UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

SANDRO RAMA FIORINI

S-Chart: Um Arcabouço para Interpretação Visual de Gráficos

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr[a]. Mara Abel
Orientadora

Prof. Dr. Claiton M. dos Santos Scherer
Co-orientador

Porto Alegre, março de 2009.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Fiorini, Sandro Rama

S-Chart: Um Arcabouço para Interpretação Visual de Gráficos / Sandro Rama Fiorini. Porto Alegre: Programa de Pós-Graduação em Computação, 2009.

122 p.:il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação. Porto Alegre, BR – RS, 2009. Orientador: Prof. Dr[a]. Mara Abel; Co-orientador: Prof. Dr. Claiton M. dos Santos Scherer.

1.Visão Computacional. 2.Interpretação Semântica de Gráficos. 3.Representação de Conhecimento Visual. I. Abel, Mara. II. Scherer, Claiton. III. Título.

AGRADECIMENTOS

Vejo os agradecimentos que faço aqui como uma continuação dos agradecimentos que fiz no meu trabalho de conclusão. No entanto, agora tenho um problema mais complicado, pois agradecer a todos que contribuem em um trabalho deste tamanho é quase impossível.

Inicialmente devo agradecer a UFRGS e o Instituto de Informática por mais dois anos onde me propiciaram o melhor dos ambientes para o desenvolvimento do trabalho apresentado nesta dissertação. Agora tenho ainda mais orgulho de estar ligado a essas instituições. Agradeço também ao CNPq, por ter financiado o meu trabalho por esses dois.

Sou grato a todos os colegas e amigos que estiveram comigo ao longo desses dois anos no grupo de Bancos de Dados Inteligentes: Alexandre Lorenzatti, Joel Carbonera, Felipe Victoreti, Carlos Santin, Marília Mello, Laura Mastella, Willian Gonçalves e Bruno Nunes! Todos foram e são muito importantes. Sem toda essa turma certamente este trabalho seria muito mais difícil de ser completado.

Gostaria de agradecer ao meu co-orientador, o Professor Claiton Scherer, do Instituto de Geociências, pelo tempo despendido tentando ensinar um pouco de estratigrafia para um simples mestrando de computação. Sem ele, este trabalho seria impossível.

Fechando os agradecimentos acadêmicos, devo fazer um agradecimento especial a minha orientadora Professora Mara Abel. Já passados mais de seis anos de orientação, devo admitir que os seus conselhos e desafios formaram grande – se não a maior – parte do meu perfil profissional.

Amigos novamente! Posso recortar e colar essa parte dos agradecimentos que escrevi no meu trabalho de conclusão. Ter amizades duradouras é a chave para a manutenção da sanidade em um trabalho de pós-graduação. Enilton, o mais antigo e parceiro para idéias infantis e sem a menor possibilidade de realização ☺. Eduardo Castro, que desde os primórdios do colégio é o meu eterno "colega": no colégio, na faculdade, do laboratório, de trabalho. Grande parceiro para ouvir as minhas idéias caóticas que conduziram finalmente a este trabalho. Ao Grupo dos Cinco, composto pelos meus amigos Bill, Francisco, Gustavo e Leandro. Agora já mais diluído pelas responsabilidades, ainda é a minha principal fonte de risadas e relaxamento intelectual. Obrigado de novo pela companhia e apoio de vocês.

Agradeço também a minha família, especialmente à minha tia Maria e ao meu primo Fabian pelo apoio e pela companhia que não consigo encontrar em outro lugar.

Agradeço aos meus pais: o seu Olir e a dona Marlene. Não vou repetir o que já agradeci antes. Seria "chover no molhado". Sou imensamente grato pelo seu apoio nesses últimos anos, mesmo quando o filho não entendia isso direito.

Por fim, apenas para deixar registrado para o futuro: meu agnosticismo ainda continua.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

OWL	*Ontology Web Language*
SWRL	*Semantic Web Rule Language*
DL	*Description Logics*
MFS	*Maximum Flooding Durface*
MIS	Máquina de Interpretação Semântica
MIV	Máquina de Interpretação Visual
TWC	Transformada *Wavelet* Contínua

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Interpretação semântica de imagens tem se mostrado uma das fronteiras mais promissoras da área de Visão Computacional, especificamente aplicada a interpretação imagens. Nas abordagens que estão sendo propostas atualmente, conhecimento visual explicitamente modelado é utilizado com algoritmos de raciocínio simbólico combinados a algoritmos de processamento de imagem a fim de se extrair o conteúdo de imagens e associá-lo a modelos semanticamente ricos.

Este trabalho apresenta uma abordagem de interpretação semântica de imagens especificamente voltada para interpretação de gráficos de linhas, chamada S-Chart. Ela consiste um conjunto de modelos de conhecimento e algoritmos que podem ser instanciados para interpretação de gráficos em diversos domínios. Os modelos são representados em três níveis semânticos e aplicam o conceito de ancoramento simbólico (*symbol grounding*) para mapear as primitivas entre os níveis. Os algoritmos de interpretação propostos fazem a interação entre o raciocínio simbólico de alto nível e os algoritmos de processamento de sinal para os dados brutos dos gráficos analisados.

Para demonstrar a aplicabilidade do *framework* S-Chart, foi desenvolvido o sistema InteliStrata, uma aplicação no domínio da Geologia, voltada para interpretação semântica de gráficos de perfis de poço. Utilizando a aplicação, foram interpretados dois perfis de raios gama capturados em poços de exploração, de modo que o sistema identificasse a presença de Sequências Estratigráficas e superfícies de inundação máximas. Os resultados foram comparados com a interpretação de um geólogo especialista sobre os mesmos dados. O sistema aponta as mesmas sequências já identificadas e oferece outras opções de interpretação compatíveis com as do geólogo utilizando os mesmos dados.

O framework S-Chart tem seus pontos fortes nos seus modelos representação de conhecimento visual independentes de domínio, que permitem a utilização do mesmo arcabouço em diferentes aplicações e, em especial, no seu modelo de ancoramento simbólico entre primitivas de representação.

Palavras-Chave: Interpretação Semântica de Gráficos, Conhecimento Visual, Modelos de Conhecimento, Ancoramento Simbólico, Visão Computacional

S-Chart: A Framework for Visual Interpretation of Line Charts

ABSTRACT

Semantic image interpretation is one of the most promising frontiers in the Computer Vision area, specifically when applied to Image Interpretation. To reach semantic interpretation, visual knowledge explicitly represented is applied by symbolic reasoning algorithms combined with image processing algorithms in order to extract the content of the images and associate it with semantically rich models.

This work describes the S-Chart approach, a semantic image interpretation approach designed for interpretation of line charts. It is structured as a set of knowledge models and algorithms that can be instantiated to accomplish chart interpretation in other domains. The models are represented in three semantic levels and apply the concept of symbol grounding in order to map the primitives between the levels. The interpretation algorithms carry out the interaction between the symbolic reasoning in the high level, and the signal processing algorithms in the low level data.

In order to demonstrate the applicability of the S-Chart framework, we developed the InteliStrata system, an application in Geology for the semantic interpretation of well log profiles. Using the application, we have interpreted the graphs of two gamma-ray profiles captured in exploration wells, to indicate the position of Stratigraphic Sequences and the maximum flooding surfaces. The results were compared with the interpretation of an experienced geologist using the same data input. The system was able to point the same identified sequences and offered alternative interpretation that were compatible with the geologist interpretation over the data.

The S-Chart framework demonstrates its effectiveness on interpretation of pictorial information in knowledge intensive domains. The stronger points of the approach are its domain independent models for visual knowledge representation and, specially, the application of a symbol grounding model to provide a correlation between representation primitives.

Keywords: Semantic Chart Interpretation, Visual Knowledge, Knowledge Models, Symbol Grounding, Computer Vision.

1 INTRODUÇÃO

Este trabalho descreve um estudo na área de visão computacional. Mais especificamente, sobre representação de conhecimento visual e interpretação semântica de informações pictoriais.

Por muitos anos, um dos principais desafios de pesquisa na computação foi a busca por meios eficientes para armazenagem de grandes quantidade de dados, textuais e não textuais. Tendo atingido um patamar razoável de desenvolvimento nesta área, o foco passou a ser maneiras eficientes de extrair informações úteis dessa imensa quantidade de dados, armazenadas durante anos em bases distribuídas pelo mundo. Em se tratando de dados textuais, é significante o grau de desenvolvimento de ferramentas para extração de conteúdo, como ferramentas de busca e descoberta de conhecimento. Da mesma forma, com o avanço nas capacidades de armazenamento, as mesmas bases de dados estão cada vez mais repletas de dados não-textuais, como imagem, som e vídeo, cujo conteúdo também deve ser extraído e interpretado. No entanto, no caso de dados pictoriais, foco deste trabalho, ainda são relativamente raros os resultados de pesquisas na extração e interpretação do seu conteúdo.

Grande parte dos trabalhos na área de extração e interpretação de conteúdo de imagens que focam em métodos puramente numéricos e estocásticos, como em Gomez & Dactu (2007) e Fan et al. (2005), sem dar a devida atenção aos aspectos relacionado a modelagem semântica do conteúdo. Enquanto tais métodos apresentam resultados positivos em domínios visualmente menos complexos, elas falham em domínios onde o acesso ao conteúdo das imagens requer uma grande quantidade de conhecimento visual. Estes são denominados *domínios imagísticos* (YIP; ZHAO, 1996), como é o caso da medicina (OGIELA; TADEUSIEWICZ, 2003), geologia (SANTIN, 2008), vigilância (WITHAGEN, 2006) e etc.

Perceber porque os métodos de interpretação citados falham em domínios imagísticos é relativamente simples. Embora o exato mecanismo de funcionamento da visão humana (e de outros processos sensórios) ainda seja disputado pela ciência (BARSALOU, 1999), é possível entendê-la como um processo que envolve dois componentes: um componente sensório e um cognitivo. O componente sensório se encarrega do processamento básico da visão (como detecção de regiões, bordas, objetos em movimento e etc.). O componente cognitivo aplica conhecimento visual a respeito do domínio onde a cena visualizada se encontra e, com base nele, interpreta as informações sensórias em símbolos que repre-

sentam o seu conteúdo. O fluxo de informações entre esses componentes é circular. Enquanto o componente sensório processa e repassa informações visuais básicas para o componente cognitivo, este guia o processamento do componente sensório influindo na seleção das informações que serão processadas. Usualmente, estes dois componentes não existem nas abordagens clássicas. As técnicas puramente baseadas em processamento de imagens podem ser comparadas ao componente sensório. Elas produzem interpretações baseadas somente nos dados brutos das imagens, sem utilizar – ao menos explicitamente – conhecimento visual sobre a imagem, como contexto, relações espaciais e etc. Esse processo é especialmente importante em domínios imagísticos, onde a interpretação de dados visuais (imagens, cenas e etc.) é fortemente baseada em conhecimento visual sobre o domínio. Por exemplo, devido ao treinamento e experiências, médicos conseguem visualizar estruturas em imagens de raios-x que são inacessíveis a um leigo no assunto. Isso demonstra o quão necessário é haver uma integração entre o processamento de baixo nível e mecanismos de simbólicos de alto nível em sistemas de interpretação de imagens. No entanto, os tipos de sistemas citados anteriormente (numéricos e estocásticos) falham em atingir este objetivo, pois não implementam completamente componentes mínimos da visão.

Evidentemente, as abordagens clássicas ainda continuam sendo aprimoradas. Porém, atualmente é possível encontrar sistemas de visão computacional que buscam incorporar os *dois* componentes da visão. Estes sistemas são referenciados como sistemas de *interpretação semântica de imagens*, pois, além do processamento numérico, aplicam conhecimento visual para extrair o conteúdo de imagens e representá-lo em modelos semanticamente ricos. Eles buscam integrar modelos de conhecimento visual com algoritmos de processamento de imagens, a fim de implementar um mecanismo de interpretação automático inspirado na visão humana. Estes modelos de conhecimento visual descrevem as entidades existentes no domínio das imagens, bem como as suas feições visuais e inter-relações espaciais. Como será apresentado nos próximos capítulos, sistemas de interpretação semântica são mais eficientes para interpretação em domínios imagísticos.

1.1 Modelos de conhecimento visual em três níveis

Existem diversas formas de se representar conhecimento visual. Contudo grande parte das abordagens de interpretação semântica utiliza *modelos de representação em três níveis semânticos*. Cada nível semântico representa em um grau diferente de abstração as entidades contidas em uma imagem.

O nível semântico mais alto é o *nível semântico* propriamente dito. Ele representa as entidades do domínio da imagem. Usualmente, os modelos nesse nível modelam o conhecimento de domínio independe de aspectos visuais.

O nível mais baixo de abstração é representado pelo *nível analógico*. Ele captura informações de baixo nível de uma imagem, geralmente informações essencialmente quantitativas. O conjunto de primitivas desse nível é formado por

pixel, conjuntos de pixels, códigos de cores e outros índices numéricos. Em geral, esse nível faz a integração entre o domínio dos algoritmos de interpretação de imagem e o domínio simbólico.

É fácil perceber a distância semântica entre esses dois níveis opostos. Há uma grande diferença de propósito entre informações sobre, por exemplo, *pixel* e entidades do domínio. Para diminuir este "degrau" semântico, existe o *nível visual* intermediário. Ele expressa o conteúdo da imagem em termos de primitivas visuais genéricas e compartilhadas, como formas geométricas, propriedades visuais e relações espaciais. Por exemplo, na Figura 1.1 a mesma entidade é expressa como um morango no nível semântico, uma região de pixels no nível analógico e como uma forma de coração com cor vermelha e textura granulada no nível visual. Assim, em um processo de interpretação simbólico dessa imagem, o algoritmo não interpreta entidades do domínio diretamente dos dados analógicos da imagem, ele antes interpreta os dados analógicos em instâncias de primitivas visuais e, a partir destas, infere as entidades de domínio.

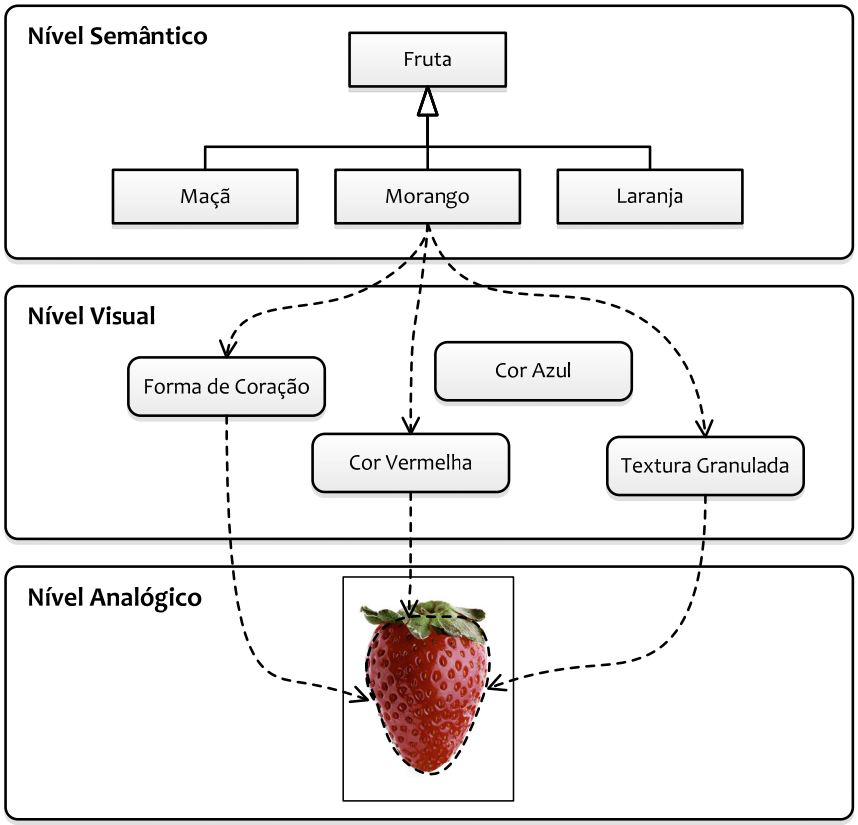

Figura 1.1: Exemplo de representação em três níveis semânticos.

Outro conceito relacionado aos modelos de conhecimento visual utilizado neste trabalho é o problema do *ancoramento simbólico* (tradução livre do inglês *symbol grounding problem*). Ele está relacionado ao problema da interpretação de um sistema de símbolos, que de acordo com Harnad (1990, 1994), está puramente a cargo do interpretador. Ou seja, o símbolo por si só não reflete a sua interpretação. Para que essa interpretação seja inerente ao sistema, os símbolos devem ser "ancorados" em um substrato sensório-motor. Como, no exemplo da

figura Figura 1.1, o símbolo "morango" só tem real significado se associado com as informações sensórias básicas (pixels, regiões de pixels e etc.) representadas pela imagem no nível analógico. Essa questão está diretamente relacionada à interpretação semântica de imagens.

Este trabalho propõe um *framework* para interpretação semântica, que integra os conceitos de modelos em três níveis semânticos e ancoramento simbólico. Este framework é voltado para um tipo específico de dados pictoriais, apresentado a seguir.

1.2 Interpretação semântica de gráficos

As tarefas realizadas por especialistas humanos freqüentemente incluem a organização e interpretação de todo tipo de dados colhidos do ambiente. Esses dados podem variar consideravelmente na sua forma e apresentação. Em alguns casos, o especialista que realiza a interpretação busca maneiras de projetar esses dados complexos de alguma forma que evidencie feições importantes. Por exemplo, dado um economista que busca identificar tendências de variação de

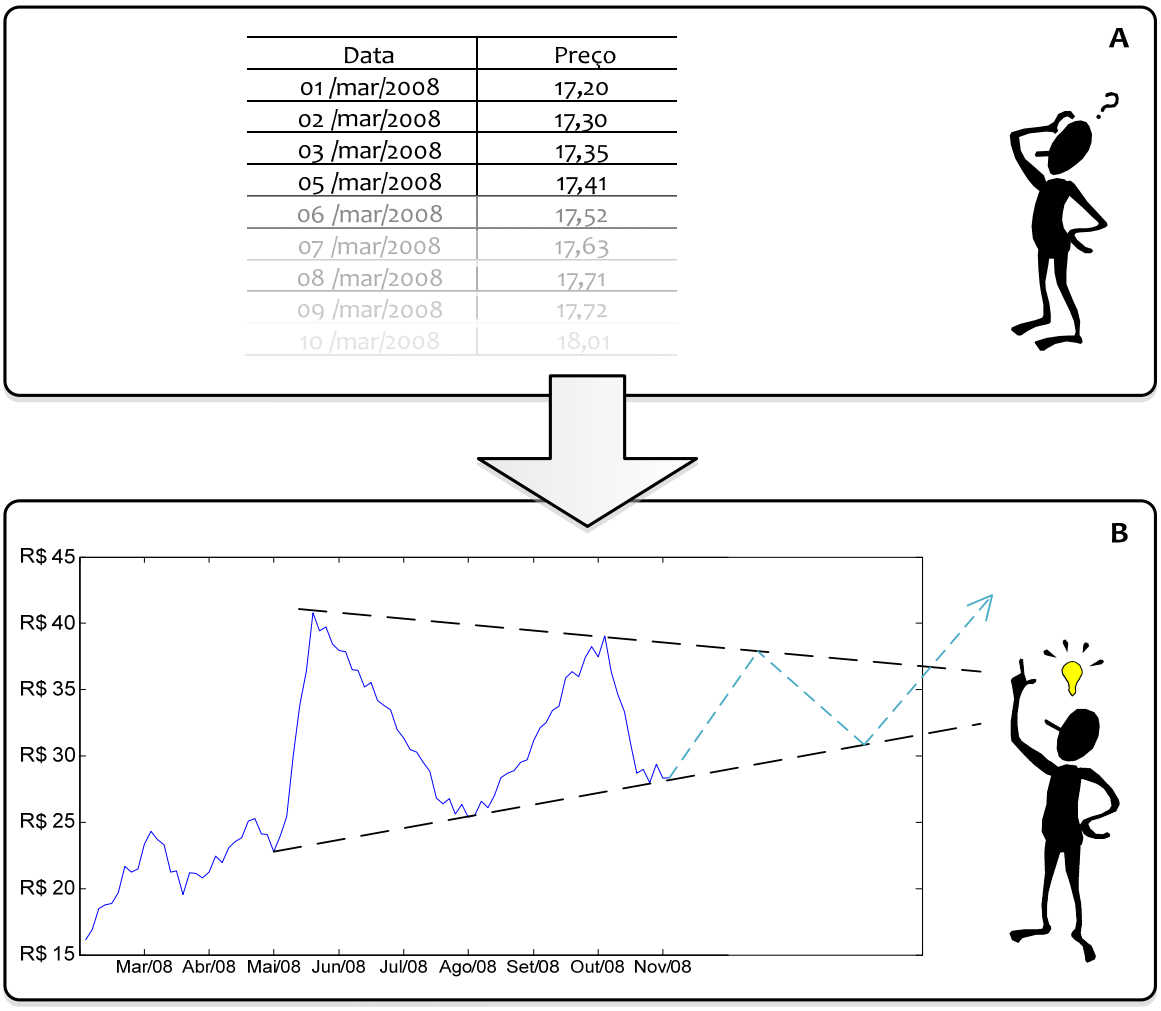

Figura 1.2: Representação gráfica de dados numéricos.

preços de um dado ativo financeiro. Analisando somente os dados brutos da série temporal desse ativo (ex.: uma tabela simples de preços indexada pelo tempo), o economista dificilmente perceberá que tipo de tendência ou padrão de comportamento que eles apresentam (Figura 1.2a). A projeção dos dados em um gráfico (ex.: *tempo × preço*) converte a sua variação numérica em um padrão visual, mais fácil de ser reconhecido e interpretado. Além disso, em domínios complexos, o processo de interpretação visual desse tipo de dado é fortemente calcado em conhecimento de domínio. Ou seja, padrões visuais apresentados nos gráficos são reconhecidos e relacionados com entidades do domínio onde se dá a interpretação. Este trabalho propõe uma forma de capturar este comportamento de interpretação visual realizado por um especialista ao analisar um gráfico como o exposto na Figura 1.2b. Mais especificamente, será demonstrado como algoritmos de processamento numérico destas séries de dados podem ser combinados com algoritmos e modelos de raciocínio simbólico. A proposta apresentada utiliza os princípios de interpretação semântica, de forma a combinar mecanismos de extração de padrões com conhecimento de domínio, através de um modelo de *representação em três níveis semânticos*.

A necessidade de uma abordagem de interpretação semântica específica para gráficos como o da Figura 1.2b pode não parecer tão importante em um primeiro momento. Entretanto, ela fica evidente quando se compara a natureza dos dados que compõem um gráfico. As abordagens de interpretação semântica de imagens usuais focam especificamente na interpretação de imagens naturais[1], digitalizadas por câmeras e outros dispositivos de captura. Nesse caso, os dados gerados são representados em uma matriz de duas dimensões, com a informação de cor em cada posição da matriz (Figura 1.3a). Os algoritmos de processamento de baixo nível (segmentação, agrupamento e etc.) geralmente analisam esses dados como se este fosse um sinal de duas dimensões espaciais. Ou seja, uns conjuntos de valores indexados por duas dimensões contínuas. Já no caso dos gráficos considerados aqui, a série de dados que carrega a informação relevante tem essencialmente uma dimensão contínua (Figura 1.3b). Essa série pode ser analisada como um sinal de uma dimensão. Dessa forma, existe uma incompatibilidade essencial entre a natureza dos dados processados pelas abordagens clássicas de interpretação semântica de imagens e os dados que são interpretados como gráficos. A implicação disso é que novas primitivas devem ser criadas a fim de que seja possível extrair e representar informações de baixo nível de sinais de uma dimensão contínua. De forma semelhante, a interpretação visual de gráficos requer primitivas visuais específicas, que capturem feições como tipos de curvas e suas propriedades. Este trabalho propõe novas primitivas analógicas e visuais para dar suporte a interpretação semântica de

[1] O termo *imagem natural* se refere aqui a representações que capturam cenas de uma realidade física. Fotos ou mesmo pinturas são exemplos de imagens naturais.

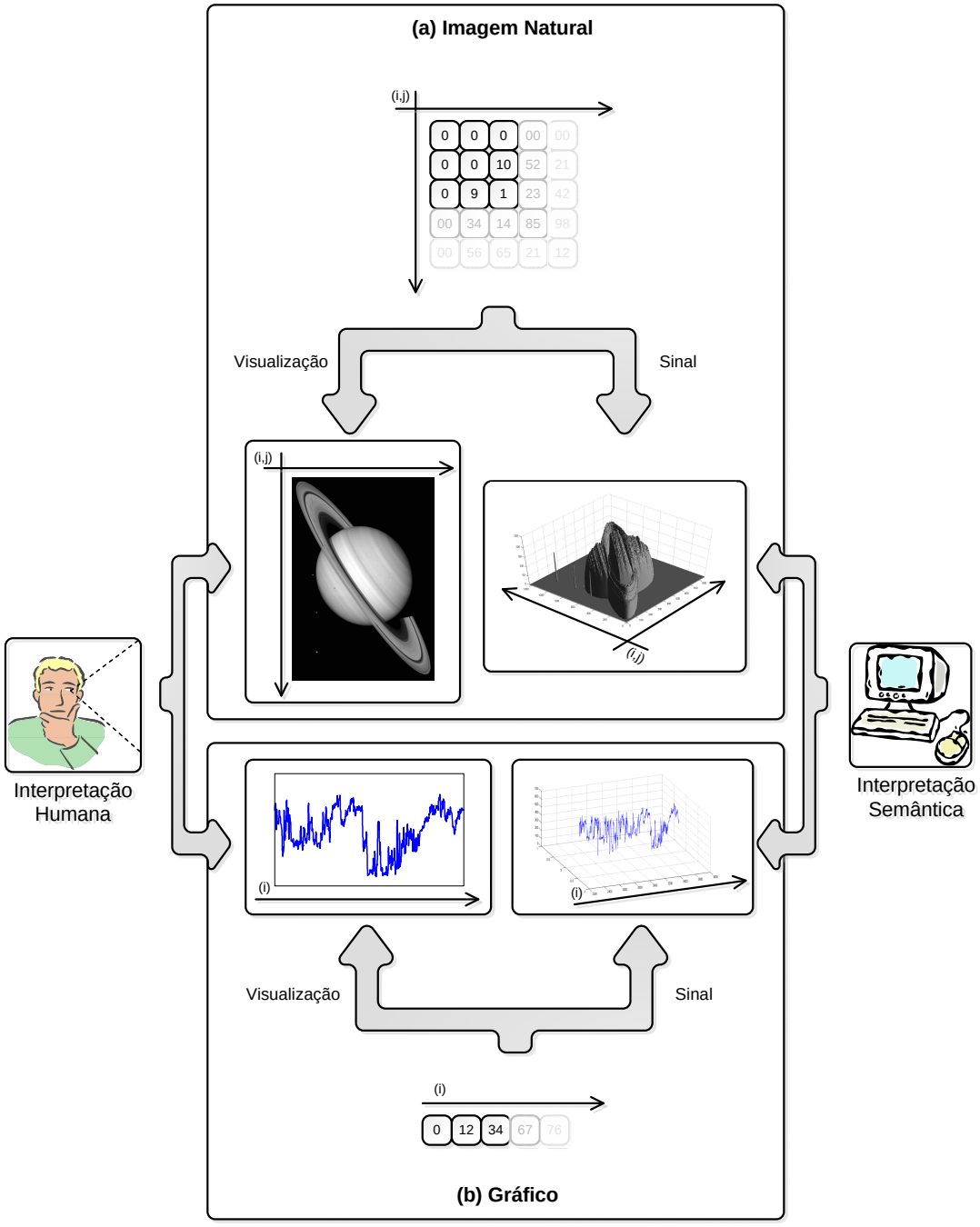

Figura 1.3: Diferenças entre interpretação de imagens naturais e gráficos.

gráficos, em conjunto com alguns algoritmos para processamento e interpretação desse tipo de dados.

Deste ponto em diante, o termo “gráfico” se refere a projeções gráficas com linhas como a da Figura 1.2b e o termo “sinal” ao conjunto de dados brutos que as geram.

1.3 Objetivo

O objetivo principal deste trabalho é propor um framework para interpretação semântica de gráficos, chamado S-Chart,. Ele apresenta os seguintes componentes:

- Um modelo de representação de conhecimento visual para suporte a interpretação de gráficos. Este modelo é composto por três modelos representando os níveis semânticos e um modelo de mapeamento entre estes níveis. Cada nível apresenta primitivas específicas para garantir a representação de entidades da imagem em diferentes níveis de abstração. O modelo de mapeamento apresenta primitivas para modelagem da ligação entre os níveis;
- Algoritmos de interpretação simbólicos para operacionalizar a interpretação semântica com base nos modelos. Eles coordenam os algoritmos de processamento de imagem e os integram no processo de raciocínio simbólico.

1.4 Domínio de aplicação: sistema InteliStrata

Este trabalho apresenta também o sistema InteliStrata, uma implementação do framework S-Chart voltada para interpretação de gráficos no domínio da Geologia, mas especificamente na Estratigrafia de Sequências. Este sistema demonstra como os conceitos apresentados no framework S-Chart podem ser implementados em um sistema real de interpretação.

O sistema InteliStrata representa um dos componentes de um projeto em desenvolvimento de um sistema de conhecimento para interpretação estratigráfica de bacias sedimentares a partir de dados de poço e afloramento. O sistema se estrutura a partir de uma ontologia de domínio de estruturas sedimentares e um conjunto de ferramentas de visualização e integração de dados.

A tarefa realizada pelo sistema InteliStrata é a interpretação estratigráfica no contexto da Estratigrafia de Sequencias. A Estratigrafia de Sequencias é uma área de Geologia que estuda a relação entre as variações históricas do nível do mar na Terra e a deposição de sedimentos. A tarefa de interpretação consiste em analisar os dados sobre os estratos de sedimentos que compõem uma bacia de deposição em uma dada região e, a partir das suas características e padrões de empilhamento, inferir os ciclos (“sequências”) de variações do nível no mar na região. Essa informação é bastante importante para a atividade prospecção de reservatórios de petróleo.

A análise dos estratos é feita de duas formas. Diretamente, com base em amostras de rocha extraídas de poços de prospecção. Ou indiretamente, com base em medidas feitas em poços de onde não se extraem testemunhos (visto que este é um processo caro). Uma dessas medidas é o perfil de raios gama, uma medida de radioatividade da rocha feita ao longo do poço. O resultado é um gráfico semelhante a um eletro-encefalograma. Com base na análise visual des-

se gráfico, o geólogo tenta inferir feições geológicas relativas às variações do nível do mar, como sequências e parassequências. A interpretação considera que progressão (avanço) ou regressão (retrocesso) do nível do mar gera registros com desenhos característicos nos perfis de raios gama. Esse padrão visual é reconhecido pelo geólogo quando realiza a interpretação.

A tarefa de interpretação do sistema InteliStrata é, com base em um perfil de raios gama, sugerir interpretações de sequências e parassequências. Uma sequência sedimentar é uma sucessão de estratos geneticamente relacionados (formados na mesma época pelos mesmos processos sedimentares) cujos limites são definidos em resposta a uma queda relativa do nível do mar. Enquanto parassequências são os blocos de construção das sequências, limitados por oscilações menores do nível do mar (DELLA FÁVERA, 2001; VAN WAGONER et al., 1990).

Um objetivo secundário do sistema InteliStrata é demonstrar a implementação do framework S-Chart com a utilização de formalismos e ferramentas *off the shelf*. Por exemplo, para representação dos modelos de conhecimento visual que compões os níveis semânticos são utilizadas linguagens padrão da Web Semântica. Isso permite que ferramentas padronizadas sejam utilizadas na implementação do sistema.

1.5 Organização dos capítulos

O capítulo 2 aprofunda a discussão dos conceitos relacionados à interpretação semântica de imagens, já descritos anteriormente. Ela também faz uma análise de alguns frameworks para interpretação já existentes na literatura e os compara.

O capítulo 3 apresenta e detalha o framework S-Chart. A sua arquitetura de modelos de conhecimento visual é apresentada, bem como os modelos de mapeamento entre níveis semânticos. Ao fim do capítulo são apresentados e detalhados os algoritmos de interpretação simbólica que operacionalizam a interpretação semântica.

O capítulo 4 apresenta uma breve discussão sobre a Estratigrafia de Sequências, domínio de aplicação deste trabalho.

O capítulo 5 descreve o sistema InteliStrata. É demonstrado como o framework S-Chart pode ser implementado em um sistema real para interpretação de expressões gráficas de perfis de poço na estratigrafia de sequências. Aspectos de sistema não abordados pelo framework são desenvolvidos nesse capítulo. Ao final do capítulo são analisados dois casos de interpretação de perfil utilizando o sistema e confrontando seus resultados com a interpretação dos mesmos perfis pelo especialista no domínio.

No capítulo 6 são apresentadas as conclusões e possíveis trabalhos futuros para extensão dos conceitos apresentados.

2 INTERPRETAÇÃO SEMÂNTICA DE IMAGENS

O principal desafio dos sistemas de visão computacional é interpretar corretamente o conteúdo de imagens. Um sistema desse tipo deve ter como saída uma descrição, formal ou não, dos objetos presentes em uma imagem, bem como a sua classificação e suas propriedades visuais. Por exemplo, um sistema de visão computacional para interpretação de imagens de satélite pode ter como saída quais artefatos de uma dada imagem são casas, quais são carros e quais são estradas, além da maneira como esses objetos estão relacionados topologicamente.

Até hoje foram pesquisadas diversas formas de solução para o problema de interpretação de imagens. A primeira geração de sistemas de interpretação buscava solucionar o problema de interpretação através da aplicação simples de algoritmos processamento de imagens para reconhecimento de formas geométricas. No entanto, essas abordagens se mostraram ineficazes ao tentar replicar o comportamento humano na interpretação em domínios com tarefas de grande complexidade visual, como geologia e medicina. Nesses domínios, especialistas humanos utilizam uma grande quantidade de conhecimento de domínio – geralmente implícito – para realizar a interpretação. Para uma pessoa leiga, os artefatos presentes em uma imagem de Raios-X podem não fazer sentido algum, enquanto que para um médico, esses artefatos representam órgãos internos e possíveis doenças. O médico *sabe* quais as feições visuais que um órgão ou uma doença apresentam em uma imagem de Raio-X. Ele utiliza esse conhecimento visual para interpretar os artefatos da imagem em informações significativas. Abordagens puramente baseadas em processamento de imagens não conseguem capturar esse comportamento de interpretação, pois carecem de uma representação explícita do conhecimento visual do domínio onde estão sendo aplicados, que possa guiar a solução do problema (MAILLOT, 2005).

A partir da década de 80, começou o surgimento dos primeiros trabalhos unindo conceitos de representação de conhecimento visual com técnicas de processamento de imagens (CREVIER; LEPAGE, 1997), dando origem aos primeiros sistemas de *interpretação semântica de imagens*, como os sistemas SIGMA e ACRONYM (MATSUYAMA, 1987). Esses sistemas buscavam imitar o comportamento de interpretação de imagens de um especialista em domínios visualmente complexos.

Um sistema de interpretação semântica integra algoritmos de processamento de imagem com mecanismos de raciocínio simbólico, com o intuito de capturar

o conteúdo de uma imagem através um modelo simbólico semanticamente rico. Enquanto os algoritmos de processamento extraem as feições visuais básicas da imagem (bordas, regiões, cores, texturas e etc.), os mecanismos de raciocínio simbólico combinam e interpretam tais feições de acordo com um modelo que represente conhecimento visual do domínio onde a imagem se insere. Esse modelo define as entidades que podem estar presentes na imagem e os seus inter-relacionamentos. O resultado da interpretação é um modelo simbólico, semanticamente rico, do conteúdo da imagem. Uma vez capturado, esse conteúdo pode ser disponibilizado para consulta ou outros tipos de inferências de mais alto nível.

O principal componente do processo de interpretação semântica é o modelo de conhecimento visual que lhe dá suporte. Esse modelo explicita as entidades do domínio de discurso das imagens, bem como as suas inter-relações espaciais, mereológicas, taxonômicas e outras relações pertinentes ao domínio, visuais ou não. Por exemplo, um modelo de conhecimento visual para interpretação de imagens de satélite certamente conterá os conceitos como `Casa`, `Estrada` e `Automóvel`, em conjunto com relações espaciais como `aoLadoDe`, `sobre` e `próximoDe`. Modelos de conhecimento visual como este dão algumas vantagens aos sistemas de interpretação. Em primeiro lugar, eles explicitam o conhecimento visual utilizado no sistema, facilitando a sua depuração e manutenção. Eles também ajudam a melhorar o resultado do processo de interpretação como um todo, uma vez que podem guiar os algoritmos de processamento de baixo nível. No mesmo exemplo, se uma região da imagem foi identificada como uma instância de `Estrada`, e se o modelo especifica que casas estão sempre próximas de estradas, então o mecanismo de interpretação pode buscar por feições visuais de casas próximas a região classificada como uma estrada.

Em geral, os modelos de conhecimento tentam proporcionar alguma forma de *ancoramento simbólico* (HARNAD, 1990, 1994) das entidades do domínio. Sistemas que utilizam esse conceito partem do princípio que cada símbolo do domínio é reconhecido no ambiente dado algum padrão básico de impulsos nas suas entradas analógicas. Ou seja, as entidades de alto nível semântico são "ancoradas" nas feições de baixo nível que indicam a sua detecção. Assim, uma vez que essas feições são reconhecidas nas entradas, o sistema pode assinalar a detecção da entidade representada pelo símbolo. No caso de sistemas de interpretação de imagens, as entidades que devem ser reconhecidas das imagens são mapeadas nas feições visuais que possibilitam a sua detecção, como uma região de pixels com alguma característica específica.

