Universidade Federal de Uberlândia

Faculdade de Ciência da Computação

Análise de Nomes da Química Orgânica à luz da Teoria do Léxico Gerativo

-

Da Análise Sintático-Semântica à Geração das Estruturas Químicas através dos Combinadores de *Parser*

Autor: Márcio de Souza Dias

Orientadora: Prof[a]. Dr[a]. Rita Maria da Silva Julia

Uberlândia, MG

Agosto/2006

Universidade Federal de Uberlândia

Faculdade de Ciência da Computação

Análise de Nomes da Química Orgânica à luz da Teoria do Léxico Gerativo

-

Da Análise Sintático-Semântica à Geração das Estruturas Químicas através dos Combinadores de *Parser*

Dissertação de Mestrado apresentada à Faculdade de Ciência da Computação como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação. Área de concentração: **Inteligência Artificial**.

Autor: Márcio de Souza Dias

Orientadora: Prof[a]. Dr[a]. Rita Maria da Silva Julia

Uberlândia, MG
Agosto/2006

Dissertação de Mestrado sob o título *"Análise de Nomes da Química Orgânica à luz da Teoria do Léxico Gerativo - Da Análise Sintático-Semântica à Geração das Estruturas Químicas através dos Combinadores de Parser"*, defendida por Márcio de Souza Dias e aprovada em 18 de Agosto de 2006, em Uberlândia, Estado de Minas Gerais, pela banca examinadora constituída pelos doutores:

Profª. Drª. Rita Maria da Silva Júlia
Orientadora

Profª. Drª. Rosa Maria Viccari
Universidade Federal do Rio Grande do Sul

Prof. Dr. Antônio Eduardo Costa Pereira
Universidade Federal de Uberlândia

Dedico esta dissertação primeiramente a Deus,
que me deu força e perseverança para a conclusão do mestrado,
a minha família que sempre me apoiou e se orgulha de mim,
a minha namorada, Marise, em especial, que esteve ao meu lado a todo o momento, me ajudando em tudo, afeto, carinho e amor,
ao meu grande amigo Sérgio, que foi companheiro de verdade,
me dando até hospedagem de graça na sua casa,
enfim, a todos que torceram por mim.

Agradecimentos

Dedico meus sinceros agradecimentos para:

– a minha orientadora professora doutora Rita Maria da Silva Júlia, pela orientação e incentivo;

– ao professor doutor Antônio Eduardo Costa Pereira, pelos seus conhecimentos que a mim foi passado;

– ao professor mestre Alexssandro Santos Soares, por me apoia desde a graduação;

– a todos os colegas do Mestrado em Ciência da Computação da FACOM/UFU.

Resumo

O presente trabalho propõe um sistema automático de análise de nomes de compostos da Química Orgânica visando a geração do desenho de suas estruturas químicas. Para tanto, o sistema recebe um nome de um composto orgânico, analisa-o sintática e semanticamente e, caso ele represente um composto quimicamente correto, gera uma saída visual para a estrutura química que lhe corresponda. Um avanço que o sistema apresenta com relação a outros que se propõem a efetuar tarefa semelhante é o fato de ele conseguir analisar tanto nomes de compostos que se enquadram nos padrões das nomenclaturas oficiais vigentes, quanto aqueles que, apesar de não se enquadrarem nos mesmos, representam compostos orgânicos verdadeiros (quando ocorrer tal situação, o sistema terá resolvido um problema de ambigüidade de nomenclatura). As análises sintática e semântica são guiadas pelos tipos dos componentes dos nomes químicos, fato que motivou a implementação do sistema nos moldes do formalismo da Teoria do Léxico Gerativo (TLG). Além disso, as análises guiadas pelo tipo motivaram a escolha dos combinadores de *Parser* e da Linguagem de Programação Funcional Clean como utilitários eficazes e adequados na execução das análises lingüísticas. O sistema implementado representa uma ferramenta muito útil como instrutor automático de Química Orgânica.

Palavras Chaves: Teoria do Léxico Gerativo, Química Orgânica e Ambigüidade Lexical.

Abstract

This work proposes an automatic system of analysis for Organic Chemistry compound names aiming to generate pictures of chemical structures. In order to accomplish this, the system receives an organic compound name, analyzes it syntactically and semantically and, if it represents a correct chemical compound, generates a visual output for the corresponding structure. An advance that the system shows in relation to other systems which deal with the same problem consists on being able to analyse both compound names that satisfy the current official nomenclature constraints and those that, despite of do not respect them, represent correct organic compounds (in this case, the system will have solved a nomenclature ambiguity problem). The semantic and syntactic analysis are guided by the name component types of the chemical compounds, which motivated an implementation that fits into the Generative Lexicon Theory (GLT) formalism. Furthermore, the analysis guided by types justified the decision of implementing the system by using the parser combinators and the Clean Funcional Language as very adequate and efficient tools. The implemented system represents a significative utilitarian as an automatic Organic Chemistry instructor.

Keywords: Generative Lexicon Theory, Organic Chemistry and Lexical Ambiguity.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

A compreensão da estrutura química é a chave para muitos desafios na química moderna, ou seja, compreendê-la significa interpretar melhor os componentes naturais e sintéticos que têm impacto em aspectos fundamentais do mundo atual, tais como pesquisa em materiais, indústria farmacêutica, indústria alimentícia etc.

A organização *PubMed*[1] (Livraria de Medicina) tem colecionado cerca de 13 milhões de resumos e publicações nas áreas de bioquímica e de medicina, a que são acrescidos de1.500 à 3.000 entradas por dia. Aproximadamente 42% das informações biológicas ainda são codificadas em arquivos de jornais e apenas 17% já são estruturadas em Banco de Dados [Anstein e Kremer 2005]. Para reverter esses dados a favor da pesquisa bioquímica automatizada, os campos da Lingüística Computacional e da Tecnologia da Informação Aplicada vêm investindo pesados esforços nas pesquisas em Processamento de Linguagem Natural (PLN) ligadas ao contexto da Bioquímica, inclusive em sua parte relacionada à Química Orgânica. O desenvolvimento de ferramentas adaptativas para extração semi-automática de conhecimento, isto é, métodos como a mineração de texto ou a sumarização automática, são essenciais para acessar e explorar uma grande quantidade de dados textuais.

Regras Gramaticais e lexicais para a decomposição de termos bioquímicos podem ser implementadas (uma aproximação semelhante ao tratamento que se dá a linguagem natural) em sistemas de nomenclatura de compostos orgânicos. Os nomes científicos dos compostos orgânicos seguem regras sintáticas, semânticas e pragmáticas que os tornam passíveis de serem analisados automaticamente por agentes similares aos que tratam a questão da análise na Linguagem Natural. Problemas nesta última, tal como o da ambigüidade lexical, também ocorre na nomenclatura dos compostos orgânicos. Logo, as técnicas da Lingüística Computacional são completamente apropriadas para tratar o problema da análise automática da nomenclatura de compostos químicos. Como os nomes dos compostos orgânicos normalmente são os objetos de interesse em textos bioquímicos, sua identi-

[1] www.pubmed.gov

ficação, classificação e compreensão é de grande importância. [Anstein e Kremer 2005].

Para se padronizar a nomenclatura dos nomes dos compostos químicos, criou-se uma nomenclatura associada aos mesmos denominada *International Union of Pure and Applied Chemistry* - União Internacional de Química Pura e Aplicada (IUPAC) - que uniformizou toda a nomenclatura dos compostos químicos, tornando-a oficial para todos que trabalham e que queiram trabalhar com a química moderna [Usberco e Salvador 2001]. Contudo, há maneiras alternativas de representar um composto químico que, apesar de não obedecerem à nomenclatura oficial, apontam, corretamente, para o composto a que se quer referir. Neste contexto, o objetivo deste trabalho é implementar uma ferramenta automática capaz, não somente de analisar sintática e semanticamente nomes de compostos de Química Orgânica que sigam as padronizações oficiais, gerando o desenho da estrutura química correspondente a eles (caso, obviamente, eles representem compostos corretos), como também de efetivar o mesmo procedimento nos casos em que os nomes analisados, apesar de não seguirem os padrões IUPAC, correspondem a compostos corretos. Neste último caso, o sistema terá resolvido um problema de ambigüidade lingüística, fato que o distingue de outros sistemas existentes [Brecher 1999], [Cambridgesoft 2005], [Gerstenberger 2001] e [Anstein e Kremer 2005].

Para cumprir os objetivos que foi designado ao sistema, o mesmo utiliza de ferramentas e teorias como a Teoria do Léxico Gerativo (TLG) [Pustejovsky 1995], os combinadores de *Parsers* [Koopman et al. 1997], o pacote em Latex denominado Xymtec [Fujita 1993] e a linguagem de programação funcional Clean [Koopman et al. 2002].

O sistema será denominado Resolvedor de Ambigüidades Lingüisticas na Química Orgânica (RALQ).

A TLG é um formalismo lingüístico para estruturação de informações semânticas referentes aos itens lexicais. Para tanto, ela utiliza uma estrutura léxica formada por estruturas de argumentos, eventos, qualia e de herança. Através da combinação de tipos ela tenta buscar o melhor complemento que levará à semântica de um item léxico.

Sendo assim, técnica análoga a da TLG é usada no presente trabalho com a finalidade de auxiliar na desambiguação de nomes de compostos da química orgânica através da combinação de tipos compatíveis. Isso é feito nas etapas das Análise Sintática e Semântica.

A análise sintática usa os combinadores de parser para detectar os tipos (itens léxicos) relevantes, a saber: prefixos com suas respectivas posições na cadeia principal, cadeia principal, sufixo e locais das eventuais insaturações. A análise Semântica visa checar se os

itens léxicos levantados pela Análise Sintática são passíveis de se combinarem de modo a satisfazer as restrições semânticas do composto da cadeia principal (ou seja, do composto formado pela cadeia principal seguida do sufixo).

As possíveis ambigüidades existentes nos nomes da química orgânica, provocadas, por exemplo, pela posição inadequada de um prefixo ou de uma ligação na cadeia principal do composto, ocasionam uma incompatibilidade de tipos que deve ser detectada pelo sistema.

A contribuição deste trabalho é a implementação de um instrutor automático que auxiliará no ensino das nomenclaturas dos compostos orgânicos em instituições de ensino, em laboratórios químicos etc. O protótipo implementado também executa a desambiguação de nomes dos compostos orgânicos, ou seja, nomes que não são reconhecidos pelo nomenclatura oficial são trabalhados pelo protótipo no intuito de descobrir a quais estruturas químicas correspondem.

Para elaboração deste trabalho será realizado, em um primeiro momento, um levantamento de fontes bibliográficas relacionadas a ambigüidade léxica e de trabalhos computacionais voltados ao domínio da química. Posteriormente, será feito um estudo de técnicas e ferramentas que serão utilizados na implementação do sistema RALQ a ser desenvolvido.

A organização dessa dissertação é descrita da seguinte forma: o capítulo 1 será composto pela introdução do trabalho. No capítulo 2 será feito o levantamento dos trabalhos relacionados a desambiguação lingüística e trabalhos voltados ao tratamento computacional da química orgânica. O capítulo 3 trará a fundamentação teórica da presente pesquisa. No capítulo 4 virá toda a descrição do procedimento de construção do sistema RALQ e de como será utilizado o formalismo da TLG junto ao domínio da química orgânica, bem como o processo de desambiguação dos nomes dos compostos orgânicos. E, por fim, o capítulo 5 apresentará as conclusões e as propostas de trabalhos futuros.

2 Estado da Arte

Conforme apresentado anteriormente, este trabalho focaliza a questão da ambigüidade lexical no problema da análise de nomes da Química Orgânica.

Devido a isso, o objetivo desta seção é apresentar um cenário abrangente de trabalhos recentes que atacam o problema da ambigüidade lexical nos mais diversos contextos. Finalizando a seção, serão citados alguns trabalhos que também se concentram em problemas ligüísticos da Química Orgânica.

2.1 Desambiguação Lexical

A Desambiguação Léxica de Sentido (DLS) mostra-se, a partir de 1960, como uma subárea de pesquisa de Processamento de Linguagem Natural, pois a ambigüidade lexical tornou-se um problema em evidência. Desde então, trabalhos vêm sendo propostos para resolver tal problema [Specia e Nunes 2004].

Esse capítulo tratará das produções científicas mais relevantes relacionadas à DLS, levando-se em conta a sua importância histórica ou inovadora, as diferentes características com relação as outras abordagens existentes e sua possível influência nas novas propostas de modelos de desambiguação lexical de sentido.

Estudos relacionados a DLS focalizaram mais os sistemas monolingües[1], principalmente a língua inglesa. Entretanto, os principais conceitos e procedimentos de desambiguação lexical de sentido aplicados nesses tipos de sistemas também podem ser aplicados em sistemas multilingües[2], uma vez que estes últimos podem adequar-se aos aspectos primordiais dos sistemas monolingües [Specia e Nunes 2004].

Os trabalhos de DLS serão apresentados aqui segundo uma classificação baseada nos

[1]São os sistemas que tratam de um único idioma específico, por exemplo, sistemas de correção ortográfica para o Português

[2]Sistemas que trabalham com mais de um idioma, por exemplo, um sistema de Tradução Automática (TA).

métodos que utilizam.

2.1.1 Método Fundamentado em Conhecimento

Esse método é baseado em conhecimento lingüístico e/ou extralingüístico explicitamente especificado, manualmente ou semi-automaticamente (léxicos computacionais, dicionários eletrônicos e thesauri[3])[Specia e Nunes 2004].

2.1.1.1 Conhecimento codificado manualmente

Os primeiros trabalhos de desambiguação lexical de sentido, os quais eram fundamentos em conhecimento, usavam redes semânticas e *frames*[4] como técnicas simbólicas de representação e manipulação do conhecimento da Inteligência Artificial.

Em 1961, no trabalho de Masterman (voltado para a tradução automática) [Masterman 1961], a rede semântica foi utilizada como formalismo de representação do conhecimento expresso pelas sentenças em inglês. No caso, o objetivo é buscar retirar possíveis termos ambíguos das sentenças através da formação de uma rede, cujo nós seriam uma estrutura de "tipos de conceitos primitivos", ou seja, cada tipo seria um conjunto de conceitos relacionados entre si (oriundos de um dicionário), utilizando um mecanismo de herança entre os conceitos mais genéricos para os mais específicos dentro desses tipos. Com isso, a desambiguação de sentido é feita de forma implícita através da seleção do conceito que melhor representa o relacionamento do termo (ambíguo) da sentença com o tipo primitivo da rede semântica. Esse tipo de tratamento trabalha primeiramente com a questão da desambiguação da própria língua fonte, para depois poder traduzi-la para uma língua alvo sem ambigüidades.

O trabalho monolíngüe (língua inglesa) de Hayes, em 1976 [Hayes 1976], usa *frames* e uma rede semântica para uma representação mista do conhecimento. O métodos enfatiza a busca por associações de sentidos entre as palavras junto aos nós da rede que representam os significados dos substantivos e os verbos representados por*frames*. Essa representação é considerada uma rede semântica onde cada *frame* é uma parte da rede, facilitando, assim, a busca por associações semânticas entre as palavras. Para nós e *frames* predecessores

[3]Instrumento que reúne termos escolhidos a partir de uma estrutura conceitual previamente estabelecida, destinados à indexação e à recuperação de documentos e informações num determinado campo do saber. Não é simplesmente um dicionário, mas um instrumento que garante aos documentaristas e pesquisadores o processamento e a busca dessas informações[Pereira 2004].

[4]Coleções de atributos e valores que ajudam da representação da informação semântica [Specia e Nunes 2002].

ou sucessores, outras associações são identificadas, de acordo com as relações hierárquicas padrão de hiponímia[5] e meronímia[6] das redes semânticas, levando então a uma cadeia de conexões na base de conhecimento. Possuía também, as estruturas de caso e restrições de seleção semântica.

Assim sendo, a desambiguação dos substantivos é obtida pelas estruturas de casos, associações e restrições de seleção semântica, e é realizada a partir da estrutura sintática produzida para a sentença de entrada. Enfim, o processo de desambiguação de Hayes leva em consideração todos os substantivos ambíguos, com todas as possibilidades analisadas em paralelo, ou seja, utiliza a rede semântica (formada por substantivos e verbos detectados no módulo de análise sintática) e os métodos restritivos de seleção para achar o melhor sentido de um substantivo ambíguo através do mecanismo de associações entre substantivos e verbos da sentença de entrada representados por nós e pelos *frames* respectivamente na rede semântica. Para tanto, varre-se toda essa rede (nós e os frames) na busca do melhor sentido do substantivo ambíguo associado ao verbo da sentença de entrada, até que o sistema tenda a uma solução, respeitando os métodos restritivos.

É importante salientar que nos sistemas baseados em restrições de seleção a desambiguação lexical de sentido só acontece após a análise sintática e, em alguns casos, somente depois que o contexto (assunto) da sentença é conhecido.

De acordo com Hayes, as associações semânticas são importantes principalmente na manipulação de casos de homonímia[7], devido a sua grande distinção entre os seus sentidos, mas não são muito indicadas para o tratamento de polissemia[8]. Isso possivelmente pode explicar um dos motivos pelos quais Hayes não trabalhou na desambiguação da classe gramatical dos verbos, pois esses tendem a ser polissêmicos e os substantivos a serem homônimos [Hirst 1987].

Small em 1980, que também trabalha somente com o idioma inglês, fundamenta-se na teoria de que "o conhecimento humano sobre a língua é primariamente organizado na forma de conhecimento sobre as palavras e, não, na forma de regras" [Small 1980]. Com

[5]É uma relação lexical correspondente à inclusão de uma classe em outra. Para identificar essa relação, pode-se fazer uso do seguinte esquema: X é um Y, ou X é um tipo de Y. [Por ex.:, A UFU é uma universidade, ou uma universidade é um tipo de instituição de ensino][Teixeira 2004]

[6]É uma relação semântica a qual atribui-se o nome merônimo à parte constituinte, a substância ou membro de algo, ou seja, X é parte de Y. [Por ex.: A FACOM faz parte da UFU.][Teixeira 2004]

[7]Duas ou mais palavras com significados totalmente distintos, sem traços comuns, são idênticas quanto ao som [homofonia. Ex.: cela (substantivo) e sela (verbo)] e/ou à grafia [homografia. Ex.: gosto (substantivo) e gosto (1.ª pess.sing. pres. ind. - verbo gostar)] [Specia e Nunes 2004].

[8]Uma mesma palavra tem dois ou mais significados diferentes, mas relacionados entre si, sendo que, normalmente, somente um dos significados se ajusta a um determinado contexto. Ex.: Ele ocupa um alto *posto* na empresa. Abasteci meu carro no *posto* da esquina. [Specia e Nunes 2004].

isso, monta uma estrutura lexical, na qual devem estar presentes as informações relativas às funções sintáticas, às informações semânticas e ao domínio do discurso das palavras da oração a ser desambiguada, de modo que o sentido, a estrutura e o significado da oração sejam obtidos por meio do processo de desambiguação.

Small utilizou um mecanismo especialista para cada palavra, o qual possuía informações e procedimentos para distinguir todos os seus sentidos. Esses mecanismos operavam em grupo sobre a sentença de entrada, trabalhando na escolha do melhor sentido e, em paralelo, executando análise sintática e semântica, resultando na representação semântica da sentença. Assim, não há estrutura sintática intermediária. Os vários mecanismos especialistas usados para a análise de uma sentença são disparados e controlados em um ambiente distribuído. Esse procedimento foi empregado no sistema denominado *Word Expert Parser* de Small, sendo *word expert* a denominação dada a um mecanismo especialista considerado repositório de uma grande quantidade de informações que, em outros trabalhos, normalmente, estão distribuídas entre os módulos de análise sintática, léxica, semântica, entre outros.

Small certifica que as informações que controla a desambiguação devem ser armazenadas juntamente com o conhecimento declarativo sobre a palavra. Ele também acrescenta que os trabalhos nos quais há uma distribuição, em módulos, das informações de controle não apresentam ganhos em termos de funcionalidade e são difíceis de compreender ou alterar. Todavia, o autor destaca que os *experts* não são cópias uns dos outros, com pequenas alterações, e é por isso que cada um deles requer um processo elaborado para a sua criação. Por exemplo, a descrição do *expert* para o verbo *throw* possui seis páginas e, segundo Small, poderia ser ainda mais detalhadamente especificada, com um tamanho dez vezes maior. Uma importante deficiência desse trabalho é a falta de mecanismos de generalização em função do princípio de definição individualizada e altamente refinada de cada *expert*, fato que, certamente, limita a abrangência do sistema. Contudo, o trabalho é significativo ao mostrar que regras e conhecimento específicos sobre cada palavra tornam a desambiguação mais precisa.

Sistemas de Tradução Automática, como o EUROTRA [Copeland et al. 1991] e METAL [Gajek 1991], utilizam procedimentos mais simples para tratar a ambigüidade lexical, através da definição de estruturas argumentais (verifica os argumentos dos verbos e substantivos da uma sentença para especificar o domínio para a escolha de um possível sentido) e de restrições (restrições gramaticais) ou preferências de seleção (uso frequente de um determinado sentido), ou seja, através desses procedimentos realizam uma análise mais

simples da frase de entrada, obtendo informações lexicais que serão úteis na escolha da melhor tradução. De certa forma, esse é o caso mais simples de desambiguação.

No sistema EUROTRA[9], em particular, uma estrutura reticulada com 54 hierarquias simples de tipos semânticos (entidade, humano, não-humano, etc.) é utilizada para tratar os casos de desambiguação mais complicado. O sistema trabalha com a noção de distância semântica entre os nós dessa hierarquia. Para a desambiguação de um substantivo que complementa o verbo em uma sentença, por exemplo, a hipótese é de que quanto menor a distância entre o nó que representa o sentido de um substantivo e os nós que representam as restrições impostas na estrutura argumental do verbo em questão, mais provável será esse o sentido do substantivo. Segundo Pedersen [Pedersen 1997], essas restrições são simples e limitadas, resolvendo alguns casos mais simples de ambigüidade.

Um sistema de tradução automática, do coreano para o inglês foi apresentado em 1994 por um grupo de pesquisadores liderados por Egedi [Egedi et al. 1994], implementado de acordo com o formalismo de representação *Synchronous Tree Adjoining Grammar*[10] (STAG) [Shieber e Schabes 1990]. O seu módulo de desambiguação, que baseia-se na unificação de restrições de seleção semântica, trata da ambigüidade de alguns verbos. Essas restrições são definidas na estrutura argumental dos verbos através dos traços semânticos dos substantivos que lhe servem como argumentos, tal como a classificação semântica desses substantivos (homem, objeto, etc). O método trata manualmente as regras de transferência que compreendem as restrições de seleção e traços semânticos. É no processo de transferência lexical, com base nas possíveis traduções oriundas de um dicionário bilíngüe e nas restrições de seleção e de traços semânticos da língua-alvo, que ocorre a desambiguação.

Em resumo, o sistema de Egedi e sua equipe visa traduzir sentenças do coreano para o inglês, com um enfoque maior nos substantivos e verbos. Utilizando um dicionário bilíngüe, restrições de seleção semântica e traços semânticos, o sistema primeiramente tenta eliminar a ambiguidade dos substantivos presentes na sentença (pois são os substantivos que darão as informações necessárias para a seleção da melhor tradução para os verbos da sentença) através da busca de uma tradução que se encaixe melhor nas restrições de seleção semântica do substantivo da língua fonte. Feita a desambiguação do substantivo, agora é só utilizar o seu significado para delimitar o traço semântico e as restrições semânticas para poder desambiguar o verbo da sentença, comparando as

[9]Projeto que teve início em 1982, que objetiva a tradução automática de todas as línguas da Comunidade Européia [ILTEC]

[10]Gramática Adjacente de Árvore Síncrona

características semânticas e sintáticas oriundas desses procedimentos na língua fonte com as características advindas das possíveis traduções feitas pelo dicionário.

Um problema com esse método aparece sempre que os argumentos do verbo também forem ambíguos, já que a desambiguação de um verbo depende da tradução correta dos seus argumentos, e estes precisam ser primeiramente traduzidos para a língua-alvo para que possam ser identificados os traços semânticos, podendo prejudicar a desambiguação dos verbos.

Pedersen em 1997 descreve um trabalho baseado em teorias da semântica lexical para a desambiguação de um subconjunto de verbos de movimento polissêmicos para ser utilizado num possível sistema de tradução automática do dinamarquês para o inglês [Pedersen 1997]. Levando em conta somente o fenômeno da polissemia sistemática do subconjunto de verbos, a autora tem como objetivo (em seu trabalho) identificar padrões para o tratamento de polissemia sistemática, ou seja, que possam ser aplicados a diversos verbos com significado relacionado, dentre os verbos que expressam movimento. Formalizar e implementar esses padrões na forma de regras lexicais que possam ser usadas para a desambiguação lexical na tradução automática. Para isso, inicialmente, foi realizada uma análise das ocorrências de 100 verbos que expressam movimento em diferentes corpus do idioma dinamarquês para verificar propriedades estatísticas (freqüência, co-ocorrências, etc.) e, outras características do uso desses verbos, bem como os tipos de conhecimento que são necessários para diferenciar os seus sentidos. Para essa análise, foram selecionados de 100 a 300 exemplos de ocorrência de cada um dos verbos. De acordo com a análise de suas propriedades sintáticas e semânticas os exemplos puderam ser categorizados manualmente, impondo também algumas restrições, por exemplo, foram descartados exemplos do uso do verbo em expressões idiomáticas e metafóricas.

Como resultado dessa etapa, foram formados grupos de exemplos com propriedades similares, por exemplo, exemplos de verbos que expressam movimento que têm uma direção específica, cujo agente é animado e que implica o movimento de partes do corpo ou de uma máquina. Nesses verbos, segundo a autora, a polissemia deve se manifestar de maneira sistemática, de modo que todos os verbos do grupo podem receber o mesmo tratamento na desambiguação lexica de sentido. Para representar os verbos dos grupos, a autora definiu um modelo lexical, baseado na teoria Frame Semantics [Atkins e Fillmore 1994] e na Estrutura de Eventos do léxico gerativo de Pustejovsky [Pustejovsky 1995]. Também foram definidas uma hierarquia conceitual parcial e restrições de seleção para substantivos distribuídos nessa hierarquia. Os verbos foram então

especificados de acordo com o modelo definido, utilizando uma grande quantidade de informações lingüísticas na língua-fonte, em diversos níveis, que indicam os desvios de significado e, portanto, podem auxiliar na desambiguação. Os esquemas especificados foram implementados na forma de regras lexicais e incorporados a um sistema de interpretação do dinamarquês. A autora realizou um teste com 42 sentenças com os verbos ambíguos. Desses verbos, 39 foram corretamente desambiguados. Uma obervação importante a ser destacada nesse trabalho é que apesar da aplicação ser voltada para a tradução automática, esse trabalho de desambiguação lexical de sentido visa a especificação das regras lexicais com informações suficientes para permitir capturar os padrões sistemáticos entre os diferentes sentidos de um verbo, de modo a evitar descrições ambíguas. Assim, a desambiguação ocorre, basicamente, na língua-fonte.

Em 1998 um mecanismo de seleção lexical para verbos e substantivo na tradução automática do inglês para o espanhol é definido por Dorr e Katsova [Dorr e Katsova 1998]. Esse mecanismo é baseado na estrutura argumental dos substantivos e verbos, representada por meio de estruturas conceituais lexicais[11] (LCSs), e nos sentidos da WordNet que é um sistema de referência léxica online cujo projeto foi inspirado em teorias psicolinguísticas atuais sobre a memória léxica humana. Substantivos, verbos, adjetivos e advérbios da língua inglesa são organizados em conjuntos de sinônimos (*synset*, cada conjunto representando um conceito ou categoria léxica diferente). Os conjuntos são então relacionados por relações estruturais (sinônimos, hiponímia, hipernímia, holonímia e meronímia) [Laboratory 1998]. As autoras supunham que a tradução de um elemento da língua-fonte pode ser desambiguada se forem escolhidos, na língua-alvo, elementos que tenham a mesma LCS e que pertençam ao mesmo synset da WordNet, ou seja, que sejam sinônimos do elemento na língua-fonte. Com o intuito de avaliar essa suposição Dorr e Katsova implementaram um algoritmo de seleção lexical que utiliza um sistema de representação de conhecimento baseado em *frames* denominado *Parka* para codificar sentenças em suas representações LCSs. Esse sistema (*Parka*) também possui um léxico do inglês e outro do espanhol, cujas entradas estão codificadas como LCSs e, um código, anotado manualmente, que corresponde ao *synset* da WordNet ao qual pertencem. Com base na estrutura gerada pelo *Parka* para uma sentença, o algoritmo extrai a estrutura LCS do verbo a ser desambiguado e recupera do léxico do espanhol todas as entradas correspondentes a verbos que têm a LCS com as mesmas propriedades estruturais. Por exemplo, para o verbo *sap*, são recuperados 358 verbos do espanhol (por exemplo, *agotar, desaguar, escurrir, evacuar, reducir, vaciar, zapar* etc) com a mesma estrutura de LCS.

[11] Arranjos hierárquicos de funções e argumentos [Lieber 2004]

Desse conjunto de verbos, o algoritmo seleciona apenas aqueles que apresentam o mesmo código do *synset* da WordNet que o verbo que está sendo desambiguado. Se o verbo puder pertencer a vários *synsets*, são selecionados todos os verbos em todos os seus synsets. Para o verbo *sap*, apenas um verbo (*escurir*) pertence ao mesmo *synset*. Esse verbo é então escolhido como a tradução mais adequada para o verbo do inglês.

Caso haja mais de um verbo com a mesma estrutura e o mesmo código de *synset*, o algoritmo retorna todos eles. Por outro lado, caso não seja encontrado nenhum verbo com a LCS equivalente no mesmo *synset*, o algoritmo estende a busca aos *synset* hiperônimos em um nível (mais genéricos) de todos os *synset* aos quais o verbo pertence. O algoritmo é bastante flexível, pois pode operar tanto na desambiguação lexical de sentido multilíngüe quanto na monolingüe. Nesse caso, as buscas por LCSs equivalentes são feitas no léxico da própria língua.

De acordo com as autoras, o método desenvolvido é mais útil para a desambiguação lexical de sentido monolingüe. Assim sendo, essa desambiguação deve ser realizada como um pré-processamento para a multilingüe, podendo reduzir as ambigüidades, melhorando a precisão na tradução.

Como são recuperadas todas as LCSs estruturalmente equivalentes, podem ser recuperados verbos que, apesar de estarem no mesmo *synset*, não são válidos como tradução do verbo na língua-fonte, ocasionando um problema. Uma possível solução para o problema seria buscar apenas as estruturas dos verbos que podem ser traduções do verbo na língua-fonte, a partir da consulta a um dicionário bilíngüe. O sistema também é limitado pelo fato de somente conseguir lidar como as LCSs, que já estão codificadas no léxico às quais já tenha sido atribuído um código de *synset*. Ademais, o fato de serem retornadas todas as traduções possíveis indica que o sistema não elimina todas as ambigüidades. O trabalho não é considerado fundamentado em conhecimento pré-codificado, mesmo usando informações da WordNet, pois a maior parte do conhecimento é especificada manualmente, incluindo os *synsets* correspondentes às entradas lexicais.

Os trabalhos mostrados acima foram os que expressaram melhor a sua própria funcionalidade e todos os conceitos envolvidos no método fundamentado em conhecimento codificado manualmente. A seguir, serão apresentados trabalhos com um ponto de vista um pouco diferente em relação à fonte de conhecimento.

2.1.1.2 Conhecimento Pré-Codificado

Os trabalhos a serem apresentados nesta subseção são baseados em conhecimento lingüístico semi-automáticos, por meio de recursos como léxicos computacionais, dicionários eletrônicos e thesauri.

Os trabalhos baseados em léxicos computacionais utiliza a WordNet ou versões multilingües desse recurso, por exemplo, a EuroWordNet, como depósito de sentidos e/ou fonte de informações variadas. É importante salientar que os trabalhos que usam os léxicos computacionais, não são voltados para a aplicação da tradução automática [Specia e Nunes 2004].

Um trabalho levado para a aplicação na Recuperação de Informações foi feito por M. Sussna em 1993 [Sussna 1993], fundamentado nas distâncias entre os sentidos na hierarquia conceitual da WordNet. Segundo o autor, para um conjunto de termos ocorrendo próximos uns dos outros em um texto, cada um deles podendo ter vários sentidos, o sentido que minimiza a distância[12] entre eles na hierarquia representa o sentido mais adequado.

Sussna considera, em seu trabalho, a desambiguação de todos os substantivos de um texto, previamente identificados por um etiquetador morfossintático. O cálculo da distância semântica para cada substantivo do texto a ser desambiguado é feito através da análise da distribuição na hierarquia conceitual da WordNet de todos os possíveis sentidos desse substantivo, bem com de todos os possíveis sentidos das palavras vizinhas na sentença em uma janela (estrutura) de contexto pré-definida, visando identificar o nível de relação entre os pares de sentidos das palavras.

Levando em consideração cinco documentos jornalísticos e variações no tamanho e no tipo da janela de contexto, Sussna realizou diversos experimentos de avaliação, bem como diferentes esquemas de pesos das relações hierárquicas para o cálculo da distância semântica. Segundo o autor, os resultados desses experimentos foram considerados relevantes, levando-se em conta que nenhum conhecimento lingüístico adicional ao disponível na WordNet é necessário.

O trabalho de Sussna tem dois problemas primordiais: o primeiro é o custo computacional, pois são testadas todas as combinações possíveis entre todos os sentidos do substantivo ambíguo e de todos os demais substantivos no seu contexto, tornando-o inviável se for considerar contextos maiores; o segundo, é a aplicação do filtro para os testes, onde o autor considerou apenas casos de ambigüidade de resolução mais simples [Resnik 1995].

[12]Distância Semântica

As classes semânticas e a hierarquia de classes da WordNet são usadas no trabalho de Agirre e Rigau em 1996 [Agirre e Rigau 1996] com o intuito de realizar a desambiguação lexical de sentido de substantivos. Para tanto, é utilizada uma medida já definida e denominada por eles de "densidade conceitual", a qual calcula a distância entre conceitos, ou seja, a proximidade do significado entre eles. Para uma hierarquia semântica, a distância conceitual pode ser definida como o comprimento do menor caminho que conecta dois conceitos nessa hierarquia. São analisados todos os substantivos em contextos parametrizáveis de ocorrência do substantivo a desambiguar (substantivos vizinhos), os possíveis sentidos desse substantivo na WordNet e a sua distribuição, juntamente com a distribuição dos sentidos dos substantivos no contexto, na hierarquia da WordNet. Mais especificamente, verifica-se, na sub-hierarquia da WordNet, para cada um dos possíveis sentidos do substantivo a ser desambiguado, a quantidade de sentidos dos demais substantivos do contexto. A sub-hierarquia que contiver mais sentidos, relativo ao total de sentidos da hierarquia, leva a uma densidade mais alta na medida de densidade conceitual. O sentido do substantivo ambíguo nessa hierarquia é, então, escolhido para desambiguar tal substantivo. Assim, o sentido mais adequado para o substantivo ambíguo é o que maximiza a densidade conceitual entre esse sentido e os dos substantivos vizinhos. Dependendo do caso, o procedimento pode falhar e retornar vários ou nenhum sentido.

O procedimento foi testado pelos seus autores e, segundo os próprios, foram obtidos resultados menos satisfatórios do que os de outros trabalhos porque o procedimento deles considera a desambiguação de todos os substantivos, enquanto que outros trabalhos visam à desambiguação de apenas um subconjunto de palavras.

A utilização de um filtro com o intuito de selecionar somente os sentidos com probabilidade de maior co-ocorrência, ou seja, mais freqüentes, foi utilizado por Mihalcea e Moldovan em 1999 [R.Mihalcea e Moldovan 1999], antes de aplicar a medida da densidade conceitual. As informações de freqüência de co-ocorrências são obtidas a partir de buscas em textos da web e a densidade conceitual é medida com base na hierarquia da WordNet.

O trabalho visa a desambiguação de substantivos, verbos, advérbios e adjetivos em textos independente do tipo de assunto que é tratado por esses textos, usando os sentidos da WordNet. Para isso, as palavras são emparelhadas em pares numa tentativa de se realizar a disambiguação de uma das palavras do par, ou seja, a palavra ambígua. Para tanto, são realizadas buscas na web, utilizando as palavras que definiram o contexto da palavra ambígua, (usando o buscador AltaVista®), nas quais cada sentença de busca consiste de um dos possíveis sentidos da palavra ambígua, enquanto a outra palavra do par

é mantida fixa. Os sentidos são então ordenados de acordo com o número de documentos resultantes nas respectivas buscas. Os sentidos com mais documentos são considerados mais freqüentes. Isso é feito com todas as palavras da sentença a desambiguar. Considerando apenas os sentidos mais freqüentes para cada palavra, de acordo com um limite inferior pré-definido, há um refinamento na ordenação dos sentidos utilizando a medida de densidade conceitual, considerando o número de palavras comuns que estão a uma distância semântica, dada pela hierarquia da WordNet, de duas ou mais palavras. É importante dizer que esse passo de refinamento só ocorre para os verbos e substantivos, pois adjetivos e advérbios não estão incluídos na hierarquia da WordNet. Para esses últimos, somente as informações de freqüência de co-ocorrência são utilizadas.

Esse trabalho se diferencia da grande maioria dos trabalhos de desambiguação lexical de sentido por retornar os vários possíveis sentidos (classificados de acordo com sua densidade conceitual e freqüencia), em vez de um único sentido. Mihalcea e Moldovan comentam que essa característica permite a escolha por outras opções de sentido, quando a primeira não for aplicável.

Em 2002, um grupo de pesquisadores apresentava uma interface para a desambiguação de substantivos e verbos desenvolvida para ser ligada a sistemas multilingües de Recuperação de Informações [Montoyo et al. 2002]. Eles utilizaram o idioma espanhol e o inglês como línguas-fonte para a definição das sentenças de busca e a geração de sentenças equivalentes em catalão e basco; usaram também uma classificação de uma versão da WordNet, a chamada EuroWordNet[13] [Vossen 1998], e definições das palavras, dos exemplos e das relações entre as palavras já existentes na WordNet. A EuroWordNet possui também ligações entre as palavras das diversas línguas por meio de um índice interlingual, o qual não inclui a língua portuguesa. Com isso, a identificação do sentido da palavra em uma língua produz de forma automática, um conjunto de sentidos equivalentes em outras línguas.

A realização da desambiguação consiste em identificar qual é o sentido correspondente à palavra a ser desambiguada, ainda na língua-fonte e, em seguida, encontrar a(s) palavra(s) na língua-alvo com o mesmo código da EuroWordNet. Assim, embora seja voltada para aplicações multilingües, a desambiguação é feita de maneira monolingüe. Além disso, para aplicações multilingües, esse trabalho só é possível para línguas previstas no projeto EuroWordNet, para as quais já existem códigos correspondentes aos itens lexicais.

[13]Inclui outras línguas, além do inglês, para realizar o mapeamento entre as palavras dessas duas línguas para o catalão e o basco

Dicionários eletrônicos também são utilizados como fontes de informações para muitos trabalhos de desambiguação lexical de sentido monolíngüe (essas informações variam de acordo com o dicionário empregado).

O primeiro trabalho a usar o recurso do dicionário eletrônico foi de M. Lesk em 1986 [Lesk 1986], o qual tinha interesse na Recuperação de Informações. O autor baseia-se na proposição de que cada sentido de uma palavra ambígua é distinto a partir dos possíveis sentidos em um contexto limitado e que um bom dicionário eletrônico deve permitir identificar esse sentido por meio das palavras disponíveis na definição de cada sentido. Assim, quaisquer palavras ambíguas em textos de diferentes domínios (contextos) poderiam ser desambiguados com a utilização de um dicionário eletrônico.

O autor utiliza uma estrutura chamada "assinatura"[14] associada a cada sentido de uma palavra ambígua e a cada sentido das suas palavras vizinhas em uma sentença. De acordo com a seleção do sentido da palavra ambígua cuja assinatura apresenta o maior número de sobreposições com alguma assinatura das palavras vizinhas na sentença, a escolha do sentido é realizada, ou seja, a desambiguação é executada.

Segundo Lesk, não foi feita uma avaliação sistemática do seu trabalho, mas apenas alguns testes com pequenos exemplos. Mas de acordo com esses testes, o autor considera seus resultados relevantes, levando-se em conta, que são empregadas distinções de sentido relativamente requintados (dadas pelo dicionário), considerando somente as informações disponíveis no dicionário e, não utiliza da informação do contexto que uma determinada palavra ambígua possa estar inserido. Lesk sugere que essa abordagem seja utilizada, por exemplo, nas relações sintáticas entre as palavras, como complemento de algum mecanismo que utilize outras informações.

