inicialmente em C_C, sendo que 323 foram selecionados já no passo de inicialização. A aplicação de somente dois algoritmos naturalmente gera menos *clusters*, já que menos partições são geradas. Foi a combinação (1) que menos gerou *clusters* para o C_C, 3080 no total. A combinação que mais gerou *clusters* foi a combinação de todos os algoritmos (4), 9892 no total. Proporcionalmente o ASA_{Clu} também teve um número maior de *clusters* em C_R para a combinação (4), 1793 no total, sendo que 990 *clusters* foram selecionados somente no passo de inicialização. O ASA_{Clu} é sensível ao número de *clusters* em C_C, quanto mais *clusters* em C_C maior o número de *clusters* selecionados em C_R. Por isso é importante a escolha do conjunto de algoritmos que alimenta o ASA_{Clu}, para conseguir gerar um número não muito grande de *clusters*, e que esses *clusters* tenham boa qualidade.

A Tabela 5 mostra o número de *clusters* parcialmente recuperados (PR), em C_C, C_R e C_I, para as combinações de algoritmos da Tabela 3 e os conjuntos de dados de ajuste.

		armstrong-2002	ds2c2sc13	golub	laryngeal2	miRNAcancer	spiralsquare	twoDiamonds	wingNut	yeoh-2002-v1	monkey	total
C_C	(1)	3	19	7	1	21	8	2	2	2	12	77
	(2)	4	19	7	1	22	8	2	2	2	13	80
	(3)	4	19	7	1	23	8	2	2	2	13	81
	(4)	4	19	7	1	23	8	2	2	2	13	81
C_R	(1)	3	19	7	1	19	8	2	2	2	12	**75**
	(2)	3	19	7	1	21	8	2	2	2	13	78
	(3)	4	19	7	1	21	8	2	2	2	13	**79**
	(4)	4	19	7	1	21	8	2	2	2	13	79
C_I	(1)	1	19	2	0	15	8	0	2	0	7	54
	(2)	1	19	4	1	17	8	1	2	2	10	65
	(3)	1	19	5	1	19	8	2	2	2	10	69
	(4)	1	19	7	1	20	8	2	2	2	11	73

Tabela 5 – Número de *clusters* parcialmente recuperados.

Para o número total de PR (coluna *total*) de todas as combinações, o ASA_{Clu} perdeu somente 2 *clusters* dos *clusters* recuperados de C_C (75 de 77 para a combinação (1), 78 de 80 para a combinação (2), 79 de 81 para a combinação (3) e 79 de 81 para a combinação (4)). Comparando C_R com C_I, podemos observar alguns casos onde os passos do ASA_{Clu} depois do passo de inicialização não acrescentaram nenhum valor a respeito da qualidade dos *clusters* selecionados. Para os conjuntos de dados artificiais `ds2c2sc13`, `spiralsquare` e `wingNut`, o passo de inicialização já recuperou o mesmo número de *clusters*

presentes em C_C, para todas as combinações de algoritmos. O mesmo aconteceu para algumas outras combinações de algoritmos e conjuntos de dados, mas na maioria dos casos, C_R manteve o mesmo ou quase o mesmo número de *clusters* recuperados de C_C, isto é, ASA_{Clu} recuperou quase todos os *clusters* de boa qualidade presentes nos multiconjuntos correspondentes C_C. De fato, olhando para o PR considerando o conjunto de dados `miRNAcancer`, ASA_{Clu} perdeu somente 2 *clusters* nos piores casos para as combinações (1), (3) e (4).

O C_R produzido pelo ASA_{Clu} do C_C gerado pela combinação (1) contem 839 *clusters* no total (Tabela 4), do qual 75 correspondem aos *clusters* parcialmente recuperados (Tabela 5). Além de ter um menor custo em aplicar somente dois algoritmos, a combinação (1) pode selecionar um pequeno número de *clusters* e ao mesmo tempo manter *clusters* de boa qualidade. As outras combinações obtiveram uma ligeira vantagem em recuperar *clusters* de qualidade, 78 e 79 no total (Tabela 5). Por outro lado, a melhor delas (3) em termos de redução e qualidade, selecionou 40% mais *clusters* do que (1) (1180 contra 839 - Tabela 4). Levando em consideração o número de *clusters* selecionados e o número de *clusters* recuperados, a combinação (1) teve um melhor compromisso entre esses dois objetivos.

4.6 O Parâmetro n_r

O parâmetro n_r é utilizado no passo de inicialização do ASA_{Clu}, e ele indica a multiplicidade n_i mínima que um *cluster* $c^i \in C_C$ deve assumir para ser inserido diretamente no conjunto C_R ($n_i \geq n_r$). Quando o valor do parâmetro n_r for menor do que 1, o algoritmo está condicionado a não selecionar nenhum *cluster* no passo de inicialização, isto é, nenhum *cluster* é inserido diretamente em C_R. Quando o valor do parâmetro n_r for igual a 1, o algoritmo seleciona todos os *clusters* distintos de C_C e insere diretamente em C_R. Isso é equivalente a selecionar todos os *clusters* distintos sem mais nenhum passo de seleção.

O passo de inicialização seleciona os *clusters* mais relevantes de acordo com sua multiplicidade. Quanto mais vezes um mesmo *cluster* for encontrado em C_C, maior sua relevância. Essa suposição foi testada com partições em (SAKATA et al., 2010). Nesta seção, será investigada no contexto de *clusters*. Para estudar o comportamento da variação do parâmetro n_r em relação a relevância dos *clusters* selecionados, foi feito um experimento observando o algoritmo no que se refere ao número de *clusters* selecionados $|C_R|$, o número de *clusters* selecionados pelo passo de inicialização $|C_I|$ e o número de *clusters* parcialmente e integralmente recuperados PR e CR, em C_R e C_I. Todos os conjuntos de dados utilizados para esse experimento são os conjuntos de dados de ajuste, da Tabela 1. Esses conjuntos de dados são utilizados para ajustes no ASA_{Clu}, assim como o ajuste do conjunto ótimo de algoritmos, realizado nesta seção. A escolha desses conjuntos de dados foi justificada na Seção 4.2. A combinação de algoritmos $KM + SL$ foi utilizada para a construção do C_C

utilizado para o ASA_{Clu}. A escolha dessa combinação de algoritmos foi justificada na Seção 4.5. O procedimento para a aplicação desses algoritmos e a metodologia para a avaliação dos resultados foram descritos na Seção 4.3. As tabelas com todas essas informações para cada conjunto de dados estão no Apêndice A.

