4
Classificação Automática de dados semi-estruturados

Este capítulo apresenta a implementação de um sistema de classificação automática de dados baseada numa estrutura de dados chamada *frame* e nas teorias de classificação, nos algoritmos e nas técnicas de agrupamento apresentadas ao longo deste trabalho. A seção 4.1 apresenta os fundamentos utilizados na estratégia de classificação e a noção de *frames*, a seção 4.2 apresenta os detalhes da implementação do sistema e, por fim, a seção 4.3 apresenta uma análise feita dos dados obtidos a partir dos testes realizados.

4.1.
Fundamentos

4.1.1.
Noção de categorização

A categorização é o processo pelo qual idéias e objetos são reconhecidos, diferenciados e classificados.

Em uma visão objetivista, podemos identificar ou entender um objeto ou idéia através de um grupo de propriedades inerentes a eles. De acordo com Lakoff [27], este não é o único modo que as pessoas utilizam para categorizar coisas ou experiências. Na verdade, o que se faz é compreender o mundo de uma maneira flexível, decidindo se um objeto pertence ou não a uma categoria considerando não somente as propriedades inerentes ao objeto. Desta forma podemos considerar alguns pontos:

- De acordo com Rosch [28], as pessoas categorizam “coisas” em termos de protótipos, porém existem “coisas” que são não-prototípicas e, assim, estes objetos devem ser categorizados pelas relações com os protótipos ou por sua semelhança.

- Propriedades interacionais, que incluem propriedades funcionais, perceptivas, atividade motora e proposital.

- Visualizar as categorias de acordo com o seu propósito. Para isso, existem modificadores lingüísticos que escolhem o protótipo para uma categoria e que definem vários tipos de relacionamentos para ele. Por exemplo: Par Excellence (membros prototípicos de uma categoria), Strictly Speaking (membros não-prototípicos que pertencem à categoria), Loosely Speaking (membros que pertencem a uma categoria por compartilhar propriedades centrais, exemplo: membro: baleia, categoria: peixes), Technically (membro que pertence a uma categoria de acordo com algum aspecto técnico) e outros que ajudam a determinar a categoria para um objeto de acordo com diferentes propósitos.

O que vimos até agora é como as pessoas reconhecem objetos em sua forma geral, ou seja, como a mente humana funciona. De acordo com Rosch et al [28], existe um nível básico de categorias que retém mais informações, ou seja, existe um nível de abstração privilegiado em uma taxonomia de objetos. Contudo, descobrir onde está o nível básico, depende do conceito a ser estudado, por exemplo, o nível básico de um conceito como “animal” está em algum nível dentro (mais específico) desta classe, como “pássaro”, já em outro conceito como “móveis”, teríamos um conceito básico como “cadeira”. Dependendo dos casos, o nível pode ser um pouco mais específico ou geral.

O nível básico de Rosch continuou a ser desenvolvido e se tornou conhecido como a teoria do protótipo e níveis básicos ou, simplesmente, teoria do protótipo. Esta teoria é amplamente respeitada e representa um marco em psicologia experimental, revolucionando idéias de categorização e substituindo teorias clássicas que definem categorias mais rígidas. A teoria do protótipo é definida como o membro mais central de uma categoria e a estrutura dessas categorias é radial, isto é, alguns dos membros (mais central) são mais representativos que outros (menos central). Assim, propomos uma estratégia de classificação automática de dados semi-estruturados baseada na teoria do protótipo utilizando frames.

4.1.2. Noção de frames

4.1.2.1. Definição de frames

A noção de *frames* foi introduzida por Marvin Minsky em 1975 em seu artigo "*A framework for representing knowledge*" [24], onde ele define *frame* como uma estrutura de dados para representar uma situação estereotipada ou um conceito, como "*being in a certain kind of living room*" ou "*going to a child's party*". A idéia fundamental é bem simples, *frames* podem ser compostos de atributos, também conhecidos como *slots*, e esses *slots* podem possuir valores associados a ele.

Uma definição bem intuitiva de *frames* no contexto do modelo Entidade-Relacionamento é descrito em [25]. Nele, temos que o *slot* é uma expressão da forma "P:V" ou "P:", aonde P e V são denominados como *slot name* e *slot value*, respectivamente. Além disso, o *slot* deve satisfazer uma das seguintes condições:

- P é um atributo da entidade sendo descrita e V, se definido, é um valor único (atributo mono-valorado, por suposição), ou

- P é da forma R/1, onde R é um relacionamento binário em que a entidade é o primeiro participante e V, se definido, é um único valor ou um grupo de valores (o relacionamento é não total e multivalorado, por suposição), ou

- P é da forma R/2, onde R é um relacionamento binário em que a entidade é o segundo participante e V, se definido, é um único valor ou um grupo de valores (o relacionamento é não total e multivalorado, por suposição).

Então, o *frame* pode ser definido como um grupo de *slots* distintos. Um *top frame* é um grupo vazio. Uma instância de um *frame* é um *frame* cujos *slots* estão todas na forma "P:V", e um *class frame* é um *frame* com pelo menos um *slot* da forma "P:". Vejamos alguns exemplos:

Top frame

[nome: , idade: , área: , nível:] - (classe estudante)

Instância de um *frame*

[nome: João, idade: 20, área: Engenharia, nível: graduação] - (instância da classe estudante)

Class frame

[nome: , idade: , área: Direito, nível:] - (classe estudante de direito)

4.1.2.2. Sistemas baseados em frames

Sistemas baseados em *frames* são sistemas de representação do conhecimento que usam *frames* como um recurso primário para representar um domínio do conhecimento. Como podemos notar, *frames* por si só não são muito úteis, mas um conjunto de *frames* é uma ferramenta poderosa. Vejamos, por exemplo, um conjunto de *frames* como uma rede semântica, tabela 3.

Frame name	***Slot***	***Slot value***
Bob	Is a	Builder
	Owns	Fido
	Eats	Cheese
Fido	Is a	Dog
	Chases	Fang
Fang	Is a	Cat
	Chases	Mice
Mice	Eats	Cheese
Cheese		
Builder		
Dog		
Cat		

Tabela 3 Exemplo de relacionamentos entre frames.

Também podemos visualizar o diagrama na figura 12, o que nos permite perceber uma relação entre os *frames* de forma mais clara.

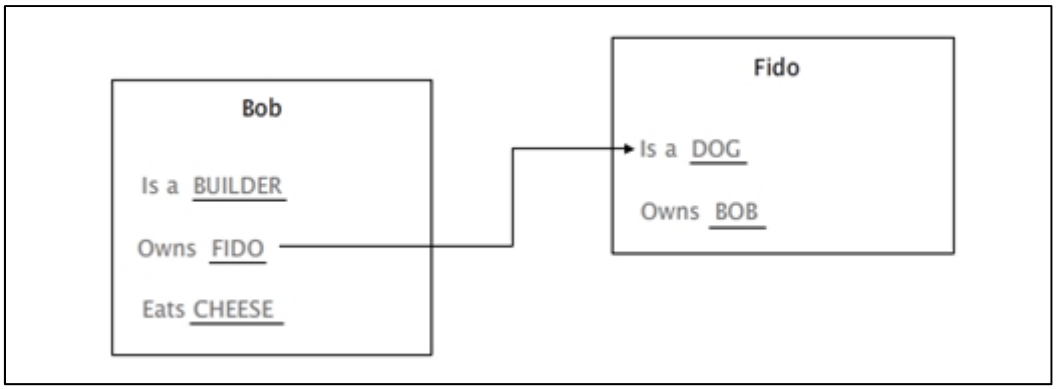

Figura 12 Diagrama de relacionamentos entre frames.

Conforme pode ser visto neste diagrama, Fido é uma instância da classe Dog. Além disso, a noção de herança pode ser facilmente representada em *frames*, imaginemos que todos os *Dogs* são *Mammals*, então teremos pelo menos uma superclasse (que especifica propriedades) *Mammals* de *Dogs* e Fido, e uma subclasse (que herda propriedades da superclasse) *Dogs* de *Mammals*.

Baseados na proposta original de *frames* [25], muitos sistemas de representação do conhecimento têm sido criados e a teoria de *frames* tem se desenvolvido. Algumas aplicações do uso de *frames* podem ser vistas em sistemas de inferência, explanação, *problem solving* e classificação. A maior vantagem de se utilizar *frames*, é que toda a informação do objeto que estamos tratando está em apenas um lugar, o que nos permite um acesso mais rápido em uma dedução [26].

4.2. Implementação

O processo de classificação automática de dados utiliza algoritmos não-supervisionados para encontrar uma taxonomia. A grande vantagem da sua utilização é não haver a necessidade do conhecimento prévio da massa de dados a ser tratada e nem ter a árdua tarefa de treinar previamente uma massa de dados para posterior classificação.

Identificamos as seguintes macro-etapas para um processo de classificação automática:

- Transformar dados semi-estruturados na forma de *frames*;
- Determinar o número e encontrar os *clusters* do conjunto inicial de *frames* (nível básico, ver seção 4.1);
- Especializar os *clusters* encontrados ao longo do processo;
- Generalizar os *clusters* encontrados ao longo do processo.

Através deste processo foi possível definir cada método ou técnica utilizada na criação deste sistema. Vejamos a seguir os detalhes deste processo de classificação automática.

4.2.1. Entrada de dados

Como visto na seção 4.1, os *frames* são compostos de atributos (*slots)* e valores associados a eles ou não. Sendo assim, o slot é caracterizado por uma expressão da forma “P:V” ou “P:”, onde P e V são denominados como atributo e valor, respectivamente.