A dependência das definições de um dicionário específico, ou seja, a presença ou ausência de uma palavra na definição de um dicionário, pode mudar os resultados, ocasionando um problema no trabalho de Lesk. Outro problema seria a contagem das palavras nas assinaturas, podendo privilegiar a escolha de sentidos com definições mais longas. Mesmo assim, o trabalho é de grande importância para a desambiguação lexical de sentido, pois serve de base para vários outros trabalhos estatísticos.

O trabalho de Lesk serviu de base para muitos autores, os quais procuraram melhor desempenho com a incorporação, através do dicionário eletrônico, de campos adicionais de informações para os diferentes sentidos, além da própria definição das palavras. A maioria dos trabalhos relacionados à utilização de dicionário emprega o dicionário LDOCE

[14]Uma lista de palavras de conteúdo que aparecem na definição daquele sentido em um dicionário.

(*Longman Dictionary of Contemporary English*), usando suas informações de freqüência dos sentidos, os códigos de área (economia, engenharia etc.), os traços semânticos de substantivos (abstrato, humano etc.) e as restrições de seleção dos argumentos de verbos [Specia e Nunes 2004].

Em 1991, por exemplo, pesquisadores descreveram um trabalho voltado para a Recuperação de Informações que se baseia nos códigos de área do LDOCE [Guthrie et al. 1991]. Apesar de similar ao trabalho de Lesk, diferenciava-se do mesmo por efetuar sobreposição das definições em que o código de área também coincide com o código das palavras vizinhas, em vez de considerar a sobreposição entre as definições de todos os possíveis sentidos da palavra ambígua e das suas palavras vizinhas. Com isso, procuram estabelecer relações de co-ocorrência dependentes de área entre as palavras.

As relações citadas acima (a partir de uma palavra ambígua) são identificadas através da aquisição de algumas palavras que aparecem nas definições dos sentidos de cada uma das suas possíveis áreas. Nem todos os sentidos possuem códigos de área, uma vez que alguns são considerados genéricos, independentes de área. As palavras que aparecem nas definições desses sentidos (não possuem códigos de área) são então agrupadas em uma área denominada "geral". Apesar da hierarquia provida pelo LDOCE disponibilizar códigos de área em dois níveis, somente o nível principal (aproximadamente 100 códigos) é utilizado.

A correspondência dos sentidos entre os códigos é feita, primeiramente, através da verificação da intersecção das palavras recolhidas para cada área da palavra ambígua com as suas palavras vizinhas na sentença. Assim, a área que apresentar mais palavras coincidentes, respeitando-se um limite inferior pré-estabelecido, é escolhida como a área da palavra ambígua.

Os possíveis sentidos de cada área, exceto os da área geral, expressam uma relação de significado implícita, que os diferencia dos demais sentidos em outras áreas, não sendo o bastante para a desambiguação [Guthrie et al. 1991]. Devido a isso, os autores procuraram identificar qual o sentido na área selecionada, de forma que a intersecção das definições dos possíveis sentidos da área selecionada fosse comprovada com as palavras da sentença. Assim, o sentido com a maior intersecção é escolhido como o sentido da palavra ambígua.

Não houve avaliação do trabalho por parte dos autores. Eles citam o problema da quantidade limitada de informações disponíveis no LDOCE para cada código de área e propõem (para estudos futuros), que se utilize, em vez das definições do dicionário, córpus de diferentes áreas para a identificação de palavras significativas em cada área.

Uma proposta apresentada em 2000 por C. Brun [Brun 2000], utiliza um dicionário eletrônico de maneira diferente dos demais trabalhos. O modelo proposto de desambiguação é incorporado a um ambiente integrado para o processamento de textos, que dispõe de um dicionário bilíngüe bidirecional inglês/francês, de mecanismos de acesso a esse dicionário, de um *parser* e de um extrator de regras de desambiguação lexical de sentido. Esse extrator de regras usa as definições monolingües do dicionário para extrair, de forma automática, as regras de desambiguação lexical de sentido para o inglês. Para cada entrada do dicionário que possui mais de um sentido, há a análise dos exemplos, incluídos nos sentidos, a qual é feita pelo *parser* (gera as relações de dependência funcional presentes nesses exemplos, tal como sujeito-verbo, verbo-objeto e modificadores de vário tipos).

As regras lexicais produzidas para os sentidos de cada palavra, considerando a co-ocorrência da palavra na relação sintática, são então generalizadas para classes de palavras co-ocorrentes. Para tanto, a cada regra construída, a hierarquia conceitual da WordNet é consultada e a palavra que co-ocorre com a palavra ambígua na regra é substituída pelo código de todas as classes da WordNet que contêm essa palavra (classes semânticas). Dessa forma, aumenta-se, a abrangência do sistema, que pode cobrir casos de ambigüidade com outras palavras vizinhas, além daquelas apresentadas nos exemplos do dicionário.

Todas as regras são armazenadas em uma espécie de banco de dados para futuras consultas visando a resolver novos casos de ambigüidade. Nos casos em que nenhuma regra se aplica, é automaticamente atribuído o primeiro sentido do dicionário, considerado o mais freqüente, à palavra ambígua. No processo de aplicação das regras, aquelas produzidas para analisar diferentes partes de uma sentença podem cooperar entre si. Entretanto, pode haver também conflito entre as regras, quando várias dessas podem ser aplicadas para a desambiguação da mesma palavra. Sendo assim, Brun sugere uma medida que analisa o contexto das dependências funcionais.

De acordo com os resultados de avaliação do trabalho de Brun, percebe-se que a generalização usada não é suficiente para garantir regras abrangentes [Brun 2000]. Contudo, deve-se considerar que o trabalho é simples, pois não requer processamento lingüístico profundo, além de permitir a desambiguação de todas as categorias de palavras, conforme um conjunto de sentidos refinado. Mesmo não tendo sido mencionado pela autora, o uso dessa proposta para a tradução automática seria possível, pois cada sentido de uma palavra do inglês no dicionário usado, além da definição e dos exemplos em inglês, possui a sua tradução para o francês.

A última fonte de informações considerada nesse capítulo, dentro do conhecimento pré-codificado, é o *thesaurus*. São poucos os trabalhos (normalmente os mais antigos) que utilizam apenas um *thesaurus* como fonte de informações. Trabalhos mais recentes normalmente utilizam *thesauri* juntamente com outras fontes de informações.

M. Masterman em 1957 foi o primeiro a usar *thesaurus* como fonte de informações em seu trabalho, no qual também utilizava um dicionário bilíngüe [Masterman 1957].

O autor utiliza o *thesaurus Roget* para a desambiguação lexical de sentido em um sistema de tradução automática do latim para o inglês. A tradução da raiz de cada palavra em latim é retomada através de um dicionário latim-inglês. O sistema busca, para cada tradução, seu índice de classes correspondente no *thesaurus*. Assim, cada raiz de uma palavra em latim é associada a uma lista de índices do *Roget* relacionados a seus equivalentes em inglês. Os índices para todas as palavras na mesma sentença são então examinados para verificar as sobreposições entre eles. As palavras em inglês correspondentes aos índices com maior sobreposição são escolhidas para a tradução.

O *thesaurus Roget* foi também utilizado por A. B. Patrick em 1985 no seu trabalho de desambiguação [Patrick 1985]. O objetivo desse *thesaurus* distinguir os sentidos entre os verbos, examinando agrupamentos semânticos derivados desse *thesaurus*, em função de sinônimos mais fortes. Essas distinções de sentido bastante requintadas, uma vez que são baseadas somente nas palavras mais semanticamente relacionadas no *thesaurus*. De acordo com o autor, seu sistema é capaz de desambiguar entre vários sentidos de verbos como *inspire* (causar inspiração, inalar, respirar, etc.) e *question* (duvidar, fazer uma pergunta) com um alto nível de confiabilidade. Entretanto, o trabalho limitou-se a um pequeno conjunto de palavras e não foi levado adiante e nem sequer estendido.

Um observação importante é que, os trabalhos que utilizam a hierarquia conceitual da WordNet, descritos anteriormente, classificados como baseadas em léxicos computacionais, poderiam ser também considerados baseados em thesaurus, já que a hierarquia da WordNet, apesar de mais complexa, inclui as informações de um thesaurus.

2.1.2 Método Fundamentado em Corpus

Os trabalhos baseados em corpus (alguns voltados para a tradução automática) consideram abordagens supervisionadas e não-supervisionadas de desambiguação lexical de sentido, bem como abordagens desenvolvidas sob os diferentes paradigmas de aprendizagem de máquina.

2.1.2.1 Desambiguação Não-Supervisionada

A área de Aprendizado de Máquina (AM) vem crescendo no Processamento de Linguagem Natural, com a utilização de métodos que permitem extrair conhecimento automaticamente a partir de córpus, visando minimizar o problema da aquisição de conhecimento. Um córpus provê um conjunto de exemplos que, quando submetidos a algoritmos de AM, permitem o desenvolvimento de modelos capazes de descrever esses exemplos e de predizer o comportamento de novos exemplos. Os trabalhos baseados em córpus, também chamadas de empíricos, realizam a desambiguação com o uso de informações obtidas automaticamente a partir de um córpus. Nesses trabalhos, normalmente há uma etapa de treinamento, que resulta no aprendizado do modelo de desambiguação, com base no córpus. Após o treinamento, o modelo de desambiguação é gerado (com exceção dos trabalhos baseados em instâncias) e pode ser usado para desambiguar novos casos de ambigüidades.

Os trabalhos não-supervisionados que serão mencionados a seguir utilizam vários algoritmos para diferenciar os sentidos, dentre eles, os baseados em diferentes técnicas de *clustering*. Também são apresentados trabalhos que empregam outros algoritmos, associados ao uso de córpus paralelos, para a desambiguação na TA.

Um modelo estatístico da Teoria da Informação, foi utilizado por um grupo de pesquisadores em 1991 [Brown et al. 1991], o qual era baseado em informação mútua para a seleção lexical de itens ambíguos na tradução automática do francês para o inglês. Para tanto, é usado um corpus paralelo entre as duas línguas onde se buscam as possíveis traduções das palavras do francês para o inglês que apresentem um alinhamento direto (um para um). Logo em seguida, um possível conjunto de características significativas é definido para a distinção, de acordo com o contexto local das sentenças em francês (língua-fonte) ou inglesas (língua alvo). As características incluem diferentes palavras do contexto da palavra ambígua na língua-fonte, por exemplo, o primeiro substantivo à direita, o primeiro verbo à direita, a primeira palavra à esquerda etc (as possíveis palavras vizinhas da palavra ambígua).

O algoritmo Flip-Flop[15] tenta encontrar, para cada palavra ambígua, uma única característica (dentre as pré-definidas) que indica, com um alto nível de confiabilidade, qual a sua tradução. Esse algoritmo considera apenas duas possíveis classes de traduções de uma determinada palavra ambígua de uma língua fonte, ou seja, considere uma palavra em frânçês w e que palavras em inglês que são possíveis traduções de w tenha sido divididas em

[15]Modelo estatístico empregado em [Brown et al. 1991]

duas classes. Considerando o problema binário sobre a informação em potencial devido as duas classes de palavras em inglês. O processo é iterativo e, a cada interação, o algoritmo procura aumentar a informação mútua (possíveis relações sintáticas e semânticas) entre o idioma francês e o inglês obtida com o emprego da característica para a desambiguação da palavra em questão. O critério de parada é indicado pela estabilização da informação mútua, ou seja, na iteração em que essa medida não pode mais ser aumentada.

Definida a característica que melhor divide o conjunto de treinamento, os exemplos (em francês) passam por um processo de *clustering*, ou seja, são divididos em dois grupos, para as duas traduções possíveis, de acordo com o valor que apresentam para essa característica. Na verdade, cada um dos grupos pode ter várias traduções, mas elas são ranqueadas de acordo com uma estimativa da probabilidade de cada uma das traduções no córpus do inglês. A tradução com a maior probabilidade de ocorrência em cada grupo é então escolhida para etiquetar a palavra ambígua em todos os exemplos do francês selecionados.

Na questão prática, uma avaliação do módulo de desambiguação lexical de sentido foi realizada. O modelo foi treinado para a desambiguação das 500 palavras mais comuns do inglês e as 200 mais comuns do francês e o módulo final foi incorporado a um sistema de tradução automática por transferência, também estatístico, na fase de análise. Na tradução de 100 sentenças aleatoriamente selecionadas com essas palavras, os autores relatam uma diminuição de 13% na taxa de erro das traduções resultantes do sistema com o uso do módulo de desambiguação.

Em 2002, P. Pantel e D. Lin propõem um algoritmo de *clustering* de similariedade distribucional (palavras que ocorrem no mesmo contexto tendem a serem similares) para agrupar as palavras de textos em grupos semanticamente similares, que correspondem aos sentidos das palavras, voltado para a Recuperação de Informação [Pantel e Lin 2002]. É importante a análise do contexto de ocorrência da palavra em um corpus, pois de acordo com as abordagens distribucionais, as palavras que ocorrem nos mesmos contextos tendem a ser similares. Devido a isso, o algoritmo proposto pelo autores identifica os sentidos das palavras agrupando-as, de acordo com a sua similaridade distribucional, em *clusters* com identificações numéricas que representam os sentidos.

Baseado nos sentidos da WordNet, os autores avaliaram seu trabalho fazendo um mapeamento entre os sentidos da WordNet e os indicados pelos clusters. Para tanto, utilizaram textos do gênero jornalístico. De acordo com o resultado obtido pelos autores, o algoritmo supera outros algoritmos de *clustering*, como os hierárquicos ou híbridos, na

tarefa de desambiguação lexical de sentido.

2.1.2.2 Desambiguação Supervisionada

Os trabalhos mencionados a seguir foram desenvolvidos de acordo com diferentes paradigmas de aprendizado[16], utilizando vários algoritmos. Dentre eles, há trabalhos que envolve combinação ou comparação entre diferentes paradigmas e outros que aperfeiçoam abordagens anteriores, considerando também o uso de outros paradigmas. Alguns desses trabalhos são voltados para a tradução automática.

Em 1.994, Yarowsky propôs um trabalho [Yarowsky 1994] que utiliza uma técnica baseada nas Listas de Decisão de Rivest [Rivest 1987] para a resolução de ambigüidades lexicais de vários tipos e aplica essa mesma técnica em seus diversos trabalhos posteriores [[Yarowsky 1995] e [Yarowsky 1994]], particularmente, para a ambigüidade lexical de sentido. Essa técnica de listas de decisão consiste em processar os exemplos de treinamento para extrair as características relevantes, que recebem pesos de acordo com uma medida de verossimilhança. Tal medida considera distinções entre apenas dois sentidos para cada palavra, mas, de acordo com o autor, pode ser adaptada para distinções entre qualquer quantidade de sentidos. A lista de todas as características ordenadas (em ordem decrescente) de acordo com seus pesos constitui a lista de decisão. Para desambiguar novos exemplos, a lista de decisão é percorrida em ordem e a característica com o peso mais alto no exemplo de teste seleciona o sentido mais apropriado.

Em 1998, um trabalho supervisionado que utiliza redes neurais para aprender um modelo de classificação, aplicado na Recuperação de Informações foi apresentado por G. Towell e E. M. Voorhess [Towell e Voorhees 1998].

A desambiguação lexical de sentido ocorre através de um classificador que combina as saídas de duas redes de características distintas, uma com o contexto local e outra com o contexto global. O contexto local inclui informações sobre a ordem das palavras, suas distâncias e algumas informações sobre a estrutura sintática, considerando todas as palavras vizinhas à palavra ambígua na sentença. O contexto global é constituído de substantivos que têm a probabilidade de co-ocorrer com determinados sentidos da palavra ambígua.

Um ponto importante a ser observado nesse trabalho é que a rede precisa ser inicialmente alimentada com exemplos configurados por marcações (*tags*). Para isso, Towell e

[16]Simbólico, estatístico, conexionista e baseado em instâncias.

Voorhess utilizam um corpus de onde extraem o mesmo número de exemplos para cada sentido de uma palavra ambígua, objetivando a eliminação dos efeitos da freqüência de cada sentido.

Usaram-se três palavras ambíguas de três classes gramaticais (verbo, substantivo e adjetivo), para testar o desempenho individual e em conjuntos das redes, baseado nos diferentes números de exemplos. De acordo com o resultados obtidos, o melhor desempenho é quando há a combinação das redes com a utilização do maior número de exemplos possíveis [Towell e Voorhees 1998].

Em decorrência do uso de redes neurais para o aprendizado, cria-se o problema do grande número de exemplos etiquetados necessários, consequentemente, eleva-se o tempo necessário para o treinamento da rede e também para a classificação de novos casos. Esse tempo elevado pode causar prejuízos para a sua utilização, por exemplo, em um sistema de recuperação de informações on-line.

Para tentar minimizar a quantidade de exemplos necessários, os autores propõem um algoritmo semi-supervisionado, em que a rede neural procura gerar classificadores precisos com um pequeno número de exemplos de treinamento etiquetados e um grande número de exemplos não etiquetados. Esse algoritmo se baseia na similaridade dos exemplos etiquetados com aqueles que não possuem uma etiqueta para criar "exemplos sintéticos". Sendo esses exemplos sintéticos criados a partir de um exemplo etiquetado utilizado como semente. A partir dele, são coletados exemplos próximos. Não são incluídos, nesse conjunto, exemplos que já possuem outras etiquetas ou exemplos que são mais próximos de outros exemplos etiquetados. Desse modo, formam-se grupos de exemplos que recebem o sentido do exemplo mais similar.

Um trabalho comparativo é realizado por Y.K. Lee e H. T. Ng em 2002 [Lee e Ng 2002], que consiste em avaliar, de forma comparativa, quatro algoritmos de aprendizado para a desambiguação lexical de sentido e quatro tipos de conhecimento. O interesse dos autores é analisar a contribuição de cada tipo de conhecimento nos diferentes algoritmos e a viabilidade do uso de um método de seleção automática de características para o treinamento.

Dentre os tipos de conhecimento são utilizados: características para a categoria gramatical da palavra ambígua e de três palavras vizinhas à esquerda e à direita; características binárias para todas as palavras vizinhas à palavra ambígua na sentença, lematizadas, excluindo-se palavras de uma lista de *stop-words*; características binárias para 11 *collocations*, por exemplo, a primeira palavra à direita e à esquerda da palavra ambígua; e

características para relações e informações sintáticas da sentença, geradas por um *parser* baseado em dependências.

A categoria gramatical da palavra ambígua é uma informação importante, pois, o número e a natureza das relações sintáticas dependem dessa informação. Por exemplo, para o substantivo são representados o seu núcleo na relação de dependência, a categoria gramatical do núcleo, a voz do núcleo, caso ele seja um verbo, e a posição relativa do núcleo.

Para todas as características binárias, em que o algoritmo simplesmente verifica a sua existência ou não no exemplo para gerar o modelo, os algoritmos foram testados com e sem o uso de um método para a seleção de características. Esse método consiste de um único parâmetro, configurado com um valor (três, neste caso), que determina a quantidade mínima de vezes que a característica ocorre nos exemplos de treinamento. Assim, na geração do modelo para cada palavra ambígua, para cada característica, o método verifica se ela ocorre para algum sentido da palavra pelo menos três vezes. Se isso acontecer, tal característica é utilizada.

Os algoritmos analisados são: SVM (Support Vector Machines), AdaBoost, Naive Bayes e C4.5. Todos foram usados nas versões implementadas no ambiente Weka[17], com seus parâmetros padrões.

Na avaliação dos trabalhos foram utilizados os exemplos de treinamento e teste das duas primeiras edições do SENSEVAL para a tarefa de classificação de um conjunto de palavras. Foram consideradas 36 palavras na primeira edição e 73 na segunda, compreendendo substantivos, verbos e adjetivos.

Segundo os autores o classificador com o melhor desempenho foi o SVM, mesmo comparando todos os resultados com os obtidos pelos três sistemas mais bem colocados nas duas edições do SENSEVAL.

A contribuição relativa de cada um dos tipos de conhecimento para o desempenho do trabalho depende do algoritmo utilizado. Por exemplo, as características de *collocations* são as que mais contribuem para o SVM, enquanto que as características das categorias gramaticais são as mais relevantes para o Naive Bayes.

[17]http://www.cs.waikato.ac.nz/ ml/weka/

2.1.3 Método Híbrido

A utilização de diferentes combinações, consistindo do uso de conhecimento codificado manualmente ou proveniente de qualquer recurso lingüístico já existente e do uso de corpus em alguma das variações no modo de aprendizado (supervisionado, não-supervisionado ou semi-supervisionado), nos paradigmas e nos algoritmos, é o que compõem os trabalhos híbridos.

Trabalhos como em 1.991 [Dagan, Itai e Schwall 1991] e em 1.994 [Dagan e Itai 1994] propõem um mecanismo de desambiguação fundamentado em corpus comparativos entre L1 e outra língua L2 e nas correspondências lexicais entre essas duas línguas indicadas por um dicionário bilíngüe. Os autores trabalham com a hipótese de que os vários sentidos de uma palavra ambígua em uma língua se manifestam por diferentes itens lexicais em outras línguas. Os autores procuram desambiguar os sentidos das palavras de L1, usando as suas traduções em L2. Pra isso, o mecanismo procura identificar a tradução correta das palavras de conteúdo de uma sentença em L1 na L2, cujo processo foi denominado, pelos autores, de "seleção da palavra-alvo".

A seleção da palavra-alvo envolve dois procedimentos: o primeiro lingüístico e o segundo estatístico. No lingüístico, de início são identificadas todas as possíveis traduções dessa palavra na L2 disponíveis em um dicionário bilíngüe L1 pra L2. Em seguida, é realizada a análise sintática superficial da sentença na qual a palavra ambígua ocorre, de modo a identificar relações sintáticas binárias entre as palavras. As relações sintáticas consideradas importantes para as línguas utilizadas são: relações entre o verbo e o sujeito, seus complementos e adjuntos; relações entre o substantivo e seus complementos e adjuntos e relações entre os adjetivos e advérbios e seus modificadores.

As possíveis traduções da palavra ambígua são obtidas através da aquisição dos exemplos de sentenças oriundas do corpus de L2, submetendo esses exemplos à análise sintática superficial, de forma que sejam identificadas suas relações sintáticas. Interessante seria, se um analisador sintático similar fosse utilizado, para extrair relações sintáticas que pudessem ser comparadas às de L1. Não sendo isso possível, precisa-se criar um mecanismo para mapear relações (filtro no processo de seleção da palavra-alvo) em diferentes idiomas, para que elas possam ser comparáveis, produzidas por analisadores diferentes.

Como resultado desse procedimento, obtém-se um subconjunto de ocorrências das possíveis traduções da palavra ambígua na L2. Se essa palavra só ocorre com uma tradução em L2, nas diversas relações sintáticas da L1 em que ela aparece, então essa tradução é

automaticamente escolhida. Caso contrário, é preciso recorrer ao procedimento estatístico.

No procedimento estatístico, inicialmente há a contagem do número de ocorrências de cada uma das possíveis traduções da palavra ambígua na L2 adquirido no procedimento lingüístico. Este número é utilizado como parâmetro inicial para um modelo probabilístico simples que se baseia na freqüência de ocorrências de cada uma das possíveis traduções, sendo essas traduções restritas pela relação sintática. A seleção de uma palavra-alvo para cada palavra ambígua leva em consideração, também, a ocorrência dessa palavra ambígua em mais de uma relação sintática (por exemplo, um verbo irá ocorrer na relação com o seu sujeito e com os seus objetos). A escolha da tradução deve ser consistente com todas as relações em que a palavra ocorre.

Para realizar o experimento de avaliação do trabalho os autores utilizaram, como L1, 103 palavras ambíguas do hebreu e 54 palavras ambíguas do alemão e, como L2, o inglês. Além disso, para a análise de freqüências, usaram um corpus em inglês com 25 milhões de palavras.

De acordo com os seus autores, o trabalho tem como a vantagem o fato de não requerer um corpus etiquetado com sentidos, nem tampouco corpus paralelos corretamente alinhados para a o treinamento do modelo. Entretanto, ele apresenta problemas, dentre os quais: o filtro das relações sintáticas exige que as duas línguas sejam sintaticamente similares, de modo que possam ser analisadas pelo mesmo *parser* ou, pelo menos, que as relações de uma língua possam ser mapeadas nas relações da outra; possíveis ambigüidades entre as relações, ou seja, uma relação de uma língua ter mais de uma equivalente em outra língua. Os autores, não apontam uma estratégia que possa contornar esses problemas. Ademais, o fato de terem realizado parte do processo manualmente pode ter deixado de revelar outros problemas. Por fim, apesar de os autores afirmarem que seu objetivo é a desambiguação monolingüe em L1, seus trabalhos indicam apenas parte desse processo, que corresponde à desambiguação multilingüe, na tradução de L1 para L2. O mecanismo proposto, segundo os próprios autores, tem, de fato, aplicação direta para a desambiguação na tradução automática.

Em 1992, D. Yarowsky apresentou um trabalho não-supervisionado para a desambiguação de sentidos em textos irrestritos utilizando um modelo estatístico sobre as categorias mais genéricas do *thesaurus* Roget, localizando o contexto da palavra ambígua na descrição dessas categorias no *thesaurus* [Yarowsky 1992]. Nesse trabalho, o autor considera o sentido como sendo a correspondência com as categorias genéricas do *thesaurus*, ou seja, um sentido é atribuído a uma palavra ambígua a partir da identificação de qual

a categoria na qual ela se enquadra. Essa identificação se dá pela análise do contexto da palavra ambígua, seguida da sua comparação com palavras indicativas de cada uma das categorias.

Utiliza um corpus para extrair todos os possíveis exemplos em que pelo menos uma palavra (seu lema) da categoria aparece, juntamente com uma janela de 100 palavras (50 à direita e 50 à esquerda), denominada "assinatura" dessa palavra, para obter um contexto de palavras indicativas de cada categoria do *thesaurus*. Um etiquetador morfossintático é utilizado no pré-processamento para usar somente as palavras da categoria gramatical adequada. Os exemplos reunidos podem incluir palavras ambíguas, que pertencem a outras categorias. Essa possível ambigüidade pode ser minimizada, através da atribuição de peso 1, a princípio, a cada ocorrência da busca da palavra, sendo esse peso dividido pelo número de vezes que a palavra aparece em todos os exemplos.

Um algoritmo é usado para estimar a informação mútua para identificar as palavras mais comuns em cada categoria, ou seja, mais indicativas dessa categoria, de acordo com os exemplos. O mecanismo desse algoritmo funciona com a divisão da probabilidade da palavra aparecer na categoria para a qual foi encontrada pela probabilidade de ocorrer em todas as outras categorias, obtendo com isso, uma medida da "importância" da palavra para identificar uma categoria do *thesaurus*.

Consequentemente, a lista de palavras importantes ou indicativas de cada categoria inclui, em média, 3.000 palavras de maior probabilidade de co-ocorrer com os membros da categoria. Segundo o autor, essa quantidade é muito mais significativa que as pequenas definições de palavras disponibilizadas nos dicionários.

Ainda de acordo com ele, o seu trabalho é mais adequado para a extração de informações sobre um contexto global e, portanto, à desambiguação de substantivos.

O algoritmo utilizado no trabalho de Yarowsky, falha quando um sentido se distribui por várias categorias, ou seja, quando uma distinção de sentido deve ser realizada independentemente da categoria, gerando um problema nesse trabalho. Um exemplo desse problema seria o sentido "vantagem" da palavra *interest* (como em *self-interest*). Esse sentido pode ocorrer independentemente da área (finanças, música, etc.).

Um trabalho híbrido baseado em similaridades de palavras e contextos (sentenças), que também pode ser considerado um trabalho de aprendizado semi-supervisionado, foi proposto por Y. Karov e S. Edelman em 1998 [Karov e Edelman 1998]. Duas palavras são similares se aparecem em contextos similares (se possuem palavras similares). Como

o trabalho se baseia em definições de dicionário, isso pode levar a uma medida transitiva de similaridade, na qual dois contextos são considerados similares mesmo que não compartilhem as mesmas palavras, e duas palavras são consideradas similares mesmo que não compartilhem palavras vizinhas similares. Utilizando do conhecimento codificado em um dicionário eletrônico (podendo também ser em um thesaurus) e um algoritmo de aprendizado a partir de corpus.

O trabalho de Karov e Edelman tenta minimizar o problema de dados esparsos, ou seja, da inexistência de exemplos (ou existência de poucos exemplos) de cada sentido de uma palavra, e também a eliminação da necessidade de etiquetação manual prévia dos exemplos de treinamento.

Segundo os autores, na sua avaliação, esse trabalho leva a resultados melhores que os dos trabalhos baseados apenas em co-ocorrência, principalmente quando os dados são esparsos [Karov e Edelman 1998].

Paliouras e outros pesquisadores em 1999 empregaram um algoritmo de aprendizado simbólico de árvores de decisão (C4.5) para a desambiguação lexical de sentido que focaliza a produção de um modelo genérico, capaz de desambiguar todas as palavras do conteúdo de um texto, para a aplicação da extração de informações [Paliouras et al. 2000].

Esses foram os principais trabalhos relacionados ao Método Híbrido de Desambiguação Lexical de Sentido. Trabalhos secundários foram tratados ao longo dos anos, mas todos baseados nesses trabalhos anteriormente citados.

2.2 Trabalhos voltados ao domínio da Química

Nesta subseção são apresentados alguns sistemas que, como o RALQ, concentram-se na tarefa de processar a linguagem química.

A ferramenta “Chemfinder” produz o desenho da estrutura química molecular para um nome químico, isto é, através de um nome de consulta o sistema gerará o desenho da estrutura molecular correspondente a consulta de entrada, além de fornecer informações detalhadas sobre o composto. Esse sistema inclui uma lista de sinônimos entre os nomes usuais e oficiais. Ela é baseada em uma análise lingüística do nome do composto químico de entrada, aceita mais de uma variante de nomenclatura e compostos que não fazem parte das nomenclaturas oficiais não são tratados por essa ferramenta(IUPAC, IUBMB[18]

[18] *International Union of Biochemistry and Molecular Biology* - União Internacional de Bioquímica e Biologia Molecular

e CAS[19]).

Na verdade, a ferramenta "Chemfinder"[20] é uma máquina de busca cuja entrada será o nome do composto químico e cuja saída é o desenho da estrutura química do composto, além das informações técnicas sobre o composto.

A base do sistema "Chemfinder" é o *parser Name=Struct* como descrito em [Brecher 1999] e [Cambridgesoft 2005].

A ferramenta alemã "ACD/Name[21]" provida pelo *ScienceServe/Scientific Software Solutions* oferece a geração de estruturas a partir de nomes de composto químico e vice-versa. Como é uma ferramenta comercial, nenhuma documentação detalhada está disponível (há descrição do tipo de abordagem na qual o sistema se baseia).

Um trabalho mais recente na área da lingüística computacional voltada para a química orgânica foi desenvolvido por Stefanie Anstein e Gerhard Kremer [Anstein e Kremer 2005].

Esse trabalho consiste no desenvolvimento de um sistema de processamento de linguagem natural construído na linguagem de programação lógica Prolog. Esse sistema gera uma representação semântica intermediária para um nome de composto químico-orgânico.

O processamento dessa estrutura semântica gerará cadeias de caracteres SMILES [22] que descreverão a estrutura molecular do composto e que permitirão a classificação do mesmo. O sistema é apto para analisar, inclusive, nomes usuais dos compostos químicos.

[19] *Chemical Abstracts Service* - Serviço Abstrato Químico

[20] www.chemfinder.com

[21] Disponível em: http://www.scienceserve.com/Software/ACD/ACD_Name.htm

[22] Uma cadeia de caracteres SMILES é uma notação estrutural de uma molécula que consecutivamente lista os elementos de cadeia principal com suas propriedades e ramificações [Anstein e Kremer 2005].

3 Fundamentação Teórica

Nesse capítulo serão apresentados as técnicas e ferramentas que darão ambasamento teórico para a construção do RALQ.

3.1 Princípios da Química Orgânica

A contextualização dos princípios da Química Orgânica nesta seção tem o objetivo de dar um embasamento teórico do domínio a que a Teoria Léxico Gerativo será aplicada e mostrar como a literatura da Química Orgânica trata a formação dos nomes orgânicos e, conseqüentemente, a estrutura química dos compostos químicos.

A química orgânica é o ramo da química que estuda os compostos do elemento carbono.

O carbono (C) é o principal elemento que aparece na formação dos compostos orgânicos. Além dele, são também freqüentes o hidrogênio (H), o oxigênio (O), o nitrogênio (N) e ainda os halogênios, o enxofre (S) e o fósforo (P), conforme mostra a tabela 1.

3.1.1 Encadeamento : Uma Propriedade Importante

Os átomos de carbono têm propriedade de se unir, formando estruturas denominadas cadeias carbônicas. Essa propriedade é a principal responsável pela existência de milhões de compostos orgânicos. Na figura 1 estão alguns exemplos de cadeias:

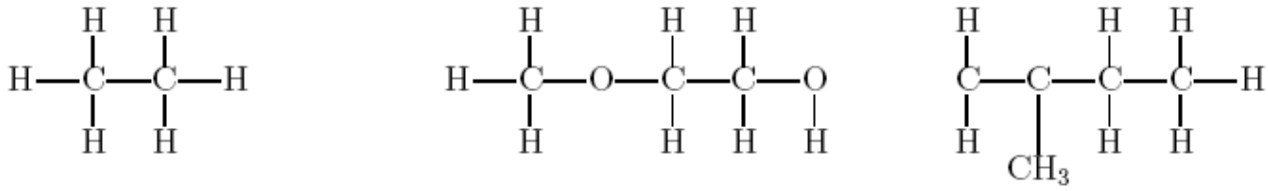

Figura 1: Tipos de Cadeias Carbônicas.

Elemento	Valência	Possibilidade de ligações
Carbono	Tetravalente	—C—  C═ —C≡ ═C═
Hidrogênio	Monovalente	H—
Oxigênio (enxofre)	Bivalente	—O— O═
Nitrogênio (fósforo)	Trivalente	—N— —N═ N≡
Halogênio	Monovalente	F— Cl— Br— I —

Tabela 1: Principais Elementos da Química Orgânica e suas Valências

Uma cadeia carbônica pode apresentar, além de átomos de carbono, átomos de outros elementos, desde que estes estejam entre os átomos de carbono. Os elementos diferentes de carbono que mais frequentemente podem fazer parte da cadeia carbônica são: O, N, S, P. Nesta situação, tais átomos são denominados heteroátomos [Usberco e Salvador 2001].

3.1.2 Classificação do Carbono

Numa cadeia, cada carbono é classificado de acordo com o número de outros átomos de carbono a ele ligados, conforme mostrado na tabela 2.

Carbono	Definição
Primário	ligado diretamente, no máximo, a 1 outro carbono
Secundário	ligado diretamente a 2 outros carbono
Terciário	ligado diretamente a 3 outros carbono
Quaternário	ligado diretamente a 4 outros carbono

Tabela 2: Classificação do Carbono

3.1.3 Classificação das Cadeias Carbônicas

Cadeia carbônica ou estrutura química é o conjunto de todos os átomos de carbono e de todos os heteroátomos que constituem a molécula de qualquer composto orgânico [Usberco e Salvador 2001].

Existem vários critérios para classificar as cadeias. O primeiro é quanto à disposição dos átomos: Cadeia aberta, acíclica ou alifática e Cadeia fechada ou cíclica

Cadeia aberta apresenta pelo menos duas extremidades e nenhum ciclo ou anel. Por exemplo:

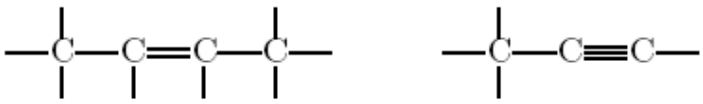

Figura 2: Cadeias Carbônicas Abertas.

A Cadeia fechada não apresenta extremidades, e os átomos originam um ou mais ciclo, (anéis). Por exemplo:

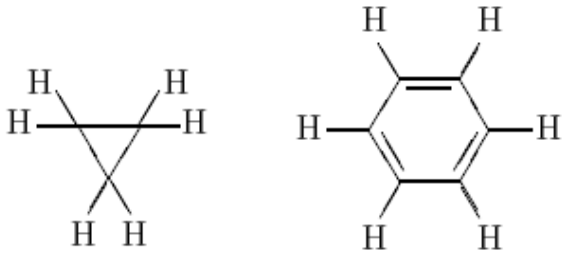

Figura 3: Cadeias Carbônicas Fechadas.

As cadeias abertas podem, ainda, ser subdivididas quanto à disposição dos átomos de carbono, dando, assim, a Cadeia normal, reta ou linear e a Cadeia ramificada.

A Cadeia normal apresenta somente duas extremidades e seus átomos estão dispostos numa única seqüência, de acordo com a figura 4:

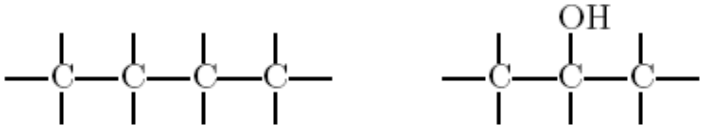

Figura 4: Cadeias Carbônicas Normais.

Já a cadeia ramificada apresenta no mínimo três extremidades e seus átomos não estão dispostos numa única seqüência, como mostra a figura 5.

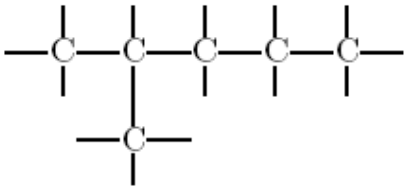

Figura 5: Cadeia Carbônica Ramificada.

A classificação das Cadeias Carbônicas ou Estruturas Químicas pode também ser feita por meio da natureza da ligação entre átomos de carbono, ou seja, Cadeia Saturada e Cadeia Insaturada.

As cadeias saturadas apresentam somente ligações simples entre os átomos de cadeia, como mostra a figura 6.

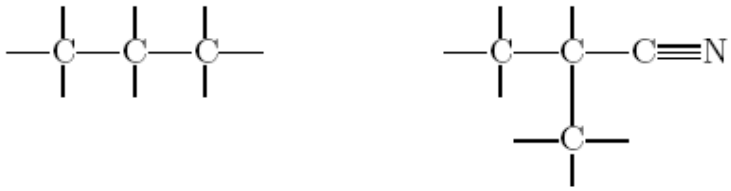

Figura 6: Cadeias Carbônicas Saturadas.

As cadeias Insaturadas ou não-saturadas apresentam pelo menos uma dupla ou tripla ligação entre seus átomos, como está apresentado na figura 7.

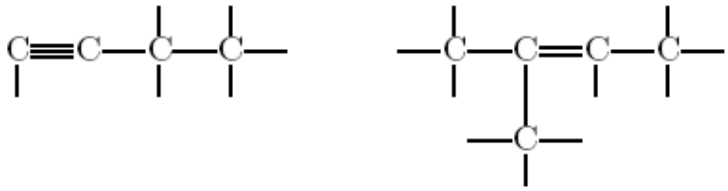

Figura 7: Cadeias Carbônicas Insaturadas.

A natureza dos átomos que compõem a cadeia carbônica é uma outra característica de classificação. A cadeia carbônica ou a estrutura química do composto pode ser Cadeia homogênea (mostrada na figura 8), a qual é constituída somente por átomos de carbono, e a Cadeia heterogênea, que apresenta pelo menos um heteroátomo na cadeia carbônica (figura 9).

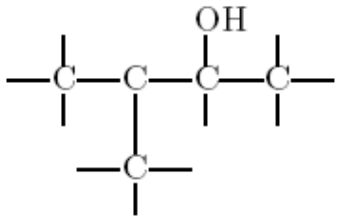

Figura 8: Cadeia Carbônica Homogência

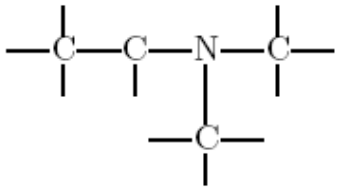

Figura 9: Cadeia Carbônica Heterogênia

3.1.4 Hidrocarbonetos e sua Nomeclatura

A Química Orgânica agrupa as substâncias químicas com base em critérios de semelhanças estruturais. Desse modo, cada função orgânica é caracterizada por um grupo funcional e apresenta uma nomenclatura característica. A nomenclatura oficial vem sendo desenvolvida pela IUPAC[1] (União Internacional de Química Pura e Aplicada). Algumas substâncias, no entanto, ainda são identificadas pelos nomes consagrados pelo uso comum: é a nomenclatura usual.