O ancoramento simbólico é importante para a interpretação semântica, pois permite a interpretação seja feita em dois sentidos. No sentido *top-down*, as entidades de alto nível indicam aos algoritmos de processamento quais são as feições visuais específicas que estão esperando. Os algoritmos buscam especificamente por essas feições e indicam a sua presença ou não. Caso positivo, a entidade visual pode ser indicada como reconhecida. No sentido *bottom-up*, em primeiro lugar os algoritmos de processamento extraem todas as feições possí-

veis da imagem. Após isso, as feições extraídas são analisadas em busca das feições esperadas por cada uma das entidades do domínio. Nos casos onde há correspondência, as entidades são marcadas como reconhecidas. No entanto, em muitas das abordagens de interpretação semântica são utilizados os dois sentidos de interpretação, iterativamente. Por exemplo, uma dada imagem de satélite é analisada em um programa de processamento de imagens (Figura 2.1a), resultando na extração de duas regiões de pixels. A aplicação inicial de uma interpretação *bottom-up* interpreta os dois segmentados como os conceitos `Estrada` e `Rio` em um modelo simbólico (Figura 2.1b). Esse mesmo modelo define que na superposição de entidades do tipo `Estrada` e `Rio` pode haver uma entidade do tipo `Ponte`. Então o mecanismo de interpretação, em uma interpretação no sentido *top-down*, ajusta o foco dos algoritmos de processamento de imagem para buscar uma entidade com as características visuais da entidade `Ponte` no cruzamento das duas regiões segmentadas anteriormente (Figura 2.1c). Se estas características visuais forem encontradas, então a interpretação de `Ponte` é obtida.

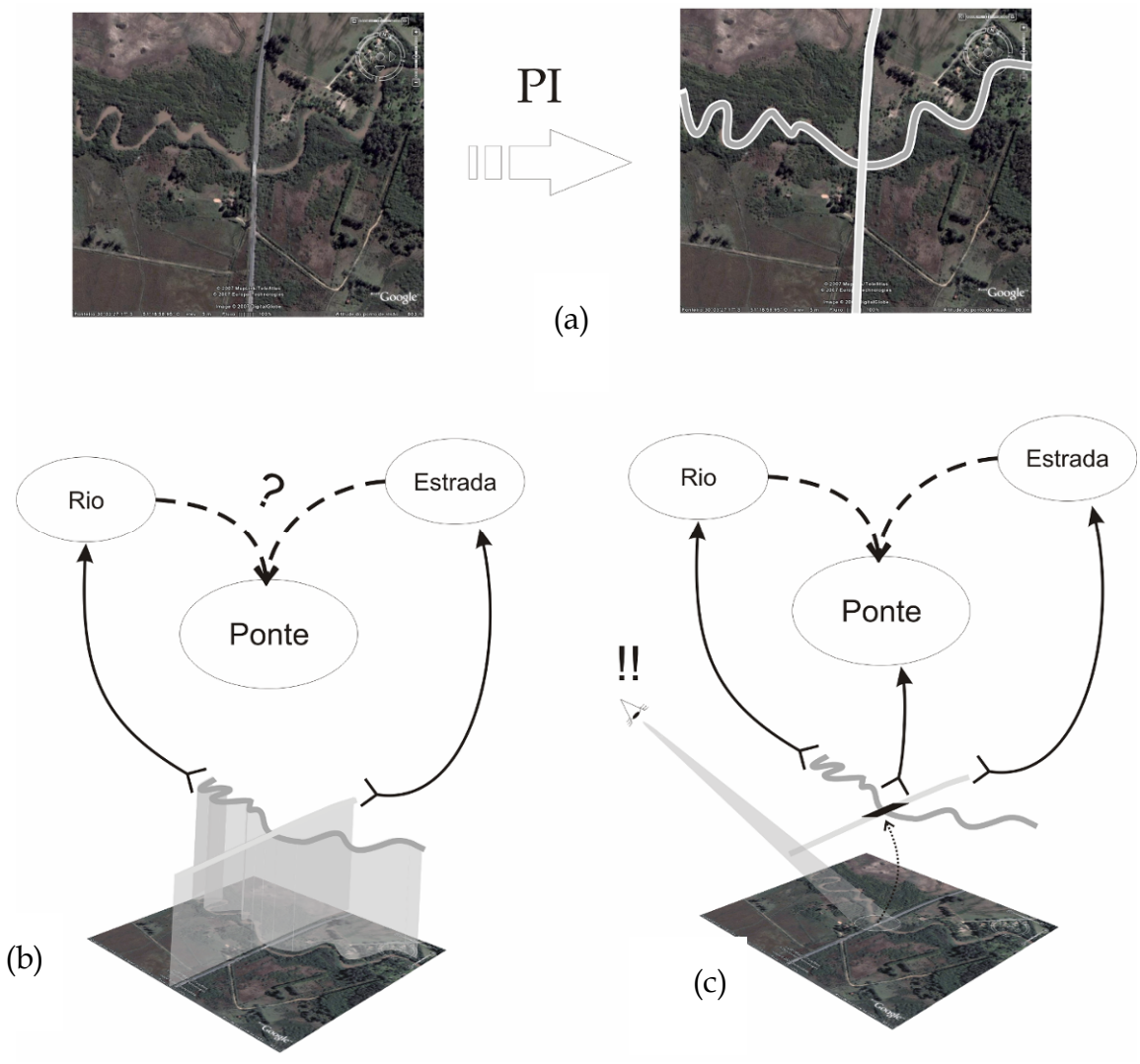

Figura 2.1: Exemplo de interpretação por ancoramento simbólico.

Geralmente, uma vez que o símbolo é reconhecido em uma imagem, o algoritmo realiza um mapeamento explícito entre ele e as feições visuais que lhe deram origem. Isso provê uma maneira do mecanismo de interpretação – ou o seu usuário – revisitarem as informações que deram origem a uma dada interpretação.

Por fim, a interpretação semântica de imagens permite associar significado aos dados visuais analógicos, através de modelos conceituais que representam o conhecimento contido nas imagens analisadas. Esses modelos conceituais dão suporte aos algoritmos de processamento de imagens e possibilitam a utilização do conhecimento visual capturado na solução de problemas e em mecanismos de inferência para suporte a tomada de decisão.

2.1 Representação de Conhecimento Visual

Uma vez que o conhecimento visual tem um papel fundamental na interpretação semântica de imagens, diversos trabalhos focam em encontrar maneiras de incorporá-lo em sistemas de visão computacional de forma efetiva. Provavelmente, o maior desafio, do ponto de vista de sistema, é representá-lo de forma explícita em algum tipo de estrutura com alto nível semântico e que dê suporte a todo o processo de raciocínio visual.

Como apontado por Hudelot e colegas (2005), a análise de sistemas de visão computacional baseados em conhecimento visual (VISIONS[2], SIGMA[3], PROGAL (OSSOLA et al., 1996), MESSIE[4] e etc.) induz a conclusão que os seus modelos de representação são estruturados em pelo menos três níveis de abstração (Figura 1.1): um baixo *nível analógico*, um alto *nível semântico* e um *nível visual* intermediário. Cada nível busca tratar os aspectos de representação de um tipo específico de informação.

O *nível semântico*, mais alto na escala de abstração, representa as entidades do domínio que podem ser reconhecidas na imagem. O *nível analógico* engloba as informações ligadas diretamente aos *pixels* que compõem a imagem, como bordas, valores de cor e regiões. Em geral, informações no nível analógico são obtidas diretamente por algoritmos de processamento de imagem. O mapeamento intermediário entre os dois é representado pelo *nível visual*. Ele é composto por primitivas visuais genéricas, como formas geométricas e relações topológicas

As entidades de cada nível são mapeadas para uma ou mais entidades do nível inferior. Primitivas do nível semântico são definidas em termos de primi-

---

[2] (HANSON; RISEMAN, 1978).

[3] (MATSUYAMA, 1987).

[4] (SANDAKLY; GIRAUDON, 1994).

tivas no nível visual e essas, por sua vez, são definidas em termos de primitivas no nível analógico. De fato, a existência de um modelo intermediário especificamente construído para definição de aspectos visuais de entidades do domínio traz diversas vantagens a sistemas de interpretação. A primeira é a diminuição do grande lacuna semântica (*semantic gap*) entre nível analógico e o nível visual, a fim de flexibilizar o processo de interpretação e torná-lo mais imune a ruídos advindos do processamento inicial nos dados analógicos (CHELLA et al., 1997). Ele também melhora a comunicação do sistema com o usuário, pois o ser humano utiliza informações no nível visual para representar feições visuais e se comunicar. Em especial, "especialistas em diferentes domínios freqüentemente usam e trocam um vocabulário visual genérico para descrever conceitos semânticos nos seus domínios" (HUDELOT et al., 2005). Por fim, o nível visual permite uma maior independência entre a representação e a aplicação, o que proporciona maior reusabilidade dos modelos gerados.

A seguir são apresentados alguns trabalhos encontrados na literatura que utilizam os conceitos apresentados até aqui.

2.2 Abordagens de interpretação semântica

Em geral, os projetos de sistemas de interpretação semântica de imagens apresentam as suas propostas de representação de conhecimento visual dentro de um *framework* completo, incluindo mecanismos de inferência visual (ancoramento semântico, raciocínio simbólico e etc.) e algoritmos de processamento de imagens.

Nas seções seguintes, serão descritas as quatro abordagens de representação de conhecimento visual que melhor se encaixam na visão de representação em três níveis semânticos. Em cada uma será dada uma visão geral das características da abordagem e uma descrição mais detalhada de cada nível de representação. Ao fim, as abordagens são confrontadas.

2.2.1 Abordagem 1: Agregados

A abordagem apresentada em Neumann & Möller (2008) foca na utilização de lógica de descrição (DL) para representação de conhecimento e sistema de raciocínio pra interpretação de cenas. Inicialmente criada para monitoração de tráfego, a abordagem também inclui primitivas para representação de objetos ao longo do tempo.

Os construtos básicos de modelagem são os *agregados* (do inglês *aggregates*), que representam conceitos presentes nas imagens através da agregação das suas partes, satisfazendo restrições espaço-temporais. Os agregados são mapeados para a representação de baixo nível (*pixels*) através da GSD (*Geometrical Scene Description*). GSDs são "descrições quantitativas de propriedades visuais de cenas que variam no tempo" (JARKE et al., 1989) no nível visual. Os autores ainda apontam que os serviços de inferência comuns de lógica descritiva (verificação

de consistências, classificação e etc.) podem ser utilizados em modelos semânticos de imagens representados com esse paradigma.

2.2.1.1 *Nível Semântico*

As principais entidades dessa abordagem são os *agregados*, utilizados para representar situações e objetos no modelo conceitual. Um agregado é basicamente um conceito definido pela descrição das partes que o compõe. Ele ainda contém um conjunto de restrições espaço-temporais que definem as condições e o contexto de ocorrência do agregado (e as suas partes) na imagem. Uma parte também pode ser um agregado, o que induz uma estrutura mereológica dos conceitos. Além disso, os agregados podem constituir uma hierarquia taxonômica.

Nessa representação, os objetos mais básicos de uma cena são agregados especiais, compostos por: um objeto físico ou um "*corpo*" no mundo 3D; uma "*visão*" que constitui a evidência visual do objeto no campo de visão da câmera. Através dessas duas propriedades, as entidades de alto nível são mapeadas para uma representação no nível visual.

Os autores defendem que o nível semântico pode ser representado com uma lógica de descrição $ALCF^{(D)}$, pois ela é o suficiente para atender os requisitos de representação dos agregados. Além disso, a representação em DL permite que se utilizem ferramentas de inferência da DL (como RACER[5] ou FaCT++[6]) para checagem de consistência e inferência sobre os modelos visuais.

2.2.1.2 *Nível Visual*

A abordagem não especifica claramente como é constituído o nível visual de representação. Segundo apresentado Neumann e seus colegas, a representação intermediária se dá pela GSD (*Geometrical Scene Description*). Já segundo (HERZOG, 1995), a GSD tem o objetivo de representar a cena original sem nenhuma perda de informação. Ela descreve, para cada *frame* de uma cena, dados de posição e orientação dos objetos, bem como seu modelo tridimensional.

A construção de uma descrição GSD pode ser automática ou manual. Por exemplo, o sistema VITRA descrito por Herzog, consegue reconstruir automaticamente modelos tridimensionais de carros, mas o cenário de fundo deve ser construído manualmente.

2.2.1.3 *Nível analógico*

Embora citado, nenhum dos trabalhos estudados descrevem exatamente como se dá o mapeamento do GSD para o *pixel*. Sabe-se apenas que os dados

---

[5] http://www.racer-systems.com/

[6] http://owl.man.ac.uk/factplusplus/

brutos são mapeados por algoritmos de processamento de imagens que resultam em descrições GSD.

2.2.2 Abordagem 2: Abdução

Inicialmente proposta por Murray Shanahan e colegas em (2004; 1996) para aplicação em robótica, a abordagem *abdutiva* consiste em um modelo lógico de percepção visual. Ele é caracterizado da seguinte forma: seja Σ uma teoria que define as relações entre os objetos de uma cena e as informações de baixo nível que lhes dão origem. Então, dada uma conjunção Γ de fórmulas representando uma coleção de informações imagísticas de baixo nível, o papel da percepção visual é encontrar uma conjunção de fórmulas Δ tal que,

$$\Sigma \wedge \Delta \models \Gamma.$$

Em outras palavras, a abordagem define, dentro de um contexto lógico, o ancoramento semântico onde o modelo conceitual Σ é mapeado para os dados sensoriais Γ através do mapeamento definido em Δ. A tarefa de um sistema de percepção visual é encontrar o conjunto Δ que melhor explica os dados sensoriais em Γ.

O modelo de inferência descrita por Shanahan e colegas utiliza as abordagens *top-down* e *bottom-up*. O algoritmo infere diversos possíveis conjuntos de hipóteses de explicação Δ e escolhe, com base em um cálculo de probabilidade, qual a hipótese que melhor explica a imagem.

2.2.2.1 Nível Semântico

Embora Shanahan não defina explicitamente um nível semântico, Cohn e colegas (2003) sugerem, dentro do mesmo paradigma, que o modelo conceitual Σ deve estar definido como uma hierarquia mereológica, como apresentado no exemplo da Figura 2.2.

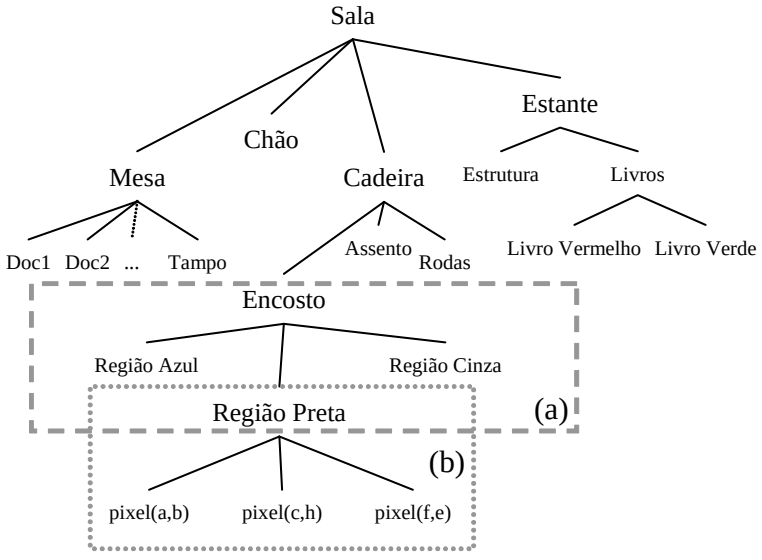

Figura 2.2: Exemplo de hierarquia de composição (adaptado de (COHN, ANTHONY G. et al., 2003)).

2.2.2.2 *Nível Visual*

O nível Visual é definido como um conjunto de axiomas lógicos de primeira ordem contidos no conjunto Σ. Esses axiomas relacionam primitivas de baixo nível, como retas e pontos, com primitivas do nível visual, como regiões e volumes. Além disso, uma peculiaridade da abordagem é a possibilidade de mapear ruído identificado pelos algoritmos de baixo nível.

Por exemplo, dada uma primitiva de baixo nível *Line(w,p_1,p_2)*, que identifica uma reta w entre os pontos p_1 e p_2. Um conjunto de axiomas válidos no nível visual seria (copiado do original):

$$\exists\, p_1, p_2\, [Line(w, p_1, p_2)] \leftarrow \exists\, r\, [\mathrm{Re}\, gion(r) \wedge SideOf\,(w, r)]$$
$$[SideOf\,(w_1, r) \wedge SideOf\,(w_2, r) \wedge w_1 \neq w_2] \leftarrow [Parallel(w_1, w_2) \vee Joins(w_1, w_2)]$$
$$\exists\, p_1, p_2\, [Line(w, p_1, p_2)] \leftarrow Noise(w)$$

O primeiro axioma explica a presença de uma linha visível pela presença de uma região visível, enquanto a segunda linha define uma região com quatro lados. A abordagem abdutiva também permite a modelagem de ruído no nível intermediário. Assim, instâncias de primitivas do baixo nível, como *Line*, que não são mapeadas por nenhum axioma no nível visual podem ser interpretadas como ruído.

Cohn e colegas também descrevem o mapeamento para o nível semântico como uma Hierarquia Observacional, como no mapeamento do objeto `Encosto` na figura Figura 2.2a. O mapeamento para o nível analógico é descrito por uma Hierarquia Sensorial, como no exemplo da Figura 2.2b, onde uma entidade visual e descrita em termos de *pixels* por *pixels* em determinadas coordenadas do espaço.

2.2.2.3 *Nível analógico*

O nível analógico da abordagem abdutiva é composto por primitivas simples como ponto ou linha, extraídos da imagem por algoritmos de processamento de imagem. O nível analógico raramente conterá o dado bruto em si.

Figura 2.3: Passos para extração das linhas como primitivas de baixo nível de uma imagem (SHANAHAN, MURRAY; RANDELL, DAVID, 2004).

Por exemplo, em Shanahan e colegas (2004), imagens capturadas por um robô são processadas por um algoritmo de detecção de bordas baseado no operador Sobel (Figura 2.3). Então um algoritmo seguidor de bordas é utilizado para extrair as instâncias de primitivas de baixo nível, como:

$$Line(1,[238,157],[241,147])$$
$$Line(2,[240,159],[247,157])$$

A partir dessas primitivas é possível inferir regiões e conceitos de maior nível semântico nos níveis superiores, como descrito na seção 2.2.2.2.

2.2.3 Abordagem 3: Espaços Conceituais

A *Teoria dos Espaços Conceituais*, apresentadas por Peter Gärdenfors (2004), apresenta o conceito de *espaço conceitual*, um espaço métrico onde entidades são caracterizadas por um número de dimensões qualitativas (como cor, coordenadas espaciais, tamanho, etc.). Essa teoria pode ser aplicada em várias áreas de Inteligência Artificial, sendo que Chella e colegas (1997) propuseram a sua utilização na área de visão computacional, dentro do contexto da robótica. Nessa abordagem, o espaço conceitual faz o papel de ponte entre a representação de baixo nível e a representação simbólica.

Chella e colegas introduzem o conceito de *knoxel*, um ponto genérico no espaço conceitual, correspondendo a uma entidade primitiva no nível visual (como uma região ou volume). Na abordagem, os *knoxels* são volumes obtidos da imagem por algoritmos de processamento e mapeados para o nível semântico (CHELLA et al., 2001). Por exemplo, um martelo "genérico" pode ser representado por dois *knoxels* no nível visual, como na Figura 2.4a. O conjunto de *knoxels* $\{k_1, k_2\}$ é então mapeado para o conceito *Hammer* no nível semântico como representado na Figura 2.4b.

A correlação entre os *knoxels* e as entidades do nível semântico é realizada através de um algoritmo de atenção, utilizando geração e teste de hipóteses.

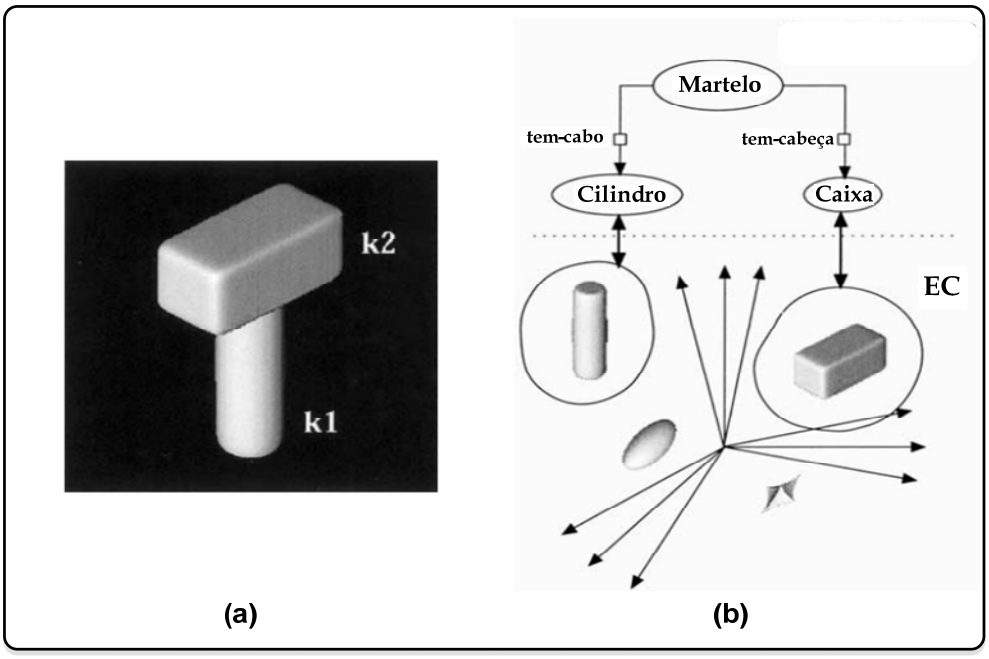

Figura 2.4: Representação de um martelo em um espaço superquadrático e conceitual (CHELLA et al., 2001).

2.2.3.1 *Nível Semântico*

O nível semântico descreve a cena percebida em termo de uma linguagem lógica de alto nível. É adotado um formalismo de representação híbrido, constituído por um componente terminológico e um componente assertivo. O primeiro contém o modelo conceitual propriamente dito, enquanto o segundo guarda a descrição de uma cena individual (instâncias).

O componente terminológico é baseado em uma taxonomia de objetos, além de induzir relações de composição entre os objetos. Por exemplo, a Figura 2.5 demonstra a especialização da entidade genérica *Object* - e sua relação *has-part* - na a entidade *Hammer*. Nesse caso, conforme a Figura 2.4b, as entidades primitivas *Box-shaped* e *Cylinder-chaped* são representações diretas de *knoxels* do nível visual.

Já o componente assertivo representa as instâncias de uma cena utilizando lógica de primeira ordem. Por exemplo, a declaração da instância *Hammer#1* do conceito *Hammer* é dada pela fórmula

$$Hammer(Hammer\#1\ ).$$

Para expressar que a relação *has-handle* entre *Hammer#1* e o *knoxel Cylinder-shaped*#1, declara-se a fórmula

$$has\text{-}handle(Hammer\#1, Cylinder-shaped\#1).$$

Além disso, o componente de terminológico contém representações para relações espaciais e topológicas entre objetos.

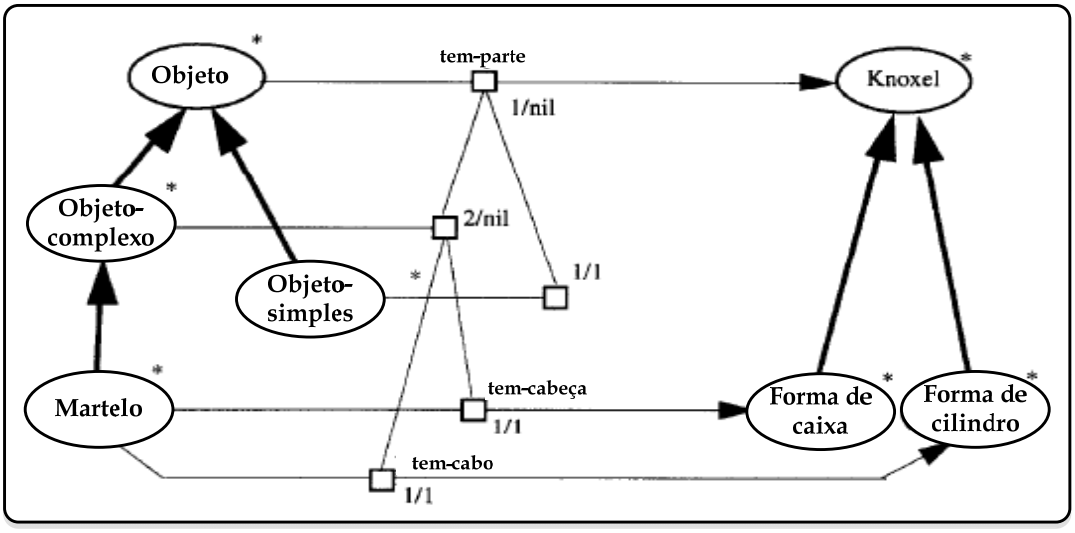

Figura 2.5: Fragmento do componente terminológico da abordagem, utilizando a representação gráfica de KL-ONE (CHELLA et al., 1997).

2.2.3.2 *Nível Visual*

Como dito, a primitiva de representação do nível visual é uma forma 3D primitiva chamada *knoxel*. Nos trabalhos de Chella e colegas, as primitivas para representação do *knoxel* são as superquadráticas.

Superquadráticas são formas geométricas derivadas de uma equação quadrática paramétrica com as funções trigonométricas elevadas a dois expoentes reais. Em resumo, um *knoxel* superquadrático é representado por um vetor no espaço $\mathbb{R}^{11}$, ou seja, um *knoxel* **k** é dado por

$$\boldsymbol{k} = \left[a_x\ a_y\ a_z\ \varepsilon_1\ \varepsilon_2\ p_x\ p_y\ p_z\ \varphi\ \upsilon\ \omega\right]$$

onde $\left[a_x\ a_y\ a_z\ \varepsilon_1\ \varepsilon_2\right]$ definem a forma da superquadrática e $\left[p_x\ p_y\ p_z\ \varphi\ \upsilon\ \omega\right]$ definem a posição especial e inclinação. Além disso, os *knoxels* podem ser agrupados em *clusters* de percepção (PC). Por exemplo, a Figura 2.4a sugere um PC dado por k_1 e k_2.

A relação entre os *knoxels* e as entidades no nível superior é representada através da função de interpretação

$$\Phi^{\mathrm{C}}: \mathrm{C} \rightarrow \mathrm{PC}$$

que mapeia o conjunto C de instâncias no componente assertivo para o conjunto de todos os *clusters* de percepção no nível visual. Por exemplo, no exemplo da figura Figura 2.4:

$$\Phi(Hammer\#1) = \{k_1, k_2\}$$

ou ainda,

$$\Phi(\textit{Cylinder-shaped}\#1) = \{k_2\},$$

$$\Phi(\textit{Cylinder-shaped}\#1) \subset \Phi(Hammer\#1).$$

A função Φ é inferida por um algoritmo baseado em atenção que utiliza o conhecimento contido no nível semântico em um esquema de inferência *top-down*. O algoritmo busca por possíveis objetos através dos *knoxels* já identificados. Ainda no exemplo da Figura 2.4, se o *knoxel* k_2 já foi identificado, o algoritmo irá supor a existencial potencial de um martelo e irá realizar uma buscar no nível visual pelo *knoxel* k_1. Se k_2 sugerir a existência de outros objetos, será feita uma busca por evidências que comprovem a existência de todos eles.

2.2.3.3 Nível analógico

O nível analógico, na abordagem dos espaços conceituais, é constituído somente dos dados brutos da imagem, sem nenhum tipo de abstração inicial. (CHELLA et al., 2001) listam diversos trabalhos que tratam da extração de formas superquadráticas em imagens reais, mesmo com a presença de oclusões.

2.2.4 Abordagem 4: Abordagem Orion

A proposta apresentada pelo Grupo Orion (HUDELOT, 2005; HUDELOT et al., 2005; MAILLOT, 2005) define uma abordagem completa de visão cognitiva que se encaixa com precisão na representação em três níveis semânticos Figura 2.6.

Uma das principais preocupações da abordagem é a modelagem do ancoramento simbólico entre o nível semântico e o nível analógico. Por isso, duas ontologias são propostas para mapeamento entre os três níveis.

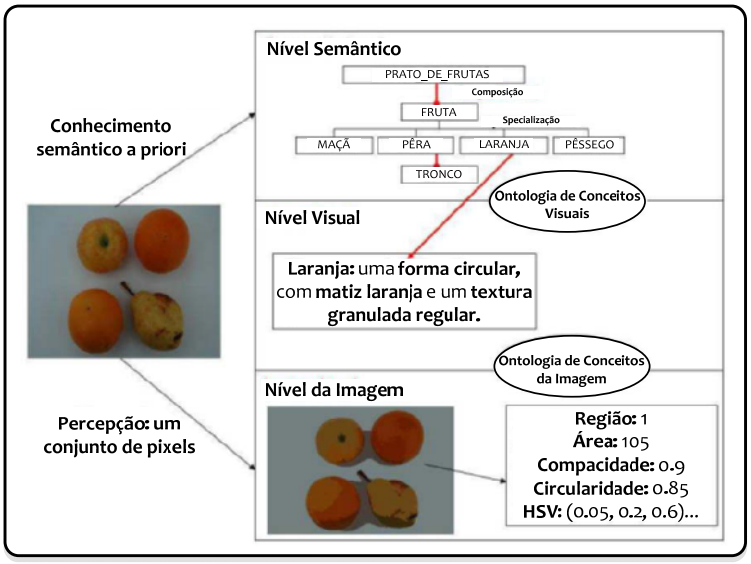

Figura 2.6: Exemplo de representação de uma laranja em uma imagem utilizando três níveis semânticos (HUDELOT et al., 2005).

A *Ontologia de Conceitos Visuais* faz o papel de um modelo intermediário entre o nível semântico e o nível analógico, representando as entidades do modelo conceitual através de "um vocabulário comum utilizado por humanos para representar visualmente objetos e cenas" (HUDELOT et al., 2005).

Já a *Ontologia de Conceitos Imagísticos* mapeia os resultados brutos dos algoritmos de processamento de imagens para o nível visual. Ela descreve as imagens e os resultados dos algoritmos do ponto de vista do especialista em processamento de imagens. Além disso, a ontologia inclui conceitos para representar as funcionalidades dos algoritmos de processamento de imagem, o que permite atrelar as entidades do nível intermediário aos processos que são utilizados para identificá-las e extraí-las das imagens.

A inferência do ancoramento semântico utiliza as duas formas de interpretação semântica de imagens (*top-down* e *bottom-up*).

Céline Hudelot (2005) propõe uma plataforma completa para interpretação semântica de imagens utilizando a abordagem Orion. A implementação do sistema foi realizada com a plataforma LAMA (CRUBÉZY et al., 1998) para desenvolvimento de sistemas baseados em conhecimento. A aplicação foi validada dentro do domínio da Biologia, para diagnóstico de doenças em roseiras.

2.2.4.1 Nível Semântico

O nível semântico representa o conhecimento visual eliciado de um especialista no domínio da aplicação. A abordagem define a forma com que o conhecimento deve ser eliciado, definindo os seguintes passos para a tarefa:

- Primeiramente, o especialista no domínio da aplicação provê uma taxonomia de conceitos do domínio. Esse modelo também inclui relações como *parte-de*;

- Então, os especialistas utilizam a Ontologia de Conceitos Visuais para descrever os conceitos do domínio com base na sua aparência visual e suas relações espaciais;
- Opcionalmente, o processo pode incluir a anotação de amostras de imagens utilizando o modelo semântico.

2.2.4.2 Nível Visual

O nível visual é representado pela já citada *Ontologia de Conceitos Visuais*. Ela é uma ontologia independente de domínio que permite modelar explicitamente os aspectos visuais de objetos do mundo, abstraídos de qualquer dado numérico. Ela é composta por:

- *Conceitos Espaciais* que descrevem objetos geometricamente. São conceitos como *forma*, *tamanho* e *localização*. São também representadas relações espaciais como *topologia*, *distância* e *orientação*.
- Conceitos para Cor, com termos relacionados a matiz, luminosidade e saturação.
- *Conceitos para Textura*, obtidos por experimentos da comunidade de ciência cognitiva. São conceitos como *textura granulada* e *orientada ou uniforme*.

Como dito na seção anterior, essa ontologia é diretamente mapeada para a ontologia de domínio no nível semântico.

2.2.4.3 Nível analógico

Como apresentado na seção 2.2.4.2, no nível analógico os dados brutos da imagem são interpretados por algoritmos de processamento de imagem e o resultado é representado pela *Ontologia de Conceitos Imagísticos*, formada por primitivas de representação ligadas ao domínio de processamento de imagens na computação. Ela é formada pelas seguintes partes:

- Um conjunto de 11 conceitos representando diferentes tipos de estruturas de dados que podem ser extraídos de imagens, como *regiões* e *arestas*, ou estruturas mais complexas, como *grafos de regiões e de vizinhança*.
- Um conjunto de 167 conceitos para representar características que podem ser medidas em imagens, como área, posição e propriedades de formas, cor e texturas.

A ontologia também contém conceitos para representar o conhecimento ligado aos algoritmos de processamento de imagem.

Todos esses conceitos permitem a construção de uma requisição de processamento de imagem com base no conhecimento representado no nível visual e que o resultado do processamento seja mapeado para ele.

O mapeamento pode acontecer de duas formas. Na forma automática, para um dado conceito visual (*azul claro*, por exemplo), são escolhidas as características imagísticas que melhor descrevem o conceito. A escolha é feita utilizando a

técnica de Analise Discriminante Linear (ADL), com base em imagens manualmente anotadas por especialistas no domínio. As características escolhidas formam a entrada do detector do conceito visual. Ele é constituído por uma Máquina de Vetor de Suporte (SVM, do inglês *Support Vector Machine*), treinada com base em imagens anotadas manualmente e específicas para o treinamento.

No mapeamento manual, a ligação entre os conceitos do nível visual e as características da imagem é explicitamente construída. A correspondência entre as características imagísticas numéricas e os conceitos no nível visual é representada por conjuntos fuzzy. Por exemplo, na Figura 2.7, o conceito *Superfície Elíptica* no nível visual é mapeado para o nível analógico por cinco conjuntos *fuzzy* definidos sobre a primitiva analógica excentricidade.

ConceitoVisual {
nome *Superfície_Elíptica*
SuperConceito *Superfície_Geométrica*
Link de Ancoramento
Símbolo nome excentricidade
comentário razão entra corda mais longa e a corda mais longa perpendicular a ele
Codomínio Linguistico: [muito_baixo baixo médio alto muito_alto]
ConjuntoFuzzy
$F_{muitobaixo}$ = {0.0, 0.0, 0.19, 0.21}
F_{baixo} = {0.19, 0.21, 0.38, 0.42}
$F_{médio}$ = {0.38, 0.42, 0.575, 0.625}
F_{alto} = {0.575, 0.625, 0.76, 0.84}
$F_{muitoalto}$ = {0.76, 0.84, 1, 1}
Domínio: [0 1]
unidade: nenhuma
Símbolo nome compacidade
comentário *Medida de como a forma é compacta*
...
Símbolo nome elipcidade
comentário Elipcidade Euclidiana. Distância entre a elipse circunscrita e a borda da região.
...
}

Figura 2.7. Exemplo de mapeamento do conceito *Superfície Elíptica* do nível visual para o nível analógico utilizando conjuntos *fuzzy* (HUDELOT, 2005).

2.3 Discussão e Comparação entre Abordagens de Interpretação Semântica de Imagens

No capítulo anterior foram descritas quatro abordagens de interpretação de conhecimento visual baseados em modelos de representação em três níveis conhecimento.

A primeira abordagem estudada, dos agregados, foca principalmente na representação no nível semântico com uma lógica descritiva formal. Ela trata de questões importantes no domínio visual, como representação taxonômica e mereológica, com a introdução de uma entidade conceitual básica chamada agregado. No campo da inferência, os autores justificam a utilização de uma lógica descritiva mínima, pois isso permite que os já bem estudados serviços de infe-

rência da lógica descritiva sejam utilizados para validar os modelos visuais e derivar novas interpretações.

No entanto, o ponto fraco da abordagem dos agregados está nos níveis inferiores. Pouco se fala sobre as formas de mapeamento do nível semântico para os níveis inferiores. As informações a respeito do modelo GSD são demasiadamente esparsas na literatura. Ainda sim, pelo estudo das informações disponíveis, o GSD não provê uma forma de representação clara e a forma de mapeamento entre ele e os outros níveis não está totalmente clara.

Já na abordagem abdutiva, o foco está na representação nos níveis visual e imagístico e no mapeamento entre os dois. A modelagem é baseada em lógica de primeira ordem, onde as entidades do nível visual são definidas como axiomas sobre primitivas do nível analógico. Isso permite a verificação formal do conhecimento modelado. Além disso, a abordagem inclui noção de ruído no nível visual, tornando a proposta menos suscetível a erros dos algoritmos de processamento de imagens. Como as primitivas do nível analógico são visualmente simples (linhas e pontos), a extração delas também se torna simples e robusta.

O ponto fraco da abordagem abdutiva está na sua simplicidade. Nenhum dos níveis inferiores busca representar características visuais como cor e textura. Isso limita a interpretação no nível conceitual a somente à forma e contorno dos objetos. Além disso, a modelagem no nível conceitual ainda foi pouco explorada.

A proposta dos espaços conceituais se encaixa melhor que as duas anteriores na abordagem em três níveis conceituais. Ela define um ótimo arcabouço teórico para representação de conhecimento visual em todos os níveis. O nível semântico define uma ontologia de representação genérica, que pode ser estendida, e uma representação para as suas instâncias. O nível visual é mapeado para o nível semântico através de funções de mapeamento formais e bem definidas, interpretadas através de um mecanismo de raciocínio imagístico baseado em atenção.

A abordagem, embora bastante sólida, sofre o mesmo problema de simplicidade da abordagem abdutiva. O nível visual leva em conta somente a forma de objetos visuais, deixando de fora atributos como cor, textura e etc. Isso inviabiliza a utilização da abordagem em áreas onde essas são as características que definem os objetos.

Finalmente, ao contrário das outras propostas, a abordagem Orion foca realmente na representação e interpretação em domínios intensivos em conhecimento visual. Ela define uma abordagem completa de visão cognitiva, definindo modelos de representação em todos os três níveis e seus respectivos mapeamentos. O problema do ancoramento semântico é abordado diretamente, com propostas de ancoramento manual e com técnicas de aprendizado de máquina.

A proposta Orion falha em não apresentar uma possibilidade de expansão para interpretação de vídeos, integrando noções formais de modelagem de eventos. Além disso, embora completa, a proposta pode se tornar muito comple-

xa para aplicação em dispositivos com pouco poder computacional embarcado (como robótica).