A Tabela 6 mostra as porcentagens de *clusters* parcialmente recuperados em relação ao número de *clusters* selecionados no passo de inicialização para cada valor do n_r, aplicado ao conjunto de dados artificial `ds2c2sc13`. Para esse conjunto, foram utilizadas 50 partições para alimentar o C_C, 25 partições geradas pelo KM e 25 pelo SL. Não há *cluster* com multiplicidade maior que 25, pois o passo de inicialização não selecionou nenhum *clusters* (coluna $|C_I|$) com $n_r > 25$. Para esse conjunto de dados artificial, a relevância de um *cluster* se mostrou maior, ao ser encontrado mais vezes. Os *clusters* parcialmente recuperados PR_I permanecem com uma boa proporção em relação aos *clusters* selecionados $|C_I|$ a medida que a multiplicidade aumenta (coluna n_r). A partir do $n_r = 14$, todos os *clusters* selecionados $|C_I|$ são parcialmente recuperados.

n_r	$\|C_I\|$	PR_I	%
1	239	19	7,94
2	101	19	18,81
3	63	18	28,57
4	45	16	35,55
5	36	16	44,44
6	30	15	50
7	28	15	53,57
8	25	15	60
9	21	13	61,90
10	19	12	63,15
11	15	12	80
12	14	11	78,57
13	13	10	76,92
14	10	10	100
15	10	10	100
16	8	8	100
17	6	6	100
18	4	4	100
19	4	4	100
20	4	4	100
21	1	1	100
22	1	1	100
23	1	1	100
24	1	1	100
25	1	1	100

Tabela 6 – Proporção dos *clusters* recuperados de acordo com o valor do n_r para o conjunto artificial `ds2c2sc13`.

Esse mesmo comportamento da relevância de um *cluster*, observado para o conjunto

de dados `ds2c2sc13` se repete para os conjuntos de dados artificiais `monkey`, `spiralsquare` e `wingNut`, exceto para o conjunto `twoDiamonds`. Ao Considerar somente a multiplicidade de um *cluster*, foi possível recuperar todos os *clusters* de qualidade para os conjuntos `ds2c2sc13` e `spiralsquare` (ver Apêndice A, Tabelas 13 e 9).

A Tabela 7 mostra os resultados de acordo com o valor do n_r para o conjunto real `golub`. Foram utilizadas 14 partições para alimentar o C_C. A maior multiplicidade encontrada foi de 7, pois o passo de inicialização não selecionou nenhum *clusters* com $n_r > 7$. A relevância de um *cluster* não se mostrou maior, ao ser encontrado mais vezes. Os *clusters* recuperados diminuem (coluna PR_I) a medida que os *clusters* com maior multiplicidade são selecionados (coluna $|C_I|$). Os conjuntos reais `armstrong`, `laryngeal2` e `yeoh` tiveram o mesmo comportamento que o conjunto `golub` em relação a relevância de um *cluster*, exceto o conjunto `miRNAcancer`.

n_r	$\|C_I\|$	PR_I	%
1	43	7	16,27
2	9	2	22,22
3	7	1	14,28
4	5	1	20
5	3	0	0
6	2	0	0
7	1	0	0

Tabela 7 – Proporção dos *clusters* recuperados de acordo com o valor do n_r para o conjunto real `golub`.

O número de partições usadas para compor o C_C é o valor limite para o parâmetro n_r selecionar algum *cluster*, já que um mesmo *cluster* não aparece mais de uma vez numa mesma partição. Não há uma relação entre o número de partições e uma boa escolha para o valor do parâmetro n_r. O valor 2 para esse parâmetro garante que todos os *clusters* que aparecem mais de uma vez sejam adicionados no conjunto reduzido. Os resultados para os conjuntos de dados de ajuste com o $n_r = 2$ têm se mostrado bons (ver Apêndice A). Nos próximos experimentos, o valor 2 será usado para esse parâmetro.

4.7 Considerações Finais

Foi investigado o comportamento do ASA_{Clu} e do passo de inicialização quando aplicado a multiconjuntos completos de *clusters* produzidos usando diferentes combinações de algoritmos de agrupamento. Foi avaliado o número de *clusters* selecionados em C_R e C_I junto com o número de *clusters* reais recuperados em C_R e C_I. Também, foi apresentado o procedimento usado para ajustar o critério de parada usado no código do ASA_{Clu}, levando em consideração a diferença na taxa de redução e o número de *clusters* reais recuperados em C_R. Para tal, foram utilizados 10 conjuntos de dados de ajuste. A combinação de

algoritmos $KM + SL$ se mostrou eficiente ao manter um bom número de *clusters* reais recuperados enquanto selecionou o menor número de *clusters*, representando a opção mais barata para produzir C_C, em termos de número de algoritmos.

Por fim, foi investigado o comportamento do algoritmo ASA_{Clu} e do passo de inicialização ao variar o parâmetro n_r, com o objetivo de avaliar a suposição de que um *cluster* relevante aparece mais vezes no multiconjunto de *clusters* C_C. Essa suposição se mostrou verdadeira para a maioria dos conjuntos de dados artificiais (exceto para `twoDiamonds`), sendo que para alguns, somente essa suposição foi o suficiente para recuperar os *clusters* de qualidade (`ds2c2sc13` e `spiralsquare`). Já os *clusters* reais não tiveram esse mesmo comportamento em relação a suposição, somente um conjunto de dados reais (`miRNAcancer`) se mostrou aderente a essa suposição. Não foi encontrado nenhum padrão mais apropriado para a escolha do valor do parâmetro n_r, e os resultados para os conjuntos de dados de ajuste com o $n_r = 2$ têm se mostrado bons (ver Apêndice A). Sendo assim, nos próximos experimentos, o valor 2 será usado para esse parâmetro.

5 Resultados

5.1 Considerações Iniciais

Para validar o potencial apresentado pelo ASA_{Clu} no capítulo anterior com a configuração escolhida, na Seção 5.2 será apresentado um novo experimento com uma coleção independente de conjuntos de dados, da Tabela 1. Para tal, foram utilizados 27 conjuntos de dados independentes apresentados no Capítulo 4, não utilizados como conjuntos de dados de ajuste. Nesse experimento, o C_C do ASA_{Clu} e do $MBCS$ e o Π_C do ASA foram gerados com a aplicação dos algoritmos de agrupamento KM e SL. A escolha dessa combinação de algoritmos foi justificada no Capítulo 4. Foi atribuído o valor 2 para o parâmetro p do ASA, que representa o número de algoritmos utilizados para formar Π_I, em (SAKATA et al., 2010). Foi utilizado o valor 2 para o parâmetro n_r do ASA_{Clu}. A escolha deste valor do n_r foi justificada na Seção 4.6. Como dito anteriormente, a técnica $MBCS$ equivale ao passo de inicialização do ASA_{Clu} com o valor 2 para o parâmetro n_r. Os resultados do ASA_{Clu} foram comparados com os resultados do ASA e do $MBCS$. Os resultados foram comparados em relação a qualidade e quantidade dos *clusters* obtidos. A metodologia para avaliação da qualidade dos *clusters* obtidos foi descrita no Cápitulo 4, junto com o procedimento para a aplicação e escolha dos algoritmos KM e SL.