De acordo com a IUPAC, o nome de uma substância de cadeia aberta é formado pela união de três componentes, cada um deles indicado na característica do composto: o radical da cadeia principal indica o número de átomos de carbono de cadeia. O intermediário indica o tipo de ligação entre os carbonos. O sufixo indica a função a que pertence o composto orgânico. O quadro da figura 10 ilustra a nomeclatura de algumas funções orgânicas.

<table>
<tr><td colspan="3">Nome</td></tr>
<tr><td>Radical
Nº de carbonos</td><td>Intermedário
Saturação da Cadeia</td><td>Sufixo
Função</td></tr>
<tr><td>1 C – MET</td><td>Saturadas – AN</td><td>Hidrocarbonetos
O</td></tr>
<tr><td>2 C – ET</td><td rowspan="7">Insaturadas
1 dupla – EN
2 duplas – DIEN
3 duplas – TRIEN
1 – tripla – IN
2 – triplas – DIIN
3 – triplas – TRIIN</td><td rowspan="2">Álcool
OH</td></tr>
<tr><td>3 C – PROP</td></tr>
<tr><td>4 C - BUT</td><td rowspan="3">Aldeído
AL
Cetona
ONA</td></tr>
<tr><td>5 C – PENT</td></tr>
<tr><td>6 C – HEX</td></tr>
<tr><td>7 C – HEPT</td><td rowspan="3"></td></tr>
<tr><td>8 C – OCT</td></tr>
<tr><td>9 C – NON</td><td rowspan="3">1 dupla e 1 tripla - ENIN</td><td rowspan="3">Ácido Carboxílico
ÓICO</td></tr>
<tr><td>10 C – DEC</td></tr>
<tr><td>11 C – UNDEC</td></tr>
</table>

Figura 10: Quadro que representa a Nomeclatura de algumas Funções Orgânicas.

[1] www.iupac.org

3.1.5 Hidrocarbonetos Alifáticos

3.1.5.1 Alcanos ou parafinas

São hidrocarbonetos alifáticos saturados, ou seja, apresentam fórmulas moleculares formadas somente por Carbono (C) e Hidrogênio (H) e possuem cadeia aberta constituída apenaspor ligações simples. A figura 11 mostra um exemplo do funcionamento da regra da nomeclatura dos Alcanos:

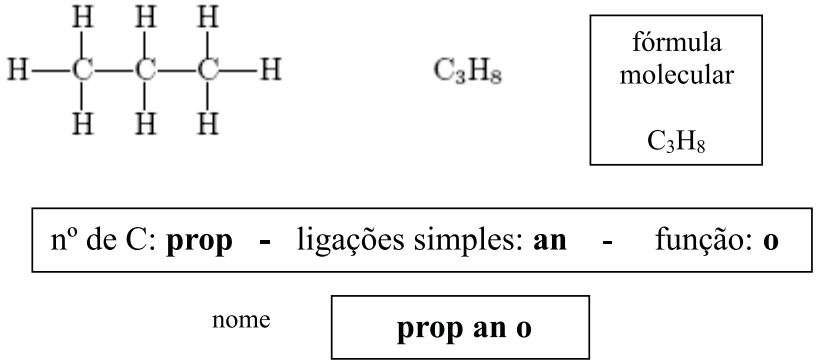

Figura 11: Nomeclatura dos Alcanos.

3.1.5.2 Alquenos, alcenos ou olefinas

São hidrocarbonetos alifáticos insaturados que apresentam uma dupla ligação. Quando um alqueno apresenta quatro ou mais átomos de carbono, torna-se necessário indicar a localização da dupla ligação através de um número. Esse número é obtido numerando-se a cadeia a partir da extremidade mais próxima da insaturação (dupla ligação). O número que indica a posição da dupla ligação deve ser o menor possível e deve anteceder o nome do composto, precedido por um hífen. Exemplo: A partir do nome, pode-se determinar as fórmulas:

3.1.5.3 Alquinos ou alcinos

São hidrocarbonetos alifáticos insaturados por uma tripla ligação. As regras para estabelecer a nomenclatura dos alquinos são as mesmas utilizads para os alquenos, como mostra o exemplo abaixo:

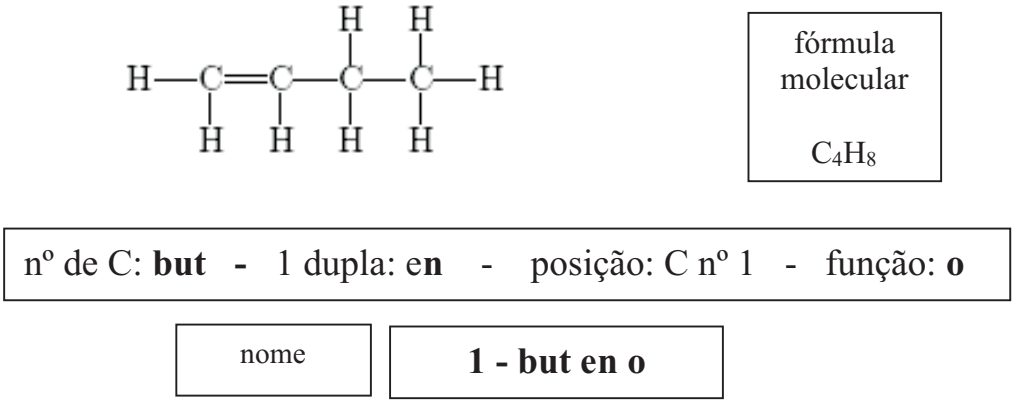

Figura 12: Nomeclatura dos Alcenos.

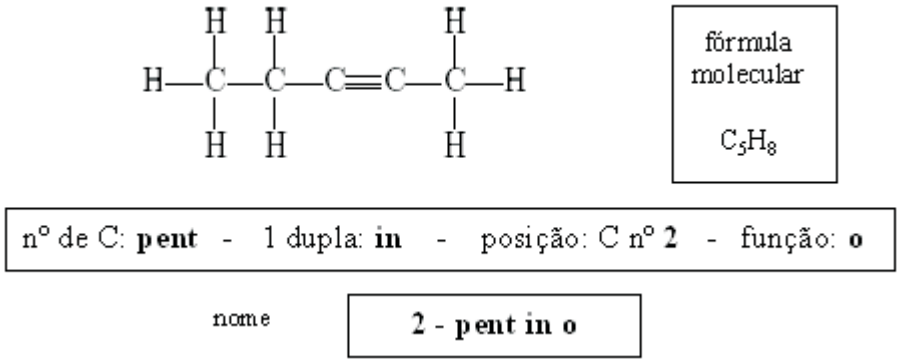

Figura 13: Nomeclatura dos Alcinos.

3.1.5.4 Alcadienos ou dienos

São hidrocarbonetos alifáticos insaturados por duas duplas ligações. Nesse caso, como existem duas duplas ligações na cadeia, quando necessário, seu nome é precedido de dois números, separados por vírgula. A nomeclatura dos nomes dos compostos desse hidrocarboneto é dada na figura 14.

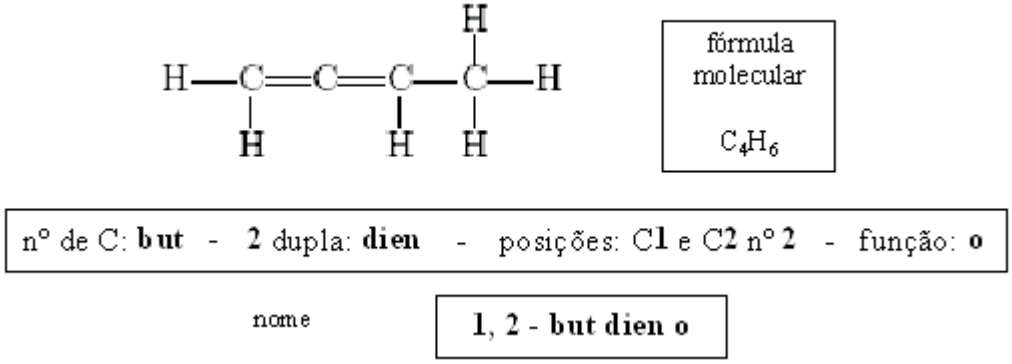

Figura 14: Nomeclatura dos Alcadienos.

3.1.6 Hidrocarbonetos Ramificados

3.1.6.1 Radicais

Radicais são átomos ou agrupamentos de átomos eletricamente neutros que apresentam pelo menos um elétron não-compartilhado (valência livre). Podem ser representados, genericamente, por R - [Usberco e Salvador 2001].

Os principais radicais ou grupos orgânicos monovalentes são obtidos, a partir de hidrocarbonetos, pela retirada de um átomo de hidrogênio (H). A figura 15, exemplifica a formação de um radical:

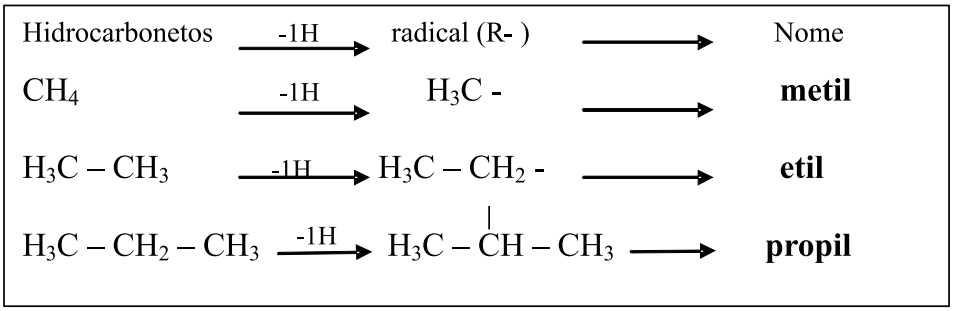

Figura 15: Formação dos Radicais.

3.1.7 Nomenclatura de Hidrocarbonetos Ramificados

3.1.7.1 Hidrocarbonetos Alifáticos Saturados

Alcanos

Para nomenclatura de alcanos ramificados, são usados as seguintes regras da IUPAC:

- Regra 1 (Regra da Cadeia): Determinar a cadeia principal e seu nome. Onde, a cadeia principal é a maior seqüência contínua de átomos de carbono, não necessariamente dispostos em linha reta.
- Regra 2: Verificar quantas ramificações apresenta o composto e quais são;
- Regra 3: Localizar as ramificações, enumerando a cadeia; esta numeração deve obedecer **à regra dos menores números**, ou seja, "a cadeia carbônica deve ser enumerada, segundo as duas possibilidades (ou dois sentidos); prevalecerá, para efeito de nomenclatura, o sentido que produzir menor somatóriode posições de ramificações";

- Regra 4: Quando houver mais de um radical do mesmo tipo, seus nomes devem ser precedidos de radiciais que indicam suas quantidades (multiplicadores): di, tri, tetra etc.

- Regra 5: Quando houver dois ou mais radicais de tipos diferentes, seus nomes devem ser escritos em ordem alfabética. Não se devem considerar, para efeito de ordem alfabética, os prefixos di, tri, séc, terc, etc.

A figura 16 é uma estrutura química que será utilizada para exemplificar a aplicação dessas regras na definição de um nome para essa estrutura.

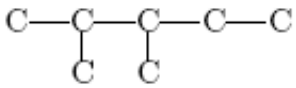

Figura 16: Estrutura Química.

Com a aplicação da regra da cadeia, acha-se a cadeia principal da estrutura química, como está ilustrado na figura 17

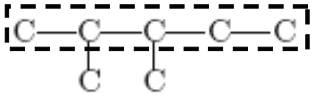

Figura 17: Cadeia Principal.

De acordo com a regra 2 (dois), a estrutura possui duas ramificações do tipo **metil**. Em seguida vem a aplicação da regra 3 (três), em que a numeração da cadeia principal é realizada (figura 18).

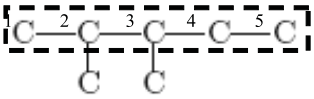

Figura 18: Numeração da Cadeia Principal.

Com isso, a regra dos menores números executa da seguinte forma:

1ª Possibilidade:

da direita para esquerda: | 3 + 4 = 7 |

2ª Possibilidade:

da esquerda para direita: | 2 + 3 = 5 |

Esses valores correspondem às posições em que as ramificações encontram-se ligadas na cadeia principal. A orientação para a nomeclatura, portanto, é dada na segunda possibilidade.

Neste exemplo, como os dois radicais são do mesmo tipo (**metil**), então, por meio da regra 4 (quatro), o nome do composto terá como prefixo **2,3-dimetil** (multiplicador).

Assim, o nome do composto, cuja estrutura química é dada pela figura 16, é **2,3-dimetil-pentano**, uma vez que não houve a necessidade da aplicação da regra 5 (cinco) devido ao fato de não haver ocorrência de outro radical.

Uma observação sobre a nomeclatura dos Alcanos é que quando em uma estrutura, duas ou mais cadeia carbônicas apresentarem o mesmo número máximo de carbonos, será considerada principal a cadeia que tiver o maior número de ramificações.

No caso dos Alcenos, Alcinos e Alcadienos, a única diferença em relação aos Alcanos é que as insaturações devem obrigatoriamente fazer parte da cadeia principal e devem receber sempre os menores números possíveis.

3.1.8 Funções Orgânicas Contendo Oxigênio

3.1.8.1 Álcoois

São compostos orgânicos derivados dos hidrocarbonetos pela substituição de H de carbono saturado por -OH (hidroxila), como mostra a figura 19.

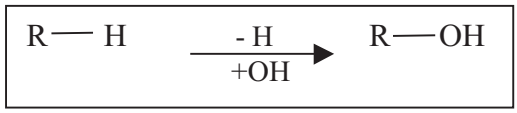

Figura 19: Regra de formação dos Álcoois

3.1.8.2 Nomenclatura dos Álcoois

A nomenclatura oficial dos álccois segue as mesmas regras estabelecidas para os hidrocarbonetos; a única diferença está na terminação (**ol**). Exemplos: H_3C - OH (metanol); H_3C - CH_2 - OH (etanol) etc.

No caso de a cadeia do álcool apresentar mais de 3 (três) carbonos, deve-se indicar a posição do **-OH** por número; para isso, numera-se a cadeia carbônica a partir da extremidade mais próxima do **-OH**

Se o álcool apresentar cadeia ramificada, em primeiro lugar, deve-se identificar a cadeia principal, a qual contém a hidroxila **OH**.

3.1.8.3 Aldeídos

Os aldeídos apresentam o grupo carbonila (mostrado na figura 20) na extremidade da cadeia. De acordo com as regras da IUPAC, sua nomenclatura recebe o sufixo **al**.

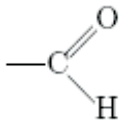

Figura 20: Grupo Carbonila dos Aldeídos

Os aldeídos ramificados e/ou insaturados seguem as regras já vistas para os hidrocarbonetos insaturados e ramificados. Como o grupo funcional está sempre na extremidade, esse carbono sempre será o número 1 (um); portanto, sua posição não precisa ser indicada.

3.2 Teoria Léxico Gerativo

James Pustejovsky descreve, por meio de seus principais trabalhos ([Pustejovsky 1991], [Pustejovsky e Boguraev 1993] e [Pustejovsky 1995]), um formalismo para estruturação de informações semânticas referentes aos itens lexicais, na forma de uma estrutura denominada Qualia Geral, e discute como essa representação lexical pode ser usada para gerar estruturas hierárquicas que associam a informação conceitual ao conceito de um léxico global. O autor aponta para um modelo de composição lexical que incorpora 4 (quatro) aspectos (estruturas) centrais do significado e leva em conta a interação da semântica de diferentes classes de palavras. Tais estruturas são:

1. **Estrutura de Argumentos**: indica como a palavra deve ser mapeada para uma expressão sintática.

2. **Estrutura de Eventos**: identifica o tipo de evento particular para uma palavra ou sentença.

3. **Estrutura Qualia**: contém os atributos essenciais de um objeto, eventos e relações que definem o item lexical.

4. **Estrutura de Herança**: determina a relação entre uma palavra e outros conceitos no léxico.

3.2.1 Estruturas do Modelo

3.2.1.1 Estrutura de Argumentos

A semântica de um item lexical α pode ser definida como uma estrutura composta a partir dos seguintes componentes: A, ε, Q e I, onde A é a estrutura de argumentos, ε é a especificação do tipo de evento, Q provê a ligação dos dois parâmetros anteriores na estrutura qualia e I é um processo de transformação que coloca o item lexical em uma espécie de rede de conceitos, determinando quais informações do item serão hierarquizadas na estrutura léxica global [Pustejovsky 1995].

A estrutura de argumentos especifica o número e o tipo de argumentos lógicos associados normalmente aos verbos, bem como o modo como estes argumentos estão organizados sintaticamente. A estrutura de argumento para uma palavra pode ser vista como uma especificação mínima para a semântica lexical dessa palavra. Somente a estrutura de argumento é inadequada para obter a caracterização semântica do item lexical, mas não deixa de ser um componente necessário.

O autor identifica quatro tipos de argumentos para os itens lexicais e ilustra a distinção entre os mesmos através dos verbos:

- Argumentos verdadeiros (*True Arguments*): parâmetros sintaticamente realizados para um item lexical, ou seja, define todos os parâmetros que são necessariamente expressados na sintaxe.

  Ex.: John levantou tarde (o item lexical levantar (verbo) tem por sujeito João que é um argumento verdadeiro).

- Argumentos padrão (*Default Arguments*): parâmetros de um item lexico que não precisam ser, necessariamente, expressados sintaticamente.

  Ex.: John construiu a casa com tijolos poderia ser omitido, pois, normalmente, a construção ocorre com tijolos).

- Argumentos sombra (*Shadow Arguments*): parâmetros que são semanticamente incorporados ao item lexical. Eles podem ser expressos somente por operações de subtipagem[2] ou especificação do discurso.

  Ex.: Mary untou sua torrada com uma manteiga cara. (o argumento "com uma

[2]Relação de tipos que ocorre a nível semântico. Por exemplo: o item lexico **margarina** é um tipo de **manteiga**.

manteiga cara" pressupõe um argumento sombra de uma marca de manteiga que é conhecida pelo seu alto preço).

- Adjuntos verdadeiros (*True Adjuncts*): parâmetros que modificam a expressão lógica, mas são parte de uma interpretação situacional, não sendo ligados a nenhuma representação semântica particular de itens lexicais. Estes parâmetros incluem as expressões de modificação temporal ou espacial.

  Ex.: Mary rumou para Goiânia na terça-feira.

Os argumentos verdadeiros definem os parâmetros que são necessariamente expressos na sintaxe. Mudanças verbais entre formas polissêmicas de um verbo que resultam em expressões de argumentos verdadeiros devem ser distinguidas. Incluindo mudanças tais como incoativo[3]/causativo [Pustejovsky 1995], por exemplo:

A janela quebrou.

John quebrou **a janela**.

e mudanças de material e produto [Pustejovsky 1995], por exemplo:

1. Mary esculpiu **a boneca** com **madeira**.
2. Mary esculpiu na **madeira uma boneca**.
3. Mary esculpiu **uma boneca** (produto).
4. Mary esculpiu a **madeira** (matéria-prima).

Como a expressão de material é optativa, observe a sentença 3 acima, a qual faz referência ao produto e não à matéria-prima, seu estado como argumento depende do objeto criado. Argumentos optativos, em mudanças como material/produto, são denominados de "argumentos padrão". Eles são necessários para uma lógica bem formada da sentença, mas podem ser omitidos em uma sintaxe superficial. No exemplo 3, madeira (matéria-prima) foi eliminado. Já no exemplo 4, boneca (produto) é quem foi eliminado, sem maiores prejuízos nesses dois exemplos. Por isso eles são denominados "Argumentos Padrão".

[3]Verbo que exprime começo de ação ou de estado.

Para uma expressão cuja sintaxe se apresenta como a do exemplo "a" abaixo, com argumentos verdadeiros A e B e um argumento padrão C, a interpretação pode ser representada, esquematicamente, como mostra o exemplo "b" logo em seguida.

a) A verbo B com C

b) **verbo**'(A,B,C).

Se o argumento padrão não é expresso, então a representação é mostrada no exemplo a seguir.

∃x [verbo'(A,B,x)]

Semelhante aos argumentos padrão, os argumentos sombra referem-se ao conteúdo semântico que não é necessariamente expresso na sintaxe, como o conteúdo semântico incorporado nos verbos "ntar" e "chutar" nas sentenças abaixo.

- Mary untou sua torrada.
- John chutou o muro.

O "argumento oculto" (sombra), na primeira sentença acima é o material que foi espalhado na torrada, enquanto que na segunda fica subtendido o "é", que entra em contato com o muro. Diferente dos argumentos padrão, que são expressados de forma opcional devido a fatores contextuais e de discurso, os argumentos sombra são exprimíveis somente sob condições específicas dentro da própria sentença, isto é, quando os argumentos expressados participam de uma relação de subtipo para o argumento sombra, como as sentenças de exemplo a seguir.

a. Mary untou sua torrada **com margarina**. (com manteiga, seria o argumento sombra).

b. Mary e João dançam **uma valsa**. (uma dança, seria o argumento sombra).

c. John esbarrou em mim **com seu cotovelo artrítico**. (com seu cotovelo, seria o argumento sombra).

Os adjuntos verdadeiros são definidos como forma de complementar os termos de propriedades específicas da classe. Os adjuntos verdadeiros são associados a classes verbais, e não a verbos individuais [Pustejovsky 1995]. Por esta razão, por exemplo, a capacidade

do verbo "dormir" ser modificado pela expressão temporal "a Terça-feira" na primeira sentença abaixo é herdada em virtude da classificação desse verbo como um evento individualizado; da mesma forma é para o verbo "ver" e modificadores de localização tais como "em Boston" na segunda sentença abaixo:

a) John dormiu tarde na Terça-feira.

b) Mary viu o Bill em Boston.

Pustejovsky assume que os argumentos de um item lexical α são representados por uma lista ARGSTR onde o tipo de argumento é diretamente codificado, como mostrado na figura 21, onde D-ARG contém os argumentos padrão, S-ARG contém os argumentos sombra e ARG1, ..., ARGn podem ser tanto argumentos verdadeiros como adjuntos verdadeiros.

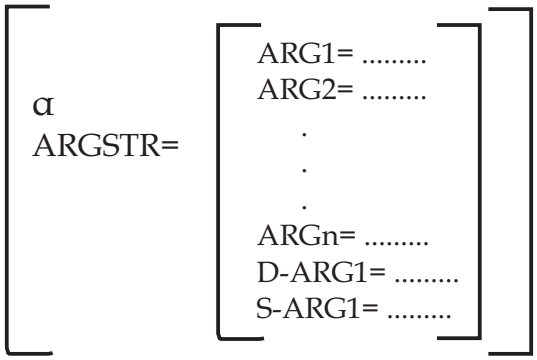

Figura 21: Formato da Estrutura de Argumentos.

As representações parciais da semântica lexical dos verbos "onstruir", "Uuntar" e "chutar" são ilustradas pelas estruturas de argumentos das figuras 22, 23 e 24, respectivamente. Note que tais representações são baseadas nos argumentos dos verbos.

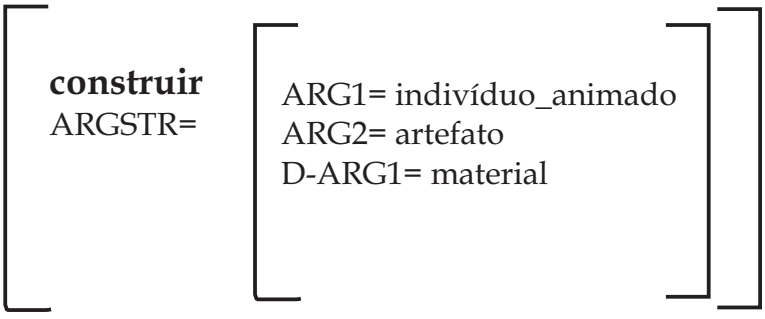

Figura 22: Estrutura de Argumentos para o item lexical *construir*.

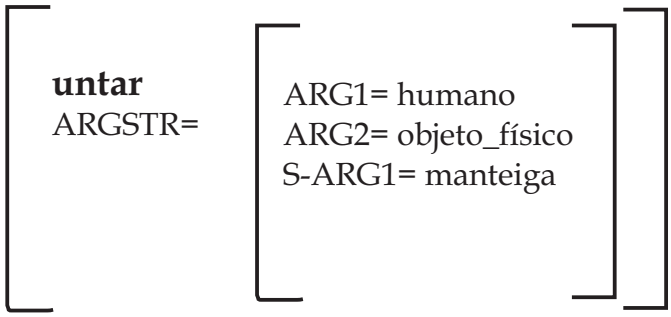

Figura 23: Estrutura de Argumentos para o item lexical *untar*.

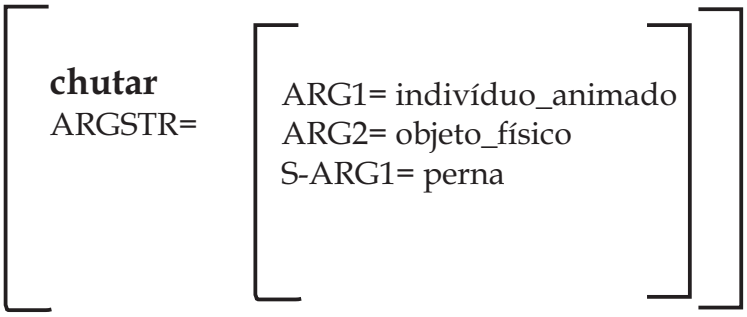

Figura 24: Estrutura de Argumentos para o item lexical *chutar*.

3.2.1.2 Estrutura de Eventos

A estrutura de eventos é a estrutura que define o tipo de evento associado a um item lexical verbal e a uma frase.

Um evento associado a um verbo em uma semântica baseada em eventos é listado como um único argumento junto com o parâmetro lógico definido por um predicado ou relação particular que possibilite a semântica parcial de um verbo. Por exemplo, uma representação lexical para o verbo "construir" unindo aspectos das análises tanto de Davison [Davison 1967] quanto de Parson [Parsons 1990] poderia ser como:

$\lambda y \lambda x \lambda e$[construir(e,x,y) $\wedge$ Θ1(e,x) $\wedge$ Θ2(e,y)]

Onde, Θ1 e Θ2 seriam os subeventos "construindo" e "construído", respectivamente, do verbo "construir", x é aquele que constrói e y é algo construído.

A representação lexical acima, assume uma visão atômica na estrutura de eventos, onde aspectos internos (como argumentos do verbo) dos subeventos não são descritos. Assim, é necessário definir um significado para a representação de estruturas subeventuais associadas a itens lexicais, expressando a relação entre eventos e argumentos de um verbo. Pustejovsky em 1995 [Pustejovsky 1995], esboçou um mecanismo chamado Ligação de Parâmetros de Orthognais (*Orthogonal Parameter Binding*) que permite ligar em uma

expressão uma lista de parâmetros independentes, isto é, estrutura de argumento e estrutura de evento. Dado uma lista de argumentos e uma estrutura de evento representada como uma lista de variáveis de evento:

[ARGSTR = ARG1, ARG2, ..., ARGn]

[EVENTSTR = EVENT1, EVENT2, ..., EVENTn]

A semântica do verbo pode ser vista como sendo centralmente definida pela estrutura Qualia Geral (especificada na próxima subseção), mas restringida pelo tipo de informação de duas listas de parâmetros. O predicado na Qualia Geral refere diretamente aos parâmetros:

[QUALIA GERAL = [. . .[Q_i = PRED(EVENT$_j$, ARG$_k$)] . . .]

Assume-se que os eventos podem ser classificados em três classes: PROCESSOS, ESTADOS e TRANSIÇÃO. Além disso, assume-se uma estrutura subevento para essas classes de eventos.

Num evento semântico é necessário representar a relação entre este evento e seus próprios subeventos. Isso é feito através da "estrutura de eventos estendida", que é uma tupla [Pustejovsky 1995]:

$< E, \leq, <, ^\circ, \subseteq, * >$

onde:

E: conjunto de eventos;

$\leq$: ordem parcial *parte_de*;

$<$: ordem parcial estrita;

$^\circ$: justaposição (overlap);

$\subseteq$: inclusão;

*: "cabeça" de um evento.

Uma estrutura de eventos com subeventos estruturados, abaixo (figura 25), é ilustrada como uma estrutura em árvore em termos de relações de um "ordenamento exaustivo parte_de".

Esta estrutura de eventos é utilizada por verbos como "acompanhar", que envolve dois subeventos que ocorrem, de certa forma, ao mesmo tempo (um processo, acompanhar, e um estado, acompanhando). Diante disso, existem dois aspectos da estrutura de eventos

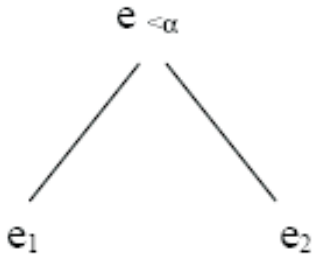

Figura 25: Uma Árvore de Eventos com Subeventos Estruturados.

que necessitam ser representados em uma estrutura lexical: os eventos específicos e seus tipos, bem como a restrição ordenada destes eventos. Isto é esquematicamente ilustrado na figura 26.

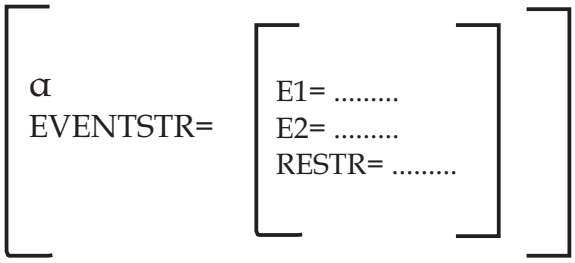

Figura 26: Formato da Estrutura de Eventos.

O verbo construir, por exemplo, é tipicamente analisado envolvendo um processo de desenvolvimento (*construindo*) e um estado resultante (*construído*), ordenados pela relação "ordenamento exaustivo parte_de", $<_\alpha$, como ilustrado na figura 27. Onde essa relação de ordenação parte do princípio temporal, ou seja, *construindo* precede *construído*, sendo que cada um desse processo (evento) é uma parte lógica do *construir*.

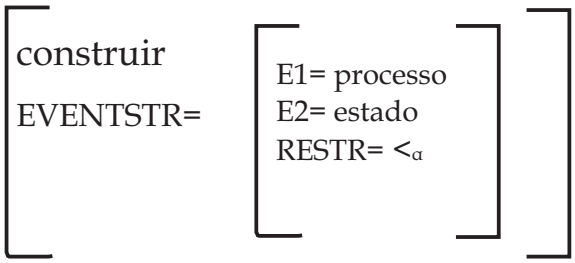

Figura 27: Estrutura de eventos do verbo *construir*.

Já o verbo *acompanhar*, cuja estrutura de evento é mostrada na figura 28, permite qualquer evento objetivo, como TRANSIÇÕES ou PROCESSOS. Diferentemente do verbo construir, que obriga que os tipos de seus subeventos sejam PROCESSO e ESTADO.

A informação estruturada é necessária mas não suficiente para capturar distinções lexicais feitas sistematicamente pelas linguagens, com respeito a importância relativa dos

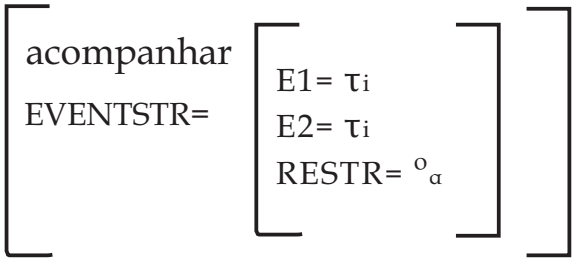

Figura 28: Estrutura de eventos do verbo *acompanhar*.

subeventos de um evento. Nota-se que a informação associada a um evento originária de um verbo pode ser mais rica do que uma estrutura de "seqüência de eventos" codificada como visto acima. Uma instância de importância para um evento é fornecida por um marcador (*HEAD marker*), simbolizado por e*. A função de um marcador em uma representação sintática é indicar a importância de um determinado evento. Um marcador é definido como o subevento mais importante na estrutura de eventos, que contribui para o "foco" da interpretação. Pode-se visualizar * como uma relação entre eventos, * (e_i, e_j) onde e_i é a "cabeça" de e_j, $e_i \leq e_j$.

Logo abaixo, uma estrutura de eventos adicionada pelo *headedness* (propriedade dos subeventos), onde se discrimina sua importância relativa, sendo a sua estrutra ilustrada na figura 29.

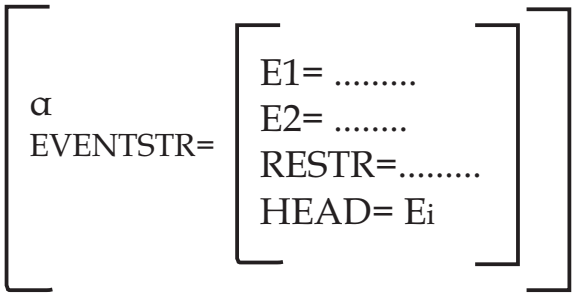

Figura 29: Estrutura de eventos com *Headedness*.

Um exemplo de uma estrutura de evento para o item léxico "construir" utilizando a propriedade *headedness* é mostrado na figura 30.

A estrutura mostrada na figura 30 representa verbos de realização (E1 e E2), onde o evento inicial é encabeçado (*headed*), focando a ação de "construindo" sobre o estado "contruído".

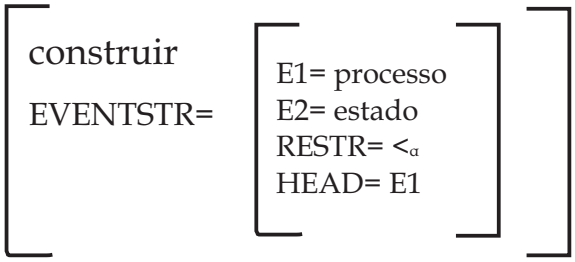

Figura 30: Estrutura de eventos com *Headedness* para o item lexical *construir*.

3.2.1.3 Estrutura Qualia

A estrutura Qualia é um sistema de relações que caracteriza a semântica de itens lexicais. É inspirada na interpretação de Moravcsik sobre "modos de explanação (*aitiae*)" de Aristóteles. Os elementos que constroem a estrutura Qualia incluem noções de constituintes, espaço, figura, superfície, manufatura etc, e estão organizados em quatro aspectos do significado da palavra. Cada aspecto se refere a um "papel" ou função associada ao significado. Tais papéis são denominados, respectivamente, papel constituinte, papel formal, papel télico e papel agente.

- Papel Constituinte ("Const"): material, peso, partes e elementos componenciais. Relaciona um objeto e suas partes constituintes.
- Papel Formal ("Form"): orientação, magnitude, forma, dimensionalidade, cor, posição. Distingue os objetos dentro de um domínio maior.
- Papel Télico ("Telic"): objetivo de um agente na execução da ação, que especifica atividades. Este papel especifica objetivo e função.
- Papel Agente ("Agent"): criador, artefato, cadeia causal. Especifica fatores envolvidos na origem ou construção de um objeto.

Um exemplo de estruturação do substantivo *carro* na forma de estrutura Qualia é ilustrado na figura 31. Conforme expresso na figura 31, o carro x tem como partes constituintes a carroceria, o motor e as rodas; tem as características de um objeto físico, tem por função ser dirigido e tem por origem ser um objeto construído.

No exemplo da figura 31 os papéis da estrutura Qualia do carro não são somente listas de predicados. Os valores dos papéis da Qualia apresentam-se como expressões com tipos bem definidos e estruturas relacionais. Esta representação é utilizada por Pustejovsky para proporcionar um tratamento mais polimórfico da semântica e permitir

carro(x)
CONSTITUINTE: {carroceria, motor, rodas}
FORMAL: x: objeto_físico
TÉLICO: dirigir (e, y, x)
AGENTE: construir (e, z, x)

Figura 31: Estrutura Qualia para o substantivo *Carro*.

interpretações distintas de itens lexicais em contextos particulares. A estrutura Qualia contém expressões que relacionam os tipos definidos nas estruturas de argumentos e de eventos, e é equivalente a uma expressão λ (expressão LAMBDA [Bach 1989]). Como exemplo, a expressão λ equivalente ao exemplo da figura 31 é apresentada abaixo.

λx[carro(x) $\wedge$ ... TÉLICO = dirigir(e1, y, x) $\wedge$ AGENTE = construir(e2,z,x)]

Observa-se que as estruturas Qualia são definidas sobre outras estruturas Qualia formando, assim, uma rede de hierarquias por papéis. Assim, estas redes são mais poderosas do que as hierarquias IS_A, pois as generalizações e especificações são dadas por diferentes papéis, e não somente pela função de pertinência IS_A. Uma ilustração do poder desta rede aparece na figura 32.

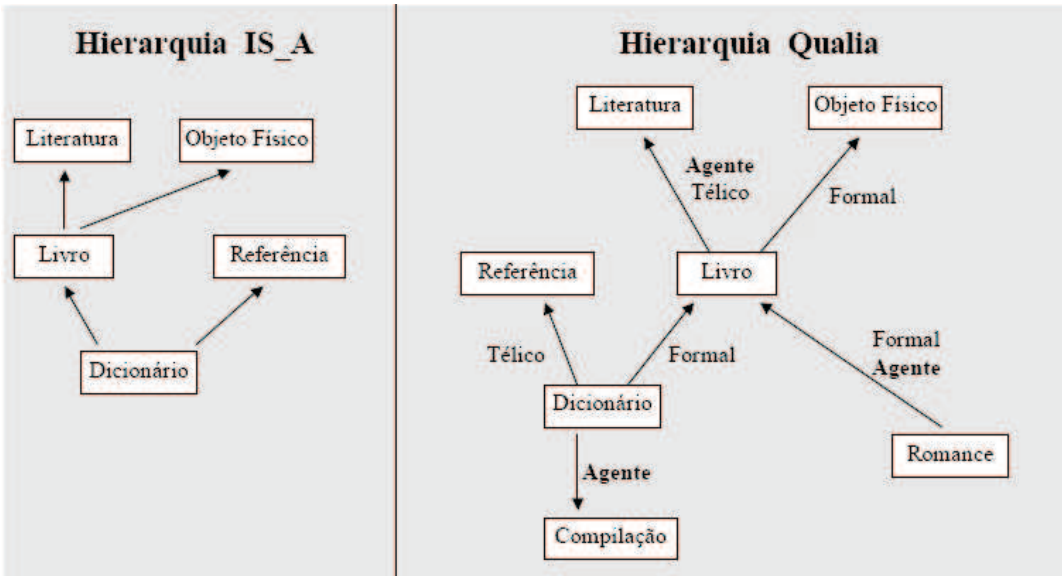

Figura 32: Hierarquias IS_A X Hierarquias Qualia.

A hierarquia IS_A, ilustrada na figura 32, apresenta um problema que normalmente ocorre neste tipo de estrutura: um dicionário é representado como um livro, que por sua vez constitui uma obra de literatura. Conforme definido, livros normalmente constituem

literatura mas, ao associar um dicionário à categoria livro, ocorre um erro semântico, pois dicionários não constituem literatura. Na estrutura Qualia, este problema é facilmente solucionado: dicionários têm a forma de um livro (papel Formal) herdando sua forma "objeto físico". Contudo, são provenientes de uma compilação, e não literatura, o que ocorre através da mudança do papel Agente de livro. Da mesma forma, têm como objetivo (função) "Referência", e não "Literatura", o que é indicado pela mudança do papel Télico do livro.

3.2.1.4 Estrutura de Herança

James Pustejovsky no seu artigo "*The Generative Lexicon*" [Pustejovsky 1991] assume que há dois mecanismos para representar as relações conceituais no léxico: Herança Fixada e Herança Projetiva (*projective*). A primeira inclui os métodos de herança tradicionais, assumidos na Inteligência Artificial (IA) e na pesquisa léxica, isto é, uma rede de relações, que é percorrida para descobrir a existência de conceitos relacionados e associados a um determinado item lexical. Além dessa representação estática, o autor introduz outro mecanismo para o conhecimento léxico estruturado, a "herança projetiva", que opera gerativamente da estrutura qualia de um item léxico para criar uma estrutura relacional para categorias *ad hoc*[4]. Ambas são necessárias para projetar a representação semântica de um item léxico individual dentro de uma frase a nível de interpretação. O texto desse artigo limita a descrever a herança projetiva e a noção de "graus de prototipação" de predicação. Putesjovsky argumenta que tais graus de relações salientes ou coerentes podem ser explicados em termos estruturais examinando uma rede de itens léxicos relacionados. O autor ilustra a distinção entre estes mecanismos (herança projetiva e grau de prototipação) considerando as duas orações abaixo, assim como suas relativas prototipações.

