Vanessa Antunes

Estratégia Híbrida de Seleção de Partições para o Problema de Agrupamento de Dados

Sorocaba, SP

26 de Janeiro de 2018

Vanessa Antunes

Estratégia Híbrida de Seleção de Partições para o Problema de Agrupamento de Dados

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: Computação Científica e Inteligência Computacional.

Universidade Federal de São Carlos – UFSCar

Centro de Ciências em Gestão e Tecnologia – CCGT

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

Orientadora: Profa. Dra. Tiemi Christine Sakata

Sorocaba, SP

26 de Janeiro de 2018

Antunes, Vanessa

Estratégia Híbrida de Seleção de Partições para o Problema de Agrupamento de Dados / Vanessa Antunes. -- 2018.
61 f. : 30 cm.

Dissertação (mestrado)-Universidade Federal de São Carlos, campus Sorocaba, Sorocaba
Orientador: Profa. Dra. Tiemi Christine Sakata
Banca examinadora: Profa. Dra. Tiemi Christine Sakata, Profa. Dra. Ana Lucia Cetertich Bazzan, Prof. Dr. Tiago Agostinho de Almeida
Bibliografia

1. Agrupamento multiobjetivo. 2. Pareto-otimalidade. 3. Seleção multiobjetivo. 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 da candidata Vanessa Antunes, realizada em 26/01/2018:

Profa. Dra. Tiemi Christine Sakata
UFSCar

Profa. Dra. Ana Lúcia Cetertich Bazzan
UFRGS

Prof. Dr. Tiago Agostinho de Almeida
UFSCar

Dedico este trabalho à minha família pelo apoio e incentivo que sempre recebi para avançar em meus estudos.

Agradecimentos

A Deus por sempre guiar minha vida e ter tornado possível a realização deste projeto.

À minha orientadora, professora Tiemi, por sua dedicação e paciência comigo.

Aos professores Marcílio de Souto e Katti Faceli pelas discussões e contribuições para a realização deste trabalho.

Aos colegas de laboratório pela companhia e apoio durante esta jornada.

À CAPES pelo apoio financeiro.

Ao Programa Pós-graduação em Ciência da Computação da UFSCar Sorocaba.

"Já que é pra sonhar, sonhe com algo impossível. Se fosse pra querer coisas fáceis, então onde ficaria a graça? Quanto mais difícil maior será a vontade de tornar real. A vida é isso, contrariar o incontrariável." (Carolina Patrocínio)

Resumo

Inaptidão para identificar partições de diferentes tamanhos e formas é uma limitação fundamental de qualquer algoritmo de agrupamento, especialmente quando diferentes regiões do espaço de busca contêm *clusters* com características distintas. A aplicação de diferentes algoritmos de agrupamento, com diferentes parâmetros, é uma possibilidade, porém, neste caso, é necessário lidar com um grande conjunto de partições. Técnicas como *ensemble* e agrupamento multiobjetivo empregam diferentes abordagens para tratar este problema, porém ambas possuem um custo computacional elevado. Além disso, as técnicas de *ensemble* geram uma única solução, que pode não representar toda partição real presente no conjunto de dados. O agrupamento multiobjetivo, por outro lado, pode gerar um conjunto grande de partições, inviável de ser analisado manualmente. Nesta dissertação, é proposto um algoritmo multiobjetivo híbrido, *HSS* (*Hybrid Selection Strategy*), cujo objetivo é retornar um conjunto reduzido e ao mesmo tempo diverso de partições. Ele pode ser dividido em três passos: (i) aplicação de um algoritmo multiobjetivo em um conjunto base de partições para a geração de uma aproximação da Fronteira de Pareto (FP), (ii) divisão das soluções presentes na aproximação da FP em um certo número de regiões e (iii) seleção de uma partição por região através da aplicação do *Adjusted Rand Index* (*ARI*). Experimentos mostram a eficácia do *HSS* na seleção de um número reduzido de partições.

Palavras-chaves: Agrupamento multiobjetivo. Pareto-otimalidade. Seleção multiobjetivo.

Abstract

Inability to identify partitions of different sizes and shapes is a fundamental limitation of any clustering algorithm, especially when different regions of the search space contain clusters with varied characteristics. It is possible to apply diverse clustering algorithms, with different parameters, but then, it is necessary to deal with a large number of partitions. Techniques such as ensemble and multiobjective clustering treat this problem using distinct criteria, but they have high computational cost. Moreover, the ensemble technique generates a single solution, which may not represent every real partition present in the data. On the other hand, multiobjective clustering may generate a large number of partitions, which is difficult to analyze manually. In this dissertation, we propose a hybrid multiojective algorithm, *HSS* (Hybrid Selection Strategy), that aims to return a reduced and yet diverse set of solutions. It can be divided in three steps: (i) the application of a multiobjective algorithm to a set of base partitions for the generation of an approximation of the Pareto Front, (ii) the division of the solutions from the approximation of the Pareto Front into a certain number of regions and (iii) the selection of a solution per region, through the application of the Adjusted Rand Index. Experiments show the effectiveness of *HSS* in selecting a reduced number of partitions.

Key-words: Multiobjective clustering. Pareto-optimality. Multiobjective selection.

Lista de ilustrações

Figura 1 – Exemplo de agrupamento de dados. 1
Figura 2 – Bases de dados com estruturas homogêneas. 3
Figura 3 – Base de dados com estrutura heterogênea. 4
Figura 4 – Conjuntos de dados com diferentes estruturas. 4
Figura 5 – Partições geradas para a base de dados `Aggregation` pelos algoritmos de agrupamento AL, CeL, CoL, KM e SL com número de *clusters* $k = 7$ e partição real. 5
Figura 6 – Diagrama da estratégia de *ensemble*: conjunto completo de partições Π_C, partições $\pi^1, \pi^2, ..., \pi^n$ e partição consenso π^*. 9
Figura 7 – Diagrama de agrupamento multiobjetivo: conjunto completo de partições $\Pi_C = \{\pi^1, \pi^2, ..., \pi^n\}$ e conjunto reduzido de partições $\Pi_R = \{\pi^1, \pi^2, ..., \pi^m\}$. 10
Figura 8 – Estrutura de uma FP com duas funções objetivo (f_1 e f_2). O pontilhado em (a) demarca a área contendo as soluções dominadas por cada ponto. Os pontos 1, 2 e 3, destacados (b), representam soluções não-dominadas presentes da FP. 11
Figura 9 – Diagrama do *MOCLE*: n^A é o número de algoritmos diferentes utilizados, Π_I é o conjunto de partições iniciais, Π_S é o conjunto de soluções, n^I é o número de partições no conjunto de partições iniciais e n^S é o número de partições no conjunto de soluções. 15
Figura 10 – Diagrama de seleção de partições: conjunto completo de partições $\Pi_C = \{\pi^1, \pi^2, ..., \pi^n\}$ e conjunto reduzido de partições $\Pi_R = \{\pi^1, \pi^2, ..., \pi^m\}$. . 16
Figura 11 – Diagrama do algoritmo *HSS*. 18
Figura 12 – Estruturas conhecidas para a base de dados `monkey`. 20
Figura 13 – Partições geradas pelos algoritmos SL, AL, CoL, CeL, KM e SNN para a base de dados `monkey`. Cada ponto representa uma partição de Π_C. A linha verde conecta as partições não-dominadas presentes na aproximação da FP. 21
Figura 14 – Divisão da Fronteira de Pareto em regiões $(\Omega^1, ..., \Omega^7)$ para a base de dados artificial `monkey`. 22
Figura 15 – Seleção de partições: as estrelas vermelhas representam as partições selecionadas pelo *HSS* para cada região $(\Omega^1, ..., \Omega^7)$ para a base de dados artificial `monkey`. 23
Figura 16 – Partições selecionadas pelo *HSS* para a base de dados `monkey`. 25
Figura 17 – Conjuntos de dados empregados com até três dimensões. 29

Figura 18 – Gráfico de linhas mostrando o *ARI* entre as soluções retornadas pelo *HSS* (linha azul). Quanto menor o *ARI*, maior a dissimilaridade entre as partições sendo comparadas. 36
Figura 19 – Representação do *HSS* aplicado às partições geradas pelos algoritmos evolutivos *MOCK* e *MOCLE*. 37
Figura 20 – FP obtida para as partições geradas pelos algoritmos SL, AL, CoL, CeL, KM e SNN para as base de dados: `Aggregation`, `Flame`, `ds2c2sc13` e `R15`. Cada ponto representa uma partição de Π_C. A linha verde conecta as partições não-dominadas presentes na FP. 53

Lista de tabelas

Tabela 1 – Principais características das bases de dados usadas nos experimentos: número de objetos n, número de atributos d, número de estruturas conhecidas n^E e número de *clusters* K para cada estrutura E_i. 28

Tabela 2 – Número de soluções e qualidade da melhor solução (ARI) retornada pelo HSS com diferentes valores de *percentage* (10, 30, 50 e 100). Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. 31

Tabela 3 – Número de soluções e qualidade da melhor solução (ARI) retornada por Π_C e HSS com a variação do parâmetro alg: KM, SL, CoL, AL, ward (HSS_{KM}, HSS_{SL}, HSS_{CoL}, HSS_{AL}, HSS_{ward}). O número de soluções retornada por todas essas variações é a mesma e está na coluna HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. 33

Tabela 4 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, ASA e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. 35

Tabela 5 – Número de soluções e qualidade da melhor solução (ARI) retornados pelo $MOCK$, pelo conjunto recomendado do $MOCK$ ($MOCK_R$) e pelo HSS recebendo como entrada as partições retornadas pelo $MOCK$ ($MOCK_{HSS}$). Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. 38

Tabela 6 – Número de soluções e qualidade da melhor solução (ARI) retornados pelo $MOCLE$ e pelo HSS recebendo como entrada as partições retornadas pelo $MOCLE$ ($MOCLE_{HSS}$). Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. 40

Tabela 7 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, SR, BRP, SRD, $Diversity$ (Div) e HSS. Como as estratégias de *ensemble* SR, BRP, SRD, Div retornam o mesmo número de partições, esse número está na coluna CES. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. 42

Tabela 8 – Tabela de contingência para a comparação de duas partições: π^1 e π^2. . 51

Tabela 9 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, SR_{10}, SR_{25}, SR_{50}, SR_{75} e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. 58

Tabela 10 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, BRP_{10}, BRP_{25}, BRP_{50}, BRP_{75} e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. . . 59
Tabela 11 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, SRD_{10}, SRD_{25}, SRD_{50}, SRD_{75} e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. . . 60
Tabela 12 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, Div_{10}, Div_{25}, Div_{50}, Div_{75} e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI. 61

Lista de abreviaturas e siglas

AL	*Average-Link*
ARI	*Adjusted Rand Index*
ASA	*Automatic Selection Algorithm*
BRP	*Best Rank Position*
CeL	*Centroid-Link*
CES	*Cluster Ensemble Selection*
CL	*Complete-Link*
CRI	*Combination of Relative Indexes*
Div	*Diversity*
FP	Fronteira de Pareto
HSS	*Hybrid Selection Strategy*
KM	K-Médias
MCLA	*Meta-CLustering Algorithm*
MOCK	*Multi-Objective Clustering with automatic K-determination*
MOCLE	*Multi-Objective Clustering Ensemble algorithm*
MSTs	*Minimum Spanning Trees*
PESA-II	*Pareto Envelope-based Selection Algorithm version 2*
POM	Problema de Otimização Multiobjetivo
SL	*Single-Link*
SNN	*Shared Nearest Neighbor*
SR	*Sum of Ranks*
SRD	*Sum of Ranks with Diversity*

Lista de símbolos

$\delta(.,.)$	Função de distância escolhida
μ_i	Centróide do *cluster* c_i
$\|\Omega^i\|$	Número de partições na região Ω^i
Ω^i	i-ésima região
$\|\Pi\|$	Número de partições no conjunto Π
Π_C	Conjunto completo de partições
Π_C^i	i-ésimo conjunto completo de partições
Π_I	Conjunto intermediário de partições
Π_R	Conjunto reduzido de partições
Π_R^i	i-ésimo conjunto reduzido de partições
π^*	Partição consenso
π^i	i-ésima partição
ari_m	ARI médio
C	Número de partições
c_j^i	j-ésimo *cluster* da i-ésima partição
d	Número de atributos
E_i	i-ésima estrutura
f_i	i-ésima função-objetivo
K	Número de *clusters*
K^i	Número de *clusters* da i-ésima partição
L	Número de vizinhos
M	Número de funções-objetivo
n	Número de objetos

n^E	Número de estruturas conhecidas
n_i	Número de partições idênticas a π^i
n_{ij}	Número de objetos comuns para os *clusters* c_i^1 e c_j^2
$n_{i.}$	Número de objetos no *cluster* c_i^1
$n_{.j}$	Número de objetos no *cluster* c_j^2
nn_{ij}	j-ésimo vizinho mais próximo do objeto x_i
nR	Número de regiões
O	Lista ordenada
o_i	i-ésima solução da lista ordenada O
P	Conjunto não-dominado
v	Número de vizinhos mais próximos
x	Solução candidata
x_i	i-ésimo objeto do conjunto de dados X
X	Conjunto de dados

Sumário

1 INTRODUÇÃO 1
1.1 Contextualização 1
1.2 Descrição do Problema 2
1.3 Objetivos e Abordagem Proposta 5
1.4 Organização do Trabalho 6

2 AGRUPAMENTO 7
2.1 Definição e Principais Características 7
2.2 *Ensemble* de Agrupamento 8
2.3 Agrupamento Multiobjetivo 9
2.3.1 Pareto 11
2.4 Técnicas de Agrupamento Multiobjetivo 13
2.4.1 *MOCK* 13
2.4.2 *MOCLE* 14
2.5 Seleção de Partições 14
2.5.1 *ASA* 15

3 *HYBRID SELECTION STRATEGY* (*HSS*) 17
3.1 Algoritmo de Seleção *HSS* 17
3.2 Etapas do Algoritmo 18
3.2.1 Fase 1: Geração da Fronteira de Pareto 18
3.2.2 Fase 2: Divisão da Fronteira de Pareto em Regiões 21
3.2.3 Fase 3: Seleção de Partições 22
3.2.4 Custo Computational 23

4 EXPERIMENTOS E RESULTADOS 27
4.1 Bases de Dados 27
4.2 Metodologia Experimental 28
4.3 Estudo Empírico dos Parâmetros do *HSS* 30
4.3.0.1 Variação do Parâmetro *porcentagem* 30
4.3.0.2 Variação do Parâmetro *alg* 32
4.4 Resultados 34
4.4.1 *ASA* e *HSS* 34
4.4.2 *MOCK* e *MOCLE* 36
4.4.3 Seleções de *Ensemble* 40

5 CONCLUSÃO 43

Referências 45

APÊNDICE A – *CLUSTER ENSEMBLE SELECTION* (*CES*) . . . 49

APÊNDICE B – *ADJUSTED RAND INDEX* (*ARI*) 51

APÊNDICE C – FRONTEIRA DE PARETO (*FP*) 53

APÊNDICE D – *AUTOMATIC SELECTION ALGORITHM* (*ASA*) 55

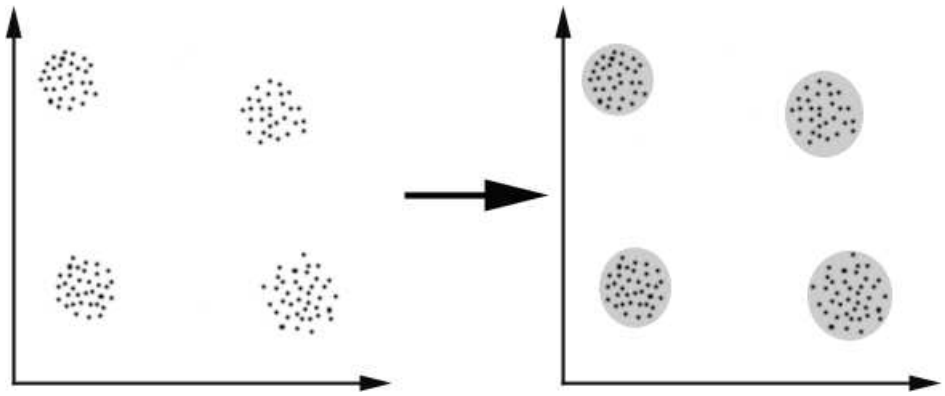
APÊNDICE E – ESTUDO DAS SELEÇÕES DE *ENSEMBLE* . . . 57

1 Introdução

1.1 Contextualização

O volume de dados em diversas aplicações tem crescido exponencialmente nos últimos anos e, consequentemente, a necessidade de entendê-los. Agrupamento de dados, cujo objetivo é agrupar uma coleção de dados não-rotulados em *clusters* (isto é, grupos) significativos, tem sido amplamente adotado na descoberta de conhecimento. Possui aplicações em diversos segmentos da computação, tais como análise de padrões, tomada de decisão, mineração de dados e segmentação de imagens (JAIN; MURTY; FLYNN, 1999). Também pode ser aplicado em outras áreas do conhecimento, tais como biologia, botânica, medicina, psicologia, geografia, marketing, psiquiatria, arqueologia (EVERITT et al., 2011). Usualmente, informações sobre esses dados são escassas ou até mesmo inexistentes. A Figura 1 exibe um exemplo de agrupamento no qual a base de dados é dividida em quatro *clusters* esféricos bem definidos.

Figura 1 – Exemplo de agrupamento de dados.



Fonte: <http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/images/clustering.gif> (acessado em 14/12/2017).

Jain (2010) declara que não existe um melhor algoritmo de agrupamento, pois cada algoritmo impõe uma estrutura nos dados de forma explícita ou implícita. Dessa forma, cada algoritmo pode ser melhor para uma base de dados específica: o *single-link*, por exemplo, é ideal para a identificação de *clusters* no formato de correntes; o k-médias, por outro lado, é empregado para a identificação de *clusters* esféricos. Visto que em agrupamento as características dos dados não são conhecidas previamente, a escolha de um algoritmo de agrupamento pode ser desafiadora. Esse é um dos motivos pelo qual, em diversas aplicações, é interessante retornar, ao invés de uma única partição, um conjunto de partições. Além disso, muitas bases de dados apresentam vários *clusters* com formatos diferentes, difíceis

de serem identificados por um único algoritmo de agrupamento. Para esses casos uma boa estratégia é empregar um algoritmo capaz de combinar partições (STREHL; GHOSH, 2002).

Métodos de agrupamento tradicionais buscam uma única solução - ou melhor, partição - para a base de dados de interesse. No entanto, dados reais podem ser multi-facetados, especialmente em casos de bases de dados grandes e complexas. Lei (2016) defende que a definição de uma boa partição depende do usuário em questão. Ademais, ele pode ter interesse em analisar diferentes visões sobre os dados. Uma possível alternativa para resolver este problema é a aplicação de diferentes métodos e/ou do mesmo método com diferentes parâmetros, gerando, assim, uma variedade de partições de agrupamento que podem ser igualmente relevantes se não houver nenhum conhecimento prévio sobre os dados. O problema, agora, é que entre as partições geradas pode haver algumas melhores do que outras e também pode haver redundância. Há três alternativas frequentes para lidar com este problema: (i) *ensemble* de agrupamento (GHAEMI et al., 2009) - combina múltiplas partições em uma partição consenso; (ii) agrupamento multiobjetivo (LAW; TOPCHY; JAIN, 2004) - encontra múltiplas partições através da aplicação de diversos algoritmos de agrupamento e otimiza um número conflitante de funções objetivo; e (iii) seleção de partições (SAKATA et al., 2010; FACELI et al., 2010) - dado um conjunto de partições gerado através da aplicação de diversos algoritmos de agrupamento, um subconjunto deste é selecionado.

As estratégias de agrupamento multiobjetivo mais populares usam, usualmente, algoritmos evolutivos (MUKHOPADHYAY; MAULIK; BANDYOPADHYAY, 2015). No entanto, estes possuem um custo computacional muito alto (GOEL; STANDER, 2010) e retornam, com frequência, um número muito grande de partições. Além disso, amiúde, estes algoritmos retornam partições similares.

O desenvolvimento de algoritmos de seleção foi impulsionado pelo interesse de retornar ao usuário um conjunto reduzido de partições a um custo computacional menor do que as estratégias multiobjetivas tradicionais.

1.2 Descrição do Problema

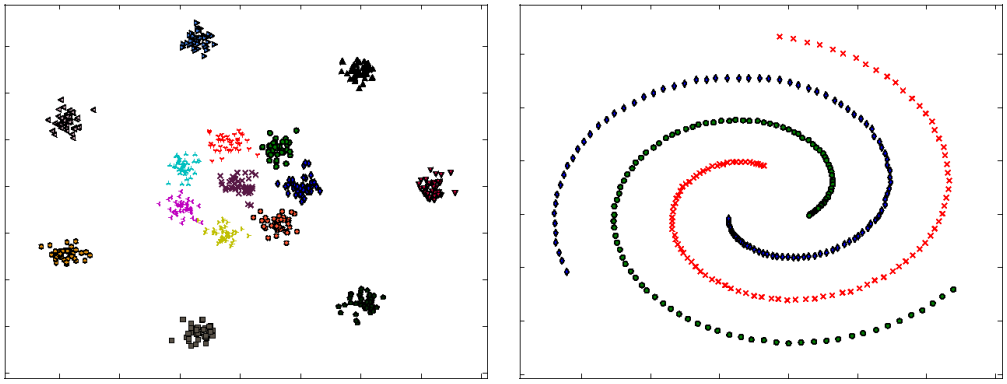
Agrupamento de dados é uma técnica de aprendizado não-supervisionado, o que significa que não há conhecimento sobre os dados a serem agrupados e nem sobre as estruturas presentes nos dados (definição formal de agrupamento disponível na Seção 2.1). Cada algoritmo de agrupamento, normalmente, é apropriado para identificar um tipo de estrutura diferente - o k-médias (KM), por exemplo, é apropriado para a identificação de *clusters* compactos, esféricos, mas é falho na identificação de *clusters* na forma de corrente, escadeados. Cada base de dados apresenta características distintas, o que dificulta a escolha

de um único algoritmo de agrupamento para ser usado universalmente, em todas as bases de dados com que se deseja trabalhar.

A Figura 2 mostra duas bases de dados com estruturas homogêneas, isto é, com *clusters* de um único formato. A base de dados `R15`, ilustrada na Figura 2a, possui quinze *clusters* esféricos, que são facilmente identicados pelo algoritmo KM, mas que não são identificados pelo *single-link* (SL). Já a base de dados `spiral`, ilustrada na Figura 2b, possui três *clusters* no formato de corrente e é identificada através do algoritmo SL e pelo *Shared Nearest Neighbor* (SNN), mas não pelos algoritmos KM, *complete-link* (CL), *average-link* (AL) e *centroid-link* (CeL).

Figura 2 – Bases de dados com estruturas homogêneas.

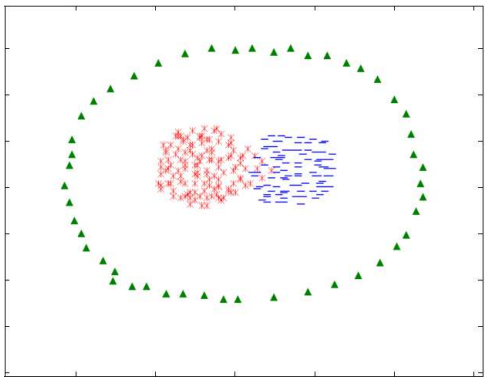
(a) `R15` (*clusters* esféricos) (b) `spiral` (*clusters* encadeados)



Grande parte dos problemas de agrupamento, no entanto, apresenta partições com estruturas heterogêneas, isto é, de diferentes tamanhos, densidades e/ou formatos. Muitas, difíceis de serem identificadas pelos algoritmos de agrupamento, em virtude de eles, frequentemente, adotarem um critério homogêneo em todo o espaço de atributos (LAW; TOPCHY; JAIN, 2004). A Figura 3 apresenta uma base de dados com dois *clusters* no formato globular e um no formato de anel. Nenhum dos algoritmos (KM, SL, AL, CL, CeL, SNN) consegue identificar os três *clusters*.