Na Seção 5.3, será apresentado outro experimento para a comparação do ASA_{Clu} com o ASA. Só que dessa vez, utilizando a configuração do conjunto de algoritmos validado para o ASA, em (SAKATA et al., 2010). Essa configuração utiliza 4 algoritmos KM, SL, AL e SNN, e foi responsável por determinar o valor do critério de parada do ASA de 0.12. O parâmetro p do ASA foi 4, pois isso representa o número de algoritmos iniciais. O parâmetro n_r do ASA_{Clu} foi 2, conforme descrito na Seção 4.6. Os mesmos 27 conjuntos de dados independentes (Tabela 1) foram utilizados. Sendo assim, os experimentos da Seção 5.2 favorecem a configuração escolhida para o ASA_{Clu} e os experimentos da Seção 5.3 favorecem a configuração escolhida para o ASA. O principal propósito dessas comparações, é validar as vantagens em selecionar *clusters* em vez de partições. Já, que o ASA_{Clu} é uma adaptação do ASA para a seleção de *clusters*.

5.2 Comparação do ASA_{Clu} com o $MBCS$ e o ASA, com a configuração do conjunto de algoritmos validado para o ASA_{Clu}

A Figura 13 apresenta a porcentagem de *clusters* de C_C selecionados pelo ASA_{Clu}, $MBCS$ e ASA. O $MBCS$ conseguiu uma redução melhor em todos os casos. Em 8 casos,

ele selecionou menos que 12% dos *clusters* de C_C, e em apenas um caso `voice3`, ele selecionou mais que 24%. Já, o ASA_{Clu} complementou C_I com mais *clusters*, aumentando seu conjunto reduzido C_R. Em 4 casos, o ASA_{Clu} selecionou 72% dos *clusters*, e em 10 casos, selecionou menos que 36%. O ASA não conseguiu reduzir o C_C em 7 casos, e em 4 casos, selecionou menos que 24%. Em um caso (`iris`), o ASA_{Clu} obteve a mesma redução que o ASA (42,5%). Em 11 casos, o ASA selecionou menos *clusters* que o ASA_{Clu}, e em 15 casos o ASA_{Clu} selecionou menos *clusters* que o ASA. No quesito redução do multiconjunto C_C, o $MBCS$ obteve o melhor resultado e o ASA_{Clu} teve uma ligeira vantagem em relação ao ASA.

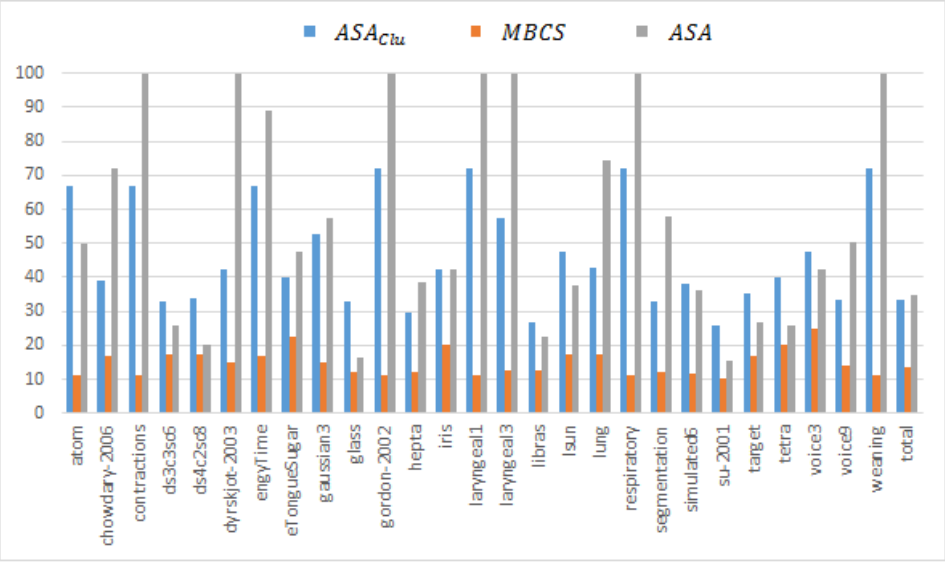

Figura 13 – Porcentagem de *clusters* de C_C selecionados pelo ASA_{Clu}, $MBCS$ e ASA.

A Figura 14 apresenta a porcentagem de *clusters* parcialmente recuperados (PR) de C_C pelo ASA_{Clu}, $MBCS$ e ASA. O ASA_{Clu} conseguiu recuperar parcialmente todos os *clusters* de C_C em todos os casos, exceto para os casos em que o C_C não teve nenhum *cluster* parcialmente recuperado (`chowdary-2006`, `laryngeal1`, `respiratory`, `voice3` e `weaning`). O $MBCS$ conseguiu recuperar todos os *clusters* em 9 casos. Em 6 casos não conseguiu recuperar nenhum, não contando os casos em que não há *clusters* recuperados em C_C. O ASA conseguiu recuperar todos os *clusters* em quase todos os casos, exceto para `ds3c3sc6` e `su-2001`. O caso em que ele menos recuperou *clusters* foi para `su-2001` com 75%. No quesito de recuperar *clusters* parcialmente do multiconjunto C_C, o ASA_{Clu} obteve o melhor resultado, conseguindo recuperar parcialmente todos os *clusters* de C_C. O ASA só deixou de recuperar todos os *clusters* em 2 casos, mas conseguiu recuperar uma boa parte, 75% no pior caso. O $MBCS$ foi o que menos recuperou *clusters*, isso mostra que o ASA_{Clu} conseguiu complementar C_I com *clusters* de qualidade.