Respeitando a definição de *frames*, construímos um padrão de linguagem de marcação utilizando XML (eXtensible Markup Language) para compor a entrada de dados do sistema proposto, vide código abaixo.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
	<frames>
		<frame id="IDENTIFICADOR DO OBJETO" nome="NOME DO OBJETO" nomeClasse="NOME DA CLASSE A QUE PERTENCE" classe="IDENTIFICADOR DA CLASSE A QUE PERTENCE">
				<slot attributeName="NOME DO ATRIBUTO" attributeType="TIPO DO DADO">
				<value><![CDATA[ VALOR DO ATRIBUTO]]></value>
			</slot>
		</frame>
	</frames>
```

A seguir vejamos a descrição dos marcadores utilizados:

<frames></frames> - Marcação da lista de *frames* a ser provida ao sistema.

<frame></frame> - Marcador que contém as informações sobre o *frame*. Aninhado ao marcador do *frame*, teremos o marcador <*slot*> que descreve os atributos pertencentes ao *frame*. E, por sua vez, o marcador <slot> poderá possuir valores associados que são descritos pelo marcador <*value*>.

O marcador <*frame*> possui propriedades que descrevem o objeto do *frame* como:

- id - identificador do objeto;
- nome - nome do objeto.

Além dessas propriedades, este marcador pode conter propriedades extras que podem ser utilizadas para análise dos dados como, por exemplo, o método visto na seção 3.5 para validação do *cluster* pelo método relativo, ou seja, utilizando uma classificação prévia para verificar a corretude do agrupamento realizado. Estas propriedades são:

- classe – identificador da classe a que um objeto pertencia previamente;
- nomeClasse - nome da classe a que um objeto pertencia previamente.

Em seguida, descrevemos o marcador <slot> que representa os atributos que um *frame* pode conter.

<slot></slot> - Marcador que descreve atributos pertencentes a um determinado *frame.* As propriedades que o descrevem são:

- attributeName – nome do atributo;
- attributeType[2] – tipo de dados do atributo (ex.: String, booleano, ...).

Por fim, o marcador <value> responsável pela descrição dos valores associados ao atributo ao qual ele pertence.

[2] Attribute type é utilizado para medir a distância entre objetos de acordo com o tipo de dados a que este pertence, como visto na seção 3.5.

<value></value> - Valores associados a um determinado atributo.

4.2.2. Determinando o número de clusters

Esta é uma das etapas mais importantes no processo de agrupamento automático, pois determina o número de grupos que uma massa de dados possui. Usualmente, técnicas de agrupamento utilizam grupos pré-estabelecidos ou métodos supervisionados tornando necessário o conhecimento prévio dos grupos existentes na massa de dados. Desta maneira, apresentaremos o método para determinar o número de *clusters* que utilizamos neste trabalho e uma breve análise de outros métodos destinados a abordar este problema.

Uma análise feita por [22] sobre as técnicas apresentadas na seção 3.7, nos mostra que a maioria destas técnicas não funciona como deveriam na prática. Os métodos de *Cross Validation* e *Penalized likelihood estimation* são computacionalmente custosos e, normalmente, necessitam ser executados diversas vezes para que se tenha uma estimativa razoável. Já os métodos de *Resampling* e *Permutation tests* são inviáveis, pois devem executar os algoritmos de agrupamento centenas ou milhares de vezes. E, por fim, os métodos baseados em *Finding the knee of error curve* que geralmente demoram mais tempo para avaliar os *clusters* do que o próprio algoritmo que o gerou.

Um método sugerido por Kaufman e Rousseeuw [20] é o método de silhuetas (*silhouettes method*) utilizado para determinar se um objeto pertence ou não a um grupo e, também, avaliar a qualidade do *cluster* obtido.

A silhueta de um determinado objeto é obtida através da form. (6).

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{6}$$

onde a(i) representa a distância média entre o objeto i e os demais objetos do próprio *cluster* C_i; Para todo o *cluster* $C \neq C_i$, d(i, C) é a distância média de i para os objetos de C. Depois de computadas todos d(i, C) para todos os *clusters* $C \neq C_i$, b(i) corresponderá à menor média entre eles.

Assim, s(i) representa a medida de quão bem um objeto está localizado em relação ao *cluster* vizinho. Se s(i) for igual a um, temos um objeto classificado perfeitamente. Caso s(i) seja igual a zero, temos uma situação em que há uma incerteza na alocação do objeto no *cluster* atual ou em algum outro *cluster* vizinho. Agora, se s(i) for igual a menos um, é bem provável que o objeto tenha sido classificado erradamente. De posse de todos os valores de s(i) de um dado *cluster,* sua média é conhecida como a largura média da silhueta deste *cluster*. E a média de todas as larguras dos k *clusters* é conhecida como s(k), onde o k é o número de *clusters* e s(k) é a média da qualidade do agrupamento realizado sobre uma determinada massa de dados. Assim, determina-se o k apropriado como sendo aquele que possui a média máxima entre os testes realizados.

Através deste método não é possível identificar a silhueta para o caso de k igual a um, já que esta utiliza medidas inter-*clusters* para sua determinação. Desta maneira, ainda é necessário verificar o caso de um determinado grupo de objetos não ter que ser particionado em dois ou mais grupos, ou seja, k ser igual a um. Visando solucionar essa deficiência, adicionamos duas heurísticas para determinar quando não é interessante o algoritmo separar os objetos de um grupo de dados.

A primeira heurística verifica se o nível de coesão e isolamento (medida calibrada pelo usuário através do sistema) intra-*cluster* e inter-*clusters* são satisfatórios para k igual a dois. Se isso for verdade, então o *cluster* é particionado em dois grupos, caso contrário este grupo não mais sofrerá particionamentos.

A segunda heurística testa se a média da qualidade do agrupamento para k elementos é superior à média configurada pelo usuário ou para o s(k) padrão previamente configurado como zero. Se o valor de s(k) é superior ao estipulado então o número de *clusters* retornado é k, caso contrário retorna k igual a um.

4.2.3. Algoritmo de Agrupamento

Embora existam diversos algoritmos utilizados no ato de agrupar dados, nos baseamos na teoria do protótipo para eleger um algoritmo que melhor

representasse esta teoria e, por conseqüência, a visão das pessoas sobre objetos (coisas, animais, pessoas, objetos materiais e outros).

A teoria do protótipo é descrita como sendo uma estrutura radial, onde os membros mais próximos do centro são mais representativos e os mais distantes menos representativos. A primeira tarefa é encontrar os representantes de cada grupo, elementos prototípicos da massa de dados a ser agrupada. De acordo com Rosch[28] este nível é determinado como básico.

Para seguir a estrutura radial a qual a teoria do protótipo se refere, necessitamos de um algoritmo que mantenha essa forma e, por isso, escolhemos como base o algoritmo K-Means, como visto na seção 3 e na figura 13.

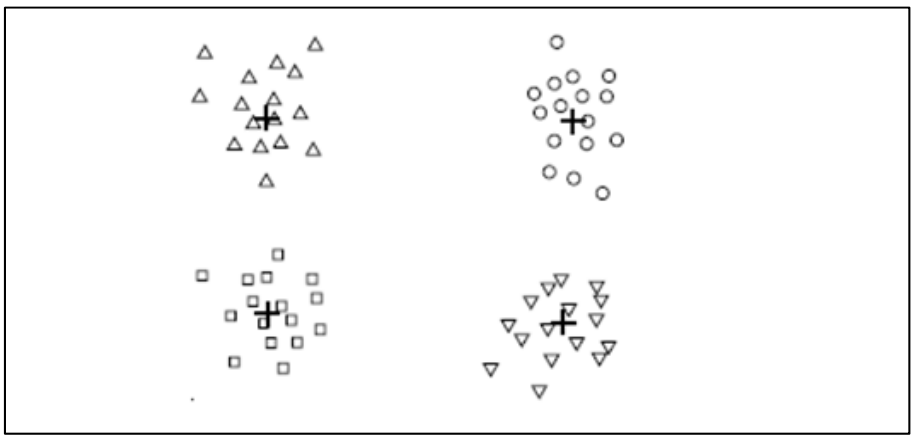

Figura 13 Clusterização Radial.

O sinal de **+** na figura 13 representa o centróide encontrado após a execução do algoritmo K-Means, satisfazendo assim, parte da teoria. Contudo, ainda é necessário eleger um elemento que represente prototipicamente um grupo de objetos e por este motivo utilizaremos uma variação deste algoritmo conhecido como K-Medoide, pois ao invés de determinar o centróide através de sua posição geométrica, este elege um dos próprios membros do grupo para representante. Vejamos a seguir, em pseudo-código, o algoritmo do K-Medóide implementado neste trabalho.

Algoritmo K-Medóide

Escolhe-se k medóides iniciais aleatoriamente.
Enquanto a condição de parada não é satisfeita
Visita todos os elementos não-medóides e os associa ao medóide mais próximo.
Recalcula-se o melhor representante para cada *cluster.*

Algoritmo Melhor Representante de um *cluster*

Calcula-se a distância de todos os objetos ao *cluster* atual.
Escolhe-se um novo representante para o *cluster* em questão.
Calcula-se a distância dos demais objetos até o novo *cluster.*
Se a soma das distâncias do *cluster* atual for maior que a do novo *cluster*
Então o novo representante é atualizado.
Caso contrário
Representante atual não é alterado.