- *The prisoner escaped last night*[5].
- *The prisoner ate dinner last night*[6].

De acordo com o autor, a predicação da primeira oração é "mais ajustada" ou mais prototipada do que a da segunda oração. A que se atribuiria tal diferença? Intuitivamente, associa-se *prisoner* (prisioneiro) com um evento *escaping* (escapando) mais fortemente do que a um evento *eating* (comendo). Esta não é uma informação que vem de uma

[4]Coleções de um tipo particular [Barsalou 1983].
[5]O prisioneiro escapou ontem a noite.
[6]O prisioneiro jantou ontem a noite.

estrutura de herança fixada, mas é melhor assumida como um conhecimento do sentido comum. Segundo Pustejovsky, tais distinções podem ser capturadas dentro de uma teoria de semântica léxica por meio da geração de categorias *ad hoc*. O autor utiliza algumas definições para explanar a estrutura de herança projetiva, além de usar duas definições (citadas em seguida), as quais definem a estrutura de herança fixada.

Definição 1:

A seqüência <Q1, P1,..., PN> é um caminho de herança que pode ser lido como uma conjunção de pares ordenados $<x_1, y_i> \mid 1 \leq i \leq n$. Além disso, o espaço de conclusão é definido como o conjunto de conceitos que se encontra num caminho de herança.

Definição 2:

O espaço de conclusão de um conjunto de seqüências Φ é o conjunto de todos os pares <Q,P> tais que uma seqüência <Q,...,P> aparece em Φ.

Destas duas definições acima, define-se a tradicional relação IS_A, onde os elementos dos pares acima se relacionam pelo operador de generalização, $\leq_G$, bem como relações de hipernímia, hiponímia e troponímia[7]. Supondo que, em adição destas estruturas relacionais fixadas, a semântica permite dinamicamente criar conceitos arbitrários através da aplicação de certas transformações para o significado lexical. Por exemplo, para algum predicado Q (por exemplo, o valor de um papel qualia), pode-se gerar sua oposição, $\neg$Q. Relacionando estes dois predicados temporariamente para gerar eventos de transição arbitrários para esta oposição.

Por exemplo:

1. $\neg Q(x) \leq Q(x)$
2. $Q(x) \leq \neg Q(x)$
3. $Q(x) \leq Q(x)$
4. $\neg Q(x) \leq \neg Q(x)$

Similarmente, pela operação acima, outros valores de papéis qualia podem gerar conceitos relacionados semanticamente. Segundo o autor, tal operação é chamada de "transformação projetiva", definida abaixo:

Definição 3:

[7]Relata verbos por meio de relações

Uma transformação projetiva, π, aplicada a um predicado Q1 gera um predicado Q2, tal que π(Q1) = Q2, onde Q2 $\notin \Phi$ (conjunto de seqüência). O conjunto de transformações inclui: $\neg$ (negação), $\leq$ (precedência temporal), $\geq$ (sucessão temporal), = (equivalência temporal) e *act*, um operador que adiciona uma ação ao argumento. Intuitivamente, o espaço de conceitos percorrido pela aplicação de tais operadores será relacionado nas expressões da vizinhança do item léxico original. Este espaço pode ser caracterizado pelas duas definições seguintes:

Definição 4:

Uma série de aplicações de transformações, π1,..., πN, gera uma seqüência de predicados <Q1,...,QN>, chamada de expansão projetiva de Q1, representada por P(Q1).

Definição 5:

O espaço de conclusão projetiva, P(Φ_R), é o conjunto de expansão projetiva gerado por todos os elementos do espaço de conclusão Φ no papel R do predicado Q: como: P(Φ_R) = <P(Q_1), P(Q_N)> | <Q_1,...,Q_N> $\in \Phi_R$. Para esta representação resultante, pode-se gerar uma estrutura relacional que pode ser considerada o conjunto de categorias ad hoc e relações associadas com um item lexical. Utilizando as definições acima, retoma-se o exemplo das frases do início desta subseção. O autor assume que o substantivo *prisoner* (prisioneiro) tem a seguite estrutura qualia:

```
Prisoner(*x*)
    Form: human(*x*)
    Telic: [confine(y,*x*) & location(*x*, prison)]
```

Além do mais, ele assume a seguinte representação lexical para *scape*.

λxλe$^T\exists$e^P,e^S[*escape*(e^T) $\wedge$ *act* (e^P) $\wedge$ *confined*(e^P) $\wedge$ *agent*(e^P, x) $\wedge$ $\neg$*confined*(e^S) $\wedge$ *object*(e^P, x)]

Estruturas de herança são definidas para cada papel qualia de um elemento. Por exemplo, o papel Telic do *prisoner* (prisioneiro) podendo ser generalizado para o conceito de *ser confinado - confine(y,x)*:

$\leq_G$: [*confine*(y, x) $\wedge$ *loc*(x, *prison*)] $\Rightarrow$ *confine*(y,x)

Deste conceito pode-se aplicar a operação de negação, gerando o predicado de oposição de *not-confined* e *confined*:

$\neg$: $\exists$E1[$\neg$*confine*(E1, y, x)]

$\exists$E2[*confine*(E2, y, x)]

Para isto, aplicando as duas operações temporais, $\leq$ e $\geq$, gerando dois estados: *free before capture*(livre antes da captura) e *free after capture*(livre depois da captura).

$\leq$: E1 $\leq$ E2 = T1

$\leq$: E2 $\leq$ E1 = T2

Finalmente, para estes conceitos, se aplicar o operador *act*, variando quem é o responsável pelo evento de transição resultante, gerando os conceitos: *turn in*(volta), *capture*(captura), *escape*(escapa), e *release*(soltura).

act: act(x, T1) = "turn in"

act: act(y, T1) = "capture"

act: act(x, T2) = "escape"

act: act(y, T2) = "release"

Essas relações constituem o espaço de conclusão projetiva para o papel Telic do *prisoner* (prisioneiro) relativo à aplicação das transformações mencionadas acima. De acordo com o autor, a diferença entre as duas frases (citadas no exemplo acima) em suas prototipações (ou as condições relevantes nos seus predicados) ficou mais clara logo depois que todas as definições foram aplicadas (teoria de herança lexical). O predicado *eat* (come) está fora do espaço de conceitos gerados na semântica do substantivo the *prisoner* (o prisioneiro); *escape* (escapa), entretanto, está dentro do espaço de conclusão projetiva para o papel Telic do *prisoner* (prisioneiro). Segundo Pustejovsky, todas as definições vistas até aqui tem como objetivo gerar conceitos relacionados ao valor de cada papel Qualia.

$escape \in \mathrm{P}(\Phi_T(prisoner))$

$eat \notin \mathrm{P}(\Phi_T\ (prisoner))$

De acordo com o autor, tal procedimento pode ser usado como uma métrica de avaliar a "proximidade" de predicados. Nos exemplos acima, a diferença na semântica pode agora ser vista como uma distinção estrutural entre as representações semânticas para os elementos na frase. O que o autor não focalizou, entretanto, é como as informações da herança fixada de um item léxico são formalmente derivadas durante a composição.

3.2.1.5 Interação entre os Níveis Semânticos

Segundo o autor, para a construção de um léxico semântico é necessária uma interação entre os níveis semânticos (argumento, evento e Qualia). Pustejovsky utiliza o termo "Qualia" para designar tanto a sua estrutura real, composta pelos seus papeis Qualia, quanto para uma representação que envolve a estrutura de Argumento, Evento e a Qualia, ou seja, a interação dessas estruturas. Devido a isso, esse trabalho denominou essa estrutura de interação de "Estrutura Qualia Geral", para que haja uma melhor identificação das estruturas envolvidas.

Pustejovsky [Pustejovsky 1995] ilustra o modo como estes níveis formam uma representação semântica integrada considerando, novamente, a semântica do verbo "construir". Para o verbo construir existem dois argumentos verdadeiros e um argumento padrão. Além disso, nota-se o envolvimento de dois subeventos: um processo e um estado resultante. Estes estão ilustrados na Qualia, representada na figura 33.

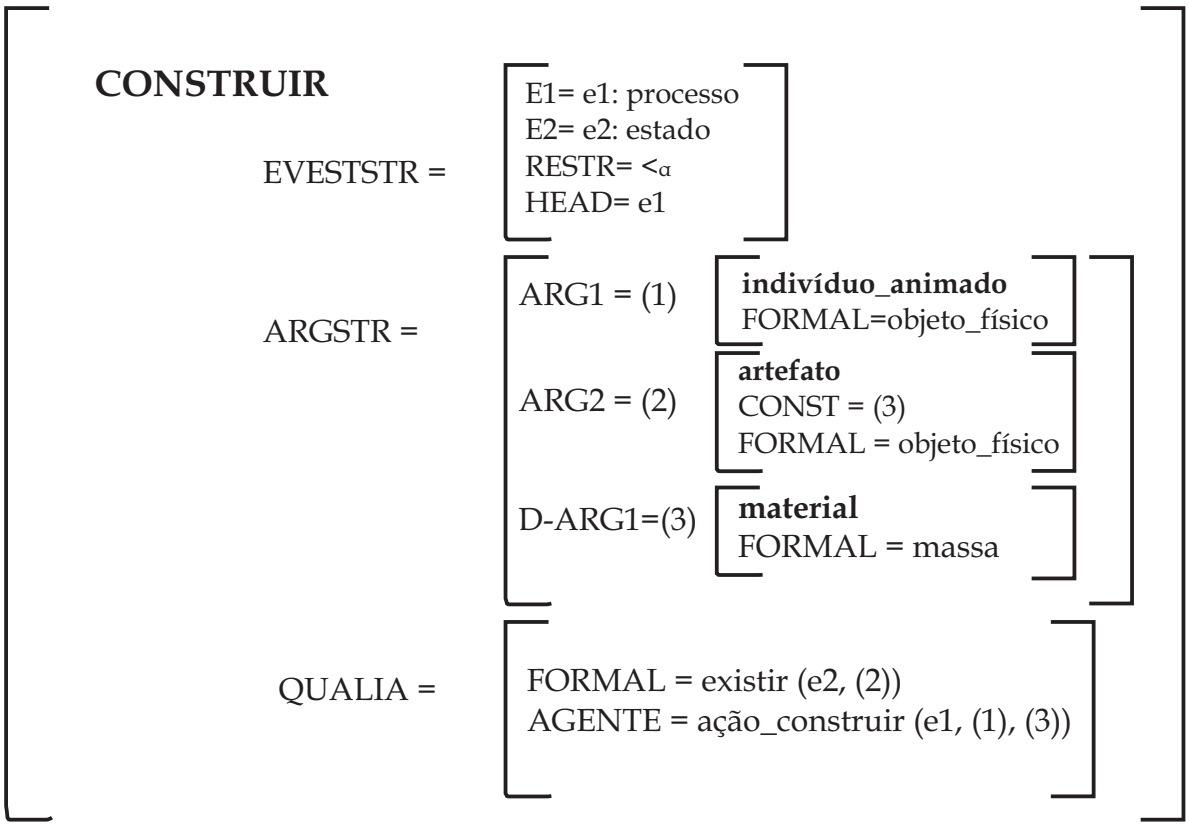

Figura 33: Estrutura Qualia Geral de interação entre os níveis semânticos para o verbo *construir*.

O processo é identificado como um papel Agente envolvendo um objeto sintático, ARG1, e um argumento padrão, D-ARG1, que está relacionado a um objeto lógico pelo papel Constituinte de ARG2. O papel Formal expressa o estado resultante disto como

um objeto ARG2.

3.2.2 Paradigmas Conceituais Lexicais

Pustejovsky acredita que, para representar os diferentes comportamentos semânticos dos substantivos polissêmicos, seja necessária uma interação entre a estrutura de argumentos e a estrutura Qualia. Nas sentenças a seguir, palavras como janela ou porta apresentam duas interpretações distintas:

- João passou pela janela.
- Maria quebrou a janela.
- Maria pintou a porta.
- João passou pela porta.

Cada substantivo, nas frases acima, tem um dos dois significados: um objeto físico ou uma denotação de abertura. Pustejovsky caracteriza esta situação como um caso de "dupla figuração" dos substantivos, onde todos os argumentos são logicamente parte do significado desse substantivo. Assim, a habilidade de um item lexical reunir múltiplos significados é o que autor denomina Paradigma Conceitual Lexical (*Lexical Conceptual Paradigm* ou lcp). Substantivos como jornal, no exemplo abaixo, aparecem em inúmeros contextos com significados diferentes (uma organização, objeto físico ou informação):

- O jornal atacou a presidente. (Organização)
- Maria derramou café no jornal. (Objeto Físico)
- João leu horríveis notícias no jornal. (Informação)

Um paradigma conceitual lexical provê o significado de um item lexical como uma meta-entrada. O tipo construtor, *lcp*, cria um tipo complexo para um termo α possui sentidos τ_1 e τ_2, dano a seguinte regra:

$$\frac{\alpha : \tau_1 \qquad \alpha : \tau_2}{lcp(\alpha) : \tau_1 \ . \ \tau_2}$$

Um lcp resultante é representado por um tipo agrupador, construído de dois tipos básicos ($\tau 1$ e $\tau 2$) e um tipo composto, mostrado abaixo:

lcp = { τl. τ2, τ1, τ2 }

Assim, caso τl e τ2 sejam os significados do substantivo "porta", representados respectivamente pelos tipos *objeto_físico* e *abertura*, então o lcp resultante sobre estes tipos poderia ser:

objeto_físico.abertura_lcp = {objeto_físico.abertura, objeto_físico, abertura}

De acordo com Pustejovsky, entre os tipos compostos encontrados na linguagem natural, podemos verificar as seguintes combinações:

a) objeto_físico.informação: ex.: livros, fitas;

b) evento.evento: ex.: construção, examinação;

c) evento.questão: ex.: pesquisa;

d) evento.alimento: ex.: almoço, jantar;

e) evento.humano: ex.: nomeação;

Em um lcp, todos os tipos definidos são válidos nas expressões contidas na estrutura Qualia, como será visto na próxima subseção.

3.2.3 Relacionamento das Estruturas Componentes do Modelo com a Semântica das Palavras

O papel Formal distingue um objeto, em um universo de objetos. Existem duas possíveis associações com o papel Formal na Qualia:

Tipagem simples: o valor do papel Formal é idêntico ao do tipo do argumento.

Tipagem complexa: o valor do papel Formal define a relação entre os argumentos de diferentes tipos.

No primeiro caso incluem-se os itens lexicais simples, que não apresentam polissemia. Neste caso, o papel Formal representa uma restrição de tipo definida na estrutura de argumentos. Assim, uma estrutura Qualia para este tipo de item lexical tem a forma esquemática representada na figura 34.

Para itens lexicais com tipos complexos, ou seja, itens lexicais em que se verifica uma polissemia como livro e jornal, o valor do papel Formal da Qualia define como um

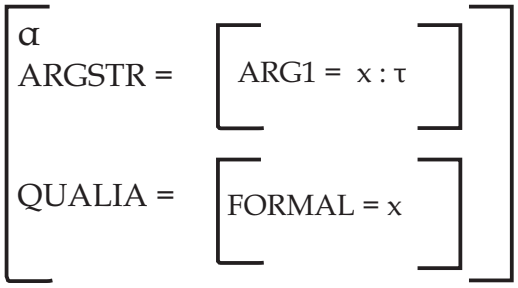

Figura 34: Estrutura Qualia Geral para o papel Formal de Tipagem Simples.

argumento se relaciona com os outros argumentos. A forma esquemática da figura 35 representa a estrutura Qualia para estes substantivos.

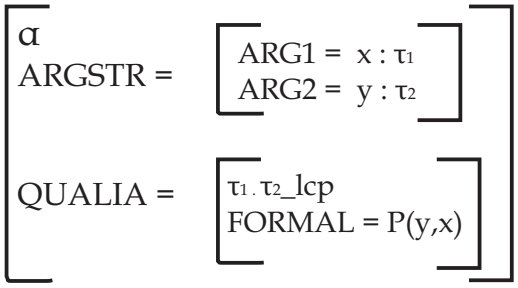

Figura 35: Estrutura Qualia Geral para o papel Formal de Tipagem Complexa.

O conhecimento do surgimento de um objeto é codificado no papel Agente de um item lexical. A maneira como algo é criado serve para explicar distinções tais como os tipos naturais e os tipos artefatos.

Se a forma lexical é um substantivo, o papel Agente é representado como um evento, onde o ser do objeto definido é ligado tipicamente ao segundo argumento da relação, a estrutura qualia geral é esquematizada de acordo com a figura 36.

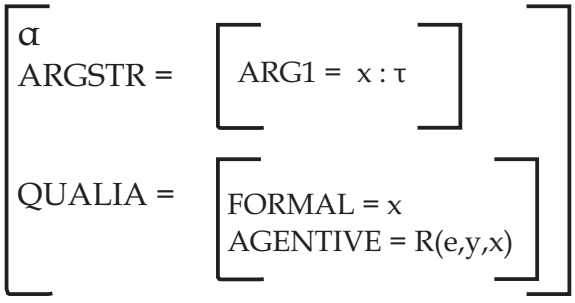

Figura 36: Estrutura Qualia Geral envolvendo o papel Formal e o Agente.

A representação na figura 36 corresponde as semânticas de artefatos. Por exemplo, objetos tais como biscoitos, bolos e pães são tipicamente assados, ou seja, esse processo de assar é uma atividade criativa, enquanto relativos objetos tais como “batatas”, “cenouras”,

e outros tipos naturais são simplesmente uma troca de estado (estado de não cozido para cozido).

Para tipos compostos, o papel Agente pode referenciar o argumento complexo diretamente, quando necessário, ou pode referenciar somente um dos argumentos. Para o exemplo "livro", o papel Agente seria *escrever(e, w, informação.objeto_físico)*. O esquema do papel Agente para este tipo de item pode ser visto na figura 37.

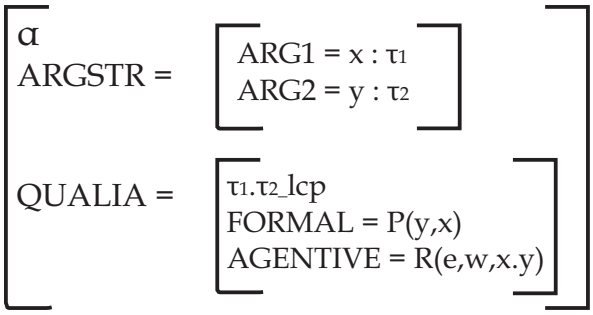

Figura 37: Estrutura Qualia Geral mostrando a aplicação do papel Agente para tipos compostos.

O papel Constituinte não se refere somente às partes constituintes ou material constituinte de um objeto, mas define, para este objeto, uma relação parte-de de uma estrutura maior (da qual este objeto é parte), se existir. A relação parte-de permite, assim, duas abstrações: uma relação que permite referenciar do que algo é constituído, bem como o que este algo constitui. A figura 38 apresenta, para o item lexical *mão*, o papel Constituinte, o qual define "mão" como parte constituinte do corpo humano.

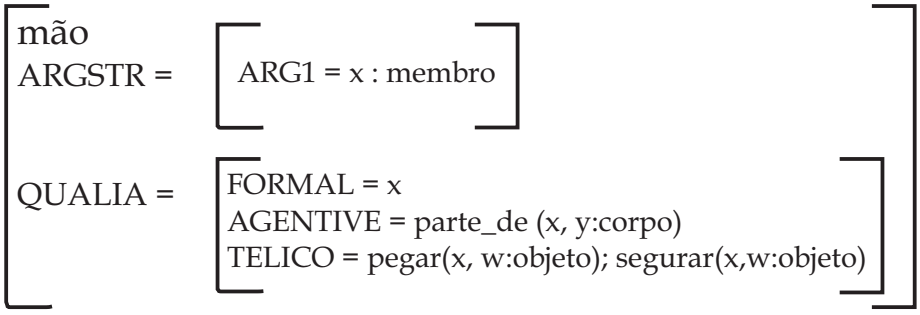

Figura 38: Estrutura Qualia Geral para o item lexical *mão*.

O papel Télico define qual o propósito ou função de um item lexical. Esta função pode ser direta, algo que age diretamente, como a cerveja, que tem como papel télico *beber(e, y, cerveja)*, onde o líquido é bebido por alguém; ou proposital, usada para realizar uma atividade particular, como a faca, que tem como papel télico *cortar(e, faca, y)*, onde algo é cortado com a faca.

A estrutura Qualia Geral para função direta pode ser visualizada na figura 39.

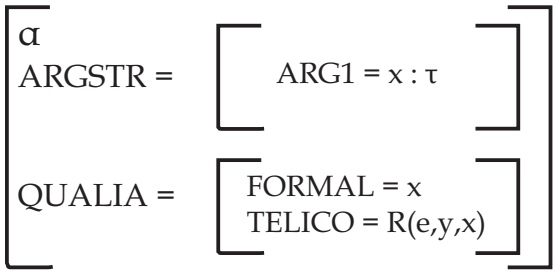

Figura 39: Estrutura Qualia Geral para o papel Telico Direto.

Como exemplo, dessa estrutura Qualia Geral para a função direta, considere a estrutura Qualia Geral sobre o papel Telico para o item léxico "cerveja" ilustrada na figura 40:

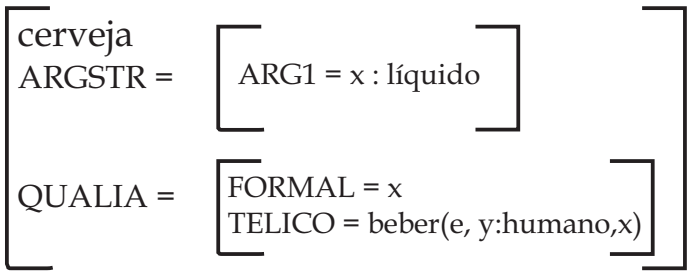

Figura 40: Estrutura Qualia Geral para o item léxico *cerveja*.

Já a estrutura Qualia Geral para função proposital pode ser visualizada na figura 41.

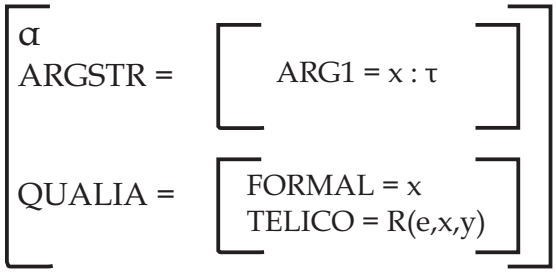

Figura 41: Estrutura Qualia para o papel Télico Proposital.

Um exemplo do secundo tipo do uso Telico, proposital, é encontrado com objetos que são usados na execução de atividades, tais como ferramentas, como ilustrado na figura 42 para a ferramenta ou item léxico "faca".

3.2.4 Aspectos Gerativos

Segundo Pustejovsky [Pustejovsky 1995], mecanismos gerativos são mecanismos que conectam os níveis de representação semântica (estrutura de argumento de tipo com-

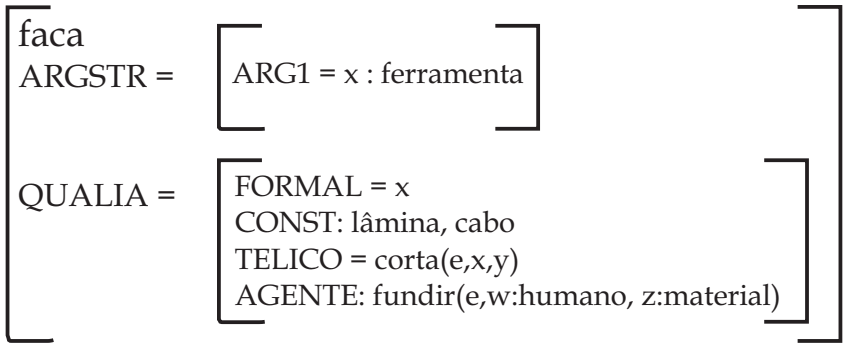

Figura 42: Estrutura Qualia Geral para o item léxico *faca*.

plexo, estrutura de evento e estrutura qualia), provendo a interpretação composicional de palavras em diferentes contextos. A mais importante destas estratégias é uma transformação semântica, chamada de coerção de tipos.

3.2.4.1 Coerção de Tipo

A coerção de tipos é uma operação semântica que converte um argumento para o tipo esperado por uma função. Isto é, coerção de tipos é uma mudança do tipo de uma expressão dentro de uma sentença. Como exemplo, um tipo de expressão padrão <ser, objeto> para o verbo "querer", e outros tipos possíveis como <ser, ação>. Ao analisar a sentença "João quer nadar", o tipo de expressão padrão, ou seja, o tipo do argumento dentro de uma estrutura de representação do significado de "querer" (<ser, ação>), muda para <ser, ação>. Então ocorre uma coerção de tipo.

Para ilustrar informalmente os efeitos deste mecanismo, Pustejovsky utiliza o verbo querer, apresentado nas sentenças abaixo.

1. Maria quer que João viva.
2. Maria quer viver.
3. Maria quer uma cerveja.

Destas sentenças, dois pontos a ressaltar: os diferentes complementos do verbo "querer" e as diferentes interpretações que aparecem para o complemento, que parecem requerer uma enumeração de significados em um contexto. Assim, se a forma sintática que aparece no complemento do verbo é igual ao tipo requerido pelo verbo, então a estrutura resultante é bem formada, isto é, existe uma representação semântica bem formada. Caso

contrário, esta estrutura sofre uma coerção pelo verbo, igualando-se ao tipo requerido pelas restrições do tipo do verbo.

Desse modo, em vez de assumir uma nova entrada lexical para um determinado verbo em diferentes meios sintáticos ou em interpretações distintas, o autor assume que a semântica de um verbo deve ser expandida com elementos na sua composição, acompanhados de operações gerativas (a base de seu léxico gerativo). Entre estas operações, tem-se a coerção de subtipos e a coerção de complementos verdadeiros.

Coerção de Subtipos

Para demonstrar a propriedade formal de coerção de tipos é necessário, primeiramente, examinar o caso mais simples de coerção, denominado coerção de subtipos. Observe as senteças abaixo, onde tanto o sujeito como o objeto (PNs - predicados nominais) são subtipos de uma especificação de classes para os argumentos do verbo.

- **Maria** dirigiu **um Honda**.
- **João** leu **a Veja**.

Embora as sentenças acima, são tratadas de maneira trivial pelo ponto de vista da sintaxe, para uma semântica com tipos, deve-se estabelecer uma relação entre o tipo denotado pelo PN em cada uma das posições de seus argumentos e o tipo que é formalmente selecionado pelos verbos "dirigir" e "ler". A relação expressa entre esses tipos é uma relação de subtipo. Assim, é necessário assegurar que, se uma função seleciona o tipo $\tau 1$ e posteriormente ocorre uma forma $\tau 2$, onde $\tau 2$ é um subtipo de $\tau 1$ ($\tau 2 \leq \tau 1$), este deve também ser aceito pela função como um argumento legítimo. Por exemplo, assumindo a representação lexical para o substantivo carro na figura 43, então, como "Honda" (definida na figura 44) é um subtipo de carro, estabelece-se a seguinte relação: Honda $\leq$ carro $\leq$ veículo. Com outro mecanismo de herança lexical, o valor mais específico para o papel AGENTE na estrutura da figura 44 substitui o valor mais geral associado com o artefato "carro", enquanto ainda herdando os valores para outra qualia [Pustejovsky 1995]. Observe que os valores de AGENTE e TÉLICO são herdados de "carro", com uma diferença que é o tipo específico do AGENTE para "Honda", o qual é localmente definido (Honda_Co).

Assume-se que o tipo selecionado pelo verbo na primeira sentença (Maria dirigiu um Honda) é veículo, como ilustrado na figura 45 (onde veículo é um argumento de dirigir) - esta é uma semântica parcial do verbo, voltada ao exemplo - então, os requisitos de seleção do verbo podem ser satisfeitos somente nos casos em que existir a relação de subtipo, que

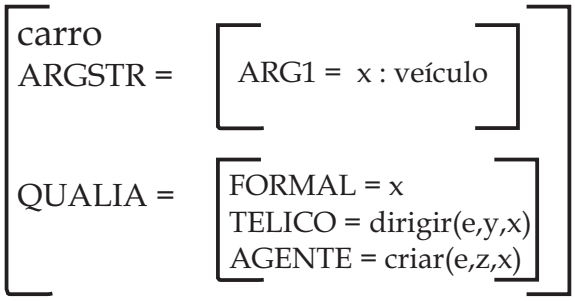

Figura 43: Estrutura Qualia Geral para o item lexical *carro*.

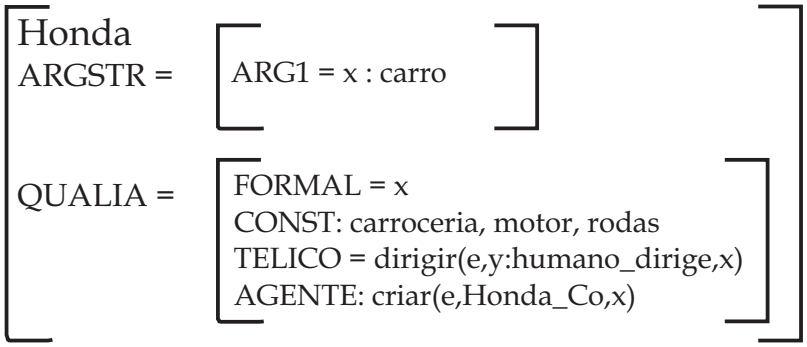

Figura 44: Estrutura Qualia Geral para o item lexical *Honda*.

associa formalmente o tipo do objeto atual com o tipo lexicalmente especificado (relaciona veículo com Honda).

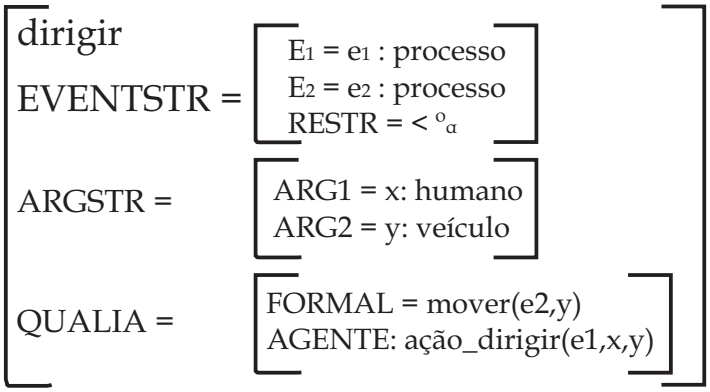

Figura 45: Estrutura Qualia Geral para o item lexical *dirigir*.

Assim, verifica-se, por exemplo, que o verbo "dirigir" tem como argumento "veículo" e "Honda" é um subtipo de carro, que é um "veículo", então a função que aceita veículo como parâmetro deve aceitar também "Honda", através de uma de coerção de subtipo do complemento do verbo.

A segunda sentença do exemplo acima (João leu a Veja) é tratada da mesma forma que a primeira sentença, ou seja, Veja $\leq$ revista $\leq$ texto. Define uma relação entre o tipo

selecionado pelo verbo "ler" e o indivíduo (João).

Coerção de Complementos

Pustejovsky denomina coerção de complementos à coerção de tipos que envolve uma mudança estrita de um tipo para outro tipo específico, autorizada por uma dominância lexical. Essa mudança não é arbitrária, mas encaixa o tipo existente no tipo resultante de uma operação de coerção. Segue como exemplo a sentença abaixo:

João iniciou **um livro**.

Para capturar a semântica dessa forma verbal, é necessário invocar uma regra de coerção, para assegurar que o tipo semântico do verbo seja satisfeito.

Numa estrutura lexical como a associada ao verbo "iniciar", mostrada na figura 46, o tipo do segundo argumento é um *evento*.

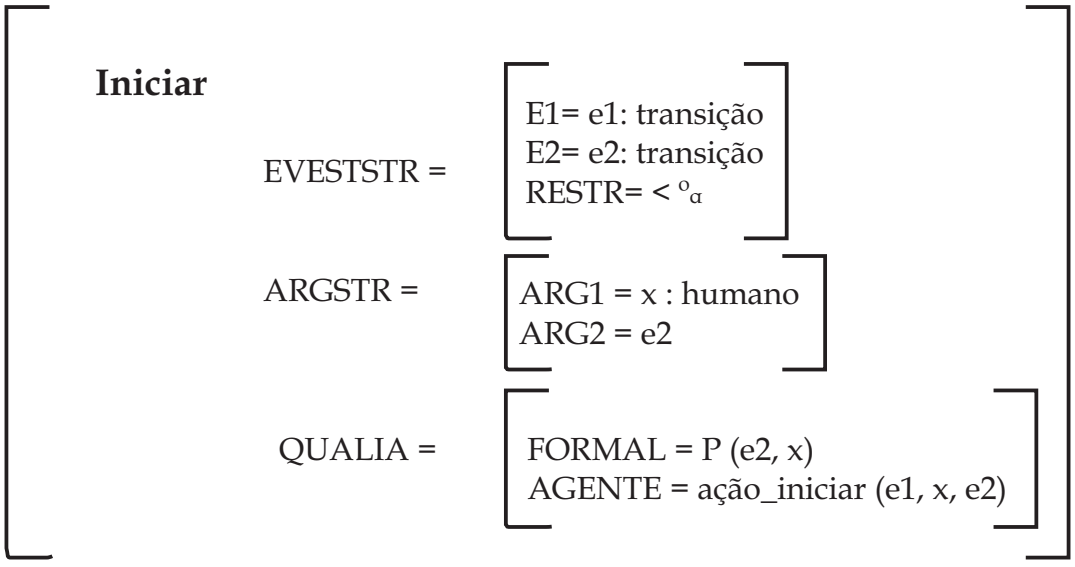

Figura 46: Estrutura Qualia Geral para o item lexical *iniciar*.

Na sentença de exemplo acima, o complemento do verbo, **um livro**, faz referência na sua Qualia Geral (figura 47) a informação e objeto_físico, não satisfazendo o tipo requerido pelo verbo "iniciar" (um livro não é um evento).

Quando algum tipo da estrutura não é diretamente satisfeito, uma coerção é aplicada para reconstruir a semântica do verbo. Uma coerção tem sucesso somente se a Qualia do PN possui um valor do tipo apropriado. Para uma sentença como a do exemplo em questão, o tipo evento é "forçado" pelo complemento "um livro", e se torna um evento ler, valor do papel Télico da estrutura Qualia do PN.

De acordo com a estrutura Qualia do verbo iniciar, a qual mostra dois argumentos,

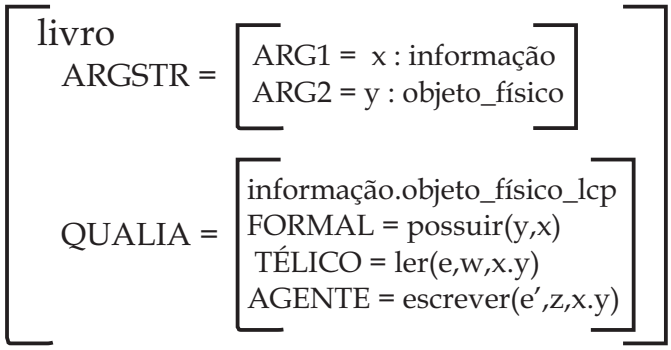

Figura 47: Estrutura Qualia Geral para o item lexical *livro*.

"humano" e "evento", conforme a árvore de representação mostrada na figura 48.

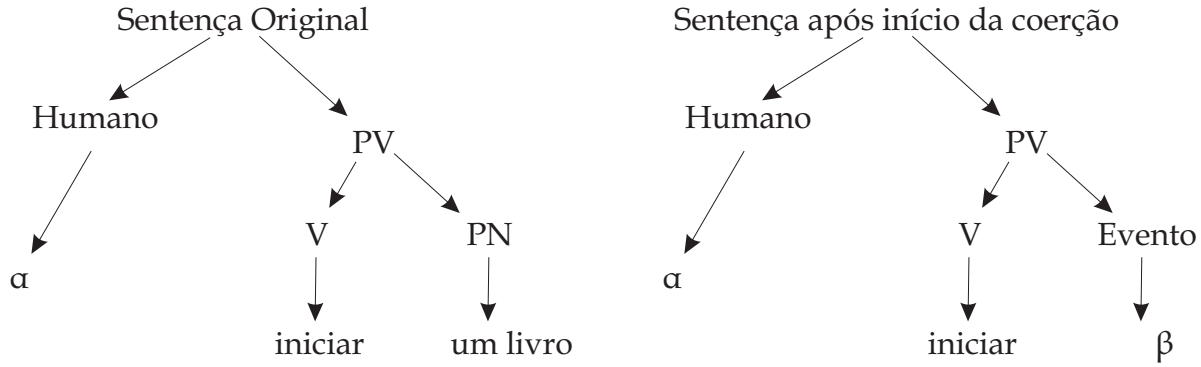

Figura 48: Árvore de Representação da Sentença.

A coerção no complemento do verbo *iniciar* pode ser vista como uma exigência deste verbo. Busca-se uma expressão que denota um evento, e então encaixa-se à semântica do PN esta expressão. Isso é ilustrado esquematicamente na figura 49 onde o evento é a expressão do papel Télico do PN um livro.

Assim, o complemento do verbo é quem constrói a semântica da sentença, quando este verbo não tem seus tipos de argumentos satisfeitos. Isso ocorre através de uma coerção do complemento (no exemplo, o tipo do complemento passou de PN para um evento).

3.2.4.2 Co-composição

Para ilustrar o procedimento de co-composição, o autor utiliza verbos polissêmicos como "assar" (*bake*), que tem dois significados: mudança de estado (passar do estado cru para o estado cozido) ou o significado de criar (para assar um bolo).

Para capturar a polissemia sem necessidade de recorrer a múltiplas listas de palavras, Pustejovsky propõe que o complemento do verbo carregue informações que incidam sobre

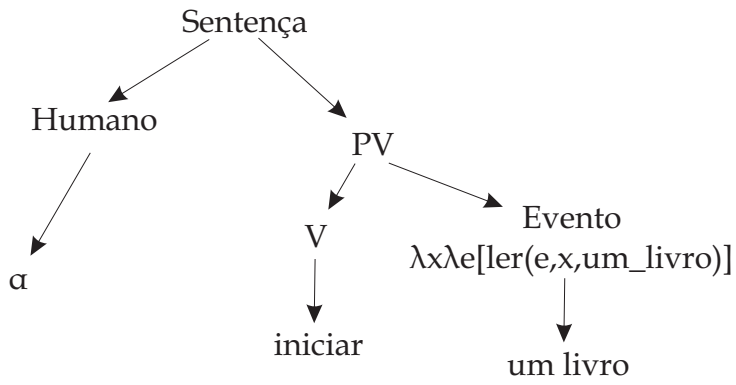

Figura 49: Árvore de Representação com a aplicação da Coerção de Complemento.

o verbo, levando o verbo essencialmente como argumento e mudando seu tipo de evento.

Desta forma, a semântica para um PV (predicado verbal) como "assou o bolo" resulta de diversas operações. Primeiramente, deve ocorrer a aplicação de uma função para ligar o objeto (bolo, nesse caso) na estrutura de argumentos do verbo "assar". Em seguida, deve ocorrer um tipo de unificação de traços do papel Agente da estrutura Qualia, isto é, $\mathrm{Q}_{\mathrm{agente}}(\mathrm{assar}) = \mathrm{Q}_{\mathrm{agente}}(\mathrm{o\_bolo})$.

Na verdade, a operação de co-composição resulta em uma estrutura Qualia para o PV, que reflete aspectos dos seus constituintes. O significado gerado resulta de uma operação que o autor denomina **Unificação Qualia**. As condições para as quais essa operação seja aplicada são apresentada abaixo:

Aplicação de Função com Unificação Qualia: para duas expressões, α, do tipo $<a,b>$, e β, do tipo a, com estruturas qualia QS_α e QS_β, respectivamente, então, se há um valor qualia compartilhado por α e β, $[_{QS_\alpha}$ ... $[Q_i = \gamma\ ]]$ e $[_{QS_\beta}$ ... $[Q_i = \gamma\ ]]$, então define-se a Unificação Qualia de QS_α e QS_β, $QS_\alpha \sqcap QS_\beta$. Além disso, $\alpha(\beta)$ é do tipo b com $\mathrm{QS}_{\alpha(\beta)}$ $= QS_\alpha \sqcap QS_\beta$.