Por fim, a Tabela 2.1 faz uma comparação direta entre as abordagens estudadas, levando em conta as diversas dimensões pertinentes à interpretação semântica de imagens. Ela evidencia uma dicotomia interessante entre a abordagem Orion e as demais. Nota-se que a abordagem Orion se baseia em um formalismo de representação menos estruturado (*Frames*, em contraponto com o formalismo lógico). Isso está em conformidade com a aplicação básica em domínios intensivos em conhecimento, que normalmente requerem modelos de representação flexíveis e menos estruturados, sem grande preocupação com performance. Já nas outras abordagens, em especial a abdução e de espaços conceituais, a formalização lógica e matemática (como a superquadráticas) se adaptam aos requisitos de performance e robustez exigidos por sistemas robóticos e de tempo real.

Tabela 2.1: Tabela de comparação entre as quatro abordagens estudadas.

	Abordagem 1: Agregados	**Abordagem 2: Abdução**	**Abordagem 3: Espaços Conceituais**	**Abordagem 4: Orion**
Formalismo de Representação	Lógica descritiva	Lógica de primeira ordem	Lógica, Frames e superquadráticas	Frames
Aplicação Básica	Interpretação de cenas simples	Robótica e interpretação de cenas (trânsito)	Robótica	Recuperação de imagens e interpretação de cenas em domínios intensivos em conhecimento
Ancoramento Semântico	Não definido explicitamente; possivelmente manual.	Abordagem híbrida (*top-down* e *bottom-up*)	Abordagem híbrida (*top-down* e *bottom-up*)	Abordagem híbrida (*top-down* e *bottom-up*) ou manualmente
Processamento de imagem	Qualquer um que extraia modelos GSD de imagens	Algoritmo de detecção e extração de bordas	Qualquer algoritmo que gere volumes superquadráticos de imagens	Qualquer algoritmo de processamento de imagens ou manualmente (via anotação)

3 S-Chart: UM MODELO PARA INTERPRETAÇÃO SEMÂNTICA DE GRÁFICOS

Este capítulo descreve o framework S-Chart, a principal contribuição deste deste trabalho. O framework S-Chart busca capturar em modelos e algoritmos o comportamento de interpretação visual realizado por um especialista ao analisar um gráfico como o exposto na Figura 1.2b. Mais especificamente, será demonstrado como algoritmos de processamento de sinais podem ser combinados com algoritmos e modelos de raciocínio simbólico a fim de chegar a uma interpretação simbólica do gráfico. A proposta apresentada utiliza os princípios de interpretação semântica, de forma a combinar mecanismos de extração de padrões com conhecimento de domínio.

Para chegar a interpretação semântica da projeção gráfica de um sinal, o framework S-Chart propõe modelos e algoritmos em três níveis semânticos. O estudo de outras abordagens de interpretação semântica apresentadas no capítulo anterior e do requisito de aplicabilidade em domínios intensivos em conhecimento visual, sugerem a incorporação dos seguintes conceitos ao framework S-Chart:

- **Incorporação de ontologias de domínio.** O modelo que representa os objetos do domínio é incorporado no nível semântico. Embora ele possa ser mais orientado à aplicação, é interessante que o nível semântico possa incorporar modelos de conhecimento propriamente dito, como ontologias de domínio. Ontologias de domínio capturam conhecimento compartilhado sobre um determinado domínio de forma não ambígua. Uma ontologia de domínio aplicada ao nível semântico permite que as informações extraídas dos gráficos sejam compartilhadas mais eficientemente com outros agentes e sistemas. Em especial, estas informações podem ser disponibilizadas a outros mecanismos de raciocínio simbólico para realização de inferências adicionais ao sistema de interpretação visual.
- **Primitivas de modelagem independentes de domínio.** Como explicitado anteriormente, novas primitivas se fazem necessárias para suporte a interpretação de gráficos e dos dados onde eles se baseiam. Elas se localizam no nível analógico e no nível visual. Elas também devem ser genéricas o suficiente para permitir que o framework seja reutilizável em diversos domínios de aplicação.

- **Ancoramento simbólico explícito.** O mapeamento entre os símbolos dos três níveis deve ser explicitamente representado. Em técnicas de mapeamento conexionistas, um número suficiente de casos é necessário para treinamento e teste das redes que operacionalizam os mapeamentos. No entanto, em domínios restritos, mas intensivos em conhecimento visual, nem sempre se tem acesso ao número de casos necessários para treinamento. Uma vez que este trabalho foca em domínios como este, se faz necessária uma forma de mapeamento direta e que possa ser explicitamente representada.
- **Modelo processável por computador**: o formalismo escolhido para representação dos modelos de conhecimento deve ser formal e estruturado o suficiente para ser diretamente processável por computador. Ou seja, os modelos de conhecimento devem ser processáveis diretamente pelos algoritmos de interpretação, sem a necessidade de mapeá-los manualmente para formalismos de baixo nível.

As seções descrevem seguintes os modelos e algoritmos que compões o framework S-Chart incorporando os conceitos apresentados. Na seção seguinte, são feitas algumas definições sobre a linguagem de representação utilizada ao longo do capítulo. A seguir, são apresentados os modelos de conhecimento visual em três níveis, seguidos de algoritmos para interpretação semântica sobre estes modelos.

3.1 Linguagem de Representação

O formalismo escolhido para representação dos modelos de conhecimento visual e ancoramento simbólico é a linguagem OWL 1.0 (MCGUINNESS; HARMELEN, 2004). OWL foi proposto pela W3C como uma linguagem para representação de conhecimento na Web. No entanto, a sua padronização tem levado a uma aceitação maior para a formalização de conhecimento e informação nos mais diferentes tipos de sistemas.

A escolha de OWL como linguagem de representação neste trabalho se dá por fatores de natureza conceitual a prática. Do ponto de vista conceitual, a linguagem combina uma semântica bem definida com uma expressividade suficiente para representar os construtos do framework S-Chart. Todos os modelos apresentados a seguir (com exceção dos modelos de mapeamento) podem ser modelados com a sublinguagem OWL DL, que equivale à lógica de descrição $\mathcal{SHOIN}^{(\mathcal{D})}$. A Tabela 3.1 mostra quais primitivas OWL DL são utilizadas pra representar as principais primitivas S-Chart detalhadas nas próximas sessões. Adicionalmente, modelos descritos em OWL DL herdam as propriedades formais dessa $\mathcal{DL}$, o que possibilita uma modelagem mais precisa. De fato, Neumann e colegas (2008) suportam idéia da lógica descritiva como uma ferramenta para formalização de conhecimento visual. Uma segunda característica importante da linguagem é a possibilidade de *metamodelagem*, ou seja, é possível relações entre elementos dos modelos criados na linguagem com elementos do

metamodelo da própria linguagem (como os elementos *classe* e *propriedade*). Dessa forma é possível definir extensões da própria linguagem de representação. Essa característica é importante para criação das regras de mapeamentos entre os níveis de conhecimento visual, como apresentado na seção 3.2.4 a seguir.

Tabela 3.1: Primitivas OWL necessárias para representação de primitivas visuais do framework S-Chart.

Primitiva S-Chart	**Primitiva OWL DL**
Entidades Visuais	*Class*
Propriedades Visuais	*ObjectProperty* e *Class*
Relações Espaciais	*ObjectProperty*
Entidades Analógicas	*Class*
Propriedades Analógicas	*DatatypeProperty*

Do ponto de vista prático – e não menos importante –, a linguagem OWL apresenta uma série de características que facilitam a sua utilização em sistemas de conhecimento, como uma sintaxe bem defina (XML) e uma grande comunidade de usuários (com o desenvolvimento de ferramentas de suporte a modelagem e desenvolvimento). Na opinião do autor deste trabalho, a utilização de OWL diretamente como linguagem de representação no framework S-Chart pode facilitar a instanciação do framework nas diversas áreas de aplicação. No Capítulo 5, será demonstrado como a linguagem pode utilizada para implementação de um sistema de interpretação visual.

Nas seções seguintes, é utilizada a sintaxe OWL Manchester (HORRIDGE et al., 2006) para representação de expressões OWL, a fim de facilitar a leitura. Ao contrário da sintaxe padrão XML/OWL, a sintaxe Manchester é menos prolixa e mais simples de ser compreendida por seres humanos.

3.2 Arquitetura do Framework S-Chart

A arquitetura do framework S-Chart é composta por três elementos fundamentais (Figura 3.1): um componente de representação de conhecimento visual, um componente de ancoramento simbólico (ou "mapeamento") e um componente de interpretação e inferência de informações visuais.

O componente de representação é composto por um modelo de dividido em três níveis de representação (semântico, visual e analógico). Como em outras abordagens de interpretação simbólica, esses três níveis representam as informações extraídas do sinal em graus crescentes de abstração. O nível analógico captura informações numéricas extraídas diretamente do sinal, o nível visual captura feições visuais da projeção gráfica desse sinal, e o nível semântico captura as entidades do domínio reconhecidas visualmente no gráfico.

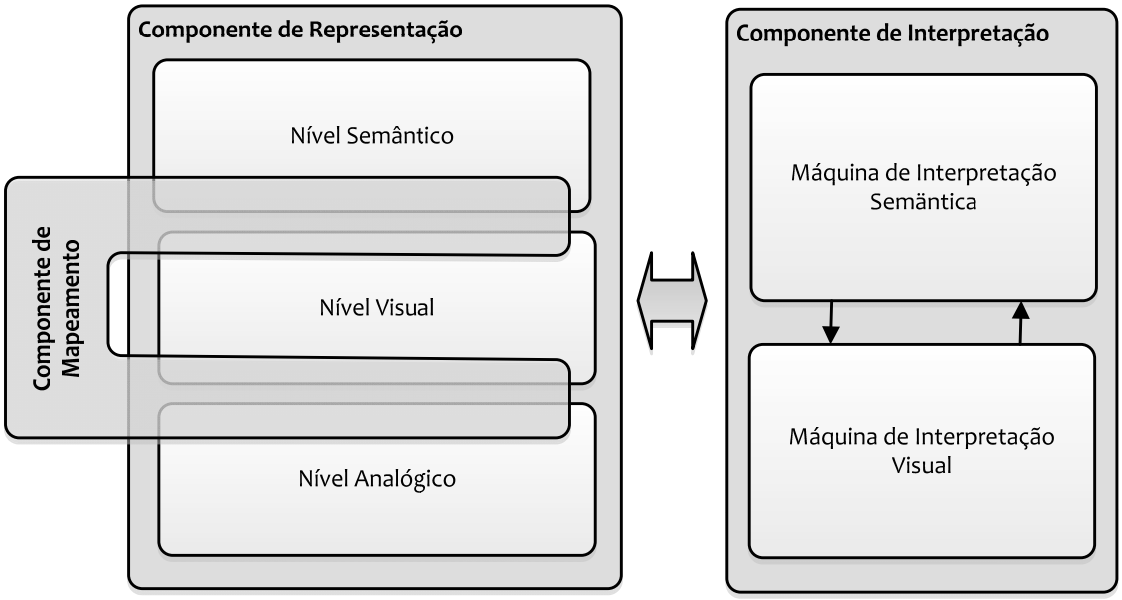

Figura 3.1: Componentes da arquitetura do *framework* S-Chart.

A informação utilizada para mapeamento entre entidades de níveis adjacentes é representada pelo componente de ancoramento simbólico. Ele é que indica explicitamente como o processo de interpretação deve abstrair entidades de um nível para o outro. Esse mapeamento indica, por exemplo, como em qual configuração certas feições analógicas devem aparecer no sinal para indicar a presença de uma feição visual.

O componente de interpretação processa o sinal em conjunto com os modelos de representação e ancoramento simbólico até inferir algum objeto do domínio presente no sinal. Esse mecanismo se baseia em geração e teste de hipóteses e se divide em duas partes. A *máquina de interpretação visual* infere feições visuais das informações contidas no nível analógico. Instâncias de entidades visuais são criadas (geração) e classificadas conforme os dados analógicos extraídos do sinal (teste). De forma análoga, *a máquina de interpretação semântica*, infere entidades do nível semântico conforme as feições visuais extraídas pela máquina de interpretação visual.

Nas subseções seguintes, cada um dos componentes da arquitetura do framework S-Chart é descrito em detalhes.

3.2.1 Nível semântico

O *nível semântico* comporta o modelo de conhecimento que define as entidades do domínio que podem ser identificadas no sinal interpretado. Todas essas entidades devem estar organizadas em uma única estrutura taxonômica e/ou mereológica. Ou seja, todos os conceitos do domínio devem ser estar relacionados por alguma relação de sub-classe ou de sub-parte.

Como apontado nos objetivos, ontologias de domínio podem ser utilizadas como modelo para o nível semântico. Ontologias são construídas com o intuito de capturar o conhecimento consensual de determinado domínio. Quando mo-

deladas corretamente prevendo integração de modelos, podem ser utilizadas como suporte para comunicação de diversos agentes (sistemas, usuário e etc). Assim, um sistema de conhecimento baseado em uma ontologia de domínio poderá integrar um sistema S-Chart se essa ontologia atender a estrutura básica apresentada anteriormente. No Capítulo 5, é apresentada uma aplicação onde é modelada uma ontologia no domínio da Estratigrafia de Sequências utilizada como modelo de conhecimento de domínio no nível semântico.

Para facilitar a importação desses modelos de conhecimento, o componente de representação define uma primitiva auxiliar no nível semântico na forma da classe `EntidadeSemântica`. Essa entidade representa qualquer objeto do domínio de discurso que foi extraído pelo sistema de interpretação. Ou seja, qualquer objeto interpretado será instância de um conceito do domínio e da classe `EntidadeSemântica`, em uma relação de herança múltipla. O seu papel no processo de interpretação ficará mais claro nas seções seguintes.

3.2.2 Nível visual

Ao contrário do nível semântico, o *nível visual* é independente do domínio de aplicação. Ele apresenta exclusivamente as primitivas que representam feições visuais genéricas, identificáveis por qualquer ser humano em gráficos como o da Figura 1.2b. Por causa disso, o modelo de conhecimento que captura as primitivas visuais pode ser classificado como uma *ontologia visual*, uma vez que ela representa simbolicamente os elementos visuais de uma conceitualização compartilhada.

As feições visuais de um gráfico são capturadas por três tipos de construtos básicos: (i) as *entidades visuais*, que representam os objetos visuais concretos; (ii) as *propriedades visuais*, que caracterizam as entidades; (iii) *relações espaciais*, definidas entre as entidades visuais. Todas essas primitivas modeladas em uma ontologia, organizadas como uma taxonomia de conceitos e de relações.

3.2.2.1 Entidades Visuais

Uma entidade visual corresponde a uma feição visual concreta, que foi reconhecida no gráfico. Esse conceito é capturado pela classe `EntidadeVisual` no modelo ontológico. Somente entidades visuais têm extensão na realidade, logo, qualquer objeto visual inferido pela interpretação visual será instância da classe `EntidadeVisual` ou de alguma de suas subclasses.

A classe `EntidadeVisual` é especializada nos diversos tipos de entidades visuais que podem aparecer em um gráfico (Figura 3.2). Como apresentado no Capítulo 2, diversos trabalhos propuseram ontologias para representação de conhecimento no nível visual. Em geral, essas ontologias compartilham as mesmas primitivas elementares. As primitivas mais escuras na Figura 3.2 foram propostas no modelo de conhecimento visual proposto pela abordagem Orion (seção 2.2.4) e reutilizadas aqui. As novas primitivas definidas neste trabalho descrevem feições visuais específicas de gráficos, como tipos de curvas e pontos

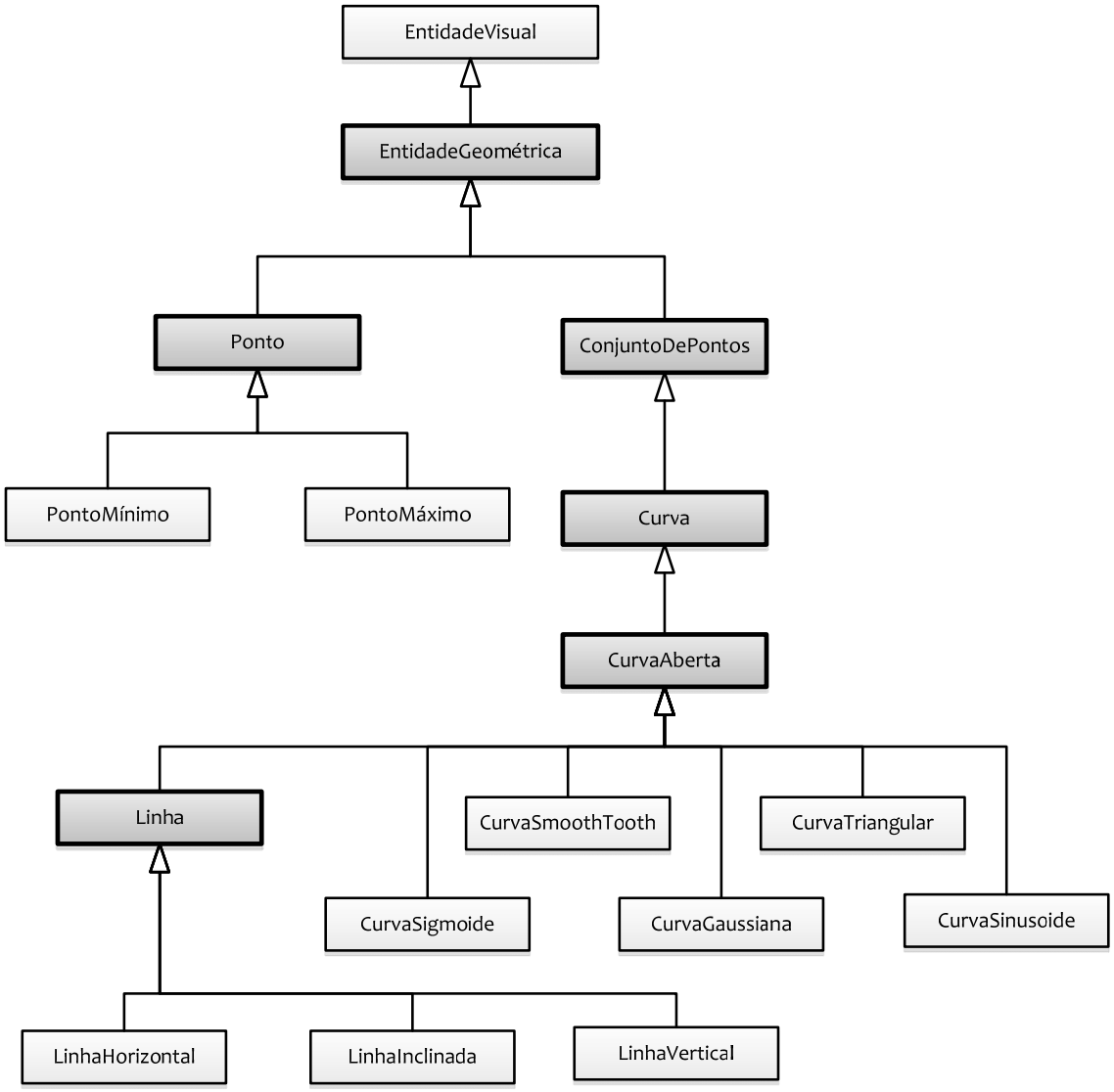

Figura 3.2: Taxonomia de entidades visuais.

máximos e mínimos. Essa listagem não pretende ser extensiva e pode ser estendida com outras primitivas (ex.: outros tipos de curva) conforme a necessidade.

3.2.2.2 Propriedades Visuais

Algumas feições visuais são modeladas como propriedades de entidades visuais. Ou seja, são feições visuais que tem a sua existência dependente da existência de uma entidade visual. Dessa forma, todas as propriedades visuais têm o seu domínio definido em `EntidadeVisual` ou em alguma de suas subclasses. As propriedades visuais também são estruturadas como uma taxonomia de propriedades, segundo a relação de subpropriedade de OWL (Figura 3.3). No entanto, a definição do contradomínio de propriedades visuais necessita de uma discussão adicional.

O nível visual descreve simbolicamente aspectos visuais genéricos de uma imagem. Aspectos que possam ser descritos por uma linguagem comum a qualquer imagem, livre de detalhes de baixo nível. Essa opção exclui a possibi-

lidade de definição de propriedades visuais através de contradomínios contínuos, numéricos. Ou seja, no framework S-Chart, as propriedades visuais mapeiam instâncias de entidades visuais em conjuntos de valores expressos na forma de símbolos lingüísticos ou valores simbólicos. Por exemplo, propriedades como `comprimento` podem assumir somente valores lingüísticos como `longo`, `muito longo`, `curto` e etc. Neste trabalho, estes valores são vistos como instâncias de uma categoria de valores possíveis (ex.: `ValoresDeComprimento`). Essa alternativa de modelagem é baseada na proposta de Alan Rector (RECTOR, ALAN, 2005), fundamentada na caracterização formal de espaços de qualidades (*quality spaces*) (GANGEMI et al., 2003). Assim, o contradomínio de cada propriedade visual é modelado como um conceito específico, cujas instâncias caracterizam os possíveis valores que a propriedade pode assumir. A Figura 3.4 apresenta um exemplo disso.

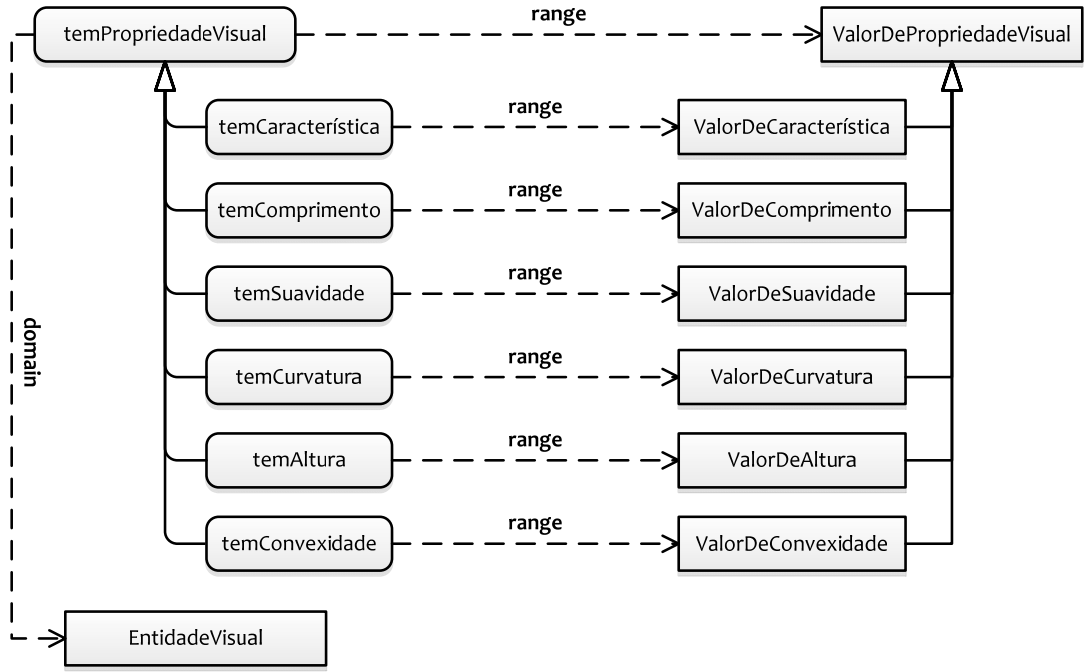

Figura 3.3: Taxonomia de propriedade visuais com definição de contradomínio de valores. A especialização do domínio é omitida.

Contudo, existem alguns aspectos de propriedade visuais que devem ser levados em conta. É sabido que a interpretação humana de propriedades visuais como *comprimento* é bastante subjetiva. Ela varia conforme o contexto que se está considerando ou a entidade a que se refere. Por exemplo, do ponto de vista visual, a representação de um avião com comprimento longo certamente é diferente da interpretação do comprimento longo para a representação de um carro. Ambos adquirem um significado diferente, pois se referem a entidades visuais em escalas diferentes. Se o mesmo conjunto de valores simbólicos para comprimento é utilizado para caracterizar entidades em escalas diferentes, será impossível interpretá-las de forma correta. Uma propriedade visual não pode ter um contradomínio com um único conjunto de valores. Pelo menos não obrigatoriamente.

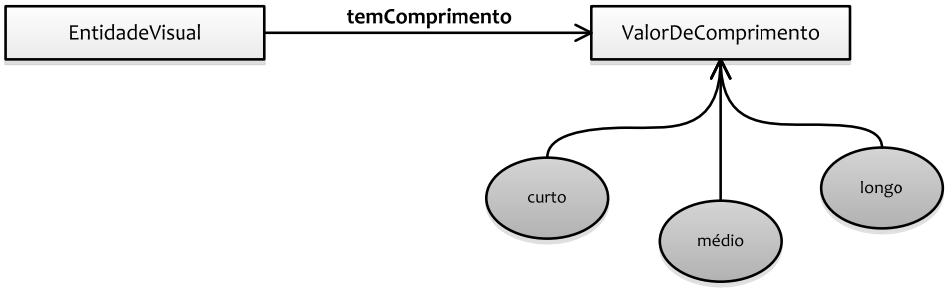

Figura 3.4: Exemplo de modelagem de contradomínios de propriedades visuais.

Para resolver este problema – ao mesmo tempo em que se mantém a restrição de contradomínio com valores simbólicos – escolheu-se estender o modelo apresentado anteriormente permitindo a representação dos contradomínios de propriedades visuais como uma partição de classes (Figura 3.5). Nesse modelo as instâncias que a compõem são os possíveis valores para a propriedade. Assim, propriedades visuais que tem mais de uma interpretação em um dado domínio de discurso podem ser especializadas para suportar mais de um tipo de contradomínio[7].

Em alguns casos é preciso que se mantenha uma relação de ordem explícita entre os símbolos que constituem o contradomínio de propriedades visuais, a fim de se simplificar o processo de interpretação. No caso da propriedade `comprimento`, por exemplo, existe uma relação de "tamanho" entre os elementos do seu contradomínio (`longo` é maior que `médio` que, por sua vez, é maior que `curto`). Para representar esse tipo de relação, o nível visual define a propriedade `maiorQue` e a sua inversa `menorQue`, ambas transitivas, anti-simétricas e irreflexivas.

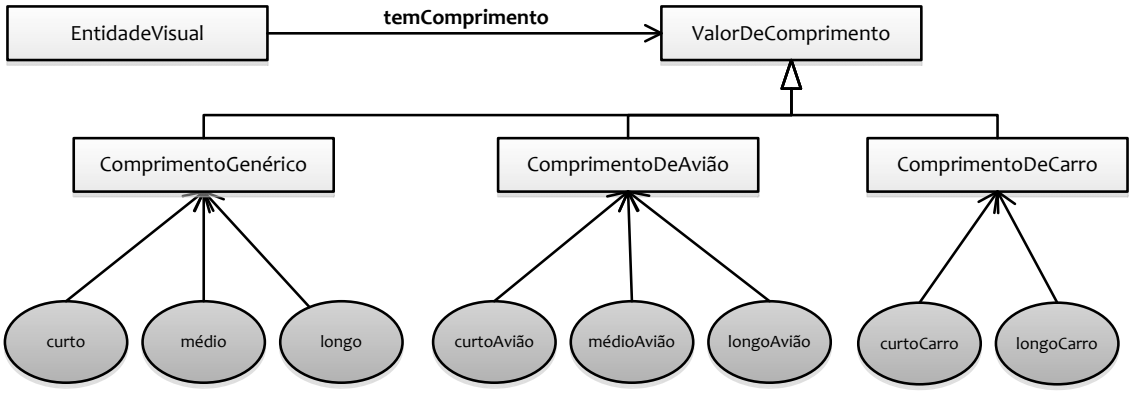

Figura 3.5: Extensão da modelagem de contradomínios de propriedades visuais.

[7] A relação de subpropriedade permite também a especialização das propriedades visuais, o que também colabora para solução do problema de múltiplos domínios.

3.2.2.3 *Relações Espaciais*

Relações espaciais capturam as formas de relacionamento entre entidades do domínio visual. Elas podem ser relações de orientação ou relações topológicas. A sua função é permitir a combinação de entidades visuais em entidades mais complexas. Da mesma forma como acontece com as propriedades visuais, as relações espaciais são representadas em OWL como um taxonomia de propriedades (Figura 3.6). A propriedade OWL `temRelaçãoEspacial` constitui a raiz dessa taxonomia. Se domínio e contradomínio são definidos no conceito `EntidadeVisual`, bem como todas as suas subpropriedades.

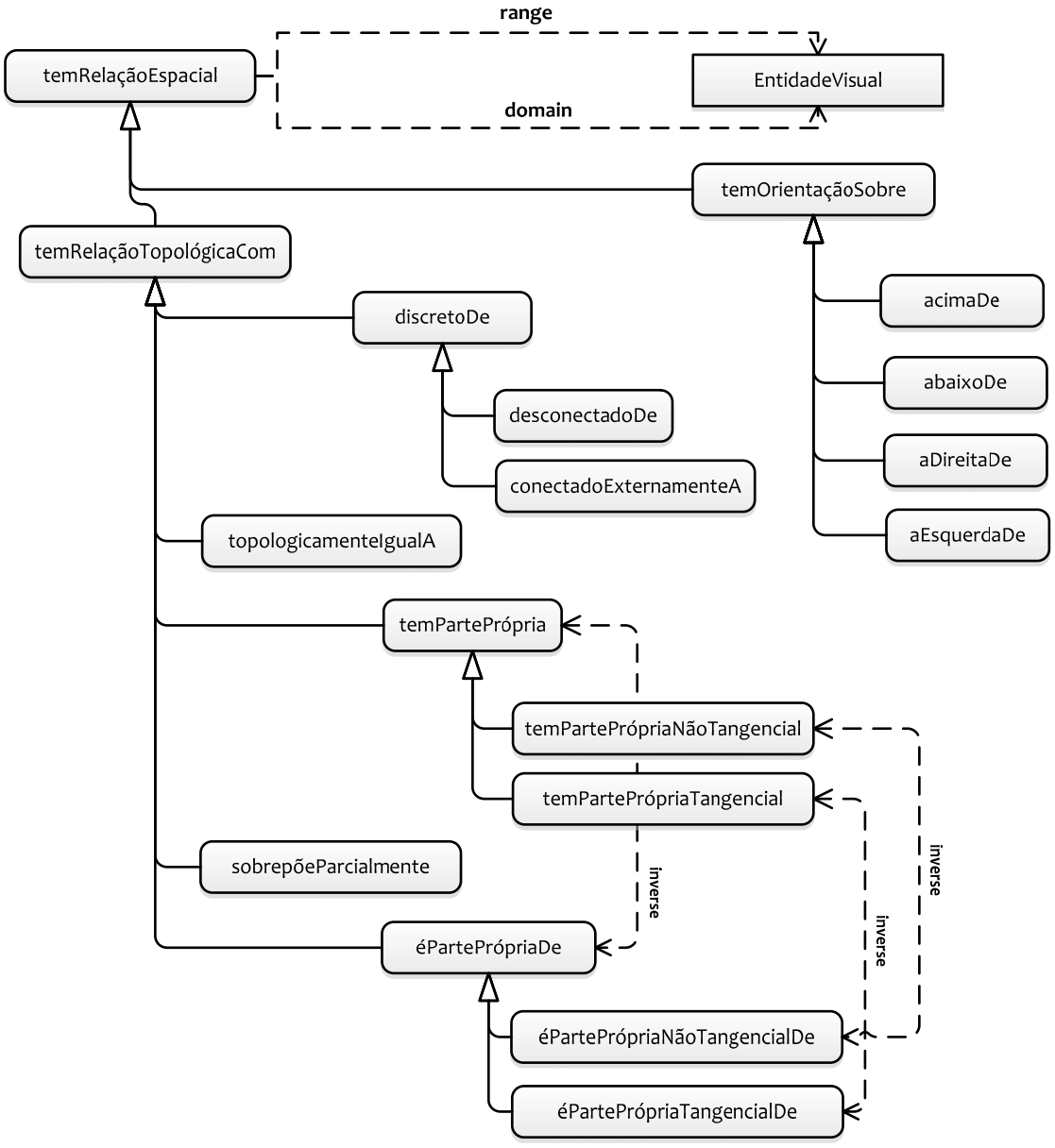

Figura 3.6: Taxonomia de relações espaciais.

Representação de conhecimento e raciocínio sobre relações topológicas são tópicos bastante recorrentes na literatura. Em um dos trabalhos mais importante da área, Cohn e colegas (1997; 1992) propuseram um conjunto de primitivas

simples e robusto para representação de relações topológicas, chamado RCC-8. O seu objetivo é capturar todas as possíveis relações topológicas entre duas entidades físicas. No framework S-Chart as primitivas do RCC-8 são modeladas na forma de uma taxonomia, como em Santin (2008), conforme a Figura 3.6.

Relações de orientação também são importantes para modelar como dois objetos estão posicionados um em relação ao outro. O nível visual disponibiliza as propriedades `acimaDe`, `abaixoDe`, `aDireitaDe` e `aEsquerdaDe`. Essas quatro relações induzem um espaço de duas dimensões espaciais no nível visual. Como dito anteriormente, embora o processamento do sinal se dê em um espaço de uma dimensão, o raciocínio no nível visual é baseado em um espaço duas dimensões.

3.2.3 Nível analógico

As primitivas no nível analógico representam informações extraídas diretamente pelos algoritmos de processamento de sinal, como pontos, intervalos e padrões. Analogamente ao nível visual, as primitivas analógicas se dividem em entidades analógicas e suas propriedades (Figura 3.7).

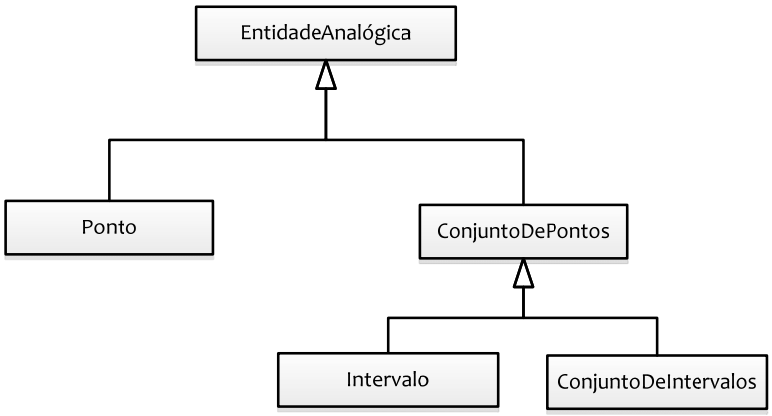

Figura 3.7: Taxonomia de entidades analógicas e relações de composição.

Um sinal analógico digitalizado é caracterizado como um conjunto de pontos ordenados por uma dimensão espacial. Uma *entidade analógica* é um conjunto de um ou mais pontos em alguma disposição. Neste trabalho, uma entidade visual pode ser um ponto, um conjunto de pontos, um intervalo ou um conjunto de intervalos (Figura 3.7). Entidade básica de um sinal, o `Ponto` é descrito pelas propriedades *valor* e *índice*:

```
DatatypeProperty: valor
   Annotations:
      rdfs:comment "Valor da medida de um Ponto (y)"
   Domain: Ponto
   Range: float
   Characteristics: Functional

DatatypeProperty: indice
   Annotations:
```

```
      rdfs:comment "Índice da medide de um Ponto (x)"
   Domain: Ponto
   Range: float
   Characteristics: Functional
```

Uma entidade analógica pode ser formada por outras. Essas relações de conjuntos são capturadas por relações de composição, como a propriedade `temPonto` entre `ConjuntoDePontos` e `Ponto`. No caso da primitiva `Intervalo`, alguns pontos podem ter um papel especial de ponto inicial, final, central, entre outros. Abaixo segue uma lista destas relações.

```
ObjectProperty: temPonto
    Domain: ConjuntoDePontos
    Range: Ponto

ObjectProperty: temPontoInicial
    SubPropertyOf: temPonto
    Domain: Intervalo
    Characteristics: Functional

ObjectProperty: temPontoCentral
    SubPropertyOf: temPonto
    Domain: Intervalo
    Characteristics: Functional

ObjectProperty: temPontoFinal
    SubPropertyOf: temPonto
    Domain: Intervalo
    Characteristics: Functional

ObjectProperty: temPontoMáximo
    Annotations:
      rdfs:comment "Ponto de valor máximo de um Intervalo"
    SubPropertyOf: temPonto
    Domain: Intervalo
    Characteristics: Functional

ObjectProperty: temPontoMínimo
    Annotations:
      rdfs:comment "Ponto de valor mínimo de um Intervalo"
    SubPropertyOf: temPonto
    Domain: Intervalo
    Characteristics: Functional
```

Semelhante ao que o ocorre no nível visual, as entidades analógicas são caracterizadas por *propriedades analógicas*, feições analógicas extraídas por algoritmos de processamento numérico. Cada entidade analógica é definida pelo seu próprio conjunto de propriedades. Em geral, uma propriedade analógica é um índice numérico calculado sobre as informações analógicas contidas no sinal. Alguns exemplos:

```
DatatypeProperty: temPropriedadeAnalógica
   Domain: EntidadeAnalógica
   Range: float

DatatypeProperty: temAreaAbsoluta
   Domain: Intervalo
   Annotations:
      rdfs:comment "Valor de área total sob a curva de um Intervalo"
   SubPropertyOf: temPropriedadeAnalógica

DatatypeProperty: temArea
   Domain: Intervalo
   Annotations:
      rdfs:comment "Valor da integral de um Intervalo"
   SubPropertyOf: temPropriedadeAnalógica

DatatypeProperty: temComprimento
   Domain: Intervalo
   Annotations:
      rdfs:comment "Valor de comprimento de um Intervalo"
   SubPropertyOf: temPropriedadeAnalógica

DatatypeProperty: temTendencia
   Domain: Intervalo
   Annotations:
      rdfs:comment "Valor de inclinação de uma reta entre Ponto inicial
         e Ponto final de um Intervalo"
   SubPropertyOf: temPropriedadeAnalógica

DatatypeProperty: temMédia
   Domain: Intervalo
   Annotations:
      rdfs:comment "Valor médio da curva em um Intervalo"
   SubPropertyOf: temPropriedadeAnalógica


DatatypeProperty: temGrauDeRuído
   Annotations:
      rdfs:comment "Grau de ruído em uma EntidadeAnalógica"
   SubPropertyOf: temPropriedadeAnalógica
```

Os algoritmos de processamento de sinal disponíveis também são modelados no nível analógico, dado o importante papel que exercem na interpretação. Outros trabalhos buscam representar esse tipo de informação no nível analógico, como é o caso do projeto Orion apresentado na seção 2.2.4. Dessa forma, é possível manter uma relação explícita entre as entidades visuais extraídas do sinal e os algoritmos que a extraíram. Essa informação é importante, uma vez que o algoritmo usado para extrair uma entidade pode dizer muito sobre a sua natureza. Aqui, os algoritmos são classificados em uma taxonomia simples, demonstrada na Figura 3.8. O relacionamento entre as entidades visuais e os algoritmos de processamento se dá propriedade pela propriedade `éExtraídoCom`. Uma entidade é extraída com um dado grau de certeza, capturado pela relação `temForçaDeExtração`. Mais especificamente:

```
ObjectProperty: éExtraídoComo
    Domain: EntidadeAnalógica
    Range: Segmentador

DatatypeProperty: temForçaDeExtração
   Domain: EntidadeAnalógica
   Range: float
```

Embora o nível analógico não busque representar todos os possíveis tipos de algoritmos, ele serve como uma base para futuras extensões, conforme as necessidades de cada domínio de aplicação.