A Figura 15 apresenta a porcentagem de *clusters* integralmente recuperados (CR) de C_C pelo ASA_{Clu}, $MBCS$ e ASA. Em 15 casos, o C_C não obteve nenhum cluster

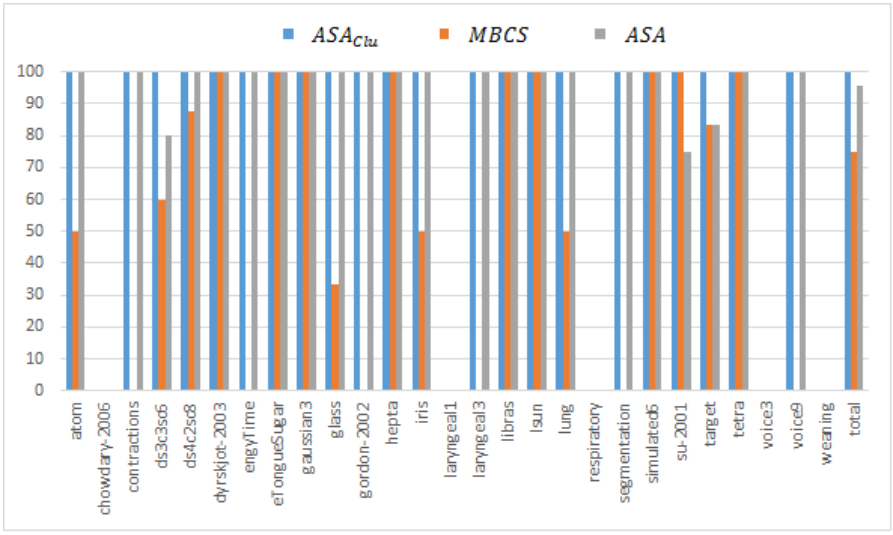

Figura 14 – Porcentagem de *clusters* parcialmente recuperados de C_C pelo ASA_{Clu}, $MBCS$ e ASA.

integralmente recuperado. Sendo assim, nenhum algoritmo conseguiu recuperar *clusters* (barras do gráfico da Figura 15 em branco). Nos demais casos, os 3 algoritmos conseguiram recuperar integralmente uma boa margem de *clusters*, exceto para `atom` em que essa margem foi de 50% para os 3 algoritmos. Em 8 casos, os 3 algoritmos obtiveram o mesmo resultado. Em apenas 2 casos (`target` e `tetra`), eles obtiveram resultados diferentes. Para o `target`, o ASA_{Clu} conseguiu recuperar 100% dos *clusters* contra 83,33% do $MBCS$ e do ASA. Para o `tetra`, o ASA conseguiu recuperar 100% dos *clusters* contra 75% do ASA_{Clu} e do $MBCS$. No geral, o ASA e o ASA_{Clu} obtiveram o mesmo resultado de recuperar 93,33% dos *clusters* de todos os conjuntos de dados. O $MBCS$ teve uma ligeira desvantagem em recuperar 90% de todos os *clusters*.

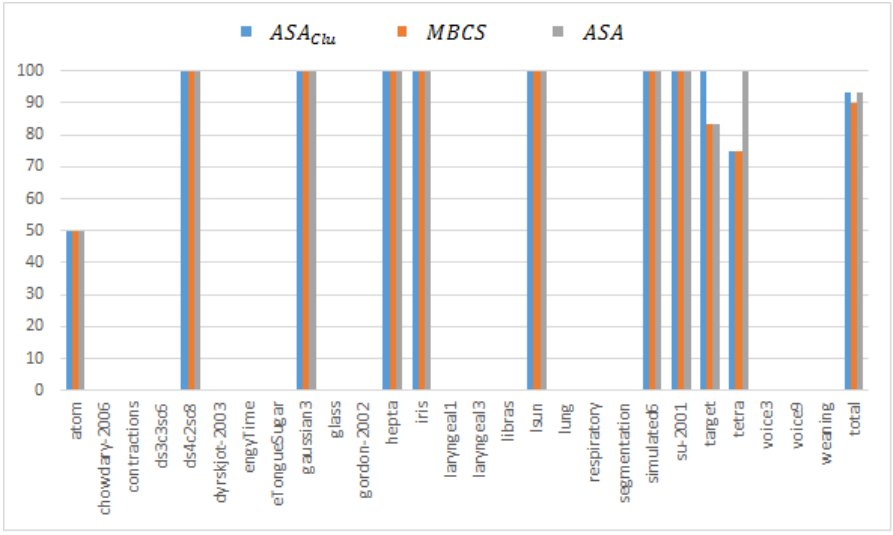

Figura 15 – Porcentagem de *clusters* integralmente recuperados de C_C pelo ASA_{Clu}, $MBCS$ e ASA.

5.3 Comparação do ASA_{Clu} com o ASA, com a configuração do conjunto de algoritmos validado para o ASA

A Figura 16 apresenta a porcentagem de *clusters* selecionados de C_C pelo ASA e ASA_{Clu}. Em todos os casos, o ASA_{Clu} selecionou menos que 60% dos *clusters*, e em 2 casos `hepta` e `su-2001`, selecionou menos que 20% dos *clusters*. O ASA não conseguiu reduzir o C_C em um único caso `contractions`, e em 4 casos `eTongueSugar`, `libras`, `su-2001` e `voice3`, selecionou menos que 20% dos *clusters*. Em 9 casos, o ASA selecionou menos *clusters* que o ASA_{Clu}, e em 18 casos o ASA_{Clu} selecionou menos *clusters* que o ASA. No total, o ASA_{Clu} obteve uma melhor redução, selecionando 27,19% dos *clusters* de C_C contra 30,49% selecionados pelo ASA.

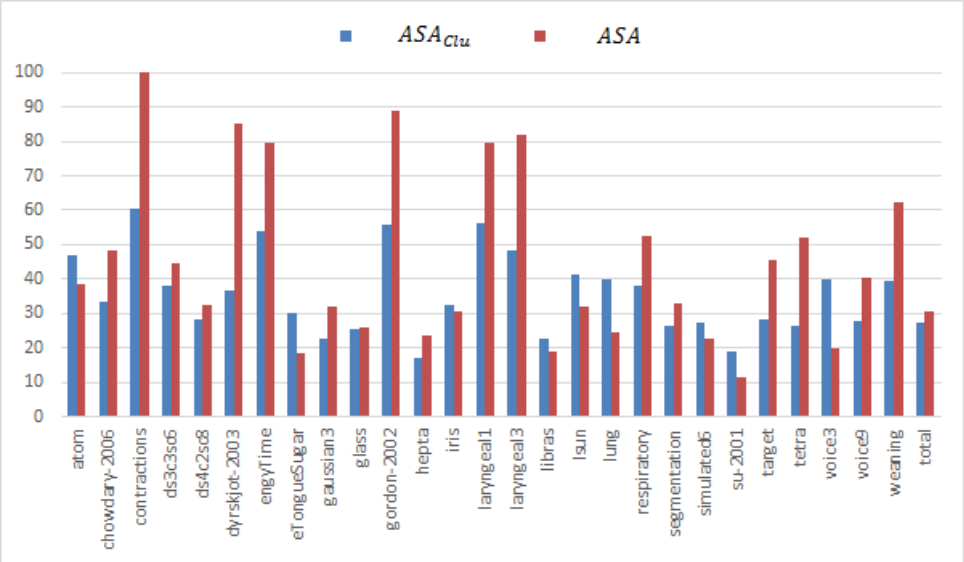

Figura 16 – Porcentagem de *clusters* selecionados de C_C pelo ASA e ASA_{Clu}.