Um conjunto de dados pode ter mais de uma estrutura, cada uma representando uma interpretação diferente dos dados (HANDL; KNOWLES, 2007). A Figura 4 expõe a base de dados `ds2c2sc13`, que possui diferentes estruturas, em níveis de refinamento variados. Faceli (2006) comenta que, para essa base, a estrutura mais facilmente identificada por algoritmos de agrupamento é a que possui dois *clusters* (exibida na Figura 4a), uma vez que estes possuem formatos quase esféricos e estão bem separados um do outro. A Figura 4b mostra a base `ds2c2sc13` dividida em cinco *clusters*. Já a Figura 4c apresenta treze *clusters*, exibindo o melhor nível de refinamento entre as partições apresentadas. A aplicação de um algoritmo de agrupamento tradicional sobre essa base de dados forneceria ao especialista no domínio somente uma visão sobre estes dados, deixando de lado outras visões também interessantes.

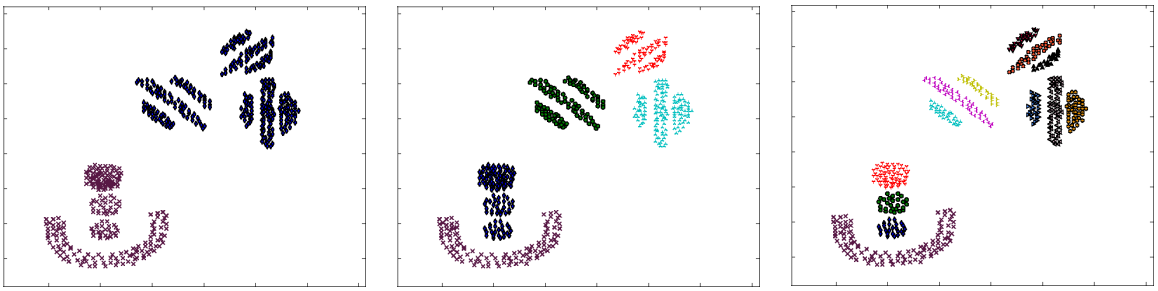
Figura 3 – Base de dados com estrutura heterogênea.



Fonte: (FACELI, 2006).

Figura 4 – Conjuntos de dados com diferentes estruturas.

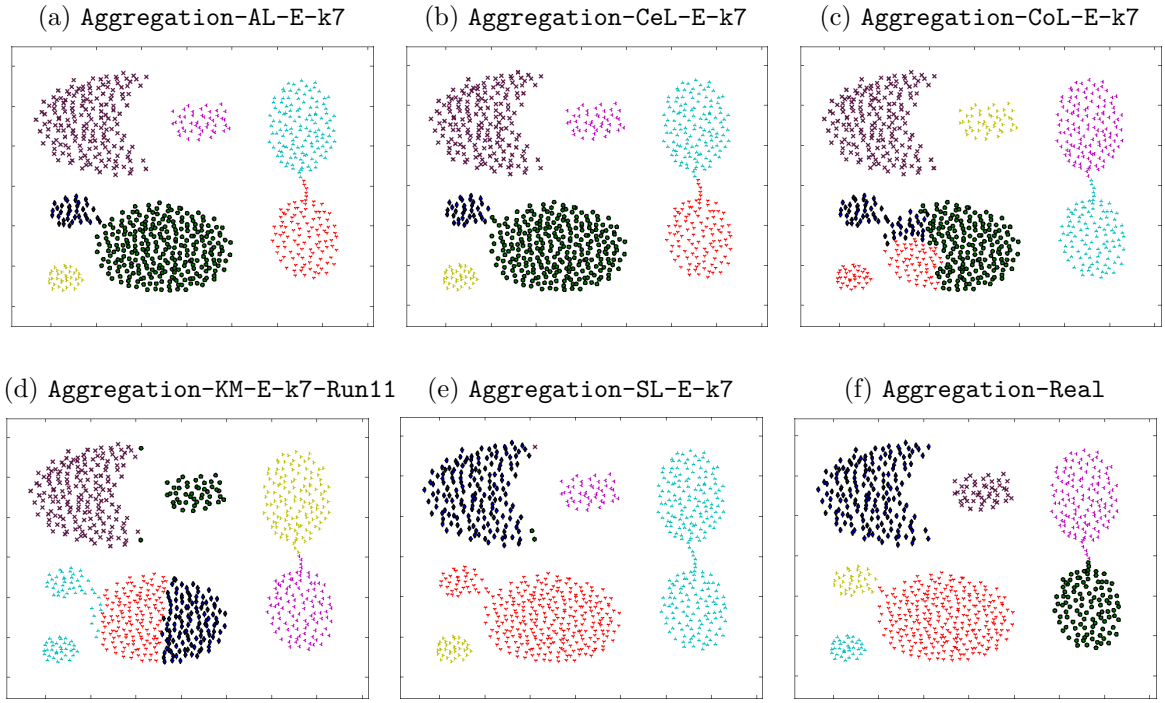
(a) `ds2c2sc13` (2 *clusters*) (b) `ds2c2sc13` (5 *clusters*) (c) `ds2c2sc13` (13 *clusters*)



A Figura 5 exibe o resultado da aplicação de diferentes algoritmos de agrupamento - AL, CeL, CoL, KM e SL - na base de dados `Aggregation` com o número de *clusters* $k = 7$, uma vez que o conjunto real, conforme ilustrado na Figura 5f, possui 7 *clusters*. As partições que mais se aproximam da partição real, embora não sejam iguais a ela, são a retornada pelo AL (Figura 5a) e pelo CeL (Figura 5b). Há uma diferença considerável entre as estruturas retornadas por diferentes algoritmos e sem o conhecimento da partição real seria difícil eleger a melhor partição. Como em problemas de agrupamento também não há conhecimento sobre o valor do k, a identificação de uma partição boa, que atenda as expectativas do especialista no domínio, é ainda mais desafiadora.

Uma técnica de agrupamento multiobjetivo pode ser aplicada com o propósito de sanar os problemas acima descritos, uma vez que ela retorna diferentes partições obtidas através da execução de diversos algoritmos de agrupamento, executados com diferentes parâmetros. Isso permite que o especialista no domínio tenha acesso a um conjunto variado de partições. Porém, como citado anteriormente, os algoritmos multiobjetivos costumam retornar um conjunto grande de partições, tornando viável a elaboração de um algoritmo de seleção de partições. Com o *HSS* ambos os conceitos - agrupamento multiobjetivo e seleção de partições - são adotados na tentativa de resolver os problemas aqui aludidos.

Figura 5 – Partições geradas para a base de dados `Aggregation` pelos algoritmos de agrupamento AL, CeL, CoL, KM e SL com número de *clusters* $k = 7$ e partição real.



1.3 Objetivos e Abordagem Proposta

Nesta dissertação, é proposto um algoritmo de seleção de partições, chamado *HSS*, para o problema de agrupamento, visando fornecer ao usuário diferentes visões - partições - sobre a base de dados fornecida como entrada. É importante destacar que, como um algoritmo de seleção, o *HSS* não gera partições novas, somente seleciona partições dentre aquelas recebidas como entrada. Por isso, para que o *HSS* selecione um conjunto diverso de partições, é importante que o usuário forneça como entrada um conjunto de partições geradas através de diferentes algoritmos de agrupamento, executados com diferentes parâmetros. Foram definidos três objetivos para o *HSS*:

1. Retornar, ao especialista no domínio, um conjunto reduzido de partições;
2. Retornar partições de boa qualidade;
3. Retornar partições dissimilares.

O *HSS* é dividido em três etapas. Na primeira etapa, que toma como entrada um conjunto de partições geradas através da aplicação de diversos algoritmos de agrupamento (com diferentes parâmetros), um algoritmo multiobjetivo é aplicado com o desígnio de identificar as partições mais promissoras: aquelas que possuem o melhor compromisso

entre os dois objetivos adotados - conectividade e variância. Estas partições compõem a chamada aproximação da Fronteira de Pareto.

As partições obtidas na etapa anterior são, na segunda etapa, dividas em *clusters* através da aplicação de um algoritmo de agrupamento. O propósito desta etapa é decompor o problema original em um número de subproblemas, a diversidade entre os subproblemas implica na diversidade das partições obtidas para o problema original.

Finalmente, na terceira etapa, é realizada a seleção de partições. Esta seleção é executada, de forma independente, em cada um dos *clusters* obtidos na etapa anterior. A seleção tem como objetivo eleger, para cada *cluster*, uma partição que seja, entre as partições presentes no *cluster* em questão, a mais parecida com as demais; duas partições são consideradas semelhantes se possuem estruturas parecidas (ou até mesmo algumas estruturas iguais). A ideia de adotar a similaridade entre partições como critério de seleção parte da pressuposição de que estruturas recorrentes em um conjunto de partições - ou seja, mesmas estruturas identificadas por diversos algoritmos - implicam na relevância destas partições. Portanto, a partição mais parecida com as demais é selecionada.

Em suma, o *HSS*, dado um conjunto de partições, seleciona um subconjunto deste que contenha partições de boa qualidade e que sejam diversas entre si.

1.4 Organização do Trabalho

Esta dissertação está organizada em 5 capítulos de forma que o leitor possa compreender todos os detalhes desta pesquisa. Mais especificamente:

- Capítulo 2: é exposta a fundamentação dos conceitos necessários para o desenvolvimento desta pesquisa. Nele, encontra-se um embasamento relativo a *ensemble* de agrupamento, agrupamento multiobjetivo, técnicas de agrupamento multiobjetivo de interesse e seleção de partições.
- Capítulo 3: a técnica de seleção desenvolvida - o *HSS* - é descrita, discorrendo, inicialmente, a respeito do problema que motivou a elaboração do algoritmo. Neste capítulo são detalhadas as três etapas do *HSS* e seu custo computacional.
- Capítulo 4: é exibida uma descrição das bases de dados utilizadas, dos experimentos realizados com a técnica descrita no Capítulo 3 e dos resultados obtidos.
- Capítulo 5: é apresentada a conclusão deste trabalho, resumindo os principais resultados obtidos e os possíveis trabalhos futuros.

2 Agrupamento

A maioria dos algoritmos de agrupamento adotam um critério homogêneo para toda a base de dados, não sendo capazes de identificar todas as estruturas presentes em um conjunto se cada região deste possui uma forma, tamanho e/ou densidade diferente (LAW; TOPCHY; JAIN, 2004). Mesmo técnicas como Rodriguez e Laio (2014) e Ertoz, Steinbach e Kumar (2002), capazes de identificar *clusters* com diferentes densidades e formas, retornam somente uma partição - existem aplicações que possuem interesse em obter diversas visões (partições) sobre os dados. Um exemplo é uma tarefa de roteamento, cujo objetivo é encontrar um conjunto de rotas para uma determinada combinação de rede/tráfego, de modo que os custos de comunicação e o congestionamento sejam minimizados. Este é um problema de agrupamento com dois objetivos a serem minimizados: (i) custos de comunicação e (ii) congestionamento.

Uma alternativa factível para a resolução deste problema é a aplicação de diversos métodos e/ou o mesmo método com diferentes parâmetros, produzindo uma variedade de partições de agrupamento que podem ser consideradas igualmente relevantes se não houver conhecimento prévio sobre os dados. Há duas técnicas frequentes para tratar este problema: *ensemble* e agrupamento multiobjetivo, ambas descritas neste capítulo. Outra técnica que surgiu, devido às limitações das técnicas existentes, foi a seleção de partições.

Na sequência, será apresentada a fundamentação dos conceitos necessários para o desenvolvimento desta pesquisa. Primeiramente, na Seção 2.1 são expostas a definição e as principais características do processo de agrupamento de dados. A Seção 2.2 explica como funcionam os algoritmos de *ensemble*, responsáveis por combinar, através de uma função consenso, diversas partições. Já a Seção 2.3 define o que é agrupamento multiobjetivo, detalhando Pareto, a principal abordagem para tratar este problema. A Seção 2.4 descreve *MOCK* e *MOCLE*, dois algoritmos de agrupamento multiobjetivo. Finalmente, a Seção 2.5 descreve a seleção de partições de agrupamento e apresenta o *ASA*, um algoritmo de seleção.

2.1 Definição e Principais Características

Agrupamento de dados é considerado uma técnica de aprendizado não-supervisionado responsável pela descoberta de grupos e pela identificação de padrões nos dados. O processo de agrupamento baseia-se no fato de que não há classes pré-definidas para os dados e, também, não há conhecimento sobre relações válidas ou esperadas entre os dados. Esta técnica, dado um conjunto de objetos $X = \{x_1, x_2, ..., x_n\}$ de tamanho n, particiona X em K^i *clusters* $\pi^i = \{c^i_1, c^i_2, ..., c^i_{K^i}\}$ baseado em uma distância de similaridade/dissimilaridade.

O número de *clusters* pode ou não ser conhecido a priori. Uma vez que π^i é uma partição de X, $c_k^i \cap c_l^i = \emptyset$ para $k \neq l$ e $\bigcup_{k=1}^{K^i} c_k^i = X$.

Os algoritmos de agrupamento são, tradicionalmente, distinguidos entre métodos particionais, hierárquicos e baseados em densidade (AGGARWAL; REDDY, 2013). Os métodos particionais usam uma métrica baseada em distância para agrupar os dados de acordo com uma medida de similaridade. Um exemplo de algoritmo deste tipo é o k-médias (JAIN; DUBES, 1988). Métodos hierárquicos, por outro lado, particionam os dados em diferentes níveis (hierarquias). Podem adotar a abordagem *bottom-up (agglomerative clustering)* ou *top-down (divisive clustering)*. Exemplos de algoritmos hierárquicos: *single-link*, *complete-link*, *BIRCH* (JAIN; DUBES, 1988; ZHANG; RAMAKRISHNAN; LIVNY, 1996). Os métodos baseados em densidade são capazes de identificar *clusters* com formatos arbitrários. Exemplo: *DBSCAN* (ESTER et al., 1996).

O processo tradicional de agrupamento resulta, usualmente, em uma partição distinta para os dados de acordo com o critério de agrupamento empregado. Este processo pode ser dividido nos seguintes passos: (i) aplicação de um algoritmo de agrupamento; (ii) validação dos resultados, é importante verificar a qualidade da partição retornada, uma vez que não se tem conhecimento prévio sobre a partição e/ou os *clusters* desejados; e (iii) interpretação dos resultados, o especialista na área precisa, muitas vezes, analisar o resultado do agrupamento e integrá-lo aos seus experimentos/análises (HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2001).

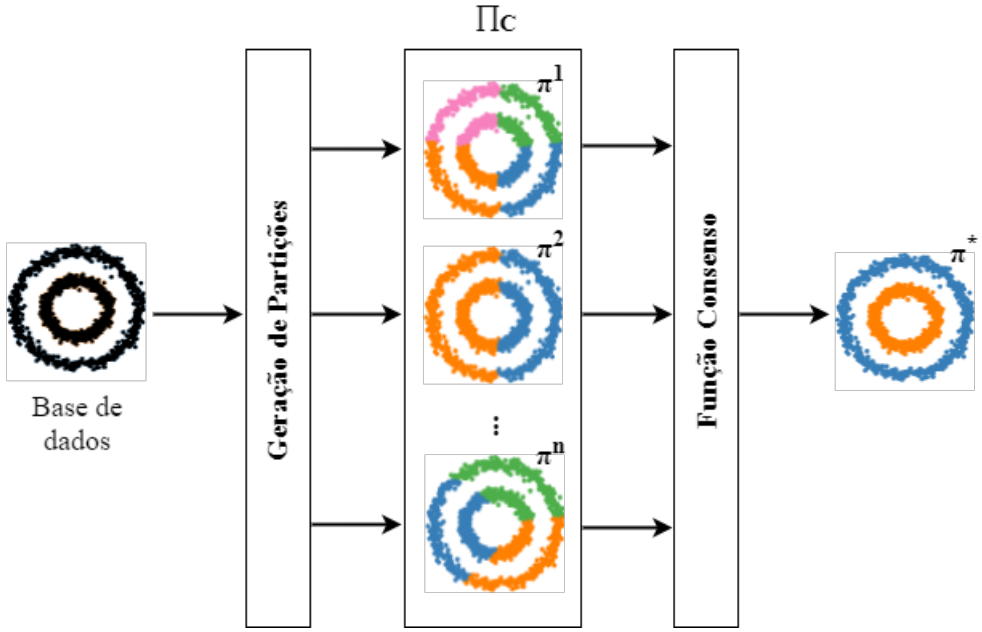
Conforme supracitado, as técnicas tradicionais de agrupamento possuem diversas limitações. Como cada algoritmo adota um critério de agrupamento diferente, a escolha do algoritmo adequado é desafiadora. Esses algoritmos têm dificuldades na identificação de *clusters* heterogêneos em uma partição e limitam-se em fornecer somente uma visão sobre os dados. A seguir serão descritas as três abordagens usuais para lidar com estas limitações: (i) *ensemble* de agrupamento (GHAEMI et al., 2009); (ii) agrupamento multiobjetivo (LAW; TOPCHY; JAIN, 2004); e (iii) seleção de partições (SAKATA et al., 2010; FACELI et al., 2010).

2.2 *Ensemble* de Agrupamento

Uma técnica de *ensemble* combina múltiplas partições, geradas através de diferentes algoritmos de agrupamento, em uma única partição por meio de uma função consenso. Métodos de *ensemble* são frequentemente mais robustos e apresentam melhor qualidade do que métodos de agrupamentos individuais, o que indica que a combinação de objetivos de agrupamento distintos apresenta bons resultados. Porém, os algoritmos de *ensemble* retornam uma única solução, não sendo capazes de explorar uma ampla gama de partições boas de acordo com uma lista de objetivos pré-determinados. A Figura 6 apresenta o

funcionamento da técnica: dada uma base de dados, é gerado um conjunto completo de partições (Π_C) que são combinadas, por meio de uma função consenso, em uma partição consenso (π^*).

Figura 6 – Diagrama da estratégia de *ensemble*: conjunto completo de partições Π_C, partições $\pi^1, \pi^2, ..., \pi^n$ e partição consenso π^*.



Fonte: base de dados extraída de <http://scikit-learn.org/stable/_images/sphx_glr_plot_cluster_comparison_0011.png> (acessado em 14/12/2017).

Esta técnica apresenta duas grandes desvantagens: (i) um grande número de partições base de baixa qualidade pode resultar em uma partição consenso ruim, mesmo havendo algumas partições base boas; e (ii) dificuldade de geração de uma estrutura heterogênea (com diferentes tipos de *clusters*), uma vez que a informação desses *clusters* pode ser sobrescrita por outros de qualidade inferior (FACELI; CARVALHO; SOUTO, 2007).

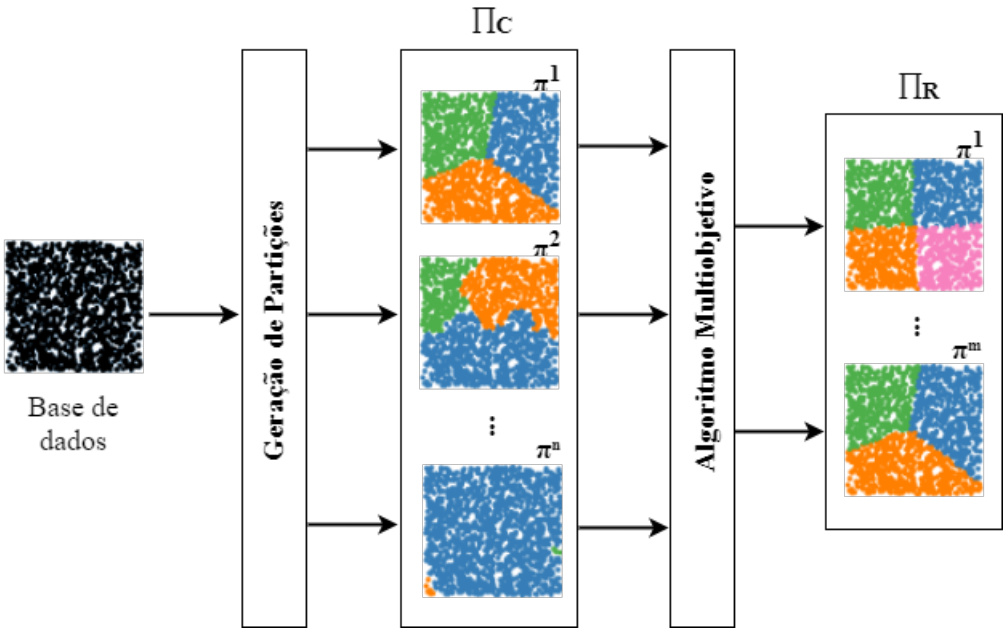
Na tentativa de contornar o possível impacto negativo de partições base ruins, foram desenvolvidos algoritmos para a seleção de partições a serem usadas como entrada pelos algoritmos de *ensemble*. Um trabalho neste contexto foi proposto por Naldi, Carvalho e Campello (2013) (descrito no Apêndice A).

2.3 Agrupamento Multiobjetivo

Diferentemente das técnicas de *ensemble*, cuja principal limitação está no fato de retornarem somente uma partição, o agrupamento multiobjetivo fornece, ao especialista no domínio, um conjunto de partições. Tem como finalidade a otimização simultânea de dois ou mais critérios de agrupamento - objetivos - que sejam complementares. A melhora

no valor de um objetivo pode causar piora no valor do outro. Usualmente, não há uma única solução ótima, mas um conjunto de soluções. Estas são ótimas no sentido de que não há soluções no espaço de busca que sejam superiores a elas quando todos os objetivos são considerados (ZITZLER; THIELE, 1998). A Figura 7 descreve o funcionamento desta técnica: dada uma base de dados, é gerado um conjunto completo de partições (Π_C), estas partições são tomadas como entrada pelo algoritmo multiobjetivo, que retornará um conjunto reduzido de partições (Π_R) que pode conter, além de partições tomadas como entrada, no caso de algoritmos evolutivos, outras partições geradas durante o processo evolutivo.

Figura 7 – Diagrama de agrupamento multiobjetivo: conjunto completo de partições $\Pi_C = \{\pi^1, \pi^2, ..., \pi^n\}$ e conjunto reduzido de partições $\Pi_R = \{\pi^1, \pi^2, ..., \pi^m\}$.



Fonte: base de dados extraída de <http://scikit-learn.org/stable/_images/sphx_glr_plot_cluster_comparison_0011.png> (acessado em 14/12/2017).

O problema de otimização que envolve mais de uma função objetivo é conhecido como Problema de Otimização Multiobjetivo (POM). O POM pode ser definido, em palavras, como um vetor de variáveis de decisão que satisfaz restrições e otimiza uma função vetorial cujos elementos representam as funções-objetivo. Essas funções formam uma descrição matemática dos critérios de desempenho que são, usualmente, conflitantes. Portanto, o termo otimizar significa encontrar uma solução capaz de prover, ao especialista no domínio, um valor aceitável para cada função objetivo (OSYCZKA, 1985).