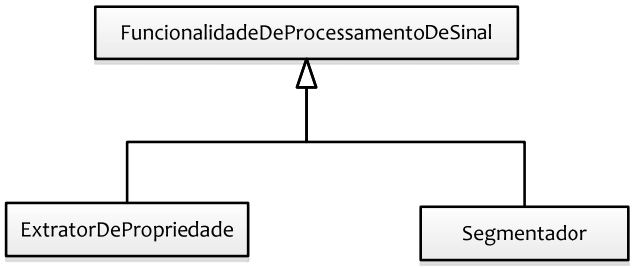

Figura 3.8: Taxonomia básica de funcionalidades de processamento de sinal.

3.2.4 Mapeamento e ancoramento simbólico

Cada nível semântico é um modelo por si só, independente dos demais. No entanto, para que os níveis possam ser combinados, é necessário algum tipo de mapeamento. Essencialmente, o problema de ancoramento simbólico entre os níveis pode ser dividido em duas partes. A primeira tange ao mapeamento de primitivas mais abstratas em primitivas menos abstratas. Ou seja, como entidades básicas de um nível podem ser combinadas para gerar uma entidade no nível semântico superior. Por exemplo, como e quais primitivas analógicas (ex.: pixels e conjuntos de pixels) devem ser combinadas para indicar a existência de uma entidade no nível visual (ex.: uma curva de algum padrão específico). A segunda parte do problema consiste em, depois de realizada a extração de uma instância de entidade no nível superior, manter uma relação concreta desta com as instâncias que serviram como evidências no nível inferior.

A fim de manter os modelos de mapeamento corretamente estruturados, é definido que entidades e objetos de um nível de abstração podem ser mapeados exclusivamente no nível imediatamente inferior. Ou seja, entidades do nível semântico podem ser mapeadas exclusivamente no nível visual e estas, por sua vez, no nível analógico.

Uma peculiaridade deste modelo é que a formalização do mapeamento entre primitivas é definida com base no mapeamento entre instâncias. Embora essa relação pareça contra-intuitiva, ela se deve a forma como os algoritmos de interpretação operam. A necessidade dessa relação é detalhada na seção 3.2.5 a seguir.

3.2.4.1 *Mapeamento entre instâncias*

O ancoramento simbólico requer que entidades interpretadas sejam ligadas explicitamente às evidências que suportam essa interpretação. Como descrito anteriormente, neste trabalho essas entidades interpretadas – e as suas evidências – são representadas como instâncias de entidades de conhecimento visual. Assim, um modelo de ancoramento simbólico deve incorporar uma forma de mapeamento explícito entra as instâncias que representam as entidades detectadas em um nível e as instâncias que representam as evidências encontradas no nível inferior. Para isso, o modelo de mapeamento incorpora uma relação básica e quatro relações específicas:

```
ObjectProperty: temMapeamentoSimbólico

ObjectProperty: mapeamentoSemânticoParaVisual
   SubPropertyOf: temMapeamentoSimbólico
   Domain: EntidadeSemântica
   Range: EntidadeVisual
   InverseOf: mapeamentoVisualParaSemântico

ObjectProperty: mapeamentoVisualParaSemântico
   SubPropertyOf: temMapeamentoSimbólico
   InverseOf: mapeamentoSemânticoParaVisual

ObjectProperty: mapeamentoVisualParaAnalógico
   SubPropertyOf: temMapeamentoSimbólico
   Domain: EntidadeVisual
   Range: EntidadeAnalógica
   InverseOf: mapeamentoAnalógicoParaVisual

ObjectProperty: mapeamentoAnalógicoParaVisual
   SubPropertyOf: temMapeamentoSimbólico
   InverseOf: mapeamentoVisualParaAnalógico
```

As quatro relações ajudam a definir limites claros entre os níveis, estruturando-os. Ou seja, entidades do nível semântico podem ser mapeadas *somente* para entidades do nível visual e estas *somente* para entidade no nível analógico. Não é possível mapear o nível semântico diretamente no nível analógico.

3.2.4.2 *Mapeamento entre primitivas: detectores simbólicos*

O mapeamento entre primitivas de níveis de abstração distintos é definido por *detectores simbólicos*. Um detector simbólico define quais são as condições necessárias para que uma dada primitiva semântica ou visual seja corretamente detectada. Os mecanismos de interpretação utilizam essas condições para testar as hipóteses de entidades semânticas e visuais. Por exemplo, um detector simbólico para a primitiva semântica `Sequência` define que uma instância de sequência será interpretada somente se uma instância da primitiva visual `Curva-Gaussiana` estiver presente e sua propriedade tem`Comprimento` tiver o valor `longo`.

Cada primitiva detectável é mapeada por um conjunto com um ou mais detectores especializados. Da mesma forma que o mapeamento entre instâncias, um detector relaciona primitivas somente entre níveis de abstração adjacentes (semântico/visual e visual/analógico). Cada detector é modelado como uma única regra. O formalismo escolhido aqui para a definição de regras é a linguagem SWRL (*Semantic Web Rule Language*), uma linguagem de regras definida sobre a linguagem OWL (HORROCKS et al., 2004). Para ser eficiente na modelagem de detectores simbólicos, a definição de SWRL deve ser restringida. Inicialmente, uma regra SWRL tem a forma

$$antecedente \Rightarrow consequente,$$

onde ambos – antecedente e conseqüente – são compostos por uma conjunção de *átomos* na forma $a_1 \wedge a_2 \wedge a_3 \wedge ... \wedge a_n$. Os átomos são predicados *n*-ários (ou *datalogs*) semelhantes aos da lógica de primeira ordem. Eles são de três tipos básicos: (i) predicados unários representando classes do modelo (ex.: *Pessoa(?x)*); (ii) predicados binários, representando propriedades do modelo (ex.: *temFilho(?x, ?y)*); e (iii) predicados *n*-ários, representando funções embutidas da linguagem SWRL (ex.: *somar(?r, ?op1, ?op2)*). Um exemplo de regra SWRL é

```
Pessoa(?x) ∧ temFilho(?x, João) ⇒ temPai(João, ?x)
```

que infere a propriedade `temPai` sobre o individuo de nome `João`, com base na classe `Pessoa` e na propriedade `temFilho`. Uma regra SWRL é satisfeita se, para uma dada combinação de valores possíveis para as variáveis (símbolos precedidos de ponto-de-interrogação), o antecedente for satisfeito. Se isso acontecer, o mecanismo de inferência SWRL infere que o conseqüente também é verdadeiro.

Para aplicação em detectores simbólicos, os possíveis predicados que compõem o antecedente e o conseqüente devem ser restritos, a fim de não permitir mapeamentos entre níveis não adjacentes. Assim, dado um conjunto P de predicados unários (conceitos), binários (propriedades) e predicados *n*-ários (funções), sobre primitivas de um modelo de conhecimento visual. Dados os conjuntos de predicados P_{inf} de primitivas de um dado nível (ex.: n. analógico), P_{sup} de primitivas do nível de semântico imediatamente superior (ex.: n. visual), P_{map} de primitivas de mapeamento entre instâncias e P_{fun} de funções de SWRL, tal que $P_{\text{inf}} \cup P_{\text{sup}} \cup P_{map} \cup P_{fun} \in P$. Dada uma regra r, um conjunto A de predicados do seu antecedente e um conjunto C de predicados do seu conseqüente. Ou seja, se $A = \{a_1, a_2, a_3, ..., h_n\}$ e $C = \{c_1, c_2, c_3, ..., c_m\}$, a regra r é dada por

$$a_1 \wedge a_2 \wedge a_3 \wedge ... \wedge a_n \Rightarrow c_1 \wedge c_2 \wedge c_3 \wedge ... \wedge c_m .$$

A regra r é um detector simbólico se $A \subset (P_{\text{inf}} \cup P_{\text{sup}} \cup P_{map} \cup P_{fun})$ e $C \subset (P_{\text{sup}} \cup P_{fun})$. Nessa forma, um detector simbólico pode ter como antecedente um conjunto de primitivas em ambos os níveis relacionados, porém pode gerar novas conclusões somente no nível de abstração superior considerado. Isso, em conjunto com as relações de mapeamento entre instâncias da seção anterior,

organiza o domínio e evita que entidades do domínio sejam inferidas diretamente a partir do sinal.

Como mencionado anteriormente, cada primitiva detectável é mapeada por um conjunto de detectores simbólicos. Essa relação é formalizada restringindo-se o conjunto de axiomas *C* da seguinte forma. Dado um conjunto P_d composto de predicados representando primitivas detectáveis de um modelo, um predicado $\rho \in P_d$ e um conjunto $S_\rho \subseteq P_d$ contendo ρ e todas as suas subprimitivas. Dado um conjunto *DS* de todas as regras de detecção desse modelo, uma regra $r \in DS$ e um conjunto C_r corresponde ao conjunto de predicados do conseqüente da regra *r*. Então existe uma relação $\langle \rho, DS_\rho \rangle$ tal que $DS_\rho \subset DS$ e $\forall r \exists s (r \in DS_\rho \wedge s \in C_r \wedge s \in S_\rho)$. Dessa forma, toda regra de detecção associada a uma primitiva deve conter no seu conseqüente essa mesma primitiva ou alguma de suas subprimitivas (subclasse ou subpropriedade). Por exemplo, considerando um detector simbólico que mapeia a primitiva semântica `Sequência` nas primitivas visuais `CurvaGaussiana` e `temComprimento` com valor `longo`. Para atender a todas restrições apresentadas anteriormente, ele teria a forma:

```
EntidadeSemântica(?es) ∧ mapeamentoSemânticoParaVisual(?es, ?ev)
  ∧ EntidadeVisual(?ev) ∧ CurvaGaussiana(?ev)
  ∧ temComprimento(?ev,longo)
⇒ Sequencia(?es)
```

É importante que os detectores possam ser integrados diretamente nos modelos OWL utilizados para representar o conhecimento visual, uma vez que conectam os níveis de abstração representados nesse formalismo. O padrão SWRL define ainda um metamodelo para representação de regras integradas a modelos OWL. Ele associa explicitamente átomos que compõem as regras às classes, propriedades e funções correspondentes. Assim, classes e propriedades utilizadas nas regras podem ser checadas contra o modelo onde estão representadas.

Além disso, o metamodelo pode ser estendido para adicionar novas características às regras. Essa característica é importante, pois permite estender a linguagem SWRL para dar suporte ao agrupamento de regras em conjuntos de detectores simbólicos e ao relacionamento explícito destes conjuntos com as primitivas de conhecimento visual que devem ser extraídas. Na prática, Martin O'Connor (O'CONNOR et al., 2005) propõe uma extensão no metamodelo de SWRL, incluindo alguns construtos para agrupamento de regras. De forma simplificada, eles são:

```
Class: RuleGroup
   SubClassOf: Entity

ObjectProperty: hasRuleGroup
   Domain: swrl:Imp
   Range: RuleGroup
```

onde `swrl:Imp` caracteriza a meta-entidade que captura uma regra SWRL. Ou seja, por esse modelo, regras SWRL pode ser agrupados em instâncias de `RuleGroup`. Os serviços de raciocínio de regras que dão suporte a esse construto podem executar conjuntos de regras separadamente.

Ainda sim, não é possível representar a relação (ρ, DS_ρ) entre uma primitiva e um conjunto de regras. Para isso, este trabalho define as seguintes metapropriedades:

```
ObjectProperty: extraídoPorDS
   Annotations:
      rdfs:comment "Relaciona primitivas com grupos de
         detectores simólicos. Domínio pode ser qualquer construto"
   Range: swrla:RuleGroup
   InverseOf: extraiPrimitiva
   Characteristics: Functional, InverseFunctional

ObjectProperty: extraiPrimitiva
   Domain: swrla:RuleGroup
   Range: owl:Thing
   InverseOf: extraídoPorDS
```

O mecanismo de interpretação utiliza a relação `extraídoPorDS` junto com os grupos de regras SWRL para acionar somente o conjunto de detectores simbólicos necessários a extração de uma primitiva.

Os detectores simbólicos são utilizados para mapeamento entre quaisquer níveis. No entanto, o mapeamento entre do nível visual e o nível analógico deve levar em conta a conversão de valores numéricos em objetos simbólicos. Por exemplo, dever ser possível representar explicitamente que o valor `MuitoLongo` para a propriedade visual `comprimento` corresponde a um comprimento analógico em um dado intervalo de valores (ex.: entre 200m e 1500m).

De fato, o problema de quantização de valores é recorrente na Inteligência Artificial. Entre as diversas possíveis soluções existentes na literatura, está a solução de mapeamento por conjuntos *fuzzy* proposta inicialmente por Coradeschi e colegas (CORADESCHI et al., 2001). No seu trabalho, Coradeschi define que uma propriedade analógica pode ser convertida para valores simbólicos através de conjuntos *fuzzy* definidos sobre o seu contradomínio. Um conjunto *fuzzy* é definido como um par $\langle D, m \rangle$ onde D é um conjunto de valores possíveis e m uma função tal que $m: D \rightarrow [0,1]$, chamada *função de associação*. Para cada $x \in D$, $m(x)$ dá o grau de associação de x ao conjunto D. Se $m(x) = 0$, então diz-se que o x não pertence ao conjunto *fuzzy* $\langle D, m \rangle$. Dando rótulos simbólicos para os conjuntos e agrupando-os sobre um mesmo espaço de valores D, é possível quantizar esse espaço de valores.

A inclusão de conjuntos *fuzzy* para mapeamento entre os níveis visual e analógico requer que a definição dos detectores simbólicos seja sutilmente ampliada. Em especial, o conjunto de predicados A definido anteriormente, que cor-

responde às primitivas de hipótese, deve prover algum tipo operador *fuzzy*. Esse operador é definido como uma função SWRL *n*-ária na forma

$$fuzzy : associa(?v, ?r, ?fa_1, ?tc_1, ?fa_2, ?tc_2, \cdots, ?fa_n, ?tc_n),$$

onde as variáveis ?v e *?r* são interpretadas como o valor testado pela função e o termo resultante, respectivamente. Cada dupla $?fa_n$ e $?tc_n$ define uma função de associação trapezoidal com a sintaxe *[v1 v2 v3 v4]* ($\{v1, v2, v3, v4\} \in D \wedge D \subset \square$) e respectivo termo que representa o conjunto. Os termos devem pertencer ao mesmo domínio de valores de uma propriedade visual. Por exemplo, o detector simbólico

```
EntidadeVisual(?ev) ∧ EntidadeAnalógica(?ea)
  ∧ mapeamentoVisualParaAnalógico(?ev, ?ea)
  ∧ Intervalo(?ea) ∧ comprimento(?vn)
  ∧ fuzzy:associa(?vn, ?r, "[0,1 5 10 15]", comprimentoCurto,
      "[10 15 20 25]", comprimentoLongo)
  ⇒ temComprimento(?ev, ?r)
```

converte valores analógicos de `comprimento` para valores simbólicos de `temComprimento` no nível visual. A Tabela 3.2 abaixo mostra uma série de possíveis valores para as variáveis *?vn* e *?r* do exemplo anterior e o resultado da avaliação do predicado.

Tabela 3.2: Avaliação da função `fuzzy:associa`.

?vn	*?r*	**Avaliação**
0,0	comprimentoCurto	*Falso*
1,2	comprimentoCurto	*Verdadeiro*
	comprimentoLongo	*Falso*
11,0	comprimentoCurto	*Verdadeiro*
	comprimentoLongo	*Verdadeiro*
21	comprimentoCurto	*Falso*
	comprimentoLongo	*Verdadeiro*
30	comprimentoLongo	*Falso*

Os detectores simbólicos permitem, então, que se estruturem os modelos de conhecimento organizadamente. Eles fornecem meios para facilitar a inferência entre os níveis semânticos, ao mesmo tempo em que mantém o nível de expressividade dos modelos.

3.2.5 Componente de Interpretação

A máquina de interpretação semântica da abordagem S-Chart é operacionalizada por dois algoritmos que, juntos, realizam a extração de feições simbólicas do sinal por geração e teste de hipótese. O seu funcionamento é baseado na manipulação dos modelos de conhecimento visual através dos seus três níveis semânticos. Eles também fazem uso dos detectores simbólicos e dos mapeamentos entre instâncias do componente de ancoramento descrito anteriormente, além de acionar os algoritmos de processamento analógico do sinal.

A arquitetura da máquina de interpretação é demonstrada na Figura 3.9. A sua entrada é composta pelo sinal digitalizado e um conceito do domínio que deve ser encontrado no sinal. Esse conceito faz o papel de hipótese inicial, uma vez que a leva em conta também suas subclasses e subpartes. O resultado final da interpretação são instâncias de conceitos do domínio relacionados à hipótese inicial de busca.

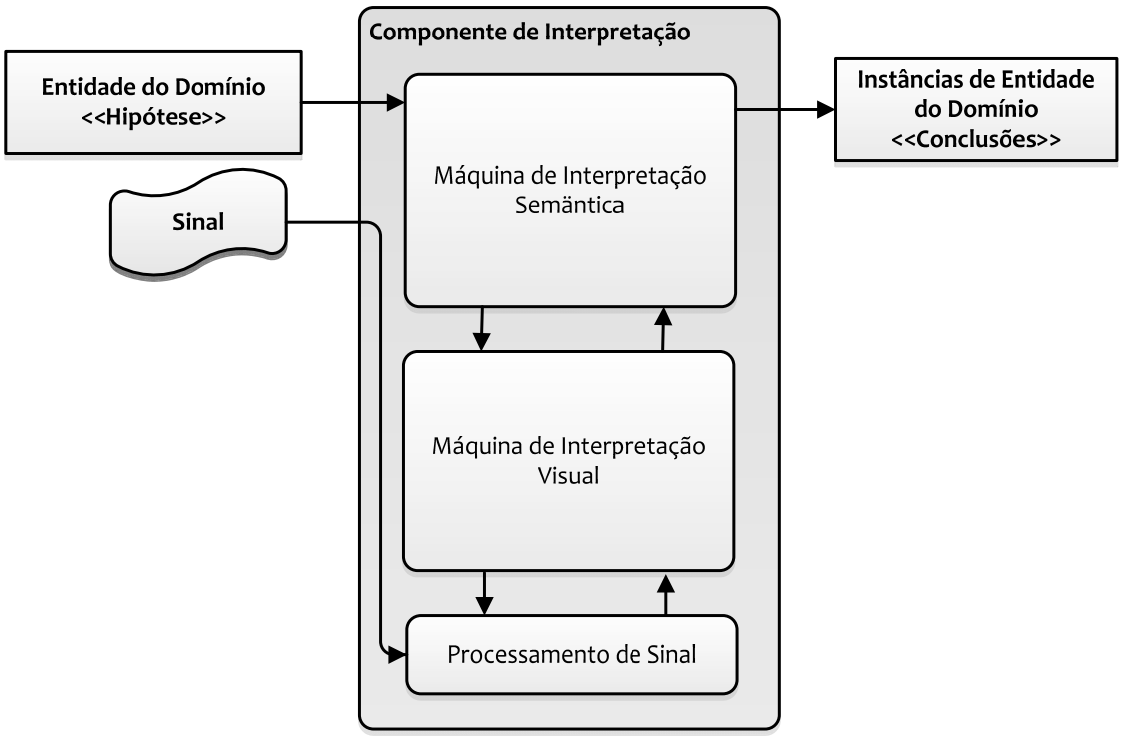

Figura 3.9: Arquitetura do componente de interpretação.

Essa arquitetura se divide em dois componentes básicos, conforme mostra a Figura 3.9. A *máquina de interpretação semântica* (MIS) operacionaliza o ancoramento dos conceitos do nível semântico para o nível visual. Ela tenta confirmar a presença de conceitos no domínio com base na suas extensões no nível visual, definidos por detectores simbólicos. Ao mesmo tempo, a MIS coloca a cargo da *máquina de interpretação visual* (MIV) a extração das primitivas visuais diretamente do nível analógico. O seu papel é acionar os algoritmos de processamento de sinal para extração de feições analógicas no sinal dado como entrada no componente de interpretação e interpretá-los em termos de primitivas visuais via os seus próprios detectores simbólicos.

A representação dos algoritmos neste trabalho requer algumas explicações. Ambos os algoritmos utilizam informações de contexto durante raciocínio, como os modelos de conhecimento visual, instâncias do modelo e o sinal propriamente dito. O contexto é proporcionado por uma estrutura `Contexto` definida na Tabela 3.3. Essa estrutura é utilizada para armazenar os resultados das inferências e troca de informações entre as máquinas MIS e MIV.

Além disso, em alguns lugares, os algoritmos se referem a primitivas de representação de conhecimento do OWL (na forma `prefixo:primitiva`). Essas

primitivas se organizam em uma ontologia de representação de conhecimento (ver Anexo).

Tabela 3.3: Variáveis de contexto para interpretação semântica

`Contexto.NivelSemântico.TBox`	Conjunto de primitivas do nível semântico
`Contexto.NivelSemântico.ABox`	Conjunto de instâncias de primitivas do nível semântico
`Contexto.NívelVisual.TBox`	Conjunto de primitivas do nível visual
`Contexto.NívelVisual.ABox`	Conjunto de instâncias de primitivas do nível visual
`Contexto.NívelAnalógico.Box`	Conjunto de primitivas do nível analógico
`Contexto.NívelAnalógico.ABox`	Conjunto de instâncias de primitivas do nível analógico
`Contexto.Sinal`	Representação digital do sinal analisado

Nas seções seguintes, as máquinas MIS e MIV são descritas em maior detalhe.

3.2.5.1 Máquina de interpretação semântica (MIS)

A MIS busca combinar feições visuais extraídas do nível analógico para detectar entidades semânticas que tenham expressão visual. A sua entrada corresponde a um conjunto de conceitos do nível semântico que indica uma hipótese inicial de entidade a ser extraída. As saídas correspondem às instâncias de entidades do domínio efetivamente encontradas no sinal. O processo de interpretação visual é operacionalizado pelo Algoritmo 3.1.

Mais especificamente, a sua entrada corresponde a um conjunto de classes do nível semântico (no caso, classes do tipo `owl:Class`) em `hipoInicial`. Elas correspondem às primitivas do nível semântico que formam as hipóteses de entidades presentes no sinal. O primeiro laço do algoritmo (linhas 5-8) verifica se todas as primitivas de `hipoInicial` são mesmo primitivas do nível semântico.

O segundo laço (linhas 12-14) busca por todos os conceitos relacionados às primitivas iniciais através da função `buscaRelacionados(owl:Class primitiva)`. Essa função busca o fecho transitivo de todas as subclasses e subpartes de `primitiva`. O laço resulta no conjunto `hipoCompleta` de todas as primitivas que podem ser encontradas no sinal.

O terceiro laço (linhas 22-30) separa as primitivas de `hipoCompleta` que apresentam uma expressão visual no nível visual. Isso é realizado buscando por primitivas do domínio que estão relacionadas a conjuntos de detectores simbólicos pela relação de mapeamento `extraídoPorDS`. Os detectores simbólicos de cada conjunto são tomados um a um para processamento (linha 24). Eles são agrupados no conjunto `detectoresSimbólicosSelecionados` para posterior execução. A processo de separação das primitivas visuais (linhas 26-29) consiste em procurar no antecedente de cada detector simbólico átomos que representem primitivas pertencentes ao nível visual. As primitivas encontradas (conceitos ou propriedades visuais) são separadas no conjunto `primitivasVisuaisEsperadas`.

As primitivas visuais esperadas são então encaminhadas para processamento na `interpretaçãoAnalógica` (linha 35). O resultado da interpretação é composto de instâncias de primitivas visuais que podem ou não ser significan-

tes para interpretação. Após, cada uma dessas mesmas instâncias são mapeadas em um única instância de `EntidadeSmântica` através do procedimento `mapVisualSemântico()`, descrito no Algoritmo 3.2.

O procedimento `executaDetectoresSimbólicos(swrl:Imp cds{})` aplica as regras que constituem os detectores simbólicos selecionados sobre o modelo de conhecimento simbólico (modelos mais instâncias), realizando a inferência propriamente dita. Uma vez que os detectores são formalizados em na linguagem SWRL, essa função pode corresponder à execução de uma ferramenta de interpretação de regras SWRL. O resultado será a classificação das instâncias de `EntidadeSemântica` em classes do domínio mais específicas. Ao fim, instâncias do nível semântico não classificadas como alguma entidade do domínio – portanto instâncias somente da classe `EntidadeSemântica` – são removidas no ultimo laço do algoritmo (linhas 45-50).

O resultado de todo o processo de interpretação visual pode ser encontrado em `Contexto.NívelSemântico.Inst`. No entanto, por conveniência, a função de interpretação retorna diretamente as instâncias resultantes do algoritmo.

```
1  interpretaçãoSemântica (owl:Class hipoInicial{}) : EntidadeSemântica{}
2
3     //verifica se todas os conceitos de hipotese
4     //pertencem ao domínio
5     owl:Class hipoInicialVerificada{};
6     para-cada c ∈ hipoInicial faça
7        se (c ∈ Contexto.NivelSemântico.TBox) então
8           hipoInicialVerificada = hipoInicialVerificada ∪ c;
9
10
11    //busca todas os conceitos relacionados com as hipoteses inicial
12    owl:Class hipoCompleta{};
13    para-cada c ∈ hipoInicialVerificada faça
14       hipoCompleta = hipoCompleta ∪ buscaRelacionadosSemântico(c);
15
16    //seleciona quais entidades do domínio têm expressão visual,
17    //ou seja, quais são mapeadas por detectores simbólicos
18    //e quais primitivas visuais (classes e propriedades) são utilizadas
19    owl:Class entidadesMapeadas{};
20    swrl:Imp detectoresSimbólicosSelecionados{};
21    rdf:Resource primitivasVisuaisEsperadas{};
22    para-cada c ∈ hipoCompleta faça
23       se :extraídoPorDS(c, DS) então
24          para-cada r ∈ DS faça {
25            detectoresSimbólicosSelecionados = detectoresSimbólicosSelecionados ∪ r;
26             rdf:Resource atomosC = r.antecedente;
27             para-cada atomo ∈ DS faça
28                se atomo ∈ Contexto.NivelVisual.TBox então
29                    primitivasVisuaisEsperadas = primitivasVisuaisEsperadas ∪ atomo;
30          }
31
32    //aciona o componente de interpretação analógica
33    // as entidade visuais resultantes são acessiveis
34    // em Contexto.NivelVisual.Inst
35    interpretaçãoAnalógica(primitivasVisuaisEsperadas);
36
37    //mapeia cada entidade visual em Contexto.NivelVisual.Inst
38    //como uma instancia de :EntidadeSemântica em Contexto.NivelSemântico.Inst
39    mapVisualSemântico();
```

```
40
41    //executa detectores simbólicos das entidades semânticas separados anteriormente
42    executaDetectoresSimbólicos(detectoresSimbólicosSelecionados);
43
44    //remove instâncias não classificadas, ou seja, hipoteses que não se confirmaram
45    EntidadeSemântica resultado{};
46    para-cada es ∈ Contexto.NívelSemântico.ABox faça
47       se ∀c(rdf:type(es, c) ∧ c ≡ EntidadeSemântica) então
48          Contexto.NívelSemântico.ABox = Contexto.NívelSemântico.ABox − es
49       senão
50          resultado = resultado ∪ es;
51    //para conveniencias, retorna entidades classificadas
52    //elas podem ser acessadas fora do algortimo em
53    retorna resultado;
54 }
```

Algoritmo 3.1: função de operação da máquina de interpretação semântica

```
55 mapVisualSemântico (){
56
57    //agrupa instâncias de EntidadeVisuais
58    :EntidadeVisual entidadesDetectadas{};
59    para-cada inst ∈ Contexto.NivelVisual.ABox faça
60       se owl:SubClassOf(inst, :EntidadeVisual) então
61          entidadesDetectadas = entidadesDetectadas ∪ r;
62
63    //cria uma EntidadeSemântica para EntidadeVisual não mapeada
64    para-cada iev ∈ entidadesDetectadas faça
65       se !(∃x(mapeamentoSemânticoParaVisual(x, iev))) então {
66          EntidadeSemantica novaEntidade = novaInstancia(:EntidadeSemantica);
67          Contexto.NivelVisual.ABox = Contexto.NivelVisual.ABox
68             ∪ novaEntidade ∪ mapeamentoSemânticoParaVisual(novaEntidade, iev);
69       }
70 }
```

Algoritmo 3.2: procedimento de mapeamento visual/semântico

3.2.5.2 Máquina de interpretação visual (MIV)

A máquina MIV é executada a partir de requisições da máquina MIS. Ela recebe como entrada um conjunto de primitivas visuais que possivelmente existem no sinal e tenta confirmá-las através dos seus detectores simbólicos mapeados em primitivas do nível analógico. A MIV se encarrega ainda de acionar os mecanismos de processamento de sinal para extração das características analógicas do sinal. Embora o algoritmo não tenha uma saída propriamente dita, os resultados da sua execução (instâncias de `EntidadeVisual`) são alocados em `Contexto.NívelVisual.Abox` para que a maquina MIS tenha acesso. O algoritmo que operacionaliza a máquina MIV é descrito pelo Algoritmo 3.3.

O primeiro laço (linhas 4-6) busca por primitiva relacionadas na taxonomia onde elas estão inseridas. Essa entidade pode ser uma classe ou uma relação. Mais especificamente, a função `buscaRelacionadosVisual(rdf:Resource res)` retorna o conjunto de todas as subclasses de `res` e todas as suas superclasses, caso `res` seja uma classe. Se `res` é uma relação, então o retorno da função é um conjunto com todas as sub-relações e super-relações de `res`.

Com a hipótese visual completa, busca-se por detectores simbólicos associados às primitivas visuais em `hipoVisualCompleta` (linhas 12-24). De forma

semelhante ao que ocorre do terceiro laço do algoritmo de interpretação semântica, o algoritmo analisa os detectores simbólicos encontrados em busca de primitivas analógicas referenciadas (classes ou relações). Essas entidade são guardadas no conjunto `primitivasAnalógicasEsperadas`.

As linhas 28-34 operacionalizam a extração das primitivas analógicas esperadas. Cada uma das primitivas é repassada ao componente de processamento de sinal pela função `extrairEntidadeAnalógica(rdf:Resource p)`. Essa função deve acionar um algoritmo de processamento de sinal (armazenada na estrutura `Contexto.Sinal`) específico para extração da primitiva `p`. As instâncias de primitivas extraídas devem ser guardadas em `Contexto.NívelSemântico.ABox`. A função retorna a quantidade de instâncias extraídas. De fato, a implementação dessa funcionalidade pode corresponder a um componente de software complexo por si só. Uma vez que essa função não depende de aspectos de representação e interpretação de conhecimento visual, a sua descrição detalhada foge do escopo deste trabalho e é omitida. No entanto, no estudo de caso descrito no Capítulo 5 é apresentada uma versão simplificada – mas efetiva – para ela. O laço é executado até que nenhum novo resultado seja gerado. Isso garante que possíveis relações de dependência entre primitivas não influenciem nos resultados.

O procedimento `mapAnalógicoVisual()` executa função semelhante ao procedimento descrito em Algoritmo 3.3, porem dessa vez mapeando instâncias de `EntidadeAnalógica` para instâncias de `EntidadeVisual`. Já o procedimento `executaDetectoresSimbólicos(swrl:Imp cds{})` executa os detectores simbólicos de entidade visuais para classificá-las em primitivas mais específicas. Por fim, o algoritmo retorna o controle da execução para a máquina MIS, sendo que o resultado da execução está armazenado em `Contexto.NivelVisual.TBox`.

```
1  interpretaçãoVisual(rdf:Resource hipoVisualInicial{}){
2
3     //busca todas os conceitos relacionados com as hipoteses inicial
4     rdf:Resource hipoVisualCompleta{};
5     para-cada c ∈ hipoVisualInicial faça
6        hipoVisualCompleta = hipoVisualCompleta ∪ buscaRelacionadosVisual(c);
7
8
9     //seleciona as entidades visuais que realmente têm expressão analógica,
10    //ou seja, quais são mapeadas por detectores simbólicos
11    //e quais primitivas analógicas (classes e propriedades) são utilizadas
12    rdf:Resource entidadesMapeadas{};
13    swrl:Imp detectoresSimbólicosSelecionados{};
14    rdf:Resource primitivasAnalógicasEsperadas{};
15    para-cada c ∈ hipoVisualCompleta faça
16       se :extraídoPorDS(c, DS) então
17          para-cada r ∈ DS faça {
18            detectoresSimbólicosSelecionados = detectoresSimbólicosSelecionados ∪ r;
19             rdf:Resource atomosC = r.antecedente;
20             para-cada atomo ∈ DS faça
21                se atomo ∈ Contexto.NívelAnalógico.TBox então
22                     primitivasAnalógicasEsperadas = primitivasAnalógicasEsperadas
23                            ∪ atomo;
24          }
```

```
25
26    //Aciona os algoritmos de processamento de sinal para extração
27    // de entidades analógicas esperadas
28    Inteiro resultado;
29    faça{
30       resultado = 0;
31       para-cada p ∈ hipoVisualCompleta faça
32          resultado = extrairEntidadeAnalógica(p);
33
34    } enquanto resultado ≠ 0;
35
36    //mapeia cada EntidadeAnalógica em Contexto.NivelAnalógico.Inst
37    //como uma instancia de EntidadeVisual em Contexto.Nivelvisual.Inst
38    mapAnalógicoVisual();
39
40    //executa detectores simbólicos das entidades visuais separados anteriormente
41    executaDetectoresSimbólicos(detectoresSimbólicosSelecionados);
42 }
```

Algoritmo 3.3: procedimento de operação da máquina de interpretação visual

3.2.5.3 Considerações sobre ancoramento simbólico

O funcionamento geral do componente de interpretação pode ser classificado com uma abordagem mista de ancoramento simbólico. Ele operacionaliza o raciocínio *top-down*, ao restringir as possíveis interpretações a partir da hipótese inicial informada. Somente os detectores simbólicos relacionados às hipóteses iniciais – por relações de taxonomia ou partonomia – serão acionados. Ou seja, a geração de hipóteses no "topo" da arquitetura é restrita pela estrutura do modelo de domínio e pela hipótese inicial. O caminho até a "base" é realizado através do processamento dos detectores simbólicos.

Por outro lado, o processo *bottom-up* é iniciado no momento que a máquina MIV extrai informações sensórias ao acionar os algoritmos de processamento de sinal. As informações extraídas são propagadas aos níveis superiores aplicando-se os detectores simbólicos previamente selecionados no processamento *top-down*.

4 ESTRATIGRAFIA DE SEQUÊNCIAS

Este capítulo descreve brevemente o domínio de aplicação deste trabalho, a Estratigrafia de Sequências, sub-área da Geologia. Os conceitos apresentados aqui suportam a discussão sobre o sistema InteliStrata, apresentado no capítulo seguinte. O sistema InteliStrata é uma aplicação do framework S-Chart para interpretação de gráficos no domínio da Estratigrafia de Sequências.

O foco da discussão sobre a Estratigrafia de Sequencias feita aqui está na tarefa interpretação estratigráfica, que consiste na identificação de feições geológicas em perfis de poço, mais especificamente, perfis de raios gama, como o representado na Figura 4.1. A identificação das feições baseia-se na busca por padrões específicos na curva de raio gama e nas suas inter-relações espaciais.

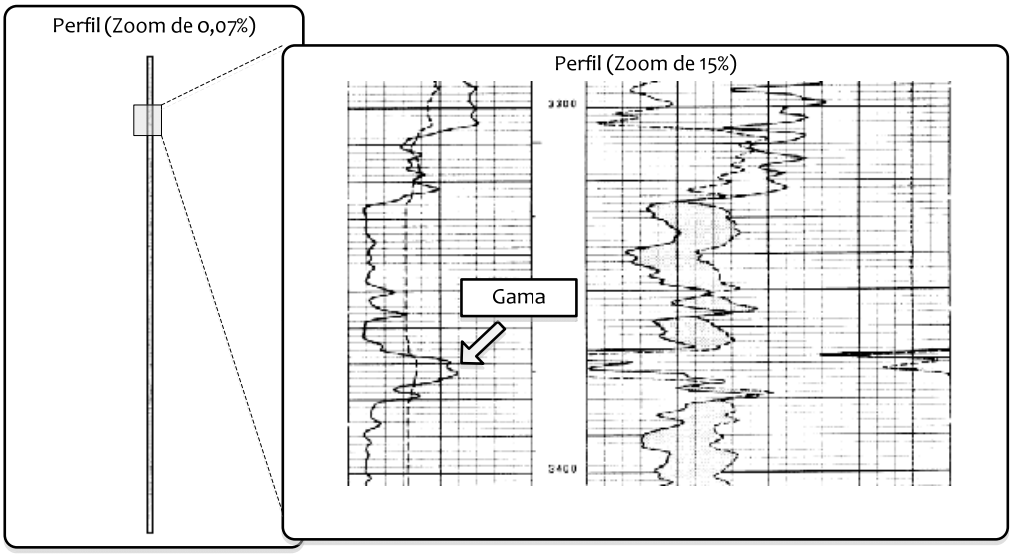

Figura 4.1: Perfil típico em diferentes escalas. Raios gama em destaque.