Após determinarmos o número de grupos que a massa de dados possui, executamos o algoritmo visto acima. Ao final da execução obtemos k representantes e a massa de objetos dividida em k classes. Para obtermos uma taxonomia completa, necessitamos ainda executar as etapas para generalizar e especializar as classes obtidas no primeiro passo deste algoritmo. Consideramos que a primeira execução deste algoritmo corresponde ao nível básico (protótipos) de Rosch supracitado.

4.2.4. Algoritmo de Especialização

Como o próprio nome diz, o objetivo deste algoritmo é especializar conceitos dentro de uma massa de dados, por exemplo, dada uma massa de objetos, executa-se o algoritmo de especialização para encontrar subgrupos de objetos. O algoritmo é novamente executado para cada subgrupo de dados até que o critério de parada seja atingido. O critério de parada é definido a partir do algoritmo responsável por determinar o número de *clusters* (ver seção 4.2.2), caso este algoritmo retorne k=1, então o algoritmo de especialização pára de ser executado. Vejamos o pseudo-código deste algoritmo.

Algoritmo de especialização

Executa o algoritmo para determinar o número de *clusters* (k) que uma massa de dados possui.
se k=1 então
retorna;
caso contrário
Executa o algoritmo de agrupamento para o k encontrado;
Para cada medóide encontrado
executa o algoritmo de especialização.

4.2.5. Algoritmo de Generalização

4.2.5.1. Noções básicas

Ao contrário do algoritmo de especialização, este algoritmo generaliza grupos de objetos, por exemplo, dado dois grupos de objetos, o algoritmo elegerá um novo objeto, pertencente a um dos dois grupos, que melhor os representará.

Existem três diferentes abordagens para definir os elementos que representarão os grupos de objetos: *Abstrata*, *Medóide* e *Híbrida*. Independente da abordagem, todas elas utilizam o mesmo algoritmo base para encontrar os grupos de objetos que serão representados por um novo elemento.

Algoritmo de generalização base

Dado dois ou mais grupos de objetos e a distância entre eles é menor que a distância máxima para aglomeração. (Distância máxima configurada no sistema)
Identificam-se os dois grupos mais próximos.
Executa o método (abstrata, híbrida ou medóide) sobre os grupos mais próximos.
Senão
Os grupos não são aglomerados e retorna.
Executa o algoritmo de generalização para os grupos restantes.

Para compreender melhor o algoritmo de generalização base exemplificaremos um caso de agrupamento. Uma base de dados fornece ao sistema objetos de dados estruturados na forma de *frames.* Esses objetos passam inicialmente pelo algoritmo de agrupamento para obtenção do nível básico. Supondo que tenhamos três tipos de objetos (pessoa, estudante e empregado), um possível agrupamento básico pode ser visto na figura 14.

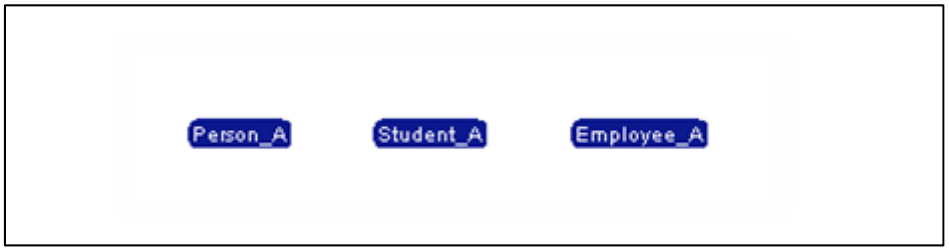

Figura 14 Exemplo da terceira etapa do processo de classificação automática de dados semi-estruturados. Agrupamento de dados de medóides.

Neste caso, o algoritmo encontrou três representantes básicos. Estes objetos são os elementos mais representativos de seus grupos, a figura exibe apenas os representantes (medóides) dos grupos encontrados.

O próximo passo do algoritmo de agrupamento é dado pelo algoritmo de especialização. Neste exemplo, uma possível especialização para essa massa de dados é dada pela figura 15.

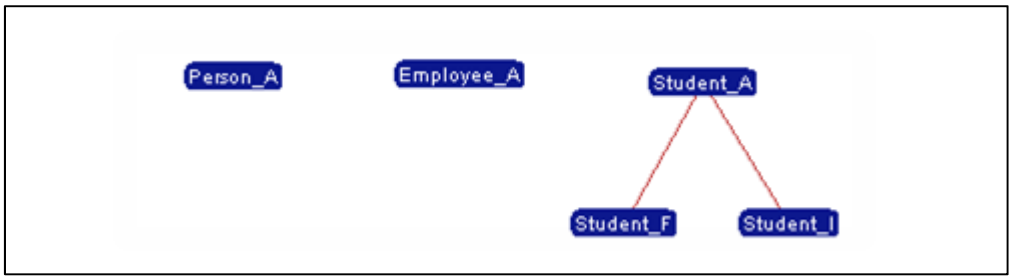

Figura 15 Exemplo do algoritmo de especialização sobre a classe de estudantes.

Como pode ser visto na figura acima, a classe representada pelo medóide *Student_A* foi dividida em mais duas classes, *Student_F* e *Student_I*, que representam estudantes de engenharia e direito, respectivamente. À frente, veremos em detalhe todos os objetos e suas propriedades; aqui basta saber que a estrutura utilizada é a de um grafo orientado e acíclico e como os objetos são preparados para a execução do algoritmo base de generalização.

Depois do algoritmo de especialização ser executado, o algoritmo base de generalização pode ser aplicado e o primeiro passo do algoritmo consiste em verificar a existência de grupos (representados por medóides) disjuntos, ou tecnicamente falando, existência de raízes no grafo como os representantes *Person_A, Employee_A* e *Student_A*. O algoritmo identifica os dois grupos mais próximos através de seus representantes e os aglomera formando apenas um grupo. O novo representante deste grupo é definido a partir da abordagem escolhida (abstrata, medóides ou híbrida).

4.2.5.2. Medóides

A generalização dada pelos medóides é uma abordagem que utiliza dos próprios elementos da massa de dados para encontrar um elemento representativo para um determinado grupo de elementos. O funcionamento deste algoritmo é aparentemente simples, porém a dificuldade da implementação é alta. Por exemplo, supondo que os dois elementos mais próximos da figura 15 sejam Employee_A e Student_A, então o seguinte algoritmo é executado:

Algoritmo de generalização – Medóide

Executa algoritmo de agrupamento para k=1 sobre os medóides recebidos mais próximos e recebe o elemento mais representativo do agrupamento destes medóides.
Remove o elemento mais representativo e adiciona-o como o elemento mais representativo dos medóides recebidos.
Se o elemento representativo escolhido já for um medóide em algum nível da hierarquia abaixo então
Executa o algoritmo de generalização – Medóide – para obter um elemento representativo para este grupo (grupo que perdeu seu representante).

Tendo *Employee_A* e *Student_A* como os medóides mais próximos, suponhamos que o elemento mais representativo dessa generalização seja o medóide *Student_F.* Então, teremos que o elemento mais representativo dos grupos *Employee_A* e *Student_A* é o elemento *Student_F*. Entretanto, como pode ser visto na figura 15, o elemento escolhido é um medóide de outro grupo em uma hierarquia abaixo. Desta maneira, quando realocarmos o elemento *Student_F* para ser o representante dos grupos de *Employee_A* e *Student_A* seu

antigo grupo ficará sem representante. Assim, executamos novamente o algoritmo de generalização para o grupo sem representante para atribuir-lhe um novo representante até que não mais ocorram realocações de medóides.

O interessante desta abordagem é a utilização dos próprios elementos da massa de dados como representantes dos grupos formando uma taxonomia de elementos representativos dos grupos (elementos prototípicos).

4.2.5.3. Abstrata

Ao contrário da abordagem realizada pelos *Medóides*, a abordagem *Abstrata* não utiliza os elementos de sua própria massa de dados para elegê-los como medóides. Ao invés disso, cria-se um novo elemento representante, um elemento abstrato, através do conceito de *meet* (Δ) apresentado em [25].

Dado dois *frames*, F e G, define-se FΔG como o *frame* M tal que o *slot* s$\in$M sse:

(1) s$\in$F e existe um g$\in$G tal que g $\sqsubseteq$ s, ou

(2) s$\in$G e existe um f$\in$F tal que f $\sqsubseteq$ s

Note que a criação do *frame* M é sempre possível, pois quando não existem *slots* que satisfaçam uma das condições acima, o *frame* M é vazio, isto é, um *top frame*, conceito visto anteriormente.

Entretanto, existe no método aqui proposto uma pequena diferença em relação à abordagem citada por [25]. Aqui, o conceito de *meet* precisa satisfazer mais uma condição:

(3) distância(F, G) ≤ α

onde α é a distância mínima definida manualmente e a função de distância é dada pela similaridade entre os *frames* em questão.

O parâmetro baseia-se na similaridade dos *frames* e caso a similaridade seja menor que a definida, a operação de *meet* não é realizada. No caso do

parâmetro ser configurado para o grau de similaridade zero, então a operação *meet* sempre será realizada.

O algoritmo proposto para esta abordagem é realizado em duas etapas. A primeira etapa realiza a generalização dos medóides encontrados pelo algoritmo de especialização e do nível básico. A segunda etapa realiza a generalização dos medóides acima do nível básico.

Algoritmo de generalização – preparação (1ª etapa)

Para cada medóide encontrado na etapa do algoritmo de especialização e nível básico que não seja abstrato, faça o seguinte:

Crie um novo elemento "abstrato" através do *meet* dos medóides logo abaixo dele.

Substitua este medóide pelo elemento "abstrato" criado.

Realoca o elemento no nível mais baixo da classificação.

Algoritmo de generalização – Abstrata (2ª etapa)

Executa o algoritmo base de generalização

Se os elementos recebidos satisfazem as condições 1, 2 e 3 então:

Cria um novo elemento "abstrato" através da operação *meet*.

O elemento "abstrato" criado é o novo representante dos dois medóides recebidos.

Executa o algoritmo de generalização da 2ª etapa.

Considerando que após a execução desta etapa obteremos uma estrutura baseada em árvore, os objetos de dados reais (não abstratos) serão os filhos desta estrutura. Neste caso, teremos uma classificação baseada em *frames* (abstratos) criados a partir dos medóides descobertos através de todo o processo. A característica interessante desta abordagem é possuir um elemento que representa os medóides abaixo na hierarquia através do conceito de *meet*. Por exemplo, a generalização dos medóides:

F1 = [nome: Maria, área: engenharia, nível: graduação]
F2 = [nome: João, área: direito, nível: graduação]

é dado pelo elemento abstrato criado a partir da operação F1 Δ F2, obtendo-se o novo medóide F3:

F3 = [nome:, área:, nível: graduação]

4.2.5.4. Híbrida

Como o próprio nome sugere esta abordagem é uma combinação das abordagens *Abstrata* e *Medóide,* sendo este algoritmo o mais simples. Após a etapa de especialização, onde obtemos todos os níveis hierárquicos inferiores, executamos a primeira etapa da generalização abstrata, onde obtemos os níveis superiores ao nível básico.

A diferença entre as abordagens anteriores é que este algoritmo não realiza a troca dos medóides encontrados no processo de especialização. Desta maneira, temos uma classificação híbrida, ou seja, do nível básico para baixo temos uma classificação das instâncias de objetos reais e do nível básico para cima temos uma criação abstrata de classes.

4.2.6. Isolamento e Coesão

As técnicas de isolamento (separação inter-*cluster*) e coesão são utilizadas para validação do processo de agrupamento. Neste trabalho, a validação deste processo é realizada a partir das duas técnicas apresentadas anteriormente, coesão e isolamento e o objetivo é verificar se o processo de determinação do número de clusters de uma massa de dados é satisfatório.

A coesão é medida da seguinte maneira:

$$\frac{\sum_{i=0}^{j} d(a_i, m)}{|a|} \tag{7}$$

onde *m* é o medóide do *cluster*, $d(a_i,m)$ é a função de distância entre o objeto a_i e o objeto *m*, a_i são os objetos ligados ao medóide *m* e |*a*| é o total de objetos ligados a *m*. Assim temos que a coesão é dada pela soma da distância entre todos objetos e o medóide, dividido pelo número de objetos deste *cluster*, ver fig.16.

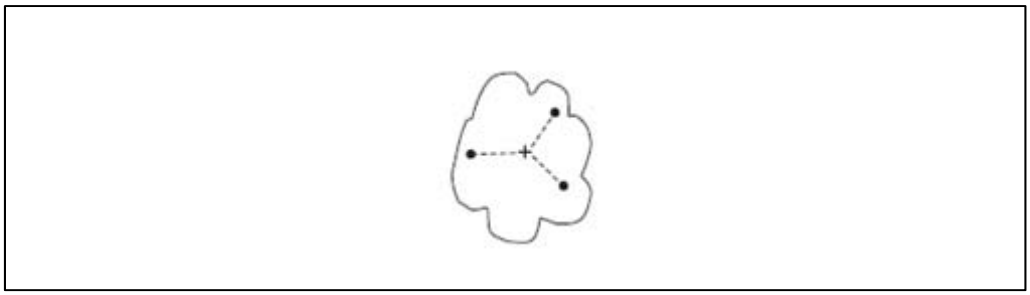

Figura 16 Ilustração da coesão de um cluster em relação a um elemento central (protótipo).

Já no isolamento inter-*clusters,* utilizamos a distância entre os medóides de cada agrupamento para medir o isolamento entre os grupos, ver figura 17. Por utilizarmos o algoritmo K-Medoide, temos agrupamentos com uma estrutura radial e, assim, medimos o isolamento entre os medóides para definirmos o isolamento entre os grupos. Com isso, o cálculo é mais fácil e menos custoso computacionalmente.

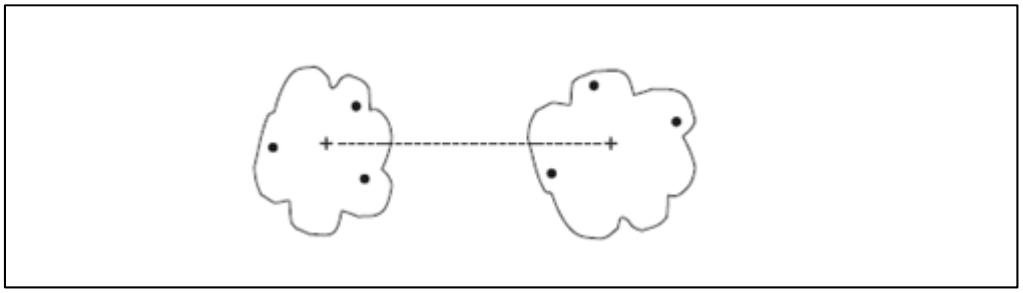

Figura 17 Ilustração da separação de dois clusters em relação a um elemento central (protótipo).

Mais à frente veremos que a coesão e o isolamento inter-*clusters* são parâmetros configuráveis através do sistema, possibilitando a calibração para uma determinada massa de dados.

4.2.7. Métrica de similaridade

Normalmente encontramos na literatura métricas baseadas em estruturas com dimensões fixas como, por exemplo, a *Distância euclidiana*, *Distância de Manhattan* e *Similaridade do Cosseno,* supracitadas. No entanto, na estrutura de dados baseada em *frames* não é possível determinar um número fixo de dimensões. Uma alternativa para utilizar as métricas usuais seria transformar cada *slot* em uma dimensão. Na prática teríamos uma matriz de *frames* por *slots*

e, assim, poderíamos utilizar a similaridade do cosseno entre as linhas desta matriz e determinar a distância entre elas.

De acordo com [25], o critério de similaridade utilizado para a classificação de *frames* é definido a partir de três variáveis (a_1, a_2, a_3), onde:

a) a_1 é o número de atributos compartilhados entre dois *frames*;

b) a_2 é o número de valores presentes em ambos os *frames* e pertencentes ao mesmo atributo;

c) a_3 é a profundidade do nível de uma determinada classe na estrutura.

Entretanto, a componente (c) apresentada nesta métrica não se aplica nesta classificação de dados, pois não existe uma estrutura pré-definida. Sendo assim, utilizamos apenas as variáveis correspondentes às componentes (a) e (b) acima. Com isso, limitamos o número de dimensões dos objetos representados pelos *frames* independentemente do número de atributos que os *frames* possuam, surgindo duas possibilidades de mensurar a distância entre dois objetos:

- Por atributos: consideram-se apenas os atributos para o cálculo da distância entre objetos.
- Por atributos e valores: consideram-se os atributos e seus valores para o cálculo da distância entre objetos.

Para o cálculo da distância entre os objetos utilizando apenas atributos, temos a form. (8):

$$distância(a,b) = \frac{100 \ * \ \text{número de atributos compartilhados entre a e b}}{\max(\text{atributos a}, \text{atributos b})} \qquad (8)$$

onde a e b são *frames*.

Para o cálculo da distância entre os objetos utilizando atributos e valores, utiliza-se a mesma métrica para calcular a distância entre atributos, porém atribuem-se pesos diferentes para os atributos e os valores. A distância

resultante entre os objetos é a média ponderada dos atributos e seus valores. O cálculo da distância por atributos e valores é melhor elucidado através do exemplo: Dado dois objetos distintos A e B que possuem n atributos cada um e valores associados a cada atributo.

- Objeto A: (atributo_x {10}, atributo_y {-12});
- Objeto B: (atributo_x {10,3}, atributo_y {-12}, atributo_z {8}).

O cálculo inicia-se verificando quais atributos são compartilhados entre os dois objetos. Assim, temos que os objetos A e B compartilham os atributos *atributo_x* e *atributo_y*. Agora, para cada atributo compartilhado verificam-se os valores associados a ele que ocorrem em ambos. Ao final de cada atributo avaliado, atribuímos a ele um grau de similaridade, dada pela form. (9).

$$distância(a,b) = \frac{100 \ * \ \text{n. de valores compartilhados entre um atrib. de a e b}}{\max(\text{n. valores do atrib. a, n. valores do atrib. b})} \quad (9)$$

onde a e b são *frames* e o atributo de a e b são iguais.

Armazenam-se todas as distâncias dos valores de cada atributo e realiza-se a média aritmética delas. Assim, obtemos o valor da distância para os valores dos objetos a e b. Agora, é só calcular a distância dos atributos e efetuar a média ponderada comentada anteriormente para obter a similaridade dos objetos. Desta maneira, o exemplo acima teria como cálculo da distância resultante entre os objetos A e B:

Distância entre os atributos:

$$distância(a,b) = \frac{100 \ * \ 2}{\max(2,3)} = \frac{200}{3} = 66{,}6667\%$$

Distância entre os valores dos atributos:

Atributo: atributo_x

$$distância(a,b) = \frac{100 \ * \ 1}{\max(1,2)} = 50\%$$

Atributo: atributo_y

$$distância(a,b) = \frac{100 * 1}{\max(1,1)} = 100\%$$

Assim, a similaridade entre os dois objetos é de 70,83% como mostra o cálculo a seguir.

$$distânciaTotal(a,b) = \frac{66{,}6667 + \frac{50+100}{2}}{2} = 70{,}83\%$$

4.2.8. Classificação de novos elementos

"Comparar instâncias de *frames* com classes de *frames* pode ser considerado a tarefa central do processo de classificação de *frames*. Contudo, ..., ainda temos que lidar com a incompletude da informação disponível em uma dada instância e devemos inventar um critério apropriado para escolher entre classes candidatas" [25].

O critério de proximidade utilizado por [25] e citado na seção 4.2.7, agora pode ser utilizado em sua forma original. Isso se deve a criação de uma taxonomia no processo de agrupamento mostrado anteriormente onde os medóides representam tais classes.

A similaridade é medida entre cada objeto novo a ser classificado e as classes pré-definidas por elementos prototípicos. A comparação é realizada em uma ordem lexicográfica, ou seja, (a1, a2, a3) > (b1, b2, b3), onde (a1, a2, a3) e (b1, b2, b3) são a similaridade de um objeto à classe A e à classe B.