A Figura 17 apresenta a porcentagem de *clusters* parcialmente recuperados de C_C pelo ASA e ASA_{Clu}. O ASA_{Clu} conseguiu recuperar todos os *clusters* parcialmente recuperados em quase todos os casos, exceto para `iris`. Em alguns casos, o C_C não teve nenhum *cluster* parcialmente recuperado (`chowdary-2006`, `laryngeal1`, `respiratory` e `weaning`). Sendo assim, não foi possível recuperar nenhum *cluster* pelo ASA ou ASA_{Clu}, e esses casos não foram contabilizados. O ASA conseguiu recuperar todos os *clusters* em quase todos os casos, exceto para `ds3c3sc6` e `lung`. O ASA conseguiu recuperar mais que 70% dos *clusters* em todos os casos, e o ASA_{Clu} conseguiu recuperar mais que 60% dos *clusters* em todos os casos. No total, o ASA_{Clu} conseguiu recuperar mais *clusters*, 98,57% contra 97,14%.

A Figura 18 apresenta a porcentagem de *clusters* integralmente recuperados de C_C pelo ASA e ASA_{Clu}. O ASA_{Clu} conseguiu recuperar integralmente todos os *clusters* de C_C. Os casos em que o ASA_{Clu} não recuperou nenhum *cluster*, são os casos que não

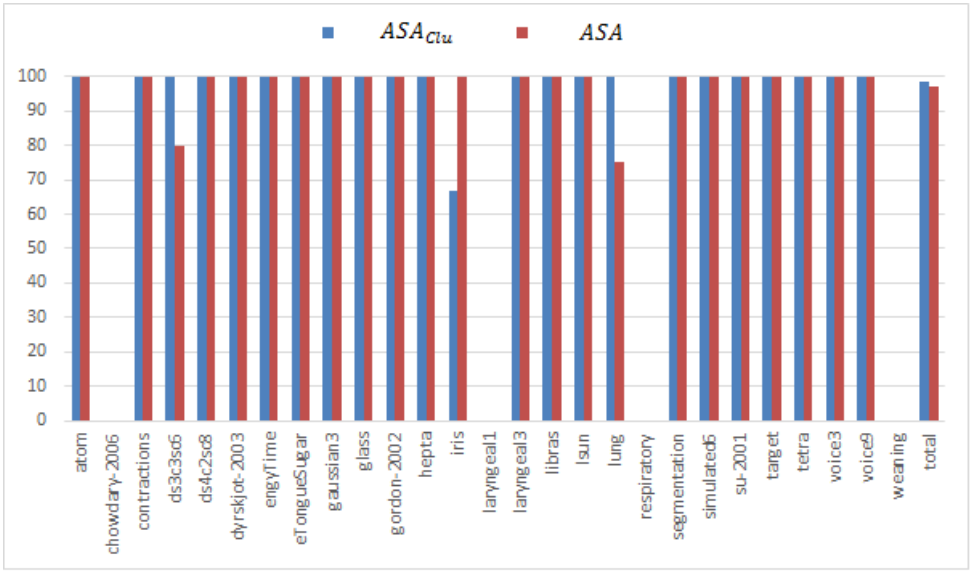

Figura 17 – Porcentagem de *clusters* parcialmente recuperados de C_C pelo ASA e ASA_{Clu}.

há *clusters* em C_C. Por isso não foram contabilizados. O ASA não conseguiu recuperar nenhum *cluster* em `su-2001`, e nos demais casos conseguiu recuperar todos os *clusters*. No total, o ASA_{Clu} conseguiu recuperar mais *clusters*, 100% contra 96,87%.

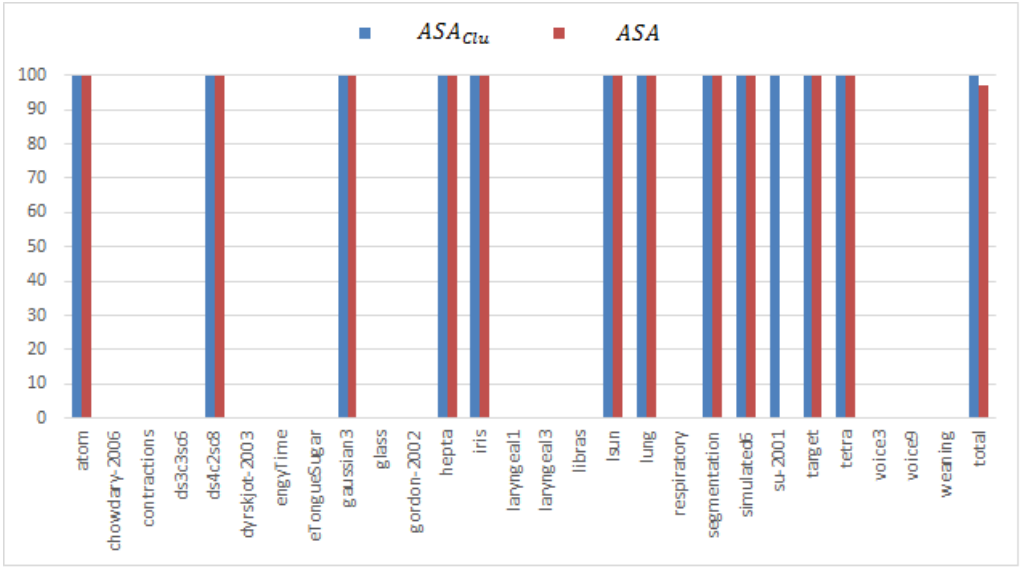

Figura 18 – Porcentagem de *clusters* integralmente recuperados de C_C pelo ASA e ASA_{Clu}.