Inicialmente, será apresentado o domínio da Estratigrafia de Sequências, A seguir o foco é direcionado a interpretação estratigráfica. Por fim, é discutida a eficácia da utilização de wavelets para análise básica de perfis de raios gama.

4.1 Apresentação do domínio

A Estratigrafia de Sequências é uma área de Geologia que estuda o histórico de formação dos estratos que compõem o subsolo terrestre em termos de variações do nível do mar e taxas de sedimentação. Nessa área, diferentes sequências

de estratos indicam processos de formação distintos. O correto entendimento destes processos é importante para diversas outras áreas, especialmente para a Geologia de Petróleo, onde a interpretação dos processos que controlaram a formação de um reservatório define as estratégias de exploração e produção de óleo.

O objetivo principal do sistema InteliStrata é sugerir interpretações de *sequências*. Uma sequência corresponde a um ciclo completo de descida e subida do nível do mar. Em geral, esse fenômeno afeta uma região localizada (800 a 80 mil quilômetros quadrados) e pode levar de 100 mil a 5 milhões de anos para se completar. Os diversos momentos de um ciclo influenciam diferentemente na ordem de deposição dos sedimentos na região onde ele acontece. Por exemplo, à medida que o nível do mar sobe em um dado local, ocorre a deposição de um sedimento mais lamoso e menos poroso. Quando o nível do mar desce no mesmo local, ocorre a deposição de um sedimento mais grosso, formando uma rocha mais porosa. De fato, variações de porosidade da rocha têm uma grande correlação com variações no nível do mar. Dessa forma, estudando-se as variações de porosidade ao longo da profundidade do subsolo terrestre, é possível se ter uma ideia aproximada dos ciclos variação do nível do mar naquela região, e logo, das sequências.

A interpretação de sequências é feita com base em dados coletados em poços de sondagem com até 8000 metros de profundidade, perfurados ao longo de uma região de interesse. Desses poços, dois tipos básicos de dados são coletados: testemunhos e perfis de poço. Durante a perfuração do poço, um tipo especial de broca extrai amostras de rochas na forma de pequenos cilindros. Estas amostras dão ao geólogo a possibilidade de analisar diretamente a rocha que constitui o terreno. Ele descreve precisamente as características de cada estrato da amostra (como tipo de rocha, porosidade, tamanho de grão, estrutura e etc.) e as organiza visualmente em uma coluna estratigráfica. Com base na análise visual da coluna estratigráfica, o geólogo consegue identificar visualmente os padrões de empilhamento de sedimentos e, a partir daí, identificar a sequência com um grande grau de precisão. Esse processo é feito em diversos poços, de forma que, após as informações serem correlacionadas, é possível identificar como uma mesma sequência afeta toda uma região.

No entanto, o processo de perfuração com testemunhagem é consideravelmente caro. Durante o estudo de uma região, poucos poços são testemunhados. Os demais são analisados através de perfilagem sísmica (elétrica, raios gama, entre outras). Após ser perfurado, o poço é perpassado por uma sonda ao longo de toda a sua profundidade. Durante a descida, a sonda faz diversos tipos de medidas da rocha a sua volta, em intervalos de aproximadamente 20 cm (variando conforme a medida). E resultado é um perfil do poço, um gráfico que demonstra a variação das diversas medidas ao longo da sua profundidade. Existem vários tipos de medidas. Cada uma correlacionada com alguma característica. Padrões de variação nessas medidas refletem diferentes tipos de características do terreno. Ou seja, os perfis de poço servem como um instrumento indire-

to de análise do terreno, quando, por exemplo, a análise direta da rocha pelo testemunho não é realizada. Um dos tipos de medida para interpretação de sequências é o perfil de raios gama. De forma simplificada, ele mede a radiação gama emitida pelas rochas de um poço. Através da radiação é possível inferir a porosidade de uma rocha. Em geral, quanto maior a leitura de raios gama, menor é a porosidade e vice-versa. Como dito anteriormente, a porosidade da rocha tem uma grande correlação com a ocorrência de sequências, por isso, geólogos utilizam o perfil de raios gama como uma ferramenta para identificá-las.

Em geral, a interpretação de sequências combina a análise de testemunhos com a análise de perfis. No entanto, por ser de custo inferior ao da testemunhagem, freqüentemente o perfil é o único dado disponível, sendo necessário extrair interpretações geológicas úteis somente a partir dos perfis. A identificação dos tipos e limites de sequências é uma tarefa que requer ao geólogo um treinamento razoável em Estratigrafia. Mesmo quando domina o conhecimento necessário, a análise de perfilagens com centenas de metros de profundidade mostra-se uma atividade manual morosa e consumidora de tempo. Essa é a motivação do sistema InteliStrata. Ele deve identificar sequências possíveis a partir de um perfil de raios gama e sugeri-las para o usuário. Outra vantagem desse sistema é que ele possibilita ao geólogo estratigráfico economizar tempo, sugerindo uma interpretação inicial.

Na seção seguinte, a interpretação de sequência por perfil de raios gama é detalhada e é demonstrada a efetividade de um algoritmo de processamento para identificação de sequências e parassequências.

4.2 Interpretação de sequências em perfis de raios gama

A identificação de sequências em perfis de raios gama é um raciocínio predominantemente visual. O geólogo especialista analisa o gráfico do perfil visualmente, buscando padrões de variação na curva que casem com padrões de variação conhecidos e que indicam a presença de determinadas feições geológicas. Além disso, a análise visual é condicionada pelo contexto e conhecimento prévio a respeito da geologia da região, visto que o padrão visual de uma feição geológica pode mudar conforme a bacia sedimentar onde ela se insere.

Existem inúmeras feições geológicas que podem ser interpretadas em um perfil. Neste trabalho, o foco é na interpretação de três tipos de feições: sequências, parassequências e superfícies de inundação máxima.

Sequências, como mencionando anteriormente, são formadas por grandes ciclos de variação do nível do mar, que influenciam na formação dos estratos de sedimentos. Em um perfil de raios-gama, uma sequência tem a expressão semelhante a uma curva gaussiana (Figura 4.2a): um movimento de subida e descida do nível do mar relativamente suaves. Uma sequência pode ter, no perfil, uma espessura entre 15 e 1500 metros aproximadamente.

Já parassequência são, simplificadamente, pequenos ciclos locais de variação no nível do mar dentro de uma sequência. Podem ser vistas como um ciclo de

menor ordem. No perfil, uma parassequência se caracteriza como uma curva que inicia por uma variação negativa abrupta e uma recuperação suave até o mesmo nível (Figura 4.2b). A sua espessura está em uma escala inferior à sequência, variando entre aproximadamente 3 e 60 metros.

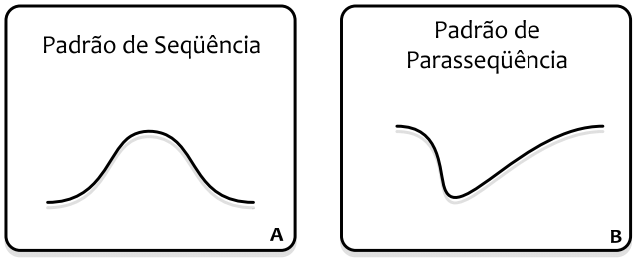

Figura 4.2: Expressões genéricas das feições geológicas no perfil.

A superfície de inundação máxima é o ponto de uma sequência onde o nível do mar atinge o seu ponto máximo e começa a baixar. No perfil, ele é facilmente identificado como o ponto mais alto da curva que forma a sequência.

O relacionamento entre essas três feições é utilizado para refinar a interpretação. Por exemplo, parassequências geralmente estão inseridas em uma sequência. Logo, a interpretação de uma parassequência é realizada após a identificação dos limites da seqüência onde ela se insere. Da mesma forma, a escala da análise é determinante para a correta identificação de sequências ou parassequências, uma vez que a assinatura das duas feições é muito similar, exceto pelo contexto e escala em que ocorrem.

A máquina de interpretação do sistema InteliStrata depende de um algoritmo de processamento de sinal que faça a segmentação inicial do perfil, de acordo com a escala onde as feições a serem interpretadas ocorrem e as feições buscadas. A seguir é analisada uma proposta de segmentação por *wavelets*, baseado na detecção dos padrões de curva que formam as feições geológicas procuradas.

4.3 Análise de perfis por *wavelets*

Até a data de escrita deste trabalho, não existe nenhuma solução automática para interpretação de sequências descrita na literatura. Da mesma forma, nenhum sistema estratigráfico atualmente no mercado oferece essa facilidade. Por outro lado, alguns autores exploram a utilização de técnicas matemáticas e computacionais, em especial processamento de sinal, para auxilio da interpretação de fenômenos cíclicos em dados de perfil. Enquanto alguns autores focam na análise da transformada de fourier (SILVA, 2001), algumas propostas exploram a possibilidade da análise espectral por *wavelets* ser uma boa ferramenta para detecção manual de sequências (CHOUDHURY et al., 2007).

Do ponto de vista empírico, o uso de *wavelets* é a opção mais interessante. A análise de sinais via transformada de Fourier é eficaz somente em sinais periódicos. Ao contrário disso, sequências – e as demais feições geológicas aqui consideradas – são não-periódicas quanto a sua ocorrência em um perfil, onde co-

existem em diferentes escalas e freqüências. Exatamente neste ponto que a transformada *wavelet* leva vantagem. A transformada *wavelet* contínua (TWC) responde a essas variações de ciclos em diferentes escalas e freqüências.

Conceitualmente, o processamento da TWC é relativamente direto. O espectro resultante de um TWC pode ser visto como uma resposta de similaridade de um determinado pulso de onda periódica com todas as partes de um sinal, em várias escalas. As diferentes wavelets são caracterizadas pelo formato de pulso aplicado na transformada. Por exemplo, a Figura 4.3 representa o pulso utilizado pela *wavelet* gaussiana 2. Já o espectro resultante de um TWC tem o mesmo comprimento do sinal e altura correspondente ao número de escalas definidas para análise pelo usuário. Ele demonstra a resposta do sinal em comparação ao pulso da *wavelet*. Em geral, partes do sinal que se assemelham a *wavelet* têm uma resposta maior a partes que não se assemelham. Esse comportamento torna a TWC uma ferramenta interessante para reconhecimento de padrões, se o padrão buscado puder ser codificado como uma *wavelet*.

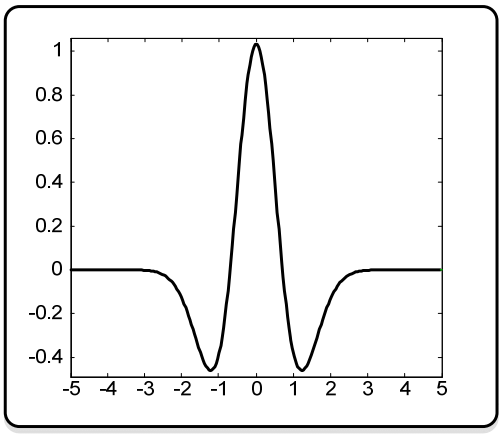

Figura 4.3: Wavelet gussiana 2.

Com base nesta capacidade, este trabalho propõe a utilização de duas TWC para processamento inicial do sinal que compõem os gráficos dos perfis de raios gama. Os algoritmos propostos tomam o papel dos algoritmos de processamento de imagem convencionais. O objetivo é segmentar o sinal em intervalos, conforme a resposta de similaridade dada pelo TWC. A escolha das *wavelets* utilizadas se deu através de testes empíricos de resposta aos padrões de curva desejados. Um estudo mais sistemático de diferentes *wavelets* seria necessário para se garantir que a opção utilizada garante o melhor resultado na segmentação, mas a seleção do algoritmo de processamento de imagem é secundária para a contribuição buscada neste trabalho.

Os testes foram realizados sobre um perfil já interpretado pelo especialista e que cobre aproximadamente 340 metros. O perfil digitalizado contém 16.465 amostras, resultando em um período de amostragem de aproximadamente 0,018 metro. A seguir as duas TWC testadas são detalhadas.

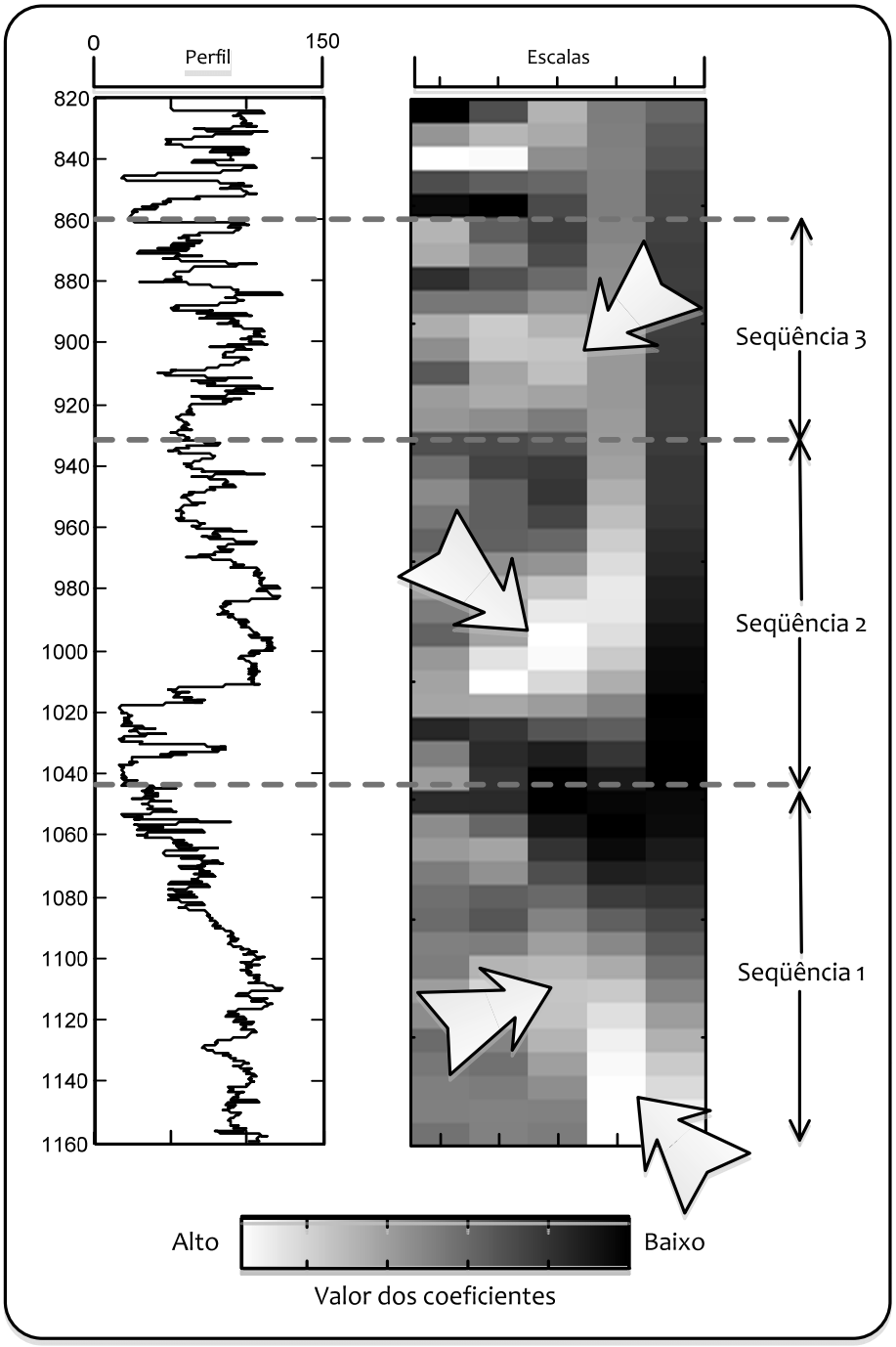

Figura 4.4: Transformada *wavelet* gaussiana 2 em um perfil de raios gama.

4.3.1 Detecção de sequências: wavelet gaussiana

Como descrito anteriormente, uma sequência se apresenta no perfil de forma semelhante a uma curva gaussiana. Isso induz a ideia de que a TWC com a wavelet gaussiana possa ser eficiente na detecção de sequências em perfis. Para demonstrar isso, o perfil de teste foi analisado através da TWC disponível no software Matlab (MATHWORKS INC.). O teste é descrito a seguir.

Inicialmente, é considerada a banda de freqüências correspondente ao intervalo de períodos típico de sequências. Como mencionado anteriormente, uma

sequência pode ter um período que varia entre 15 e 1500 metros de espessura no perfil. Para simplificar a TWC, o perfil é re-amostrado para uma freqüência 0,013 amostras por metro, correspondendo a um período de amostragem de 7,62 metros aproximadamente. Isso minimiza a presença de ciclos com menos de 15 metros no perfil (ruídos de alta frenqüência).

A TWC é aplicada sobre o sinal resultante, agora com 44 das 16.465 amostras anteriores. Os parâmetros utilizados para a TWC são os seguintes:

- **Wavelet:** Gaussiana número 2 (disponível no pacote de wavelets do Matlab), escolhida com base na sua semelhança com o padrão de curva de uma sequência;
- **Escalas**: determinadas pela fórmula $s(n) = 2^n$, onde, no caso deste sinal, $n = 0, 2, \cdots, 4$. Essas escalas garantem que o resultado da TWC fique restrito a ciclos com períodos de 15 a 1500 metros.
- **Extensão:** simétrica em ambos os lados do sinal, a fim de minimizar as distorções de borda.

O resultado da aplicação dessa *wavelet* sobre o perfil é demonstrado na Figura 4.4. À direita, o espectro de coeficientes resultando da TWC demonstra onde o padrão de curva gaussiana está mais evidente (regiões mais claras). A linha pontilhada marca a interpretação de limites de sequência conforme feita pelo especialista sobre o perfil de teste (esquerda). É possível notar que os picos no espectro (setas) correspondem a aproximadamente ao centro das sequências, e os seus correspondentes vales as bordas de sequência. Isso é mais evidente nas sequências de maior escala, como a Sequência 2 e, em certa medida, a Sequência 1. Isso demonstra a boa resposta da *wavelet* gaussiana 2 ao padrão de curva gaussiano definido para sequência.

Com base nesse resultado, este trabalho considera que a TWC é uma ferramenta suficiente para processamento inicial do perfil. Ela identifica padrões correspondentes a sequências, que podem ser utilizados para alimentar os algoritmos simbólicos no sistema InteliStrata. No entanto, os algoritmos simbólicos devem garantir que ruídos sejam minimamente propagados no processo de interpretação (especialmente ruídos de alta freqüência).

4.3.2 Detecção de parassequências: wavelet smoothtooth

A detecção de parassequências por *wavelets* se dá de forma semelhante à detecção de sequências, porém com algumas diferenças. Inicialmente, o padrão de curva de uma parassequência é diferente de uma gaussiana, como demonstrado na Figura 4.2 anterior. Além disso, essa curva não tem correspondente entre as *wavelets* da biblioteca padrão. Por isso, este trabalho propõe uma nova wavelet específica pare detecção deste padrão de curva. Ela foi criada dentro da função de criação de *wavelets* do aplicativo Matlab, seguindo o perfil de uma parassequência padrão, e batizada de *wavelet smoothtooth*, devido a semelhança do padrão original com um período da onda periódica padrão dente-de-serra.

Neste teste, o sinal foi reamostrado para o período de 1,5 metros, conforme o período mínimo desejado. O resultado é um sinal com 217 amostras. Além disso, os parâmetros utilizados para a TWC são:

- **Wavelet:** Smoothtooth;
- **Escalas**: determinadas pela fórmula $s(n)=2^n$, onde $n=1,2,\cdots,5$. Essas escalas garantem que o resultado da TWC fique restrito a faixa de ciclos de parassequências, com períodos de 3 a 60 metros.

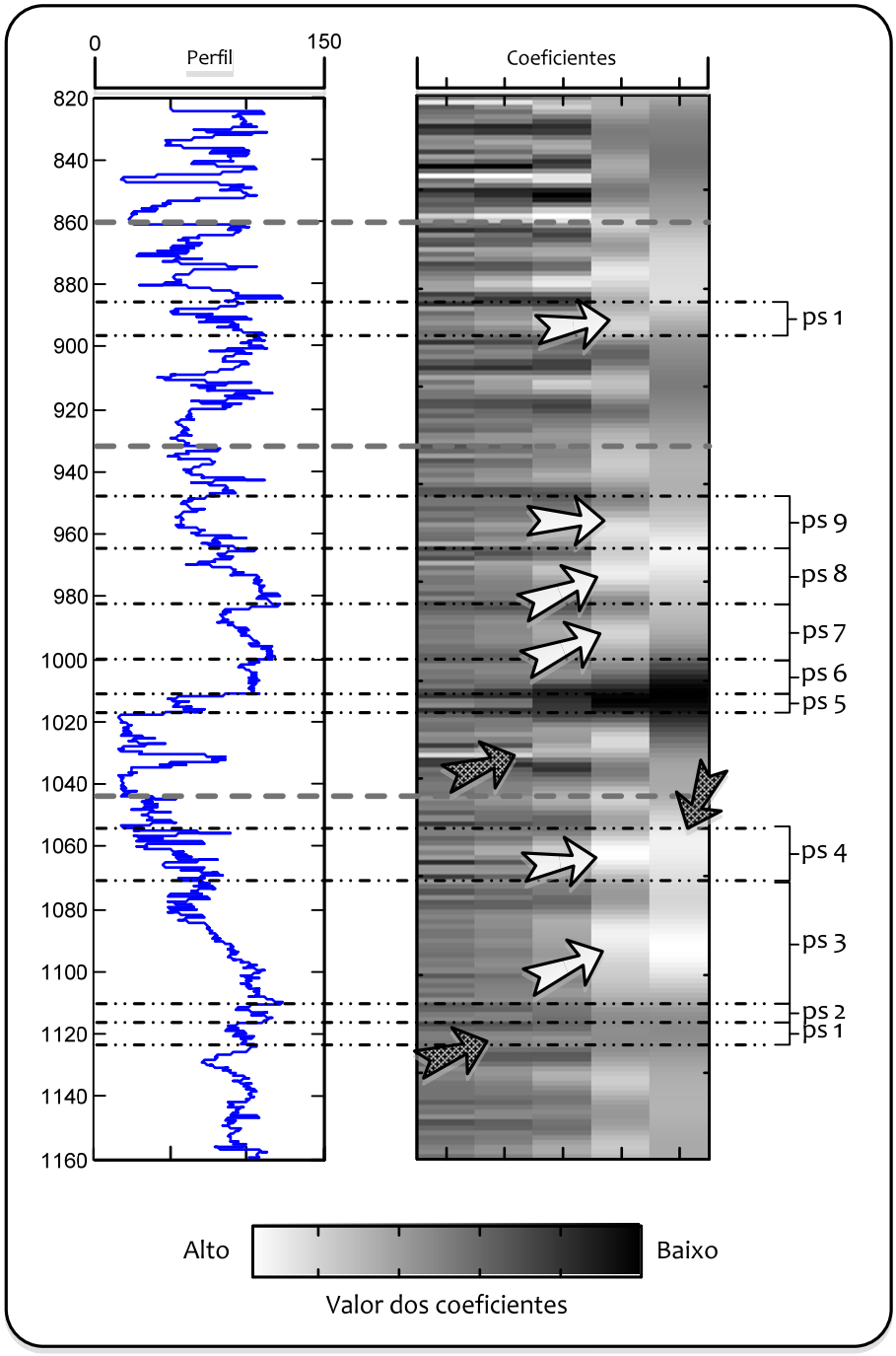

Figura 4.5: Transformada wavelet sawtooth em um perfil de raios gama.

- **Extensão:** também simétrica em ambos os lados do sinal, a fim de minimizar ruídos de borda.

Os resultados na aplicação da TWC com essa *wavelet* em um perfil é demonstrado na Figura 4.5. As setas claras marcam os picos que correspondem aproximadamente às parassequências determinadas pelo especialista (ps). Embora parte delas tenha sido discriminada, algumas não o são, como é o caso ps1, onde não aconteceram picos (seta escura inferior). Há também falsos positivos, como as marcadas pelas duas setas escuras superiores. Isso ocorre devido a escala relativamente pequena das parassequências. Uma vez que elas aparecem em freqüências maiores, elas se confundem com ruídos de alta freqüência do perfil. Ainda sim, pode-se concluir que a wavelet smoothtooth pode ser utilizada para auxiliar na extração de parassequências do perfil.

5 SISTEMA INTELISTRATA

Este capítulo detalha a implementação do sistema InteliStrada, uma aplicação do framework S-Chart para interpretação de gráficos no domínio da Estratigrafia de Sequências O objetivo do sistema InteliStrata é, dado como entrada um perfil de raios gama, fornecer ao usuário uma sugestão de interpretação de sequências, parassequências e superfícies de inundação máxima.

Naturalmente, o sistema InteliStrada complementa a definição do framework S-Chart do ponto de vista da implementação. Nas seções seguintes, serão explorados aspectos da infra-estrutura de software necessária para operacionalizar os conceitos apresentados no capítulo 3. Além disso, são também explorados aspectos de implementação específicos da interpretação estratigráfica.

5.1 Objetivos do sistema

O objetivo do sistema InteliStrata é sugerir interpretações estratigráficas sobre perfis de raios gama. Padrões visuais de comportamento da curva em perfis de raio gama sugerem a presença de sequências, parassequências e superfícies de inundação máxima. O sistema captura e combina esses padrões com conhecimento de domínio, em um processo de interpretação semântica. Mais especificamente, o sistema sugere possíveis instâncias de sequências, parassequências e superfícies de inundação máxima presentes no perfil. Para realizar essa tarefa, o sistema InteliStrata implementa e especializa o framework S-Chart.

Do ponto de vista de sistema, o desenvolvimento do InteliStrata segue dois objetivos principais:

- **Utilização de modelos semanticamente ricos na implementação**. Um dos princípios mais importantes na implementação de sistema intensivos em conhecimento é o princípio da preservação de estrutura (SCHREIBER et al., 1999; FIORINI, 2006). Por ele, o modelo de implementação (arquitetura, modelo de classes e etc.) de um sistema de conhecimento deve buscar manter ao máximo a estrutura dos modelos de conhecimento. Isto facilita a manutenção em um cenário onde os próprios modelos de conhecimento são suscetíveis a mudanças constantes, além de facilitar o trabalho de implementação e manutenção. Em um caso ideal, o modelo de conhecimento deve ser especificado em uma linguagem possível de ser acessada diretamente pelo sistema de conhecimento (como linguagens de representação baseadas em XML). Dessa forma, o código do sistema de

conhecimento pode se restringir ao máximo à implementação dos algoritmos de raciocínios.

- **Emprego de ferramentas *off the shelf***. O sistema InteliStrata é baseado no framework S-Chart, cujos modelos de representação são definidos em linguagens formais de representação de conhecimento, como OWL e SWRL. Uma vez que elas são processáveis por software, existem diversas ferramentas que dão suporte a manipulação, manutenção e raciocínio sobre modelos descritos nessas linguagens. Assim, é possível que estas ferramentas sejam utilizadas como componentes de infra-estrutura para o desenvolvimento de um sistema de interpretação semântica, minimizando o esforço de implementação nessas áreas (como estrutura de dados, por exemplo).

Estes objetivos influenciam diretamente a arquitetura do sistema InteliStrata e sua forma de implementação.

5.2 Arquitetura do sistema

Um visão geral da arquitetura do sistema InteliStrata é apresentada na Figura 5.1. Os componentes que a formam podem ser divididos em três conjuntos: (i) um conjunto de componentes que constituem o sistema de interpretação propriamente dito; (ii) com conjunto de componentes de software para infra-estrutura; (iii) um conjunto de modelos OWL que representam os modelos de conhecimento utilizados pelo software.

Nas próximas seções, cada um desses componentes é detalhado, bem como a sua interrelação.

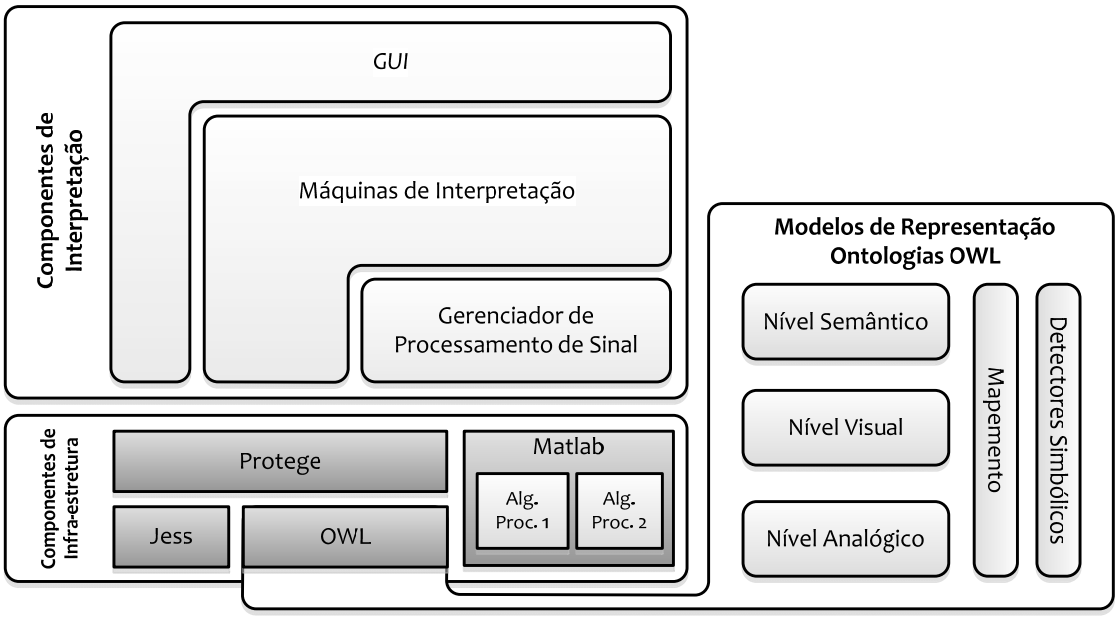

Figura 5.1: Arquitetura de componentes do sistema InteliStrata.

5.2.1 Modelos de representação e extensões

Como apresentado anteriormente, o framework S-Chart define um conjunto de modelos de conhecimento visual para dar suporte a interpretação semântica. Alguns desses modelos devem ser supridos pelo sistema que implementa o framework (como é o caso do modelo de domínio no nível semântico) enquanto outros modelos são disponibilizados pelo próprio framework, com a possibilidade de extensão (modelos no nível visual e analógico).

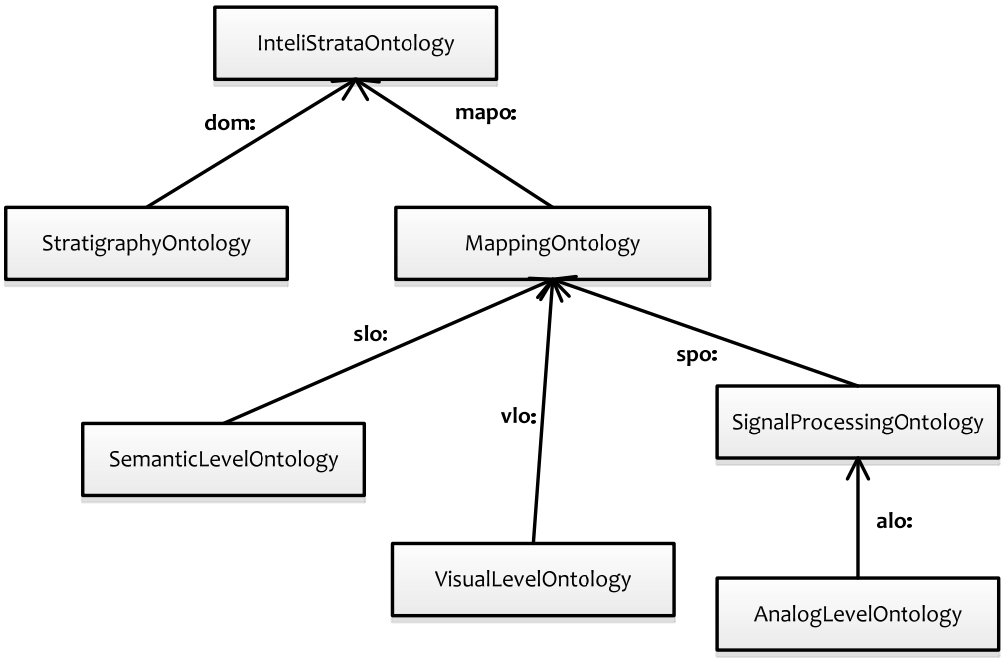

Figura 5.2: Dependências entre ontologias OWL do sistema InteliStrata.

No sistema, cada um dos modelos é descrito em uma ontologia OWL[8]. A Figura 5.2 mostra a relação de dependência entre as ontologias e os seus respectivos prefixos. As ontologias `AnalogLevelOntology`, `VisualLevelOntology`, `SemanticLevelOntology` e `SignalProcessingOntology` modelam, respectivamente, os três níveis semânticos e a ontologia de funcionalidades de processamento de sinal. A `StratigraphyOntology` captura o conhecimento de domínio da estratigrafia de sequências. Todas elas são agregadas em uma ontologia de aplicação chamada `InteliStrataOntology`. A sua função é agregar as demais em um só arcabouço, além de servir de repositório para as extensões realizadas nos modelo de conhecimento visual. Nas seções seguintes, cada uma dessas extensões é detalhada.

[8] Embora o OWL descreva qualquer modelo da sua linguagem como uma "ontologia", claramente o termo não é o mais preciso. Um modelo OWL – e de qualquer outra linguagem – pode ser caracterizado como uma ontologia somente se modela **formalmente** uma **conceitualização compartilhada** (como a definição dada por Gruber e Borst (1998)). No entanto, mantém-se o termo aqui para manter consistência com o vocabulário de OWL.

5.2.1.1 *Ontologia de domínio: Estratigrafia de Sequências*

O conhecimento sobre Estratigrafia que dá suporte a interpretação estratigráfica é capturado por uma ontologia modelada em `StratigraphyOntology`. Uma vez que esta ontologia não existe publicamente, parte do desenvolvimento do sistema InteliStrata focou em, junto com um especialista em Estratigrafia, eliciar o conhecimento de domínio mínimo necessário para realização da interpretação estratigráfica. A principal parte do conhecimento eliciado é modelada na ontologia de domínio (cuja estrutura básica é apresentada na Figura 5.3). Outros conceitos são capturados na forma de extensões do nível visual e na definição dos detectores simbólicos (detalhados a seguir).

Como mencionado anteriormente, o sistema utiliza conhecimento de domínio para extrair instâncias de sequências, parassequências e superfícies de inundação máxima. Estes três tipos de entidades são capturados na ontologia (Figura 5.3). O modelo também apresenta relações de composição entre estas as entidades do modelo, semelhantes à relação de composição existente entre os conceitos *coração* e *pessoa*. Neste modelo, a composição é representada intuitivamente pela relação transitiva `hasPart`. Essa relação será inferida através das relações topológicas entre as entidades visuais que ancoram cada um desses conceitos.

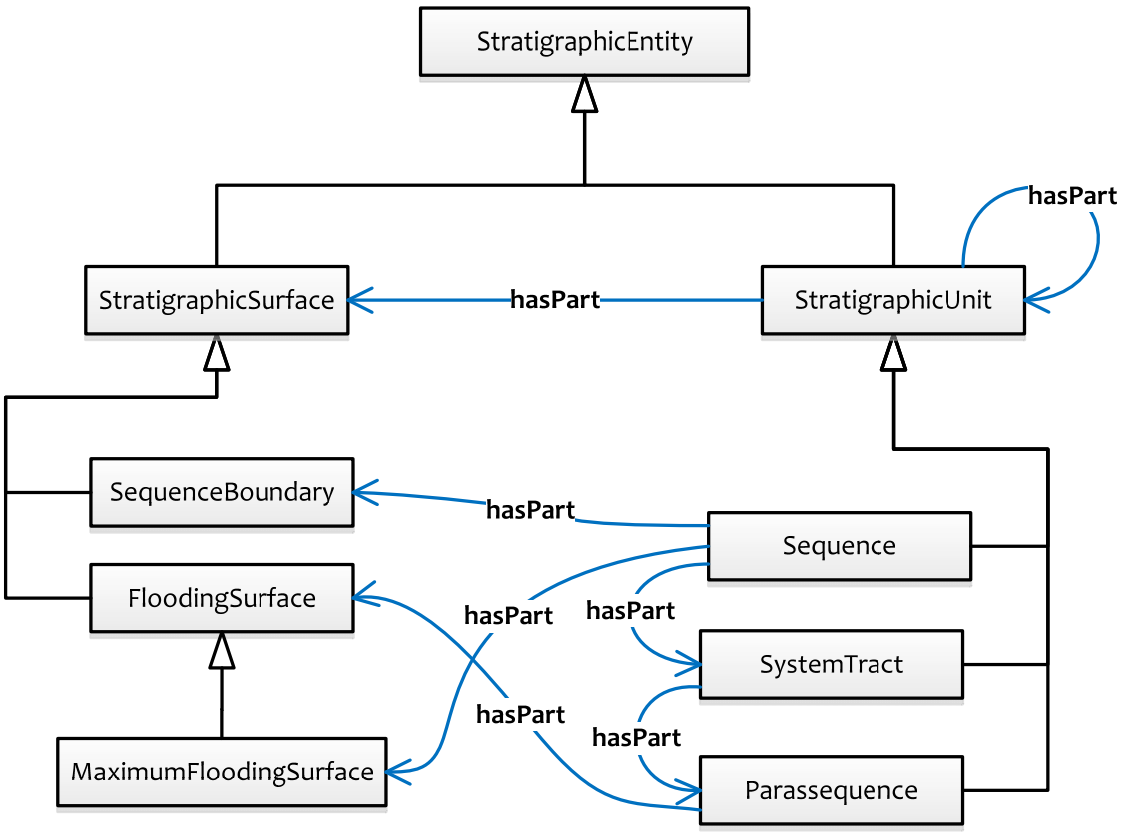

Figura 5.3: Ontologia da Estratigrafia.

A ontologia modelada certamente está longe de representar uma conceitualização completa o suficiente para ser considerada uma ontologia da Estratigrafia. No entanto, o modelo mínimo apresentado é o suficiente para suportar a interpretação estratigráfica a que este sistema se propõe fazer [9].