Outra abordagem para a classificação de um novo elemento é a utilização das métricas apresentadas na seção 4.2.7. O objeto é comparado com todos os elementos da taxonomia criada e é incorporada à classe mais próxima. Caso a classe mais próxima não seja suficientemente próxima de acordo com um limite determinado, o objeto é adicionado como um novo medóide.

4.3. Análise dos dados e testes realizados

4.3.1. Testes com dados sintéticos

Algumas massas de dados sintéticos foram criadas para testar o processo de classificação automática. Além disso, uma ferramenta com visualização gráfica chamada *taxonomy creator* foi desenvolvida para melhor compreensão e manuseio dos dados no processo de classificação. Como não é o objetivo deste trabalho descrever a ferramenta, apenas mostraremos imagens ilustrativas com resultados de testes realizados para apresentar a ferramenta criada.

4.3.1.1. Teste 1 – Agrupamento de dados do tipo Person

Utilizaremos a massa de dados sintéticos A, vide anexo A, para mostrar o caso base do algoritmo de classificação. A massa de dados A é composta por *frames* com atributos e valores relativos a uma única classe denominada *Person* que é definida como:

Person [name:, age:].

Os parâmetros utilizados para execução do algoritmo de classificação automática podem ser configurados utilizando a ferramenta *taxonomy creator* conforme ilustra a figura 18.

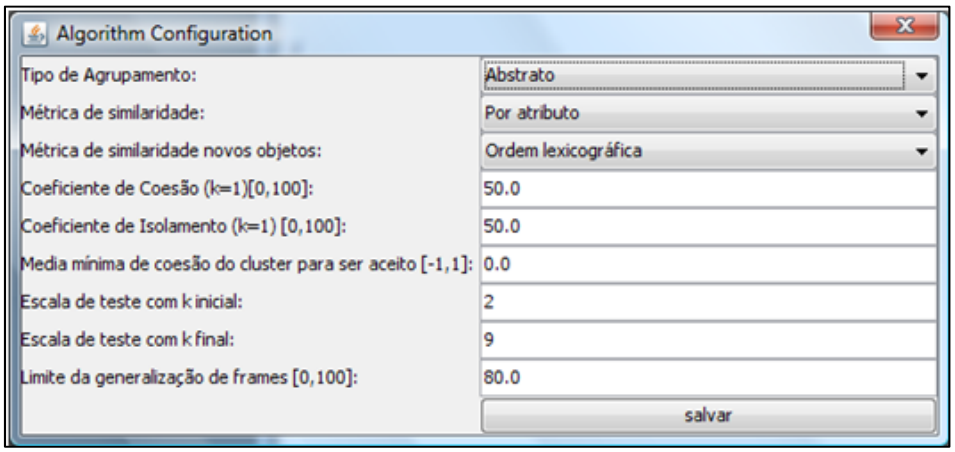

Figura 18 Configuração a ferramenta taxonomy creator.

Algumas das opções configuradas para este teste não serão necessárias, pois o que se espera neste teste é obter apenas um grupo de objetos representados por um único medóide. As variáveis que consideramos relevantes nesta etapa são: métrica de similaridade, média mínima de coesão, coeficiente de coesão, coeficiente de isolamento e escala de teste com k inicial e k final.

A métrica de similaridade é responsável pela métrica utilizada no cálculo da similaridade entre *frames.*

A média mínima de coesão determina o grau mínimo de coesão na validação do número de grupos encontrados. Este campo pode ser preenchido com valores entre o intervalo [-1, 1], onde 1 indica total coesão na média encontrada por todos os grupos do agrupamento resultante e -1 que indica total dispersão dos objetos.

Os coeficientes de coesão e de isolamento são utilizados em conjunto com a variável da média mínima de coesão. Note que o método utilizado na média mínima de coesão (método da silhueta, visto na seção 4.2.2) apenas verifica a média para dois ou mais grupos. Estes coeficientes são utilizados para avaliar a coesão mínima de um grupo e o isolamento máximo entre grupos no caso em que a melhor configuração encontrada pelo algoritmo da silhueta seja igual a dois grupos. Desta maneira, se os coeficientes de coesão e de isolamento não forem satisfeitos, o número de grupos resultantes será igual a um.

A escala de teste com k inicial e k final também é utilizada para determinar o número de grupos encontrados. Submete-se a massa de dados dentro do intervalo determinado pelas variáveis k inicial e k final com o intuito de estimar o número ideal de grupos da massa de dados a ser utilizada.

4.3.1.1.1. Teste 1 – Métrica de similaridade: por atributos

Após a contextualização das variáveis utilizadas para a realização do primeiro teste, vejamos o resultado encontrado para a configuração mostrada na figura 19 somente para objetos do tipo *Person*.

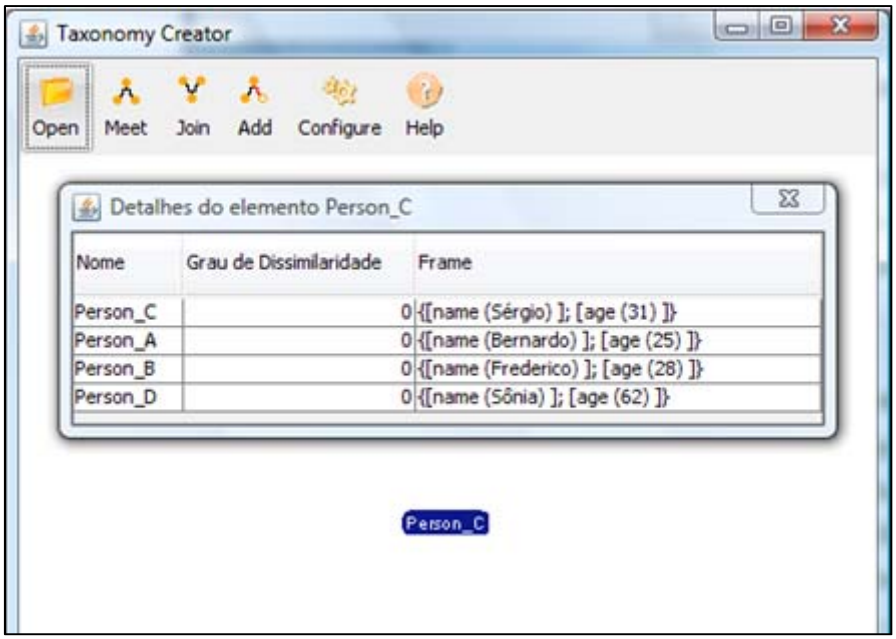

Figura 19 Teste de classificação com objetos do tipo *Person*.

Como previsto, a figura 19 ilustra o resultado obtido mostrando o medóide *Person_C,* representante do grupo, e seus elementos na janela de detalhes. Podemos observar que o grau de dissimilaridade do medóide para os demais objetos é zero, comprovando o resultado esperado para a métrica de similaridade *por atributos*, ou seja, já que todos os *frames* possuem os mesmos atributos e os valores não são contabilizados no cálculo da distância a dissimilariade é zero.

Supondo que o algoritmo tenha inicialmente estimado que o número de grupos seja igual a dois, os coeficientes de coesão e de isolamento devem ser verificados para um possível agrupamento, generalização dos grupos. Tendo *Person_A* e *Person_B* fazendo parte de um grupo 1 fictício e *Person_C* e *Person_D* fazendo parte de um grupo 2 também fictício, podemos verificar que a coesão dos grupos é máxima e que o isolamento é mínimo, o que satisfaz os coeficientes pré-determinados de coesão e isolamento, então o algoritmo estipula que o número de grupos ideal seja um.

4.3.1.1.2. Teste 1 – Métrica de similaridade: por atributos e valores

Utilizando a mesma massa de dados, configuramos a ferramenta para utilizar a métrica de similaridade: *por atributos e valores.* O resultado é mostrado pela tabela de detalhes do elemento *Person_D* encontrado como medóide.

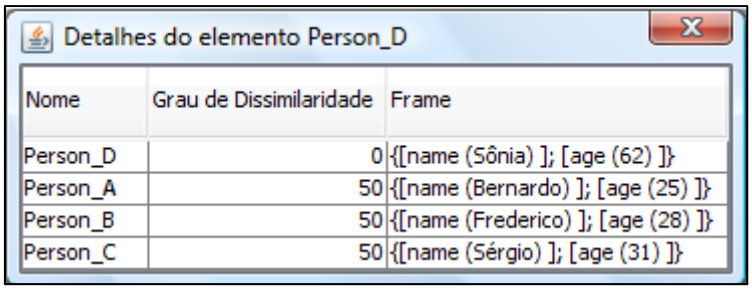

Figura 20 Detalhes do medóide Person_D.

Neste teste também foi encontrado apenas um medóide, Person_D, porém como consideramos os valores relativos aos atributos e como nenhum valor é similar ao valor do medóide escolhido, temos uma dissimilaridade de 50% em relação os objetos e o medóide.

4.3.1.1.3. Análise do teste 1

A grande importância de realizar este teste está na estimativa do número de grupos encontrados. O algoritmo responsável pela estimativa do número de grupos proposto é uma variação do método da silhueta, visto na seção 4.2.2. Este método identifica o grau de coesão média do número de grupos encontrados. Entretanto, não é capaz de identificar se uma massa de dados possui apenas um grupo. Desta maneira, em conjunto com o método da silhueta, utilizamos os coeficientes de coesão e de isolamento para encontrar o número esperado de grupos e como resultado, obtivemos o agrupamento esperado, um grupo de objetos de mesmo tipo em ambos os testes.

4.3.1.2. Teste 2 – Classificação de dados quanto ao tipo de agrupamento

Utilizaremos a massa de dados sintéticos B, vide anexo A, populada com três diferentes tipos de *frames: Pessoa, Empregado e Estudante.*

Person [name:, age:]
Employee [name:, age:, works:, area:, salary:]
Student [name:, age:, level:, area:, fee:]

Submetemos esses dados aos três tipos de agrupamento possíveis no processo de classificação automática (*Abstrato, Híbrido e Medóide)* e as duas

métricas de similaridades disponíveis para a realização do agrupamento (por atributo e por atributo e valor). Vejamos a seguir os resultados obtidos.

4.3.1.2.1. Teste com agrupamento do tipo abstrato

Ao longo desta seção veremos os testes realizados com o agrupamento do tipo *Abstrato* e as duas métricas de similaridades já apresentadas. As configurações atribuídas às variáveis são as que obtiveram os melhores resultados durante a bateria de testes.

4.3.1.2.1.1. Métrica de similaridade: por atributos

As configurações utilizadas para as demais variáveis podem ser vistas na figura 21. Caso haja alguma alteração destes valores, a seção responsável por ela informará os novos valores utilizados.

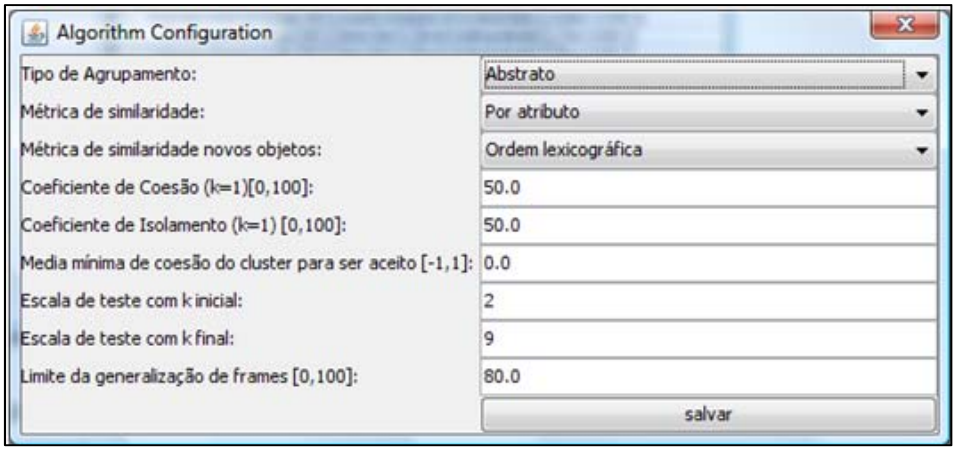

Figura 21 Configurações de ajuste das variáveis para o processo de classificação automática.

O resultado obtido através do processo de classificação automática pode ser visto na figura 22.

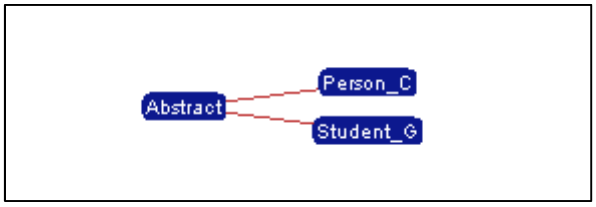

Figura 22 Resultado da classificação para a massa de testes sintéticos.

Note que o processo de classificação encontrou dois níveis de hierarquia e que o nível mais alto da hierarquia corresponde ao medóide abstrato criado a partir do *meet* dos medóides *Person_C* e *Student_G.*

A tabela da figura 23 mostra os detalhes do objeto *Abstract* em relação aos demais objetos onde podemos considerar que a classe criada a partir do *meet* dos medóides representa a classe *Person* e que os grupos criados são coesos. Outra observação válida é que os *frames* do tipo *Employee* foram atribuídos ao grupo dos *frames Student,* vide figura 25, e isso se deve à pequena dissimilaridade entre esses *frames*, ou seja, há apenas dois atributos diferentes entre eles. Desta maneira o algoritmo sugeriu o agrupamento dos elementos do tipo *Employee* com os elementos *Student_G.*

Detalhes do elemento Abstract

Nome	Grau de Dissimilaridade	Frame
Abstract	0	{[name ()]; [age ()]}
Person_C	0	{[name (Sérgio)]; [age (31)]}
Person_A	0	{[name (Bernardo)]; [age (25)]}
Person_B	0	{[name (Frederico)]; [age (28)]}
Person_D	0	{[name (Sônia)]; [age (62)]}
Student_G	60	{[name (Marco)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (1000)]}