5.4 Considerações Finais

Esses experimentos tiveram o objetivo de avaliar os resultados do ASA_{Clu} em relação ao ASA e ao $MBCS$, no aspecto de reduzir o multiconjunto C_C, mantendo os *clusters* de qualidade. No quesito redução do multiconjunto, o $MBCS$ obteve o melhor resultado. No quesito de manter os *clusters* de qualidade, o ASA_{Clu} obteve o melhor

resultado. O ASA_{Clu}, apesar de ter um desempenho inferior ao $MBCS$ em reduzir o multiconjunto, ele se mostrou superior ao manter os *clusters* de qualidade, validando assim a sua eficiência em complementar o conjunto C_I com *clusters* de qualidade. O ASA foi comparado ao ASA_{Clu} em dois cenários: um com a configuração escolhida para o ASA_{Clu} e o outro com a configuração escolhida para o ASA. Nos dois cenários, o ASA_{Clu} obteve o melhor desempenho no aspecto de redução e qualidade, validando assim a eficiência em selecionar *clusters* em vez de partições.

6 Conclusão

Nesse trabalho de pesquisa foi demonstrada a técnica de seleção de *clusters* ASA_{Clu}, que a partir de um multiconjunto de *clusters* C_C produzido por algoritmos de agrupamento, seleciona um conjunto reduzido de *clusters* diversos e relevantes (C_R). O objetivo principal desse trabalho foi fazer uma adaptação do ASA para a seleção de *clusters*, e demonstrar as principais vantagens em selecionar *clusters* em vez de partições. O seu princípio de funcionamento também foi motivado por outras técnicas também descritas nesse trabalho ($MBCS$ e MOC).

O Capítulo 4 foi dedicado aos experimentos de ajuste do algoritmo ASA_{Clu} e testes de suas principais suposições. Foram analisados os resultados do ASA_{Clu} e do passo de inicialização quando aplicados a multiconjuntos de *clusters* produzidos usando diferentes combinações de algoritmos de agrupamento. A combinação de algoritmos $KM + SL$ se mostrou eficiente ao manter um bom número de *clusters* reais recuperados enquanto selecionou o menor número de *clusters*, representando a opção mais barata para produzir C_C, em termos de número de algoritmos. Também, foi apresentado o procedimento usado para ajustar o critério de parada usado no código do ASA_{Clu}, levando em consideração a diferença na taxa de redução. Foi definido o valor de 0.13 para o critério de parada, pois não foi constatada nenhuma redução expressiva após a diferença nas taxas atingirem esse valor.

Ainda, no Capítulo 4 foi avaliada a suposição de que um *cluster* relevante aparece mais vezes no multiconjunto de *clusters* C_C. Para isso, foi contabilizado o número de vezes que os *clusters* reais foram encontrados em C_C. Essa suposição se mostrou verdadeira para a maioria dos conjuntos de dados artificiais (exceto para `twoDiamonds`), sendo que para alguns, somente essa suposição foi o suficiente para recuperar os *clusters* de qualidade (`ds2c2sc13` e `spiralsquare`). Já os *clusters* reais não tiveram esse mesmo comportamento em relação a suposição, e somente um conjunto de dados reais (`miRNAcancer`) se mostrou aderente a essa suposição. Por fim, não foi encontrado nenhum padrão mais apropriado para a escolha do valor do parâmetro n_r, e os resultados para os conjuntos de dados de ajuste com o $n_r = 2$ mostraram-se bons (ver Apêndice A). Sendo assim, foi determinado o valor 2 para esse parâmetro, nos experimentos do Capítulo 5.

No Capítulo 5 foi possível comparar os resultados do ASA_{Clu} com os resultados do ASA e do $MBCS$, possibilitando assim, demonstrar as vantagens dessa nova abordagem. No quesito redução do multiconjunto, o $MBCS$ obteve o melhor resultado. No quesito de manter os *clusters* de qualidade, o ASA_{Clu} obteve o melhor resultado. O ASA_{Clu}, apesar de ter um desempenho inferior ao $MBCS$ em reduzir o multiconjunto, ele se mostrou superior

ao manter os *clusters* de qualidade, validando assim a sua eficiência em complementar o conjunto C_I com *clusters* de qualidade. O ASA foi comparado ao ASA_{Clu} em dois cenários: um com a configuração do conjunto de algoritmos validada para o ASA_{Clu} e o outro com a configuração do conjunto de algoritmos validada para o ASA ($KM + SL$ e $KM + SL + SNN + AL$, respectivamente). Nos dois cenários, o ASA_{Clu} obteve o melhor desempenho no aspecto de redução e qualidade, validando assim a eficiência em selecionar *clusters* em vez de partições.

Foi publicado um artigo no congresso Bracis 2016, que demonstra o algoritmo ASA_{Clu} e alguns resultados preliminares da avaliação e configuração do algoritmo (ALMEIDA; SAKATA; FACELI, 2016). Um site do projeto do ASA_{Clu} foi criado para outros pesquisadores usarem o algoritmo, disponível em <http://lasid.sor.ufscar.br/asacluproject/>. Os principais resultados descritos nesse artigo foram os ajustes das configurações do algoritmo, como o conjunto ótimo de algortimos de agrupamento, o valor do critério de parada, e a avaliação da habilidade do ASA_{Clu} em complementar o conjunto inicial C_I com *clusters* de qualidade. O principal propósito desse artigo foi mostrar o potencial da técnica, em selecionar *clusters*, dado um multiconjunto inicial gerado pela aplicação de diversos algoritmos de agrupamento.

Ainda, no Capítulo 4 foi investigado o conjunto ótimo de algoritmos de agrupamento para formar o multiconjunto C_C. Esse conjunto de algoritmos é responsável por alimentar o ASA_{Clu}, e desempenha um papel muito importante na qualidade dos *clusters* obtidos, já que o ASA_{Clu} somente seleciona os *clusters* gerados por esses algoritmos. Como trabalho futuro, um estudo mais aprofundado dos algoritmos que compõem o C_C se faz necessário, para mostrar todo o potencial dessa técnica de seleção de *clusters*. Uma possibilidade dentro desse estudo dos algoritmos que compõem o C_C, é investigar outras medidas de distância, como a distância de Pearson. Além disso, investigar algoritmos que encontrem *clusters* em sub-espaços do espaço de atributos original. Por fim, um outro trabalho futuro é o de um estudo de métodos de visualização de *clusters*, para facilitar a interpretação do conjunto reduzido obtido pelo ASA_{Clu}.