5.2.1.2 Extensões: nível visual

A ontologia no nível visual proposta pelo framework S-Chart é constituída por primitivas visuais básicas, utilizadas na interpretação de qualquer tipo de gráfico. No entanto, há a possibilidade de extensão do modelo com primitivas visuais específicas de cada domínio de interpretação. No caso do sistema InteliStrata, foi detectada a necessidade de apenas um tipo de extensão.

Como já mencionado anteriormente, em domínios complexos, propriedades visuais idênticas podem ter interpretações diferentes, dependendo do contexto onde são aplicadas. Tomando-se como exemplo sequências e parassequências. A extensão visual de ambas as entidades contém a propriedade que captura a ideia de *comprimento*. No entanto, a escala de interpretação da propriedade varia entre elas. Uma sequência longa (entre 142 m e 845 metros) está em uma escala muito diferente de uma parassequência longa (entre 43 metros e 76 metros). Se essa diferença não for levada em conta, o algoritmo de interpretação não será capaz de diferenciar em qual escala se encontra uma dada entidade visual ainda não classificada como sequência ou parassequência.

Dessa forma, a primitiva `LengthValue` é estendida com duas subclasses definidas como:

```
Class: SequenceLengthValue
   EquivalentTo:
      ValorDeComprimento
      and ({VeryShortLengthSequence,
         ShortLengthSequence,
         MediumLengthSequence,
         LongLengthSequence,
         VeryLongLengthSequence})

Class: ParassequenceLengthValue
   EquivalentTo:
      ValorDeComprimento
      and ({ShortLengthParassequence,
         MediumLengthParassequence,
         LongLengthParassequenc})
```

[9] A arquitetura aqui proposta deverá ser integrada a ontologia completa de domínio que está sendo elaborada pelo mestrando Alexandre Lorenzatti, neste mesmo programa de pós-graduação. Resultados são previstos para o final do ano de 2009.

As instâncias que formam os dois contradomínios de valores de comprimentos são modeladas também como diferentes entre si.

Estas são as únicas extensões necessárias para o nível visual. Eventualmente, elas serão utilizadas para definição dos detectores simbólicos entre os níveis semânticos.

5.2.1.3 Extensões: nível analógico e processamento de sinal

O nível analógico, embora não receba uma extensão propriamente dita, recebe novas entidades necessárias para suporte de aspectos ligados ao sistema. Essas extensões permitem que entidades do modelo referenciem de forma homogênea containeres dos dados brutos que compõem o sinal.

Idealmente, os dados digitais que formam o sinal deveriam ser totalmente representados em termos de primitivas do nível analógico. Por exemplo, como uma cadeia de instâncias da primitiva `Ponto`. No entanto, tal solução é tecnicamente inviável. Uma vez que um sinal pode ter até dezenas de milhares de amostras, seria necessário o mesmo número de instâncias da classe `Ponto` para representá-los. Isso sobrecarregaria desnecessariamente os algoritmos de interpretação, dado que somente alguns desses pontos são significativos para o processo de inferência. A solução para esse problema é a inclusão de primitivas que possam representar os dados digitais que compõem um sinal de forma eficiente, sem que se perca a ligação entre o modelo e esses dados. Para isso, são definidas as primitivas apresentadas na Figura 5.4.

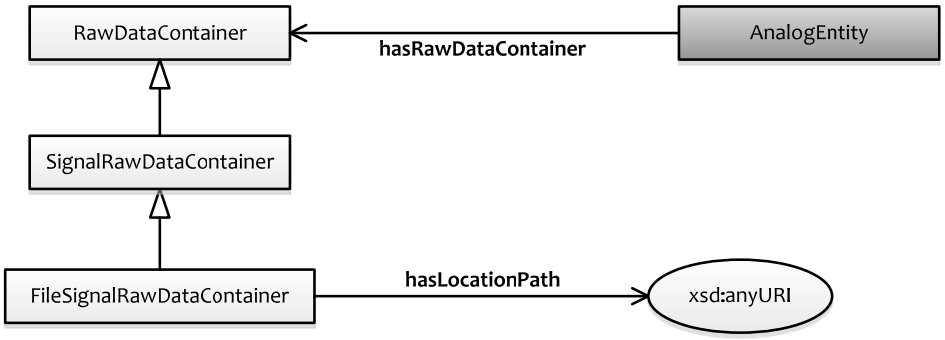

Figura 5.4: Modelo para representação de dados brutos.

No modelo proposto, todas as entidades analógicas devem referenciar algum container de dados pela propriedade `hasRawDataContainer`. Cabe aos algoritmos de processamento e interpretação carregar e interpretar estes dados de forma correta.

A segunda extensão é simples e direta. O framework S-Chart disponibiliza um ontologia simples para representação de algoritmos de processamento de sinal. Algoritmos utilizados no sistema devem ser representados como instâncias desta ontologia. Evidentemente, essas instâncias não se preocupam em representar o funcionamento interno dos algoritmos (detalhados nas seções 4.3 e 5.2.2.1). Elas existem para que os próprios algoritmos tenham a sua existência descrita no modelo, de modo que possam ser referenciados como discriminado-

res no processo de interpretação, por exemplo. Os dois algoritmos são representados da seguinte forma:

```
Individual: SmoothtoothSPF
    Types: WaveletSegmentator

Individual: Gaussian2SPF
    Types: WaveletSegmentator
```

Cabe ao sistema manter a ligação entre estas instâncias e os componentes de software que elas representam. De fato, isso é realizado pelo componente de gerenciamento de processamento de sinal, descrito na seção 5.2.3.1 a seguir.

As novas extensões definidas das últimas seções são suficientes para permitir a modelagem dos modelos de conhecimento visual. No entanto, ainda resta a modelagem dos detectores simbólicos sobre essas primitivas.

5.2.1.4 Detectores simbólicos

Os detectores simbólicos mapeiam explicitamente entidades entre os níveis semânticos do modelo de conhecimento visual. O framework S-Chart define que os detectores simbólicos sejam modelados como regras SWRL com uma estrutura bem definida. Em geral, os detectores são definidos especificamente para cada domínio de aplicação, uma vez que a interpretação visual pode variar entre domínios distintos.

O sistema InteliStrata contém um total de 14 detectores (regras), divididos em 9 conjuntos, que são responsáveis pelo mapeamento. Todos eles são apresentados no Apendice. Neste capítulo é descrito o exemplo de mapeamento do conceito `dom:Sequence` da ontologia de domínio no nível semântico até as entidades analógicas correspondentes.

Inicialmente, o conceito `dom:Sequence` é mapeado para um grupo de detectores simbólicos chamado `sdgDomSequence`. Esse grupo mantém referência para os detectores formados para extrair a entidade semântica de entidades visuais. No caso do conceito `dom:Sequence`, apenas um detector é utilizado (`sdSequence`):

```
mapo:mappingSemanticToVisual(?de, ?ve)
  ∧ vlo:GaussianCurve(?ve)
  ∧ vlo:gaussianCurveCharacteristic(?ve, ?pattern)
  ∧ swrlineq:greaterThanOrEqual(?pattern, vlo:StrongPattern)
  ∧ vlo:length(?ve, ?length)
  ∧ swrlineq:greaterThanOrEqual(?length, MediumLengthSequence)
⇒ dom:Sequence(?de)
```

De acordo com o detector, uma sequência tem como expressão visual uma curva gaussiana com característica forte, com um comprimento maior ou igual a *curto* dentro da escala de tamanho de sequência. Cada uma dessas primitivas visuais, por sua vez, deve ser extraída do nível analógico. Por esse motivo, elas

também apresentam detectores específicos definidos em termos de primitivas analógicas. A primitiva `vlo:GaussianCurve`, por exemplo, é associada a um grupo de detectores (`sdgGaussianCurve`). Esse grupo contém o seguinte detector.

```
mapo:mappingVisualToAnalogic(?ve, ?al)
  ∧ al:Interval(?al)
  ∧ spo:apExtractedWith(?al, spo:Gaussian2SPF)
  ∧ spo:apExtractionStrength(?al, ?vstrength)
  ∧ fuzzy:match(?vstrength, ?strength,
      "[0, 0, 0.2, 0.3]", vlo:VeryWeakPattern,
      "[0.2, 0.3, 0.4, 0.5]", vlo:WeakPattern,
      "[0.4, 0.5, 0.6, 0.7]", vlo:MediumPattern,
      "[0.6, 0.7, 0.8, 0.9]", vlo:StrongPattern,
      "[0.8, 0.9, 1, 1]", vlo:VeryStrongPattern)
 ⇒ vlo:GaussianCurve(?ve)
  ∧ vlo:gaussianCurveCharacteristic(?ve, ?strength)
```

Ele define que uma curva gaussiana é expressa no sinal na forma de um `a-lo:Interval` extraído pelo algoritmo `spo:Gaussian2SPF`. O mesmo detector se encarrega de inferir o valor característico da curva. O valor é inferido através da função fuzzy embutida `fuzzy:match`, que atribui os valores característicos conforme o valor de intensidade (`spo:apExtractionStrength`) correspondente. Em outro exemplo, a propriedade visual `vlo:hasLength` é extraída por um conjunto contento dois detectores (`sdSequenceLength` e `sdParassequence-Length`):

```
mapo:mappingVisualToAnalogic(?ve, ?al)
  ∧ al:Interval(?al)
  ∧ al:apLength(?al, ?value)
  ∧ fuzzy:match(?value, ?fv,
      "[0, 0, 66, 82]", VeryShortLengthSequence,
      "[66, 82, 146, 210]", ShortLengthSequence,
      "[146, 210, 466, 722]", MediumLengthSequence,
      "[466, 722, 1746, 2770]", LongLengthSequence,
      "[1746, 2770, 5000, 7500]", VeryLongLengthSequence)
⇒ vlo:length(?ve, ?fv)

mapo:mappingVisualToAnalogic(?ve, ?al)
  ∧ al:Interval(?al)
  ∧ al:apLength(?al, ?value)
  ∧ fuzzy:match(?value, ?fv,
      "[5, 10, 50, 60]", ShortLengthParassequence,
      "[50, 60, 140, 150]", MediumLengthParassequence,
      "[140, 150, 200, 250]", LongLengthSequence)
⇒ vlo:length(?ve, ?fv)
```

Ambos extraem valores de comprimento contextualizados – para sequência ou parassequência. Ambos também utilizam a função `fuzzy:match` para inferir

os valores lingüísticos para propriedade visual. Os valores utilizados para os conjuntos foram escolhidos com base no conhecimento eliciado do domínio.

Embora ausentes no exemplo anterior, as relações espaciais também são ancoradas no nível analógico. Os seus detectores simbólicos são relativamente independentes de domínio, pois são definidos em função de entidades genéricas no nível analógico. Todos eles são agrupados em um conjunto relacionado à propriedade `vlo:hasSpatialRelationWith`, que inclui todas as propriedades espaciais. Um exemplo de detector para inferência da relação topológica de parte própria sobre intervalos:

```
mapo:mappingVisualToAnalogic(?ve1, ?al1)
  ∧ alo:Interval(?al1)
  ∧ alo:hasStartPoint(?al1, ?sp1)
  ∧ alo:hasEndPoint(?al1, ?ep1)
  ∧ alo:index(?sp1, ?vsx1)
  ∧ alo:index(?ep1, ?vex1)
  ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2)
  ∧ alo:Interval(?al2)
  ∧ alo:hasStartPoint(?al2, ?sp2)
  ∧ alo:hasEndPoint(?al2, ?ep2)
  ∧ alo:index (?sp2, ?vsx2)
  ∧ alo:index (?ep2, ?vex2)
  ∧ swrlb:greaterThan(?vsx2, ?vsx1)
  ∧ swrlb:greaterThan(?vex1, ?vex2)

⇒ vlo:isNonTangentialProperPartOf(?ve2, ?ve1)
  ∧ vlo:hasForNonTangentialProperPartOf(?ve1, ?ve2)
```

Esse detector relaciona intervalos que estão propriamente contidos em outros intervalos, testando os limites de cada um. O resultado é uma relação `isNonTangentialProperPartOf` (e a sua inversa `hasForNonTangentialProperPartOf`) entre o intervalo contido e o intervalo que contém.

Os detectores definidos no sistema InteliStrata, bem como os demais modelos de conhecimento visual estão completamente descritos no Anexo.

5.2.2 Componentes de infra-estrutura

Para dar suporte a todos os serviços de raciocínio e processamento de sinal, o sistema InteliStrata emprega componentes desenvolvidos por terceiros. Sobre estes componentes de infra-estrutura (Figura 5.5) são construídos os demais algoritmos de interpretação. A fim de justificar os componentes escolhidos, algumas considerações são feitas.

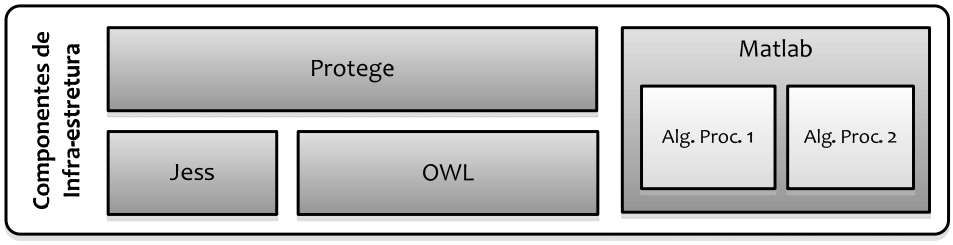

Figura 5.5: Componentes de infra-estrutura do sistema InteliStrata.

A primeira se refere à representação dos modelos de conhecimento. Como discutido na seção 3.1, os modelos de conhecimento do framework S-Chart usão formalizados na linguagem OWL. Essa linguagem tem suporte de algumas ferramentas modelagem, sendo o software Protégé (KNUBLAUCH et al., 2004) o escolhido para utilização no InteliStrata. Essa escolha é pautada por alguns motivos, sendo os principais:

- No momento da escrita deste trabalho, o Protégé é certamente a mais popular e estável ferramenta para modelagem a manipulação de OWL. Ela conta com uma grande comunidade de usuários e sua distribuição é livre.
- A ferramenta disponibiliza uma interface de programação que permite a manipulação e consulta de modelos OWL diretamente de outras aplicações. Esta facilidade permite que o sistema InteliStrata manipule os modelos de conhecimento visual diretamente, sem a necessidade da sua conversão para paradigmas de representação de informação menos expressivos (como paradigma relacional). Isso atende diretamente ao objetivo de desenvolvimento com preservação de estrutura.
- A arquitetura da própria ferramenta é extensível, permitindo a criação e integração de extensões e a sua utilização via a interface de software. Para o sistema InteliStrata é utilizada a extensão SWRLTab (O'CONNOR et al., 2005), direcionado manipulação de regras SWRL. Essa extensão tem a sua importância em si, de forma que permite a definição e implementação de extensões sobre a própria linguagem SWRL, uma característica importante para definição dos detectores *fuzzy*.

A execução das regras SWRL é garantida pelo software Jess (FRIEDMAN, 2003), uma máquina para execução de regras também extensível. Embora o Jess seja uma pacote de software independente, a sua interface de programação pode ser facilmente integrada ao Protégé, via SWRLTab. De fato, atualmente a máquina Jess é a única máquina que permite a execução de regras SWRL funções embutidas definidas pelo usuário. Adicionalmente, é importante ressaltar que o sistema InteliStrata não faz uso de raciocinadores $\mathcal{DL}$, como os implementados nas ferramentas Pellet (PARSIA et al., 2005) ou FaCT++ (TSARKOV; HORROCKS, 2006), uma vez que os modelos do sistema não apresentam relações implícitas e as instâncias dos modelos são geradas já na sua classificação definitiva.

A escolha das ferramentas Protégé e Jess também implica que a implementação seja baseada no ambiente Java (LINDHOLM; YELLIN, 1999), uma vez que essas duas disponibilizam as suas interfaces de programação nessa linguagem.

Qualquer sistema de interpretação como este requer algum tipo serviço para processamento numérico que seja eficiente. No caso do sistema InteliStrata, esses serviços de processamento devem dar suporte ao desenvolvimento de processamento de sinal de uma dimensão. Existe pouco suporte nativo para a linguagem Java. Por isso, escolheu-se utilizar o software matemático Matlab R2008a já mencionado anteriormente, que disponibiliza diversos pacotes para processamento de dados numéricos e, em especial, processamento de sinal. Por isso, o terceiro e último grande componente de infra-estrutura é o software Matlab. Os processamentos realizados com ele são melhor detalhados na subseção seguinte.

Por fim, as ferramentas apresentadas garantem uma infra-estrutura adequada que atende aos objetivos do sistema InteliStrada. O suporte da ferramenta Protégé à manipulação direta dos modelos de conhecimento ajuda na manutenção da estrutura.

5.2.2.1 Algoritmos de processamento por wavelet

O processamento inicial do sinal é realizado dentro do software Matlab através da aplicação da TWC com as *wavelets* apresentadas nas seções 4.3. No entanto, a aplicação não é direta. O sinal necessita de tratamento pré-processamento e pós-processamento, onde o resultado bruto da TWC é tratado antes de ser disponibilizado aos níveis superiores de raciocínio.

A TWC para ambas as wavelets passam por processamentos semelhantes, variando em alguns parâmetros. Os passos do algoritmo são os seguintes:

- O sinal é re-amostrado para freqüência mínima necessária para detecção de sequências (wavelet gaussiana) ou parassequências (wavelet *smoothtooth*). A freqüência mínima é derivada através do período mínimo de uma sequência (15 metros) ou parassequência (3 metros). O resultado é um sinal com menor quantidade de amostras, menor ruído da alta freqüência e mais eficiente para processamento;
- A TWC é aplicada sobre o sinal re-amostrado. Em ambos os casos, a transformada é configurada para utilizar extensões espelhadas no sinal, a fim de amenizar os problemas de processamento na borda;

Neste pontos, os o algoritmo para a *wavelet* guassiana se diferencia da *wavelet smoothtooth*. Para *wavelet* gaussiana:

- O espectro resultante é normalizado por nível;
- Considera-se o espectro como uma superfície e procura-se pelos máximos locais na superfície.

- Sendo que cada máximo esta contida em uma das escalas, são procurados os mínimos à direita e a esquerda dentro da escala, considerando apenas essa escala. Os pares de mínimos formam os limites de segmentação

No caso da *wavelet sawtooth*:

- O espectro é normalizado e invertido;
- Consideram-se cada escala do espectro e procura-se pelos máximos locais em cada uma.
- A seguir, são encontrados os mínimos locais de cada lado dos máximos. Os pares de mínimas formam os limites de segmentação.

Nesse ponto os algoritmos voltam a ser iguais:

- Em seguida, um laço iterativo busca pelos limites de cada intervalo encontrado e segmenta o sinal de acordo com esses limites. O pico de cada intervalo é dado como a força padrão (que acabada dando origem à propriedade característica no nível visual). Quanto mais alto o número, mais a curva para o intervalo se parece com o padrão da *wavelet*.

Ao final, os dados são organizados e repassados ao ambiente Java para conversão em entidades do modelo de conhecimento (`alo:Interval` em geral).

5.2.3 Componentes de interpretação

Os componentes de interpretação caracterizam a máquina de interpretação semântica em si. Estes componentes são três (Figura 5.6): (i) o *componente gerenciador de processamento de sinal*, que trata de gerenciar a segmentação e extração do sinal a ser analisado pelo sistema; (ii) o *componente de interpretação*, que implementa as máquinas MIS e MIV descritas no capítulo anterior; e (iii) o *componente de interface gráfica* com o usuário (GUI), que disponibiliza o resultado da interpretação ao usuário final.

Os três componentes trocam informações dentro de um contexto compartilhado. Esse contexto é dado pela ontologia OWL `InteliStrataOntology`. De fato, essa é uma extensão da utilização da estrutura `Contexto` das máquinas MIS e MIV, descritas na seção 3.2.5 no capítulo anterior.

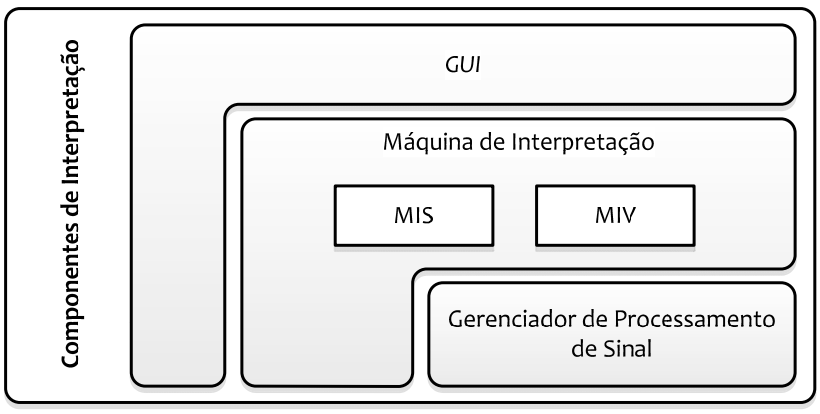

Figura 5.6: Componentes de interpretação do sistema InteliStrata.

5.2.3.1 *Gerenciador de processamento de sinal*

O gerenciador de processamento de sinal tem por função delegar e gerenciar a aplicação dos algoritmos de processamento presentes no sistema. Ele recebe primitivas analógicas que devem ser extraídas diretamente do sinal e aciona o algoritmo adequado para a sua extração. O resultado da extração são instâncias dessas primitivas incluídas no modelo OWL para que sejam utilizados no processo de interpretação simbólico.

Cada algoritmo de processamento pode processar mais de uma primitiva por vez. Por exemplo, um algoritmo que extrai um conjunto de intervalos pode ao mesmo tempo calcular outras propriedades como comprimento, ponto inicial e final e etc.

Grande parte dos algoritmos é implementado dentro do ambiente Java do sistema InteliStrata. No entanto, a extração por *wavelets* é repassada, via Java RMI, para módulos nativos do sistema Matlab. Nele, os algoritmos de extração utilizam as funcionalidades dos pacotes de processamento de sinal para ajusta o sinal que compõe o perfil de poço e extrair as feições visuais conforme descrito na seção anterior. O resultado bruto é repassado para o ambiente Java e convertido em entidades analógicas dentro da ontologia OWL.

5.2.3.2 *Componente de interpretação*

O componente de interpretação implementa as máquinas MIS e MIV, conforme os algoritmos especificados na seção 3.2.5. Embora a implementação seja bastante direta em relação à especificação, alguns aspectos necessitam uma atenção especial.

Inicialmente, todas as manipulações de entidades dos modelos de conhecimento são realizadas diretamente sobre as ontologias OWL que representam esses modelos. Isso é feito através da interface de programação do aplicativo Protégé. Dessa forma, as entidades manipuladas não precisam ser convertidas para algum outro formalismo (como relacional, por exemplo) a fim de serem utilizadas.

Os algoritmos das máquinas MIS e MIV necessitam de algum mecanismo para execução dos detectores simbólicos, definidos como regras SWRL. No sistema InteliStrata, a execução das regras é feita pelo componente Jess. A interface do componente Jess com o sistema InteliStrata é realizada pelo componente SWRLTab.

5.2.3.3 *Componente de interface*

O componente de interface gráfica (GUI) fornece ao usuário um meio para manipular e visualizar os resultados gerados pela interpretação. Idealmente, a interface deve prover as seguintes funcionalidades:

- **Filtragem interativa das informações interpretadas**. A interface deve permitir ao usuário filtrar os resultados da interpretação, seja para

melhor visualização, seja para excluir interpretações espúrias que podem ocorrer;

- **Carregamento de perfis de poço e salvamento de interpretação**: o sistema deve permitir que dados de perfis de poços sejam importados e que o resultado das interpretações seja salvo;
- **Exibição gráfica interativa**: a exibição dos resultados deve ser gráfica e interativa (com operações de zoom e etc.)

Definidos esses requisitos, é necessário ressaltar que o foco principal deste trabalho é o desenvolvimento de um aplicativo protótipo que demonstre a viabilidade do framework S-Chart para interpretação de gráficos. Devido a esse fato, o protótipo implementado aqui apenas demonstra como estas funcionalidades podem ser alcançados. Uma versão operacional deste sistema demandaria um estudo de interface e estratégias de visualização de dados.

O componente de interface projetado para o protótipo permite a visualização de todos os resultados da interpretação. Funções de carregamento de dados, salvamento e filtragem foram implementadas diretamente no código, como uma prova de conceito. A interface, com o resultado da interpretação para o perfil do poço *Tenneco Rattlesnake State 2-12* é demonstrado na Figura 5.7.

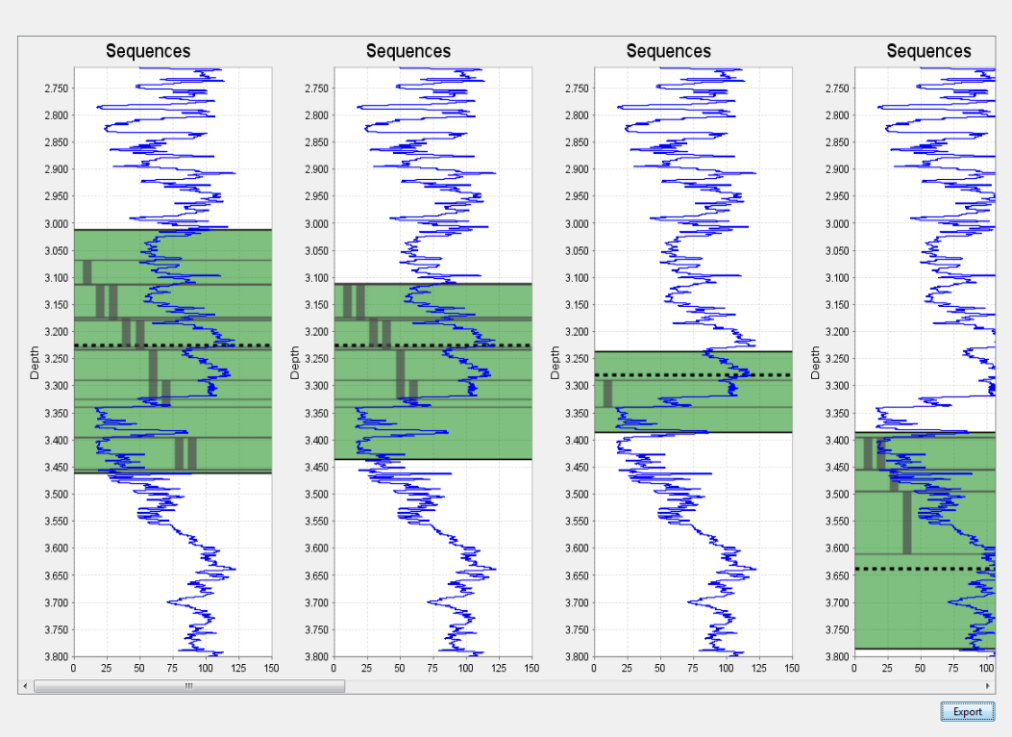

Figura 5.7: Interpretação de parte do perfil *Tenneco Rattlesnake State 2-12*.

Um dos aspectos importantes do componente de interface é que os resultados da interpretação são recuperados diretamente dos modelos OWL, via Protègè. A consulta é realizada com a linguagem SPARQL, linguagem padrão de consulta para bases de dados RDF e OWL. Ela representa uma forma consistente para pesquisa de informações geradas nos modelos. Além disso, os pró-

prios modelos podem orientar a pesquisa. Por exemplo, para exibição das entidades na Figura 5.7, todas as sequências são recuperadas inicialmente. Para exibição, os atributos analógicos associados com cada sequência são recuperados. A seguir, conforme o modelo, as partes de cada sequência (parassequência e superfície de inundação máxima) são recuperadas e também exibidas com base nos dados analógicos onde estão ancoradas. Outro aspecto, é que consultas SPARQL podem ser geradas dinamicamente. Em um sistema completo, elas podem ser usadas para implementar um mecanismo de filtragem dinâmica das informações do modelo.

5.3 Validação do sistema InteliStrata

Aqui será demonstrada a avaliação de dois perfis de poços para os quais existe interpretação detalhada. As duas seções escolhidas para estudo de caso deste trabalho tem significado histórico para a Geologia. Seu estudo deu origem às propostas originais de interpretação de bacias sedimentares a partir da identificação da oscilação do nível dos oceanos ao longo do tempo geológico, ou seja, deu origem à própria área de Estratigrafia de Sequências. Elas foram utilizadas como exemplos na publicação introdutória da área de Estratigrafia de Sequências de Van Wagoner e colegas (1990) e constituem a escolha ideal para teste do sistema InteliStrata pois a sua interpretação estratigráfica já é consolidada e reconhecida.

A metodologia de avaliação é direta e típica em sistemas de conhecimento: o sistema é válido se, para um dado perfil e uma dada interpretação desse perfil, o especialista no domínio considera o resultado da interpretação relevante.

Para validação do sistema InteliStrata foram utilizadas duas seções retiradas de dois perfis de raios gama distintos. As seções são:

- **Seção 1** de raios gama com aproximadamente 340 metros, retirada a partir da profundidade 820m do perfil medido no poço *Tenneco Rattlesnake State 2-12*, localizado na região de Book Cliffs, em Utah, EUA.
- **Seção 2** de raios gama com aproximadamente 360 metros, retirada a partir da profundidade 30m do perfil medido no poço *Exxon Production Research Co. Sego Canyon nº 2*, também localizado na região de Book Cliffs.

5.3.1 Preparação dos perfis

As versões originais dos perfis foram obtidas no site do Departamento de Recursos Naturais do governo do estado de Utah, EUA (DEPARTMENT OF NATURAL RESOURCES, 2009). Eles são disponibilizados como imagens digitalizadas dos perfis em papel, no formato TIFF com 1 bit de cor, como a Figura 4.1 no início deste capítulo.

Para utilização no sistema InteliStrata, o perfil deve ser totalmente digitalizado. O processo de digitalização das imagens dos perfis foi implementado da seguinte forma:

- A imagem do perfil é limpa de outras informações até restar somente a área que representa o perfil de raios gama. Em seguida a imagem é erodida para eliminar a maior parte de ruídos e imperfeições.
- Da imagem do perfil é derivada uma máscara que descreve a somente curva do perfil com uma espessura maior. Esse processo é realizado manualmente.
- A imagem limpa e a máscara são carregadas no programa Matlab. Um algoritmo desenvolvido como parte deste trabalho faz a digitalização. A cada linha de pixels da imagem limpa, encontram-se os limites da máscara. Dentro deste limite, o algoritmo busca pela maior linha de pixels pintados e calcula o seu centro. Este deve ser o centro da linha que caracteriza o perfil. Esse valor é gravado num vetor e repete-se o processo para a próxima linha. Posteriormente, os valores de posição do vetor são convertidos para a escala de leitura do perfil de raios gama.
- Eventualmente, algumas leituras são rejeitadas e o sinal ganha o valor zero em alguns pontos. Estes problemas de digitalização são eliminados com algoritmos de interpolação.

O resultado do processo é uma representação vetorial do sinal original do perfil. Para utilização no sistema InteliStrata, uma instância de `FileRawDataHolder` é criada, apontando para um arquivo com esse vetor.

Numa aplicação real, os próprios sistemas de perfilagem de poço geram o arquivo com o vetor de dados que seria a entrada do sistema InteliStrata.

5.3.2 Avaliação dos resultados e validação

O primeiro fator importante na avaliação das interpretações obtidas pelo sistema InteliStrata se refere a forma como é feita a interpretação estratigráfica. Como descrito anteriormente, uma interpretação completa é obtida através da correlação de diversas informações, não somente do perfil de raios gama. De acordo com o especialista, a interpretação baseada puramente no perfil ocorre quando as informações complementares ainda não estão disponíveis. O resultado da interpretação do perfil tem o papel de dar a primeira ideia de quais são as feições geológicas presentes (sequência, parassequências e etc.), para que, posteriormente, sejam ajustadas conforme a disponibilidade de dados de testemunho, sísmica ou estudos geológicos anteriores da bacia sedimentar. Dessa forma, o resultado da interpretação realizada pelo sistema InteliStrata deve ser comparado com o resultado da análise preliminar do perfil, feita pelo especialista sem a complementação de outros dados geológicos.

O primeiro teste foi realizado com o perfil da Seção 1 e é demonstrado na Figura 5.8. A Interpretação A apresenta a interpretação realizada pelo especialista levando em conta somente o perfil. Já as interpretações B (B1, B2, B3, B4 e B5) são as obtidas pelo sistema InteliStrata.

Inicialmente, é possível notar uma discordância entre o número de interpretações do especialista e do sistema. De fato, o sistema sugere interpretações al-

ternativas para as mesmas sequências. Três delas (B1, B2 e B4) concordam razoavelmente com os limites de sequência da interpretação gabarito. No entanto, não há nenhuma interpretação do sistema que sugira a presença da Sequência 3. É possível ajustar os níveis de ruído para possibilitar a sua interpretação. No entanto, tal ajuste possibilita a passagem de sequências espúrias na interpretação final.

No caso das superfícies de inundação máxima (mfs), todas as interpretações do sistema concordam com a do especialista (ex.: seta B).

Já a interpretação das parassequências é mais ruidosa, gerando interpretações duplas e (ex.: seta A) falsos positivos (ex: seta C). No entanto, em alguns casos, há concordância quase exata, como as parasequências 8 e 9 na Sequência 2 e, em um menor grau, a parasequências 3 na Sequência 1. Essa maior discordância na interpretação ocorre devido a menor escala das parassequências. Feições em escalas muito pequenas se confundem com ruídos de alta freqüência durante segmentação do sinal, sendo, por isso, eliminadas no processamento simbólico.

O segundo teste foi realizado com o perfil da Seção 2, demonstrado na Figura 5.9. A Interpretação A apresenta a interpretação realizada pelo especialista levando em conta somente o perfil. Já as interpretações B (B1, B2, B3 e B4) são as obtidas pelo sistema InteliStrata.

Do ponto de vista qualitativo, a interpretação da Seção 2 é semelhante a da Seção 1, com destaque positivo para a grande coincidência da interpretação B3 com a Sequência 1.

Por fim, os resultados apresentados aqui foram descritos como satisfatórios pelo especialista, mesmo havendo discrepâncias entre a interpretação sugerida e a feita pelo sistema. A racionalização por trás desse fato é a imprecisão inerente à interpretação estratigráfica por perfis de raios gama. A marcação precisa dos limites das feições é uma tarefa significativamente complexa e um tanto subjetiva, especialmente quando se leva em conta somente a informação do perfil. Um fato que corrobora com essa análise é a afirmação feita pelo especialista ao confrontar os resultados, declarando que algumas das feições interpretadas pelo sistema são visões válidas sobre o perfil, mesmo contrariando a sua própria interpretação. Um exemplo é a diferença de interpretação da mfs apontada pela seta D na figura Figura 5.9. O especialista afirma que a interpretação dada pelo sistema para a mfs também é uma interpretação possível, quando não estao disponíveis outros dados geológicos.

Os resultados mostram-se ainda mais válidos quando se considera o tempo demandado para a interpretação manual dos perfis pelos profissionais aptos a realizar interpretação de sequências. O processamento automático associado ao ajustes manuais necessários para obter um conjunto válido de interpretações requerem um tempo significativamente menor para a tarefa de interpretação quando comparado ao necessário para realizar a atividade na realidade atual.

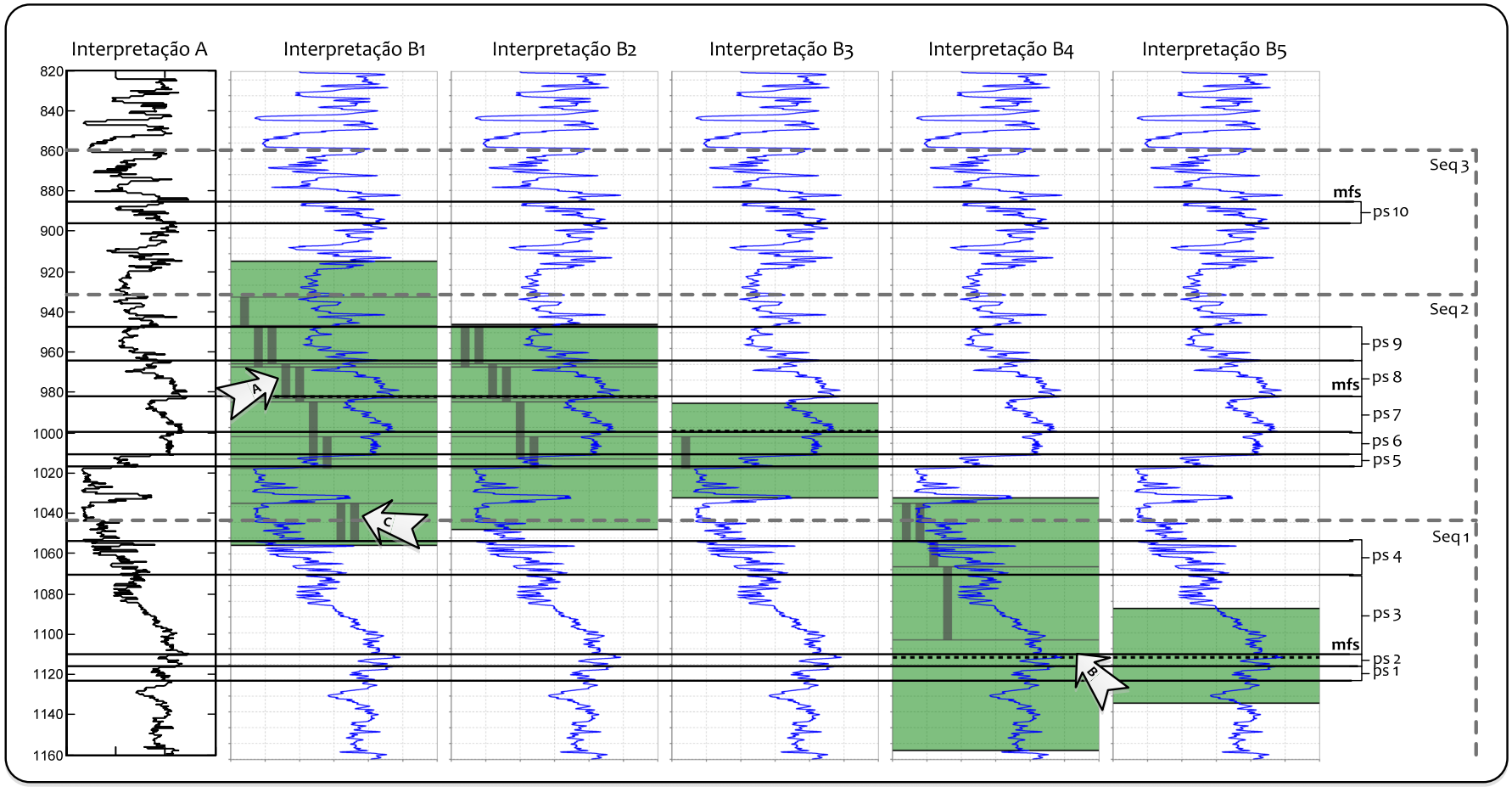

Figura 5.8: Comparação das interpretações do especialista e do sistema InteliStrata para Seção 1.