Employee_A	60	{[name (Bruna)]; [age (25)]; [works (Company A)]; [area (engineering)]; [salary (1000)]}
Employee_B	60	{[name (Ana)]; [age (32)]; [works (Company A)]; [area (engineering)]; [salary (1500)]}
Employee_C	60	{[name (Jorge)]; [age (32)]; [works (Company A)]; [area (engineering)]; [salary (1000)]}
Employee_D	60	{[name (Luiz)]; [age (41)]; [works (Company B)]; [area (law)]; [salary (100)]}
Employee_E	60	{[name (Gilda)]; [age (27)]; [works (Company C)]; [area (law)]; [salary (10000)]}
Employee_F	60	{[name (Marcos)]; [age (40)]; [works (Company D)]; [area (law)]; [salary (1200)]}
Student_A	60	{[name (Alexia)]; [age (20)]; [area (law)]; [level (undergraduate)]; [fee (1200)]}
Student_B	60	{[name (Diana)]; [age (20)]; [area (law)]; [level (undergraduate)]; [fee (1200)]}
Student_C	60	{[name (Lili)]; [age (24)]; [area (law)]; [level (graduate)]; [fee (1200)]}
Student_D	60	{[name (Marcelo)]; [age (24)]; [area (law)]; [level (graduate)]; [fee (1000)]}
Student_E	60	{[name (Alice)]; [age (22)]; [area (law)]; [level (undergraduate)]; [fee (1000)]}
Student_F	60	{[name (Antonio)]; [age (25)]; [area (law)]; [level (graduate)]; [fee (1000)]}
Student_H	60	{[name (Guilherme)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (1000)]}
Student_I	60	{[name (Joao)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (800)]}
Student_J	60	{[name (Ane)]; [age (22)]; [area (engineering)]; [level (undergraduate)]; [fee (1100)]}
Student_L	60	{[name (Carlos)]; [age (21)]; [area (engineering)]; [level (undergraduate)]; [fee (1100)]}
Student_M	60	{[name (Karin)]; [age (21)]; [area (engineering)]; [level (graduate)]; [fee (1100)]}

Figura 23 Detalhes do medóide abstrato do teste de agrupamento abstrato.

A seguir mostramos os *frames* ou objetos relacionados à *Person_C,* onde podemos perceber que o grupo possui somente objetos do mesmo tipo.

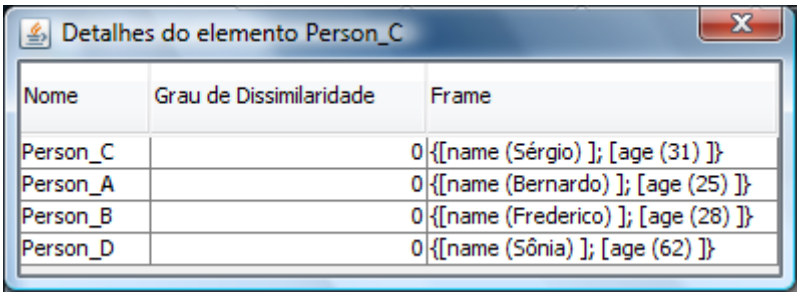

Figura 24 Detalhes do medóide Person_C do teste de agrupamento do tipo abstrato.

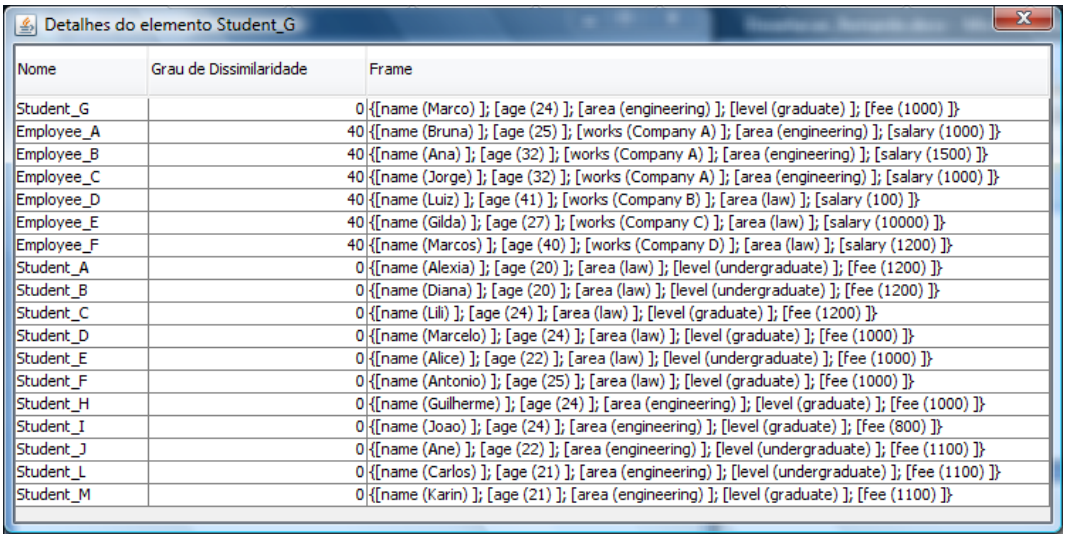

Figura 25 Detalhes do medóide Student_G do teste de agrupamento do tipo abstrato.

4.3.1.2.1.2. Métrica de similaridade: por atributos e valores

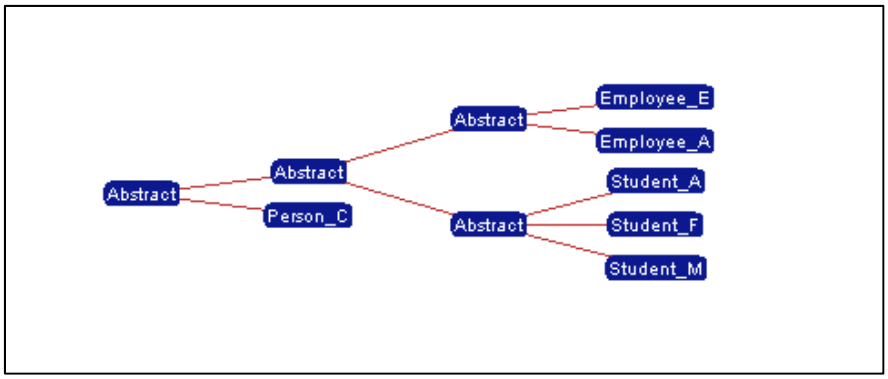

Figura 26 Classificação utilizando a métrica por atributos e valores no agrupamento Abstrato.

Utilizando a métrica de similaridade considerando atributos e valores no cálculo da distância entre os objetos, verificamos que os valores dos atributos têm um grande poder de influência na decisão do algoritmo de classificação automática. Quanto mais informações disponíveis, mais detalhada é a classificação encontrada. Trivialmente vemos que quanto mais informações

tivermos sobre os *frames,* mais fácil será criar grupos de elementos que melhor se encaixam na classificação. Vejamos na tabela 4 o resultado dos objetos classificados.

1	Abstract	{[name ()]; [age ()]}
1.1	Abstract	{[name ()]; [age ()]; [area ()]}
1.1.1	Abstract	{[name ()]; [age ()]; [works ()]; [area ()]; [salary ()]}
1.1.1.1	Employee_E	{[name (Gilda)]; [age (27)]; [works (Company C)]; [area (law)]; [salary (10000)]}
1.1.1.1.1	Employee_D	{[name (Luiz)]; [age (41)]; [works (Company B)]; [area (law)]; [salary (100)]}
1.1.1.1.2	Employee_F	{[name (Marcos)]; [age (40)]; [works (Company D)]; [area (law)]; [salary (1200)]}
1.1.1.2	Employee_A	{[name (Bruna)]; [age (25)]; [works (Company A)]; [area (engineering)]; [salary (1000)]}
1.1.1.2.1	Employee_B	{[name (Ana)]; [age (32)]; [works (Company A)]; [area (engineering)]; [salary (1500)]}
1.1.1.2.2	Employee_C	{[name (Jorge)]; [age (32)]; [works (Company A)]; [area (engineering)]; [salary (1000)]}
1.1.2	Abstract	{[name ()]; [age ()]; [area ()]; [level ()]; [fee ()]}
1.1.2.1	Student_A	{[name (Alexia)]; [age (20)]; [area (law)]; [level (undergraduate)]; [fee (1200)]}
1.1.2.1.1	Student_E	{[name (Alice)]; [age (22)]; [area (law)]; [level (undergraduate)]; [fee (1000)]}
1.1.2.1.2	Student_B	{[name (Diana)]; [age (20)]; [area (law)]; [level (undergraduate)]; [fee (1200)]}
1.1.2.1.3	Student_C	{[name (Lili)]; [age (24)]; [area (law)]; [level (graduate)]; [fee (1200)]}
1.1.2.2	Student_F	{[name (Antonio)]; [age (25)]; [area (law)]; [level (graduate)]; [fee (1000)]}
1.1.2.2.1	Student_H	{[name (Guilherme)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (1000)]}
1.1.2.2.2	Student_G	{[name (Marco)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (1000)]}
1.1.2.2.3	Student_D	{[name (Marcelo)]; [age (24)]; [area (law)]; [level (graduate)]; [fee (1000)]}
1.1.2.3	Student_M	{[name (Karin)]; [age (21)]; [area (engineering)]; [level (graduate)]; [fee (1100)]}
1.1.2.3.1	Student_J	{[name (Ane)]; [age (22)]; [area (engineering)]; [level (undergraduate)]; [fee (1100)]}
1.1.2.3.2	Student_I	{[name (Joao)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (800)]}
1.1.2.3.3	Student_L	{[name (Carlos)]; [age (21)]; [area (engineering)]; [level (undergraduate)]; [fee (1100)]}
1.2	Person_C	{[name (Sérgio)]; [age (31)]}
1.2.1	Person_A	{[name (Bernardo)]; [age (25)]}
1.2.2	Person_B	{[name (Frederico)]; [age (28)]}
1.2.3	Person_D	{[name (Sônia)]; [age (62)]}

Tabela 4 Tabela do resultado da classificação para o agrupamento abstrato.

Analisando os objetos no nível mais especializado da hierarquia de classificação, como o objeto 1.1.1.1 (*Employee_E)* e o objeto 1.1.1.2 (*Employee_A),* verificamos a partir de seus atributos que todos eles devem pertencer à mesma classe, o que de fato é comprovado conforme o elemento indica o 1.1.1 porém, esses elementos estão em grupos diferentes. O motivo da especialização deste grupo decorre dos valores dos atributos considerados no cálculo de similaridade. Desta forma, notamos que os objetos pertencentes ao grupo *Employee_E* possuem a mesma área de atuação (*engineering*), enquanto os objetos do grupo *Employee_A* pertencem à área de atuação (*law*). A mesma

análise pode ser realizada para o grupo de estudantes subindo no nível hierárquico da classificação, ou seja, generalizando os conceitos.

Neste momento chegamos aos objetos criados através do *meet*, visto na seção 4.2.5.1, a partir de outros objetos e que são representados pelos objetos nomeados como *Abstract*. Esses objetos são bem informativos, pois representam o que de fato encontraremos abaixo de sua hierarquia. Assim, o nível 1.1.1 representa a classe de *Employee* e o nível 1.1.2 representa a classe *Student.* Como esses elementos são bem próximos, a generalização deste conceito leva à criação de um novo objeto (1.1) que representa o *meet* das classes Employee e *Student*. Este novo objeto representa a classe *Person,* e como esperado, possui dois filhos: *Person_C* que representa o grupo de objetos do tipo pessoa e o objeto Abstract (1.1) que representa os grupos de pessoas que possuem uma área de atuação.

4.3.1.2.2. Teste com agrupamento do tipo medóide

Ao longo desta seção veremos os testes realizados com o agrupamento do tipo *Medóide* e as duas métricas de similaridades já apresentadas.

4.3.1.2.2.1. Métrica de similaridade: por atributos

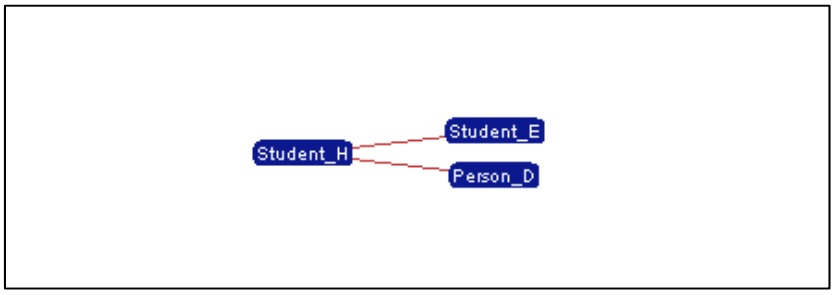

Figura 27 Resultado da classificação utilizando a métrica de similaridade por atributos e o agrupamento por medóide.

Foram utilizados os mesmos valores configurados e a mesma massa de dados sintéticos na seção 4.3.1.2.1.1, porém com o tipo de agrupamento configurado como *Medóide.* Notamos que obtivemos praticamente o mesmo resultado que a classificação automática realizada pelo tipo de agrupamento *Abstrato*, sendo a única exceção a utilização de um objeto existenta na massa

de dados para representar esses dois grupos e não um elemento a partir do *meet* de dois medóides.

A questão aqui seria o porquê de um elemento como *Student_H* ser o elemento representante deste grupo de objetos. A resposta é dada a partir da análise da massa de dados sintéticos utilizada onde verificamos que existem aproximadamente 55% de objetos do tipo estudante, 18% de objetos do tipo pessoa e 27% de objetos do tipo empregado. Desta maneira, o elemento predominante da massa de dados é considerado como o elemento mais representativo, porém este método não nos fornece tanta informação quanto o método *Abstrato.*

4.3.1.2.2.2. Métrica de similaridade: por atributos e valores

Utilizando a métrica de similaridade por atributos e valores, obtemos o resultado mostrado na figura 28.

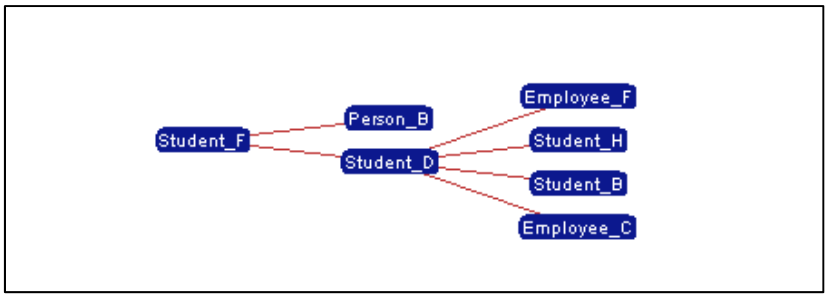