Um POM pode ser definido, matematicamente, como:

$$\begin{aligned} \min/\max \quad & F(x) = (f_1(x), f_2(x), ..., f_m(x)) \\ \text{sujeito a} \quad & x \in \Omega \end{aligned} \tag{2.1}$$

em que f_i é a i-ésima função-objetivo, m é o número de funções-objetivo, $f_i(x)$ é uma função objetivo escalar, n é o número de variáveis de decisão, $x \in \mathbb{R}^n$ é uma solução candidata e $\Omega \subset \mathbb{R}^n$ é a região viável.

Uma vez que os objetivos da Equação 2.1 são conflitantes, não há nenhum ponto em Ω capaz de minimizar/maximizar todos eles simultaneamente. Para resolver este problema é preciso equilibrar seus objetivos. Deb (2011) levanta dois objetivos ideais para a otimização multiobjetivo:

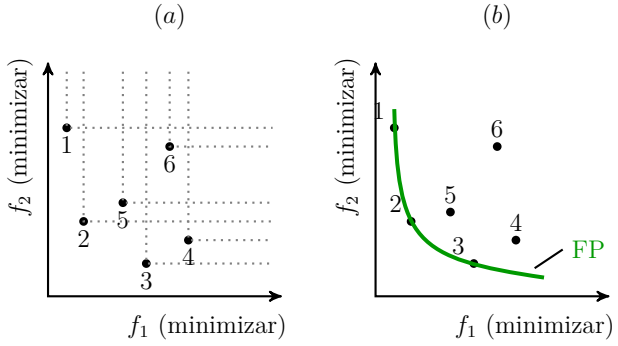
1. Encontrar um conjunto de soluções que esteja na Fronteira Pareto-ótima;

2. Encontrar um conjunto de soluções que seja diverso o suficiente para representar toda a Fronteira Pareto-ótima.

2.3.1 Pareto

Uma abordagem popular para manipular um POM é a aplicação de um método baseado em Pareto. A otimização multiobjetivo baseada em Pareto tem como meta determinar um conjunto de soluções Pareto-ótimas ou um subconjunto representativo do mesmo (KONAK; COIT; SMITH, 2006). Uma solução é considerada Pareto-ótima se não há nenhuma solução que melhore pelo menos um dos objetivos sem deteriorar os demais. Um conjunto Pareto-ótimo consiste de todas as soluções não-dominadas presentes no conjunto de soluções. A Fronteira de Pareto (FP) é o conjunto de todas as soluções Pareto-ótimas no espaço de objetivos. Porém, para muitos problemas reais complexos, não é possível encontrar todas as soluções ótimas, o que torna prática a busca por uma aproximação do conjunto Pareto-ótimo (FACELI, 2006). A Figura 8 mostra a estrutura de uma FP para um conjunto de seis partições, representadas pelos pontos $1, 2, ..., 6$, neste exemplo deseja-se minimizar as duas funções objetivos, f_1 e f_2; os pontos 1, 2 e 3 representam soluções não-dominadas presentes na FP.

Figura 8 – Estrutura de uma FP com duas funções objetivo (f_1 e f_2). O pontilhado em (a) demarca a área contendo as soluções dominadas por cada ponto. Os pontos 1, 2 e 3, destacados (b), representam soluções não-dominadas presentes da FP.



A seguir são apresentadas definições que consideram um POM no qual as funções objetivo são minimizadas.

Definição 1 (Pareto-dominância)**.** Uma solução candidata $x^1 \in \Omega$ Pareto domina uma solução condidata x^2, denotado por $x^1 \preceq x^2$, se e somente se $f_i(x^1) \leqslant f_i(x^2), \forall i \in \{1, ..., m\}$ e $f_j(x^1) < f_j(x^2)$ para pelo menos um $j \in \{1, ..., m\}$.

Definição 2 (Pareto-otimalidade)**.** Uma solução candidata $x^* \in \Omega$ é considerada Pareto-ótima se não há nenhuma outra solução candidata $x \in \Omega$ tal que $x \preceq x^*$. $f(x^*)$ é então chamado de vetor Pareto-ótimo.

Definição 3 (Fronteira de Pareto)**.** O conjunto de todas as soluções Pareto-ótimas é conhecido como conjunto Pareto-ótimo (P). O conjunto de todos os vetores Pareto-ótimos, $FP = \{F(x) \in \mathbb{R}^n | x \in P\}$, é conhecido como Fronteira Pareto-ótima.

Esta abordagem, porém, não é prática para resolver problemas com muitos objetivos (mais de três). Com o aumento da dimensionalidade, as relações de Pareto-dominância não funcionam bem devido ao fato de elas trabalharem com ordenação parcial. Isso significa que dados dois vetores, eles podem ser: idênticos, superior/inferior ou incomparáveis (GIAGKIOZIS; PURSHOUSE; FLEMING, 2013). Com o aumento no número de objetivos, há também um aumento no número de soluções não-dominadas, enfraquecendo a eficiência do método (ISHIBUCHI; TSUKAMOTO; NOJIMA, 2008).

Existem diversos algoritmos na literatura voltados para a identificação da Fronteira de Pareto, tais como Kung, Luccio e Preparata (1975), Deb et al. (2000), Ding, Zeng e Kang (2003) e Mishra e Harit (2010). O pior caso da complexidade de tempo de todos esses algoritmos é $O(M(|\Pi|)^2)$ (M é o número de funções objetivo e $|\Pi|$ é o número de partições). Entretanto, o método proposto por Mishra e Harit (2010) apresenta a complexidade de tempo de melhor caso $O(|\Pi| log(|\Pi|))$ para qualquer número de objetivos.

Mishra e Harit (2010) é executado seguindo os seguintes passos:

1. Ordene todas as partições (π^1,...,π^C) em ordem decrescente do valor da primeira função objetivo f_1 e crie uma lista ordenada O.

2. Inicialize o conjunto não-dominado P e adicione o primeiro elemento da lista O em P.

3. Para cada solução $o_i \in O$ (exceto a primeira), compare a solução o_i com as soluções do conjunto P:

   - Se algum elemento de P domina o_i, delete o_i da lista.
   - Se o_i domina alguma solução do conjunto P, delete essa solução de P.

- Se o_i é uma partição não-dominada em relação ao conjunto P, então atualize o conjunto $P = P \cup \{o_i\}$.
- Se P se tornar vazio, adicione a solução imediata de O em P.

4. Retorne o conjunto não-dominado P.

2.4 Técnicas de Agrupamento Multiobjetivo

Nesta seção são apresentados dois algoritmos evolutivos multiobjetivos: *MOCK* e *MOCLE*.

2.4.1 *MOCK*

Multi-Objective Clustering with automatic K-determination (*MOCK*) (HANDL; KNOWLES, 2007) seleciona soluções da Fronteira de Pareto baseado em um modelo nulo (*null model*) e determina o número de *clusters* automaticamente. Consiste em duas fases: (i) utiliza, nesta fase inicial, um algoritmo evolutivo multiobjetivo - o *PESA-II* (*Pareto Envelope-based Selection Algorithm version 2*) (CORNE et al., 2001) - para otimizar dois objetivos de agrupamento complementares - *overall deviation* e conectividade, retornando uma FP o mais completa possível, com conjunto de partições mutuamente não-dominadas com diferentes compromissos entre os dois objetivos; (ii) na segunda etapa, o *MOCK* analisa o formato da curva e a compara com os compromissos obtidos para um modelo nulo apropriado - através do agrupamento de dados aleatórios -, baseado nesta análise o algoritmo fornece uma estimativa da qualidade das partições e determina um conjunto de soluções potencialmente promissoras (HANDL; KNOWLES, 2007).

Durante a primeira fase, o *MOCK* usa a representação de adjacência baseada em lócus (*locus-based adjacency*). Esta apresenta como vantagem o fato de não precisar do número de *clusters* (automaticamente determinado durante a decodificação da representação). O cruzamento uniforme é adotado como operador de recombinação. O operador de mutação escolhido é baseado nos vizinhos mais próximos: cada item pode, somente, ser ligado a um de seus L vizinhos mais próximos. A inicialização do *MOCK* é baseada em dois algoritmos diferentes, com o objetivo de obter uma boa distribuição de soluções que esteja próxima de uma aproximação da FP. Um desses algoritmos é o *Minimum Spanning Trees* (*MSTs*), que apresenta bons resultados em relação ao objetivo de conectividade adotado. O segundo algoritmo escolhido é o k-médias, uma vez que ele apresenta boas soluções em relação ao objetivo de *overall deviation*.

2.4.2 *MOCLE*

Multi-Objective Clustering Ensemble algorithm (*MOCLE*) (FACELI, 2006) aplica ambos os conceitos de agrupamento multiobjetivo e de *ensemble* na tentativa de reduzir as limitações de ambas as estratégias. *MOCLE* é capaz de lidar com estruturas heterogêneas com diferentes tipos de *clusters* em diferentes níveis de refinamento (FACELI et al., 2009).

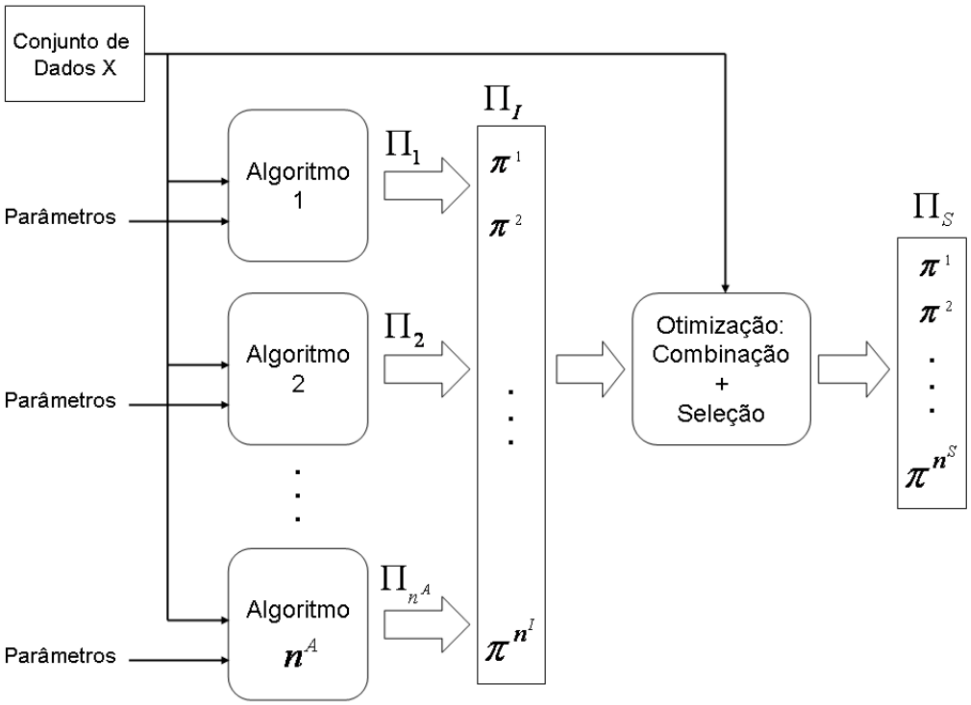
Primeiramente, um conjunto de partições individuais é gerado através da execução de diferentes algoritmos tradicionais de agrupamento com diferentes parâmetros. Este conjunto é usado como entrada para um algoritmo genético multiobjetivo baseado em Pareto. A otimização é aqui adotada para combinar e selecionar o melhor conjunto de partições alternativas (FACELI; CARVALHO; SOUTO, 2007). Pares de partições são combinados iterativamente nesse processo em uma tentativa de evitar a influência negativa de partições de baixa qualidade. O conjunto de soluções pode conter partições que são combinações de outras partições ou partições de alta qualidade presentes no conjunto inicial. A Figura 9 exibe o diagrama do *MOCLE*: dado um conjunto de dados, são executados vários algoritmos de agrupamento com diferentes parâmetros para a geração do conjunto inicial de partições Π_I, essas partições iniciais são selecionadas (validação) e combinadas (*ensemble*) durante o processo de otimização do *MOCLE*, resultando em um conjunto reduzido de partições Π_S.

O operador de mutação empregado em algoritmos genéticos não é aplicado no *MOCLE*, que restringe seu espaço de busca às partições base e suas combinações (FACELI; CARVALHO; SOUTO, 2007). Um algoritmo de *ensemble*, chamado *MCLA* (*Meta-CLustering Algorithm*), é empregado pelo *MOCLE* para a operação de *crossover*. O número de *clusters* das partições no conjunto de soluções está limitado ao intervalo entre o número mínimo e máximo de *clusters* das partições do conjunto inicial (FACELI; CARVALHO; SOUTO, 2007).

2.5 Seleção de Partições

As estratégias de seleção de partições surgiram na tentativa de contornar os principais problemas das técnicas de agrupamento multiobjetivo que, muitas vezes, retornam um número grande de partições e possuem um custo computacional elevado (GOEL; STANDER, 2010). Um algoritmo de seleção recebe como entrada um conjunto de partições e seleciona um subconjunto deste. A Figura 10 descreve o funcionamento desta técnica: dada uma base de dados, é gerado um conjunto completo de partições (Π_C), estas partições são tomadas como entrada pelo algoritmo de seleção, que retornará um conjunto reduzido de partições (Π_R), que contém um subconjunto das partições recebidas como entrada.

Figura 9 – Diagrama do *MOCLE*: n^A é o número de algoritmos diferentes utilizados, Π_I é o conjunto de partições iniciais, Π_S é o conjunto de soluções, n^I é o número de partições no conjunto de partições iniciais e n^S é o número de partições no conjunto de soluções.



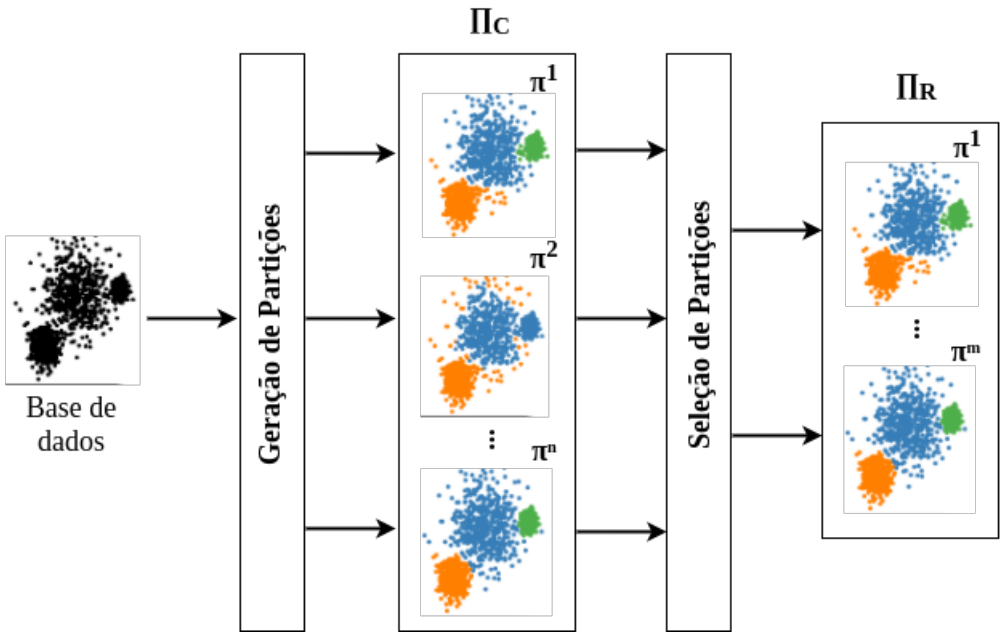
Fonte: (FACELI, 2006).

2.5.1 *ASA*

O algoritmo de seleção de partições *Automatic Selection Algorithm* (*ASA*), proposto em Sakata et al. (2010), tem como objetivo a minimização de um problema recorrente em técnicas multiobjetivo: estas retornam um conjunto de soluções grande, inviável de ser analizado manualmente. O *ASA* seleciona as partições mais diferentes entre si através da aplicação de um índice de validação de agrupamento, o *ARI* (HUBERT; ARABIE, 1985). O tamanho do conjunto de soluções é controlado por um limiar do valor do *ARI*.

Este algoritmo recebe como entrada um conjunto de partições base, geradas através da aplicação de diversos algoritmos tradicionais de agrupamento. As partições mais evidentes são, então, adicionadas ao conjunto reduzido (uma partição é considerada evidente quando o número de partições idênticas a ela é maior do que o número de algoritmos de agrupamento usado para compor o conjunto de partições base). Na sequência do algoritmo, são selecionadas as partições que possuem os maiores *ARIs* médios entre as partições ainda presentes no *pool* de partições e as partições similares as selecionadas são eliminadas. O Apêndice D apresenta uma descrição detalhada do passo a passo do *ASA*.

Figura 10 – Diagrama de seleção de partições: conjunto completo de partições $\Pi_C = \{\pi^1, \pi^2, ..., \pi^n\}$ e conjunto reduzido de partições $\Pi_R = \{\pi^1, \pi^2, ..., \pi^m\}$.



Fonte: base de dados extraída de <http://scikit-learn.org/stable/_images/sphx_glr_plot_cluster_comparison_0011.png> (acessado em 14/12/2017).

3 *Hybrid Selection Strategy* (*HSS*)

Conforme descrito nos capítulos anteriores, a área de agrupamento de dados apresenta inúmeros desafios, dentre os quais está a identificação de um conjunto reduzido de partições que apresente diferentes visões sobre uma determinada base de dados. A identificação desse conjunto de partições impõe, por si própria, uma série de problemas para os quais se busca respostas. A proposta da técnica desenvolvida nessa dissertação - o *HSS* - surgiu como uma tentativa de explorar algumas soluções para os problemas acima descritos.

Este capítulo descreve o algoritmo proposto neste trabalho, o *Hybrid Selection Strategy* (*HSS*) (ANTUNES; FACELI; SAKATA, 2017), um algoritmo de seleção de partições de agrupamento híbrido, dado que ele combina as técnicas de agrupamento multiobjetivo e de seleção de partições na tentativa de obter resultados mais robustos. O *HSS* é apresentado na Seção 3.1. A Seção 3.2 apresenta uma descrição detalhada de cada uma das três etapas do algoritmo.

3.1 Algoritmo de Seleção *HSS*

O *HSS* segue o preceito de que não há conhecimento sobre nenhuma das estruturas presentes nos dados, ou seja, ele é um algoritmo de seleção de partições não-supervisionado. Além disso, foi desenvolvido a partir do princípio de que é possível identificar mais de uma partição relevante para os dados.

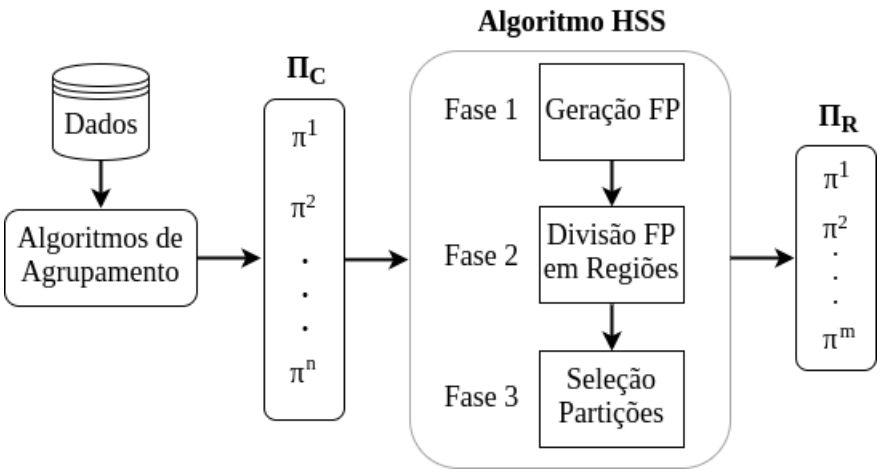
Foram estabelecidos três requisitos para o *HSS*: (i) número reduzido de soluções - dado o conjunto $\Pi_C = \{\pi^1, ..., \pi^C\}$ (conjunto completo), o intuito é identificar um subconjunto $\Pi_R = \{\pi^1, ..., \pi^R\}$ (conjunto reduzido), tal que, $\Pi_R \subset \Pi_C$ e $|\Pi_R| \ll |\Pi_C|$; (ii) qualidade - uma partição $\pi^i \in \Pi_R$ se e somente se é uma partição de alta qualidade de acordo com um ou mais critérios; (iii) dissimilaridade - uma partição $\pi^i \in \Pi_R$ se e somente se ela é tão dissimilar quanto possível de $\pi^j \in \Pi_R$, $\forall \pi^j \neq \pi^i$.

O *HSS* toma como entrada uma base de dados e um conjunto grande e diverso de partições, Π_C, produzido a partir da base de dados em questão. Estas partições podem ser geradas através de um conjunto de algoritmos tradicionais de agrupamento, preferencialmente algoritmos complementares, ou seja, aqueles que adotam diferentes critérios de agrupamento, possibilitando, assim, a identificação de uma maior variedade de *clusters* nas partições geradas. A saída do *HSS* é um conjunto reduzido de partições, Π_R.

A Figura 11 expõe o diagrama do *HSS*. Conforme supracitado, uma base de dados é tomada como entrada por um conjunto de algoritmos de agrupamento; as partições

geradas por estes são tomadas como entrada pelo *HSS*, que é dividido em três etapas - (i) geração de uma aproximação da Fronteira de Pareto, (ii) divisão desta aproximação em regiões e (iii) seleção de partições - retorna um conjunto reduzido de partições, Π_R.

Figura 11 – Diagrama do algoritmo *HSS*.



O pseudocódigo do *HSS* é exibido no Algoritmo 1 e pode ser tripartido conforme descrito a seguir:

1. Calcula a variância intra-cluster e a conectividade para cada partição π^i de Π_C ($1 \leq i \leq |\Pi_C|$) e as usa como entrada para o algoritmo de Pareto (linhas 1 a 5).
2. Divide o conjunto não-dominado P, obtido no Passo 1, em nR regiões, de Ω^1 a Ω^{nR}: $\bigcup_{i=1}^{nR} \Omega^i = P$ (linhas 6 e 7).
3. Seleciona uma partição de boa qualidade por região, retornando um conjunto reduzido Π_R (linhas 8 a 15).

3.2 Etapas do Algoritmo

Cada uma das etapas do *HSS* e seu custo computacional serão descritos detalhadamente a seguir. A base de dados artificial `monkey`[1] (Figura 12) será usada para exemplificar os passos do algoritmo. Ela possui 4000 objetos, apresenta um conjunto de *clusters* com diferentes tamanhos e formas e três partições conhecidas com diferentes níveis de refinamento.

3.2.1 Fase 1: Geração da Fronteira de Pareto

O *HSS* recebe como entrada um conjunto diverso de partições geradas através da execução de algoritmos tradicionais de agrupamento. Essas partições são tomadas

[1] Disponível em <http://lasid.sor.ufscar.br/clustersEvaluationBenchmark>, acessado em 15/12/2017.