O verbo "assar" possui uma estrutura Qualia ilustrada na figura 50, e à estrutura Qualia de "bolo" na figura 51, aplicando uma co-composição, o resultado, na figura 52, é uma representação semântica do PV não ambíguo.

Assim, de modo semelhante ao mecanismo de coerção, quem define a semântica da sentença, neste caso, é o complemento, que carrega informações necessárias para a desambiguação do verbo mas, ao invés de ter algum tipo alterado, tem uma composição de tipos para formar a semântica da sentença.

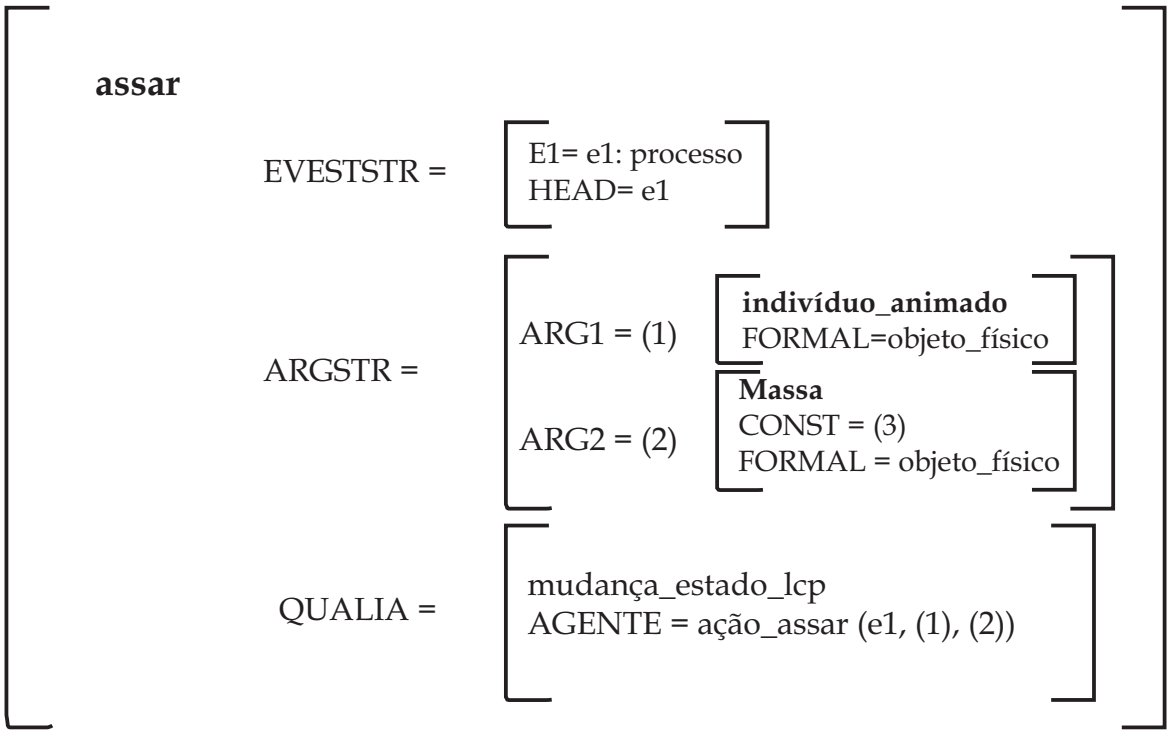

Figura 50: Estrutura Qualia Geral para o item lexical *assar*.

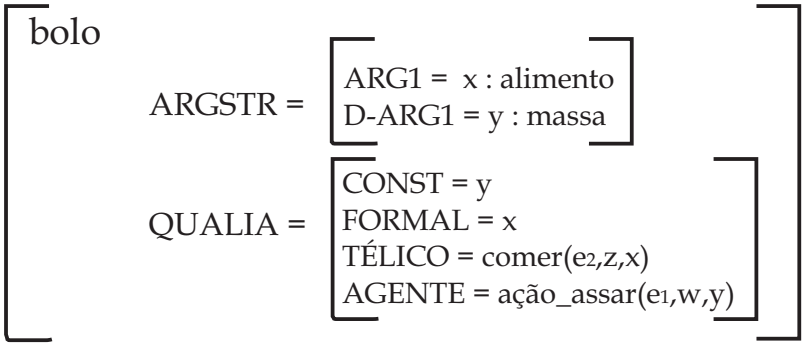

Figura 51: Estrutura Qualia Geral para o item lexical *bolo*.

3.2.4.3 Ligação Seletiva

O autor apresenta este mecanismo para solucionar o problema da polissemia dos adjetivos, como nas sentenças de exemplo abaixo.

a) João é um datilógrafo **veloz**.

b) Uma **boa** faca: uma faca que corta bem.

Pustejovsky observa dois pontos importantes: um adjetivo pode ser polissêmico, sendo capaz de modificar indivíduos ou eventos, e a interpretação do adjetivo no contexto depende da semântica do objeto a que este adjetivo se refere.

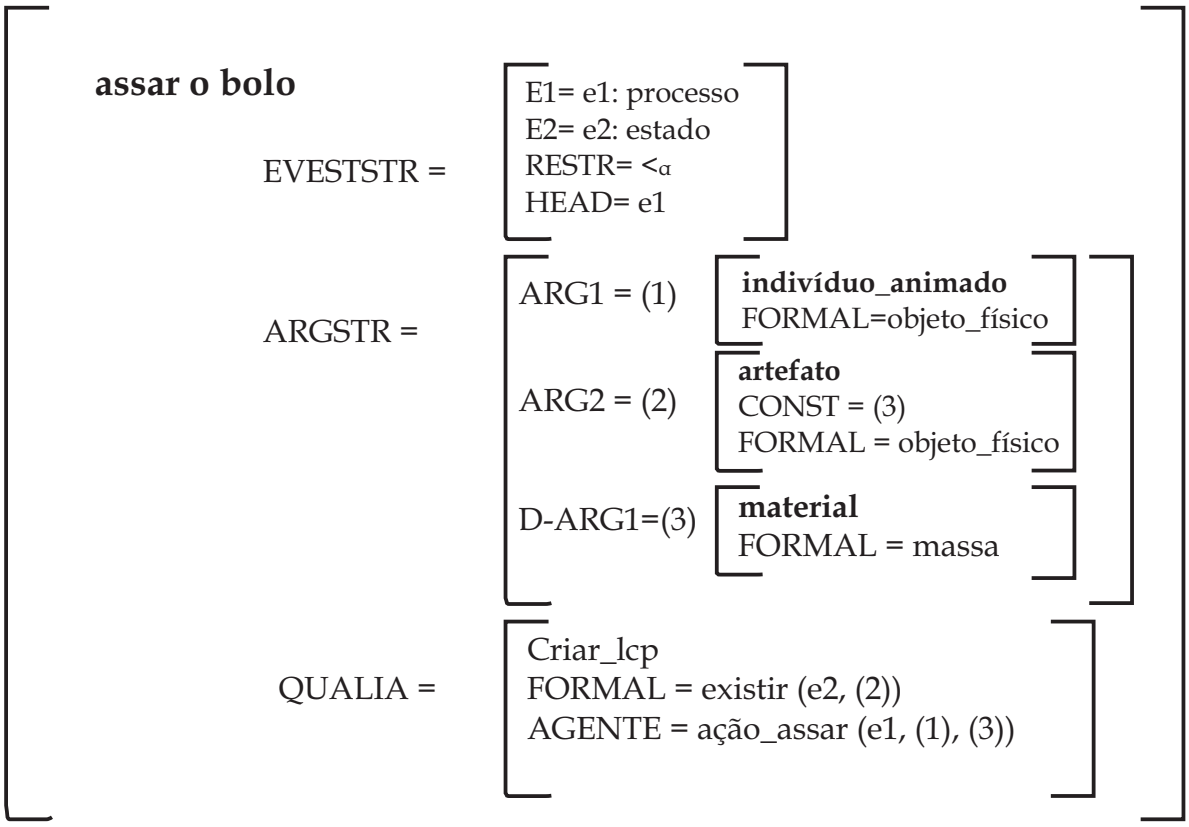

Figura 52: Estrutura Qualia Geral resultante da aplicação da co-composição.

Assim, considerando a primeira sentença do exemplo acima, utilizando o adjetivo "veloz" como um modificador dado em λx[datilógrafo(x) $\wedge$ veloz(x)], o problema se refere a alcançar a interpretação "João é um datilógrafo que é veloz na datilografia". A estrutura Qualia para "datilógrafo" é dada na figura 53.

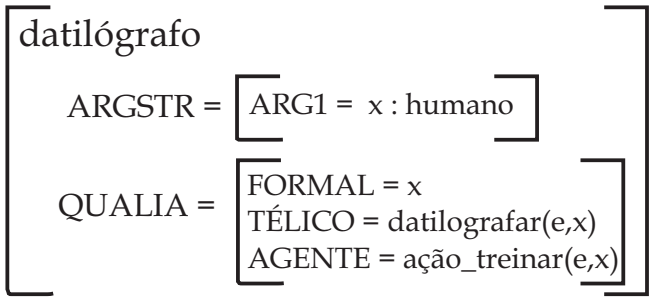

Figura 53: Estrutura Qualia Geral para o item lexical *datilógrafo*.

Se, em alguma estrutura, "veloz" está definido como um predicado de evento, então não há um modo padrão de composição que permitiria uma interpretação desejada para a sentença (a), logo acima. Essa estrutura é mostrada logo abaixo:

λx[..... Télico = λe[datilografar(e,x) $\wedge$ veloz(e)]....]

Por tanto, o adjetivo pode fazer disponível uma interpretação seletiva de uma expressão de evento contida numa estrutura Qualia para o substantivo principal a que ele se refere. O que possibilita esta interpretação é o mecanismo que Pustejovsky denomina de **Ligação Seletiva**.

Define-se Ligação Seletiva como: se α é do tipo <a,a>, β é do tipo b, e a estrutura qualia de β, QS_β, tem qualia, q do tipo a, então $\alpha\beta$ é do tipo b, onde $[\alpha\beta] = \beta \cap \alpha(q_\beta)$.

Este mecanismo trata os adjetivos como funções que são aplicadas a valores Qualia específicos do sintagma nominal. Por exemplo, na sentença (b) acima, o adjetivo **boa** para faca indica que "a faca corta bem". Possuindo na estrutura Qualia Geral de faca (como mostra a figura 54), no papel Télico, o evento *cortar(e,x,y)*, onde x é uma ferramenta e y é qualquer coisa cortável, o adjetivo boa selecionaria o evento cortar para a ligação e formaria a expressão semântica da sentença conforme abaixo:

λx[TÉLICO = λe[cortar(e,x,y) $\wedge$ boa(e)]]

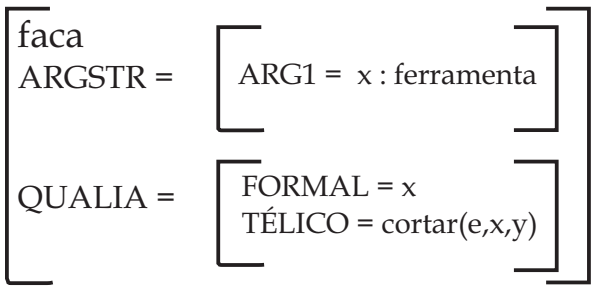

Figura 54: Estrutura Qualia Geral para o item lexical *faca*.

Supõe-se que exista, neste caso, uma estrutura Qualia para o adjetivo boa a qual permite selecionar eventos como argumentos.

Enfim, a Teoria Lexico Gerativo é uma teoria que possui um certo nível de complexidade, devido aos vários conceitos existentes e na sua execução propriamente dita; sempre objetivando, principalmente, os "tipos dos ítens léxicos", como fatos de ordem na formação dos mesmos. Buscando assim, aqueles tipos que melhor se encaixaria a um determinado domínio. Com isso, essa teoria pode auxiliar, numa possível resolução do problema da ambigüidade. Utilizando pra isso, vários mecanismos gerativos e principalmente a Estrutura Qualia Geral (ou Estrutura Léxica), a qual se faz mais presente em seus trabalhos.

3.2.5 GLB - *Greatest Lower Bound* e LUB - *Least Upper Bound*

Os conceitos matemáticos de GLB (Limite Inferior Máximo) e de LUB (Limite Superior Mínimo) são utilizados na teoria do Ponto Fixo [Lloyd 1984].

A Teoria Léxico Gerativo também emprega esses conceitos na definição e unificação de tipos. Para compreender esses conceitos é necessário definir **ordem parcial**.

Definição 6: Uma relação R em um conjunto S é de ordem parcial se as seguintes condições são satisfeitas [Lloyd 1984]:

(a) xRx, $\forall$x ϵ S.

(b) xRy e yRx $\longrightarrow$ x = y, $\forall$x,y ϵ S.

(c) xRy e yRz $\longrightarrow$ xRz, $\forall$x,y,z ϵ S.

Exemplo: A relação de inclusão ($\subseteq$) é uma relação de ordem parcial no conjunto 2^S correspondente ao conjunto de todos os subconjuntos de um conjunto S. A partir daqui será adotada a notação padrão $\leq$ para denotar uma relação de ordem parcial.

Definição 7: Seja S um conjunto com uma relação de ordem parcial $\leq$. Um elemento $a \epsilon$ S é uma borda superior (*upper bound*) de um subconjunto X de S se $\forall x \epsilon$ X, $x \leq a$. Similarmente, uma borda inferior (*lower bound*) de X se define como um elemento $a \epsilon$ S tal que $\forall$x ϵ X, $a \leq x$ [Lloyd 1984].

A **borda superior mínima** de um subconjunto X de S é uma borda superior a de X, tal que, para todo limite superior a' de X, $a \leq a$'. Já a **borda inferior máxima** (GLB) de X é uma borda inferior b de X tal que, para todo limite inferior b' de X, $b' \leq b$.

A borda superior mínima de X é única, caso exista, e é denominada *lub (X)*. Da mesma forma, a borda inferior máxima de x é única, caso exista, e é denominada glb(X).

Definição 8: Um conjunto L com uma relação de ordem parcial é um **reticulado completo** se lub(X) e glb(X) existem para todo subconjunto X de L [Lloyd 1984].

Notação: lub(L) = $\top$ (elemento *top* lub (L)), glb(L) = $\bot$ (elemento *bottom* glb(L)). $\top$ é o elemento **topo** do reticulado e $\bot$ é o elemento **base** do reticulado.

Por Exemplo: ($\mathbb{N}$, $\leq$) não é um reticulado completo, já que lub(N) não existe. Por outro lado o conjunto de todos subconjuntos de um conjunto S é um reticulado completo, onde $\top = S = \text{lub}(2^S)$ e $\bot = \emptyset = \text{glb}(2^S)$. Observe-se que o lub de um conjunto de subconjuntos de S é a união entre eles e que o glb é a intersecção dos mesmos.

O elemento *top* do "lub" é usado pela Teoria Léxico Gerativo para definir um tipo que abrange vários outros tipos, ou seja, uma disjunção (união) de tipos, conforme é mostrado na figura 55, abaixo.

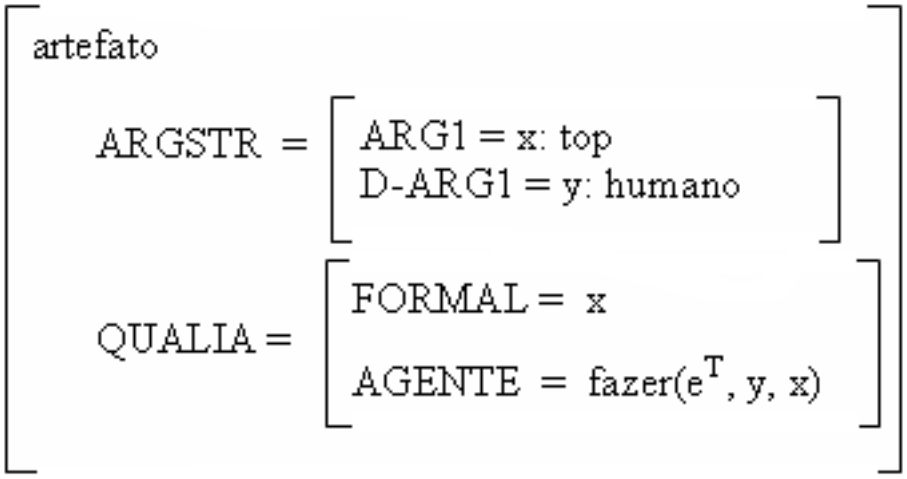

Figura 55: Estrutura lexica com o tipo *top*.

De acordo com a figura acima, o argumento "x" é tipo *top*. Como o item léxico "artefato" pode receber um dos vários tipos de argumentos (construção social, uma ação verbal, objeto físico, etc), o tipo *top* irá representar todos esses tipos de argumentos possíveis para o item lexico "artefato". É devido ao significado de união (disjunção) que o elemento *top* do "lub" é utilizado para definir um tipo que pode representar vários outros.

Além do *top*, a Teoria Léxico Gerativo utiliza o glb para realizar unificações de tipos com estruturas Qualia para gerar novos itens léxicos.

De fato, considerando o exemplo da figura 55, a estrutura Qualia do item "artefato" pode-se unificar com o sub-tipo "objeto físico" gerando a estrutura qualia da figura 56, que representa novo item léxico "artefato_físico". Este, por sua vez, corresponde a uma particularização do item artefato (isto é, representa um artefato que é um objeto físico) que pode ser visto como um tipo unificado similar à proposta de Copestake [Copestake 1992] baseada no glb entre tipos.

3.3 *Parsers*

Na Ciência da Computação, os *parsers* são formalmente considerados analisadores sintáticos, ou seja, um programa que pode analisar um encadeamento de entradas (tal como o processo de leitura de um arquivo ou de uma seqüência teclada. Essa análise determina a

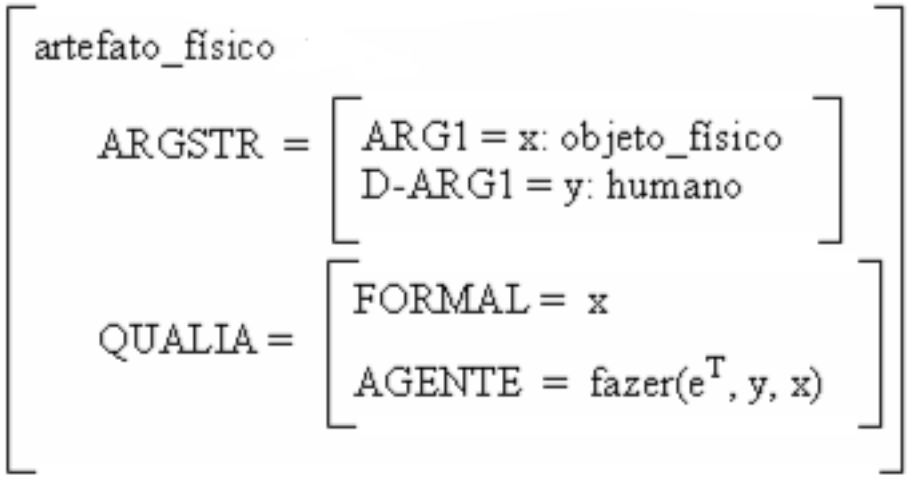

Figura 56: Estrutura lexica para o item *artefato_fisico*.

estrutura gramatical do encadeamento de acordo com uma determinada gramática formal definida [Wikipédia 2006].

O processo de análise resulta em uma estrutura de dados (normalmente, uma árvore), a qual é originária de um texto de entrada. Essa estrutura de dados é posteriormente processada de uma forma mais útil e, captura a hierarquia implícita desta entrada. Os *Parsers*, em geral, trabalham em duas etapas, a primeira reconhece os *tokens* (conjunto de caracteres (de um alfabeto, por exemplo) com um significado coletivo), expressivos na entrada, para então construir uma árvore a partir desses *tokens*.

Com isso, pode-se definir um *Parser* da seguinte forma: "É um programa (ou uma parte de um programa) que é útil para analisar a estrutura gramatical de uma entrada, por meio da manipulação dos *tokens*" [Wikipédia 2006].

3.3.1 *Parsers* Elementares

Nessa subseção, alguns *parsers* elementares que foram utilizados na construção do RALQ (mostrado no capítulo 4) serão mostrados, no intuito de elucidar a combinação dos mesmos com os operadores de *parsers* [Koopman et al. 1997].

O primeiro *parser* que será destacado é o "*symbol*", cuja função será de reconhecer um símbolo especificado como parâmetro. Por exemplo: (symbol ' ') testa a cadeia de entrada para ver se é um símbolo vazio (espaço em branco). Caso seja, ele será reconhecido.

O *parser* "*token*" é um que reconhece uma cadeia de símbolos fixos, tais como ['met'] ou ['but'], ou seja, se na cadeia de caracteres de entrada aparecer "met" ou "but" serão validados pelo *parser token*. Os tokens são utilizados da seguinte forma: token ['met'] e

token ['but'].

Já o *parser satisfy* possui um predicado, o qual será empregado na cadeia de símbolos de entrada com o objetivo de satisfazer esse predicado. Por exemplo: (satisfy isSeparators), em que o parser *satisfy* utiliza o predicado *isSeparators*, cuja função é verificar se um símbolo é um separador ou não.

3.3.2 Combinadores de Parsers

O combinadores são operadores que manipulam *parsers*. Com o uso destes combinadores, é possível escrever *parsers* para gramáticas ambíguas de uma maneira mais elegante. Nesta seção serão apresentados os combinadores de *parsers* utilizados no desenvolvimento do RALQ.

Usando os *parsers* elementares acima, novos *parsers* podem ser construídos para símbolos terminais (um símbolo) de uma gramática. Mas, o mais interessante são os *parsers* para símbolos não terminais (cadeia de símbolos). Operações importantes nos *parsers* são as composições seqüenciais e alternativas. Essas podem ser desenvolvidas por duas funções, as quais por conveniência de notação estão definidas como combinadores [Koopman et al. 1997]: <&> para composição sequêncial, e <|> para composição alternativa. Prioridades destes combinadores são definidas para minimizar parênteses em situações práticas: o combinador <&> associa à direita e terá prioridade nível 6, considerando que o combinador <|> tem prioridade nível 4.

Ambos os combinadores têm dois parsers como parâmetro e geram um terceiro *parser* como resultado de sua operação. Logo, a combinação de *parsers* geram novos *parsers*.

O combinador seqüencial (<&>) operando entre dois *parsers* P1 e P2, onde P2 corresponde a uma abstração lâmbda, funciona do seguinte modo: primeiro, o *parser* p1 deve ser aplicado à lista de entrada recebida pelo combinador. Depois disso, p2 é aplicado ao resto da lista de símbolos de p1, caracterizando a chamada listas das diferenças [Koopman et al. 1997]. Tal lista corresponde à lista original recebida por P1 desfalcada do item sintático consumido na aplicação do referido *parser*. O uso desse combinador (operador) na programação do sistema RALQ tem como objetivo reconhecer linearmente todos os componentes dos nomes orgânicos, utilizando, para isso, a combinação dos *parsers* que o formam. Assim, o combinador (<&>) repassa a lista restante obtida da aplicação do *parser* de entrada para o próximo *parser*. O resultado da aplicação do primeiro *parser* é armazenado no parâmetro da abstração lambda (que é representada por \s-> na lin-

guagem funcional Clean) [Koopman et al. 2002]. Por exemplo:

```
(1) withMult = radicalPosicao <&> \s-> (alkaneRadical <@ (\x->
                (mkAlkane x s))) <& alkaneFunction;
```

Outro combinador de *parsers* que também pode fazer uso do operador lambda é o combinador transformador <@. Considerado o combinador de transformação mais importante, ele transforma um *parser* em um outro *parser* que corresponde ao primeiro mais a aplicação de uma outra função adicional [Koopman et al. 1997].

O combinador <@ aplica uma dada função ao resultado da aplicação de um dado *parser*. Esse símbolo foi escolhido com o propósito de ter uma semântica de "aplicar" a função ao resultado de um *parser*, em que a "seta" (<), do combinador <@, sempre aponta da função em direção aos argumentos. Dado um parser **p** e uma função **f**, o combinador <@ retorna um *parser* que faz o mesmo que **p**, adicionando a aplicação de **f** ao resultado da aplicação do *parser* [Koopman et al. 1997].

Em (1), por exemplo, o combinador (operador) <@ aplica a função *mkAlkane* em todos os resultados obtidos pelos *parsers radicalPosicao* e *alkaneRadical*

Outro combinador utilizado pelo RALQ, que também está presente em (1) é o combinador <&. Esse comporta-se da mesma forma que o combinador <&>, exceto que ele descarta o resultado de um de seus dois *parsers* de argumentos, ou seja, ele vai rejeitar o resultado do *parser* que estiver a sua direita, para onde a seta (<) não está apontando. Por exemplo, no código acima (1), o resultado do *parser alkaneFunction* será descartado, fazendo com que o fluxo do programa retorne para a função *mkAlkane*.

A utilização da abstração lâmbda nos combinadores <&> e <@ é uma forma de análise gramatical baseado no conceito de Semântica Denotacional [Allison 1986]. Tal semântica descreve todas as características achadas em linguagens de programação imperativas e tem uma boa base matemática. (Ainda há pesquisa ativa em sistemas de tipo e programação paralela.)

O Cálculo Lambda foi projetado para estudar definições de funções, aplicações de funções, passagem de parâmetro e recursão. Ela é uma boa candidata para ser considerada uma linguagem de programação de protótipo e tem inspirado a liguagem de programação Lisp e as linguagens de programação moderna (Clean, por exemplo) [Allison 1986]. Com isso, o cálculo lambda, empregado junto com os *parsers*, servirá de passagem de parâmetro, o qual vai auxiliar na representação semântica do RALQ.

Uma outra variação do combinador <&> é o combinador &>. Esse, tem a mesma funcionalidade do combinador <&, diferenciando apenas por descartar o resultado do *parser* no resultado do *parser* argumento que estiver à esquerda da seta (>) do combinador &>.

Combinadores de *parsers* como <!*> e <!+> pegam símbolos de acordo com um predicado: o combinador <!*> prediz 0 (zero) ou mais ocorrência de um símbolo determinado pelo predicado e o combinador <!+> uma ou mais ocorrência de um símbolo prescrito pelo predicado. Por exemplo, o código (2)determina uma possível ocorrência de 0 (zero) ou mais símbolos vazios na cadeia de entrada. Caso o combinador em (2), fosse <!+>, haveria no mímino uma ocorrência do símbolo vazio na cadeia de entrada (neste exemplo, o *parser symbol* é o predicado utilizado por esses combinadores)

```
(2) (<!*> (symbol ' '))
```

Dentre os combinadores de *Parsers* existentes, o último que foi utilizado na construção sintática do RALQ foi o combinador <!>. ele é um oporador *or-else*, o qual habilita a detecção de erros, ou seja, ativa o segundo *parser* quando o primeiro falha [Koopman et al. 1997]. Por exemplo, em (3) todos os parsers que estão fazendo parte do combinador <!> só serão ativos se os outros parsers forem falsos.

```
(3) alkaneMainChain= withoutMult <!> withMult <!> withWithoutMult <!>
                     withoutWithMult;
```

Esses foram os *parsers* e os operadores, que os combinam, empregados na construção da parte sintática do sistema RALQ. É bom salientar que, a linguagem de programação funcional Clean trabalha com vários outros *parsers* e combinadores. Entretanto, o objetivo dessa seção era mostrar o funcionamento *parsers* e ombinadores usados na construção do RALQ.

3.4 O *Xymtec*

Xymtec é um pacote de marcação que combina arquivos de estilo Latex desenvolvido para desenhar uma ampla variedade de fórmulas estruturais químicas. Os comandos de *Xymtex* tem um conjunto de argumentos sistemáticos para especificar substituições e as suas posições, ciclos internos, ligação dupla, e ligação padrão (simples). Em alguns casos,

eles têm um argumento adicional para especificar hetero-átomos no vértices de heterocíclos. Como resultado desta característica sistemática, *Xymtec* trabalha efetivamente como uma ferramenta prática dentro do "dispositivo independente" (TEX) [Fujita 1993].

3.4.1 Características do *Xymtec*

As principais características do *Xymtec* são dadas logo abaixo [Fujita 1994]:

- *Xymtec* requer somente o ambiente figura do Latex, assegurando portabilidade (desde que Latex seja parte da maioria das distribuições Tex). Dessa forma, vastas adaptações para computadores pessoais estão disponíveis e uma variedade de impressoras pode ser usada como dispositivos de saída;
- Xymtec pode ser usado em muitas versões do Latex;
- Fórmulas estruturais desenhadas com Xymtec possuem alta qualidade, desde que usem fontes Latex;
- Cada nome de comando corresponde a um modelo mestre a ser desenhado. Podendo ser lembrado facilmente, desde ele se origine da nomenclatura familiar de compostos orgânicos.
- O parte invariante de uma estrutura (o modelo mestre que contém fixado ligações e átomos) é automaticamente impresso sem nomeação.
- As partes variantes de uma estrutura (substituintes, ligações e átomos adicionais) é designado por até quatro argumentos: SUBSLIST, OPT, BONDLIST, e ATOMLIST.

3.4.2 Os comandos *Xymtec*

Nessa subseção será mostrada o funcionamento de todos os comandos *Xymtec* utilizados nesse trabalho. Como esse, trabalhou com compostos de cadeia aberta e algumas variações de posição das ligações, os outros comandos Xymtec que possibilitam o desenho de outros compostos orgânicos (por exemplo, de cadeia fechada), podem ser consultados em [Fujita 1993].

Para construir um composto de cadeia cabônica aberta usa-se o comando (marcador) **\tetrahedral**, o qual, desenhará uma unidade tetraédrica[8]. O comando tem a seguinte sintaxe:

```
\tetrahedral[AUXLIST]{SUBSLIST}
```

onde o argumento facultativo "AUXLIST" é usado para especificar uma carga no átomo central, por exemplo, {0+} representa uma carga "+" (positiva) no átomo central. Já o argumento "SUBSLIST" é usado para especificar cada substituição com um número que indica a locação e o modificador da ligação, mostrado na tabela 3, onde n é um número entre 1 e 4.

Caracter	Estruturas Geradas
n ou nS	ligação simples no átomo n
nD	ligação dupla no átomo n
nT	ligação tripla no átomo n
nA	ligação simples alfa no átomo n
nB	ligação simples beta no átomo n

Tabela 3: Parâmetros SUBLIST

O esquema seguinte (figura 57) mostra a numeração para designar posições de substituição em um átomo:

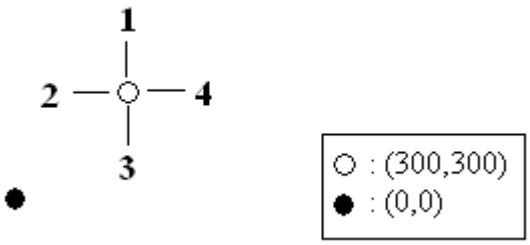

Figura 57: Esquema das posições de Substituição [Fujita 1993].

O átomo de carbono central é indicado pela escrita 0==C, ou seja, o número 0 (zero) dentro do parâmetro SUBSLIST indica o átomo que terá as suas ligações substituídas por ligações com outros átomos. Como o átomo de Carbono é tetravalente, ele vai poder ter até 4 (quatro) ligações de substituições. Por exemplo, o comandos abaixo, produzem os desenhos mostrados na figura 58.

```
\tetrahedral{0==C;1==H;2==Cl;3==F;4==Br}\qquad
```

[8]Referente a tetraedro.

```
\tetrahedral{0==C;1D==O;2==Cl;4==Cl}\qquad
\tetrahedral[{}{O+}]{0==N;1==H;2==CH$_{3}$;3==H;4==H}
```

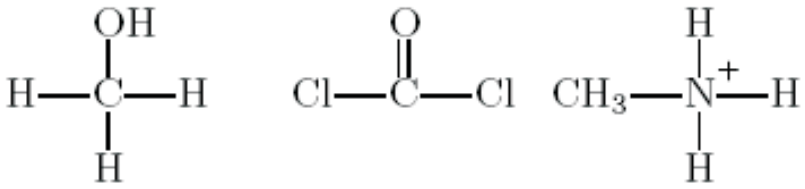

Figura 58: Desenhos feitos pelo comando \tetrahedral [Fujita 1993].

A marcação **\qquad** utilizada no exemplo acima, coloca todas as figuras geradas pelo comando \tetrahedral em paralelo, ou seja, uma do lado da outra.

Os comandos (marcação) **\rtrigonal** e **\ltrigonal** são usados para desenhar unidades trigonais pelo lado direito e esquerda. O formato desses comandos são:

```
\rtrigonal[AUXLIST]{SUBSLIST}
\ltrigonal[AUXLIST]{SUBSLIST}
```

O ângulo formado pelas ligações 2, 0 e 3 é de aproximadamente 90º nas unidades trigonais impressas com estes comandos. Possuindo os mesmos argumentos AUXLIST e SUBSLIST da marcação \tetrahedral. Por exemplo, as marcações abaixo formaram os desenhos das estruturas químicas trigonais, mostrado na figura 59.

```
\rtrigonal{0==C;1==H;2D==O;3==H}\qquad
\ltrigonal{0==C;1==H;2D==O;3==H}
```

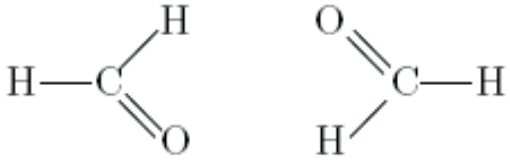

Figura 59: Desenhos Trigonais [Fujita 1993].

As marcações anteriores (trigonais) foram usadas na construção dos desenhos das estruturas químicas dos Aldeídos, tanto bem quanto a marcação \tetrahedral é usada para desenhar todas as outras estruturas das funções químicas trabalhadas pelo RALQ.

Estes foram os comandos do pacote *Xymtec* empregados nesse trabalho. No entanto, em [Fujita 1993] há várias outras marcações, além dessas vistas anteriormente, que possibilitam o desenho de novas estruturas químicas.

4 *Desambiguação das Fórmulas Químicas - A aplicação*

4.1 O Problema

O problema da ambigüidade afeta não somente a linguagem natural, mas também a química orgânica, ou seja, na linguagem natural uma palavra pode ter dois significados, como por exemplo a palavra "banco", que pode significar tanto uma instituição financeira quanto um objeto físico de assento, dependendo do tipo do complemento que esse item léxico possui, como é analisado pela Teoria Léxico Gerativo. Já na química, os nomes orgânicos podem gerar outros nomes de compostos, por meio do tipo de ramificação (Methyl, Ethyl, Propyl, etc) que é inserido na estrutura química do composto orgânico, sendo essa ambigüidade possível de ser analisada pela Teoria Léxico Gerativo, como será mostrado na seção 4.4.

Na química, se o problema da ambigüidade não for resolvido, nomes de compostos químicos incorretos ou incompatíveis com a nomenclatura oficial podem ser produzidos.

A ambigüidade na Química Orgânica ocorre devido a algumas regras aplicadas nas cadeias ramificadas de C (carbono) que originam a nomeclatura dos compostos.

Dentre essas regras, está a regra da **cadeia principal do composto**, vista no capítulo 3. É fundamental definir a cadeia principal, pois é nela que serão colocadas as ramificações. Além disso, o nome mais importante do composto é conhecido pela quantidade de carbono que essa cadeia possui [Usberco e Salvador 2001]. Para exemplificar a influência dessa regra na ambiguidade dos compostos químicos, considere os exemplos abaixo:

No exemplo da figura 60, a cadeia principal possui um encadeamento de 3 (três) átomos de carbonos na cadeia principal e uma ramificação do tipo ***ethyl*** no carbono 2 (dois). Entretanto, esse composto possui um encadeamento com uma quantidade maior de carbonos do que a mostrada na figura 60. Então, o composto **2-etil-propene** é considerado

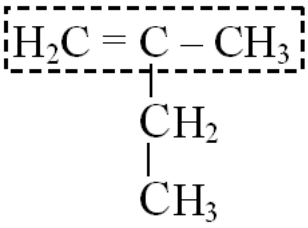

Figura 60: 2-etil-1-propene

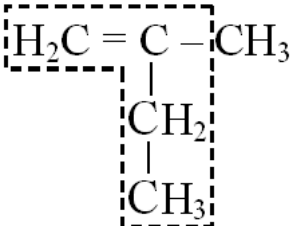

Figura 61: 2-metil-1-butene

inexistente de acordo com a regra da cadeia principal. A representação correta é mostrada na figura 61, onde o composto indicado possui uma quantidade de átomos de carbonos interligados maior do que o apresentado na cadeia principal original (figura 60).

Uma outra regra que pode influenciar a presença da ambigüidade é a chamada **regra dos menores números** definida no capítulo anterior, pois, nesta regra, uma estrutura química pode ter dois ou mais sentidos para definir a posição dos átomos de carbono na cadeia principal e, conseqüentemente, vários nomes. Para que essa regra seja aplicada é necessário, primeiramente, definir a maior quantidade de carbonos interligados presente na estrutura química composto orgânico (a cadeia principal do composto), ou seja, a regra da cadeia principal deve ser aplicada antes da aplicação da regra dos menores números. Por exemplo, uma chamada tal como **2,3-dietil-4,4-dimetil-3-pentanol** pode gerar a estrutura química mostrada na figura 62.

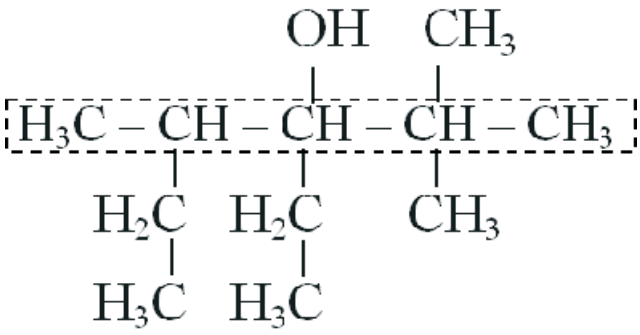

Figura 62: 2,3-dietil-4,4-dimetil-3-pentanol

O nome da estrutura química acima gerada viola a regra da cadeia principal, pois há na estrutura encadeamentos mais longos de átomos de carbono para representar a cadeia principal, como mostram as figuras 63 e 64 abaixo, onde a regra da cadeia é satisfeita.

No entanto, é necessário aplicar a regra dos menores números, tanto na estrutura da figura 63, quanto na estrutura da figura 64, para obter o nome correto do composto, caso contrário ele continuará ambígüo. Por exemplo, na figura 63, no sentido da direita para a esquerda da cadeia principal hachurada, o nome do composto formado é **2,2,4-trimethyl-**

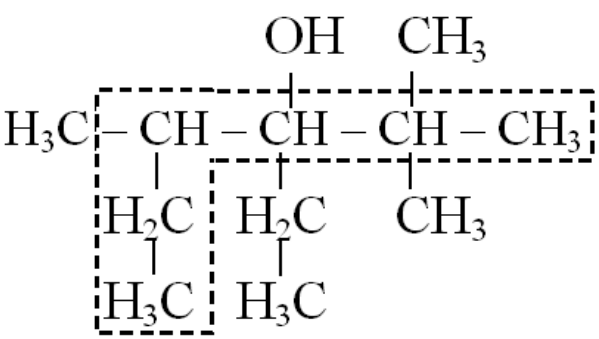

Figura 63: Estrutura química 1

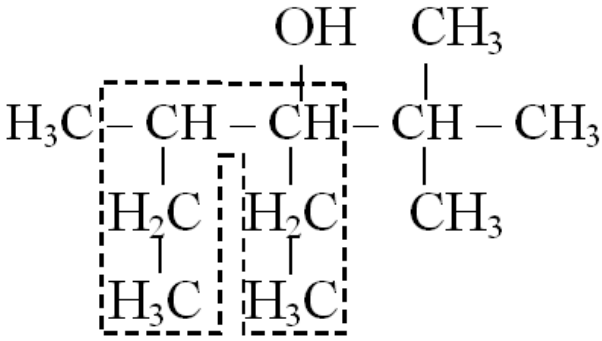

Figura 64: Estrutura química 2

3-ethyl-3-hexanol; já no sentido oposto, o nome que surge é: **3,5,5-trimethyl-4-ethyl-4-hexanol**. Contudo, este último nome do composto não é válido, de acordo com a regra dos menores números, para a função dos Álcoois. Prevalecerá assim, o nome **2,2,4-trimethyl-3-ethyl-3-hexanol** para a estrutura química 1 (um) representada na figura 63.