Figura 28 Resultado da classificação utilizando a métrica de similaridade por atributos e valores e o agrupamento por medóide.

Novamente, verificamos que existe uma semelhança entre os agrupamentos em comparação com as análises anteriores. A tabela 5 mostra como os objetos estão classificados.

1	Student_F	{[name (Antonio)]; [age (25)]; [area (law)]; [level (graduate)]; [fee (1000)]}

1.1	Person_B	{[name (Frederico)]; [age (28)]}
1.1.1	Person_A	{[name (Bernardo)]; [age (25)]}
1.1.2	Person_C	{[name (Sérgio)]; [age (31)]}
1.1.3	Person_D	{[name (Sônia)]; [age (62)]}
1.2	Student_D	{[name (Marcelo)]; [age (24)]; [area (law)]; [level (graduate)]; [fee (1000)]}
1.2.1	Employee_F	{[name (Marcos)]; [age (40)]; [works (Company D)]; [area (law)]; [salary (1200)]}
1.2.1.1	Employee_D	{[name (Luiz)]; [age (41)]; [works (Company B)]; [area (law)]; [salary (100)]}
1.2.1.2	Employee_E	{[name (Gilda)]; [age (27)]; [works (Company C)]; [area (law)]; [salary (10000)]}
1.2.2	Student_H	{[name (Guilherme)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (1000)]}
1.2.2.1	Student_G	{[name (Marco)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (1000)]}
1.2.2.2	Student_L	{[name (Carlos)]; [age (21)]; [area (engineering)]; [level (undergraduate)]; [fee (1100)]}
1.2.2.3	Student_I	{[name (Joao)]; [age (24)]; [area (engineering)]; [level (graduate)]; [fee (800)]}
1.2.2.4	Student_J	{[name (Ane)]; [age (22)]; [area (engineering)]; [level (undergraduate)]; [fee (1100)]}
1.2.2.5	Student_M	{[name (Karin)]; [age (21)]; [area (engineering)]; [level (graduate)]; [fee (1100)]}
1.2.2.6	Student_C	{[name (Lili)]; [age (24)]; [area (law)]; [level (graduate)]; [fee (1200)]}
1.2.3	Student_B	{[name (Diana)]; [age (20)]; [area (law)]; [level (undergraduate)]; [fee (1200)]}
1.2.3.1	Student_A	{[name (Alexia)]; [age (20)]; [area (law)]; [level (undergraduate)]; [fee (1200)]}
1.2.3.2	Student_E	{[name (Alice)]; [age (22)]; [area (law)]; [level (undergraduate)]; [fee (1000)]}
1.2.4	Employee_C	{[name (Jorge)]; [age (32)]; [works (Company A)]; [area (engineering)]; [salary (1000)]}
1.2.4.1	Employee_A	{[name (Bruna)]; [age (25)]; [works (Company A)]; [area (engineering)]; [salary (1000)]}
1.2.4.2	Employee_B	{[name (Ana)]; [age (32)]; [works (Company A)]; [area (engineering)]; [salary (1500)]}

Tabela 5 Tabela do resultado da classificação para o agrupamento por medóide.

Analisando os grupos representados pelo medóide *Student_D* (1.2), verificamos que seu grupo está dividido em quatro grupos representados pelos seguintes medóides: *Student_H*, *Student_B*, *Employee_C* e *Employee_F* e para compreender o motivo desta divisão, basta verificar os valores atribuídos aos atributos. Desta forma, notamos que o grupo representado pelo medóide *Student_H* possui os valores do atributo *area* iguais a *engineering* e os objetos representados pelo medóide *Student_B* possuem os valores do atributo área iguais a *law*. O mesmo acontece para os medóides *Employee_C* e *Employee_F*, respectivamente.

Outra vez, verificamos que a métrica de similaridade que considera os atributos e valores traz mais informações para o processo de classificação.

4.3.1.2.3. Teste com agrupamento do tipo híbrido

Ao longo desta seção veremos os testes realizados com o agrupamento do tipo híbrido e as duas métricas de similaridades já apresentadas. Esta

classificação se torna interessante, pois através do algoritmo de classificação criamos as hierarquias a partir do nível básico e especializamos as classes utilizando seus próprios objetos porém, ao invés de utilizarmos os medóides para definir as classes mais gerais utilizamos o conceito de *meet,* criando objetos abstratos.

4.3.1.2.3.1. Métrica de similaridade: por atributos

Para a massa de dados sintéticos utilizados, não é possível notar a diferença entre o método *Híbrido* e o *Abstrato*. Como visto anteriormente, utilizando a massa de dados B e a métrica de similaridade por atributos não foram identificados subgrupos, ou seja, não ocorreram especializações. Assim, com apenas um nível hierárquico, não notamos o funcionamento do agrupamento *Híbrido*. Deixamos para a métrica de similaridade por atributos e valores a explanação deste tipo de agrupamento.

4.3.1.2.3.2. Métrica de similaridade: por atributos e valores

Nesta seção focaremos na classificação resultante e nos medóides encontrados. O agrupamento dos dados através de seus atributos e valores é o mesmo encontrado para os tipos de agrupamento supracitados. O intuito deste teste é mostrar uma abordagem diferente, utilizando os próprios objetos para a especialização dos dados e a criação de objetos no processo de generalização.

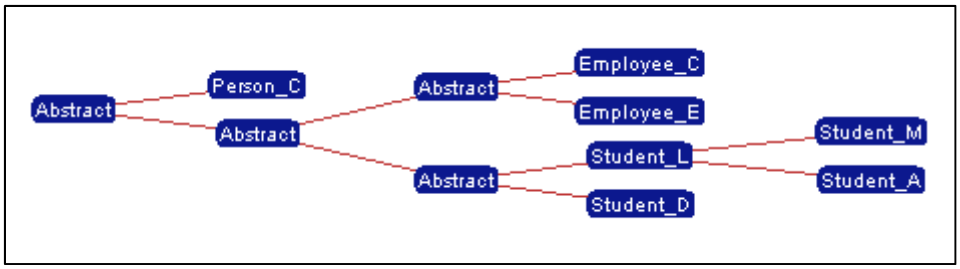

Figura 29 Resultado da classificação utilizando a métrica de similaridade por atributos e valores e o agrupamento híbrido.

Como podemos ver na figura acima, temos uma classificação composta por medóides pertencentes à massa de dados sintéticos e medóides criados através do *meet* de outros objetos.

O processo de especialização pode ser visto no grupo de objetos representado pelo medóide *Student_L*. Se estivéssemos trabalhando com o agrupamento abstrato, o medóide *Student_L* seria o *meet* dos medóides *Student_M* e *Student_A*, assim como no processo de generalização, se estivéssemos utilizando o agrupamento de medóides, os *frames* intitulados por *Abstract* seriam representados por elementos da massa de dados e não por novos elementos.

4.3.1.3. Teste 3 – Classificação de novos objetos

A classificação de novos objetos é tratada através de três diferentes métricas de similaridade: por atributos, por atributos e valores e pela ordem lexicográfica. Os novos objetos são comparados com os objetos já existentes, alocando o novo elemento àquele elemento já classificado que possuir maior similaridade.

Utilizamos a classificação vista na figura 30 para inserir novos objetos utilizando as diferentes métricas de similaridade.

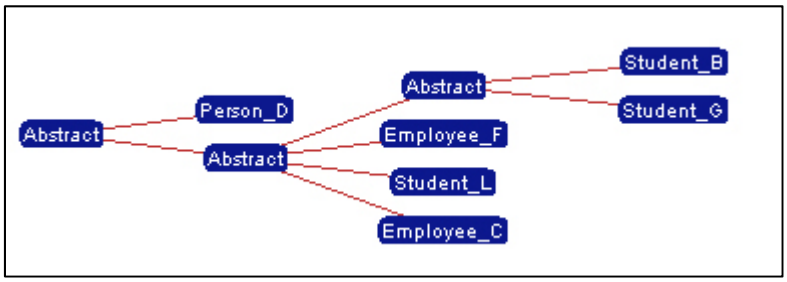

Figura 30 Resultado do processo de classificação.

Na figura 31, obtemos o resultado da inserção de um novo objeto utilizando a métrica *por atributos e valores.* O objeto inserido é do tipo *Person* e possui os seguintes atributos e valores:

Person_E [name:Igor, age:27].

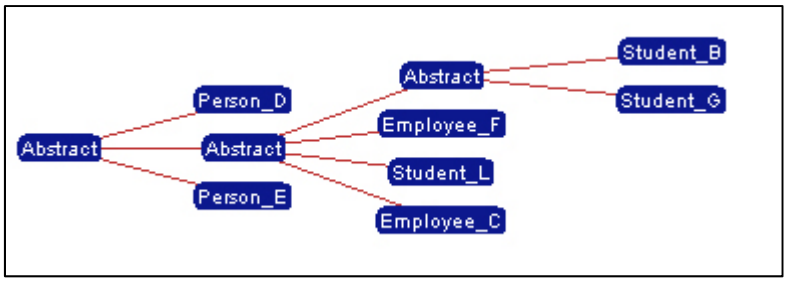

Figura 31 Classificação de novo objeto Person_E.

É trivial identificarmos o motivo pelo qual o objeto *Person_E* foi inserido abaixo do medóide Abstract (raiz), pois ambos os objetos compartilham dos mesmos atributos e embora tenhamos outro grupo representado pelo medóide *Person_D* que possui os mesmos atributos que o novo objeto, nenhum dos objetos possuem valores em comum, possuindo assim o mesmo grau de similaridade. Assim sendo, o primeiro elemento é escolhido.

Note que não precisamos repetir o teste para a métrica de similaridade *por atributos*, pois neste caso teríamos o mesmo resultado. Contudo, perceba que a utilização da métrica *por atributos* é direcionada ao casamento de instâncias com classes de objetos e que a métrica *por atributos e valores* é direcionada ao casamento entre instâncias.

Agora, utilizando a métrica chamada de ordem lexicográfica, verificamos a similaridade a partir de três critérios citados na seção 4.2.8: número de atributos compartilhados, número de valores dos atributos compartilhados e pela altura a qual a classe se encontra. Para testar essa métrica de similaridade utilizamos um *frame* do tipo Employee com os seguintes atributos e valores:

Employee_G [name:Dalia, age:27, works:Company A; area: engineering, salary:1100].

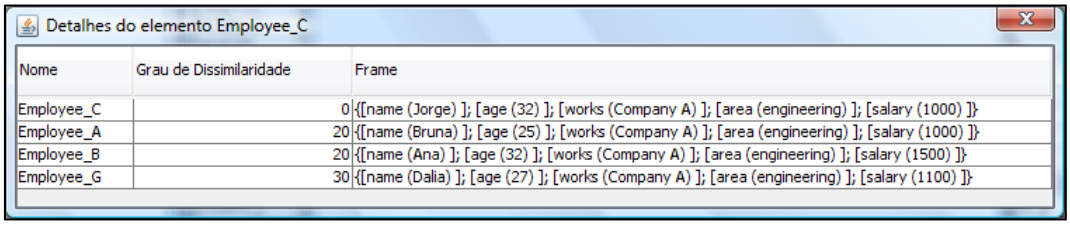

Figura 32 Employee_G inserido a partir da métrica ordem lexiográfica.

Conforme pode ser visto na figura 32 o objeto *Employee_G* foi alocado no grupo de objetos representados pelo *frame* Employee_C. Note que através desta métrica o novo objeto possui os seguintes valores em relação ao medóide:

Atributos compartilhados: 5
Valores de atributos compartilhados: 2
Altura do medóide *Employee_C:* -2

Embora esses valores tenham sido obtidos através do cálculo da distância com o medóide, porque não alocar esse novo medóide abaixo do elemento *Employee_B*? Para responder a essa pergunta vamos calcular a distância entre o novo objeto *Employee_G* e o objeto *Employee_B*.

Atributos compartilhados: 5
Valores de atributos compartilhados: 2
Altura do medóide *Employee_C*:-3