Algoritmo 1 Hybrid Selection Strategy (HSS)

Entrada:
Π_C (conjunto completo), *ds* (base de dados), *porcentagem* (ver Subseção 3.2.2), *alg* (algoritmo de agrupamento)

Saída:
Π_R (conjunto reduzido)

1: **para cada** $\pi^i \in \Pi_C$ **faça**
2: $var(\pi^i) \leftarrow variancia(\pi^i, ds)$ ▷ $O(n^2 d)$
3: $con(\pi^i) \leftarrow conectividade(\pi^i, ds)$ ▷ $O(n^2 logn)$
4: **fim para**
5: $P \leftarrow fronteiraPareto(var, con)$ ▷ $O(|\Pi_C|^2)$
6: $nR \leftarrow calculaNR(porcentagem)$ ▷ $O(1)$
7: $R \leftarrow divisaoRegioes(P, nR, alg)$ ▷ $O(|\Pi_C|^2)$
8: **para cada** $\Omega^i \in R$ **faça**
9: **para cada** $\pi^j \in \Omega^i$ **faça**
10: $ari_m(\pi^j) \leftarrow 1/|\Omega^i| \sum_{\pi^k \in \Omega^i} ari(\pi^j, \pi^k)$ ▷ $O(|\Pi_C| n^2)$
11: **fim para**
12: Seleciona π^m com max ari_m
13: $\Pi_R \leftarrow \Pi_R \cup \pi^m$
14: **fim para**
15: **retorne** Π_R

como entrada pelo algoritmo de Pareto adotado, proposto em Mishra e Harit (2010). Este tem por desígnio identificar um conjunto não-dominado de soluções para o problema de otimização multiobjetivo. O conceito de Fronteira de Pareto é adotado devido a sua capacidade, dado o número apropriado de funções objetivo, de garantir a diversidade das partições presentes na FP (quanto mais distantes duas partições estão na FP, mais dissimilares elas são).

O *HSS* trabalha com a otimização de dois objetivos (isto é, índices de validação de agrupamento). Os dois critérios de otimização, escolhidos empiricamente por apresentarem melhores resultados, refletem aspectos distintos de uma boa solução de agrupamento: a variância intra-cluster é usada para analisar a compactação dos *clusters* e a conectividade para avaliar a separação entre *clusters*. Outros índices testados nesta etapa foram Calinski-Harabasz, silhueta e Dunn.

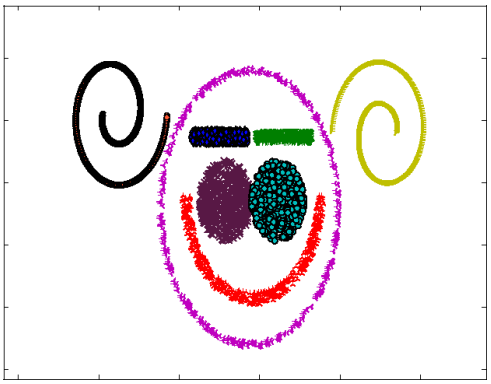
A variância intra-cluster mede a qualidade de uma partição em termos de compactação e homogeneidade. É calculada como a soma total das distâncias entre os objetos e o centro de seus *clusters*, conforme apresentado na Equação 3.1.

$$var(\pi) = \sqrt{\frac{1}{|\pi|} \sum_{c_k \in \pi} \sum_{x_i \in C_k} \delta(x_i, \mu_k)}, \tag{3.1}$$

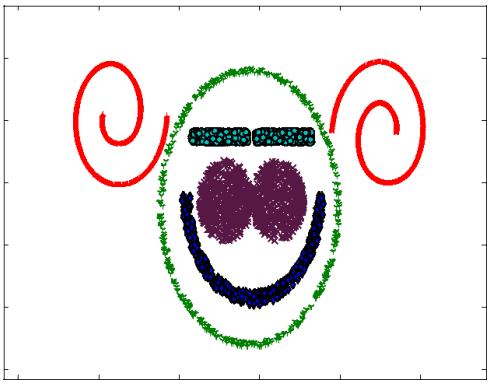
Nessa equação, π denota uma partição (conjunto de *clusters*), x_i é o objeto perten-

Figura 12 – Estruturas conhecidas para a base de dados `monkey`.

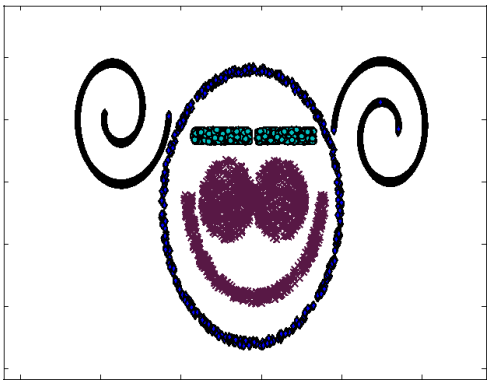
(a) `monkey-Real-k8`



(b) `monkey-Real-k5`



(c) `monkey-Real-k3`



cente ao *cluster* c_k, μ_k é o centroide do *cluster* c_k e $\delta(.,.)$ é a função de distância escolhida (no caso do *HSS*, a distância Euclidiana).

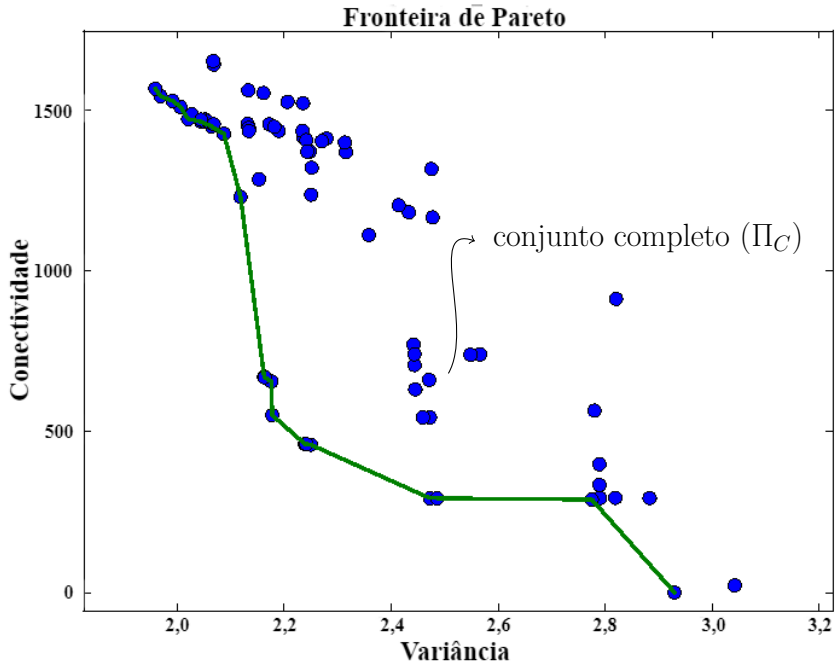
A conectividade, relacionada aos conceitos de encadeamento e ligação, avalia o grau com que objetos vizinhos são colocados em um mesmo *cluster*. É calculada conforme a Equação 3.2, em que nn_{ij} é o j-ésimo vizinho mais próximo do objeto x_i, v é o parâmetro que determina o número de vizinhos mais próximos que contribuem para a conectividade, π é uma partição e c_k é um *cluster* que pertence a π.

$$
\begin{aligned}
con(\pi) &= \sum_{i=1}^{n}\sum_{j=1}^{v} f(x_i, nn_{ij}) \\
f(x_i, nn_{ij}) &= \begin{cases} \frac{1}{j}, & \text{if } \nexists c_k : x_i, nn_{ij} \in c_k \\ 0, & \text{caso contrário} \end{cases}
\end{aligned}
\tag{3.2}
$$

A Figura 13 apresenta, como pontos azuis, todas as partições de entrada empregadas pelo *HSS* de acordo com seus valores de variância e conectividade (Algoritmo 1, linhas de 1 a 4): 76 partições, geradas através da aplicação dos algoritmos SL, AL, CoL, CeL, KM e SNN, com o número de *clusters* $k \in \{2, 3, ..., 16\}$ - o valor de k empregado varia de 2 a

$2K$, em que K é o número de *clusters* na partição conhecida. A linha verde traçada liga as partições pertencentes à aproximação da FP identificadas pelo Algoritmo 1, linha 5. Estas partições apresentam o melhor compromisso entre as medidas de variância e conectividade - sendo conhecidas como soluções não-dominadas ou soluções Pareto-ótimas (ver Definição 2). O Apêndice C exibe as partições presentes na aproximação da FP gerada pelo *HSS* para outras bases de dados.

Figura 13 – Partições geradas pelos algoritmos SL, AL, CoL, CeL, KM e SNN para a base de dados `monkey`. Cada ponto representa uma partição de Π_C. A linha verde conecta as partições não-dominadas presentes na aproximação da FP.



3.2.2 Fase 2: Divisão da Fronteira de Pareto em Regiões

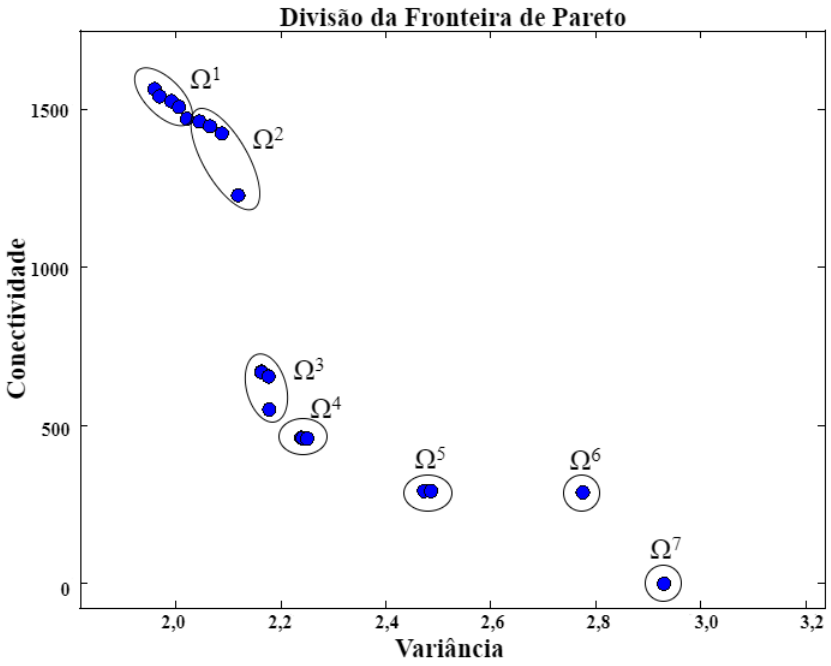
A Fase 2 é responsável pela divisão do conjunto não-dominado obtido na Fase 1 em nR regiões, de Ω^1 a Ω^{nR}, para obter uma distribuição do conjunto de objetivos, mantendo, assim, a diversidade do conjunto de soluções. É importante destacar, porém, que embora a diversidade entre as partições seja garantida, a distância entre as partições vizinhas selecionadas pode não ser uniforme. A única garantia fornecida é que cada região retorna somente uma solução, porém, na prática, duas partições, cada qual presente em uma região, podem estar bem próximas uma da outra.

O valor de nR é obtido através de um parâmetro - *porcentagem* - informado pelo usuário na execução do *HSS*. *porcentagem* determina a porcentagem de partições presentes na aproximação da FP que devem se tornar regiões, seu valor varia entre 0 e 1. No caso da base de dados `monkey`, por exemplo, que possui 23 partições na aproximação da FP, se o valor de *porcentagem* informado for $0,3$ (estudos empíricos mostraram que $0,3$ apresenta um bom resultado no balanceamento entre número de soluções e qualidade das soluções

selecionadas), o número de regiões no qual a aproximação da FP deve ser dividida é 30% de 23, ou seja, 7.

Uma vez determinado o valor de nR, é aplicado um algoritmo de agrupamento, também tomado como parâmetro, ao conjunto não-dominado P, com o objetivo de dividir P em nR *clusters* (regiões). Os algoritmos de agrupamento disponíveis para esta etapa são: k-médias, *single-link*, *complete-link*, *average-link* e *ward*. A Figura 14 ilustra a divisão feita pelo Algoritmo 1, linha 7, para as partições presentes na aproximação da FP, P (sequência da Figura 13). Esta foi dividida em sete regiões (Ω^1,...,Ω^7), o algoritmo de agrupamento adotado foi o k-médias e o valor de *porcentagem* foi $0,3$.

Figura 14 – Divisão da Fronteira de Pareto em regiões $(\Omega^1, ..., \Omega^7)$ para a base de dados artificial `monkey`.



3.2.3 Fase 3: Seleção de Partições

Na última etapa, Fase 3, é aplicada uma estratégia de seleção simples, calculada de acordo com a Equação 3.3, a cada região Ω^i $(1 \leq i \leq nR)$ da FP. Nesta equação, $|\Omega^i|$ é o número de partições em Ω^i e $ari_m(\pi_j)$ é o ARI médio entre π^j e todo π^k, com $\pi^j, \pi^k \in \Omega^i$ e $\pi^k \neq \pi^j$.

$$\max_{1 \leq j \leq |\Omega^i|} (ari_m(\pi^j))$$
$$ari_m(\pi^j) = 1/|\Omega^i| \sum_{\pi^k \in \Omega^i, \pi^k \neq \pi^j} ari(\pi^j, \pi^k) \tag{3.3}$$

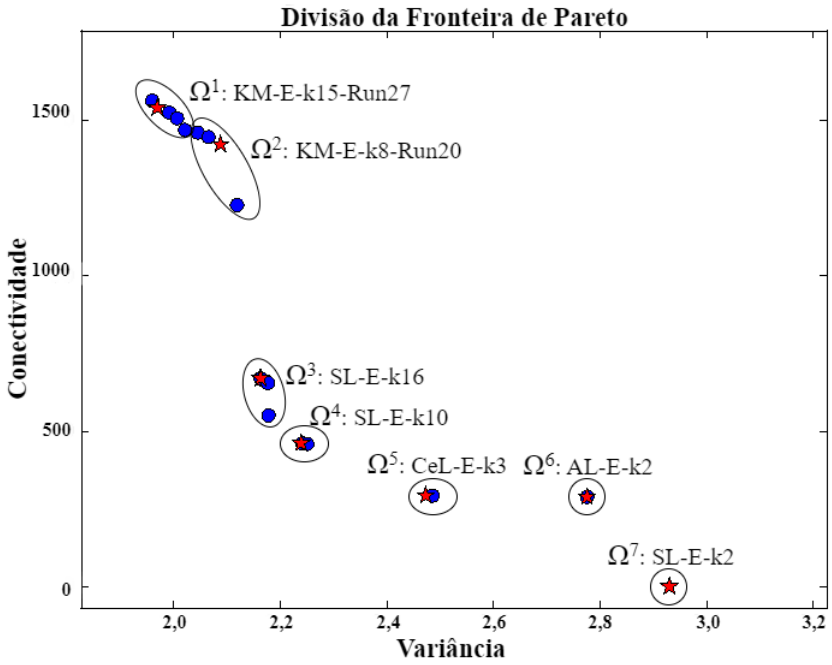
O ARI é aplicado nesta equação para mensurar a similaridade entre as partições presentes em uma mesma região da FP. Quanto maior o valor do ARI, maior a similaridade

entre as partições sendo analisadas. A partição com maior *ARI* médio é selecionada, pois ela é considerada a mais parecida com as demais partições da região em questão.

O *ARI* é a versão normalizada do *Rand Index* (HUBERT; ARABIE, 1985). Seu valor varia entre -1 e 1, com 1 indicando que as partições sendo comparadas são idênticas, possui valores próximos a 0 ou negativos para partições aleatórias. Uma das vantagens deste índice é que, diferentemente da grande maioria dos demais índices de validação, ele não é tendencioso em relação a um determinado algoritmo ou número de *clusters* na partição em questão (HUBERT; ARABIE, 1985). O Apêndice B traz uma descrição detalhada do *ARI* e da forma com que ele é calculado.

Para garantir a diversidade, somente uma partição é selecionada por região (partições próximas na aproximação da FP são consideradas similares). A Figura 15 mostra a seleção de uma partição para cada região da aproximação da FP exibida na Figura 14 para o exemplo com a base de dados `monkey`. Já a Figura 16 mostra as partições selecionadas. No Capítulo 4, há uma descrição completa dos experimentos.

Figura 15 – Seleção de partições: as estrelas vermelhas representam as partições selecionadas pelo *HSS* para cada região $(\Omega^1, ..., \Omega^7)$ para a base de dados artificial `monkey`.



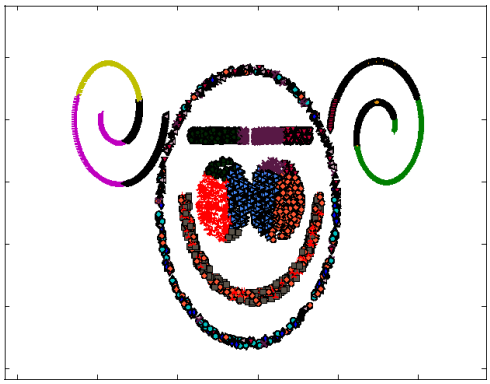
3.2.4 Custo Computational

O custo computacional do *HSS* (Algoritmo 1) pode ser dividido em 3 partes. A complexidade da Fase 1 (linhas 1 a 5) é composta pela complexidade do cálculo, para cada partição, da variância, $O(n^2d)$ (linha 2), e da conectividade, $O(n^2logn)$ (linha 3), somado à complexidade da obtenção da FP (linha 5), $O(|\Pi_C|^2)$, ou seja, $O(|\Pi_C|(n^2d+n^2logn)+|\Pi_C|^2)$,

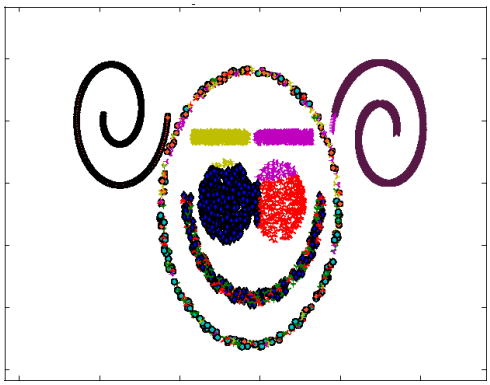
em que $|\Pi_C|$ é o número de partições em Π_C, n é o número de objetos na base de dados e d é o número de atributos. A complexidade da Fase 2 (linhas 6 e 7) depende do algoritmo escolhido para a divisão da FP em regiões: $O(|\Pi_C|^2)$ para SL, CoL, AL ou ward e $O(nR|\Pi_C|Id)$ para KM, em que nR é o número de regiões em que a aproximação da FP deve ser dividida e I é o número de iterações ($I \leq 300$ na implementação do KM adotada). Finalmente, a complexidade da Fase 3 (linhas 8 a 15) é a complexidade do cálculo do ARI médio entre cada partição e as demais partições da região, $O(|\Pi_C|n^2)$, multiplicado pelo número de partições na região, que no pior caso é $|\Pi_C|$, $O(|\Pi_C|^2 n^2)$.

Figura 16 – Partições selecionadas pelo *HSS* para a base de dados `monkey`.

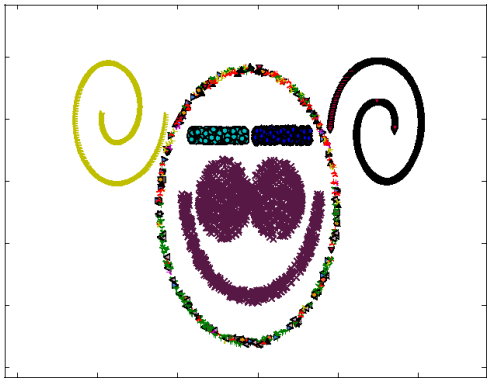
(a) `monkey-KM-E-k15-Run27`



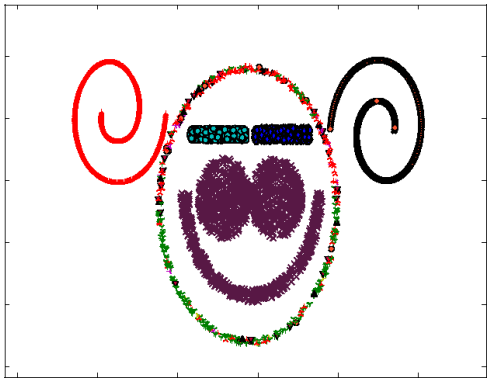
(b) `monkey-KM-E-k8-Run20`



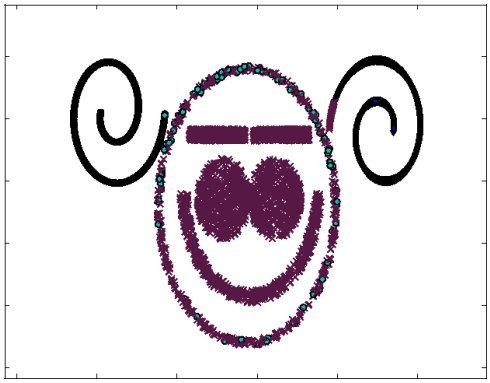
(c) `monkey-SL-E-k16`



(d) `monkey-SL-E-k10`

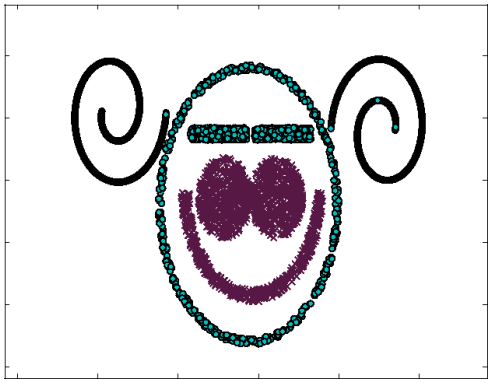


(e) `monkey-CeL-E-k3`



(f) `monkey-AL-E-k2`

(g) `monkey-SL-E-k2`



4 Experimentos e Resultados

Neste capítulo, o algoritmo *HSS*, apresentado no Capítulo 3, será avaliado através de análises experimentais. Os experimentos foram concebidos de forma a avaliar o desempenho do *HSS* em diversos contextos. Primeiramente, é feita uma comparação do *HSS* com o algoritmo de seleção *ASA* em relação ao número de partições selecionadas e a qualidade dessas partições quando comparadas às estruturas conhecidas (partições reais). Então, é investigado o comportamento do *HSS* como ferramenta de pós-processamento para os algoritmos *MOCK* e *MOCLE*. O intuito de aplicar o *HSS* às partições geradas por *MOCK* e *MOCLE* é reduzir o número de partições retornas ao usuário final, uma vez que estas técnicas retornam um número elevado de partições, amiúde redundantes. Finalmente, é estudado o desempenho do *HSS* frente a algoritmos de seleção de *ensemble*.

Três pontos foram avaliados durante a execução dos experimentos - número de soluções, qualidade das partições retornadas e dissimilaridade entre tais partições - de forma a atender os três principais requisitos do *HSS* (Seção 3.1).

Segue a organização deste capítulo. Na Seção 4.1, estão descritas as bases de dadas empregadas na execução dos experimentos realizados. A Seção 4.2 apresenta a metodologia empregada na realização dos experimentos. A Seção 4.3 discute os resultados dos estudos empíricos para os parâmetros *porcentagem* e *alg* do *HSS*. Já a Seção 4.4 discorre detalhadamente, com o auxílio de tabelas e gráficos, sobre os resultados apresentados pelo *HSS* quando comparado com outras técnicas de seleção de partições e, também, quando aplicado como uma ferramenta de pós-processamento para os algoritmos multiobjetivos *MOCK* e *MOCLE*.

4.1 Bases de Dados

Para a realização dos experimentos, foram empregadas 15 bases de dados, 11 artificiais e 4 reais, descritas na Tabela 1. As bases escolhidas apresentam características distintas, tais como diferentes números de atributos e quantidade variada de estruturas conhecidas. `dyrskjot` e `golub`, por exemplo, contêm um baixo número de objetos, mas um número elevado de atributos. As bases `ds2c2sc13`, `ds3c3sc6`, `ds4c2sc8`, `monkey`, `spiralsquare`, `glass` e `golub` têm mais de uma estrutura conhecida (duas ou três). A Figura 17 exibe as bases de dados com até três dimensões. Como é possível observar na figura, as bases de dados possuem diferentes formas e, muitas vezes, estruturas heterogêneas, mais difíceis de serem identificadas pelos algoritmos de agrupamento tradicionais.