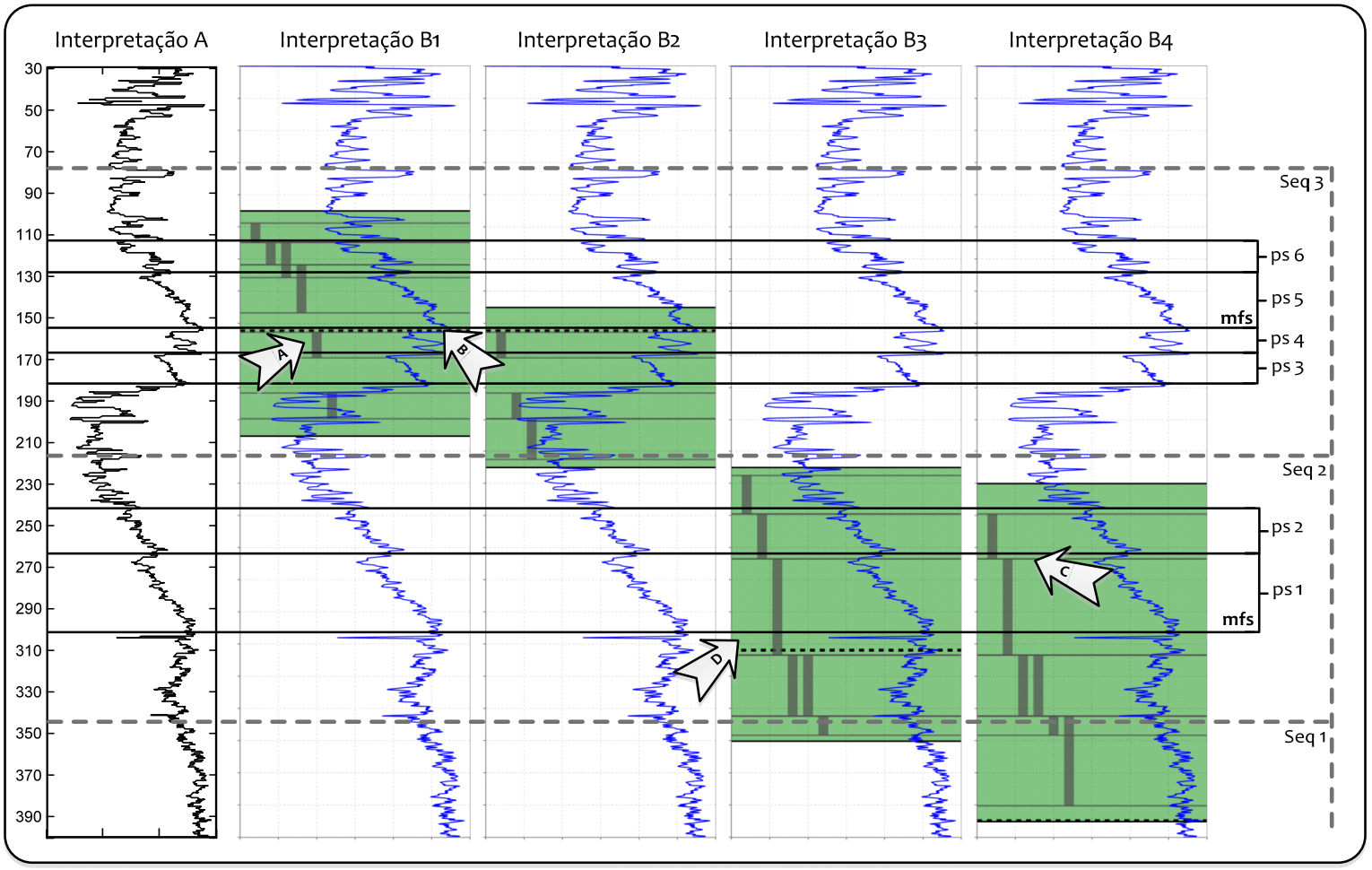

Figura 5.9: Comparação das interpretações do especialista e do sistema InteliStrata para Seção 2.

6 CONCLUSÃO

A área de visão computacional é de grande importância para computação. A tarefa de interpretar o conteúdo de dados pictóricos é bastante desafiadora e tem sido abordada de diversas maneiras nas últimas duas décadas. Uma dessas maneiras é a interpretação semântica de imagens, área deste trabalho. Ela busca unir os esforços de pesquisa alcançados nas áreas de processamento de imagens e de representação de conhecimento, especialmente conhecimento visual.

Este trabalho propôs um framework completo para interpretação semântica de gráficos, um tipo específico de imagem que representa visualmente um sinal de uma dimensão. O S-Chart, como é chamado, define modelos de conhecimento visual em três níveis semânticos e aplica conceitos de ancoramento simbólico para realizar o mapeamento entres entidades destes níveis. Além de propor algoritmos genéricos para operacionalizar a interpretação visual com base nos modelos de conhecimento.

Este trabalho também descreveu o sistema InteliStrata, uma implementação do framework S-Chart para interpretação de gráficos no domínio da Geologia, mais especificamente, no domínio da Estratigrafia de Sequências. O objetivo do sistema InteliStrata é identificar feições geológicas (sequências, parassequências e superfícies de inundação máxima) com base na interpretação semântica de gráficos de perfis de raios gama. Os resultados obtidos foram considerados pelo especialista em Estratigrafia de Sequências que acompanhou o projeto como compatíveis com a interpretação dada por um geólogo com experiência nesse tipo de interpretação a partir dos mesmos dados.

6.1 Framework S-Chart e modelos de conhecimento visual

A análise de propostas de interpretação semântica existentes na literatura demonstrou que poucas delas são voltadas para representação e interpretação de dados pictóricos em domínios imagísticos. Além disso, nenhuma das propostas aborda a interpretação de gráficos, cuja importância é relevante em diversos domínios. Tampouco foram encontradas abordagens que se baseiam em padrões de representação abertos, o que dificulta a integração dos resultados da interpretação com outros sistema de conhecimento, por exemplo. Por esses motivos, desenvolveu-se uma nova abordagem de interpretação semântica de imagens voltada para interpretação de gráficos.

Os resultados positivos do framework S-Chart se dividem em duas partes. Primeiramente, o modelo de representação de conhecimento visual em três níveis conseguiu capturar o conhecimento necessário para interpretação de gráficos de forma independente da aplicação, garantindo a reusabilidade da abordagem em outros domínios. Isso se deu na forma de primitivas de representação em todos os níveis semânticos, com destaque para as primitivas no nível visual. Outro grande avanço no framework S-Chart foi a definição de uma maneira homogênea de mapeamento entre níveis semânticos, na forma dos detectores simbólicos.

Em segundo lugar, a escolha da linguagem OWL e SWRL como formalismo de representação dos modelos de conhecimento facilita a implementação do framework em sistema de interpretação, como se demonstrou com o sistema InteliStrata. As vantagens sobre a utilização de formalismos específicos são diversas, desde a utilização de uma linguagem já bem formalizada até a integração com outros sistemas que utilizam a mesma linguagem.

O sistema InteliStrata demonstrou como o framework S-Chart pode ser implementado em um sistema real para interpretação de perfis de poço, no domínio Estratigrafia de Sequência. O sistema complementa partes do framework definidas como dependente de domínio. Essas partes são um modelo de domínio simples descrevendo os principais conceitos da Estratigrafia de Sequência, primitivas visuais dependentes do domínio e alguns detectores simbólicos. Além disso, foi demonstrado como componentes de software e sistemas de terceiros podem ser integrados para dar suporte a interpretação semântica. Os resultados apresentados pelo sistema InteliStrata foram validados e considerados satisfatórios pelo especialista no domínio.

6.2 Sugestão para trabalhos futuros

Durante o desenvolvimento do framework, foram propostas diversas primitivas de representação, especialmente no nível visual. Embora o sistema InteliStrata faça uso de apenas algumas delas, claramente novas primitivas podem ser definidas. De fato, ao contrário do que acontece com imagens reais, não existem trabalhos que definam parâmetros visuais genéricos em gráficos. Assim, à medida que novas aplicações sejam desenvolvidas, a ontologia que modela o nível visual pode ser incrementada como novas primitivas.

Um problema encontrado durante o desenvolvimento do modelo do nível visual foi dar uma representação simbólica textual adequada a alguns conceitos. A representação textual nem sempre se mostrou eficiente pra capturar o significado completo do conceito. Como no caso dos tipos de curvas modeladas e, em especial, da entidade `CurvaSigmoide`. Uma curva sigmóide tem o mesmo formato de uma curva logística. Assim, se ambos os conceitos forem modelados, eles serão equivalentes, gerando redundâncias no modelo. Entretanto, se, ao invés de um símbolo textual, fosse usado um símbolo gráfico para representar o conceito (ex.: um desenho da curva), a sua representação se tornaria mais natural. Esse problema pode ser resolvido através do desenvolvimento de primiti-

vas de representação de conhecimento que aceitem o uso de ícones e figuras como representação simbólica[10].

Uma lacuna deixada no desenvolvimento do framework foi o tratamento dos índices de pertinência resultantes das funções fuzzy utilizadas pelos descritores simbólicos. De fato, os índices de característica utilizados no nível visual consistem na alternativa encontrada para modelar os índices de pertinências resultados das funções fuzzy no modelo visual. Embora não a ideal, essa modelagem se mostrou eficiente para os fins deste trabalho. No entanto, no momento da escrita desta dissertação, uma nova versão da linguagem OWL está sendo definida e esta poderia modelar facilmente essas informações. As novas primitivas antevistas permitiriam a modelagem de relações de anotação sobre qualquer entidade do modelo, até mesmo sobre axiomas entre instâncias. Dessa forma, seria possível anotar, por exemplo, um grau de certeza na relação entre uma instância e a sua classe. Deste modo, as primitiva `temCaracteristica` usada no framework S-Chart poderia passar a ser uma relação de anotação sobre entidades visuais extraídas do nível analógico.

Por fim, como foi afirmado no desenvolvimento do trabalho, a definição completa de um módulo para gerenciar a aplicação de algoritmos de processamento de imagem fogem do escopo deste trabalho. No entanto, ele é de grande importância para um sistema de interpretação semântica. Um módulo deste tipo deveria ser capaz de, dado um conjunto de primitivas analógicas que devem ser extraídas do sinal, decidir quais algoritmos aplicar e em qual ordem. Definidos como serviços de processamento, suas interfaces devem ser formalizadas dentro dos modelos, presumivelmente em uma extensão do modelo de processamento de sinal descrito anteriormente.

[10] No momento da escrita deste trabalho, esta alternativa de representação está também sendo pesquisada por Alexandre Lorenzatti (vide note anterior).

REFERÊNCIAS

BARSALOU, L. W. Perceptual symbol systems. **Behavioral and Brain Sciences**, New York. v. 22, n. 4, p. 577-660, 1999.

CHELLA, A.; FRIXIONE, M.; GAGLIO, S. A cognitive architecture for artificial vision. **Artificial Intelligence**, Amsterdan. v. 89, n. 1-2, p. 73-111, 1997.

CHELLA, A.; FRIXIONE, M.; GAGLIO, S. Conceptual Spaces for Computer Vision Representations. **Artificial Intelligence Review**, Dordrecht. v. 16, n. 2, p. 137-152, 2001.

CHOUDHURY, S. et al. Use of Wavelet Transformation for Geophysical Well-Log Data Analysis. In: 15th International Conference on Digital Signal Processing. **Proceedings...** , Cadiff (UK) 2007. p. 647-650.

COHN, A. G. et al. Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. **GeoInformatica**, Boston. v. 1, n. 3, p. 275-316, 1997.

COHN, A. G. et al. Towards an Architecture for Cognitive Vision Using Qualitative Spatio-temporal Representations and Abduction. **Spatial Cognition III**, Berlin: Springer, 2003. p. 246-262. (Lecture Notes in Computer Science, 2685).

COLOMB, R. et al. The Object Management Group Ontology Definition Metamodel. **Ontologies for Software Engineering and Software Technology**, Berlin: Springer, 2006. p. 217-247.

CORADESCHI, S. et al. Fuzzy anchoring. In: The 10th IEEE International Conference on Fuzzy Systems. **Proceedings...** , [S.l.]. v. 1, 2001. p. 111-114.

CREVIER, D.; LEPAGE, R. Knowledge-based image understanding systems: a survey. **Comput. Vis. Image Underst.**, [S.l.]. v. 67, n. 2, p. 160-185, 1997.

CRUBÉZY, M.; MARCOS, M.; MOISAN, S. Experiments in Building Program Supervision Engines from Reusable Components. In: WORKSHOP on Applications of Ontologies and Problem-Solving Methods. **Proceedings...** , Brighton, England 1998. p. 44-53.

DELLA FÁVERA, J. C. **Fundamentos de estratigrafia moderna**. Rio de Janeiro: EdUERJ, 2001,

DEPARTMENT OF NATURAL RESOURCES. **State of Utah - Oil and Gas Program - Home Page**. 2009. Disponível em: <http://oilgas.ogm.utah.gov/>. Acesso em: 25 Fev 2009.

FAN, J. et al. Statistical modeling and conceptualization of natural images. **Pattern Recognition**, [S.l.]. v. 38, n. 6, p. 865-885, 2005.

FIORINI, S. R. **Uma Proposta de Arquitetura de Componentes pra Sistemas de Conhecimento para Avaliação de Reservatórios de Petróleo**. 2006. 65 p. Monografia (Bacharelado em Ciência da Computação) - Instituto de Informática, Universidade Federal do Rio Grande do Sul.

FRIEDMAN, E. **Jess in action: rule-based systems in java**. Greenwich, US: Manning Publications Co., 2003,

GANGEMI, A. et al. Sweetening WORDNET with DOLCE. **AI Magazine**, Menlo Park, US. v. 24, n. 3, p. 13-24, 2003.

GÄRDENFORS, P. **Conceptual Spaces: The Geometry of Thought**. Cambridge, Massachussetts: The MIT Press, 2004,

GOMEZ, I.; DATCU, M. A Bayesian multi-class image content retrieval. In: International Geoscience and Remote Sensing Symposium. **Proceedings...** , Los Alamitos, USA: IEEE, 2007. p. 326-329.

HANSON, A.; RISEMAN, E. VISIONS: A computer system for interpreting scenes. **Computer Vision Systems**, [S.l.]. 1978.

HARNAD, S. The symbol grounding problem. **Phys. D**, [S.l.]. v. 42, n. 1-3, p. 335-346, 1990.

HARNAD, S. **Computation Is Just Interpretable Symbol Manipulation: Cognition Isn't**. 1994. Disponível em: <http://cogprints.org/1592/>. Acesso em: 29 Jan 2009.

HERZOG, G. From visual input to verbal output in the visual translator. In: AAAI Fall Symposium on Computational Models for Integrating Language and Vision. **Proceedings...** , Cambridge, USA: AAAI, 1995.

HORRIDGE, M. et al. The Manchester OWL Syntax. In: OWLED*06 Workshop on OWL: Experiences and Directions. **Proceedings...** , Athens, USA v. 216, 2006.

HORROCKS, I. et al. **SWRL: A Semantic Web Rule Language Combining OWL and RuleML**. 2004. Disponível em: <http://www.w3.org/Submission/SWRL/>. Acesso em: 17 Fev 2009.

HUDELOT, C. **Towards a Cognitive Vision Platform for Semantic Image Interpretation; Application to the Recognition of Biological Organisms**. 2005. 275 p. Thesis (Phd in Computer Science) - Université de Nice Sophia Antipolis.

HUDELOT, C.; MAILLOT, N.; THONNAT, M. Symbol Grounding for Semantic Image Interpretation: From Image Data to Semantics. In: Tenth IEEE International Conference on Computer Vision, 2005. **Proceedings...** , Los Alamitos, USA: IEEE, 2005. p. 1875.

JARKE, M. et al. KBMS requirements of knowledge-based systems. **Foundations of knowledge base management**, New York: Springer-Verlag, 1989. p. 381-394.

KNUBLAUCH, H. et al. The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications. **The Semantic Web – ISWC 2004**, [S.l.]. 2004. p. 229-243.

LINDHOLM, T.; YELLIN, F. **Java Virtual Machine Specification**. Addison-Wesley Longman Publishing Co., Inc., 1999,

MAILLOT, N. **Ontology Based Object Learning and Recognition**. 2005. 166 p. Thesis (Phd in Computer Science) - École Doctorale STIC, Université de Nice.

MATHWORKS INC. **Matlab 2008a**. Natick, USA

MATSUYAMA, T. Knowledge-Based Aerial Image Understanding Systems and Expert Systems for Image Processing. **IEEE Transactions on Geoscience and Remote Sensing**, New York. v. GE-25, n. 3, p. 305-316, 1987.

MCGUINNESS, D. L.; HARMELEN, F. V. **OWL Web Ontology: Language Overview**. 2004. Disponível em: <http://www.w3.org/TR/owl-features/>. Acesso em: 17 Fev 2009.

NEUMANN, B.; MOLLER, R. On scene interpretation with description logics. **Image and Vision Computing**, [S.l.]. v. 26, n. 1, p. 82-101, 2008.

O'CONNOR, M. et al. Supporting Rule System Interoperability on the Semantic Web with SWRL. **The Semantic Web – ISWC 2005**, [S.l.]. 2005. p. 974-986.

OGIELA, M. R.; TADEUSIEWICZ, R. Artificial intelligence structural imaging techniques in visual pattern analysis and medical data understanding. **Pattern Recognition**, Oxford. v. 36, n. 10, p. 2441-2452, 2003.

OSSOLA, J.; BRÉMOND, F.; THONNAT, M. A Communication Level in a Distributed Architecture for Object Recognition. In: 8th International Conference on Systems Research Informatics and Cybernetics. **Proceedings...** , [S.l.]. 1996.

PARSIA, B.; SIRIN, E.; KALYANPUR, A. Debugging OWL ontologies. In: 14th International Conference on World Wide Web. **Proceedings...** , USA: ACM, 2005. p. 633-640.

RANDELL, D. A.; CUI, Z.; COHN, A. G. A Spatial Logic based on Regions and Connection. In: 3rd Internation Conference on Knowledge Representation and Reasoning. **Proceedings...** , San Mateo 1992. p. 165–176.

RECTOR, A. **Representing Specified Values in OWL: "value partitions" and "value sets"**. 2005. Disponível em: <http://www.w3.org/TR/swbp-specified-values/>. Acesso em: 17 Fev 2009.

SANDAKLY, F.; GIRAUDON, G. Scene analysis system. In: IEEE International Conference in Image Processing, 1994. **Proceedings...** , [S.l.]. v. 3, 1994. p. 806-810 vol.3.

SANTIN, C. E. **Construtos ontológicos para representação simbólica de conhecimento visual**. 2008. 103 p. Dissertação (Mestrado em Computação) - Instituto de Informática, Universidade Federal do Rio Grande do Sul.

SCHREIBER, G. et al. **Knowledge Engineering and Management: The CommonKADS Methodology**. Cambridge, USA: The MIT Press, 1999,

SHANAHAN, M. Robotics and the Common Sense Informatics Situation. In: Spring Symposium on Planning with Incomplete Information for Robot Problems. **Proceedings...** , Menlo Park, USA: AAAI, 1996. p. 95-106.

SHANAHAN, M.; RANDELL, D. A Logic-Based Formulation of Active Visual Perception. In: Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004). **Proceedings...** , Menlo Park, Calif. 2004. p. 64-72.

SILVA, J. G. R. D. **Estudo de cicloestratigrafia nos depósitos eopermianos do Grupo Itararé, Bacia do Paraná, nos Estados de Santa Catarina e Rio Grande do Sul, baseado em dados de testemunho e de perfis de raios gama**. 2001. Dissertação (Mestrado em Geologia) - Instituto de Geociências, Universidade Federal do Rio Grande do Sul.

STUDER, R.; BENJAMINS, V. R.; FENSEL, D. Knowledge engineering: principles and methods. **Data & Knowledge Engineering**, [S.l.]. v. 25, n. 1-2, p. 161-197, 1998.

TSARKOV, D.; HORROCKS, I. FaCT++ Description Logic Reasoner: System Description. **Automated Reasoning**, Berlin: Springer, 2006. p. 292-297. (Lecture Notes in Computer Science, 4130).

VAN WAGONER, J. C. et al. **Siliciclastic sequence stratigraphy in well logs, cores, and outcrops**. Tulsa, US: AAPG, 1990,

WITHAGEN, P. J. **Object detection and segmentation for visual surveillance**. 2006. 173 p. Thesis (PhD in Computer Science) - University of Amsterdam.

YIP, K.; ZHAO, F. Spatial Aggregation: Theory and Applications. **Journal of Artificial Intelligence Research**, [S.l.]. v. 5, p. 1--26, 1996.

ANEXO METAMODELO DA LINUGAGEM OWL

Os algoritmos das máquinas MIS e MIV descritos na seção 3.2.5 fazem referencia a objetos do metamodelo de OWL a RDF. Embora este ele esteja implícito na definição da própria linguagem, um metamodelo para OWL está sendo estandardizado pela OMG (Object Managemnt Group) em conjunto com outras linguagens de representação (COLOMB et al., 2006). A Figura A.1 apresenta os objetos do metamodelo que permitem um melhor entendimento dos algoritmos descritos.

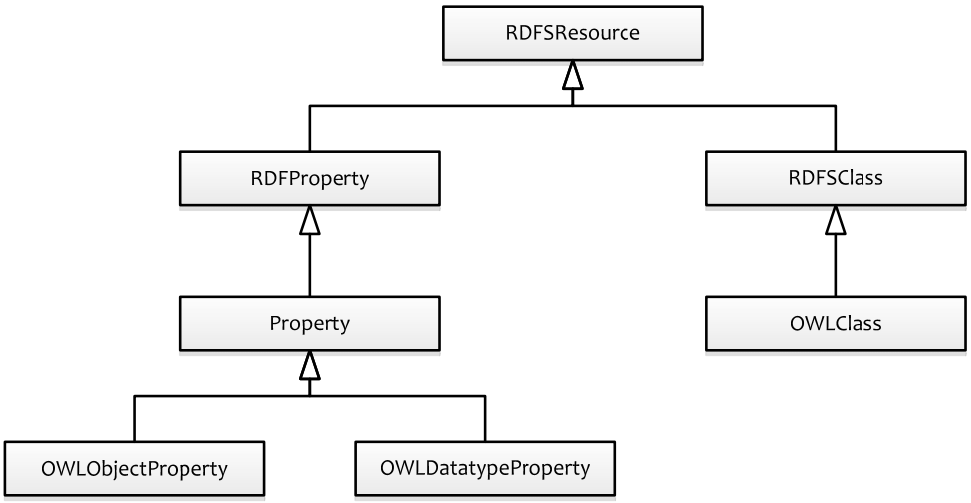

Figura A.1: Hierarquia de construtos de representação de OWL.

As primitivas da linguagem SWRL não são contemplados pela OMG. No entanto, segundo a descrição da linguagem, o construto `swrl:Imp` que aparece nos algoritmos das máquinas MIS e MIV descritas na seção 3.2.5 é uma instância de `OWLClass`. A sua função é descrever um instâncias de uma regra SWRL

APÊNDICE DESCRIÇÃO RDF/XML DOS MODELOS OWL DE CONHECIMENTO VISUAL DO SISTEMA INTELISTRATA

Neste apêndice encontra-se os modelos OWL que são utilizados pelo sistema InteliStrata. A representação utilizada é a RDF/XML padrão de OWL.

Todos os modelos apresentados, com exceção dos apresentados na seção B.5, são independentes de domínio e podem ser aplicados diretamente em outros sistemas.

B.1 Modelo do nível analógico

B.1.1 Modelo OWL do nível analógico

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
   xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns="http://www.owl-ontologies.com/AnalogLevelOntology.owl#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/AnalogLevelOntology.owl">
 <owl:Ontology rdf:about=""/>
 <owl:Class rdf:ID="PointSet">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="AnalogEntity"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="SignalRawDataContainer">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="RawDataContainer"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="IntervalSet">
  <rdfs:subClassOf rdf:resource="#PointSet"/>
 </owl:Class>
 <owl:Class rdf:ID="Interval">
  <rdfs:subClassOf>
```

```
    <owl:Restriction>
     <owl:onProperty>
      <owl:DatatypeProperty rdf:ID="hasArea"/>
     </owl:onProperty>
     <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >1</owl:maxCardinality>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty>
      <owl:DatatypeProperty rdf:ID="hasAbsoluteArea"/>
     </owl:onProperty>
     <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >1</owl:maxCardinality>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf rdf:resource="#PointSet"/>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty>
      <owl:DatatypeProperty rdf:ID="hasNoiseRate"/>
     </owl:onProperty>
     <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >1</owl:maxCardinality>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty>
      <owl:DatatypeProperty rdf:ID="hasLength"/>
     </owl:onProperty>
     <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >1</owl:maxCardinality>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty>
      <owl:DatatypeProperty rdf:ID="hasTendence"/>
     </owl:onProperty>
     <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >1</owl:maxCardinality>
    </owl:Restriction>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="FileSignalRawDataContainer">
   <rdfs:subClassOf rdf:resource="#SignalRawDataContainer"/>
  </owl:Class>
  <owl:Class rdf:ID="Point">
   <rdfs:subClassOf rdf:resource="#AnalogEntity"/>
  </owl:Class>
  <owl:ObjectProperty rdf:ID="hasRawDataContainer">
   <rdfs:range rdf:resource="#SignalRawDataContainer"/>
   <rdfs:domain rdf:resource="#AnalogEntity"/>
```

```
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasPoint">
 <rdfs:range rdf:resource="#Point"/>
 <rdfs:domain rdf:resource="#PointSet"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasStartPoint">
 <rdfs:domain rdf:resource="#Interval"/>
 <rdfs:subPropertyOf rdf:resource="#hasPoint"/>
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasCenterPoint">
 <rdfs:domain rdf:resource="#Interval"/>
 <rdfs:subPropertyOf rdf:resource="#hasPoint"/>
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasMaximumPoint">
 <rdfs:subPropertyOf rdf:resource="#hasPoint"/>
 <rdfs:domain rdf:resource="#PointSet"/>
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 <rdfs:range rdf:resource="#Point"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasEndPoint">
 <rdfs:domain rdf:resource="#Interval"/>
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 <rdfs:subPropertyOf rdf:resource="#hasPoint"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasMinimumPoint">
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 <rdfs:subPropertyOf rdf:resource="#hasPoint"/>
 <rdfs:range rdf:resource="#Point"/>
 <rdfs:domain rdf:resource="#PointSet"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:about="#hasAbsoluteArea">
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
 <rdfs:subPropertyOf>
  <owl:DatatypeProperty rdf:ID="hasAnalogicProperty"/>
 </rdfs:subPropertyOf>
 <rdfs:domain rdf:resource="#Interval"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="value">
 <rdfs:domain rdf:resource="#Point"/>
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#hasLength">
 <rdfs:subPropertyOf>
  <owl:DatatypeProperty rdf:about="#hasAnalogicProperty"/>
 </rdfs:subPropertyOf>
 <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
 <rdfs:domain rdf:resource="#Interval"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="#hasArea">
 <rdfs:domain rdf:resource="#Interval"/>
 <rdfs:subPropertyOf>
  <owl:DatatypeProperty rdf:about="#hasAnalogicProperty"/>
```

```
  </rdfs:subPropertyOf>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:ID="index">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
  <rdfs:domain rdf:resource="#Point"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:about="#hasAnalogicProperty">
  <rdfs:domain rdf:resource="#AnalogEntity"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:about="#hasNoiseRate">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
  <rdfs:domain rdf:resource="#Interval"/>
  <rdfs:subPropertyOf rdf:resource="#hasAnalogicProperty"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:about="#hasTendence">
  <rdfs:subPropertyOf rdf:resource="#hasAnalogicProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
  <rdfs:domain rdf:resource="#Interval"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:ID="hasLocationPath">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#anyURI"/>
  <rdfs:domain rdf:resource="#FileSignalRawDataContainer"/>
 </owl:DatatypeProperty>
</rdf:RDF>
```

B.1.2 Modelo OWL para processamento de sinais

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
   xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
   xmlns="http://www.owl-ontologies.com/SignalProcessingOntology.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:alo="http://www.owl-ontologies.com/AnalogLevelOntology.owl#"
  xml:base="http://www.owl-ontologies.com/SignalProcessingOntology.owl">
 <owl:Ontology rdf:about="">
  <owl:imports
        rdf:resource="http://www.owl-ontologies.com/AnalogLevelOntology.owl"/>
 </owl:Ontology>
 <owl:Class rdf:ID="PropertyExtractor">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="SignalProcessingFunctionality"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="SizePropertyExtractor">
  <rdfs:subClassOf rdf:resource="#PropertyExtractor"/>
 </owl:Class>
 <owl:Class rdf:ID="Segmentator">
```

```
  <rdfs:subClassOf rdf:resource="#SignalProcessingFunctionality"/>
 </owl:Class>
 <owl:Class rdf:ID="FourierSegmentator">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="BasicSegmentator"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:about="#BasicSegmentator">
  <rdfs:subClassOf rdf:resource="#Segmentator"/>
 </owl:Class>
 <owl:Class rdf:ID="WaveletSegmentator">
  <rdfs:subClassOf rdf:resource="#BasicSegmentator"/>
 </owl:Class>
 <owl:ObjectProperty rdf:ID="isExtractedWith">
  <rdfs:range rdf:resource="#SignalProcessingFunctionality"/>
  <rdfs:domain
     rdf:resource="http://www.owl-ontologies.com/
     AnalogLevelOntology.owl#AnalogEntity"/>
 </owl:ObjectProperty>
 <owl:DatatypeProperty rdf:ID="hasExtractionStrength">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
  <rdfs:subPropertyOf
      rdf:resource="http://www.owl-ontologies.com/
     AnalogLevelOntology.owl#hasAnalogicProperty"/>
 </owl:DatatypeProperty>
 <WaveletSegmentator rdf:ID="SmoothToothCurveSPF"/>
 <WaveletSegmentator rdf:ID="Symlet8SPF"/>
 <WaveletSegmentator rdf:ID="Gaussian2SPF"/>
</rdf:RDF>
```

B.2 Modelo do nível visual

O modelo no nível visual consiste em somente um modelo OWL.

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:ineq="http://www.owl-ontologies.com/InequalityOntology.owl#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
   xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns="http://www.owl-ontologies.com/VisualLevel.owl#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/VisualLevel.owl">
 <owl:Ontology rdf:about="">
  <owl:imports
     rdf:resource="http://www.owl-ontologies.com/InequalityOntology.owl"/>
 </owl:Ontology>
 <owl:Class rdf:ID="VerticalLine">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="Line"/>
  </rdfs:subClassOf>
```

```
</owl:Class>
<owl:Class rdf:ID="GenericCharacteristicValue">
 <owl:equivalentClass>
  <owl:Class>
   <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:ID="VeryStrongPattern">
     <ineq:greaterThan>
      <GenericCharacteristicValue rdf:ID="StrongPattern">
       <ineq:smallerThan rdf:resource="#VeryStrongPattern"/>
       <ineq:greaterThan>
        <GenericCharacteristicValue rdf:ID="MediumPattern">
         <ineq:greaterThan>
          <GenericCharacteristicValue rdf:ID="WeakPattern">
           <ineq:smallerThan rdf:resource="#MediumPattern"/>
           <ineq:greaterThan>
            <GenericCharacteristicValue rdf:ID="VeryWeakPattern">
             <ineq:smallerThan rdf:resource="#WeakPattern"/>
            </GenericCharacteristicValue>
           </ineq:greaterThan>
          </GenericCharacteristicValue>
         </ineq:greaterThan>
         <ineq:smallerThan rdf:resource="#StrongPattern"/>
        </GenericCharacteristicValue>
       </ineq:greaterThan>
      </GenericCharacteristicValue>
     </ineq:greaterThan>
    </GenericCharacteristicValue>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
    <GenericCharacteristicValue rdf:about="#VeryWeakPattern"/>
   </owl:oneOf>
  </owl:Class>
 </owl:equivalentClass>
 <rdfs:subClassOf>
  <owl:Class rdf:ID="CharacteristicValue"/>
 </rdfs:subClassOf>
</owl:Class>
<owl:Class>
 <owl:oneOf rdf:parseType="Collection">
  <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
  <GenericCharacteristicValue rdf:about="#MediumPattern"/>
  <GenericCharacteristicValue rdf:about="#WeakPattern"/>
 </owl:oneOf>
</owl:Class>
<owl:Class>
 <owl:oneOf rdf:parseType="Collection">
  <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
  <GenericCharacteristicValue rdf:about="#StrongPattern"/>
  <GenericCharacteristicValue rdf:about="#MediumPattern"/>
  <GenericCharacteristicValue rdf:about="#WeakPattern"/>
 </owl:oneOf>
</owl:Class>
<owl:Class rdf:about="#CharacteristicValue">
 <rdfs:subClassOf>
```

```
    <owl:Class rdf:ID="VisualPropertyValue"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
    <GenericCharacteristicValue rdf:about="#VeryWeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class rdf:ID="GeometricEntity">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="VisualEntity"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class rdf:ID="Point">
  <rdfs:subClassOf rdf:resource="#GeometricEntity"/>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
```

```
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
    <GenericCharacteristicValue rdf:about="#VeryWeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class rdf:about="#Line">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="OpenCurve"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="InclinedLine">
  <rdfs:subClassOf rdf:resource="#Line"/>
</owl:Class>
<owl:Class rdf:ID="SmoothToothCurve">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#OpenCurve"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="LengthValue">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#VisualPropertyValue"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
    <GenericCharacteristicValue rdf:about="#VeryWeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class rdf:ID="SinusoidCurve">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#OpenCurve"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="GenericLengthValue">
  <owl:equivalentClass>
```

```
    <owl:Class>
     <owl:oneOf rdf:parseType="Collection">
      <GenericLengthValue rdf:ID="ShortLength">
       <ineq:greaterThan>
        <GenericLengthValue rdf:ID="VeryShortLength">
         <ineq:smallerThan rdf:resource="#ShortLength"/>
        </GenericLengthValue>
       </ineq:greaterThan>
       <ineq:smallerThan>
        <GenericLengthValue rdf:ID="MediumLength">
         <ineq:greaterThan rdf:resource="#ShortLength"/>
         <ineq:smallerThan>
          <GenericLengthValue rdf:ID="LongLength">
           <ineq:smallerThan>
            <GenericLengthValue rdf:ID="VeryLongLength">
             <ineq:greaterThan rdf:resource="#LongLength"/>
            </GenericLengthValue>
           </ineq:smallerThan>
           <ineq:greaterThan rdf:resource="#MediumLength"/>
          </GenericLengthValue>
         </ineq:smallerThan>
        </GenericLengthValue>
       </ineq:smallerThan>
      </GenericLengthValue>
      <GenericLengthValue rdf:about="#VeryLongLength"/>
      <GenericLengthValue rdf:about="#LongLength"/>
      <GenericLengthValue rdf:about="#MediumLength"/>
      <GenericLengthValue rdf:about="#VeryShortLength"/>
     </owl:oneOf>
    </owl:Class>
  </owl:equivalentClass>
  <rdfs:subClassOf rdf:resource="#LengthValue"/>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
    <GenericCharacteristicValue rdf:about="#VeryWeakPattern"/>
  </owl:oneOf>
</owl:Class>
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <GenericCharacteristicValue rdf:about="#StrongPattern"/>
    <GenericCharacteristicValue rdf:about="#MediumPattern"/>
    <GenericCharacteristicValue rdf:about="#WeakPattern"/>
```

```
  </owl:oneOf>
 </owl:Class>
 <owl:Class>
  <owl:oneOf rdf:parseType="Collection">
   <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
   <GenericCharacteristicValue rdf:about="#MediumPattern"/>
   <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
 </owl:Class>
 <owl:Class rdf:ID="MinimumPoint">
  <rdfs:subClassOf rdf:resource="#Point"/>
 </owl:Class>
 <owl:Class rdf:ID="PointSet">
  <rdfs:subClassOf rdf:resource="#GeometricEntity"/>
 </owl:Class>
 <owl:Class rdf:ID="MaximumPoint">
  <rdfs:subClassOf rdf:resource="#Point"/>
 </owl:Class>
 <owl:Class rdf:about="#VisualPropertyValue">
  <owl:equivalentClass>
   <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#LengthValue"/>
     <owl:Class rdf:about="#CharacteristicValue"/>
    </owl:unionOf>
   </owl:Class>
  </owl:equivalentClass>
 </owl:Class>
 <owl:Class rdf:about="#OpenCurve">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="Curve"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class>
  <owl:oneOf rdf:parseType="Collection">
   <GenericCharacteristicValue rdf:about="#StrongPattern"/>
   <GenericCharacteristicValue rdf:about="#MediumPattern"/>
   <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
 </owl:Class>
 <owl:Class>
  <owl:oneOf rdf:parseType="Collection">
   <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
   <GenericCharacteristicValue rdf:about="#StrongPattern"/>
   <GenericCharacteristicValue rdf:about="#MediumPattern"/>
   <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
 </owl:Class>
 <owl:Class>
  <owl:oneOf rdf:parseType="Collection">
   <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
   <GenericCharacteristicValue rdf:about="#StrongPattern"/>
   <GenericCharacteristicValue rdf:about="#MediumPattern"/>
   <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  </owl:oneOf>
```

```
</owl:Class>
<owl:Class>
 <owl:oneOf rdf:parseType="Collection">
  <GenericCharacteristicValue rdf:about="#StrongPattern"/>
  <GenericCharacteristicValue rdf:about="#MediumPattern"/>
  <GenericCharacteristicValue rdf:about="#WeakPattern"/>
 </owl:oneOf>
</owl:Class>
<owl:Class rdf:ID="SigmoidCurve">
 <rdfs:subClassOf rdf:resource="#OpenCurve"/>
</owl:Class>
<owl:Class>
 <owl:oneOf rdf:parseType="Collection">
  <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
  <GenericCharacteristicValue rdf:about="#StrongPattern"/>
  <GenericCharacteristicValue rdf:about="#MediumPattern"/>
  <GenericCharacteristicValue rdf:about="#WeakPattern"/>
  <GenericCharacteristicValue rdf:about="#VeryWeakPattern"/>
 </owl:oneOf>
</owl:Class>
<owl:Class>
 <owl:oneOf rdf:parseType="Collection">
  <GenericCharacteristicValue rdf:about="#StrongPattern"/>
  <GenericCharacteristicValue rdf:about="#MediumPattern"/>
  <GenericCharacteristicValue rdf:about="#WeakPattern"/>
 </owl:oneOf>
</owl:Class>
<owl:Class rdf:ID="GaussianCurve">
 <rdfs:subClassOf rdf:resource="#OpenCurve"/>
</owl:Class>
<owl:Class>
 <owl:oneOf rdf:parseType="Collection">
  <GenericCharacteristicValue rdf:about="#VeryStrongPattern"/>
  <GenericCharacteristicValue rdf:about="#StrongPattern"/>
  <GenericCharacteristicValue rdf:about="#MediumPattern"/>
  <GenericCharacteristicValue rdf:about="#WeakPattern"/>
 </owl:oneOf>
</owl:Class>
<owl:Class rdf:about="#Curve">
 <rdfs:subClassOf rdf:resource="#PointSet"/>
</owl:Class>
<owl:Class rdf:ID="TriangularCurve">
 <rdfs:subClassOf rdf:resource="#OpenCurve"/>
</owl:Class>
<owl:Class rdf:ID="HorizontalLine">
 <rdfs:subClassOf rdf:resource="#Line"/>
</owl:Class>
<owl:ObjectProperty rdf:ID="isOver">
 <owl:inverseOf>
  <owl:ObjectProperty rdf:ID="isUnder"/>
 </owl:inverseOf>
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:ID="hasOrientationPositionTo"/>
 </rdfs:subPropertyOf>
</owl:ObjectProperty>
```

```
<owl:ObjectProperty rdf:ID="isAtRightOf">
 <owl:inverseOf>
  <owl:ObjectProperty rdf:ID="isAtLeftOf"/>
 </owl:inverseOf>
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:about="#hasOrientationPositionTo"/>
 </rdfs:subPropertyOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isDiscreteOf">
 <rdfs:domain rdf:resource="#VisualEntity"/>
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:ID="hasTopologicalRelationWith"/>
 </rdfs:subPropertyOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasSpatialRelationWith">
 <rdfs:range rdf:resource="#VisualEntity"/>
 <rdfs:domain rdf:resource="#VisualEntity"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasSmoothness">
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:ID="hasVisualProperty"/>
 </rdfs:subPropertyOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasConvexity">
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:about="#hasVisualProperty"/>
 </rdfs:subPropertyOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#hasOrientationPositionTo">
 <rdfs:subPropertyOf rdf:resource="#hasSpatialRelationWith"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#isAtLeftOf">
 <rdfs:subPropertyOf rdf:resource="#hasOrientationPositionTo"/>
 <owl:inverseOf rdf:resource="#isAtRightOf"/>
 <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#hasTopologicalRelationWith">
 <rdfs:subPropertyOf rdf:resource="#hasSpatialRelationWith"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasForNonTangentialProperPartOf">
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:ID="hasForProperPart"/>
 </rdfs:subPropertyOf>
 <owl:inverseOf>
  <owl:ObjectProperty rdf:ID="isNonTangentialProperPartOf"/>
 </owl:inverseOf>
 <rdfs:domain rdf:resource="#VisualEntity"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#hasForProperPart">
 <rdfs:subPropertyOf rdf:resource="#hasTopologicalRelationWith"/>
 <owl:inverseOf>
  <owl:ObjectProperty rdf:ID="isProperPartOf"/>
 </owl:inverseOf>
 <rdfs:domain rdf:resource="#VisualEntity"/>
</owl:ObjectProperty>
```

```
<owl:ObjectProperty rdf:about="#isUnder">
 <owl:inverseOf rdf:resource="#isOver"/>
 <rdfs:subPropertyOf rdf:resource="#hasOrientationPositionTo"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasCurvature">
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:about="#hasVisualProperty"/>
 </rdfs:subPropertyOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasHeight">
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:about="#hasVisualProperty"/>
 </rdfs:subPropertyOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasLength">
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:about="#hasVisualProperty"/>
 </rdfs:subPropertyOf>
 <rdfs:range rdf:resource="#LengthValue"/>
 <rdfs:domain rdf:resource="#Curve"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isTopologicallyEqualsTo">
 <rdfs:subPropertyOf rdf:resource="#hasTopologicalRelationWith"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="partialOverlaps">
 <rdfs:subPropertyOf rdf:resource="#hasTopologicalRelationWith"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#isNonTangentialProperPartOf">
 <rdfs:subPropertyOf>
  <owl:ObjectProperty rdf:about="#isProperPartOf"/>
 </rdfs:subPropertyOf>
 <owl:inverseOf rdf:resource="#hasForNonTangentialProperPartOf"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#isProperPartOf">
 <owl:inverseOf rdf:resource="#hasForProperPart"/>
 <rdfs:subPropertyOf rdf:resource="#hasTopologicalRelationWith"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isExternallyConnectedTo">
 <rdfs:subPropertyOf rdf:resource="#isDiscreteOf"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#hasVisualProperty">
 <rdfs:range rdf:resource="#VisualPropertyValue"/>
 <rdfs:domain rdf:resource="#VisualEntity"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isTangentialProperPartOf">
 <rdfs:subPropertyOf rdf:resource="#isProperPartOf"/>
 <owl:inverseOf>
  <owl:ObjectProperty rdf:ID="hasForTangentialProperPart"/>
 </owl:inverseOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="idDisconnectedOf">
 <rdfs:subPropertyOf rdf:resource="#isDiscreteOf"/>
 <rdfs:domain rdf:resource="#VisualEntity"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#hasForTangentialProperPart">
```

```
    <rdfs:domain rdf:resource="#VisualEntity"/>
    <owl:inverseOf rdf:resource="#isTangentialProperPartOf"/>
    <rdfs:subPropertyOf rdf:resource="#hasForProperPart"/>
  </owl:ObjectProperty>
  <owl:FunctionalProperty rdf:ID="hasCharacteristic">
    <rdfs:range rdf:resource="#GenericCharacteristicValue"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:subPropertyOf rdf:resource="#hasVisualProperty"/>
    <rdfs:domain rdf:resource="#GeometricEntity"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="hasGaussianCurveCharacteristic">
    <rdfs:subPropertyOf rdf:resource="#hasCharacteristic"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="#GaussianCurve"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="hasSmoothToothCurveCharacteristic">
    <rdfs:domain rdf:resource="#SmoothToothCurve"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:subPropertyOf rdf:resource="#hasCharacteristic"/>
  </owl:FunctionalProperty>
  <owl:AllDifferent/>
</rdf:RDF>
```

B.3 Modelo do nível semântico

O modelo nível semântico consiste em dois modelos OWL.

B.3.1 Modelo OWL do nível semântico

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns="http://www.owl-ontologies.com/SemanticLevelOntology.owl#"
  xml:base="http://www.owl-ontologies.com/SemanticLevelOntology.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="SemanticEntity"/>
</rdf:RDF>
```

B.3.2 Modelo OWL do domínio

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:part
      ="http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
```

```
   xmlns="http://www.owl-ontologies.com/StratigraphyOntology.owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/StratigraphyOntology.owl">
  <owl:Ontology rdf:about="">
   <owl:imports
rdf:resource="http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.o
wl"/>
  </owl:Ontology>
  <owl:Class rdf:ID="FloodingSurface">
   <rdfs:subClassOf>
    <owl:Class rdf:ID="StratigraphicSurface"/>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class>
   <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:ID="Parassequence"/>
    <owl:Class rdf:about="#FloodingSurface"/>
    <owl:Class rdf:ID="SequenceBoundary"/>
   </owl:unionOf>
  </owl:Class>
  <owl:Class rdf:ID="MaximumFloodingSurface">
   <rdfs:subClassOf rdf:resource="#FloodingSurface"/>
  </owl:Class>
  <owl:Class rdf:about="#Parassequence">
   <owl:disjointWith>
    <owl:Class rdf:ID="Sequence"/>
   </owl:disjointWith>
   <rdfs:subClassOf>
    <owl:Class rdf:ID="StratigraphicUnit"/>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty
rdf:resource="http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.o
wl#hasPart"/>
     <owl:allValuesFrom rdf:resource="#FloodingSurface"/>
    </owl:Restriction>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="StratigraphicEntity"/>
  <owl:Class rdf:about="#SequenceBoundary">
   <rdfs:subClassOf>
    <owl:Class rdf:about="#StratigraphicSurface"/>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class>
   <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Parassequence"/>
    <owl:Class rdf:about="#MaximumFloodingSurface"/>
   </owl:unionOf>
  </owl:Class>
  <owl:Class rdf:about="#Sequence">
```

```
   <owl:disjointWith rdf:resource="#Parassequence"/>
   <rdfs:subClassOf>
    <owl:Class rdf:about="#StratigraphicUnit"/>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:allValuesFrom>
      <owl:Class>
       <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:ID="SystemTract"/>
        <owl:Class rdf:about="#MaximumFloodingSurface"/>
        <owl:Class rdf:about="#SequenceBoundary"/>
       </owl:unionOf>
      </owl:Class>
     </owl:allValuesFrom>
     <owl:onProperty
rdf:resource="http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.o
wl#hasPart"/>
    </owl:Restriction>
   </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:about="#StratigraphicSurface">
   <rdfs:subClassOf rdf:resource="#StratigraphicEntity"/>
   <owl:disjointWith>
    <owl:Class rdf:about="#StratigraphicUnit"/>
   </owl:disjointWith>
 </owl:Class>
 <owl:Class rdf:about="#StratigraphicUnit">
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:allValuesFrom>
      <owl:Class>
       <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#StratigraphicSurface"/>
        <owl:Class rdf:about="#StratigraphicUnit"/>
       </owl:unionOf>
      </owl:Class>
     </owl:allValuesFrom>
     <owl:onProperty
rdf:resource="http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.o
wl#hasPart"/>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf rdf:resource="#StratigraphicEntity"/>
   <owl:disjointWith rdf:resource="#StratigraphicSurface"/>
 </owl:Class>
 <owl:Class>
   <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Parassequence"/>
    <owl:Class rdf:about="#MaximumFloodingSurface"/>
    <owl:Class rdf:about="#SequenceBoundary"/>
   </owl:unionOf>
 </owl:Class>
 <owl:Class rdf:about="#SystemTract">
   <rdfs:subClassOf rdf:resource="#StratigraphicUnit"/>
```

```
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty
rdf:resource="http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.o
wl#hasPart"/>
      <owl:allValuesFrom rdf:resource="#Parassequence"/>
    </owl:Restriction>
   </rdfs:subClassOf>
  </owl:Class>
</rdf:RDF>
```

B.4 Modelo de ancoramento simbólico

O modelo de ancoramento simbólico consiste em um modelo OWL.

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
   xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
   xmlns:swrlx="http://swrl.stanford.edu/ontologies/built-ins/3.3/swrlx.owl#"
   xmlns:spo="http://www.owl-ontologies.com/SignalProcessingOntology.owl#"
   xmlns:swrlm="http://swrl.stanford.edu/ontologies/built-ins/3.4/swrlm.owl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns:vlo="http://www.owl-ontologies.com/VisualLevel.owl#"
   xmlns:temporal="http://swrl.stanford.edu/ontologies/built-ins/3.3/temporal.owl#"
   xmlns:tbox="http://swrl.stanford.edu/ontologies/built-ins/3.3/tbox.owl#"
   xmlns:slo="http://www.owl-ontologies.com/SemanticLevelOntology.owl#"
   xmlns:fuzzy="http://swrl.stanford.edu/ontologies/built-ins/3.3/fuzzy.owl#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns="http://www.owl-ontologies.com/MappingOntology.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:sqwrl="http://sqwrl.stanford.edu/ontologies/built-ins/3.4/sqwrl.owl#"
   xmlns:abox="http://swrl.stanford.edu/ontologies/built-ins/3.3/abox.owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:swrla="http://swrl.stanford.edu/ontologies/3.3/swrla.owl#"
  xml:base="http://www.owl-ontologies.com/MappingOntology.owl">
  <owl:Ontology rdf:about="">
   <owl:imports
      rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/fuzzy.owl"/>
   <owl:imports
      rdf:resource="http://www.owl-ontologies.com/SemanticLevelOntology.owl"/>
   <owl:imports
      rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/swrlx.owl"/>
   <owl:imports
      rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/temporal.owl"/>
   <owl:imports
      rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.4/swrlm.owl"/>
   <owl:imports rdf:resource="http://swrl.stanford.edu/ontologies/3.3/swrla.owl"/>
   <owl:imports
      rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/abox.owl"/>
   <owl:imports
      rdf:resource="http://sqwrl.stanford.edu/ontologies/built-ins/3.4/sqwrl.owl"/>
   <owl:imports
```

```
      rdf:resource="http://www.owl-ontologies.com/VisualLevel.owl"/>
   <owl:imports
      rdf:resource="http://www.owl-ontologies.com/SignalProcessingOntology.owl"/>
   <owl:imports
      rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/tbox.owl"/>
  </owl:Ontology>
  <rdfs:Class
rdf:about="http://swrl.stanford.edu/ontologies/3.3/swrla.owl#RuleGroup"/>
  <owl:ObjectProperty rdf:ID="groundedByDG">
   <rdfs:range
rdf:resource="http://swrl.stanford.edu/ontologies/3.3/swrla.owl#RuleGroup"/>
   <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="groundingPrimitive"/>
   </owl:inverseOf>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="upwardMapLink">
   <rdfs:subPropertyOf>
    <owl:ObjectProperty rdf:ID="mappingLink"/>
   </rdfs:subPropertyOf>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="downwardMapLink">
   <rdfs:subPropertyOf rdf:resource="#mappingLink"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="mappingAnalogicToVisual">
   <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="mappingVisualToAnalogic"/>
   </owl:inverseOf>
   <rdfs:range
      rdf:resource="http://www.owl-ontologies.com/VisualLevel.owl#VisualEntity"/>
   <rdfs:subPropertyOf rdf:resource="#upwardMapLink"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#mappingVisualToAnalogic">
   <owl:inverseOf rdf:resource="#mappingAnalogicToVisual"/>
   <rdfs:domain
      rdf:resource="http://www.owl-ontologies.com/VisualLevel.owl#VisualEntity"/>
   <rdfs:subPropertyOf rdf:resource="#downwardMapLink"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="mappingSemanticToVisual">
   <rdfs:range
      rdf:resource="http://www.owl-ontologies.com/VisualLevel.owl#VisualEntity"/>
   <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="mappingVisualToSemantic"/>
   </owl:inverseOf>
   <rdfs:domain rdf:resource
     ="http://www.owl-ontologies.com/SemanticLevelOntology.owl#SemanticEntity"/>
   <rdfs:subPropertyOf rdf:resource="#downwardMapLink"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#mappingVisualToSemantic">
   <rdfs:range rdf:resource
     ="http://www.owl-ontologies.com/SemanticLevelOntology.owl#SemanticEntity"/>
   <owl:inverseOf rdf:resource="#mappingSemanticToVisual"/>
   <rdfs:domain rdf:resource
     ="http://www.owl-ontologies.com/VisualLevel.owl#VisualEntity"/>
   <rdfs:subPropertyOf rdf:resource="#upwardMapLink"/>
  </owl:ObjectProperty>
```

```
 <owl:ObjectProperty rdf:about="#groundingPrimitive">
  <rdfs:domain
    rdf:resource="http://swrl.stanford.edu/ontologies/3.3/swrla.owl#RuleGroup"/>
  <owl:inverseOf rdf:resource="#groundedByDG"/>
 </owl:ObjectProperty>
</rdf:RDF>
```

B.5 Modelos Adicionais

Os modelos adicionais constituem a ontologia de sistema e outra ontologia de relações de ordem.

B.5.1 Modelo OWL do sistema InteliStrata

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.owl-ontologies.com/InteliSrata.owl#"
    xml:base="http://www.owl-ontologies.com/InteliSrata.owl"
    xmlns:slo="http://www.owl-ontologies.com/SemanticLevelOntology.owl#"
    xmlns:dom="http://www.owl-ontologies.com/StratigraphyOntology.owl#"
    xmlns:spo="http://www.owl-ontologies.com/SignalProcessingOntology.owl#"
    xmlns:abox="http://swrl.stanford.edu/ontologies/built-ins/3.3/abox.owl#"
    xmlns:sqwrl="http://sqwrl.stanford.edu/ontologies/built-ins/3.4/sqwrl.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:fuzzy="http://swrl.stanford.edu/ontologies/built-ins/3.3/fuzzy.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:alo="http://www.owl-ontologies.com/AnalogLevelOntology.owl#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:swrlx="http://swrl.stanford.edu/ontologies/built-ins/3.3/swrlx.owl#"
    xmlns:mapo="http://www.owl-ontologies.com/MappingOntology.owl#"
    xmlns:vlo="http://www.owl-ontologies.com/VisualLevel.owl#"
    xmlns:ineq="http://www.owl-ontologies.com/InequalityOntology.owl#"
    xmlns:swrlm="http://swrl.stanford.edu/ontologies/built-ins/3.4/swrlm.owl#"
    xmlns:temporal="http://swrl.stanford.edu/ontologies/built-ins/3.3/temporal.owl#"
    xmlns:swrli="http://swrl.stanford.edu/ontologies/built-ins/3.4/swrli.owl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:tbox="http://swrl.stanford.edu/ontologies/built-ins/3.3/tbox.owl#"
    xmlns:part=
      "http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.owl#"
    xmlns:swrlineq="http://swrl.stanford.edu/ontologies/built-ins/3.3/swrli.owl#"
    xmlns:swrla="http://swrl.stanford.edu/ontologies/3.3/swrla.owl#">
   <owl:Ontology rdf:about="">
      <owl:imports
       rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/abox.owl"/>
      <owl:imports
       rdf:resource="http://sqwrl.stanford.edu/ontologies/built-ins/3.4/sqwrl.owl"/>
      <owl:imports
      rdf:resource="http://swrl.stanford.edu/ontologies/3.3/swrla.owl"/>
      <owl:imports
       rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/temporal.owl"/>
      <owl:imports
```

```
    rdf:resource="http://www.owl-ontologies.com/StratigraphyOntology.owl"/>
    <owl:imports
    rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/swrli.owl"/>
    <owl:imports
    rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/swrlx.owl"/>
    <owl:imports
   rdf:resource="http://www.owl-ontologies.com/MappingOntology.owl"/>
    <owl:imports
    rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/tbox.owl"/>
    <owl:imports
    rdf:resource="http://swrl.stanford.edu/ontologies/built-ins/3.3/fuzzy.owl"/>
</owl:Ontology>
    <rdf:Description rdf:about="&dom;MaximumFloodingSurface">
    <mapo:groundedByDG rdf:resource="#sdgMaximumFloodingSurface"/>
</rdf:Description>
<rdf:Description rdf:about="&dom;Parassequence">
    <mapo:groundedByDG rdf:resource="#sdgParassequence"/>
</rdf:Description>
<rdf:Description rdf:about="&dom;Sequence">
    <mapo:groundedByDG rdf:resource="#sdgDomSequence"/>
</rdf:Description>
<ParassequenceLengthValue rdf:ID="LongLengthParassequence">
    <ineq:greaterThan rdf:resource="#MediumLengthParassequence"/>
</ParassequenceLengthValue>
<SequenceLengthValue rdf:ID="LongLengthSequence">
    <ineq:smallerThan rdf:resource="#VeryLongLengthSequence"/>
    <ineq:greaterThan rdf:resource="#MediumLengthSequence"/>
</SequenceLengthValue>
<ParassequenceLengthValue rdf:ID="MediumLengthParassequence">
    <ineq:smallerThan rdf:resource="#LongLengthParassequence"/>
    <ineq:greaterThan rdf:resource="#ShortLengthParassequence"/>
</ParassequenceLengthValue>
<SequenceLengthValue rdf:ID="MediumLengthSequence">
    <ineq:smallerThan rdf:resource="#LongLengthSequence"/>
    <ineq:greaterThan rdf:resource="#ShortLengthSequence"/>
</SequenceLengthValue>
<owl:Class rdf:ID="ParassequenceLengthValue">
    <owl:equivalentClass>
      <owl:Class>
        <owl:oneOf rdf:parseType="Collection">
          <rdf:Description rdf:about="#LongLengthParassequence"/>
          <rdf:Description rdf:about="#MediumLengthParassequence"/>
          <rdf:Description rdf:about="#ShortLengthParassequence"/>
        </owl:oneOf>
      </owl:Class>
    </owl:equivalentClass>
    <rdfs:subClassOf rdf:resource="&vlo;LengthValue"/>
</owl:Class>
<swrla:RuleGroup rdf:ID="sdgDomSequence">
    <swrla:isRuleGroupEnabled
            rdf:datatype="&xsd;boolean">true</swrla:isRuleGroupEnabled>
    <mapo:groundingPrimitive rdf:resource="&dom;Sequence"/>
</swrla:RuleGroup>
<swrla:RuleGroup rdf:ID="sdgGaussianCurve">
    <swrla:isRuleGroupEnabled
```

```
            rdf:datatype="&xsd;boolean">true</swrla:isRuleGroupEnabled>
    <mapo:groundingPrimitive rdf:resource="&vlo;GaussianCurve"/>
</swrla:RuleGroup>
<swrla:RuleGroup rdf:ID="sdgMaximumFloodingSurface">
    <swrla:isRuleGroupEnabled
            rdf:datatype="&xsd;boolean">true</swrla:isRuleGroupEnabled>
    <mapo:groundingPrimitive rdf:resource="&dom;MaximumFloodingSurface"/>
</swrla:RuleGroup>
<swrla:RuleGroup rdf:ID="sdgMaximumPoint">
    <swrla:isRuleGroupEnabled
            rdf:datatype="&xsd;boolean">true</swrla:isRuleGroupEnabled>
    <mapo:groundingPrimitive rdf:resource="&vlo;MaximumPoint"/>
</swrla:RuleGroup>
<swrla:RuleGroup rdf:ID="sdgParassequence">
    <swrla:isRuleGroupEnabled
            rdf:datatype="&xsd;boolean">true</swrla:isRuleGroupEnabled>
    <mapo:groundingPrimitive rdf:resource="&dom;Parassequence"/>
</swrla:RuleGroup>
<swrla:RuleGroup rdf:ID="sdgSmoothToothCurve">
    <swrla:isRuleGroupEnabled
            rdf:datatype="&xsd;boolean">true</swrla:isRuleGroupEnabled>
    <mapo:groundingPrimitive rdf:resource="&vlo;SmoothToothCurve"/>
</swrla:RuleGroup>
<swrla:RuleGroup rdf:ID="sdgVloLength">
    <swrla:isRuleGroupEnabled
            rdf:datatype="&xsd;boolean">true</swrla:isRuleGroupEnabled>
    <mapo:groundingPrimitive rdf:resource="&vlo;hasLength"/>
</swrla:RuleGroup>
<swrla:RuleGroup rdf:ID="sdgVloSpatialRelation">
    <swrla:isRuleGroupEnabled
            rdf:datatype="&xsd;boolean">true</swrla:isRuleGroupEnabled>
    <mapo:groundingPrimitive rdf:resource="&vlo;hasVisualProperty"/>
</swrla:RuleGroup>
<owl:Class rdf:ID="SequenceLengthValue">
    <owl:equivalentClass>
        <owl:Class>
            <owl:oneOf rdf:parseType="Collection">
                <rdf:Description rdf:about="#LongLengthSequence"/>
                <rdf:Description rdf:about="#MediumLengthSequence"/>
                <rdf:Description rdf:about="#ShortLengthSequence"/>
                <rdf:Description rdf:about="#VeryLongLengthSequence"/>
                <rdf:Description rdf:about="#VeryShortLengthSequence"/>
            </owl:oneOf>
        </owl:Class>
    </owl:equivalentClass>
    <rdfs:subClassOf rdf:resource="&vlo;LengthValue"/>
</owl:Class>
<ParassequenceLengthValue rdf:ID="ShortLengthParassequence">
    <ineq:smallerThan rdf:resource="#MediumLengthParassequence"/>
</ParassequenceLengthValue>
<SequenceLengthValue rdf:ID="ShortLengthSequence">
    <ineq:smallerThan rdf:resource="#MediumLengthSequence"/>
    <ineq:greaterThan rdf:resource="#VeryShortLengthSequence"/>
</SequenceLengthValue>
<SequenceLengthValue rdf:ID="VeryLongLengthSequence">
```

```
      <ineq:greaterThan rdf:resource="#LongLengthSequence"/>
    </SequenceLengthValue>
    <SequenceLengthValue rdf:ID="VeryShortLengthSequence">
      <ineq:smallerThan rdf:resource="#ShortLengthSequence"/>
    </SequenceLengthValue>
    <rdf:Description rdf:about="&vlo;GaussianCurve">
      <mapo:groundedByDG rdf:resource="#sdgGaussianCurve"/>
    </rdf:Description>
    <rdf:Description rdf:about="&vlo;hasLength">
      <rdfs:domain rdf:resource="&vlo;Curve"/>
      <mapo:groundedByDG rdf:resource="#sdgVloLength"/>
    </rdf:Description>
    <rdf:Description rdf:about="&vlo;hasVisualProperty">
      <rdfs:domain rdf:resource="&vlo;VisualEntity"/>
      <mapo:groundedByDG rdf:resource="#sdgVloSpatialRelation"/>
    </rdf:Description>
    <rdf:Description rdf:about="&vlo;hasSmoothToothCurveCharacteristic"/>
    <rdf:Description rdf:about="&vlo;MaximumPoint">
      <mapo:groundedByDG rdf:resource="#sdgMaximumPoint"/>
    </rdf:Description>
    <rdf:Description rdf:about="&vlo;SmoothToothCurve">
      <mapo:groundedByDG rdf:resource="#sdgSmoothToothCurve"/>
    </rdf:Description>
</rdf:RDF>
```

B.5.2 Modelo OWL de relações de ordem (desigualdades)

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns="http://www.owl-ontologies.com/InequalityOntology.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/InequalityOntology.owl">
  <owl:Ontology rdf:about=""/>
  <owl:TransitiveProperty rdf:ID="greaterThan">
    <rdfs:subPropertyOf>
      <owl:TransitiveProperty rdf:ID="inequalityRelation"/>
    </rdfs:subPropertyOf>
    <owl:inverseOf>
      <owl:TransitiveProperty rdf:ID="smallerThan"/>
    </owl:inverseOf>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  </owl:TransitiveProperty>
  <owl:TransitiveProperty rdf:about="#smallerThan">
    <rdfs:subPropertyOf>
      <owl:TransitiveProperty rdf:about="#inequalityRelation"/>
    </rdfs:subPropertyOf>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <owl:inverseOf rdf:resource="#greaterThan"/>
```

```
  </owl:TransitiveProperty>
  <owl:TransitiveProperty rdf:about="#inequalityRelation">
   <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  </owl:TransitiveProperty>
  <owl:SymmetricProperty rdf:ID="equals">
   <owl:inverseOf rdf:resource="#equals"/>
   <rdfs:subPropertyOf rdf:resource="#inequalityRelation"/>
   <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
   <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  </owl:SymmetricProperty>
</rdf:RDF>
```

B.5.3 Detectores simbólicos

Embora os detectores simbólicos pertençam ao modelo OWL do sistema InteliStrata, a sua codificação RDF/XML é consideravelmente prolixa. Por isso, somente a representação direta dos detectores é descrita aqui.

A Tabela B.1 na página seguinte apresenta os detectores e a qual primitiva estão relacionados via grupos de detectores.

Tabela B.1: Detectores simbólicos e relação com primitivas ancoradas.

Primitiva	Detector Simbólico (regra SWRL)
dom:Sequence	mapo:mappingSemanticToVisual(?de, ?ve) ∧ vlo:GaussianCurve(?ve) ∧ vlo:hasGaussianCurveCharacteristic(?ve, ?pattern) ∧ swrlineq:greaterThanOrEqual(?pattern, vlo:StrongPattern) ∧ vlo:hasLength(?ve, ?length) ∧ swrlineq:greaterThanOrEqual(?length, MediumLengthSequence) → dom:Sequence(?de)
dom:Parassequence	mapo:mappingSemanticToVisual(?de, ?ve) ∧ vlo:SmoothToothCurve(?ve) ∧ vlo:hasSmoothToothCurveCurveCharacteristic(?ve, ?pattern) ∧ swrlineq:greaterThanOrEqual(?pattern, vlo:MediumPattern) ∧ vlo:hasLength(?ve, ?length) ∧ swrlineq:greaterThanOrEqual(?length, ShortLengthParassequence) ∧ vlo:isProperPartOf(?ve, ?sequenceVe) ∧ mapo:mappingSemanticToVisual(?sequenceDe, ?sequenceVe) ∧ dom:Sequence(?sequenceDe) → dom:Parassequence(?de) ∧ part:hasPart_directly(?sequenceDe, ?de)
dom:MaximumFloodingSurface	mapo:mappingSemanticToVisual(?de, ?ve) ∧ vlo:MaximumPoint(?ve) ∧ vlo:isProperPartOf(?ve, ?sequenceVe) ∧ mapo:mappingSemanticToVisual(?sequenceDe, ?sequenceVe) ∧ dom:Sequence(?sequenceDe) → dom:MaximumFloodingSurface(?de) ∧ part:hasPart_directly(?sequenceDe, ?de)
vlo:MaximumPoint	mapo:mappingVisualToAnalogic(?ve, ?ae) ∧ alo:Interval(?ae2) ∧ alo:hasMaximumPoint(?ae2, ?ae) → vlo:MaximumPoint(?ve)
vlo:hasLength	mapo:mappingVisualToAnalogic(?ve, ?al) ∧ alo:Interval(?al) ∧ alo:hasLength(?al, ?value) ∧ fuzzy:match(?value, ?fv, "[5, 10, 50, 60]", ShortLengthParassequence, "[50, 60, 140, 150]", MediumLengthParassequence, "[140, 150, 200, 250]", LongLengthSequence) → vlo:hasLength(?ve, ?fv)
	mapo:mappingVisualToAnalogic(?ve, ?al) ∧ alo:Interval(?al) ∧ alo:hasLength(?al, ?value) ∧ fuzzy:match(?value, ?fv, "[0, 0, 66, 82]", VeryShortLengthSequence, "[66, 82, 146, 210]", ShortLengthSequence, "[146, 210, 466, 722]", MediumLengthSequence, "[466, 722, 1746, 2770]", LongLengthSequence, "[1746, 2770, 5000, 7500]", VeryLongLengthSequence) → vlo:hasLength(?ve, ?fv)
vlo:SmoothToothCurve	mapo:mappingVisualToAnalogic(?ve, ?al) ∧ alo:Interval(?al) ∧ spo:isExtractedWith(?al, spo:SmoothToothCurveSPF) ∧ spo:hasExtractionStrength(?al, ?vstrength) ∧ fuzzy:match(?vstrength, ?strength, "[0, 0, 0.2, 0.3]", vlo:VeryWeakPattern, "[0.2, 0.3, 0.4, 0.5]", vlo:WeakPattern, "[0.4, 0.5, 0.6, 0.7]", vlo:MediumPattern, "[0.6, 0.7, 0.8, 0.9]", vlo:StrongPattern, "[0.8, 0.9, 1, 1]", vlo:VeryStrongPattern) → vlo:SmoothToothCurve(?ve) ∧ vlo:hasSmoothToothCurveCharacteristic(?ve, ?strength)
vlo:GaussianCurve	mapo:mappingVisualToAnalogic(?ve, ?al) ∧ alo:Interval(?al) ∧ spo:isExtractedWith(?al, spo:Gaussian2SPF) ∧ spo:hasExtractionStrength(?al, ?vstrength) ∧ fuzzy:match(?vstrength, ?strength, "[0, 0, 0.2, 0.3]", vlo:VeryWeakPattern, "[0.2, 0.3, 0.4, 0.5]", vlo:WeakPattern, "[0.4, 0.5, 0.6, 0.7]", vlo:MediumPattern, "[0.6, 0.7, 0.8, 0.9]", vlo:StrongPattern, "[0.8, 0.9, 1, 1]", vlo:VeryStrongPattern) → vlo:GaussianCurve(?ve) ∧ vlo:hasGaussianCurveCharacteristic(?ve, ?strength)

vlo:hasSpatialRelationWith	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasCenterPoint(?al1, ?cp1) ∧ alo:index(?cp1, ?vx1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Interval(?al2) ∧ alo:hasCenterPoint(?al2, ?cp2) ∧ alo:index(?cp2, ?vx2) ∧ swrlb:greaterThanOrEqual(?vx1, ?vx2) → vlo:isAtRightOf(?ve1, ?ve2) ∧ vlo:isAtLeftOf(?ve2, ?ve1)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasStartPoint(?al1, ?sp1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?sp1, ?vsx1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Interval(?al2) ∧ alo:hasStartPoint(?al2, ?sp2) ∧ alo:hasEndPoint(?al2, ?ep2) ∧ alo:index(?sp2, ?vsx2) ∧ alo:index(?ep2, ?vex2) ∧ swrlb:greaterThan(?vsx2, ?vex1) → vlo:isDiscreteOf(?ve1, ?ve2)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasStartPoint(?al1, ?sp1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?sp1, ?vsx1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Interval(?al2) ∧ alo:hasStartPoint(?al2, ?sp2) ∧ alo:hasEndPoint(?al2, ?ep2) ∧ alo:index(?sp2, ?vsx2) ∧ alo:index(?ep2, ?vex2) ∧ swrlb:equal(?vsx2, ?vsx1) ∧ swrlb:equal(?vex2, ?vex1) → vlo:isTopologicallyEqualsTo(?ve1, ?ve2)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Interval(?al2) ∧ alo:hasStartPoint(?al2, ?sp2) ∧ alo:index(?sp2, ?vsx2) ∧ swrlb:equal(?vsx2, ?vex1) → vlo:isExternallyConnectedTo(?ve1, ?ve2) ∧ vlo:isExternallyConnectedTo(?ve2, ?ve1)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasStartPoint(?al1, ?sp1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?sp1, ?vsx1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Interval(?al2) ∧ alo:hasStartPoint(?al2, ?sp2) ∧ alo:hasEndPoint(?al2, ?ep2) ∧ alo:index(?sp2, ?vsx2) ∧ alo:index(?ep2, ?vex2) ∧ swrlb:greaterThan(?vsx2, ?vsx1) ∧ swrlb:greaterThan(?vex1, ?vex2) → vlo:isNonTangentialProperPartOf(?ve2, ?ve1) ∧ vlo:hasForNonTangentialProperPartOf(?ve1, ?ve2)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasStartPoint(?al1, ?sp1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?sp1, ?vsx1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Interval(?al2) ∧ alo:hasStartPoint(?al2, ?sp2) ∧ alo:hasEndPoint(?al2, ?ep2) ∧ alo:index(?sp2, ?vsx2) ∧ alo:index(?ep2, ?vex2) ∧ swrlb:equal(?vsx2, ?vsx1) ∧ swrlb:greaterThan(?vex1, ?vex2) → vlo:isTangentialProperPartOf(?ve2, ?ve1) ∧ vlo:hasForTangentialProperPart(?ve1, ?ve2)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasStartPoint(?al1, ?sp1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?sp1, ?vsx1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Interval(?al2) ∧ alo:hasStartPoint(?al2, ?sp2) ∧ alo:hasEndPoint(?al2, ?ep2) ∧ alo:index(?sp2, ?vsx2) ∧ alo:index(?ep2, ?vex2) ∧ swrlb:greaterThan(?vsx2, ?vsx1) ∧ swrlb:equal(?vex1, ?vex2) → vlo:isTangentialProperPartOf(?ve2, ?ve1) ∧ vlo:hasForTangentialProperPart(?ve1, ?ve2)

	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasStartPoint(?al1, ?sp1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?sp1, ?vsx1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Interval(?al2) ∧ alo:hasStartPoint(?al2, ?sp2) ∧ alo:hasEndPoint(?al2, ?ep2) ∧ alo:index(?sp2, ?vsx2) ∧ alo:index(?ep2, ?vex2) ∧ swrlb:greaterThan(?vex1, ?vsx2) ∧ swrlb:greaterThan(?vex2, ?vex1) → vlo:partialOverlaps(?ve1, ?ve2)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasStartPoint(?al1, ?sp1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?sp1, ?vsx1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Point(?al2) ∧ alo:hasPoint(?al1, ?al2) ∧ alo:index(?al2, ?vx2) ∧ swrlb:greaterThan(?vx2, ?vsx1) ∧ swrlb:greaterThan(?vex1, ?vx2) → vlo:isNonTangentialProperPartOf(?ve2, ?ve1) ∧ vlo:hasForNonTangentialProperPartOf(?ve1, ?ve2)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasStartPoint(?al1, ?sp1) ∧ alo:index(?sp1, ?vsx1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Point(?al2) ∧ alo:hasPoint(?al1, ?al2) ∧ alo:index(?al2, ?vx2) ∧ swrlb:equal(?vsx1, ?vx2) → vlo:isTangentialProperPartOf(?ve2, ?ve1) ∧ vlo:hasForTangentialProperPart(?ve1, ?ve2)
	mapo:mappingVisualToAnalogic(?ve1, ?al1) ∧ alo:Interval(?al1) ∧ alo:hasEndPoint(?al1, ?ep1) ∧ alo:index(?ep1, ?vex1) ∧ mapo:mappingVisualToAnalogic(?ve2, ?al2) ∧ alo:Point(?al2) ∧ alo:hasPoint(?al1, ?al2) ∧ alo:index(?al2, ?vx2) ∧ swrlb:equal(?vex1, ?vx2) → vlo:isTangentialProperPartOf(?ve2, ?ve1) ∧ vlo:hasForTangentialProperPart(?ve1, ?ve2)