Referências

AHA, D. et al. Uci repository of machine learning databases. university of california, irvine, dept. of information and computer sciences. apr 2015. Disponível em: <http://archive.ics.uci.edu/ml/>. Citado na página 20.

ALMEIDA, J. L. B. de; SAKATA, T. C.; FACELI, K. Asaclu: Selecting diverse and relevant clusters. In: *5th Brazilian Conference on Intelligent Systems*. Pernambuco: [s.n.], 2016. p. 474–479. Citado na página 40.

ARMSTRONG, S. A. et al. MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nature Genetics*, v. 30, n. 1, p. 41–47, 2002. Citado na página 20.

BHATTACHARJEE, A. et al. Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma sub-classes. In: *Proc. Natl. Acad. Sci. USA*. Boston: [s.n.], 2001. v. 98, n. 24, p. 13790–13795. Citado na página 20.

CHOWDARY, D. et al. Prognostic gene expression signatures can be measured in tissues collected in RNAlater preservative. *Journal of Molecular Diagnostics*, v. 8, n. 1, p. 31–39, 2006. Citado na página 20.

DYRSKJØT, L. et al. Identifying distinct classes of bladder carcinoma using microarrays. *Nature Genetics*, v. 33, n. 1, p. 90–96, 2003. Citado na página 20.

ERTOZ, L.; STEINBACH, M.; KUMAR, V. A new shared nearest neighbor clustering algorithm and its applications. In: *Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining*. [S.l.: s.n.], 2002. Citado na página 23.

FACELI, K. *Um framework para análise de agrupamento baseado na combinação multi-objetivo de algoritmos de agrupamento*. Tese (Doutorado) — USP – São Carlos, 2007. Citado 2 vezes nas páginas xvii e 22.

FACELI, K. et al. *Inteligência Artificial: Uma abordagem de aprendizado de máquina*. [S.l.]: LTC, 2011. Citado 3 vezes nas páginas xvii, 2 e 3.

FACELI, K.; SAKATA, T. *Multiple solutions in cluster analysis: partitions x clusters*. Sorocaba - SP - Brazil, 2016. Citado 11 vezes nas páginas xvii, xix, 3, 8, 9, 13, 19, 20, 21, 22 e 23.

FACELI, K. et al. Partitions selection strategy for set of clustering solutions. In: *Neurocomputing*. [S.l.: s.n.], 2010. v. 73, n. 16–18, p. 2809–2819. Citado na página 6.

FRANK, E.; HALL, M. A.; WITTEN, I. H. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. [S.l.]: Morgan Kaufmann, Fourth Edition, 2016. Citado na página 23.

GOLUB, T. R. et al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, v. 286, n. 5439, p. 531–537, 1999. Citado na página 20.

GORDON, G. J. et al. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Research*, v. 62, n. 17, p. 4963–4967, 2002. Citado na página 20.

HANDL, J.; KNOWLES, J. *Multiobjective clustering with automatic determination of the number of clusters*. Manchester., 2004. Citado na página 22.

HANDL, J.; KNOWLES, J.; KELL, D. B. Computational cluster validation in post-genomic data analysis. *Bioinformatics*, v. 21, n. 15, p. 3201–3212, maio 2005. Citado na página 1.

HUBERT, L. J.; ARABIE, P. Comparing partitions. *Journal of Classification*, v. 2, p. 193–218, 1985. Citado na página 5.

JACCARD, P. Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. *Bulletin de la Société Vaudoise des Sciences Naturelles*, v. 37, n. 140, p. 241–272, jan. 1901. Citado na página 13.

JAIN, A.; DUBES, R. *Algorithms for Clustering Data*. [S.l.]: Prentice Hall, 1988. Citado na página 23.

JAIN, A. K. Data clustering: 50 years beyond k-means. *International Conference on Pattern Recognition (ICPR)*, v. 19, dez. 2008. Citado na página 1.

JIAMTHAPTHAKSIN, R.; EICK, C. F.; VILALTA, R. A framework for multi-objective clustering and its application to co-location mining. In: *Lecture Notes in Computer Science*. [S.l.: s.n.], 2009. v. 5678, p. 188–199. Citado 3 vezes nas páginas 3, 4 e 9.

KUNCHEVA, L. et al. Real medical data sets. apr 2015. Disponível em: <http://pages.bangor.ac.uk/~mas00a/activities/real_data.htm>. Citado na página 20.

LU, J. et al. MicroRNA expression profiles classify human cancers. *Nature*, v. 435, p. 834–838, 2005. Citado na página 20.

MONTI, S. et al. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. abr. 2015. Disponível em: <http://www.broadinstitute.org/cgi-bin/cancer/publications/view/87>. Citado na página 20.

MOON, T. K. The expectation-maximization algorithm. *IEEE Signal Processing Magazine*, v. 13, n. 6, p. 47–60, nov. 1996. Citado na página 23.

MüLLER, E. et al. Multiclust special issue on discovering, summarizing and using multiple clusterings. *Machine Learning*, v. 98, n. 1-2, p. 1–5, maio 2015. Citado na página 1.

MüLLER, E. et al. Discovering multiple clustering solutions: Grouping objects in different views of the data. *2012 IEEE 28th International Conference on Data Engineering*, v. 21, n. 15, p. 1207–1210, april 2012. Citado 4 vezes nas páginas xvii, 1, 2 e 5.

SAKATA, T. et al. Improvements in the partitions selection strategy for set of clustering solutions. In: *Proceedings of the 11th Brazilian Symposium on Neural Networks, (SBRN'2010)*. [S.l.: s.n.], 2010. p. 49–54. Citado 7 vezes nas páginas 3, 5, 6, 7, 16, 28 e 33.

SU, A. I. et al. Molecular classification of human carcinomas by use of gene expression signatures. *Cancer Research*, v. 61, n. 20, p. 7388–7393, 2001. Citado na página 20.

ULTSCH, A. et al. Fundamental clustering problems suite. abr. 2015. Disponível em: <http://www.uni-marburg.de/fb12/datenbionik/data?language_sync=1>. Citado 3 vezes nas páginas xvii, 20 e 21.

XU, R.; WUNSCH, D. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, v. 16, n. 3, p. 645–678, maio 2005. Citado na página 1.

YEOH, E.-J. et al. Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell*, v. 1, n. 2, p. 133–143, 2002. Citado na página 20.