Tabela 1 – Principais características das bases de dados usadas nos experimentos: número de objetos n, número de atributos d, número de estruturas conhecidas n^E e número de *clusters* K para cada estrutura E_i.

Tipo	**Base de dados**	n	d	n^E	K	**Fonte**
Artificial	`Aggregation`	788	2	1	7	(GIONIS; MANNILA; TSAPARAS, 2007)
	`chainlink`	1000	3	1	2	(ULTSCH, 2005)
	`Compound`	399	2	1	6	(ZAHN, 1971)
	`ds2c2sc13`	588	2	3	2, 5, 13	(FACELI; SAKATA, 2016)
	`ds3c3sc6`	905	2	2	3, 6	(FACELI; SAKATA, 2016)
	`ds4c2sc8`	485	2	2	2, 8	(FACELI; SAKATA, 2016)
	`Flame`	240	2	1	2	(FU; MEDICO, 2007)
	`monkey`	4000	2	3	8, 5, 3	(FACELI; SAKATA, 2016)
	`R15`	600	2	1	15	(VEENMAN; REINDERS; BACKER, 2002)
	`spiral`	312	2	1	3	(CHANG; YEUNG, 2008)
	`spiralsquare`	2000	2	2	2, 6	(FACELI; SAKATA, 2016)
Real	`dyrskjot`	40	1203	1	3	(DYRSKJØT et al., 2003)
	`glass`	214	9	3	2, 5, 6	(LICHMAN, 2013)
	`golub`	72	3571	2	2, 4	(GOLUB et al., 1999)
	`iris`	150	4	1	3	(LICHMAN, 2013)

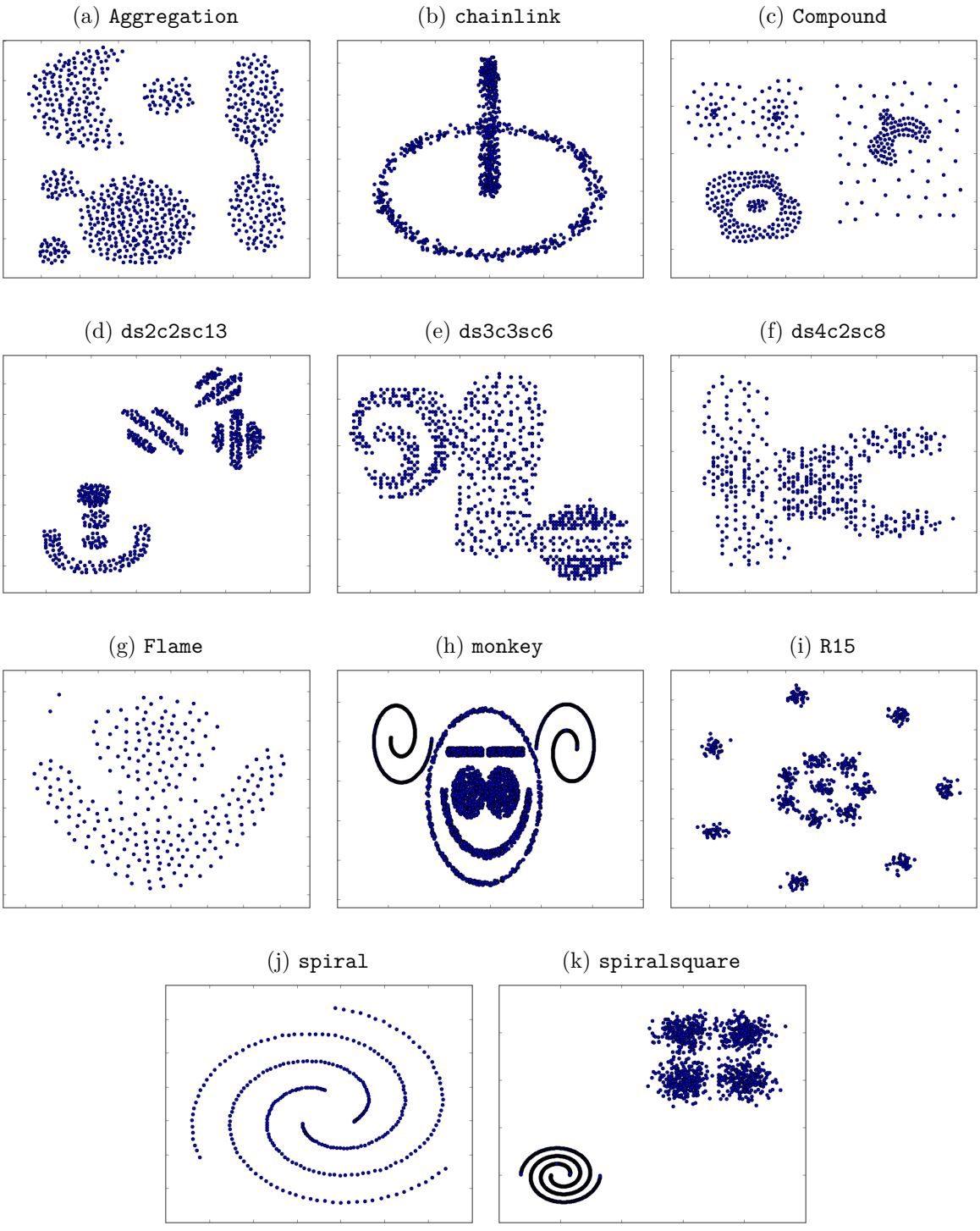
4.2 Metodologia Experimental

Para geração do conjunto de partições base, Π_C, tomado como entrada por todos os algoritmos usados nas análises experimentais (descritas na Seção 4.4), os algoritmos k-médias (KM), *single-link* (SL), *complete-link* (CoL), *centroid-link* (CeL), *average-link* (AL) e *shared nearest neighbor* (SNN) foram executados com o número de *clusters*, k, variando entre 2 e $2K$, em que K é o número de *clusters* da partição conhecida que contém o maior número de *clusters*. A medida de proximidade empregada foi a distância Euclidiana.

O k-médias foi executado 30 vezes para cada k e a partição escolhida foi a que apresentou o menor erro quadrático. O SNN foi executado com diferentes configurações de parâmetros: (i) número de vizinhos a serem considerados $NN = 2\%$, 5%, 10%, 20%, 30% e 40% de n; (ii) parâmetros relacionados ao peso das conexões do grafo dos vizinhos mais próximos compartilhados ou "similaridade SNN" $strong = -1$ (número de ligações fracas), $merge = 40\%$, 60% e 80% (porcentagem de ligações usadas para a junção de *clusters*) e $label = 0$ (todos os pontos devem ser rotulados); e (iii) parâmetros relacionados ao número de conexões fortes ou "densidade SNN" $topic = 20\%$, 50% e 80% (porcentagem de pontos representativos) e $noise = 0$ (porcentagem de pontos ruidosos a serem excluídos).

Para a execução do *MOCK*, foi empregado o programa disponível em <http://personalpages.manchester.ac.uk/mbs/julia.handl/mock.html> com a seguinte configuração:

Figura 17 – Conjuntos de dados empregados com até três dimensões.



(i) para a função de distância, foi empregada a padrão, Euclidiana; (ii) para o número de vizinhos mais próximos que o *MOCK* usa para calcular a conectividade, também foi adotado o valor padrão, 10; (iii) para o número máximo de *clusters*, foi adotado $2K$ (o mesmo valor usado para a geração das partições base aplicadas diretamente no *HSS*); (iv) para o número de superfícies de controle também foi adotado o padrão, 1.

Já para a execução do *MOCLE* foi empregado o programa disponível em <http:

//lasid.sor.ufscar.br/mocleproject/>, com a seguinte configuração: (i) para o operador de cruzamento, foi empregado o padrão, *Meta-Clustering Algorithm* (*MCLA*); (ii) para o número de vizinhos, foi empregado o padrão, 5% do tamanho da base de dados.

Como os algoritmos *MOCK* e *MOCLE* não são determinísticos, eles foram executados 30 vezes com as mesmas configurações, o que gerou 30 conjuntos completos de soluções - $(\Pi_C^1, ..., \Pi_C^{30})$. O *HSS* foi executado em cada um desses conjuntos completos, resultando em 30 conjuntos reduzidos $(\Pi_R^1, ..., \Pi_R^{30})$ para *MOCK* e *MOCLE*.

Em todos os experimentos conduzidos, é feita uma comparação entre o número de soluções retornadas pelo *HSS* e pelas demais técnicas. Para aferir a qualidade das partições retornadas pelo *HSS*, e também pelos outros algoritmos abordados na Seção 4.4, é empregado o *ARI* (HUBERT; ARABIE, 1985).

4.3 Estudo Empírico dos Parâmetros do *HSS*

O *HSS* recebe quatro parâmetros: *dataset*, Π_C, *porcentagem* e *alg*. O parâmetro *dataset* é a base de dados. Π_C é o conjunto de partições geradas através de diferentes algoritmos de agrupamento para *dataset*. Conforme descrito na Seção 4.2, foram usados os seguintes algoritmos: SL, AL, CoL, CeL, KM e SNN. Nesta seção, são descritos os resultados dos estudos empíricos para os parâmetros *porcentagem* e *alg*.

4.3.0.1 Variação do Parâmetro *porcentagem*

O *HSS* toma como entrada um parâmetro chamado *porcentagem*, cujo valor deve estar entre $0,0$ e $1,0$. Este parâmetro representa a porcentagem de partições presentes na aproximação da FP que serão selecionadas pelo algoritmo e, consequentemente, o número de regiões no qual a aproximação da FP deve ser dividida. Para analisar o comportamento do *HSS* com a variação do parâmetro *porcentagem*, o *HSS* foi executado diversas vezes, cada vez com *porcentagem* assumindo um valor diferente no intervalo entre $0,1$ e $1,0$, com passo $0,1$.

Na Tabela 2 são exibidos os valores do *ARI* obtidos pelo *HSS* executado com diferentes valores de *porcentagem*: $porcentagem = 0,1$ (HSS_{10}), $porcentagem = 0,3$ (HSS_{30}), $porcentagem = 0,5$ (HSS_{50}) e $porcentagem = 1,0$ (HSS_{100}). Esses valores são exibidos por apresentarem as maiores variações de *ARI* dentro do intervalo testado (entre $0,1$ e $1,0$). Por uma questão de compreensibilidade, para cada base de dados e estrutura conhecida (E_i), é exibido o maior valor do *ARI*, ou seja, o *ARI* da melhor partição quando comparada com E_i para Π_C, $\mathbf{HSS_{10}}$, $\mathbf{HSS_{30}}$, $\mathbf{HSS_{50}}$ e $\mathbf{HSS_{100}}$.

Tabela 2 – Número de soluções e qualidade da melhor solução (*ARI*) retornada pelo *HSS* com diferentes valores de *percentage* (10, 30, 50 e 100). Os valores destacados em vermelho (▽) indicam piora maior do que 0,05 no valor do *ARI*.

Base de Dados	**Número de Soluções**					**ARI**					
	Π_C	**HSS$_{10}$**	**HSS$_{30}$**	**HSS$_{50}$**	**HSS$_{100}$**	**STR**	Π_C	**HSS$_{10}$**	**HSS$_{30}$**	**HSS$_{50}$**	**HSS$_{100}$**
Aggregation	74,00	4,00	10,00	16,00	21,00	E_1	0,99	▽ 0,73	0,99	0,99	0,99
chainlink	27,00	2,00	5,00	7,00	7,00	E_1	1,00	1,00	1,00	1,00	1,00
Compound	62,00	3,00	8,00	13,00	18,00	E_1	0,85	0,85	0,85	0,85	0,85
ds2c2sc13	147,00	6,00	18,00	30,00	39,00	E_1	1,00	1,00	1,00	1,00	1,00
						E_2	1,00	▽ 0,89	1,00	1,00	1,00
						E_3	1,00	▽ 0,77	▽ 0,87	▽ 0,87	▽ 0,87
ds3c3sc6	65,00	3,00	7,00	12,00	23,00	E_1	0,90	▽ 0,79	▽ 0,81	0,90	0,90
						E_2	0,59	▽ 0,49	▽ 0,50	0,58	0,59
ds4c2sc8	89,00	3,00	9,00	15,00	28,00	E_1	0,87	▽ 0,80	0,87	0,87	0,87
						E_2	0,35	▽ 0,29	▽ 0,29	▽ 0,29	0,30
dyrskjot	25,00	3,00	7,00	11,00	12,00	E_1	0,55	0,54	0,54	0,54	0,54
Flame	25,00	2,00	5,00	7,00	11,00	E_1	0,94	▽ 0,51	0,94	▽ 0,71	0,94
glass	55,00	3,00	9,00	15,00	28,00	E_1	0,67	▽ 0,59	0,65	0,65	0,67
						E_2	0,56	▽ 0,45	0,56	0,56	0,56
						E_3	0,26	▽ 0,20	0,26	0,26	0,26
golub	49,00	3,00	9,00	15,00	20,00	E_1	0,87	▽ 0,42	0,87	0,87	0,87
						E_2	0,94	▽ 0,26	▽ 0,61	▽ 0,88	▽ 0,88
iris	37,00	2,00	6,00	8,00	8,00	E_1	0,82	▽ 0,59	▽ 0,76	▽ 0,76	▽ 0,76
monkey	76,00	3,00	7,00	12,00	22,00	E_1	0,67	▽ 0,61	0,65	0,65	0,65
						E_2	0,83	▽ 0,67	▽ 0,70	▽ 0,70	▽ 0,71
						E_3	0,65	▽ 0,58	▽ 0,58	▽ 0,58	▽ 0,58
R15	154,00	8,00	23,00	38,00	58,00	E_1	0,99	0,98	0,99	0,99	0,99
spiral	33,00	2,00	6,00	9,00	15,00	E_1	1,00	▽ 0,92	▽ 0,92	▽ 0,92	0,98
spiralsquare	85,00	5,00	15,00	20,00	20,00	E_1	0,67	0,63	0,67	0,67	0,67
						E_2	1,00	1,00	1,00	1,00	1,00
Média	**66,87**	**3,47**	**9,60**	**15,20**	**22,00**		**0,80**	**0,66**	**0,76**	**0,76**	**0,78**

Como é possível observar, em todos os casos ($\mathbf{HSS_{10}}$, $\mathbf{HSS_{30}}$, $\mathbf{HSS_{50}}$ e $\mathbf{HSS_{100}}$) houve uma redução significativa no número de partições retornadas enquanto a queda na qualidade (aqui representada através do ARI) foi relativamente pequena para valores de *porcentagem* maiores ou iguais a $0,3$ ($\mathbf{HSS_{30}}$, $\mathbf{HSS_{50}}$ e $\mathbf{HSS_{100}}$). O $\mathbf{HSS_{10}}$ conseguiu identiticar a melhor estrutura presente em Π_C em 4 casos, ambos $\mathbf{HSS_{30}}$ e $\mathbf{HSS_{50}}$ identificaram a melhor estrutura em 13 casos e o $\mathbf{HSS_{100}}$ obteve sucesso em 16 casos.

Tanto o $\mathbf{HSS_{30}}$ quanto o $\mathbf{HSS_{50}}$ obtiveram o mesmo desempenho em termos de ARI médio, porém o $\mathbf{HSS_{30}}$ retornou, na média, somente $9,6$ soluções, contra $15,2$ do $\mathbf{HSS_{50}}$. O número de partições retornadas está limitado ao número de partições distintas presentes na FP, por isso nos casos de `chainlink`, `iris` e `spiralsquare` o número de partições retornadas para $porcentagem = 0,5$ e $porcentagem = 1$ são os mesmos.

A redução no número de soluções retornadas por $\mathbf{HSS_{10}}$ em relação a Π_C foi de $94,81\%$, enquanto a queda na qualidade foi de $17,5\%$. No caso do $\mathbf{HSS_{30}}$, a redução no número de soluções retornadas em relação a Π_C foi de $85,64\%$, enquanto a queda na qualidade foi de 5%. Já no caso do $\mathbf{HSS_{50}}$, a redução no número de soluções retornadas em relação a Π_C foi de $77,16\%$, enquanto a queda na qualidade foi de 5%. Finalmente, a redução no número de soluções retornadas por $\mathbf{HSS_{100}}$ em relação a Π_C foi de $67,1\%$, enquanto a queda na qualidade foi de $2,5\%$.

Para os experimentos realizados neste trabalho, levando em consideração as informações providas acima, foi adotado $porcentagem = 0,3$, uma vez que este apresentou o melhor compromisso entre número de soluções retornadas e ARI.

4.3.0.2 Variação do Parâmetro *alg*

O parâmetro *alg* representa o algoritmo de agrupamento usado na divisão da aproximação da FP em regiões. A Tabela 3 exibe uma comparação dos resultados obtidos pelo *HSS* com os algoritmos de agrupamento disponíveis: KM ($\mathrm{HSS_{KM}}$), SL ($\mathrm{HSS_{SL}}$), CoL ($\mathrm{HSS_{CoL}}$), AL ($\mathrm{HSS_{AL}}$) e ward ($\mathrm{HSS_{ward}}$).

Tabela 3 – Número de soluções e qualidade da melhor solução (ARI) retornada por Π_C e HSS com a variação do parâmetro alg: KM, SL, CoL, AL, ward (HSS_{KM}, HSS_{SL}, HSS_{CoL}, HSS_{AL}, HSS_{ward}). O número de soluções retornada por todas essas variações é a mesma e está na coluna HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI.

Base de Dados	**Número de Soluções**		**ARI**						
	Π_C	**HSS**	**STR**	Π_C	**HSS$_{KM}$**	**HSS$_{SL}$**	**HSS$_{CoL}$**	**HSS$_{AL}$**	**HSS$_{ward}$**
Aggregation	74.00	10.00	E_1	0.99	0.99	$\triangledown$ 0.79	0.99	0.99	0.99
chainlink	27.00	5.00	E_1	1.00	1.00	1.00	1.00	1.00	1.00
Compound	62.00	8.00	E_1	0.85	0.85	0.84	0.85	0.85	0.85
ds2c2sc13	147.00	18.00	E_1	1.00	1.00	1.00	1.00	1.00	1.00
			E_2	1.00	1.00	$\triangledown$ 0.92	1.00	$\triangledown$ 0.92	1.00
			E_3	1.00	$\triangledown$ 0.87	$\triangledown$ 0.87	$\triangledown$ 0.87	$\triangledown$ 0.87	$\triangledown$ 0.87
ds3c3sc6	65.00	7.00	E_1	0.90	$\triangledown$ 0.81	$\triangledown$ 0.81	$\triangledown$ 0.79	$\triangledown$ 0.81	$\triangledown$ 0.79
			E_2	0.59	$\triangledown$ 0.50	$\triangledown$ 0.50	$\triangledown$ 0.49	$\triangledown$ 0.50	$\triangledown$ 0.49
ds4c2sc8	89.00	9.00	E_1	0.87	0.87	0.87	0.87	0.87	0.87
			E_2	0.35	$\triangledown$ 0.29	$\triangledown$ 0.18	$\triangledown$ 0.29	$\triangledown$ 0.29	$\triangledown$ 0.29
dyrskjot	25.00	7.00	E_1	0.55	0.54	0.54	0.54	0.54	0.54
Flame	25.00	5.00	E_1	0.94	0.94	$\triangledown$ 0.71	$\triangledown$ 0.55	$\triangledown$ 0.71	$\triangledown$ 0.55
glass	55.00	9.00	E_1	0.67	0.65	0.65	0.65	0.65	0.65
			E_2	0.56	0.56	0.56	0.56	0.56	0.56
			E_3	0.26	0.26	0.26	0.26	0.26	0.26
golub	49.00	9.00	E_1	0.87	0.87	0.87	0.87	0.87	$\triangledown$ 0.80
			E_2	0.94	$\triangledown$ 0.61	$\triangledown$ 0.61	$\triangledown$ 0.61	$\triangledown$ 0.61	$\triangledown$ 0.56
iris	37.00	6.00	E_1	0.82	$\triangledown$ 0.76	$\triangledown$ 0.76	$\triangledown$ 0.76	$\triangledown$ 0.76	$\triangledown$ 0.76
monkey	76.00	7.00	E_1	0.67	0.65	$\triangledown$ 0.61	0.62	$\triangledown$ 0.61	0.65
			E_2	0.83	$\triangledown$ 0.70	$\triangledown$ 0.70	$\triangledown$ 0.67	$\triangledown$ 0.70	$\triangledown$ 0.70
			E_3	0.65	$\triangledown$ 0.58	$\triangledown$ 0.58	$\triangledown$ 0.58	$\triangledown$ 0.58	$\triangledown$ 0.58
R15	154.00	23.00	E_1	0.99	0.99	0.99	0.99	0.99	0.99
spiral	33.00	6.00	E_1	1.00	$\triangledown$ 0.92	$\triangledown$ 0.92	$\triangledown$ 0.92	$\triangledown$ 0.92	$\triangledown$ 0.92
spiralsquare	85.00	15.00	E_1	0.67	0.67	0.67	0.67	0.67	0.67
			E_2	1.00	1.00	1.00	1.00	1.00	1.00
Média	**66.87**	**9.60**		**0.80**	**0.76**	**0.73**	**0.74**	**0.74**	**0.73**

Como pode ser observado na Tabela 3, o KM foi o algoritmo que apresentou o melhor *ARI*, $0,76$. Por isso, ele foi o algoritmo adotado nos experimentos conduzidos neste trabalho.

4.4 Resultados

Todos os experimentos foram conduzidos empregando as bases de dados apresentadas na Seção 4.1. O conjunto de partições base, Π_C, foi gerado conforme descrito na Seção 4.2.

Nesta seção, primeiro é descrita a comparação do desempenho do *HSS* e do *ASA*. Então, são apresentados os resultados obtidos pelo *HSS* como ferramenta de pós-processamento para os algoritmos *MOCK* e *MOCLE*. Finalmente, é exibida uma comparação do *HSS* com as estratégias de seleção de *ensemble SR*, *BRP*, *SRD* e *Diversity*. Todas as análises consideram a quantidade de redução no número de partições retornadas por cada técnica e a qualidade destas partições, inspecionada através do *ARI* da melhor solução em cada caso.

4.4.1 *ASA* e *HSS*

O objetivo desta subseção é apresentar uma análise do desempenho e resultados obtidos pelo *HSS* como um algoritmo de seleção de partições e verificar o comportamento dele frente ao algoritmo de seleção *ASA*, descrito na Seção 2.5.1.

A Tabela 4 mostra os resultados obtidos por *ASA* e *HSS*, tomando como entrada o mesmo conjunto de partições base, Π_C. O objetivo de ambos algoritmos é retornar um subconjunto de Π_C chamado de Π_R. A tabela exibe o número de partições presentes em Π_C, em Π_R do *ASA* e em Π_R do *HSS*. Por uma questão de compreensibilidade, para cada base de dados e estrutura conhecida (E_i), é exibido o maior valor do *ARI*, ou seja, o *ARI* da melhor partição quando comparada com E_i para Π_C, Π_R do *ASA* e Π_R do *HSS*. Como ambas as técnicas, *HSS* e *ASA*, não geram partições novas, mas somente selecionam um subconjunto das partições tomadas como entrada, a qualidade do Π_C é um fator determinante na qualidade do conjunto reduzido de soluções encontrado por ambas as técnicas.