O mesmo processo empregado na figura 63 é válido para a estrutura química 2 (dois) da figura 64, a qual possuirá o nome **3-terciobuthyl-4-methyl-3-hexanol**, de acordo com a regra dos menores números para os Álcoois.

Nesta dissertação, será usada a denominação "nome correto" para todo nome de composto orgânico que estiver em conformidade com as regras oficiais de nomenclatura. Por outro lado, usar-se-á a denominação "nome inadequado" a todo nome de composto que, apesar de não satisfazer os critérios da nomenclatura oficial, conseguir identificar, inequivocamente, um composto orgânico válido da Química. Exemplos de nomes corretos são: "2,2,4-trimethyl-3-ethyl-3-hexanol" e "3-terciobutil-4-methyl-3-hexanol" para o composto ilustrado nas figuras 62, 63 e 64. Exemplos de nomes inadequados são: "2,3-diethyl-4,4-dimethyl-3-pentanol" e "3,5,5-trimethyl-4-ethyl-4-hexanol" para o mesmo composto das figuras citadas.

O Sistema RALQ esta apto a tratar todas essas ambigüidades e, mesmo quando o nome de entrada é inadequado, ele o analisa e retorna, tanto a estrutura correta do composto identificado pelo nome de entrada, como seu nome corrigido.

Outro exemplo de ambigüidade é mostrado na estrutura química da figura 65. A cadeia principal não possui somente 6 (seis) átomos de carbonos interligados, mas, sim, 7 (sete). Isso faz com que o nome do composto da figura 65, por violar a regra da cadeia, seja inadequado. Já a figura 66 representa o nome correto da estrutura química.

Portanto, o presente trabalho pretende mostrar a aplicação das técnicas mencionadas

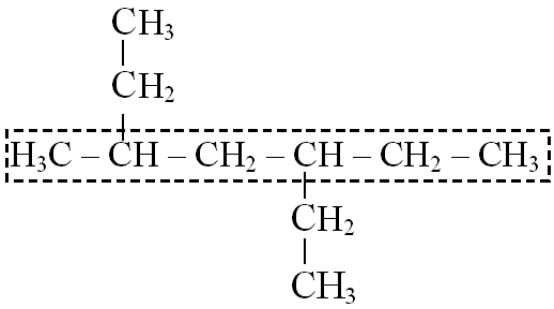

Figura 65: 2,4-dietil-hexano

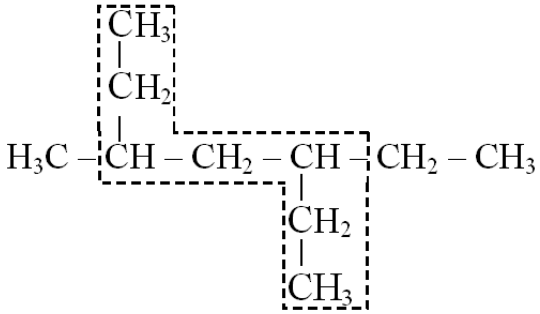

Figura 66: 3-etil-5-metil-heptano

no capítulo 3, no intuito de buscar resolver as ambigüidades que podem ocorrer nas funções orgânicas, como mostra os exemplos acima.

4.2 Especificação do RALQ

O sistema RALQ tem dois objetivos principais: primeiro, analisar sintática e sematicamente nomes de compostos químicos. Segundo, desenhar as estruturas químicas correspondentes aos mesmos. Para tanto, ele utiliza as seguintes ferramentas e técnicas que lhe possibilitam cumprir seus objetivos de forma útil e eficaz: a técnica da **Teoria do Léxico Gerativo**, os ***Parsers*** (Combinadores de *Parsers*), a **Linguagem Funcional CLEAN 2.2.1** e um pacote em Latex[1] para montar graficamente fórmulas estruturais químicas denominado **Xymtex**[Fujita 1993].

Conforme visto no capítulo 3, a Teoria do Léxico Gerativo utiliza os tipos dos itens léxicos como fatores de organização da composição dos mesmos. Tal estratégia auxilia na resolução de ambigüidades, pois, essa teoria dá todas as condições para que o tipo de um ítem léxico seja encontrado de acordo com os seus possíveis complementos. Da mesma forma, essa teoria pode trabalhar no domínio da química orgânica, levando em conta os tipos definidos para as funções químicas, bem como os prefixos e insaturações presentes nos nomes dos compostos químicos. Dessa forma, ela representa um componente importante no processo de desambiguação de compostos orgânicos, apontando os tipos próprios para cada composto orgânico, por meio das restrições implantadas na estrutura Qualia para uma determinada função química.

Essa relevância dos tipos na teoria Léxico Gerativo torna a linguagem funcional CLEAN uma excelente opção como ferramenta de implementação do RALQ, devido às suas três características principais citadas abaixo [Group 2006]:

[1] http://www.latex-project.org/

- possui um sistema de tipos único (*uniqueness typing*), ou seja, baseada no esquema de tipos de inferência Milner / Mycroft. Este inclui os tipos de alto nível, polimorfos, algébricos, abstratos, sinônimos e quantificados existencialmente.

- é transparente (seu resultado só depende dos valores de seus argumentos). Esta propriedade tem como conseqüência o fato de que, uma vez que a função foi executada com sucesso, ela sempre executará corretamente, reagindo da mesma forma, não importando o contexto em que ela se encontra.

- podem-se definir explicitamente estruturas cíclicas e compartilhadas em Clean. Devido a isto e ao fato de todos os objetos criados serem obrigatoriamente inicializados, erros em tempo de execução são raros e limitados aos 3 seguintes casos:

    1. Quando funções parciais são chamadas com argumentos fora de seu domínio (ex. divisão por zero);
    2. Quando Arrays são acessados com índices fora de seu alcance (ex. o array tem tamanho 20 e chama-se array[30]);
    3. Quando não foi atribuído memória suficiente para uma aplicação em Clean (o tamanho do programa ultrapassa o tamanho da memória dedicada ao programa).

É por essas características e também pela possibilidade de usar os combinadores de *parsers* através de interfaces, que a linguagem de programação funcional Clean foi utilizada na construção da aplicação de desambiguação de compostos químicos orgânicos.

Com a utilização do Clean, a idéia de poder realizar composições das estruturas dos componentes dos nomes dos compostos químicos e de tornar o programa útil para as pessoas que se interessam por estruturas de fórmulas da quimíca orgânica, os *Parsers* (apresentados no capítulo anterior), ou melhor, a combinação dos mesmos, foi de grande valia para que o programa pudesse ser interativo com o usuário (e, conseqüentemente, amigável).

Essa interatividade estará presente na entrada dada pelo usuário (nome da fórmula química), sendo que o programa fará a leitura de um arquivo contendo várias entradas a serem processadas. Os combinadores de *parsers* serão fundamentais para a análise de cada termo corrente no nome do composto químico, ou seja, na realização da análise sintática dos componentes dos nomes químicos. Todos os termos (radicais, multiplicadores, a cadeia principal e o sufixo funcional) presentes no nome do composto químico serão analisados

por uma combinação de *parsers* com a finalidade de serem reconhecidos. Além disso, os combinadores de *parsers* contribuem no abastecimento da semântica do RALQ, ou seja, realizam a ligação entre a análise sintática e a semântica do sistema.

Conforme visto no parágrafo anterior, a entrada da aplicação será um ou vários nomes de compostos químicos armazenados em um arquivo a serem identificados. Já a saída do programa será uma compilação em comandos Latex que fará o desenho da estrutura química do composto válido (processo discutido na subseção 4.3.6). Esses comandos em latex que possibilitam os desenhos das estruturas químicas são denominados ***Xymtec*** (ver capítulo 3). Então, o final do processamento do nome do composto químico consistirá na geração de um arquivo **.tex** (extensão do Latex) contendo, para cada nome do composto da entrada, um conjunto de comandos específicos do pacote *Xymtec* que gerará a estrutura química do composto (tal etapa corresponde a uma compilação para códigos *Xymtec* da estrutura semântica gerada pelo Analisador Semântico). Entretanto, essa compilação é feita de modo que, para cada situação de ligação de algum carbono, há um código *Xymtec* específico, fazendo assim, uma tarefa trabalhosa na implementação de todas as situações dos átomos de carbonos interligados. Além disso, há o problema das ligações das ramificações que exige uma manipulação do pacote *Xymtec* que o torne apto a tratá-las, sendo esta mais uma etapa árdua cumprida na elaboração do RALQ.

A grande vantagem da utilização desse pacote no sistema como parte final da execução da aplicação é a obtenção de uma representação ilustrada da estrutura química do composto de entrada, mostrando de forma correta as estruturas químicas válidas.

Portanto, as ferramentas e as técnicas utilizadas na resolução do problema têm atuação específica na funcionalidade de cada parte do programa. Na próxima seção, serão apresentados, minuciosamente, todas as partes dessa aplicação.

4.3 A Implementação

Nesta seção são apresentados os aspectos principais do programa que implementa o RALQ. Será mostrado como o sistema concilia a Teoria do Léxico Gerativo, os combinadores de *parser* e o pacote *Xymtec* para efetuar a análise e a representação de fórmulas da Química Orgânica, resolvendo, inclusive, problemas de ambigüidade.

O programa trabalha com as seguintes funções da química orgânica: os Hidrocarbonetos (Alcanos, Alcenos, Alcinos e Alcadienos), os Álcoois e os Aldeídos. A função orgânica dos Alcanos, foi implementada no sentido de formar uma cadeia principal com no má-

ximo 6 (seis) átomos de carbonos interligados. Já as funções químicas restantes, foram implementadas para terem, na cadeia principal, até 5 (cinco) átomos de carbonos interligados. Os compostos químicos ambíguos serão tratados quando esses forem reconhecido pela aplicação, como será mostrado na seção 4.4.

Na implementação do RALQ, todos as palavras destinadas aos nomes dos compostos orgânicos, como metil, etil, butano etc; foram transcritas para o idioma inglês, por motivo da padronização de programação utilizada nesse trabalho. Ficando como *methyl*, *ethyl*, *butane*, etc.

4.3.1 Arquitetura

A estrutura geral do programa é ilustrada na figura 67. A seqüência é a seguinte: há um processo de leitura de cada linha de um arquivo (**teste.pac**) que contém os nomes dos compostos químicos. Esse processo de leitura resulta em uma cadeia de caracteres (*Strings*[2]), para cada linha do arquivo que contém o nome químico. Essa cadeia é transformada em uma estrutura de dados passível de ser manipulados pelos combinadores de *parser*.Tal estrutura corresponde a uma lista de caracteres, ou seja, uma lista do tipo *Char* (tipo primitivo do Clean).

Colocada a entrada no formato reconhecido pelos combinadores de *parsers*, os mesmos vão trabalhar no sentido de efetuar uma análise sintática dos nomes químicos de entrada. Esta análise é feita para reconhecer os componenentes do nome químico (prefixos, a cadeia principal e a função do composto químico). Em seguida, os combinadores de *parsers* organizam tais componentes em estruturas de dados que são repassadas às funções que farão a Análise Semântica. Posteriormente, a estrutura semântica gerada será compilada para comandos *Xymtec* que, executados pelo Latex, permitem a visualização da figura que corresponde ao composto químico correspondente ao nome da entrada inicial.

4.3.2 Tipos de Dados

Todo o programa foi feito utilizando a Linguagem Funcional Clean V. 2.1.1 devido às suas especificações e vantagens mostradas na seção 4.2. Como Clean é uma linguagem que trabalha muito bem com Tipos, a primeira coisa a ser feita é a definição dos tipos utilizados no programa.

Foram utilizados tipos algébricos [Koopman et al. 2002] para representar as funções

[2]Tipo primitivo do Clean

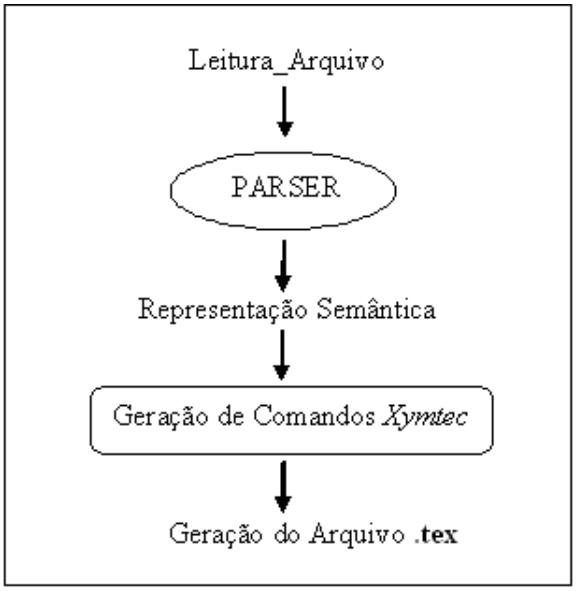

Figura 67: Estrutura do Programa

químicas, o átomo de carbono, as ramificações, as saturações, as insaturações e possíveis erros, mostrados logo abaixo:

```
(4) ::OrgType = Alkane [Atom] | Alkene [Atom] | Alkadyene [Atom]|
                Alkyne [Atom] | Alcohol [Atom]| Aldehyde [Atom] |Unknown;
```

O tipo algébrico ***OrgType*** é o tipo que indica as funções orgânicas com que o programa trabalha. Esse tipo possui construtores de dados que indicam as funções químicas (*Alkane* (Alcanos), *Alkene* (Alcenos), *Alkadyene*(Alcadienos), *Alkyne* (Alcinos), *Alcohol* (Álcoois) e *aldehyde* (Alcadieno)), que são examinados pelo RALQ. O construtor *Unknown* (desconhecido) é utilizado no tratamento de erro, quando uma função da química orgânica não é reconhecida pelo programa.

Os construtores de dados acima possuem uma lista do tipo *Atom* (Átomo) como argumento, tipo, este, definido em seguida:

```
(5)         ::Atom = C (Branch, Branch, Branch, Branch);
```

O tipo *Atom* representa o átomo de Carbono definido pelo construtor "C", o qual possui como argumento uma tupla de quatro argumentos do tipo *Branch* (ramificação). O motivo dessa tupla com 4 (quatro) argumentos é a representação das ligações que são possíveis de serem feitas pelo átomo de carbono, como especificado no capítulo 3.

O último tipo definido no programa é o tipo *Branch*. Esse tipo algébrico vai representar os tipos de ligações que o átomo de carbono pode realizar, como definido abaixo:

```
(6) ::Branch = H | O | OH | Methyl Int | Ethyl Int | Prophyl Int |
               Buthyl Int| Penthyl Int | Isoprophyl Int |
               Secbuthyl Int |Terciobuthyl Int |Isobuthyl Int |
               Vinyl Int | Alyl Int| Etinyl Int | Hidrox Int | Dimethyl |
               Diethyl | Trimethyl | Triethyl | Tetraethyl | Tetramethyl  |
               Pentaethyl | Pentamethyl | Hexaethyl | Hexamethyl |
               Hectaethyl | Hectamethyl | Octaethyl | Octamethyl | SL |
               DL | TL | WL| E;
```

O tipo *Branch* pode ser átomos, por exemplo, os construtores de dados [3] "H" que representa o átomo de hidrogênio e o "O" que representa o átomo de oxigênio. Podem representar, também, radicais através dos construtores "Methyl", "Ethyl", "Propyl", "Buthyl", "Penthyl", "Isopropyl", "Vinyl", "Alyl", "Etinyl", "Secbuthyl", "Terciobuthyl", "Isobuthyl", "Hidrox", todos esses possuem apenas um argumento do tipo primitivo inteiro (*Int*). Esse argumento inteiro é para indicar a posição que o radical estará na cadeia principal.

O tipo *Branch* pode indicar um multiplicador, o qual indica a quantidade de um radical presente no nome do composto, que é abstraído através dos construtores: "Dimethyl", "Diethyl", "Trimethyl", "Triethyl", "Tetraethyl", "Tetramethyl", "Pentaethyl", "Pentamethyl", "Hexaethyl", "Hexamethyl", "Hectaethyl", "Hectamethyl", "Octaethyl", "Octamethyl", sendo que cada um desses construtores servirá apenas para identificar um determinado multiplicador presente no nome orgânico. Identificado esse multiplicador, uma função será executada no intuito de transformar esse multiplicador numa lista do tipo *Branch*.

O tipo de ligação que o átomo faz com algum radical e outros elementos químicos, também é representado pelo tipo *Branch*, como o construtor "SL" (*simple linking* - ligação simples), "DL" (*double linking* - ligação dupla) e "TL" (*triple linking* - ligação tripla).

Por último, o construtor "E" (presente no tipo *Branch*) é utilizado para fazer o tratamento de erros na parte semântica do sistema, ou seja, na representação das estruturas químicas.

[3] Cada construtor de dados funciona como uma função (eventualmente, constante) que recebe argumentos (do tipo indicado para o construtor) e constroi um valor do novo tipo de dados.[Koopman et al. 2002]

4.3.3 Regras de Formação dos Nomes Orgânicos

No reconhecimento de nomes de compostos orgânicos, algumas regras foram criadas no intuito de legitimar e facilitar a fragmentação dos nomes presentes no arquivo de leitura da aplicação.

A regra principal encabeçada pelo **Comp_Orgânico** (ilustrada na figura 68) determina a constituição do nome de um Hidrocarboneto, formado pelo nome da cadeia principal, por 0 (zero) ou mais prefixos (sendo essa condição mostrada na figura 68 através do sinal de asterisco (*)[4]) e por um sufixo. O sufixo dará a informação da saturação ou insaturação do composto orgânico, formando, assim, nomes de cadeias saturadas ou insaturadas. Por exemplo, 2-methyl-butane (**2-methyl** - prefixo; **but** - cadeia principal; **ane** - sufixo saturado) e 3-ethyl-1-pentyne (**3-ethyl** - prefixo; **1** - Loc_insaturação; **pent** - cadeia principal; **yne** - sufixo insaturado).

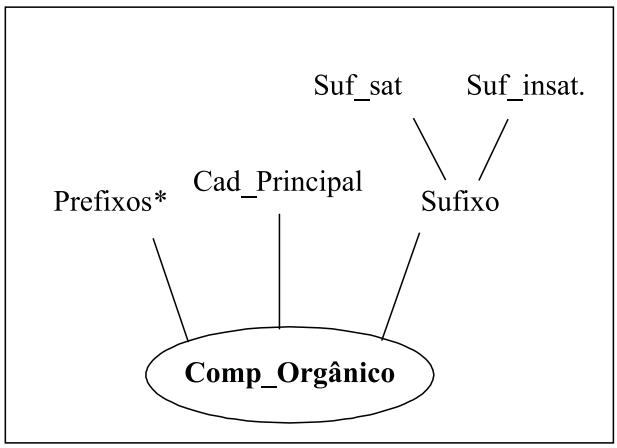

Figura 68: Regra do Nome do Composto Orgânico

Os sufixos indicativos de saturação ou insaturação são formados de acordo com regras, conforme disposto a seguir: para o nome dos compostos orgânicos saturados, a regra ilustrada na figura 69 indica 0 (zero) ou mais prefixos, o nome da cadeia principal saturada, além do sufixo de saturação que corresponde a “ane” (Alcanos).

Já para o nome dos compostos insaturados, há um acréscimo de um novo campo. Esse campo indica a localização da insaturação (ligação dupla ou tripla) na cadeia principal do composto, de acordo com as regras da IUPAC. Portanto, para ter um nome de um composto orgânico insaturado é preciso ter, pelo menos, uma ou mais localizações da insaturação (informação essa indicada pelo sinal (+), mostrada na figura 70), uma cadeia principal insaturada e o sufixo que indica a insaturação (“ene”, “yne” e “adyene”), como

[4]Notação advinda do Combinadores de *Parsers*

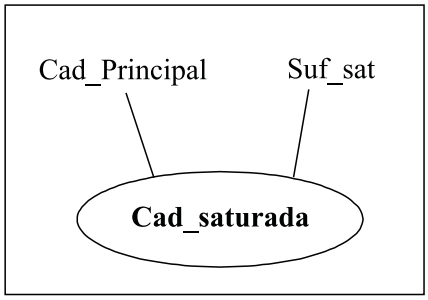

Figura 69: Regra para o Nome de Compostos Saturados

mostra a figura abaixo:

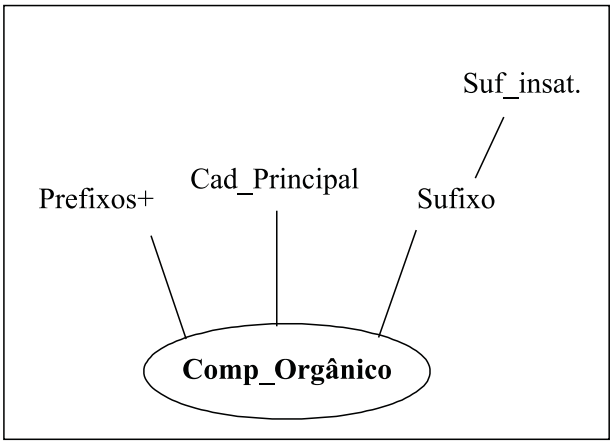

Figura 70: Regra para o Nome de Compostos Insaturados

A regra para compostos insaturados casa-se, em parte, com os nomes dos compostos das funções orgânicas Álcoois e Aldeídos. As diferenciações provêm do seguinte: no caso dos Álcoois, em vez da localização da insaturação, aparece a localização do único componente **OH** (um átomo de oxigênio junto com um átomo de hidrogênio), ligado na cadeia principal do composto químico, como mostra a figura 71, caracterizando o nome da função dos Álcoois pelo sufixo **ol** (conforme visto no capítulo 3). No caso dos Aldeídos, a sua nomeclatura possibilita especificar as posições das insaturações, quando houver, de forma bem similar a nomeclatura dos Alcenos, Alcinos e Alcadienos. Caso contrário, somente haverá ligações simples entre os carbono, dispensando a especificação das posições das insaturações. Tanto em cadeias saturadas quanto em cadeias insaturadas o que vai caracterizar o composto do tipo Aldeído é a presença de átomo de Oxigênio (O) e um átomo de Hidrogênio (H) ligados no mesmo átomo de Carbono pertencente à cadeia carbônica, formando, assim, um composto cujo nome possui o sufixo **al**, definindo um Aldeído. A regra que define o nome de um composto dos Aldeídos é apresentada na figura 72 .

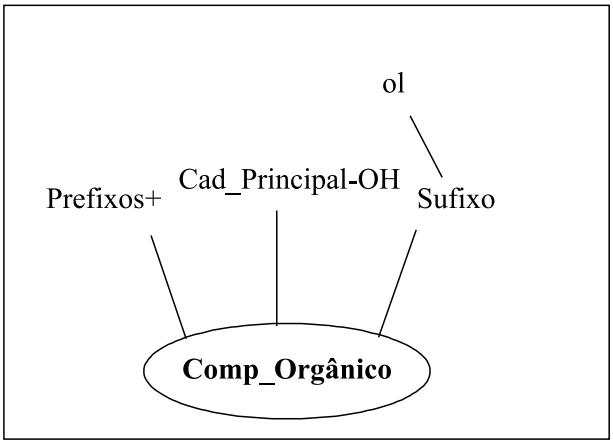

Figura 71: Regra para o Nome de Compostos da Família dos Álcoois

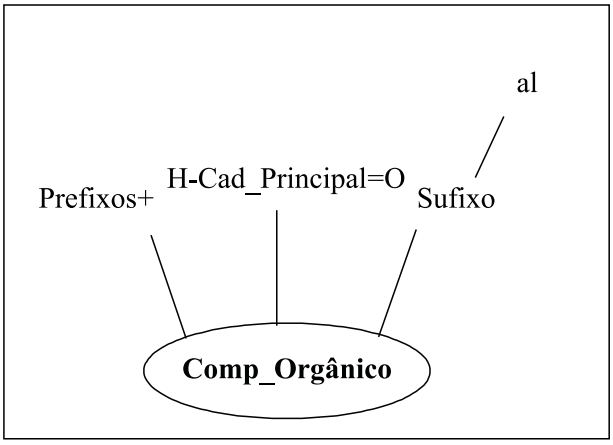

Figura 72: Regra para o Nome de Compostos do Grupo dos Aldeídos

Quando há a presença do prefixo no nome do composto orgânico, esse prefixo pode correspoder a um ou mais multiplicadores (*Rad_mult*), por exemplo, **di***methyl*, que indica dois radicais *methyl* ou, um ou mais radicais (*Radicais*, por exemplo, 3-ethyl, 2-ethyl-3-methyl, etc), ou ambos.

Tanto os multiplicadores quanto os radicais são antecidos por um número o qual indica a posição do átomo de carbono aos quais estão ligados na cadeia principal. (conforme ilustrado na figura 73).

Essas regras foram implementadas utilizando a técnica dos Combinadores de *Parsers*, que exercem o papel de analisador sintático dos componentes do nome do composto de entrada do programa. Além de reconhecer os componentes dos nomes orgânicos, os combinadores servirão também de "alimentador" para a parte semântica do sistema. Essa semântica é reconhecida, no sistema RALQ, pelas chamadas "funções de montagem", as quais realizarão a representação das cadeias químicas que, posteriormente, serão compiladas em códigos *Xymtec*. Para realizar a semântica, as funções de montagem utilizam

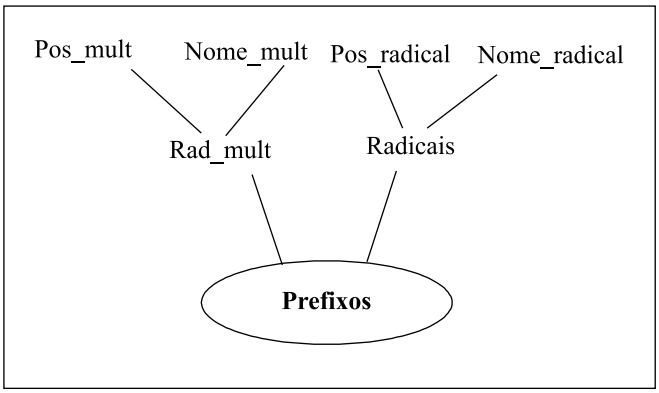

Figura 73: Regras para Nomes de Prefixos

dos dados oriundos das combinações dos *parsers*, ou seja, da análise sintática.

Na próxima subseção (4.3.4) será demostrado com mais detalhe a utilização dos combinadores de *parsers* como elemento fundamental na análise sintática e, também, como um elo de ligação entre a sintática e a semântica do sistema. Em contrapartida, na subseção 4.3.5, a representação da estrutura química dos compostos (semântica do RALQ), manipulada pelas funções de montagem, será verificada em sua totalidade. Na subseção 4.3.6 é mostrada a compilação da representação da estrutura química em códigos *Xymtec* os quais serão armazenados em um arquivo de saída (**teste.tex**).

4.3.4 A Utilização dos Combinadores de Parsers na Análise Sintática dos Compostos

Definidas as regras de formação dos nomes orgânicos e dos tipos, nesta seção será apresentado como a combinação de *Parsers* é útil na análise sintática dos nomes químicos e no reconhecimento das estruturas que darão uma representação semântica aos compostos orgânicos correspondentes a tais nomes. Conforme previsto, a parte semântica será vista em maiores detalhes na próxima subseção, ficando a presente subseção destinada a detalhar a sintaxe.

A eficácia dos combinadores de *parsers* será exemplicada na análise e na representação dos Alcanos e Alcadienos. A escolha dessas funções orgânicas ocorreu devido à natureza da ligação que cada uma possui: saturada, para os Alcanos, e insaturada para os Alcadienos. Os Alcenos ou os Alcinos também poderiam ter sido usados para exemplificar a utilização dos combinadores no tratamento dos compostos insaturados, entretanto, os Alcadienos possuem duas duplas ligações, conseqüentemente, ilustrarão melhor a aplicação da regra de nomes insaturados.

É bom ficar claro que os procedimentos para a análise dos nomes das funções orgânicas presentes neste trabalho variam de uma função para outra em virtude da natureza das ligações e de outros elementos que compõem e caracterizam a cadeia carbônica de cada função orgânica (como, no caso dos Álcoois, a presença do elemento OH na cadeia principal).

Considere o composto **3-methyl-1,2-pentadyene** para ilustrar o fluxo de execução da análise pelos *parsers*. Como os nomes dos compostos estão armazenados no arquivo **teste.pac**, o processo de leitura produzirá um dado do tipo *String*, o qual é modificado para uma lista do tipo *Char* para ser utilizado pela função *chain*, definida abaixo:

```
(7)  chain = alkaneMainChain  <!> alkeneMainChain <!>
             alkyneMainChain  <!> alkadyeneMainChain <!>
             alcoholMainChain <!> aldehydeMainChain;
```

Essa função *chain* utiliza o combinador <!> (mostrado no capítulo 3) para combinar os *parsers* que representam as funções orgânicas implementadas (*alkaneMainChain*, *alkeneMainChain*, *alkyneMainChain*, *alcoholMainChain* e *aldehydeMainChain*) que sejam compatíveis com o nome do composto de entrada. O resultado de tal combinação, quando bem sucedida, é a representação gráfica da estrutura química.

Desse modo, o nome do composto é analisado por cada função que *chain* possui, ou seja, o nome é analisado primeiramente pela função *alkaneMainChain* para ver se é um Alcano. Caso não seja, o nome do composto é analisado pela função *alkeneMainChain* seguinte, a qual realiza o mesmo procedimento da função anterior, só que para os Alcenos, e assim, sucessivamente, até conseguir um valor que satisfaça tanto os *parsers* que analisam as regras de formação dos prefixos, quanto aqueles que analisam as localizações de possíveis insaturações, o nome da cadeia principal e o sufixo de cada função orgânica. Caso o nome de entrada pertence a uma função que não seja reconhecida por nenhum *parser*, o construtor *Unknown* será retornado. No entanto, se o composto é de uma função reconhecida pelo sistema, mas que infrinja a regra da cadeia principal ou a regra dos menores números, definidas no capítulo 3, a situação de ambigüidade é caracterizada. Para resolver essa ambigüidade, o sistema RALQ propõe uma solução, baseada na posição do radical que levou o nome do composto orgânico à situação de ambigüidade, conforme detalhado na seção 4.4.

No caso do exemplo corrente, é a função *alkadyeneMainChain* que consiguirá identificar e representar o composto de entrada.

Além disso, o composto do exemplo possui um prefixo (3-methyl), duas localizações de insaturações (1,2), o radical "pent" que representa a quantidade de carbonos na cadeia principal e o sufixo (adyene[5]). Dessa forma, *alkadyeneMainChain* utilizará um de seus *parsers* componentes para reconhecer a estrutura do nome do composto orgânico. Assim, a regra que possui um prefixo (que é um radical), uma ou mais localização de insaturação, a cadeira principal e o sufixo é codificada pelo *parser withoutMult4* (mostrado logo em seguida), o qual se encaixa à estrutura do nome orgânico do exemplo.

```
(8) withoutMult4 = radicalsAlkadyene <&> \s-> (posLinkDyene) <&>
                   \j->(alkadyeneRadicals <@ (\x-> (mkAlkadyene x s j)))
                   <& alkadyeneFunction;
```

O nome do *parser withoutMult4* é baseado na ausência de multiplicadores no nome do composto de entrada. Como o exemplo não possui prefixo do tipo multiplicadores e, sim, do tipo radical, *withoutMult4* é a única função capaz de reconhecer essa estrutura. O número 4 do *withoutMult4* é somente uma indicação de que esta função pertence a codificação da função orgânica dos Alcadienos. Para cada função orgânica presente neste trabalho, as regras de formação dos nomes do compostos orgânicos são codificadas de forma similar, mudando somente algumas características pertecentes a cada função, por exemplo, os sufixos (pois cada função orgânica possui um sufixo próprio) e algumas cadeias principais (como os Alcadienos, por exemplo, cujas cadeias carbônicas têm, no mínimo, três átomos de carbonos, devido às duas duplas ligações presentes nessa função orgânica).

Essa função *withoutMult4* mostra como a combinação de *parsers* facilita o reconhecimento e a representação dos compostos orgânicos. Em *withoutMult4*, um ou mais radicais são agrupados em uma estrutura de lista do tipo *Branch*, representando todos os prefixos (radicais) presentes no nome do composto analisado. No exemplo dado, seria [(Methyl 3)], em que o inteiro 3 (três), representa a localização desse radical na cadeia principal. Tal resultado é obtido pelo *Parser radicalsAlkadyene*.

Em seguida, o combinador <&> tentará combinar o resultado do *parser* anterior (*radicalsAlkadyene*) com o do próximo (*posLinkDyene*), transferindo o fluxo de execução do programa para este último. Para tanto, o resultado obtido na função *radicalsAlkadyene* é fornecido como argumento ao parâmetro "s" da abstração lâmbda (comentado no capítulo 3), \s->. Isso indica que a lista resultante do *radicalsAlkadyene* será referenciada pela

[5]Nessa implementação o sufixo dos Alcadienos teve um acréscimo da vogal "a", apenas como um artifício de implementação no reconhecimento do sufixo "dyene" que é o sufixo padrão da IUPAC para os Alcadienos.

variável "s". Além disso, o combinador <&> transfere para o *parser posLinkDyene* a lista de diferença formada por "1,2-pentadyene" (ver capítulo 3).

O papel do *parser posLinkDyene* é reconhecer os números que indicam a localização das ramificações (ligação dupla, neste caso), resultando em uma lista do tipo inteiro [1,2] passada como argumento ao parâmetro "j" (por meio do operador <&>). A lista de diferenças é agora, formada por "pentadyene".

Em seguida, o fluxo de programa segue para o *parser* seguinte (*alkadyeneRadicals).* Neste ponto, a análise do nome **3-methyl-1,2-pentadyene** já possui o radical e as localizações das insaturações analizados, ficando somente a cadeia principal e o sufixo a serem reconhecidos e codificados.

Assim sendo, para reconhecer o nome da cadeia principal, a função *alkadyeneRadicals* utiliza um *parser* denominado *radixDyene*, afim de checar se o nome da cadeia principal do composto de entrada é realmente válido para uma cadeia principal dos Alcadienos. No código (6) abaixo, não há a presença do nome para a cadeia principal de dois carbonos interligados (**et**) e nem para uma cadeia simples, de um carbono (**met**). Isso foi feito devido a natureza da ligação dos Alcadienos (duas ligações duplas), necessitando de pelo menos uma cadeia de 3 (três) átomos de carbonos interligados. Caso um desses nomes anteriores ocorra, o composto não será reconhecido.

```
(9)  radixDyene= (token ['prop']) <!> (token ['but']) <!>
                 (token ['pent']);
```

No exemplo acima, a função *alkadyeneRadicals* resultará num nome válido para a cadeia principal do composto orgânico, que possui o tipo primitivo *String*, neste caso *alkadyeneRadicals* (que compõe a parte sintática do RALQ), retorna *"pent"*. Este resultado, por sua vez, é utilizado pela função *mkAlkadyene* que construirá a representação semântica do sistema, através da variável "x" que faz referência ao valor obtido pelo *parser alkadyeneRadicals*, juntamente com os valores referenciados pelas variáveis "s" e "j", através da aplicação do combinador <@, o qual utiliza a Semântica Denotacional na sua execução, vista no capítulo 3.

Assim sendo, concluindo o rastreamento da execução do exemplo, o fluxo de controle passará do *parser alkadyeneRadicals* para o *parser alkadyeneFunction*, ficando a função *mkAlkadyene* esperando o *alkadyeneFunction* reconhecer o sufixo **adyene**, para então, a função *mkAlkadyene* começar a sua execução.

É por meio do combinador <& que o fluxo de controle voltará para a função *mkAlkadyene*, caso a função *alkadyeneFunction* reconheça o sufixo **adyene**. Se ela não reconhecer o sufixo dos Alcadienos, isto quer dizer que o composto não pertence a função orgânica dos Alcadienos, voltando o fluxo de controle para o *parser chain*, que buscará uma outra função orgânica que reconheça o composto de entrada.

A função *mkAlkadyene*, que realizará a chegagem de compatibilidade da cadeia principal, dos possíveis prefixos e das localizações das insaturações (como será observado na subseção 4.3.5), fará a parte da semântica dos Alcadienos, pois o objetivo deste trabalho é mostrar a estrutura química de cada nome do composto da entrada (resolvendo, inclusive, problemas de ambigüidade, conforme apresentado na seção 4.4). Dessa forma, essa função construirá uma representação da estrutura química para o Alcadieno presente no arquivo de entrada do programa. Em seguida, tal estrutura será compilada em comandos *Xymtec*.

Agora, para o exemplo de um composto orgânico saturado (Alcanos), considere o composto orgânico **2,2-dimethyl-4-ethyl-hexane**. Este nome é formado por um prefixo multiplicador (2,2-dimethyl-), um prefixo radical (4-ethyl-), o nome da cadeia principal (hex) e o sufixo (ane), que identifica ser um Alcano. Ele representa um nome válido (de acordo com as regras estabelecidas pela IUPAC) e é reconhecido pela seguinte função:

```
(10) withWithoutMult = partRadical <&> \s-> (alkaneRadical <@ (\x->
                        (mkAlkane x s ))) <& alkaneFunction;
```

A função *withWithoutMult* que compõe a função *alkaneMainChain* codifica uma das possíveis regras para o reconhecimento dos nomes orgânicos para os Alcanos. Os *parsers* que formam a função *withWithoutMult* segue o mesmo padrão de execução mostrado na função (5) acima.

Nesse exemplo, há a presença dos multiplicadores, ou seja, o radical **di**, que indica que a quantidade do radical que está junto ao multiplicador é 2 (dois). No exemplo, 2,2-dimethyl equivale a 2-methyl-2-methyl. No caso do exemplo **2,2-dimethyl-4-ethyl-hexane**, tanto os prefixos multiplicadores quanto os prefixos radicais são tratatos pelo *parser partRadical*, que retornará uma lista do tipo *Branch*, contendo todos os prefixos (multiplicadores e radicais) em uma só lista. No exemplo do composto dos Alcanos o *parser partRadical* resultará em [(Methyl 2), (Methyl 2), (Ethyl 4)].

Como os Alcanos são uma funções orgânicas saturadas, não há a necessidade de reconhecimento de valores antes do nome da cadeia principal, pois esses valores só estarão presentes nas cadeias insaturadas, nos Álcoois e nos Aldeídos (de acordo com o capítulo

3). Então, como os radicais do composto já foram analisados, a função *alkaneRadical*será executada com o intuito de se reconhecer o nome da cadeia principal (que no caso dos Alcanos não haverá nenhuma restrição quanto ao número de carbonos que ela conterá).

Assim, o combinador <@ conduzirá para a aplicação de mkAlkene à variável "s" (que referenciará a lista dos radicais, por meio do combinador <&>), à variável "x" (que vai referenciar o nome da cadeia principal, do tipo *String*). Essa função tem a mesma especificação da função *mkAlkadyene* acima, tendo como diferença a quantidade de parâmetros que cada uma possui. Pois, a função *mkAlkane* possui: a função parâmetros (o primeiro, que indica o nome da cadeia principal, do tipo *String*, e o segundo, que indica a lista dos prefixos do tipo *Branch*). Já a função *mkAlkadyene* possui três parâmetros (os dois primeiros são equivalentes aos da função *mkAlkane* e o terceiro é uma lista de inteiros que indica a localização das ramificações).

Por último, os sufixos dos Alcanos são reconhecidos do mesmo modo que os dos Alcadienos e, os das demais funções, mudando apenas o valor em cada função: no caso dos Alcanos, o seu sufixo é reconhecido se a função ou *parser alkaneFunction* detectar o sufixo **ane**, que identifica que o composto é um Alcano. Uma vez reconhecido, o combinador <& transferirá o fluxo de execução para a função *mkAlkane*.

Em resumo, todas as possibilidades de formação de nomes de compostos orgânicos válidos foram implementados de forma análoga, para todas as funções químicas tratadas nesse trabalho. Porém, cada função química teve sua codificação separada, de modo a respeitar as características de cada uma e, também, a tornar essa codificação mais compreensiva no tratamento semântico e no processamento das possíveis ambigüidades.