Efetuando o cálculo propriamente dito temos que [5, 2, -2] > [5, 2, -3]. Assim, alocamos o objeto no nível de hierarquia mais alto, já que os objetos compartilham o mesmo número de atributos e valores.

4.3.2. Testes com dados reais

A massa de dados utilizada para realizar os testes no sistema é proveniente do SIAE, Sistema de Apoio a Emergências para catalogação de produtos e serviços relacionados à contingência. A partir da base de dados do SIAE foi possível produzir uma massa de dados de 21.000 *frames* espalhados por 324 classes criadas originalmente.

Para validação da classificação resultante do processo de classificação automática utilizaremos as informações provenientes do SIAE como parâmetro de avaliação dos grupos gerados. Além da classificação original, também utilizaremos um parâmetro para mensurar o nível de casamento dos objetos pertencentes a um grupo.

O nível de casamento é realizado comparando classificação original do medóide e dos objetos deste grupo, por exemplo, dado um medóide com a classificação original Veículos->Terrestres->Carro e um elemento pertencente ao grupo com a classificação original Veículos->Terrestres->Carro, então teremos que este elemento está com um nível de casamento de 100% em relação ao medóide, o que nos indica que este objeto está classificado corretamente. Agora, imaginemos que outro objeto deste mesmo grupo possua a classificação original como Veículos->Terrestres->Carro->Fiat, então continuaremos tendo uma classificação coesa, pois estamos analisando o grupo como um todo e, neste caso, o objeto pertence a mesma classe, porém em um nível mais específico. Neste caso, espera-se que o algoritmo aloque esses objetos em subgrupos que melhor o represente. Outro exemplo seria, o medóide possuir a classificação original como Veículos->Terrestres->Carro->Fiat e o objeto possuir a classificação original Veículos->Terrestres->Carro, neste caso teríamos um nível de casamento de 75% de *matching*, ou seja, três classes de um total de quatro coincidiram.

4.3.2.1. Teste 1 – Análise da amostra de dados do SIAE

Uma amostra de dados de 2100 *frames*, equivalentes a 10% da massa de dados total, foi submetida ao algoritmo de classificação automática utilizando a abordagem de generalização por medóides e a métrica de similaridade por atributos e valores. Além disso, utilizamos os mesmos parâmetros configurados nos testes realizados na massa de dados sintéticos.

Começaremos com o nível mais geral da classificação que pode ser visto na tabela 6, nela encontram-se os representantes dos grupos encontrados. O resultado gerado produziu seis grupos, sendo que apenas o grupo (f) foi subdividido em outros dois grupos e um de seus subgrupos foi novamente subdividido em outros três grupos. Como podemos notar, o grupo subdividido foi o que obteve o menor nível de casamento, desta maneira, esta divisão é realizada, pois o algoritmo entende que o nível de coesão é baixo e é necessário criar subgrupos para melhor alocar esses dados, ver tabela 7 e 8.

Já o resultado dos grupos representados por (b), (c), (d) e (e) obtiveram um nível de casamento de 100%, ou seja, todos os objetos encontrados foram

classificados corretamente de acordo com a classificação original dada pelo SIAE. Além disso, percebemos que a classificação resultante possui apenas um nível hierárquico, reproduzindo exatamente a classificação original dada pelo SIAE.

Medóide	Número de objetos pertencentes a esta classe	Classificação original do medóide	Nível de casamento
(a) Delegacia de Candeias	462	Objeto->Recurso->Entidade_Externa->Órgãos_Governamentais->Órgãos_de_Segurança_Pública->Polícia Civil	58,87%
(b) Analista Ssr. Rr.Hh. Planta Pgsm	292	Objeto->Cargo	100%
(c) Iraci Antônio Davi	285	Objeto->Contato Externo	100%
(d) Lona Leve	58	Objeto->Fabricantes	100%
(e) UN-BS/ATP-N/RES	326	Objeto->Unidade	100%
(f) Equipamento de Higiene Industrial - Explosímetros (codigo:4358)	677	Objeto->Recurso->Recurso_Material->Equipamento_de_Higiene_Industrial->Explosímetros	58,03%

Tabela 6 Resultado do nível básico do agrupamento do teste com amostra dados reais.

Medóide	Número de objetos pertencentes a esta classe	Classificação original do medóide	Nível de casamento
(f.a) nil	654	Objeto->Recurso->Recurso_Material->EPI_-_Equipamento_de_Proteção_Individual->Coletes Salva-Vidas	56,95%
(f.b) Sistema de Armazenamento Temporário - Tanque Flutuante (codigo:5360)	22	Objeto->Recurso->Recurso_Material->Sistema_de_Armazenamento_Temporário->Tanque Flutuante	82,85%

Tabela 7 Subcategorias do medóide (f)

Medóide	Número de objetos pertencentes a esta classe	Classificação original do medóide	Nível de casamento
(f.a.a) Barreiras de Contenção - Águas Calmas - Saia até 30cm (codigo:3516)	169	Objeto->Recurso->Recurso_Material->Barreiras_de_Contenção->Águas Calmas - Saia até 30cm	50,96%
(f.a.b) Equipamentos Diversos – Outros (codigo:4387)	264	Objeto->Recurso->Recurso_Material->Equipamentos_Diversos->Outros	64,28%
(f.a.c) Recursos Médicos para Emergências – Ambulância (codigo:8146)	220	Objeto->Recurso->Recurso_Material->Recursos_Médicos_para_Emergências->Ambulância	61,87%

Tabela 8 Subcategorias do medóide (f.a)

A partir dos resultados obtidos nas tabelas 6, 7 e 8, fizemos uma comparação dos dados para verificar o porquê de níveis de casamento mais baixos nos grupos (a) e (f) e o que identificamos foi uma falha na criação de classes do SIAE. Acreditando que os objetos seriam classificados de acordo com os seus atributos e valores, aplicamos o algoritmo de classificação proposto neste trabalho, porém o resultado nos mostrou que os objetos do SIAE são compostos por atributos herdados das classes superiores e que apenas algumas classes possuem novos atributos. Isso quer dizer que os objetos podem estar em classes diferentes, no entanto, possuem os mesmo atributos. Desta maneira, a classificação dos Objetos->Recursos->... dados pelos grupos (a) e (f) não puderam ser bem definidos, embora tenhamos conseguido extrair algumas informações importantes, como a modelagem intencional ou não-intencional das classes do SIAE.

4.3.2.2. Teste 2 – Análise de dados do SIAE

Figura 33 Comparação da classificação original do SIAE (esquerda) e da classificação gerada automaticamente (direita) pelo processo automático de classificação proposto.

A classificação obtida na fig.33 (direita) é o resultado gerado para a massa de dados com 21.000 *frames* do SIAE. As classes que possuem um X ao lado do nome (classificação à direita) não foram reproduzidas, por exemplo, *objeto* e *recurso*. Isso se deve ao fato descrito na seção 4.1.2.1, onde os objetos pertencentes às classes compartilham dos mesmos atributos, desta maneira a criação de classes se torna desnecessária, mostrando que a classificação original do SIAE poderia suprimir essas classes. Sendo assim, descobrimos que todas as classes derivadas da classe *entidade externa* devem ser mantidas em

apenas um grupo, sem divisões, e as classes *objeto* e *recurso* podem ser retiradas, pois não agregam nenhuma informação a classificação.

Além disso, notamos uma mudança na classificação na classe *recurso material*, onde a classe recurso material foi subdividida em dois grupos, sistema de armazenamento temporário e [ ], onde os parêntesis significam um novo subgrupo criado, que por sua vez criou mais outros 4 grupos e alguns subgrupos. Note que a classe *recurso humano* que estava no segundo nível, agora está no quinto nível da hierarquia, o que se refere à questão dos atributos já que as outras classes que estão a sua volta também possuem atributos compartilhados, como é o caso *recursos médicos para emergência*.

Desta maneira, identificamos a necessidade de verificar como as classes estão sendo criadas no SIAE, além de criar uma classificação mais simples para os dados, pois como dito anteriormente o SIAE possui 324 classes que mostramos ser desnecessárias.

4.4. Conclusões

Este capítulo apresentou a implementação de um sistema de classificação automática de dados baseada numa estrutura de dados chamada *frame* e nas teorias de classificação, nos algoritmos e nas técnicas de agrupamento apresentadas ao longo deste trabalho.

O capítulo resumiu inicialmente os fundamentos utilizados na estratégia de classificação e a noção de *frames*. Em seguida, detalhou a implementação do sistema. Por fim, apresentou vários testes com dados sintéticos e dados reais que permitem avaliar o comportamento do sistema implementado e dos dados que estão sendo submetidos a ele.