Para 10 das 25 estruturas apresentadas na Tabela 4, o *HSS* superou o *ASA*, enquanto o *ASA* superou o *HSS* em 6 casos. Para os 9 casos restantes ambas as estratégias apresentaram o mesmo desempenho. Além disso, o *ARI* médio para Π_C, *ASA* e *HSS* foi $0,80$, $0,75$ e $0,76$ respectivamente.

É possível também observar na Tabela 4 que o *HSS* apresentou uma grande redução no número de partições retornadas: enquanto o *ASA* retornou, em média, $15,67$ partições,

Tabela 4 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, ASA e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI.

Base de Dados	**Número de Soluções**			**ARI**			
	Π_C	**ASA**	**HSS**	**STR**	Π_C	**ASA**	**HSS**
`Aggregation`	74,00	13,00	10,00	E_1	0,99	$\triangledown$ 0,81	0,99
`chainlink`	27,00	14,00	5,00	E_1	1,00	1,00	1,00
`Compound`	62,00	12,00	8,00	E_1	0,85	0,80	0,85
`ds2c2sc13`	147,00	18,00	18,00	E_1	1,00	1,00	1,00
				E_2	1,00	$\triangledown$ 0,93	1,00
				E_3	1,00	$\triangledown$ 0,83	$\triangledown$ 0,87
`ds3c3sc6`	65,00	25,00	7,00	E_1	0,90	0,90	$\triangledown$ 0,81
				E_2	0,59	0,59	$\triangledown$ 0,50
`ds4c2sc8`	89,00	27,00	9,00	E_1	0,87	0,82	0,87
				E_2	0,35	0,35	$\triangledown$ 0,29
`dyrskjot`	25,00	5,00	7,00	E_1	0,55	$\triangledown$ 0,49	0,54
`Flame`	25,00	18,00	5,00	E_1	0,94	0,94	0,94
`glass`	55,00	9,00	9,00	E_1	0,67	0,67	0,65
				E_2	0,56	0,55	0,56
				E_3	0,26	0,26	0,26
`golub`	49,00	12,00	9,00	E_1	0,87	$\triangledown$ 0,65	0,87
				E_2	0,94	$\triangledown$ 0,88	$\triangledown$ 0,61
`iris`	37,00	7,00	6,00	E_1	0,82	$\triangledown$ 0,76	$\triangledown$ 0,76
`monkey`	76,00	24,00	7,00	E_1	0,67	0,65	0,65
				E_2	0,83	$\triangledown$ 0,67	$\triangledown$ 0,70
				E_3	0,65	$\triangledown$ 0,58	$\triangledown$ 0,58
`R15`	154,00	25,00	23,00	E_1	0,99	0,99	0,99
`spiral`	33,00	17,00	6,00	E_1	1,00	1,00	$\triangledown$ 0,92
`spiralsquare`	85,00	9,00	15,00	E_1	0,67	0,66	0,67
				E_2	1,00	1,00	1,00
Média	**66,87**	**15,67**	**9,60**		**0,80**	**0,75**	**0,76**

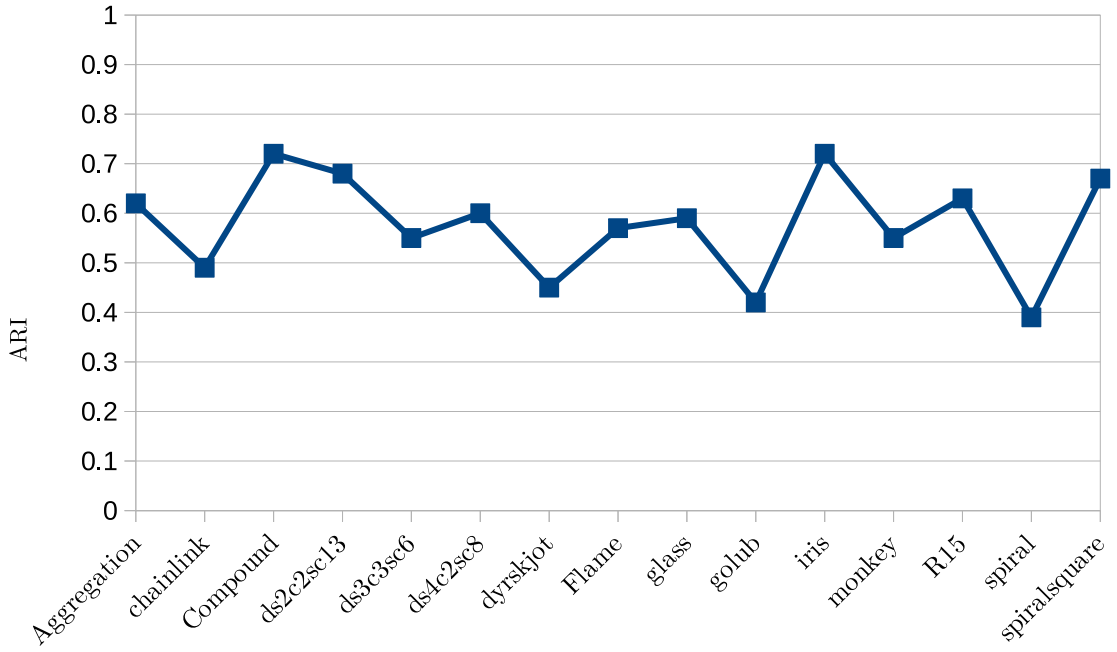
o HSS retornou somente $9,6$, o que implica em uma redução de $38,74\%$. A redução no número de partições retornadas, quando comparada com Π_C foi de $85,64\%$, enquanto a queda na qualidade das partições retornadas foi de somente 5%. A base de dados que apresentou a maior redução no número de soluções foi `monkey`: de 76 partições em Π_C, o *HSS* retornou somente 7, uma redução de $90,79\%$. Já a base de dados com a menor redução no número de soluções foi a `chainlink`: de 27 partições em Π_C, o *HSS* retornou 5, uma redução de $81,48\%$.

A grande vantagem do *HSS* é que ele retorna um número menor de soluções para a grande maioria dos casos. Ele também mantém, de forma geral, a qualidade das partições selecionadas.

A Figura 18 tem por objetivo sumarizar os resultados obtidos pelo *HSS* quanto à

dissimilaridade entre as partições retornadas. Para isso, é calculado o *ARI* médio entre todas as partições selecionadas para cada base de dados e exibido o maior valor de *ARI* médio obtido. Como é possível observar, em todos os casos o valor do *ARI* foi inferior a $0,75$, o que implica que o *HSS* conseguiu selecionar partições com uma boa diversidade.

Figura 18 – Gráfico de linhas mostrando o *ARI* entre as soluções retornadas pelo *HSS* (linha azul). Quanto menor o *ARI*, maior a dissimilaridade entre as partições sendo comparadas.

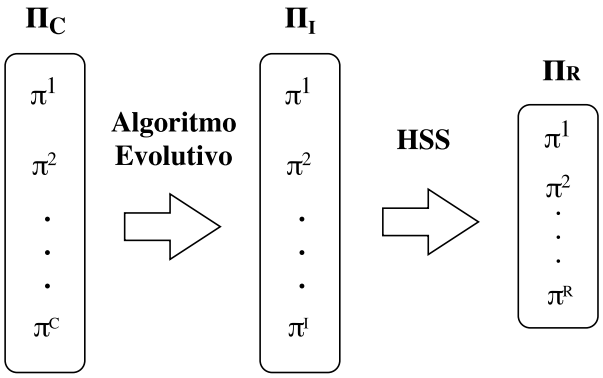


4.4.2 *MOCK* e *MOCLE*

Esta subseção apresenta os resultados obtidos pelo *HSS* usado como ferramenta de pós-processamento para os algoritmos *MOCK* e *MOCLE*. A Figura 19 resume este experimento: o conjunto de partições base, Π_C, é empregado como entrada para o algoritmo evolutivo - *MOCK* ou *MOCLE* -, as partições obtidas, Π_I, são, então, empregadas como entrada para o *HSS* que, por sua vez, retorna um conjunto reduzido de partições, Π_R. É importante destacar que, por serem algoritmos evolutivos, tanto o *MOCK* quanto o *MOCLE* podem gerar novos indivíduos (partições) a partir das soluções candidatas consideradas mais promissoras, ou seja, aquelas que possuem aptidão (função que mede a qualidade das partições) acima da média.

A Tabela 5 exibe uma comparação dos resultados obtidos pelo conjunto de soluções completo do *MOCK* e pelo conjunto de soluções recomendadas pelo *MOCK* (MOCK_R) com os resultados obtidos pelo *MOCK* com a aplicação do *HSS* (MOCK_HSS). O número de soluções é a média das 30 execuções. Para cada uma dessas execuções, é calculado o *ARI*

Figura 19 – Representação do *HSS* aplicado às partições geradas pelos algoritmos evolutivos *MOCK* e *MOCLE*.



entre cada partição e estrutura conhecida e salvo o maior *ARI*. A média e o desvio-padrão desses *ARIs* são exibidos na tabela.

Tabela 5 – Número de soluções e qualidade da melhor solução (ARI) retornados pelo $MOCK$, pelo conjunto recomendado do $MOCK$ ($MOCK_R$) e pelo HSS recebendo como entrada as partições retornadas pelo $MOCK$ ($MOCK_{HSS}$). Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI.

Base de Dados	**Número de Soluções**			**ARI**			
	MOCK	**MOCK$_R$**	**MOCK$_{HSS}$**	**STR**	**MOCK**	**MOCK$_R$**	**MOCK$_{HSS}$**
Aggregation	110,50	1,03	30,93	E_1	0,99 ± 0,00	$\triangledown$ 0,82 ± 0,04	0,99 ± 0,00
chainlink	71,60	1,20	20,30	E_1	1,00 ± 0,00	0,97 ± 0,08	1,00 ± 0,00
Compound	95,37	1,03	26,07	E_1	0,86 ± 0,00	0,81 ± 0,00	0,86 ± 0,01
ds2c2sc13	97,87	2,77	26,90	E_1	0,38 ± 0,00	0,38 ± 0,00	0,37 ± 0,01
				E_2	1,00 ± 0,00	1,00 ± 0,00	0,98 ± 0,02
				E_3	0,87 ± 0,00	$\triangledown$ 0,72 ± 0,06	0,87 ± 0,01
ds3c3sc6	104,20	3,33	28,67	E_1	0,78 ± 0,05	$\triangledown$ 0,61 ± 0,08	0,74 ± 0,04
				E_2	0,71 ± 0,04	$\triangledown$ 0,45 ± 0,07	0,69 ± 0,05
ds4c2sc8	82,70	2,77	23,03	E_1	0,90 ± 0,01	0,85 ± 0,06	0,88 ± 0,03
				E_2	0,25 ± 0,02	$\triangledown$ 0,19 ± 0,03	0,24 ± 0,02
dyrskjot	42,57	0,63	4,03	E_1	0,11 ± 0,00	$\triangledown$ -0,01 ± 0,02	$\triangledown$ 0,05 ± 0,03
flame	102,57	1,23	26,97	E_1	0,96 ± 0,04	$\triangledown$ 0,66 ± 0,21	0,92 ± 0,05
glass	97,03	2,50	24,03	E_1	0,55 ± 0,01	$\triangledown$ 0,49 ± 0,06	0,54 ± 0,01
				E_2	0,44 ± 0,01	$\triangledown$ 0,37 ± 0,04	0,42 ± 0,01
				E_3	0,20 ± 0,01	0,16 ± 0,02	0,20 ± 0,01
golub	77,47	0,73	11,03	E_1	0,87 ± 0,08	$\triangledown$ 0,15 ± 0,16	$\triangledown$ 0,67 ± 0,13
				E_2	0,82 ± 0,13	$\triangledown$ -0,02 ± 0,08	$\triangledown$ 0,49 ± 0,12
iris	67,57	1,13	16,97	E_1	0,79 ± 0,08	$\triangledown$ 0,58 ± 0,03	0,74 ± 0,06
monkey	93,27	1,93	26,37	E_1	0,85 ± 0,03	$\triangledown$ 0,79 ± 0,04	0,84 ± 0,04
				E_2	0,78 ± 0,00	$\triangledown$ 0,57 ± 0,03	0,78 ± 0,01
				E_3	0,76 ± 0,02	0,72 ± 0,03	0,75 ± 0,02
R15	89,90	3,13	25,87	E_1	0,99 ± 0,00	0,99 ± 0,00	0,98 ± 0,01
spiral	104,07	0,27	27,53	E_1	0,50 ± 0,07	$\triangledown$ 0,02 ± 0,02	0,45 ± 0,07
spiralsquare	66,30	1,83	19,80	E_1	0,99 ± 0,00	$\triangledown$ 0,82 ± 0,2	0,98 ± 0,03
				E_2	0,75 ± 0,00	0,75 ± 0,00	0,74 ± 0,02
Média	**86,87**	**1,70**	**22,57**		**0,72**	**0,55**	**0,69**

Como pode ser observado na Tabela 5, o número de soluções recomendadas pelo *MOCK* ($MOCK_R$) é bem pequeno e o *MOCK* somente recomenda a melhor partição em 4 casos. Este estudo sugere que, apesar da grande redução no número de soluções, a qualidade das partições selecionadas não é garantida. É importante destacar que em diversos casos o *MOCK* não recomendou nenhuma partição. Por isso, em alguns casos o número médio de soluções foi menor do que 1.

É possível observar, também, na Tabela 5 que o número de soluções retornadas com a aplicação do *HSS* às soluções obtidas pelo *MOCK* é muito maior do que o número de soluções presentes no conjunto recomendado do *MOCK*. Além disso, a qualidade das soluções retornadas com o *HSS* foi maior do que a do conjunto recomendado pelo *MOCK*: $0,69$ contra $0,55$.

O *HSS* conseguiu reduzir o número de soluções do *MOCK* sem causar muito impacto na qualidade das partições: reduziu em $74,02\%$ o número de soluções retornadas pelo *MOCK*, enquanto a queda na qualidade foi de aproximadamente $4,17\%$. A base de dados com a maior redução no número de partições foi `dyrskjot`: de $42,57$ partições do *MOCK*, o *HSS* retornou somente $4,03$, uma redução de 90.53%. É importante destacar que é possível reduzir o número de soluções retornadas pelo *HSS* através da modificação do parâmetro *porcentagem*.

Embora o *HSS* tenha selecionado a melhor partição do *MOCK* em somente 6 casos, para a maioria dos demais casos, a diferença no valor do *ARI* do *MOCK* com e sem o *HSS* foi relativamente pequena: somente em 3 casos foi maior do que $0,05$.

A Tabela 6 exibe uma comparação dos resultados obtidos pelo *MOCLE* com aqueles obtidos pelo *MOCLE* com a aplicação do *HSS* ($\text{MOCLE}_{\text{HSS}}$). O número de soluções e o *ARI* foram obtidos da mesma forma descrita para o *MOCK*.

A aplicação do *HSS* nas partições retornadas pelo *MOCLE* resultou em uma redução de $74,85\%$ no número de partições retornadas comparado ao *MOCLE* - da média de $28,67$ partições do *MOCLE*, *HSS* retornou somente $7,21$ na média -, enquanto a queda na qualidade das partições (representada pelo *ARI*) foi de somente $4,05\%$: o *ARI* médio do *MOCLE* foi $0,74$, com a aplicação do *HSS* nas partições do *MOCLE*, o novo *ARI* foi de $0,71$. O *HSS* conseguiu selecionar a melhor estrutura do *MOCLE* em 6 casos. Em somente 3 casos a diferença no valor do *ARI* entre *MOCLE* e *MOCLE* com *HSS* foi maior do que $0,05$.

A maior redução no número de soluções retornadas pelo *HSS* foi para a baase de dados `ds3c3sc6`: de $54,67$ partições retornadas pelo *MOCLE*, *HSS* retornou somente $10,3$, uma redução de $81,16\%$. Por outro lado, a menor redução ocorreu para a base de dados `dyrskjot`, uma redução de $58,9\%$, dado que o *MOCLE* retornou um número razoavelmente pequeno de partições, $7,3$, o *HSS* selecionou 3 dessas partições.

Tabela 6 – Número de soluções e qualidade da melhor solução (ARI) retornados pelo *MOCLE* e pelo *HSS* recebendo como entrada as partições retornadas pelo *MOCLE* ($MOCLE_{HSS}$). Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI.

Base de Dados	**Número de Soluções**		**ARI**		
	MOCLE	**MOCLE$_{HSS}$**	**STR**	**MOCLE**	**MOCLE$_{HSS}$**
Aggregation	31,37	8,03	E_1	0,99 ± 0,00	0,99 ± 0,01
chainlink	9,00	3,00	E_1	1,00 ± 0,00	1,00 ± 0,00
Compound	19,73	5,87	E_1	0,85 ± 0,01	0,83 ± 0,03
ds2c2sc13	58,50	11,40	E_1	1,00 ± 0,00	1,00 ± 0,00
			E_2	1,00 ± 0,00	0,95 ± 0,00
			E_3	0,78 ± 0,00	0,77 ± 0,00
ds3c3sc6	54,67	10,30	E_1	0,96 ± 0,01	0,94 ± 0,03
			E_2	0,67 ± 0,01	0,63 ± 0,04
ds4c2sc8	62,60	13,90	E_1	0,89 ± 0,01	0,87 ± 0,02
			E_2	0,32 ± 0,02	0,31 ± 0,01
dyrskjot	7,30	3,00	E_1	0,54 ± 0,00	0,53 ± 0,02
flame	12,70	4,03	E_1	0,96 ± 0,02	$\triangledown$ 0,87 ± 0,17
glass	28,47	8,63	E_1	0,67 ± 0,00	0,66 ± 0,01
			E_2	0,56 ± 0,00	0,56 ± 0,01
			E_3	0,26 ± 0,00	0,26 ± 0,00
golub	15,00	5,00	E_1	0,87 ± 0,00	$\triangledown$ 0,80 ± 0,00
			E_2	0,88 ± 0,00	$\triangledown$ 0,56 ± 0,00
iris	10,00	3,10	E_1	0,76 ± 0,01	0,71 ± 0,04
monkey	34,77	8,67	E_1	0,67 ± 0,00	0,64 ± 0,02
			E_2	0,71 ± 0,01	0,70 ± 0,01
			E_3	0,59 ± 0,01	0,58 ± 0,01
R15	45,67	11,60	E_1	0,99 ± 0,00	0,98 ± 0,00
spiral	16,50	4,87	E_1	0,02 ± 0,00	0,01 ± 0,01
spiralsquare	23,80	6,70	E_1	0,67 ± 0,00	0,66 ± 0,00
			E_2	1,00 ± 0,00	1,00 ± 0,00
Média	**28,67**	**7,21**		**0,74**	**0,71**

É interessante ressaltar que a redução no número de soluções retornadas pelo *HSS* como ferramenta de pós-processamento para técnicas evolutivas é menor do que a redução ocorrida quando o *HSS* é aplicado diretamente no conjunto de partições base (Π_C). Isso ocorre devido ao fato das estratégias evolutivas serem capazes, por si só, de descartar partições de baixa qualidade. Ainda assim, o *HSS* foi eficaz na redução do número de partições retornadas, sem causar um impacto muito grande na qualidade das mesmas.

4.4.3 Seleções de *Ensemble*

Foi realizada, também, uma comparação da qualidade das partições retornadas pelo *HSS* com as estratégias de seleção de partições de *ensemble SR*, *BRP*, *SRD* e *Diversity*, descritas no Apêndice A. Como essas estratégias requerem o número de partições a serem

selecionadas, elas foram testadas retornando 10, 25, 50 e 75% de Π_C (Naldi, Carvalho e Campello (2013) empregam estes valores de porcentagem nos testes que conduziram).

A Tabela 7 apresenta a comparação do *HSS* com as estratégias *SR*, *BRP*, *SRD* e *Diversity* retornando 50% de Π_C [1]. Foi escolhido 50% por ser o valor que melhor consegue conciliar número de partições e qualidade.

Dentre as estratégias de *ensemble*, *Diversity* foi a que apresentou o melhor resultado: obteve um *ARI* de $0,78$, enquanto o do *HSS* foi de $0,76$. Porém o número médio de soluções retornadas foi bem maior: $33,8$ contra $9,6$ do *HSS*. As demais técnicas apresentaram *ARI* médio inferior ao do *HSS*: *SR* $0,72$, *BRP* $0,7$, *SRD* $0,69$.

O *SR* conseguiu identificar a melhor estrutura em 13 casos, o *BRP* em 16 casos, o *SRD* em 12 casos, o *Diversity* em 18 casos e o *HSS* em 13. É interessante notar que apesar do *SR* identificar o mesmo número de melhores estruturas que o *HSS* e o *BRP* um número maior, o *HSS* apresenta, na média, desempenho superior a ambas as técnicas.

[1] Os resultados para os demais valores de porcentagem citados podem ser aferidos no Apêndice E.

Tabela 7 – Número de soluções e qualidade da melhor solução (*ARI*) retornados por Π_C, *SR*, *BRP*, *SRD*, *Diversity* (*Div*) e *HSS*. Como as estratégias de *ensemble SR*, *BRP*, *SRD*, *Div* retornam o mesmo número de partições, esse número está na coluna *CES*. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do *ARI*.

Base de Dados	**Número de Soluções**			**ARI**						
	Π_C	**CES**	**HSS**	**STR**	Π_C	**SR**	**BRP**	**SRD**	**Div**	**HSS**
Aggregation	74,00	37,00	10,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
chainlink	27,00	14,00	5,00	E_1	1,00	1,00	$\triangledown$ 0,50	$\triangledown$ 0,40	1,00	1,00
Compound	62,00	31,00	8,00	E_1	0,85	0,85	0,85	0,85	0,85	0,85
ds2c2sc13	147,00	74,00	18,00	E_1	1,00	1,00	1,00	1,00	1,00	1,00
				E_2	1,00	1,00	1,00	1,00	$\triangledown$ 0,87	1,00
				E_3	1,00	$\triangledown$ 0,62	$\triangledown$ 0,63	$\triangledown$ 0,62	1,00	$\triangledown$ 0,87
ds3c3sc6	65,00	33,00	7,00	E_1	0,90	0,90	0,90	0,90	0,90	$\triangledown$ 0,81
				E_2	0,59	0,59	0,59	0,59	0,58	$\triangledown$ 0,50
ds4c2sc8	89,00	45,00	9,00	E_1	0,87	0,87	0,87	0,87	0,82	0,87
				E_2	0,35	0,30	0,30	0,30	0,35	$\triangledown$ 0,29
dyrskjot	25,00	13,00	7,00	E_1	0,55	0,50	0,55	$\triangledown$ 0,47	0,55	0,54
Flame	25,00	13,00	5,00	E_1	0,94	0,94	$\triangledown$ 0,71	0,94	0,92	0,94
glass	55,00	28,00	9,00	E_1	0,67	0,65	0,67	0,65	0,67	0,65
				E_2	0,56	$\triangledown$ 0,50	0,55	$\triangledown$ 0,50	0,55	0,56
				E_3	0,26	0,24	0,26	0,24	0,26	0,26
golub	49,00	25,00	9,00	E_1	0,87	$\triangledown$ 0,68	$\triangledown$ 0,68	$\triangledown$ 0,68	$\triangledown$ 0,68	0,87
				E_2	0,94	0,94	0,94	0,94	0,94	$\triangledown$ 0,61
iris	37,00	19,00	6,00	E_1	0,82	$\triangledown$ 0,76	$\triangledown$ 0,76	$\triangledown$ 0,76	0,82	$\triangledown$ 0,76
monkey	76,00	38,00	7,00	E_1	0,67	0,65	0,65	0,65	0,67	0,65
				E_2	0,83	$\triangledown$ 0,71	0,83	$\triangledown$ 0,71	0,83	$\triangledown$ 0,70
				E_3	0,65	$\triangledown$ 0,58	0,65	$\triangledown$ 0,58	0,65	$\triangledown$ 0,58
R15	154,00	77,00	23,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
spiral	33,00	17,00	6,00	E_1	1,00	$\triangledown$ 0,02	$\triangledown$ 0,02	$\triangledown$ 0,02	1,00	$\triangledown$ 0,92
spiralsquare	85,00	43,00	15,00	E_1	0,67	0,67	0,67	0,67	0,63	0,67
				E_2	1,00	1,00	1,00	1,00	1,00	1,00
Média	**66,87**	**33,80**	**9,60**		**0,80**	**0,72**	**0,70**	**0,69**	**0,78**	**0,76**

5 Conclusão

Esta dissertação apresentou uma nova técnica de seleção para o problema de agrupamento: o *HSS*. Este algoritmo foi concebido a partir da necessidade de retornar, ao usuário final, um conjunto de partições de boa qualidade e diversas entre si. Os algoritmos evolutivos - principal técnica para abordar esse problema - costumam retornar um número elevado de partições e possuem um custo computacional alto. O intuito de desenvolver uma técnica de seleção - que diferentemente dos algoritmos evolutivos não cria, mas somente seleciona partições - foi prover um conjunto reduzido de partições a um custo computacional razoável.

O *HSS* usa diferentes princípios de seleção: primeiro uma pré-seleção das partições é realizada através da aplicação de um algoritmo para obtenção da aproximação de uma FP com dois objetivos - conectividade e variância - e, então, após a divisão da aproximação da FP em regiões, o *HSS* calcula o *ARI* médio para cada partição da região em questão e seleciona a partição com o maior valor de *ARI* médio entre aquelas da região. A principal limitação dessa abordagem é que sua eficiência é altamente dependente da qualidade e diversidade do conjunto completo de partições, já que o *HSS* não gera partições novas, mas somente seleciona partições.

Resultados experimentais mostraram que o *HSS* conseguiu reduzir consideravelmente o número de partições retornadas ao especialista no domínio, enquanto manteve a qualidade geral das mesmas. A análise do comportamento do *HSS* como ferramenta de pós-processamento para as técnicas *MOCK* e *MOCLE* mostrou sua capacidade de reduzir consideravelmente o número de partições retornadas com baixa perda de estruturas relevantes. A aplicação do *HSS* na seleção de partições para a execução de algoritmos de *ensemble* se mostrou promissora nos experimentos conduzidos. A maior limitação do *HSS* é que sua eficiência é altamente dependente da qualidade e diversidade das partições que ele recebe como entrada, pelo fato de ele não gerar partições.

Trabalhos futuros incluem:

- Aplicação do *HSS* na seleção de *clusters*, ao invés de partições. Um conjunto diverso de partições pode apresentar uma grande quantidade de *clusters* idênticos (ALMEIDA; SAKATA; FACELI, 2016). Essa redundância de informações é indesejável, uma vez que ocupa mais espaço em disco do que somente os *clusters* distintos e dificulta a análise do especialista no domínio. A ideia consiste em adaptar o *HSS* para receber como entrada um conjunto de *clusters* e selecionar um subconjunto do mesmo. Trabalhos nesse contexto incluem Almeida, Sakata e Faceli (2016), Jiamthapthaksin, Eick e Vilalta (2009).

- Aplicação do *HSS* na seleção de partições de *ensemble*. Como técnicas de *ensemble* combinam um conjunto de partições produzindo uma partição consenso, qualidade e diversidade são dois fatores determinantes (NALDI; CARVALHO; CAMPELLO, 2013). A seleção pode ser aplicada no conjunto inicial de partições para selecionar aquelas mais promissoras. Conforme descrito na Seção 4.4.3, o *HSS* mostrou bons resultados no experimento comparativo com algoritmos de seleção de *ensemble*. O próximo passo consiste em fazer uma comparação dos resultados obtidos por uma estratégia de *ensemble* com e sem a aplicação do *HSS*.

- Desenvolver ferramenta de visualização da FP para que o usuário possa determinar o número partições a serem selecionadas pelo algoritmo, eliminando assim a necessidade do parâmetro *porcentagem*.

- Estudar nova abordagem para a divisão da FP em regiões, através do uso da mediana.

Referências

AGGARWAL, C. C.; REDDY, C. K. *Data clustering: algorithms and applications*. [S.l.]: CRC press, 2013. Citado na página 8.

ALMEIDA, J. L. B. de; SAKATA, T. C.; FACELI, K. Asaclu: Selecting diverse and relevant clusters. In: IEEE. *5th Brazilian Conference on Intelligent Systems (BRACIS)*. Recife, Pernambuco, 2016. p. 474–479. Citado na página 43.

ANTUNES, V.; FACELI, K.; SAKATA, T. C. HSS: Compact set of partitions via hybrid selection. In: IEEE. *6th Brazilian Conference on Intelligent Systems (BRACIS)*. Uberlândia, MG, Brazil, 2017. p. 37–42. Citado na página 17.

CHANG, H.; YEUNG, D.-Y. Robust path-based spectral clustering. *Pattern Recognition*, Elsevier, v. 41, n. 1, p. 191–203, 2008. Citado na página 28.

CORNE, D. W. et al. PESA-II: Region-based selection in evolutionary multiobjective optimization. In: MORGAN KAUFMANN PUBLISHERS INC. *3rd Annual Conference on Genetic and Evolutionary Computation (GECCO)*. San Francisco, CA, USA, 2001. p. 283–290. Citado na página 13.

DEB, K. Multi-objective optimisation using evolutionary algorithms: an introduction. In: *Multi-objective evolutionary optimisation for product design and manufacturing*. [S.l.]: Springer, 2011. p. 3–34. Citado na página 11.

DEB, K. et al. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. *Lecture Notes in Computer Science (LNCS)*, Springer, p. 849–858, 2000. Citado na página 12.

DING, L.; ZENG, S.; KANG, L. A fast algorithm on finding the non-dominated set in multi-objective optimization. In: IEEE. *The 2003 Congress on Evolutionary Computation (CEC)*. Canberra, ACT, Australia, 2003. v. 4, p. 2565–2571. Citado na página 12.

DYRSKJØT, L. et al. Identifying distinct classes of bladder carcinoma using microarrays. *Nature genetics*, Nature Publishing Group, v. 33, n. 1, p. 90, 2003. Citado na página 28.

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 (SDM)*. Arlington, VA, USA: [s.n.], 2002. p. 105–115. Citado na página 7.

ESTER, M. et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: *2nd International Conference on Knowledge Discovery and Data Mining (KDD)*. Portland, Oregon, USA: [s.n.], 1996. v. 96, n. 34, p. 226–231. Citado na página 8.

EVERITT, B. et al. *Cluster Analysis*. [S.l.]: Wiley, 2011. Citado na página 1.

FACELI, K. *Um framework para análise de agrupamento baseado na combinação multi-objetivo de algoritmos de agrupamento*. Tese (Doutorado) — Instituto de Ciências

Matemáticas e de Computação - ICMC-USP, 2006. Citado 5 vezes nas páginas 3, 4, 11, 14 e 15.

FACELI, K.; CARVALHO, A. C. D.; SOUTO, M. C. D. Multi-objective clustering ensemble. *International Journal of Hybrid Intelligent Systems (HIS)*, IOS Press, v. 4, n. 3, p. 145–156, 2007. Citado 2 vezes nas páginas 9 e 14.

FACELI, K.; SAKATA, T. C. *Clusters Evaluation Benchmark*. 2016. Disponível em: <http://lasid.sor.ufscar.br/clustersEvaluationBenchmark/>. Citado na página 28.

FACELI, K. et al. Partitions selection strategy for set of clustering solutions. *Neurocomputing*, Elsevier, v. 73, n. 16, p. 2809–2819, 2010. Citado 2 vezes nas páginas 2 e 8.

FACELI, K. et al. Multi-objective clustering ensemble for gene expression data analysis. *Neurocomputing*, Elsevier, v. 72, n. 13, p. 2763–2774, 2009. Citado na página 14.

FU, L.; MEDICO, E. FLAME, a novel fuzzy clustering method for the analysis of dna microarray data. *BMC bioinformatics*, BioMed Central, v. 8, n. 1, p. 3, 2007. Citado na página 28.

GHAEMI, R. et al. A survey: clustering ensembles techniques. *World Academy of Science, Engineering and Technology (WASET)*, v. 50, p. 636–645, 2009. Citado 2 vezes nas páginas 2 e 8.

GIAGKIOZIS, I.; PURSHOUSE, R. C.; FLEMING, P. J. Generalized decomposition. In: SPRINGER. *7th International Conference on Evolutionary Multi-Criterion Optimization (EMO)*. Sheffield, UK, 2013. p. 428–442. Citado na página 12.

GIONIS, A.; MANNILA, H.; TSAPARAS, P. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, ACM, v. 1, n. 1, p. 4, 2007. Citado na página 28.

GOEL, T.; STANDER, N. A study on the convergence of multiobjective evolutionary algorithms. In: *13th AIAA/ISSMO Multidisciplinary Analysis Optimization (MAO) Conference*. Texas, USA: [s.n.], 2010. p. 9233. Citado 2 vezes nas páginas 2 e 14.

GOLUB, T. R. et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, American Association for the Advancement of Science, v. 286, n. 5439, p. 531–537, 1999. Citado na página 28.

HALKIDI, M.; BATISTAKIS, Y.; VAZIRGIANNIS, M. On clustering validation techniques. *Journal of Intelligent Information Systems (JIIS)*, Springer, v. 17, n. 2, p. 107–145, 2001. Citado na página 8.

HANDL, J.; KNOWLES, J. An evolutionary approach to multiobjective clustering. *IEEE Transactions on Evolutionary Computation (TEVC)*, IEEE, v. 11, n. 1, p. 56–76, 2007. Citado 2 vezes nas páginas 3 e 13.

HUBERT, L.; ARABIE, P. Comparing partitions. *Journal of classification*, Springer, v. 2, n. 1, p. 193–218, 1985. Citado 4 vezes nas páginas 15, 23, 30 e 51.

ISHIBUCHI, H.; TSUKAMOTO, N.; NOJIMA, Y. Evolutionary many-objective optimization: A short review. In: *The 2008 IEEE Congress on Evolutionary Computation (CEC)*. Hong Kong, China: [s.n.], 2008. p. 2419–2426. Citado na página 12.

JAIN, A. K. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters (PRL)*, Elsevier, v. 31, n. 8, p. 651–666, 2010. Citado na página 1.

JAIN, A. K.; DUBES, R. C. *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988. ISBN 0-13-022278-X. Citado na página 8.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: a review. *ACM Computing Surveys (CSUR)*, Acm, v. 31, n. 3, p. 264–323, 1999. 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: SPRINGER. *International Conference on Advanced Data Mining and Applications (ADMA)*. Beijing, China, 2009. p. 188–199. Citado na página 43.

KONAK, A.; COIT, D. W.; SMITH, A. E. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety*, Elsevier, v. 91, n. 9, p. 992–1007, 2006. Citado na página 11.

KUNG, H.-T.; LUCCIO, F.; PREPARATA, F. P. On finding the maxima of a set of vectors. *Journal of the ACM (JACM)*, ACM, v. 22, n. 4, p. 469–476, 1975. Citado na página 12.

LAW, M. H.; TOPCHY, A. P.; JAIN, A. K. Multiobjective data clustering. In: IEEE. *The 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Washington, D.C., USA, 2004. v. 2, p. 32–41. Citado 4 vezes nas páginas 2, 3, 7 e 8.

LEI, Y. *Cluster validation and discovery of multiple clusterings*. Tese (Doutorado) — University of Melbourne, Australia, 2016. Citado na página 2.

LICHMAN, M. *UCI Machine Learning Repository*. 2013. Disponível em: <http://archive.ics.uci.edu/ml>. Citado na página 28.

MISHRA, K.; HARIT, S. A fast algorithm for finding the non dominated set in multi-objective optimization. *International Journal of Computer Applications (IJCA)*, International Journal of Computer Applications, v. 1, n. 25, p. 35–39, 2010. Citado 2 vezes nas páginas 12 e 19.

MUKHOPADHYAY, A.; MAULIK, U.; BANDYOPADHYAY, S. A survey of multiobjective evolutionary clustering. *ACM Computing Surveys (CSUR)*, ACM, v. 47, n. 4, p. 61–107, 2015. Citado na página 2.

NALDI, M. C.; CARVALHO, A. C.; CAMPELLO, R. J. Cluster ensemble selection based on relative validity indexes. *Data Mining and Knowledge Discovery (DMKD)*, Springer, v. 27, n. 2, p. 259–289, set. 2013. Citado 4 vezes nas páginas 9, 41, 44 e 49.

OSYCZKA, A. Multicriteria optimization for engineering design. *Design optimization*, v. 1, p. 193–227, 1985. Citado na página 10.

RODRIGUEZ, A.; LAIO, A. Clustering by fast search and find of density peaks. *Science*, American Association for the Advancement of Science, v. 344, n. 6191, p. 1492–1496, 2014. Citado na página 7.

SAKATA, T. C. et al. Improvements in the partitions selection strategy for set of clustering solutions. In: IEEE. *11th Brazilian Symposium on Neural Networks (SBRN)*. São Paulo, Brazil, 2010. p. 49–54. Citado 4 vezes nas páginas 2, 8, 15 e 55.

STREHL, A.; GHOSH, J. Cluster ensembles-a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research (JMLR)*, v. 3, n. Dec, p. 583–617, 2002. Citado na página 2.

ULTSCH, A. Clustering wih som: U* c. In: *5th Workshop on Self-Organizing Maps (WSOM)*. Paris, France: [s.n.], 2005. v. 2, p. 75–82. Citado na página 28.

VEENMAN, C. J.; REINDERS, M. J. T.; BACKER, E. A maximum variance cluster algorithm. *IEEE Transactions on pattern analysis and machine intelligence (TPAMI)*, IEEE, v. 24, n. 9, p. 1273–1280, 2002. Citado na página 28.

ZAHN, C. T. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on computers*, IEEE, v. 100, n. 1, p. 68–86, 1971. Citado na página 28.

ZHANG, T.; RAMAKRISHNAN, R.; LIVNY, M. Birch: an efficient data clustering method for very large databases. In: ACM. *ACM Special Interest Group on Management of Data (SIGMOD) Record*. Montreal, Quebec, Canada, 1996. v. 25, n. 2, p. 103–114. Citado na página 8.

ZITZLER, E.; THIELE, L. Multiobjective optimization using evolutionary algorithms—a comparative case study. In: SPRINGER. *International Conference on Parallel Problem Solving from Nature (PPSN)*. Berlin, Heidelberg, 1998. p. 292–301. Citado na página 10.

APÊNDICE A – *Cluster Ensemble Selection* (*CES*)

O objetivo da seleção de partições no contexto de *ensemble* é prover um subconjunto do conjunto inicial de partições como entrada para algoritmos de *ensemble* na tentativa de melhorar a partição final retornada, uma vez que partições ruins no conjunto de entrada podem impactar negativamente o *ensemble* obtido.

Naldi, Carvalho e Campello (2013) investigam o uso de índices de validação no contexto de *CES* propondo diversos algoritmos para a seleção de partições de *ensemble*. Como este trabalho cita, cada índice de validação é apropriado para um tipo de dado específico e, como o conjunto inicial tende a ter partições com características diversas, é interessante, para manter a diversidade das partições selecionadas, empregar, ao invés de somente um índice de validação, um conjunto de índices na expectativa de aumentar a diversidade das partições selecionadas. Essa estratégia, é conhecida como *Combination of Relative Indexes* (*CRI*).

Naldi, Carvalho e Campello (2013) propõem três métodos *CRI* para *CES*, baseados no contexto de ranqueamento, no qual a melhor partição, de acordo com o índice de validação adotado, terá $rank = 1$, a segunda melhor partição $rank = 2$ e assim por diante. Todos estes métodos requerem o número de partições a serem selecionadas.

1. *Sum of Ranks* (*SR*): Para cada índice de validação, cria um ranking entre as partições base. A soma dos índices individuais para cada partição é, então, calculada e as partições com as menores somas são selecionadas para o *ensemble*.

2. *Best Rank Position* (*BRP*): Este método também cria um ranking entre as partições base para cada índice de validação. Porém, os índices individuais não são somados: as partições com melhores rankings, de acordo com cada índice, são selecionadas.

3. *Sum of Ranks with Diversity* (*SRD*): Idêntico ao primeiro método, exceto que a soma dos rankings para cada partição é balanceada pela diversidade entre a partição e o conjunto de partições base. O índice *Jaccard* é usado no cálculo da diversidade.

Com o objetivo de avaliar o impacto da diversidade no contexto de *CES*, Naldi, Carvalho e Campello (2013) também propõem, baseados em métodos anteriores, um método chamado *Diversity*. Este método começa selecionando a partição base com maior diversidade (calculada através do índice *Jaccard*) em relação ao conjunto de partições base. Então, em cada iteração, a partição base com maior diversidade em relação as partições já

selecionadas nas iterações anteriores é selecionada, até que o número de partições desejado seja obtido.

APÊNDICE B – *Adjusted Rand Index* (*ARI*)

O *Adjusted Rand Index* (HUBERT; ARABIE, 1985) é uma generalização do *Rand Index*. Ele introduz uma normalização induzida estatisticamente, a fim de produzir valores próximos de 0 para partições aleatórias. Seu valor varia entre -1 e $+1$.

Dado um conjunto de objetos $X = \{x_1, ..., x_n\}$, suponha que $\pi^1 = \{c_1^1, ..., c_A^1\}$ e $\pi^2 = \{c_1^2, c_B^2\}$ representam duas partições de X a serem avaliadas pelo ARI, tal que:

1. $\bigcup_{i=1}^{A} c_i^1 = X = \bigcup_{j=1}^{B} c_j^2$
2. $c_i^1 \cap c_{i'}^1 = \emptyset = c_j^2 \cap c_{j'}^2$ para $1 \leqslant i \neq i' \leqslant A$ e $1 \leqslant j \neq j' \leqslant B$

A comparação entre essas duas partições pode ser escrita na forma de uma tabela de contingência, na qual n_{ij} representa o número de objetos comuns para as classes c_i^1 e c_j^2, $n_{i.}$ o número de objetos na classe c_i^1 (linha i) e $n_{.j}$ o número de objetos na classe c_j^2 (coluna j), conforme a Tabela 8.

Tabela 8 – Tabela de contingência para a comparação de duas partições: π^1 e π^2.

		Partição π^2				
	Classe	c_1^2	c_2^2	$\cdots$	c_B^2	Somatório
Partição π^1	c_1^1	n_{11}	n_{12}	$\cdots$	n_{1B}	$n_{1.}$
	c_2^1	n_{21}	n_{22}	$\cdots$	n_{2B}	$n_{2.}$
	.	.	.	.	.	.
	.	.	.	.	.	.
	.	.	.	.	.	.
	c_A^1	n_{A1}	n_{A2}	$\cdots$	n_{AB}	$n_{A.}$
	Somatório	$n_{.1}$	$n_{.2}$	$\cdots$	$n_{.B}$	$n_{..} = n$

O ARI calcula a correspondência entre duas partições, π^1 e π^2, de acordo com a forma como os pares de objetos são classificados na tabela de contingência $A \times B$. Especificamente, há quatro tipos de relações distintas que podem ser identificadas:

1. Pares de objetos que pertencem à mesma classe em π^1 e π^2.
2. Pares de objetos que pertencem a diferentes classes em π^1 e π^2.
3. Pares de objetos que pertencem a diferentes classes em π^1 e à mesma classe em π^2.
4. Pares de objetos que pertencem à mesma classe em π^1 e a classes diferentes em π^2.

Usando uma representação baseada em tabela de contingência, o ARI pode ser calculado como:

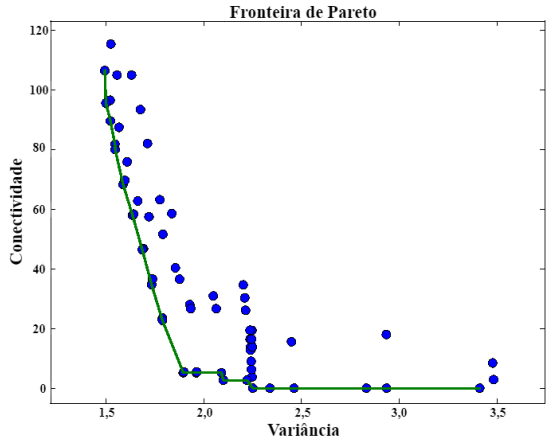
$$\frac{\overbrace{\sum_{i,j}\binom{n_{ij}}{2}}^{Index} - \overbrace{\sum_{i}\binom{n_{i.}}{2}\sum_{j}\binom{n_{.j}}{2}/\binom{n}{2}}^{ExpectedIndex}}{\underbrace{\frac{1}{2}[\sum_{i}\binom{n_{i.}}{2}+\sum_{j}\binom{n_{.j}}{2}]}_{MaxIndex} - \underbrace{\sum_{i}\binom{n_{i.}}{2}\sum_{j}\binom{n_{.j}}{2}/\binom{n}{2}}_{ExpectedIndex}} \tag{B.1}$$

APÊNDICE C – Fronteira de Pareto (*FP*)

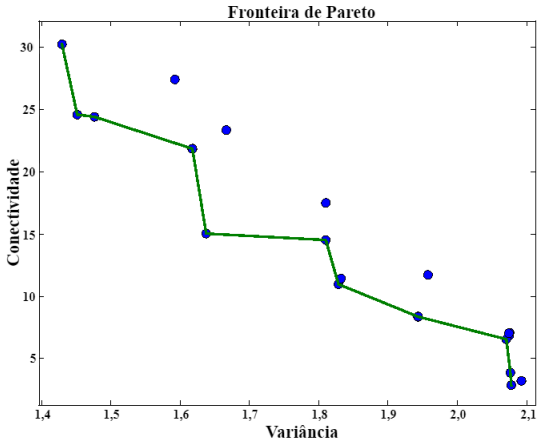
A Figura 20 apresenta a aproximação da FP gerada pelo *HSS* para diferentes bases de dados.

Figura 20 – FP obtida para as partições geradas pelos algoritmos SL, AL, CoL, CeL, KM e SNN para as base de dados: `Aggregation`, `Flame`, `ds2c2sc13` e `R15`. Cada ponto representa uma partição de Π_C. A linha verde conecta as partições não-dominadas presentes na FP.