O uso dos combinadores de *parsers* nessa parte sintática do trabalho facilitou a programação da análise dos nomes químicos na ordem em que se lê qualquer nome orgânico e possibilitou a ligação da parte sintática com a semântica do sistema, fazendo, assim, uma programação mais direta e intuitiva.

Na subseção seguinte, será mostrado como a representação da estrutura química dos compostos orgânicos é construída no RALQ.

4.3.5 A Representação da Análise Semântica

Conforme dito na seção 4.1, nome de composto correto corresponde a um nome que está de acordo com a nomenclatura oficial [Usberco e Salvador 2001] e que representa um composto que existe na química; nome de composto inadequado corresponde a um nome

que, apesar de não estar de acordo com a nomenclatura oficial representa um composto que existe na química.

As restrições semânticas de um composto indicam com quais prefixos e sufixos eles podem combinar-se e em que locais da cadeia principal tais combinações podem ser efetuadas.

O processo de análise semântica dos compostos consiste em verificar se os parâmetros detectados na Análise Sintática (isto é, prefixos com suas respectivas posições na cadeia principal, cadeia principal, sufixo e locais das eventuais insaturações) são passíveis de se combinarem de modo a satisfazer as restrições semânticas do composto da cadeia principal (ou seja, do composto formado pela cadeia principal seguida do sufixo).

Se a Análise Semântica detectar que o nome do composto de entrada é inadequado, o programa o corrige, de modo a convertê-lo no nome correto, e retorna o desenho da estrutura química do composto. Neste caso, o programa terá resolvido uma ambigüidade entre o nome inadequado e o nome correto de um mesmo composto.

Para nomes corretos dos compostos de entrada, a Análise Semântica retorna os desenhos de suas estruturas químicas.

Para os nomes de entrada incorretos ou não representados na Base de Dados Semântica (isto é, compostos cujas funções não tenham sido implementadas no programa), é retornada a mensagem "*UNKNOWN*" indicando o fato.

No RALQ, a análise semântica é desencadeada pela atuação do operador <&, que comanda a aplicação dos *parsers* semânticos aos resultados obtidos pelos *parsers* sintáticos.

Para ilustrar tal processo, considerar os exemplos tratados na seção 4.3.5. Lá, as funções *mkAlkadyene* e *mkAlkane* são responsáveis pela parte semântica da aplicação, ou seja, *mkAlkadyene* gerará a representação da estrutura química para os compostos pertencentes a função orgânica Alcadieno e, da mesma forma, a função *mkAlkane* produzirá a representação da estrutura química para os Alcanos.

De modo análogo, as demais funções orgânicas aqui tratadas têm as suas representações estruturais da molécula orgânica geradas pelas seguintes funções implementadas: *mkAlkene*, que fará a representação para os Alcenos; *mkAlkyne*, para os compostos da família dos Alcinos; *mkAlcohol* para os Álcoois e, por fim, *mkAldhyde* para os Aldeídos. Todas essas funções utilizadas para construir a representação estrutural da química orgânica serão denominadas de "Funções de Montagem".

A implementação dessas representações estruturais das funções orgânicas é baseada nos possíveis argumentos aos quais nome do composto orgânico da cadeia principal pode ser aplicado. Esses argumentos correspodem aos prefixos (radicais e multiplicadores), com suas respectivas localizações da cadeia principal, e às localizações das insaturações. Então, para cada composto da cadeia principal pertecente a uma determinada função orgânica, haverá uma possível regra, dentro das funções de montagem, que tentará satisfazer as restrições semânticas associadas àquela função orgânica, de modo a possibilitar a montagem da estrutura química correspondente ao composto de entrada.

Assim sendo, uma vez descoberto o tipo do composto de entrada[6] (Alcanos, Alcenos, Alcinos etc), o programa chamará uma das Funções de Montagem correspondente a esse tipo para realizar a parte semântica da aplicação.

Logo abaixo estarão alguns exemplos que ajudarão a compreender o mecanismo de obtenção da estrutura química dos compostos orgânicos através das funções de montagem.

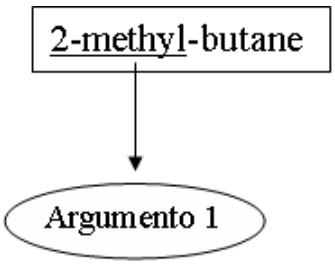

Figura 74: Composto com apenas 1 Argumento.

No primeiro exemplo da figura 74, o nome do composto possui apenas um argumento do tipo radical. Como esse composto é do grupo dos Alcanos, de acordo com o sufixo **ane**, e possui uma cadeia principal de 4 (quatro) carbonos (definida pelo **but**), a função de montagem *mkAlkane* deve ter uma regra que reconheça essa especificação no intuito de representar a estrutura química da mesma. Neste caso, a melhor regra, dentre aquelas implementadas para fazer a representação química dos Alcanos, que se encaixa perfeitamente aos elementos que compõe o nome orgânico com o objetivo de construir uma representação da estrutura química desse composto é a seguinte:

```
(11) mkAlkane "but" [x] | ((getPos x)==2) = Alkane [C(H, H, H, SL),
                                  C(H, SL, x, SL), C(H, SL, H, SL),
                                  C(H, SL, H, H)];
```

[6]Procedimento obtido pelo processamento sintático, através dos Cominadores de *Parsers*

onde a função de montagem *mkAlkane* tem como primeiro parâmetro a *string* **but**, indicando que a função trata cadeias principais que possuem 4 (quatro) carbonos interligados; o segundo parâmetro da função *mkAlkane* é uma lista que abrigará os argumentos do nome do composto, ou seja, uma lista do tipo *Branch*. No caso do exemplo, como o nome do composto orgânico só contém um argumento, a variável "x" referenciará o argumento (Metil 2).

Como o composto orgânico possui um cadeia principal com 4 (quatro) carbonos interligados, então o construtor *Alkane*, que faz parte do tipo algébrico *OrgType* definido anteriormente, possui uma lista de *Atom* como parâmetro. Em contrapartida, o tipo *Atom* possui um construtor C com uma tupla de quatro parâmetros do tipo *Branch* como argumento, sendo que estes parâmetros representam as ligações do átomo de Carbono. Assim sendo, a estrutura química do nome do composto desse exemplo (**2-methyl-butane**) será representada por uma lista de 4 átomos de carbonos do tipo *Atom* e com suas possíveis ligações.

Como falado na subseção anterior, o inteiro 2 (dois) que faz parte do argumento (Methyl 2) tem a função de dizer em qual carbono da cadeia principal será colocado este prefixo. Para que o programa saiba que este inteiro "dois" indica a posição em que deve ficar a ramificação, um teste booleano de equivalência [Souza 2002] é realizado antes, entre a função *getPos*, que retorna o inteiro que está presente no argumento, e o número 2 (dois). Se a função *getPos* retornar o inteiro 2, isso quer dizer que a ramificação (metil) está no segundo carbono da cadeia, conforme mostrado na codificação 12, onde a variável "x" (que referencia o argumento (Metil 2)) está na representação da ligação 3 do carbono de número 2, de acordo com o nome do composto de entrada.

Agora, se a função *getPos* retornar um valor diferente do inteiro 2, o programa vai verificar outras alternativs de regras que envolvam os mesmos componentes orgânicos.

A representação da semântica do composto orgânico **2-methyl-butane** será:

```
(12)  Alkane [(C(H, H, H, SL)),(C(H, SL, (Methyl 2), SL)),
              (C(H, SL,H, SL)),(C(H, SL, H, H))],
```

em que (C(H, H, H, SL)) representa o carbono de número 1 da cadeia principal, onde as ligações 1, 2, 3 são feitas com o átomo de Hidrogênio e a ligação 4 é uma ligação simples com o próximo carbono da cadeia (de acordo com o capítulo 3). O segundo carbono da cadeia principal é representado por (C(H, SL, (Methyl 2), SL)), sendo a primeira ligação deste carbono com o átomo de Hidrogênio, a segunda uma ligação simples com o carbono

1, a terceira com a ramificação *Methyl* deste carbono 2 e, a quarta ligação, uma ligação simples com o carbono 3.

Análise análoga pode ser feita para o terceiro e quarto átomos de carbono da cadeia principal.

Outro exemplo, para a obtenção da estrutura química dos compostos orgânicos através das funções de montagem com a presença dos multiplicadores é mostrado na figura 75.

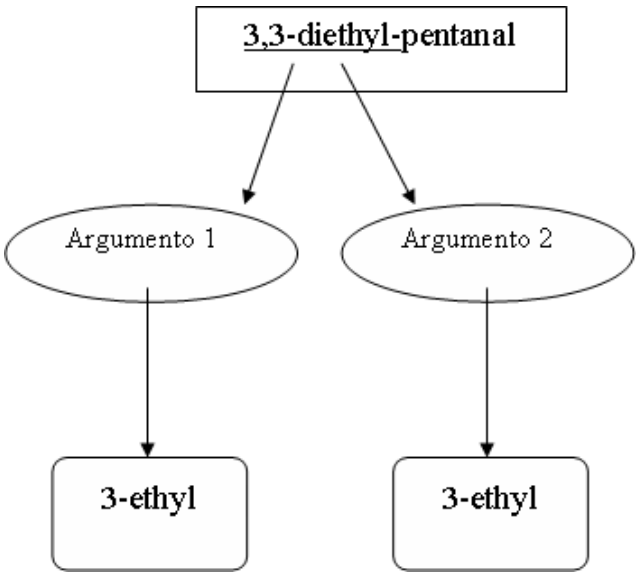

Figura 75: Nome do composto com argumentos Multiplicadores

Nesse exemplo, há a presença do prefixo do tipo multiplicador (**3,3-diethyl**). Os multiplicadores devem ser trabalhados de forma a gerar um tipo compatível com aquele reconhecido pelas funções de montagem. Para tanto, a implementação dos multiplicadores é baseada no significado dos seus termos (di, tri, tetra, etc). Por exemplo, o multiplicador **di**ethyl significa 2 (dois) prefixos *ethyl* (do tipo radical), **tri**methyl significa 3 (três) prefixos *methyl* (do tipo radical) e, assim por diante. Devido a isso, as funções *combinePosicao*, que tratam, por exemplo, os multiplicadores Dimethyl e Octamethyl, correspondem a:

```
(13) combinePosicao [i,j] (Diethyl) =[(Ethyl i), (Ethyl j)];

     combinePosicao [i,j,l,m,n,o,p,t] (Octamethyl) = [(Methyl i),
                 (Methyl j), (Methyl l), (Methyl m),
                 (Methyl n),(Methyl o), (Methyl p), (Methyl t)];
```

A *combinePosicao* é uma função utilizada na parte sintática do programa, mas fun-

damental na parte semântica ao traduzir os multiplicadores em dados utilizáveis pelas funções de montagem.

A função *combinePosicao* tem como entrada dois parâmetros, uma lista de inteiros e um *Branch* que representa os multiplicadores. A lista de inteiros serve para indicar a localização na cadeia principal das ramificações onde se ligarão os radicais dos multiplicadores. Identificado o tipo de multiplicador, a função *combinePosicao* irá produzir uma lista do tipo *Branch* com a quantidade de argumentos oriundos dos multiplicadores, combinando os inteiros da lista do primeiro parâmetro com os argumentos presentes nos construtores da lista formada. Por exemplo, no caso da figura 75, a função *combinePosicao* geraria a lista de radicais:

```
(14) [(Ethyl 3), (Ethyl 3)];
```

Tal lista será retornada ao *parser* que ativou *combinePosicao*. Este, por sua vez, inserirá ou não tais radicais na cadeia principal que representa em função de eles satisfazerem ou não as restrições semânticas do composto da referida cadeia principal.

A lista resultante da função *combinePosicao* retornará ao *parser* que a invocou. No caso do atual exemplo, a lista produzida pela função *combinePosicao* será argumento da função de montagem *mkAldehyde*, pois, de acordo com o sufixo (**al**) e com o nome orgânico da entrada, esse composto é da família dos Aldeídos.

Como o nome desse composto produziu dois argumentos, a cadeia principal possui 5 (cinco) átomos de carbonos interligados, o composto é do tipo Aldeído e não possui ligação insaturada, a regra que reconhecerá os requisitos desse composto é a seguinte:

```
(15)  mkAldehyde "pent" [x,y][]|((getPos x)==3)&&((getPos y)==3) =
                                Aldehyde [C(H, H, H, SL), C(H, SL, H, SL),
                                          C(y, SL, x, SL),C(H, SL, H, SL),
                                          C(SL, H, O, WL)];
```

Nesta regra (15), a função de montagem utilizará a *string* “pent” (que identifica a cadeia principal com 5 (cinco) carbonos), a lista de argumentos do tipo *Branch* (obtida diretamente da função *combinePosicao* mostrada acima) e um novo parâmetro que, até aqui, devido às características dos compostos então analisados, não precisou ser utilizado. Este parâmetro é uma lista de inteiros que indicam onde estão as ligações insaturadas presentes na cadeia principal do composto orgânico, pois, de acordo com o capítulo 3 os

Aldeídos, Alcenos, Alcinos e Alcadienos podem conter insaturações na cadeia principal. Mas como o nome do composto do exemplo acima não contém nenhuma insaturação, a referida lista é vazia ([ ]).

Além disso, pode-se observar que a regra (15) vale para os casos em que os radicais do prefixo estejam ambos na posição 3 (conforme chamada a "*getPos*")

Com isso, a representação da semântica do composto orgânico, **3,3-diethyl-pentanal**, será:

```
(16)  Aldehyde [(C(H, H, H, SL)), (C(H, SL, H, SL)),
                (C((Ethyl 3), SL, (Ethyl 3), SL)),
                (C(H, SL, H, SL)),(C(SL, H, O, WL))];
```

Que pode ser interpretado de modo análogo ao apresentado há pouco.

É bom salientar que o último carbono dessa cadeia principal de 5 (cinco) átomos de carbono do composto **3,3-diethyl-pentanal** é uma representação que identifica semanticamente o grupos dos Aldeídos. A primeira ligação deste carbono é executada com o carbono anterior dessa cadeia principal, por meio de uma ligação simples. A sua ligação seguinte é realizada com o átomo de Hidrogênio (H). A terceira ligação é feita com o átomo de Oxigênio (O), o qual necessita de duas ligações para completar a sua camada de valência (como mostra o capítulo 3). Devido a isso, a última representação da ligação desse carbono é considerada nula, ou seja, representada pelo construtor WL (*Without Linking* - Sem ligação), que representará a ligação nula em todos os carbonos que já estiverem completados a sua camada de valência (formada com quatro elétrons), de acordo com o capítulo 3.

Até agora, foram considerados exemplos com prefixos do tipo radicais e multiplicadores em compostos que possuem cadeias saturadas. É interessante, agora, ilustrar o exemplo de um composto de cadeia insaturada. O exemplo que mostra o uso das localizações das insaturações é dado pela figura 76.

De acordo com a figura 76, a Análise Sintática do composto vai detectar dois argumentos: um prefixo (**3-ethyl**) e o outro a localização das insaturações (**1,2**). O nome da cadeia principal do composto é "pent" e o sufixo é **adyene**, caracterizando o grupo dos Alcadienos.

Desse modo, o fluxo do programa irá para a função de montagem *mkAldyene* e, conseqüentemente a regra implementada que reconhecerá todos os componentes do nome

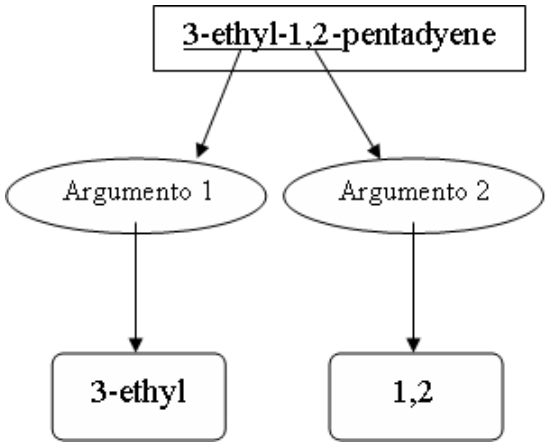

Figura 76: Argumentos do tipo Prefixo e Localizações das Insaturações

orgânico **3-ethyl-1,2-pentadyene** será executada, como mostra a regra (18), abaixo:

```
(18) mkAlkadyene "pent" [x][y,z] | ((getPos x)==3) && (y==1) &&
                                   (z==2) = Alkadyene [C(WL, H, H, DL),
                                   C(WL, DL, WL, DL), C(WL, DL, x, SL),
                                   C(H, SL, H, SL), C(H, SL, H, H)];
```

A regra (18), tem algumas diferenças com relação aos outros códigos das funções de montagem vistas anteriormente. A primeira é com relação ao terceiro parâmetro da função de montagem *mkAlkadyene* que trata a presença de insaturações no composto. A segunda diferença é a inserção do construtor DL na representação das ligações tuplas entre carbonos da cadeia principal.

Os valores das variávies "y" e "z" do terceiro argumento representam as localizações das ligações tuplas que estão presente na estrutura química do composto.

getPos checa se, de fato, o radical do prefixo recebido no argumento 1 (um) está com um rótulo para se ligar ao terceiro carbono da cadeia principal e se as duas insaturações recebidas estão com rótulos para se ligarem ao primeiro e segundo carbonos da mesma. Caso isto não ocorra, o programa vai tentar uma outra regra disponível.

A outra diferença mencionada acima é a inserção de outros construtores que indicam as insaturações na representação da estrutura química dos compostos orgânicos. Por exemplo, o carbono 2 (dois) (da regra 18), representado por (**C(WL, DL, WL, DL)**), não possui conexão alguma na primeira e nem na terceira ligações (**WL**); porém, apresenta duas ligações duplas: uma com o carbono 1 (um) e outra com o 3 (três). Já o radical *ethyl* se ligará na terceira ligação do terceiro carbono (marcado por x).

A representação final do composto orgânico **3-ethyl-1,2-pentadyene** é mostrado no código logo abaixo:

```
(19)  Alkadyene [(C(WL, H, H, DL)), (C(WL, DL, WL, DL)),
                 (C(WL, DL, (Ethyl 3), SL)),(C(H, SL, H, SL)),
                 (C(H, SL, H, H))];
```

As representações semânticas, ou seja, as representações das estruturas químicas dos nomes dos compostos orgânicos, foram utilizadas para tornar o programa mais legível com relação ao domínio da química orgânica, proporcionar facilidade na programação das restrições advindas das regras estabelecidas pela IUPAC e obter um processamento direto para os códigos *Xymtec*.

As regras estabelecidas pela IUPAC, como a regra da cadeia e a regra dos menores números, são restrições que serão vistas na parte que tratará da ambigüidade (seção 4.4), pois essas restrições estão diretamente ligadas ao fato de um nome ter ou não ambigüidade.

A próxima seção introduz o processamento para o código *Xymtec*, que corresponde à etapa final do processamento do RALQ.

4.3.6 O uso da Codificação *Xymtec*

A codificação da semântica gerada na subseção anterior em comandos *Xymtec* tem o objetivo de mostrar o desenho das estruturas químicas produzidas a partir do processamento dos nomes orgânicos provenientes do arquivo **teste.pac**.

Montada a estrutura de comandos *Xymtec* para cada nome do composto orgânico, o próximo passo da aplicação é salvar essa estrutura num arquivo denominado **teste.tex**, a ser processado no ambiente Latex.

Na verdade, o que o sistema efetua uma compilação da estrutura semântica produzida pelo Analisador Semântico para comandos *Xymtec*, a qual, de certa forma é trabalhosa, devido a necessidade da implementação de códigos *Xymtec* para cada situação de ligação que os átomos interligados na cadeia principal possuem. Há também o problema da ligação das ramificações, em que todas as possíveis ramificações tem que aparecer no desenho das estruturas químicas ligadas corretamente junto ao carbono correspondente, algo não tão simples, mas que foi realizado neste trabalho. Entretanto, a grande vantagem de se usar a codificação *Xymtec* é a qualidade dos desenhos gerados, proporcionados pelo ambiente

Latex, possibilitando assim a visualização perfeita dos desenhos das estruturas químicas dos compostos orgânicos, como uma forma mais prática de saída para o sistema RALQ

A montagem da estrutura de comandos *Xymtec* começa com a introdução de um comando de bloco, o qual vai delimitar a figura (o desenho da estrutura química), num texto gerado pelo Latex, como mostra o capítulo 3. Para isso, a implementação da aplicação utiliza duas marcações do tipo primitivo *String*, para representar essa delimitação que o Latex reconhece como o início e o fim de uma figura. Tais marcações são utilizadas na função *getFormula* com a finalidade de gerar os comandos *Xymtec* que modelarão o desenho da estrutura química do composto orgânico. Para tanto, *getFormula* utiliza o resultado da função *genMainChain*, isto é, uma codificação em *Xymtec* para cada representação de átomo de carbono recebida como argumento.

A título de exemplo, a representação estrutural produzida pelo analisador semântico para o "metano" (*metane*) será compilada para o seguinte código *Xymtec*:

```
(20)    \begin{picture}(1200, 600)(0,0)
        \put(0,0){\tetrahedral{0==C; 1==H; 2==H; 3==H; 4==H}}
        \end{picture}
```

Este código (20) será salvo no arquivo **teste.tex** e, ao ser executado pelo Latex, produz a seguinte figura:

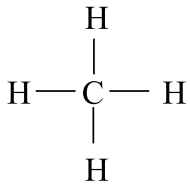

Figura 77: Metano

A biblioteca *Xymtec* precisou ser adaptada de forma especial para que o RALQ conseguisse lidar com os prefixos, pois, a biblioteca original não possui um comando que faça a conexão das ramificações com as ligações dos átomos de carbono da cadeia principal nas localizações corretas. Devido a isso, houve a necessidade de tratar os prefixos para que os mesmos pudessem ser ligados aos carbonos. Isso foi conseguido com a manipulação dos valores das coordenadas do comando "\put" do Latex (ver código (20)) de modo a atender as especificidades de cada tipo de prefixo (*ethyl*, *methyl* etc).

Os próximos exemplos mostram a saída da aplicação e os respectivos desenhos que esses comandos *Xymtec* proporcionam quando executados no Latex, para os nomes dos

compostos orgânicos subseqüentes: **2,3,4,5-tetramethyl-hexane**, **2,3,4-trimethyl-2-pentanal**, **2,2-dimethyl-1-pentanol** e **1-propyne**.

Para o Alcano **2,3,4,5-tetramethyl-hexane**, o código *Xymtec* gerado e a estrutura química são:

```
(21)
\begin{picture}(1200, 600)(0,0)
\put(0,0){\tetrahedral{0==C;1==H;2==H;3==H;4==}}
\put(240,0){\tetrahedral{0==C;1==H;2==;3==
                         \put(-260,-355){\tetrahedral{0==CH$_3$;1==}};4==}}
\put(480,0){\tetrahedral{0==C;1==H;2==;3==
                         \put(-260,-355){\tetrahedral{0==CH$_3$;1==}};4==}}
\put(720,0){\tetrahedral{0==C;1==H;2==;3==
                         \put(-260,-355){\tetrahedral{0==CH$_3$;1==}};4==}}
\put(960,0){\tetrahedral{0==C;1==H;2==;3==
                         \put(-260,-355){\tetrahedral{0==CH$_3$;1==}};4==}}
\put(1200,0){\tetrahedral{0==C;1==H;2==;3==H;4==H}}
\end{picture}
```

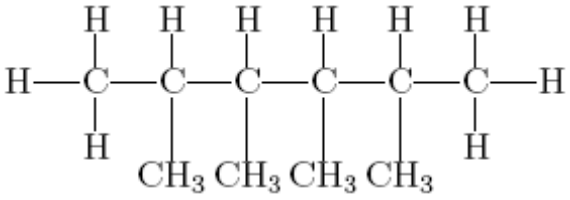

Figura 78: 2,3,4,5-tetramethyl-hexane

O nome do composto orgânico da família dos Aldeídos **2,3,4-trimethyl-2-pentanal**, possui o código *Xymtec* e o desenho da estrutura química da seguinte forma:

```
(22)
\begin{picture}(1200, 600)(0,0)
\put(0,0){\tetrahedral{0==C;1==H;2==H;3==H;4==}}
\put(240,0){\tetrahedral{0==C;1==H;2==;3==
                         \put(-260,-355){\tetrahedral{0==CH$_3$;1==}};4==}}
\put(480,0){\tetrahedral{0==C;2==;3==
                         \put(-260,-355){\tetrahedral{0==CH$_3$;1==}};4D==}}
```

```
\put(720,0){\tetrahedral{0==C;2D==;3==
                        \put(-260,-355){\tetrahedral{0==CH$_3$;1==}};4==}}
\put(960,0){\Rtrigonal{0==C;1==;2==H;3D==O}}
\end{picture}
```

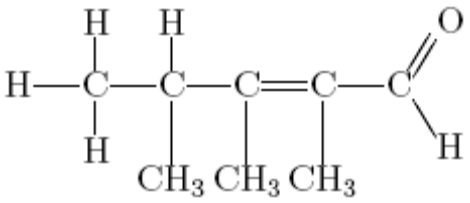

Figura 79: 2,3,4-trimethyl-2-pentanal

Agora, para o nome do composto orgânico do grupo dos Álcoois **2,2-dimethyl-1-pentanol**, o código *Xymetc* produzido e o desenho da estrutura química formada são:

```
(23)
\begin{picture}(1200, 600)(0,0)
\put(0,0){\tetrahedral{0==C;1==OH;2==H;3==H;4==}}
\put(240,0){\tetrahedral{0==C;1==
                    \put(-260,-170){\tetrahedral{0==CH$_3$;3==}};2==;3==
                         \put(-260,-355){\tetrahedral{0==CH$_3$;1==}};4==}}
\put(480,0){\tetrahedral{0==C;1==H;2==;3==H;4==}}
\put(720,0){\tetrahedral{0==C;1==H;2==;3==H;4==}}
\put(960,0){\tetrahedral{0==C;1==H;2==;3==H;4==H}}
\end{picture}
```

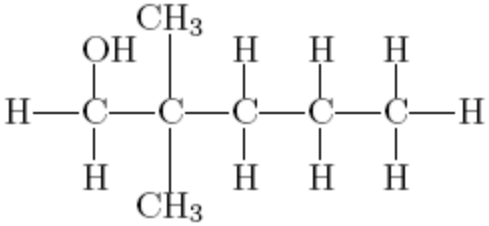

Figura 80: 2,2-dimethyl-1-pentanol

E por fim, o composto **1-propyne** que pertence a função orgânica dos Alcinos, possui o código *Xymtec* e a estrutura química, mostrados abaixo:

```
(24)
\begin{picture}(1200, 600)(0,0)
\put(0,0){\tetrahedral{0==C;3==H;4T==}}
\put(240,0){\tetrahedral{0==C;2T==;4==}}
\put(480,0){\tetrahedral{0==C;1==H;2==;3==H;4==H}}
\end{picture}
```

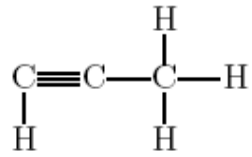

Figura 81: 1-propyne

4.4 O RALQ Visto sob a Ótica da TLG no Processo de Desambiguação Lexical

Conforme já introduzido, a ambigüidade na química orgânica pode acontecer devido a imprecisões na aplicação das regras da cadeia principal do composto e dos menores números, ambas definidas no capítulo 3.

Sempre que tais regras forem aplicadas com exatidão a um composto químico que se queira nomear, o nome produzido estará de acordo com a nomenclatura oficial da Química, ou seja, será um nome correto. Quando essas regras forem ignoradas ou aplicadas com descuido, duas coisas podem ocorrer: pode ser gerado um nome inadequado ou um nome incorreto de um composto que não existe (conforme visto na seção 4.3.5). Caso o nome seja inadequado, haverá uma ambigüidade entre ele e o nome correto, cabendo ao RALQ detectar a ambigüidade e alterar o nome inadequado de modo a torná-lo correto.

No contexto do RALQ, os itens léxicos são aqueles detectados na Análise Sintática pelos combinadores de *Parser*. Logo, as ambigüidades acima citadas referem-se à imprecisão na nomenclatura do itens léxicos, seja imprecisão ligada aos nomes das ramificações e cadeia principal, seja imprecisão na posição em que essas ramificações se ligam a esta cadeia ou, ainda, imprecisão na localização das eventuais insaturações presentes nos compostos. Como o RALQ detecta e corrige tais imprecisões, isso possibilitará a correta geração da estrutura química correta durante a Análise Semântica.

O processo de análise, detecção e resolução de ambigüidades do RALQ pode ser formalizado a luz da Teoria Léxico Gerativo (TLG).

Conforme visto no capítulo 3, a TLG trabalha em cima dos itens léxicos, criando estruturas lexicas (ou Qualia Geral) para analisar cada item. A combinação desses itens é guiada pelos seus respectivos tipos.

No domínio da química orgânica não vai ser diferente, só que em vez de lidar com verbos, substantivos e adjetivos, a Teoria Léxico Gerativo vai trabalhar com prefixos (radicais e/ou multiplicadores), sufixos e cadeia principal, ou seja, para cada componente do nome orgânico haverá uma estrutura léxica que o representará. Tanto quanto na TLG, a combinação de tais estruturas será guiada pelos seus tipos.

A seguir, através de exemplos, será mostrado como a implementação do RALQ apresentada na seção anterior pode ser enquadrada no formalismo da TLG. Para tanto, considere-se a análise do composto **2-methyl-3-ethyl-1-pentene**.

Como o sufixo é um componente de um nome orgânico, há a necessidade de se criar uma estrutura lexica para ele. Tal componente indicará a função orgânica do composto **2-methyl-3-ethyl-1-pentene**, como mostra a figura 82, abaixo:

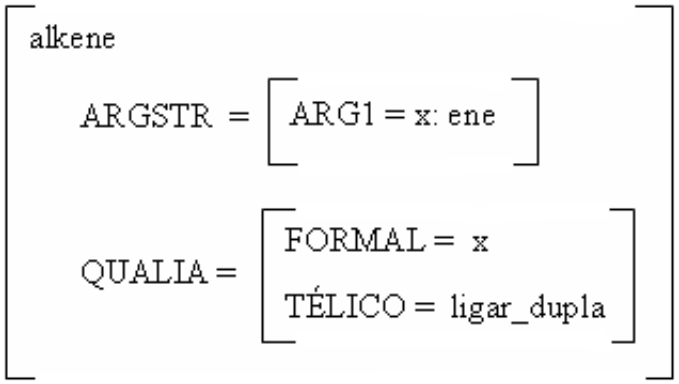

Figura 82: Estrutura Lexica para os *Alcenos*.

De acordo com a figura 82, todo composto da família dos Alcenos (item léxico "alkene") vai possuir um argumento do tipo "ene" (neste caso o sufixo), o qual vai distinguir essa função química das outras (conforme está especificado no papel Formal). O objetivo de tal item léxico é efetuar uma ligação dupla na cadeia principal de uma determinada estrutura química, fato indicado no papel Télico. Conforme visto anteriormente, tal estrutura lexica está implementada no RALQ por meio de um *parser alekeneFunction* que reconhecerá o tipo "ene".

A estrutura lexica para o termo "pent" está representada na figura 83, onde a Estrutura

de Evento (*EVENTSTR*) o classifica como um evento do tipo processo uma vez que será aplicado, na montagem de uma cadeia principal ou de um radical.

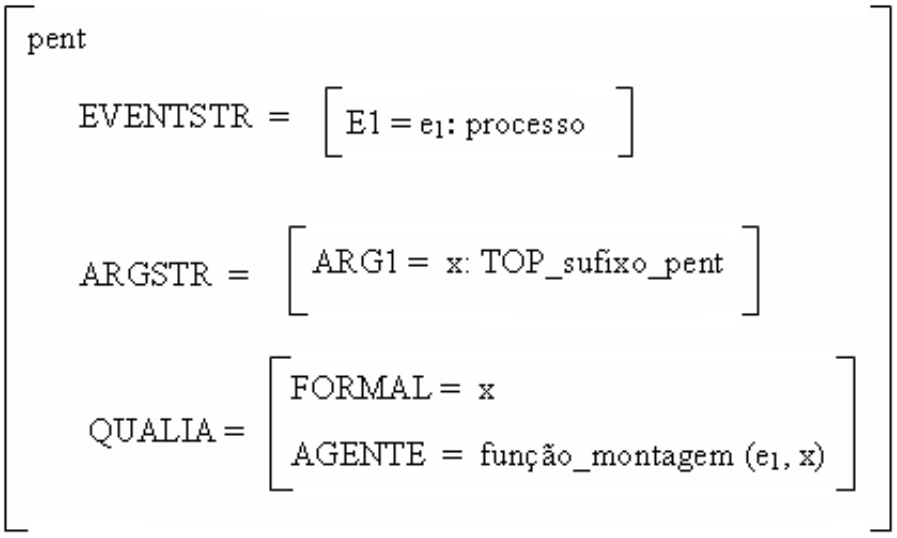

Figura 83: Estrutura Lexica para *pent*.

A estrutura de argumento (*ARGSTR*) da figura 83, possui um argumento "x" do tipo *TOP_sufixo*. Esse tipo é uma disjunção de todos os sufixos possíveis aos quais "pent" pode ser aplicado, ou seja, *TOP_sufixo_pent* pode ser: "hyl" (indica um radical), "ane" (indica um Alcano), "ene" (indica Alceno), "yne" (indica Alcino), "adyne" (indica Alcadieno), "ol" (indica Álcoois) ou "al" (indica Aldeído). Conseqüentemente, o termo "pent" pode ser um radical ou uma cadeia principal para alguma função química, dependendo do tipo do argumento ao qual será aplicado. Note que *TOP* é o LUB (*Lower Upper Bound*), do conjunto de todos os tipos aos quais "pent" pode ser aplicado, conforme definido no capítulo 3.

A estrutura Qualia de "pent" possui no papel Formal o valor "x" com o tipo *top_sufixo_pent*, ou seja, o que vai definir a forma do termo "pent" é o seu argumento. O radical ou a cadeia principal formada pelo termo "pent" surge por meio do papel Agente através da função de montagem (definida na subseção 4.3.5), cujo parâmetro será um sufixo.

Então, para especificar o termo "pent" em uma cadeia principal para um composto dos alcenos, é necessário aplicá-lo a um argumento do tipo "alkene". Como "alkene" é do tipo "ene" que, por sua vez, faz parte da disjunção que define TOP, "pent" pode tê-lo como argumento. Neste caso, diz-se que os dois itens léxicos são **Unificáveis**, pois apresentam tipos compatíveis. O resultado da aplicação corresponde ao item léxico "pentene", cujo tipo é um GLB ([Copestake 1992]) entre o tipo TOP de "pent" e o tipo "ene" de "alkene", conforme mostrado na figura 84.

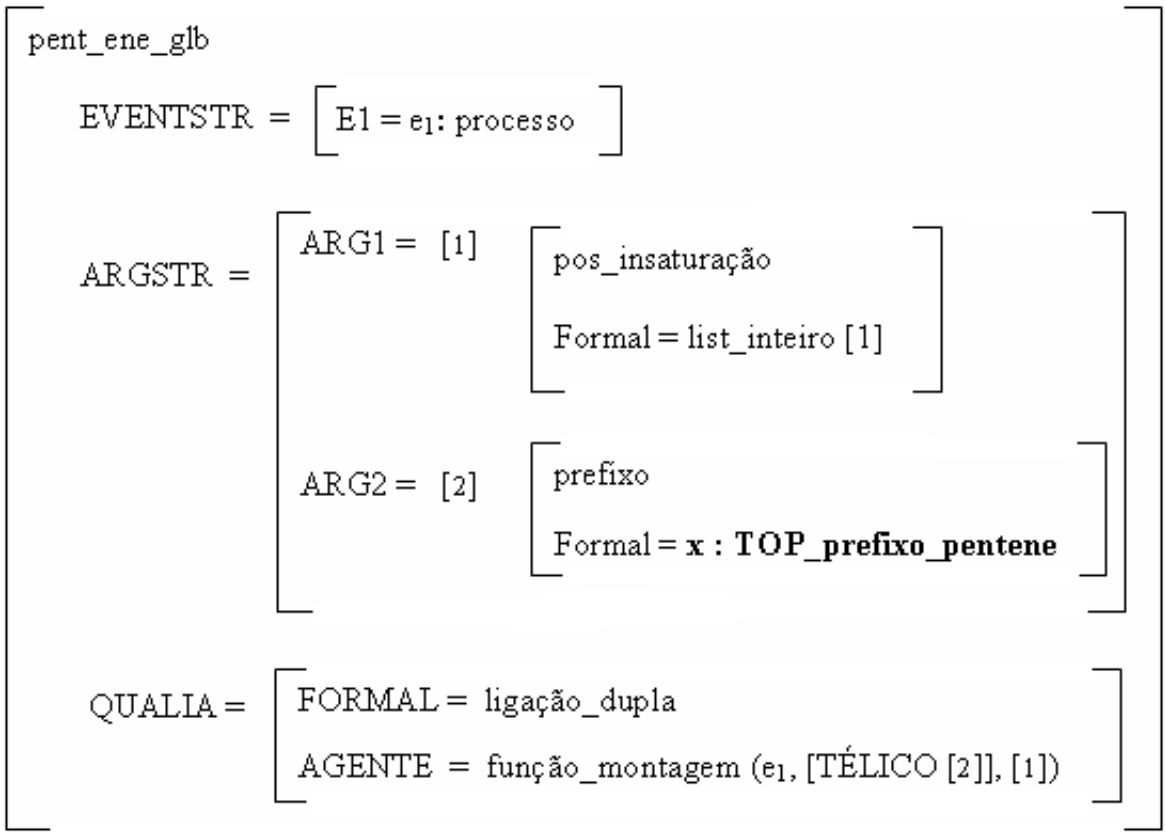

Figura 84: Estrutura Lexica para *pentene*.

O *pent_ene_glb* já pode ser considerado um nome do composto químico para os Alcenos, pois possui na sua estrutura léxica um argumento (ARG1) que indica a posição da insaturação na cadeia principal (no exemplo o carbono 1 possui a dupla ligação).

A estrutura lexica do *pent_ene_glb* prevê um segundo argumento, caso necessário. Este segundo argumento tem que ser um prefixo (radical e/ou multiplicador) que satisfaça a restrição especificada no Formal da estrutura de argumentos do *pent_ene_glb*, ou seja, tem que pertencer a um dos tipos da disjunção *TOP_prefixo_pentene*.

Analogamente ao ocorrido na análise de "pent" feita há pouco, pode-se observar que, em *TOP_prefixo_pentene*, TOP se refe ao LUB do conjunto formado por todos os prefixos aos quais um *pentene* pode ser aplicado, isto é, ramificações "*methyl*" nos carbonos 2, 3 ou 4 da cadeia principal (aqui chamadas de tipos M2, M3 e M4, respectivamente), e ramificações "*ethyl*" nos carbonos 2 ou 3 da mesma cadeia (aqui chamadas de tipos E2 e E3, respectivamente). Assim sendo, o tipo *TOP_prefixo_pentene* corresponde à disjunção entre os tipos M2, M3, M4, E2 e E3.

No RALQ, a função de montagem (papel "Agente") dos alcenos é a "mkAlkene".

Na estrutura léxica do *pent_ene_glb*, mostrada na figura 84, a estrutura Qualia possui

o papel (*role*) Formal que indica a dupla ligação, e o papel Agente aponta para a função montagem do *pent_ene_glb*, que agora já possui como parâmetros o papel TÉLICO do argumento 2 (dois), que é do tipo prefixo, e a posição da dupla ligação.

Se o segundo argumento for vazio, o nome do composto gerado será **1-pentene**. No caso do exemplo atual, **2-methyl-3-ethyl-1-pentene**, o segundo argumento corresponde a dois prefixos: o primeiro é o **2-methyl** e, o segundo, é **3-ethyl**.

A estrutura léxica do **2-methyl** está representada na figura 85. Seu argumento indica a cadeia principal a que este radical será ligado. A sua estrutura Qualia expressa o seu tipo prefixo, o seu papel Formal possui o tipo M2, significando que, o radical *methyl* ramificará no carbono 2 (dois) em alguma cadeia principal "y". O papel Télico objetiva ramificar esse prefixo na cadeia principal "y".