APÊNDICE A – Tabelas da variação do parâmetro n_r

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	27	3	0	27	3	0
2	21	3	0	7	1	0
3	21	3	0	3	0	0
4	21	3	0	2	0	0
5	21	3	0	1	0	0
6	21	3	0	0	0	0

Tabela 8 – Variação do parâmetro n_r para o conjunto `armstrong-2002`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	239	19	19	239	19	19
2	199	19	19	101	19	19
3	190	19	19	63	18	18
4	188	19	19	45	16	16
5	186	19	18	36	16	15
6	186	19	19	30	15	14
7	185	19	19	28	15	13
8	185	19	19	25	15	13
9	185	19	19	21	13	12
10	185	19	19	19	12	11
11	185	19	19	15	12	11
12	185	19	19	14	11	10
13	185	19	19	13	10	9
14	184	19	19	10	10	9
15	184	19	19	10	10	9
16	184	19	19	8	8	8
17	184	19	18	6	6	6
18	184	19	18	4	4	4
19	184	19	18	4	4	4
20	184	19	18	4	4	4
21	184	19	18	1	1	1
22	184	19	18	1	1	1
23	184	19	18	1	1	1
24	184	19	18	1	1	1
25	184	19	18	1	1	1
26	184	19	18	0	0	0

Tabela 9 – Variação do parâmetro n_r para o conjunto `ds2c2sc13`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	43	7	0	43	7	0
2	35	7	0	9	2	0
3	35	7	0	7	1	0
4	35	7	0	5	1	0
5	35	7	0	3	0	0
6	35	7	0	2	0	0
7	35	7	0	1	0	0
8	35	7	0	0	0	0

Tabela 10 – Variação do parâmetro n_r para o conjunto `golub`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	15	1	0	15	1	0
2	13	1	0	2	0	0
3	13	1	0	1	0	0
4	13	1	0	0	0	0

Tabela 11 – Variação do parâmetro n_r para o conjunto `laryngeal2`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	150	12	10	150	12	10
2	101	12	10	32	7	7
3	100	12	10	15	6	6
4	99	12	10	13	6	6
5	99	12	10	10	6	6
6	99	12	10	9	6	6
7	99	12	10	7	6	6
8	99	12	10	6	6	6
9	99	12	10	6	6	6
10	99	12	10	6	6	6
11	99	12	10	6	6	6
12	98	12	9	5	5	5
13	98	12	9	3	3	3
14	98	12	9	2	2	2
15	98	12	9	0	0	0

Tabela 12 – Variação do parâmetro n_r para o conjunto `monkey`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	73	8	4	73	8	4
2	50	8	4	22	8	4
3	47	8	3	16	7	3
4	47	8	3	13	5	3
5	47	8	3	9	4	3
6	47	8	3	8	4	3
7	47	8	3	5	1	1
8	47	8	3	4	1	1
9	47	8	3	3	1	1
10	47	8	3	1	0	0
11	47	8	4	0	0	0

Tabela 13 – Variação do parâmetro n_r para o conjunto `spiralsquare`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	14	2	2	14	2	2
2	10	2	2	3	0	0
3	10	2	2	1	0	0
4	10	2	2	0	0	0

Tabela 14 – Variação do parâmetro n_r para o conjunto `twoDiamonds`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	15	2	2	15	2	2
2	9	2	1	3	2	1
3	9	2	2	0	0	0

Tabela 15 – Variação do parâmetro n_r para o conjunto `wingNut`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	89	2	0	89	2	0
2	68	2	0	12	0	0
3	68	2	0	9	0	0
4	68	2	0	8	0	0
5	68	2	0	7	0	0
6	68	2	0	6	0	0
7	68	2	0	5	0	0
8	68	2	0	4	0	0
9	68	2	0	3	0	0
10	68	2	0	3	0	0
11	68	2	0	3	0	0
12	68	2	0	2	0	0
13	68	2	0	1	0	0
14	68	2	0	1	0	0
15	68	2	0	1	0	0
16	68	2	0	0	0	0

Tabela 16 – Variação do parâmetro n_r para o conjunto `yeoh-2002-v1`.

n_r	$\|C_R\|$	PR_R	CR_R	$\|C_I\|$	PR_I	CR_I
1	540	21	10	540	21	10
2	463	21	10	132	15	10
3	457	21	10	82	13	9
4	452	21	10	64	10	8
5	449	21	10	56	10	8
6	449	21	10	47	10	8
7	449	21	10	43	10	8
8	451	21	10	41	9	8
9	452	21	10	38	8	7
10	452	21	10	34	8	7
11	452	21	10	32	8	7
12	452	21	10	29	7	7
13	452	21	10	27	7	7
14	452	21	10	24	6	6
15	452	21	10	22	6	6
16	452	21	10	22	6	6
17	452	21	10	22	6	6
18	452	21	10	21	6	6
19	452	21	10	21	6	6
20	452	21	10	20	6	6
21	452	21	10	20	6	6
22	452	21	10	18	6	6
23	452	21	10	18	6	6
24	452	21	10	18	6	6
25	452	21	10	18	6	6
26	452	21	10	15	6	6
27	452	21	10	14	6	6
28	452	21	10	14	6	6
29	452	21	10	13	6	6
30	452	21	10	13	6	6
31	452	21	10	12	6	6
32	452	21	10	12	6	6
33	452	21	10	11	6	6
34	452	21	10	9	5	5
35	452	21	10	8	5	5
36	453	21	10	6	3	3
37	453	21	10	6	3	3
38	453	21	10	6	3	3
39	453	21	10	5	3	3
40	453	21	10	5	3	3
41	453	21	10	5	3	3
42	453	21	10	5	3	3
43	453	21	10	4	3	3
44	453	21	10	4	3	3
45	453	21	10	3	3	3
46	453	21	10	3	3	3
47	453	21	10	3	3	3
48	453	21	10	3	3	3
49	453	21	10	3	3	3
50	453	21	10	3	3	3
51	453	21	10	3	3	3
52	453	21	10	3	3	3
53	453	21	10	3	3	3
54	453	21	10	3	3	3
55	453	21	10	3	3	3
56	453	21	10	3	3	3
57	453	21	10	3	3	3
58	453	21	10	3	3	3
59	453	21	10	3	3	3
60	453	21	10	3	3	3
61	453	21	10	3	3	3
62	453	21	10	3	3	3
63	453	21	10	2	2	2
64	453	21	10	1	1	1
65	453	21	10	1	1	1
66	453	21	10	1	1	1
67	453	21	10	1	1	1
68	453	21	10	1	1	1
69	453	21	10	1	1	1
70	453	21	10	0	0	0

Tabela 17 – Variação do parâmetro n_r para o conjunto `miRNAcancer`.