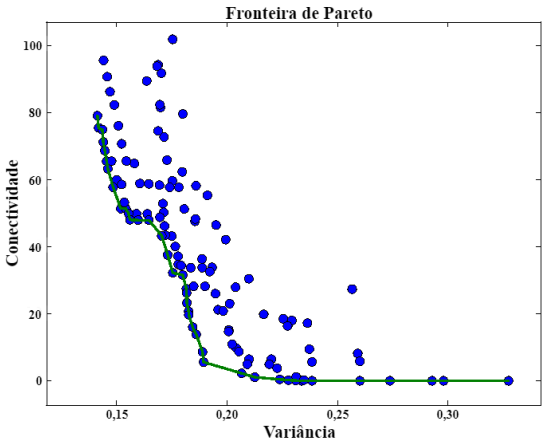
(a) `Aggregation`



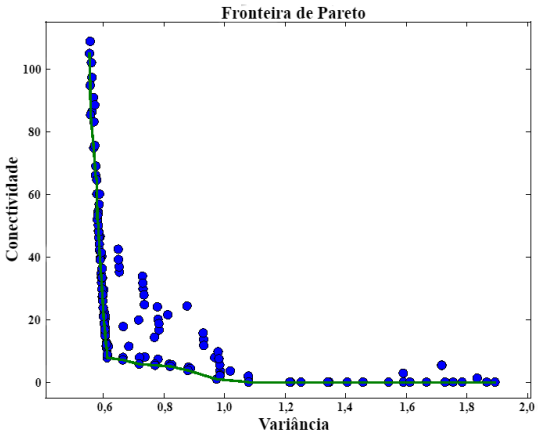
(b) `Flame`



(c) `ds2c2sc13`



(d) `R15`



APÊNDICE D – *Automatic Selection Algorithm (ASA)*

Dado um conjunto completo de partições $\Pi_C = \{\pi^1, \pi^2, ..., \pi^C\}$, o *ASA* (SAKATA et al., 2010) produz um conjunto reduzido de partições $\Pi_R = \{\pi^1, \pi^2, ..., \pi^R\}$, tal que $\Pi_R \subset \Pi_C$ e $|\Pi_R| \ll |\Pi_C|$. O ASA depende do *ARI* para descartar soluções similares. O valor do limiar de similaridade considerado para a exclusão de uma partição do conjunto de soluções é ajustado automaticamente pelo *ASA*, que possui oitos passos, conforme descrito abaixo:

1. Para cada partição $\pi^i \in \Pi_C$, conte o número de partições idênticas a π^i (n_i). Se n_i for maior que o número de algoritmos de agrupamento usados para compor Π_C, então π^i deve ser incluída em Π_R e removida de Π_C, juntamente com as demais partições idênticas a ela.

2. $nInitial \leftarrow |\Pi_C| + |\Pi_R|$, em que $|\Pi_C|$ é o número de partições em Π_C e $|\Pi_R|$ é o número de partições em Π_R.

3. Para cada partição $\pi^i \in \Pi_C$, calcule o $ARI(\pi^i, \pi^j)$ (ARI entre π^i e $\pi^j \in \Pi_C$) e $ARI_m(\pi^i) = \frac{1}{|\Pi_C|} \sum_{\pi^j \in \Pi_C} ARI(\pi^i, \pi^j)$.

4. $t \leftarrow 0.9$ (valor inicial do limiar)

5. $r_{current} \leftarrow 1.0$ (proporção atual)

6. $\Pi_{current} \leftarrow \Pi_C$

7. Repita

   a) $r_{previous} \leftarrow r_{current}$

   b) $\Pi_{previous} \leftarrow \Pi_{current}$

   c) Para cada partição $\pi^i \in \Pi_R$, remova de Π_C todas as partições π^j tal que $ARI(\pi^i, \pi^j) \geq t$.

   d) Pegue a partição π^d em que $ARI_m(\pi^d) < ARI_m(\pi^j)$ para todo $\pi^j \in \Pi_C$. Remova-o de Π_C e coloque-o em $\Pi_{current}$.

   e) Exclua de Π_C todas as partições π^j em que $ARI(\pi^d, \pi^j) \geq t$.

   f) Repita os Passos 7d e 7e até que não haja mais partições em Π_C.

   g) $t \leftarrow t - 0.1$

h) $\Pi_C \leftarrow \Pi_{current}$

i) $r_{current} \leftarrow \frac{nInitial}{nCurrent}$, em que $nCurrent$ é o número de partições em $\Pi_{current}$

até que ($r_{previous} - r_{current} \leq 0.12$ ou $t < 0.1$)

8. $\Pi_R \leftarrow \Pi_R \cup \Pi_{previous}$

APÊNDICE E – Estudo das Seleções de *Ensemble*

A Tabela 9 exibe os resultados obtidos pelo *SR* com diferentes valores de porcentagem: 10% (SR_{10}), 25% (SR_{25}), 50% (SR_{50}) e 75% (SR_{75}). Como é possível observar na tabela, o *SR* só consegue retornar um *ARI* médio maior do que o do *HSS* - $0,77$ enquanto o *HSS* retorna $0,76$ - quando retorna 75% de Π_C, neste caso, enquanto o *SR* retorna, na média, $50,53$ partições, o *HSS* retorna somente $9,6$.

A Tabela 10 exibe os resultados obtidos pelo *BRP* com diferentes valores de porcentagem: 10% (BRP_{10}), 25% (BRP_{25}), 50% (BRP_{50}) e 75% (BRP_{75}). Como é possível observar na tabela, o *BRP* só consegue retornar um *ARI* médio maior do que o do *HSS* - $0,78$ enquanto o *HSS* retorna $0,76$ - quando retorna 75% de Π_C, neste caso, enquanto o *BRP* retorna, na média, $50,53$ partições, o *HSS* retorna somente $9,6$.

A Tabela 11 exibe os resultados obtidos pelo *SRD* com diferentes valores de porcentagem: 10% (SRD_{10}), 25% (SRD_{25}), 50% (SRD_{50}) e 75% (SRD_{75}). Como é possível observar na tabela, o *SRD* só consegue retornar um *ARI* médio maior do que o do *HSS* - $0,77$ enquanto o *HSS* retorna $0,76$ - quando retorna 75% de Π_C, neste caso, enquanto o *SRD* retorna, na média, $50,53$ partições, o *HSS* retorna somente $9,6$.

Dentre as estratégias de seleção de *ensemble*, a *Diversity* (*Div*) é a que apresenta os melhores resultados, exibidos na Tabela 12. *Div* é testada com diferentes valores de porcentagem: 10% (Div_{10}), 25% (Div_{25}), 50% (Div_{50}) e 75% (Div_{75}). Como é possível observar na tabela, *Div* consegue retornar um *ARI* médio maior do que o do *HSS* quando retorna 50% e 75% de Π_C: $0,78$ e $0,8$ respectivamente, enquanto o *HSS* retorna $0,76$. Porém é importante destacar que o número de partições retornadas por *Div*, em ambos os casos, é bem maior do que o retornado pelo *HSS*: Div_{50} e Div_{75} retornam, respectivamente, na média, $33,8$ e $50,53$ partições, enquanto o *HSS* retorna $9,6$.

Tabela 9 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, SR_{10}, SR_{25}, SR_{50}, SR_{75} e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI.

Base de Dados	**Número de Soluções**						**ARI**						
	Π_C	**SR_{10}**	**SR_{25}**	**SR_{50}**	**SR_{75}**	**HSS**	**STR**	Π_C	**SR_{10}**	**SR_{25}**	**SR_{50}**	**SR_{75}**	**HSS**
Aggregation	74,00	8,00	19,00	37,00	56,00	10,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
chainlink	27,00	3,00	7,00	14,00	21,00	5,00	E_1	1,00	$\triangledown$ 0,19	$\triangledown$ 0,31	1,00	1,00	1,00
Compound	62,00	7,00	16,00	31,00	47,00	8,00	E_1	0,85	$\triangledown$ 0,44	0,81	0,85	0,85	0,85
ds2c2sc13	147,00	15,00	37,00	74,00	111,00	18,00	E_1	1,00	1,00	1,00	1,00	1,00	1,00
							E_2	1,00	0,95	0,95	1,00	1,00	1,00
							E_3	1,00	$\triangledown$ 0,57	$\triangledown$ 0,57	$\triangledown$ 0,62	$\triangledown$ 0,78	$\triangledown$ 0,87
ds3c3sc6	65,00	7,00	17,00	33,00	49,00	7,00	E_1	0,90	0,90	0,90	0,90	0,90	$\triangledown$ 0,81
							E_2	0,59	0,58	0,59	0,59	0,59	$\triangledown$ 0,50
ds4c2sc8	89,00	9,00	23,00	45,00	67,00	9,00	E_1	0,87	0,87	0,87	0,87	0,87	0,87
							E_2	0,35	$\triangledown$ 0,18	0,30	0,30	0,30	$\triangledown$ 0,29
dyrskjot	25,00	3,00	7,00	13,00	19,00	7,00	E_1	0,55	$\triangledown$ 0,02	$\triangledown$ 0,05	0,50	0,55	0,54
Flame	25,00	3,00	7,00	13,00	19,00	5,00	E_1	0,94	$\triangledown$ 0,51	$\triangledown$ 0,71	0,94	0,94	0,94
glass	55,00	6,00	14,00	28,00	42,00	9,00	E_1	0,67	$\triangledown$ 0,60	0,65	0,65	0,67	0,65
							E_2	0,56	$\triangledown$ 0,43	$\triangledown$ 0,50	$\triangledown$ 0,50	0,55	0,56
							E_3	0,26	$\triangledown$ 0,20	0,24	0,24	0,26	0,26
golub	49,00	5,00	13,00	25,00	37,00	9,00	E_1	0,87	$\triangledown$ -0,01	$\triangledown$ 0,00	$\triangledown$ 0,68	$\triangledown$ 0,80	0,87
							E_2	0,94	$\triangledown$ -0,01	$\triangledown$ 0,01	0,94	0,94	$\triangledown$ 0,61
iris	37,00	4,00	10,00	19,00	28,00	6,00	E_1	0,82	$\triangledown$ 0,57	$\triangledown$ 0,57	$\triangledown$ 0,76	$\triangledown$ 0,76	$\triangledown$ 0,76
monkey	76,00	8,00	19,00	38,00	57,00	7,00	E_1	0,67	$\triangledown$ 0,27	$\triangledown$ 0,60	0,65	0,65	0,65
							E_2	0,83	$\triangledown$ 0,29	$\triangledown$ 0,38	$\triangledown$ 0,71	$\triangledown$ 0,71	$\triangledown$ 0,70
							E_3	0,65	$\triangledown$ 0,24	$\triangledown$ 0,41	$\triangledown$ 0,58	$\triangledown$ 0,58	$\triangledown$ 0,58
R15	154,00	16,00	39,00	77,00	116,00	23,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
spiral	33,00	4,00	9,00	17,00	25,00	6,00	E_1	1,00	$\triangledown$ 0,01	$\triangledown$ 0,02	$\triangledown$ 0,02	1,00	$\triangledown$ 0,92
spiralsquare	85,00	9,00	22,00	43,00	64,00	15,00	E_1	0,67	$\triangledown$ 0,37	$\triangledown$ 0,37	0,67	0,67	0,67
							E_2	1,00	1,00	1,00	1,00	1,00	1,00
Média	**66,87**	**7,13**	**17,27**	**33,80**	**50,53**	**9,60**		**0,80**	**0,49**	**0,55**	**0,72**	**0,77**	**0,76**

Tabela 10 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, BRP_{10}, BRP_{25}, BRP_{50}, BRP_{75} e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI.

Base de Dados	**Número de Soluções**						**ARI**						
	Π_C	**BRP$_{10}$**	**BRP$_{25}$**	**BRP$_{50}$**	**BRP$_{75}$**	**HSS**	**STR**	Π_C	**BRP$_{10}$**	**BRP$_{25}$**	**BRP$_{50}$**	**BRP$_{75}$**	**HSS**
Aggregation	74,00	8,00	19,00	37,00	56,00	10,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
chainlink	27,00	3,00	7,00	14,00	21,00	5,00	E_1	1,00	$\triangledown$ 0,31	$\triangledown$ 0,31	$\triangledown$ 0,50	1,00	1,00
Compound	62,00	7,00	16,00	31,00	47,00	8,00	E_1	0,85	$\triangledown$ 0,74	$\triangledown$ 0,78	0,85	0,85	0,85
ds2c2sc13	147,00	15,00	37,00	74,00	111,00	18,00	E_1	1,00	1,00	1,00	1,00	1,00	1,00
							E_2	1,00	0,95	1,00	1,00	1,00	1,00
							E_3	1,00	$\triangledown$ 0,61	$\triangledown$ 0,61	$\triangledown$ 0,63	$\triangledown$ 0,66	$\triangledown$ 0,87
ds3c3sc6	65,00	7,00	17,00	33,00	49,00	7,00	E_1	0,90	0,90	0,90	0,90	0,90	$\triangledown$ 0,81
							E_2	0,59	0,58	0,59	0,59	0,59	$\triangledown$ 0,50
ds4c2sc8	89,00	9,00	23,00	45,00	67,00	9,00	E_1	0,87	0,83	0,87	0,87	0,87	0,87
							E_2	0,35	$\triangledown$ 0,20	$\triangledown$ 0,20	0,30	0,30	$\triangledown$ 0,29
dyrskjot	25,00	3,00	7,00	13,00	19,00	7,00	E_1	0,55	$\triangledown$ 0,47	$\triangledown$ 0,47	0,55	0,55	0,54
Flame	25,00	3,00	7,00	13,00	19,00	5,00	E_1	0,94	$\triangledown$ 0,51	$\triangledown$ 0,71	$\triangledown$ 0,71	0,94	0,94
glass	55,00	6,00	14,00	28,00	42,00	9,00	E_1	0,67	$\triangledown$ 0,60	0,65	0,67	0,67	0,65
							E_2	0,56	$\triangledown$ 0,43	$\triangledown$ 0,50	0,55	0,55	0,56
							E_3	0,26	$\triangledown$ 0,20	0,24	0,26	0,26	0,26
golub	49,00	5,00	13,00	25,00	37,00	9,00	E_1	0,87	$\triangledown$ 0,68	$\triangledown$ 0,68	$\triangledown$ 0,68	$\triangledown$ 0,80	0,87
							E_2	0,94	0,94	0,94	0,94	0,94	$\triangledown$ 0,61
iris	37,00	4,00	10,00	19,00	28,00	6,00	E_1	0,82	$\triangledown$ 0,73	$\triangledown$ 0,76	$\triangledown$ 0,76	$\triangledown$ 0,76	$\triangledown$ 0,76
monkey	76,00	8,00	19,00	38,00	57,00	7,00	E_1	0,67	$\triangledown$ 0,61	0,62	0,65	0,67	0,65
							E_2	0,83	$\triangledown$ 0,67	$\triangledown$ 0,67	0,83	0,83	$\triangledown$ 0,70
							E_3	0,65	$\triangledown$ 0,58	$\triangledown$ 0,58	0,65	0,65	$\triangledown$ 0,58
R15	154,00	16,00	39,00	77,00	116,00	23,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
spiral	33,00	4,00	9,00	17,00	25,00	6,00	E_1	1,00	$\triangledown$ 0,02	$\triangledown$ 0,02	$\triangledown$ 0,02	1,00	$\triangledown$ 0,92
spiralsquare	85,00	9,00	22,00	43,00	64,00	15,00	E_1	0,67	$\triangledown$ 0,37	$\triangledown$ 0,37	0,67	0,67	0,67
							E_2	1,00	1,00	1,00	1,00	1,00	1,00
Média	**66,87**	**7,13**	**17,27**	**33,80**	**50,53**	**9,60**		**0,80**	**0,64**	**0,66**	**0,70**	**0,78**	**0,76**

Tabela 11 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, SRD_{10}, SRD_{25}, SRD_{50}, SRD_{75} e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI.

Base de Dados	**Número de Soluções**						**ARI**						
	Π_C	**SRD_{10}**	**SRD_{25}**	**SRD_{50}**	**SRD_{75}**	**HSS**	**STR**	Π_C	**SRD_{10}**	**SRD_{25}**	**SRD_{50}**	**SRD_{75}**	**HSS**
Aggregation	74,00	8,00	19,00	37,00	56,00	10,00	E_1	0,99	0,98	0,99	0,99	0,99	0,99
chainlink	27,00	3,00	7,00	14,00	21,00	5,00	E_1	1,00	$\triangledown$ 0,19	$\triangledown$ 0,21	$\triangledown$ 0,40	1,00	1,00
Compound	62,00	7,00	16,00	31,00	47,00	8,00	E_1	0,85	$\triangledown$ 0,44	0,81	0,85	0,85	0,85
ds2c2sc13	147,00	15,00	37,00	74,00	111,00	18,00	E_1	1,00	1,00	1,00	1,00	1,00	1,00
							E_2	1,00	0,95	0,95	1,00	1,00	1,00
							E_3	1,00	$\triangledown$ 0,57	$\triangledown$ 0,57	$\triangledown$ 0,62	$\triangledown$ 0,66	$\triangledown$ 0,87
ds3c3sc6	65,00	7,00	17,00	33,00	49,00	7,00	E_1	0,90	$\triangledown$ 0,50	$\triangledown$ 0,50	0,90	0,90	$\triangledown$ 0,81
							E_2	0,59	$\triangledown$ 0,25	$\triangledown$ 0,25	0,59	0,59	$\triangledown$ 0,50
ds4c2sc8	89,00	9,00	23,00	45,00	67,00	9,00	E_1	0,87	0,83	0,87	0,87	0,87	0,87
							E_2	0,35	$\triangledown$ 0,18	0,30	0,30	0,30	$\triangledown$ 0,29
dyrskjot	25,00	3,00	7,00	13,00	19,00	7,00	E_1	0,55	$\triangledown$ 0,02	$\triangledown$ 0,05	$\triangledown$ 0,47	0,55	0,54
Flame	25,00	3,00	7,00	13,00	19,00	5,00	E_1	0,94	$\triangledown$ 0,51	$\triangledown$ 0,71	0,94	0,94	0,94
glass	55,00	6,00	14,00	28,00	42,00	9,00	E_1	0,67	$\triangledown$ 0,04	0,65	0,65	0,67	0,65
							E_2	0,56	$\triangledown$ 0,03	$\triangledown$ 0,50	$\triangledown$ 0,50	0,52	0,56
							E_3	0,26	$\triangledown$ 0,01	0,24	0,24	0,24	0,26
golub	49,00	5,00	13,00	25,00	37,00	9,00	E_1	0,87	$\triangledown$ -0,01	$\triangledown$ 0,00	$\triangledown$ 0,68	0,87	0,87
							E_2	0,94	$\triangledown$ -0,01	$\triangledown$ 0,01	0,94	0,94	$\triangledown$ 0,61
iris	37,00	4,00	10,00	19,00	28,00	6,00	E_1	0,82	$\triangledown$ 0,57	$\triangledown$ 0,57	$\triangledown$ 0,76	$\triangledown$ 0,76	$\triangledown$ 0,76
monkey	76,00	8,00	19,00	38,00	57,00	7,00	E_1	0,67	$\triangledown$ 0,20	$\triangledown$ 0,60	0,65	0,65	0,65
							E_2	0,83	$\triangledown$ 0,29	$\triangledown$ 0,38	$\triangledown$ 0,71	$\triangledown$ 0,71	$\triangledown$ 0,70
							E_3	0,65	$\triangledown$ 0,24	$\triangledown$ 0,41	$\triangledown$ 0,58	$\triangledown$ 0,58	$\triangledown$ 0,58
R15	154,00	16,00	39,00	77,00	116,00	23,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
spiral	33,00	4,00	9,00	17,00	25,00	6,00	E_1	1,00	$\triangledown$ 0,00	$\triangledown$ 0,01	$\triangledown$ 0,02	1,00	$\triangledown$ 0,92
spiralsquare	85,00	9,00	22,00	43,00	64,00	15,00	E_1	0,67	$\triangledown$ 0,37	$\triangledown$ 0,37	0,67	0,67	0,67
							E_2	1,00	1,00	1,00	1,00	1,00	1,00
Média	**66,87**	**7,13**	**17,27**	**33,80**	**50,53**	**9,60**		**0,80**	**0,41**	**0,52**	**0,69**	**0,77**	**0,76**

Tabela 12 – Número de soluções e qualidade da melhor solução (ARI) retornados por Π_C, Div_{10}, Div_{25}, Div_{50}, Div_{75} e HSS. Os valores destacados em vermelho ($\triangledown$) indicam piora maior do que $0,05$ no valor do ARI.

Base de Dados	**Número de Soluções**						**ARI**						
	Π_C	**Div$_{10}$**	**Div$_{25}$**	**Div$_{50}$**	**Div$_{75}$**	**HSS**	**STR**	Π_C	**Div$_{10}$**	**Div$_{25}$**	**Div$_{50}$**	**Div$_{75}$**	**HSS**
Aggregation	74,00	8,00	19,00	37,00	56,00	10,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
chainlink	27,00	3,00	7,00	14,00	21,00	5,00	E_1	1,00	1,00	1,00	1,00	1,00	1,00
Compound	62,00	7,00	16,00	31,00	47,00	8,00	E_1	0,85	$\triangledown$ 0,77	0,84	0,85	0,85	0,85
ds2c2sc13	147,00	15,00	37,00	74,00	111,00	18,00	E_1	1,00	1,00	1,00	1,00	1,00	1,00
							E_2	1,00	$\triangledown$ 0,87	$\triangledown$ 0,87	$\triangledown$ 0,87	0,99	1,00
							E_3	1,00	0,98	0,98	1,00	1,00	$\triangledown$ 0,87
ds3c3sc6	65,00	7,00	17,00	33,00	49,00	7,00	E_1	0,90	$\triangledown$ 0,64	$\triangledown$ 0,64	0,90	0,90	$\triangledown$ 0,81
							E_2	0,59	$\triangledown$ 0,37	$\triangledown$ 0,37	0,58	0,59	$\triangledown$ 0,50
ds4c2sc8	89,00	2,00	23,00	45,00	67,00	9,00	E_1	0,87	0,82	0,82	0,82	0,82	0,87
							E_2	0,35	$\triangledown$ 0,17	0,35	0,35	0,35	$\triangledown$ 0,29
dyrskjot	25,00	3,00	7,00	13,00	19,00	7,00	E_1	0,55	$\triangledown$ 0,47	0,55	0,55	0,55	0,54
Flame	25,00	3,00	7,00	13,00	19,00	5,00	E_1	0,94	$\triangledown$ 0,33	0,92	0,92	0,94	0,94
glass	55,00	6,00	14,00	28,00	42,00	9,00	E_1	0,67	0,67	0,67	0,67	0,67	0,65
							E_2	0,56	0,55	0,55	0,55	0,55	0,56
							E_3	0,26	0,26	0,26	0,26	0,26	0,26
golub	49,00	5,00	13,00	25,00	37,00	9,00	E_1	0,87	$\triangledown$ 0,53	$\triangledown$ 0,68	$\triangledown$ 0,68	0,87	0,87
							E_2	0,94	$\triangledown$ 0,78	0,94	0,94	0,94	$\triangledown$ 0,61
iris	37,00	4,00	10,00	19,00	28,00	6,00	E_1	0,82	0,82	0,82	0,82	0,82	$\triangledown$ 0,76
monkey	76,00	8,00	19,00	38,00	57,00	7,00	E_1	0,67	$\triangledown$ 0,47	$\triangledown$ 0,55	0,67	0,67	0,65
							E_2	0,83	$\triangledown$ 0,63	0,83	0,83	0,83	$\triangledown$ 0,70
							E_3	0,65	$\triangledown$ 0,46	0,65	0,65	0,65	$\triangledown$ 0,58
R15	154,00	16,00	39,00	77,00	116,00	23,00	E_1	0,99	0,99	0,99	0,99	0,99	0,99
spiral	33,00	4,00	9,00	17,00	25,00	6,00	E_1	1,00	$\triangledown$ 0,56	$\triangledown$ 0,92	1,00	1,00	$\triangledown$ 0,92
spiralsquare	85,00	9,00	22,00	43,00	64,00	15,00	E_1	0,67	0,63	0,63	0,63	0,67	0,67
							E_2	1,00	1,00	1,00	1,00	1,00	1,00
Média	**66,87**	**6,67**	**17,27**	**33,80**	**50,53**	**9,60**		**0,80**	**0,67**	**0,75**	**0,78**	**0,80**	**0,76**