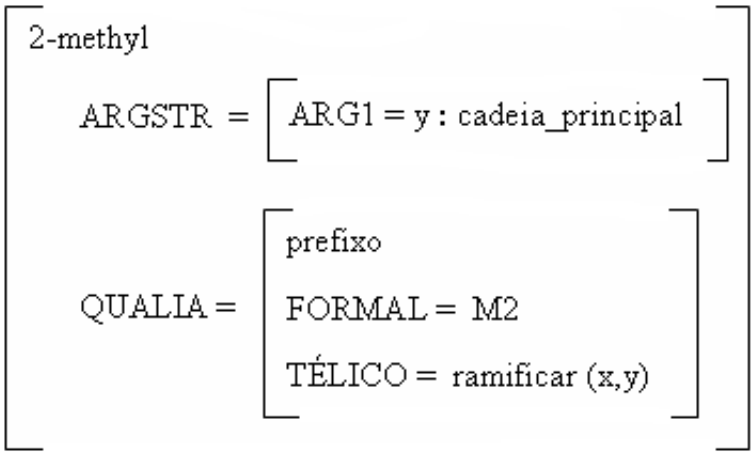

Figura 85: Estrutura Léxica para o radical *2-methyl*.

Como o radical **2-methyl** é do tipo M2, ele é unificável com o composto **1-pentene** (tal operação ilustra o processo de co-composição abordado na seção 3.2.4.2). A unificação executada gera a estrutura léxica da figura 86, representado o composto **2-methyl-1-pentene**.

Na figura 86 o radical **2-methyl** aparece corretamente como argumento do **1-pentene**. O tipo *TOP_prefixo_2_methyl_1_pentene* do composto **2-methyl-1-pentene** gerado corresponde à disjunção entre M3, M4 e E3.

Observe-se que a estrutura léxica do composto **2-methyl-1-pentene** traz informações importantes como:

- Houve alterações nas restrições relativas ao carbono 2 (dois) da cadeia principal, o qual não pode mais receber nenhuma ramificação, por não dispor mais de ligações

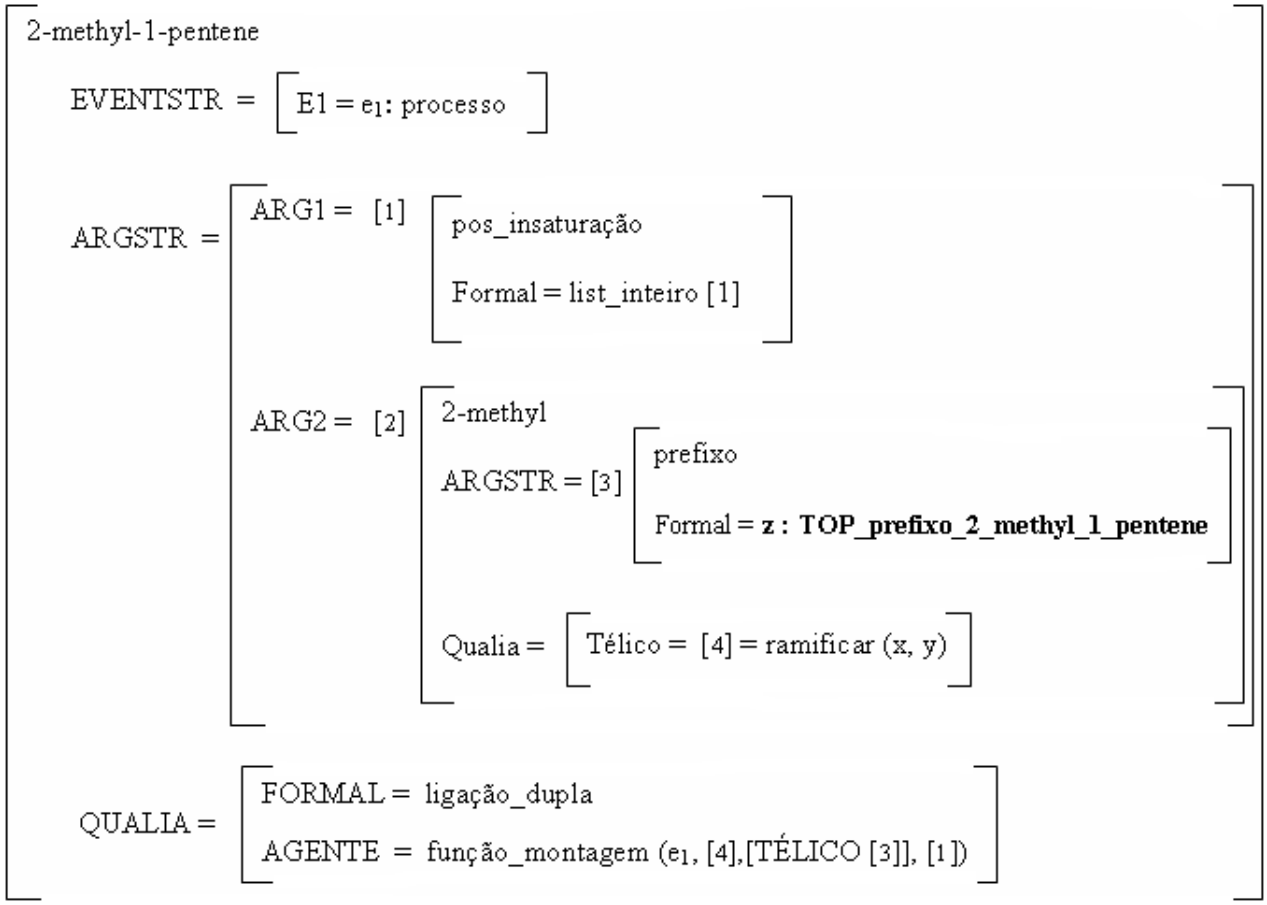

Figura 86: Estrutura Léxica para o composto *2-methyl-1-pentene*.

livres (tal fato se expressa na exclusão de M2 e E2 da disjunção de TOP). Devido a isso, o composto **2-methyl-1-pentene** está mais restrito;

- a função de montagem do papel Agentive da estrutura Qualia do composto **2-methyl-1-pentene**, recebeu por meio do Télico o radical *methyl* como uma ramificação no carbono 2 (dois) na cadeia principal do composto;

- Como o composto **2-methyl-1-pentene** ainda admite ramificações nos carbonos 3 (três) e 4 (quatro), para efetuá-las o Agente utilizará o Télico da estrutura léxica do próximo prefixo (caso haja e seja compatível).

Como o composto a ser formado é o **2-methyl-3-ethyl-1-pentene**, o próximo radical a ser analisado é o **3-ethyl**, que é do tipo prefixo a ser ligado a uma cadeia principal, ramificando-a no carbono 3, conforme mostrado na figura 87.

Como o composto **2-methyl-1-pentene** aceita como argumento o tipo prefixo e as restrições do *3-ethyl*, a operação de co-composição pode ser aplicada entre ambos, gerando,

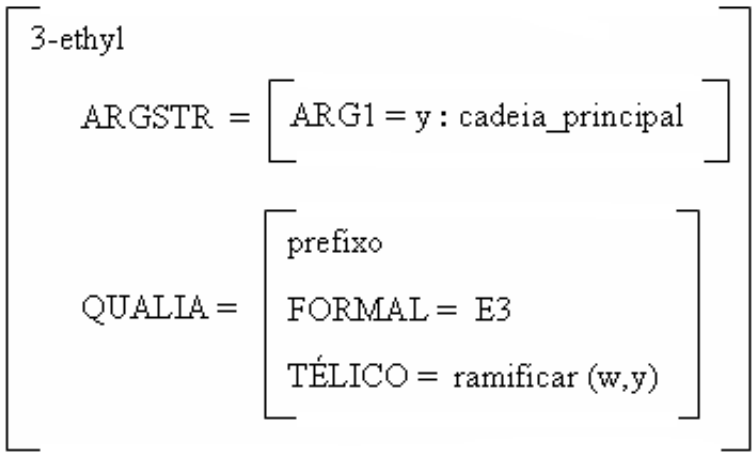

Figura 87: Estrutura Léxica para o radical *3-ethyl*.

assim, a estrutura léxica final para o composto *2-methyl-3-ethyl-1-pentene*, ilustrada na figura 88, abaixo.

Na figura 88, a *função_montagem* do Agente utiliza os 3 (três) parâmetros para montar o composto, onde, os atos de ramificar o radical do tipo prefixo *2-metil* (representado pelo número 4 (quatro)) e o radical do tipo prefixo *3-ethyl* (apontado pelo número 6 (seis)) serão os dois primeiros parâmetros. O último parâmetro é o da posição da tupla ligação, referenciada pelo número 1 (um). Como todas as restrições foram satisfeitas, o composto **2-methyl-3-ethyl-1-pentene** pode ser formado e tem como tipo **TOP_prefixo_2_methyl_3_ethyl_1_pentene**, que corresponde a uma disjunção entre M3, M4 e E3. Isso implica que este composto ainda admitiria ramificações "*methyl*" nos carbonos 3 e 4 e "*ethyl*" no terceiro.

As restrições da estrutura léxica ilustrada na figura 88 continuam as mesmas, em relação a estrutura lexica do composto **2-methyl-1-pentene** (figura 86). A diferença está na quantidade de ramificações possíveis de serem feitas. Por exemplo, o composto orgânico **2-methyl-1-pentene** possibilita 4 (quatro) ramificações, de acordo com a figura 89. Já o composto, **2-methyl-3-ethyl-1-pentene**, proporciona 3 (três) possíveis ramificações, como mostra as figuras (ilustrada na figura 90).

O exemplo acima mostra que a técnica de análise através dos *parsers* semânticos implementada no RALQ corresponde a uma estratégia que se enquadra nos moldes léxicos da TLG. Nele foi apresentado como o RALQ resolve ambigüidades lexicais do tipo: onde e em que casos um prefixo pode-se ligar? Ou, ainda: onde e em que casos pode haver uma insaturação nas cadeias orgânicas?

Para tanto, as estruturas léxicas dos componentes **pent_ene_glb**, **2-methyl-1-pente-**

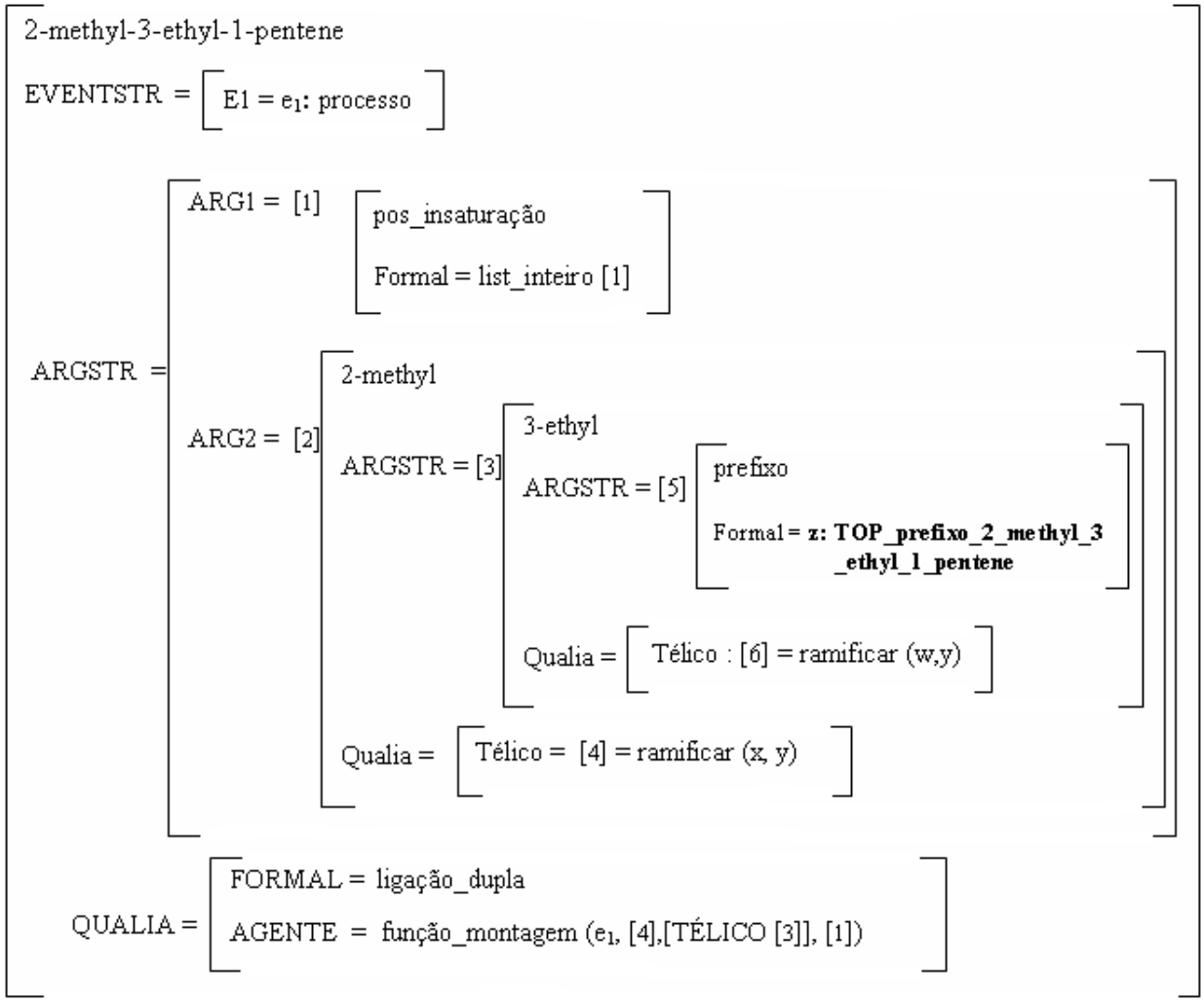

Figura 88: Estrutura Lexica para o composto *2-methyl-3-ethyl-1-pentene*.

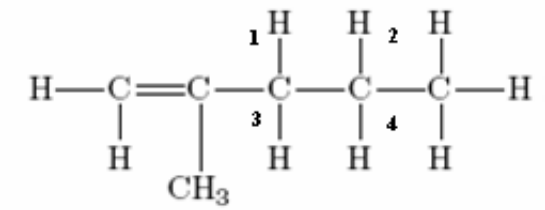

Figura 89: *2-methyl-1-pentene* com 4 (quatro) ramificações possíveis

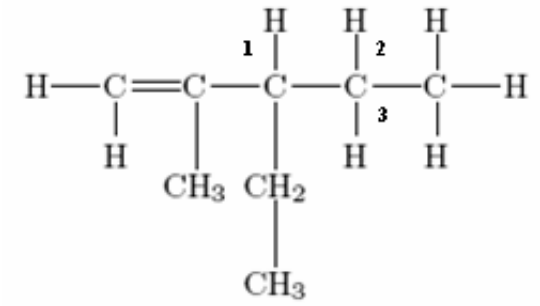

Figura 90: *2-methyl-3-ethyl-1-pentene* com 3 (três) ramificações possíveis

ne e **2-methyl-3-ethyl-1-pentene** possuem restrições quanto a localização das ramificações junto à cadeia principal, ou seja, restrições de argumento. Isso evita que uma ramificação ligada a uma carbono específico gere uma cadeia principal maior do que a original.

Como resultado da análise, o RALQ reconhece o nome correto do composto de entrada e fornece o desenho de sua estrutura. Mas é importante salientar que, no processo de desambiguação lexical, o RALQ vai além disso, pois ele consegue também analisar nomes de compostos de entrada inadequados e gerar suas estruturas químicas corretas. Isso é conseguido da seguinte forma: caso o nome de entrada não satisfaça nenhum conjunto de restrições estabelecido dentro das normas da nomenclatura oficial (isto é, caso nenhuma função de montagem o identifique como composto válido), ele será tratado por um conjunto de regras de ambigüidade implementadas, as quais sugerirão uma possível solução (estrutura química) para o composto de entrada.

Por exemplo, caso o radical **4-ethyl** (que possui na sua estrutura Qualia o papel Formal do tipo E4) fosse o próximo prefixo a ser inserido no composto **2-methyl-3-ethyl-1-pentene** para formar o composto **2-methyl-3,4-diethyl-1-pentene**, as funções montadoras não permitiriam tal inserção, pois as restrições do nome do composto **2-methyl-3-ethyl-1-pentene** não prevêem ramificação do tipo E4, como mostra a figura 88.

Assim sendo, o sistema RALQ trata o nome do composto **2-methyl-3,4-diethyl-1-pentene** (nome inadequado) como ambíguo e, através das regras de ambigüidade, detecta que há um nome alternativo correto para ele, ou seja, **2,4-dimethyl-3-ethyl-1-hexene**, gerando sua estrutura química mostrada na figura 91.

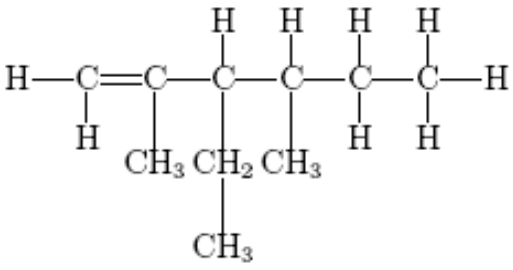

Figura 91: Estrutura Química do composto *2,4-dimethyl-3-ethyl-1-hexeno*.

Tal propriedade torna o RALQ uma opção muito útil, por exemplo, como ferramenta automática de ensino de Química.

4.5 A Pragmática

Como a pragmática é o estudo de como o contexto influencia a interpretação do significado [Levinson 1983], e sendo essa característica, juntamente com a sintaxe e a

semântica, importante no estudo da linguagem natural, este trabalho não poderia deixar de mencionar, de forma introdutória, a pragmática no domínio da química orgânica.

Na química orgânica, a pragmática se faz presente, por exemplo, na característica de movimentos atômicos na molécula, ocasionando "vibrações". Essas vibrações vão dar origem a faixas de absorção espectrais na molécula que são úteis na tarefa de identificar a qual grupo funcional pertence um composto analisado em espectrômetros de vibração molecular [Silverstein e Webster 2000].

Cada grupo funcional possui uma faixa de absorção que o caracteriza. Tal faixa está devidamente cadastrada em uma tabela de correlação que lista as faixas de absorção correlacionadas aos diversos grupos funcionais, dados estes, conseguidos através da "leitura" de vibração de moléculas por meio de um "espectrômetro" (aparelho que mede a freqüência de vibração dos átomos) [Silverstein e Webster 2000]. Consequentemente, o nome de uma molécula analisada em espectrômetro só vai ser reconhecido se houver a correspondência entre a faixa de absorção obtida na leitura do espectrômetro e uma das possíveis faixas de absorção presentes na tabela de grupos funcionais [Silverstein e Webster 2000].

Dessa forma, a definição do nome da estrutura química vai depender da faixa de absorção que os átomos (que formam a estrutura química) possuem.

Como foi visto nas seções anteriores, no RALQ é feita uma análise sintática dos nomes orgânicos de entrada que definirá a quantidade de átomos de carbonos presentes na cadeia principal do composto analisado. Tal dado será utilizado pela semântica (funções de montagem) para formar a estrutura química do composto. Pretende-se trabalhar em uma expansão do sistema que o habilite a analisar os resultados obtidos nos espectrômetros a vibração.

Um exemplo de uma leitura de espectro das vibrações dos átomos realizada pelo "espectrômetro" pode ser visto na figura 92.

A figura 92 representa as freqüências de vibrações das ligações entre os átomos de um composto químico. Através da intensidade das vibrações indicadas no gráfico detectam-se presenças tais como: de hidroxila, de insaturações, de carbonos ligados a oxigênio, de carbono ligado a hidrogênio etc. Detecta-se, também, uma possível ocorrência de anel aromático.

O conjunto desses dados levantados pelo espectrômetro correspondem às "impressões digitais" do composto. Quando tais "impressões" já estão cadastradas em catálogos químicos, a tarefa de identificar o nome do composto a partir do espectro consiste em consultar

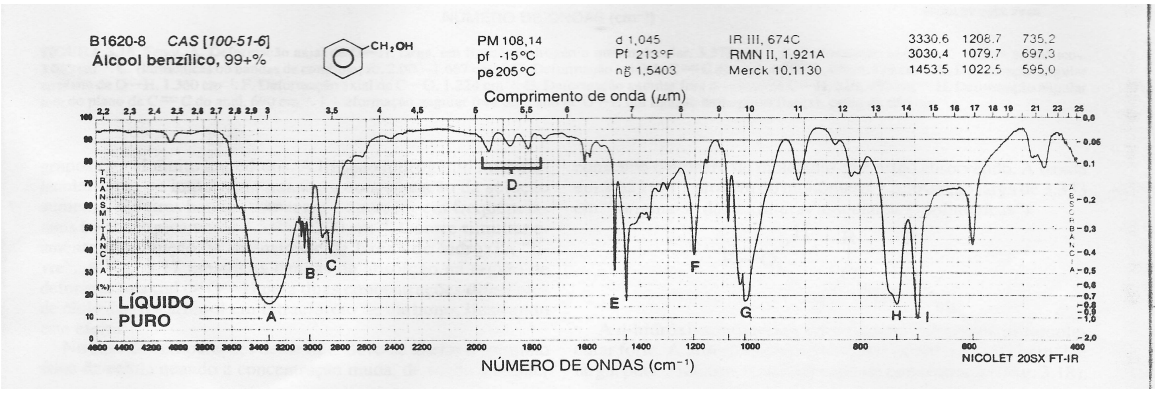

Figura 92: Álcool benzílico.

A. Deformação axial de O-H em ligação hidrogênio intermolecular, 3,331 cm^{-1}. B. Deformação axial de C-H: aromático 3.100-3.000 cm^{-1}. C. Deformação axial de C-H: metileno, 2.980-2.840 cm^{-1}. D. Harmônicas ou freqüências de combinação, 2.000-1.667 cm^{-1}. E. Deformação axial das ligações C=C do anel, 1.497, 1.454 cm^{-1}, cobertas pela deformação angular simétrica no plano de CH_2 em 1.471 cm^{-1}. F. Deformação angular de O-H, possivelmente confundida com a deformação angular no plano de C-H, 1.209 cm^{-1}. G. Deformação axial de C-O de álcoois primários, 1.023 cm^{-1}. H. Deformação angular fora do plano de C-H de aromático, 735 cm^{-1}. I. Deformação angular de C=C, 697 cm^{-1}.

tal catálogo na tentativa de identificar um gráfico análogo ao obtido na análise espectral (no caso do exemplo, será o do álcool Benzílico).

Contudo, quando o resultado do espectrômetro não corresponder a um gráfico cadastrado, a tarefa de identificação do composto será muito mais árdua, pois terão de ser analisadas todas as restrições impostas pela análise espectral no sentido de detectar a qual composto o espectro corresponde.

5 Conclusão e Trabalhos Futuros

O sistema RALQ desenvolvido neste trabalho é uma ferramenta automática de apoio ao ensino da Química Orgânica especializado na análise de nomes de fórmulas químicas e na geração do desenho de suas estruturas químicas correspondentes. É importante salientar que o RALQ consegue efetuar tais análises mesmo em situações desfavoráveis em que o nome analisado seja incorreto (ou seja, representa um composto químico inexistente), ou em que ele seja inadequado (ou seja, apesar de o nome ser correto, ele não respeita as nomenclaturas oficiais). No primeiro caso (nome incorreto), o sistema acusa o erro. No segundo caso (nome inadequado), o RALQ efetua a desambiguação do nome apresentado, corrige-o para deixá-lo concordante com a nomenclatura oficial e retorna o desenho de sua estrutura química correspondente.

As Análises Sintático-Semântica das fórmulas químicas são efetuadas à luz da TLG e implementadas através dos Combinadores de Parsers e do Clean.

As vantagens das técnicas acima mencionadas são: a alta eficiência do Clean na implementação e manipulação dos tipos dos itens léxicos, tipos, estes, que guiam a análise sintático-semântica das fórmulas; alta eficiência e flexibilidade dos Combinadores de *Parser* para conciliar análise sintática com análise semântica; enquadramento formal e teórico fornecido pela TLG para emoldurar o processo de análise guiada pelo tipos dos itens léxicos.

O protótipo implementado analisa 6 (seis) funções da Química Orgânica: Alcanos, Alcenos, Alcinos, Alcadienos, Álcoois e Aldeídos, podendo, facilmente, ser ampliado para outras funções da química orgânica. O programa efetua com êxito sua tarefa de análise desses compostos, obtendo ótimos resultados, inclusive, na detecção e resolução do programa da ambigüidade léxica.

Além da limitação na quantidade de funções implementadas no protótipo, há, ainda, uma limitação na quantidade de carbonos da cadeia principal que ele consegue tratar: por enquanto, o programa trabalha com cadeias de, no máximo, cerca de 6 átomos de

carbono. Mas, como no caso anterior, o protótipo pode ser facilmente estendido para comportar cadeias maiores.

Outro aprimoramento a ser feito no protótipo consiste em implementar uma interface mais amigável entre o usuário e o sistema, o que possibilitará uma maior interatividade entre eles.

Pretende-se expandir o RALQ também introduzindo a análise pragmática citada na seção 4.5. Com isso, espera-se que o sistema seja apto a avaliar resultados dos espectrômetros de vibração e identificar o composto analisado nos mesmos, independentemente de seus espectros já serem cadastrados ou não. No caso de não serem, a tarefa de identificação poderá demandar avaliações complementares como, por exemplo, avaliações dos resultados produzidos, também, pelos espectrômetros de massa.

Saliente-se que os programas de análise automática de resultados obtidos em espectrômetros disponíveis comercialmente são extremamente caros e raros. Além disso, eles são voltados, principalmente, para análises de resultados obtidos por espectrômetros de massa (para os de vibração são bem incipientes). Nesses últimos, a identificação dos compostos feita por espectrômetros de massa utiliza uma biblioteca (banco de dados) onde hoje estão cadastrados mais 500.000 compostos (é a maior que existe hoje no mercado). No caso, os programas comparam os resultados obtidos pelo espectrômetro de massa com os compostos registrados na biblioteca, com o intuito de identificar o composto que está sendo analisado. O alto custo de bibliotecas como essa (mais de $20.000,00) mostra o quão é relevante e oportuno o direcionamento de esforços de pesquisa voltados para a automatização de sistemas de análises químicas. Finalmente, pode-se concluir o RALQ, em sua versão atual, representa uma ferramenta automática bastante útil como auxiliar de ensino em Química Orgânica. Além disso, sua expansão, nos moldes apresentados acima, torna-lo-á uma ferramenta extremamente útil também nos laboratórios químicos, como ferramenta auxiliar na análise de resultados obtidos em espectrômetros de vibração.

Referências

[Agirre e Rigau 1996]AGIRRE, E.; RIGAU, G. Word sense disambiguation using conceptual density. In: *Proceeding of the 16th International Conference on Computational Linguistics*. Copenhagen - DK: [s.n.], 1996.

[Allison 1986]ALLISON, L. *A Pratical Introduction to Denotation Semantics*. [S.l.]: Cambridge University Press, 1986.

[Andrade 2000]ANDRADE, L. N. de. *Breve Introdução ao Latex 2E*. [S.l.], Abril 2000.

[Anstein e Kremer 2005]ANSTEIN, S.; KREMER, G. *Analysing Names of Organic Chemical Compounds From Morpho-Semantics to SMILES Strings and Classes*. Dissertação (Mestrado) — Universit¨at Stuttgart, 2005.

[Atkins e Fillmore 1994]ATKINS, S.; FILLMORE, C. Starting where the dictionaries stop: The challenge of corpus lexicography. *In Atkins & Zampolli*, p. 350–393, 1994.

[Bach 1989]BACH, E. *Informal Lectures on Formal Semantics*. [S.l.]: State University of New York, Albany, 1989.

[Barsalou 1983]BARSALOU, L. Ad hoc categories. *Memory & Cognition*, v. 11, n. 3, p. 211–227, 1983.

[Brecher 1999]BRECHER, J. Name=struct: A practical approach to the sorry state of real-life chemical nomenclature. *J. Chem. Inf. Comput. Sci.*, n. 39, p. 943–950, 1999.

[Brown et al. 1991]BROWN, P. F. et al. Word sense disambiguation using statistical methods. In: *Proceedings of the 29th annual meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1991. p. 264 – 270.

[Brun 2000]BRUN, C. A client/server architecture for word sense disambiguation. In: *Proceedings of the 18th conference on Computational linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 2000. v. 1, p. 132 – 138. ISBN 1-55860-717-X.

[Cambridgesoft 2005]CAMBRIDGESOFT. Converting chemical names to structures with name=struct. 2005. Disponível em: <http://www.cambridgesoft.com/products/pdf/whitepapers/NameStruct.pdf>. Acesso em: Novembro, 2005.

[Copeland et al. 1991]COPELAND, C. et al. (Ed.). *The Eurotra Formal Specifications*. [S.l.]: Office for Official Publications of the Commission of the European Community, 1991. (Studies in Machine Translation and Natural Language Processing, v. 2).

[Copestake 1992]COPESTAKE, A. *The Representation of Lexical Semantic Information*. [S.l.]: University of Sussex, 1992.

[Dagan e Itai 1994]DAGAN, I.; ITAI, A. Word sense disambiguation using a second language monolingual corpus. *Comput. Linguist*, MIT Press, Cambridge, MA, USA, v. 20, n. 4, p. 563 – 596, 1994. ISSN 0891-2017.

[Dagan, Itai e Schwall 1991]DAGAN, I.; ITAI, A.; SCHWALL, U. Two languages are more informative than one. In: *Proceedings of the 29th annual meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1991. p. 130–137.

[Davison 1967]DAVISON, D. *The Logical Form of Action Sentences*. [S.l.]: N. Rescher, 1967.

[Dicionário Michaelis - Inglês & Português 2001]DICIONáRIO Michaelis - Inglês & Português. São Paulo, 2001.

[Dorr e Katsova 1998]DORR, B. J.; KATSOVA, M. Lexical selection for cross-language applications: Combining lcs with wordnet. In: *Proceeding of the AMTA 1998*. Langhorne - PA - USA: [s.n.], 1998. p. 438 – 447.

[Egedi et al. 1994]EGEDI, D. et al. Korean to english translatin using synchronous tags. In: *Proceedings of the 1st Conference of the Association for Machine Translation in the Americas*. Maryland - USA: [s.n.], 1994. p. 48 – 55.

[Fujita 1993]FUJITA, S. *Xymtex: A Macro Package for Typesetting Chemical Structural Formulas*. 1.00. ed. Japan, December 1993.

[Fujita 1994]FUJITA, S. *Typesetting structural formulae with the text formatter TEX/LATEX*. [S.l.]: Computers and Chemistry, 1994. 109-116 p.

[Gajek 1991]GAJEK, O. The metal system. *Communications of the ACM*, ACM Press, New York, NY, USA, v. 34, n. 9, p. 46 – 47, 1991. ISSN 0001-0782.

[Gerstenberger 2001]GERSTENBERGER, C. V. *Semantische Analyse von Namen Organischer Verbindungen oder Was Bedeutet 3,3' Ureylen dibenzamidin?* Dissertação (Master's thesis) — University of Stuttgart, 2001.

[Group 2006]GROUP, S. T. R. Overview clean language features. In: RADBOUD UNIVERSITEIT NIJMEGEN. Nijmeegs Instituut Voor Informatica en Informatiekunde, 2006. Disponível em: <http://www.cs.ru.nl/~clean/About_Clean/Clean_Language_Features/clean_language_features.html>. Acesso em: 11 jul. 2006.

[Guthrie et al. 1991]GUTHRIE, J. A. et al. Subject-dependent co-occurrence and word sense disambiguation. In: *Proceeding of the 29th Annual Meeting of the Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1991. p. 146 – 152.

[Hayes 1976]HAYES, P. A process to implement some word sense disambiguation. *Institut pour les Etudes Sémantiques et Cognitives, Universitè de Genève - Gèneve*, v. 23, 1976.

[Hirst 1987]HIRST, G. *Semantic Intepretation and the Resolution of Ambiguity*. Cambridge - UK: Cambridge Universisty Press, 1987.

[Ide e Véronis 1998]IDE, N.; VéRONIS, J. Word sense disambiguation: The state of the art. *Computational Linguistics,*, v. 24, n. 1, p. 1–41, 1998.

[ILTEC]ILTEC. Eurotra. In: _____. [s.n.]. Disponível em: <http://www.iltec.pt/projectos/concluidos/eurotra.html>. Acesso em: 04 out. 2005.

[Karov e Edelman 1998]KAROV, Y.; EDELMAN, S. Similarity-based word sense disambiguation. *Comput. Linguist.*, MIT Press, Cambridge, MA, USA, v. 24, n. 1, p. 41 – 59, 1998. ISSN 0891-2017.

[Koopman et al. 1997]KOOPMAN, P. et al. *Parser Combinators*. Agosto 1997. Web - ftp://ftp.cs.kun.nl/pub/Clean/papers/cleanbook/.

[Koopman et al. 2002]KOOPMAN, P. et al. *Functional Programming in CLEAN*. Setembro 2002. Web - ftp://ftp.cs.kun.nl/pub/Clean/papers/cleanbook/.

[Laboratory 1998]LABORATORY, C. S. Wordnet - a lexical database for english. In: _____. Princeton University, 1998. Disponível em: <http://cogsci.princeton.edu/ wn/>. Acesso em: 04/10/2005.

[Lee e Ng 2002]LEE, Y.; NG, H. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*. Citeseer, 2002. p. 41–48. Disponível em: <citeseer.ist.psu.edu/lee02empirical.html>.

[Lesk 1986]LESK, M. Automated sense disambiguation using machine-readable dictionaries: How to tell a pine cone from an ice cream cone. In: *Proceeding of the 1986 SIGDOC Conference*. Toronto: [s.n.], 1986. p. 24–26.

[Levinson 1983]LEVINSON, S. C. *Pragmatics*. [S.l.: s.n.], 1983.

[Lieber 2004]LIEBER, R. *Morphology and Lexical Semantics*. [S.l.]: Cambridge University Press, 2004. ISBN 0521831717.

[Litowski 1997]LITOWSKI, K. C. Desiderata for tagging with wordnet sysnsets or mcaa categories. In: *ACL-SIGLEX Workshop "Tagging Text with Lexical Semantics: Why, What, and How?"*. [S.l.: s.n.], 1997.

[Lloyd 1984]LLOYD, J. W. *Foundations of Logic Programming*. [S.l.]: Springer-Verlag, 1984.

[Masterman 1957]MASTERMAN, M. *The Thesaurus in Syntax and Semantics*. [S.l.]: Mechanical Translation, 1957. 1-2 p.

[Masterman 1961]MASTERMAN, M. Semantic message detection for machine translation using an interlingua. In: *International Conference on Machine Translation of Languages and Applied Lnguage Analysis*. London: [s.n.], 1961. p. 437–475.

[Montoyo et al. 2002]MONTOYO, A. et al. The role of wsd for multilingual natural language applications. In: *Proceeding of the TSD 2002*. Czech Republic: [s.n.], 2002. p. 41–48.

[Paliouras et al. 2000]PALIOURAS, G. et al. Learning rules for large-vocabulary word sense disambiguation: A comparison of various classifiers. In: *NLP 00: Proceedings of the Second International Conference on Natural Language Processing*. London, UK: Springer-Verlag, 2000. p. 383–394. ISBN 3-540-67605-8.

[Pantel e Lin 2002]PANTEL, P.; LIN, D. Discovering word senses from text. In: *KDD 02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM Press, 2002. p. 613 – 619. ISBN 1-58113-567-X.

[Parsons 1990]PARSONS, T. *Events in the Semantics of English*. [S.l.]: MIT Press, Cambridge, MA, 1990.

[Patrick 1985]PATRICK, A. B. *An Exploration of Abstract Thesaurus Instanttiation*. Dissertação (Mestrado) — University of Kansas, Kansas - USA, 1985.

[Pedersen 1997]PEDERSEN, B. S. *Lexical Ambiguity in Machine Translation: Expressing Regularities in the Polysemy of Danish Motion Verbs*. Tese (Doutorado) — Center for Sprogteknologi, Copenhagen - DK, october 1997.

[Pereira 2004]PEREIRA, L. E. Thesaurus - todo mundo sabe o que é? In: _____. [s.n.], 2004. Disponível em: <http://listas.cev.org.br/pipermail/cevcbce/2004-May/000218.html>. Acesso em: 29/09/2005.

[Pustejovsky 1991]PUSTEJOVSKY, J. The generative lexicon. *Computational Linguistics*, v. 17, n. 4, p. 409–441, dezembro 1991.

[Pustejovsky 1995]PUSTEJOVSKY, J. *The Generative Lexicon*. [S.l.]: The MIT Express, 1995.

[Pustejovsky 1995]PUSTEJOVSKY, J. *Linguistic Constraints on Type Coercion*. [S.l.]: Computational Lexical Semantics, Cambridge University Press, 1995.

[Pustejovsky e Boguraev 1993]PUSTEJOVSKY, J.; BOGURAEV, B. Lexical knowledge representation and natural language processing. *Artificial Intelligence*, n. 63, p. 193–223, 1993.

[Resnik 1995]RESNIK, P. Disambiguating noun groupings with respect to wordnet senses. In: *Proceeding of the 3rd Workshop on Very Large Corpora*. Cambridge - UK - USA: [s.n.], 1995. p. 54 – 68.

[Rivest 1987]RIVEST, R. L. Learning decision lists. *Mach. Learn.*, Kluwer Academic Publishers, Hingham, MA, USA, v. 2, n. 3, p. 229–246, 1987.

[R.Mihalcea e Moldovan 1999]R.MIHALCEA; MOLDOVAN, D. I. A method for word sense disambiguation of unrestricted text. In: *Proceeding of the 7th Annual Meeting of the Association for Computational Linguistics*. Maryland - USA: [s.n.], 1999.

[Shieber e Schabes 1990]SHIEBER, S.; SCHABES, Y. Synchronous tree adjoining grammar. In: *Proceeding of the 13th International Conference on Computational Linguistics*. Helsinki - FI: [s.n.], 1990.

[Silverstein e Webster 2000]SILVERSTEIN, R. M.; WEBSTER, F. X. *Identificação Espectrométrica de Compostos Orgânicos*. [S.l.]: LTC, 2000.

[Small 1980]SMALL, S. L. *Word-expert Parsing, a Theory of Distributed Word-based Natural Language Based Understandig*. [S.l.], 1980. v. 954.

[Souza 2002]SOUZA, J. N. de. *Lógica para Ciência da Computação*. Rio de Janeiro: [s.n.], 2002. ISBN 8535210938.

[Specia e Nunes 2002]SPECIA, L.; NUNES, M. G. V. *Introdução aos Métodos e Paradigmas de Tradução Automática*. São Carlos - SP, março 2002. 26 p.

[Specia e Nunes 2004]SPECIA, L.; NUNES, M. G. V. *Desambiguação Lexical Automática de Sentido: um Panorama*. São Carlos - SP, Agost. 2004. v. 238, 122 p.

[Sussna 1993]SUSSNA, M. Word sense disambiguation for free-text indexing using massive semantic network. In: UNIVERSITY OF VIRGINIA. *Proceeding of the 2nd International Conference on Information and Knowledge Base Managment*. Charlottesville - VA, 1993. p. 67–74.

[Teixeira 2004]TEIXEIRA, M. Coesão referencial. estudos do discurso ii. In: _____. Centro de Ciências da Comunicação − Curso de Letras − UNISINOS, 2004. Disponível em: <www.comunica.unisinos.br/professores/marlene/arquivos/referenciacao_2004_1.pdf>. Acesso em: 04/10/2005.

[Towell e Voorhees 1998]TOWELL, G.; VOORHEES, E. M. Disambiguating highly ambiguous words. *Comput. Linguist.*, MIT Press, Cambridge, MA, USA, v. 24, n. 1, p. 125–145, 1998. ISSN 0891-2017.

[Usberco e Salvador 2001]USBERCO, J.; SALVADOR, E. *Química Essencial*. 1. ed. [S.l.]: Saraiva, 2001.

[Vossen 1998]VOSSEN, P. Eurowordnet: Building a multilingual database with wordnets for european languages. *The ELRA Newsletter*, v. 3, n. 1, p. 7–12, 1998.

[Wikipédia 2006]WIKIPéDIA. Parser. In: _____. [S.l.: s.n.], 2006.

[Wilks e Stevenson 1996]WILKS, Y.; STEVENSON, M. *The grammar of sense: Is word sense tagging much more than part-of-speech tagging?* [S.l.], 1996.

[Yarowsky 1992]YAROWSKY, D. Word-sense disambiguation using statistical models of roget's categories trained on large corpora. In: *Proceedings of the 14th conference on Computational linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1992. p. 454–460.

[Yarowsky 1994]YAROWSKY, D. Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish and french. In: *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1994. p. 88–95.

[Yarowsky 1995]YAROWSKY, D. Unsupervised word sense disambiguation rivaling supervised methods. In: *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1995. p. 189–196.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo