INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Extracção de Informação - Introdução Automática de Receitas de acordo com Ontologia

Telmo Ramos Machado

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática e de Computadores

Júri

Presidente:	Prof. Ernesto José Marques Morgado
Orientador:	Prof. Nuno João Neves Mamede
Co-orientador:	Prof. Helena Sofia Andrade Nunes Pereira Pinto
Vogais:	Prof. Jorge Manuel Evangelista Baptista

Setembro 2007

Resumo

A importância da extracção de informação tem aumentado ao longo dos anos, como consequência do aumento de informação disponível e da falta de tratamento de que essa informação era alvo. Como tal, os sistemas de extracção de informação surgem como um mecanismo para filtrar essa informação e obter apenas a necessária do domínio que se está a tratar.

É apresentado um sistema que realiza extracção de informação num domínio específico (da culinária), sendo composto por três módulos: pré-processamento das receitas, processamento e transformação da saída. Os objectivos principais são identificar os ingredientes e as quantidades associadas, e identificar as diferentes tarefas necessárias à preparação de cada receita tendo em conta os utensílios necessários e os ingredientes usados em cada uma dessas tarefas. Depois de identificados os ingredientes e as tarefas de cada receita, estes são introduzidos numa base de dados, tendo como base uma ontologia, que contém uma descrição dos conceitos usados no domínio da culinária.

Abstract

The importance of information extraction has grown up lately, as a consequence of the growing available information and the lack of treatment of that information. Therefore, information extraction systems appear as a mechanism to filter that information and only obtain the needed information, depending on the domain.

This thesis shows an information extraction system in a specific domain, the cooking domain, which is composed of three modules, pre-processing, recipe processing and output transformation. The main objectives are to identify ingredients and their associated quantities, and to identify the different tasks needed to prepare each recipe, as well as their needed utensils and the used ingredients in each of these tasks. After identifying the ingredients and the tasks of each recipe, these are introduced into a database based on an ontology, which contains a description of the concepts used in the culinary domain.

Palavras-chave
Keywords

Palavras-chave

Extracção de Informação

Culinária

Ontologia

XIP

Processamento de Receitas

Introdução Automática

Keywords

Information Extraction

Cooking

Ontology

XIP

Recipe Processing

Automatic Introduction

Índice

Lista de Figuras

Lista de Tabelas

1 Introdução

A extracção de informação é um processo de pesquisa sobre textos, com o objectivo de encontrar informação relevante para um determinado domínio, como, por exemplo, extrair entidades, relações e eventos. Os textos em língua natural são, normalmente, não estruturados ou parcialmente estruturados, tendo os sistemas de extracção de informação como função recolher a informação importante destes textos e produzir informação estruturada, que pode ser uma representação lógica, tal como predicados, ou uma base de dados relacional, onde é introduzida a informação obtida. Assim sendo, a extracção de informação tem vindo a ter cada vez mais importância, visto que a quantidade de informação disponível aumenta exponencialmente de ano para ano e há a necessidade de ter apenas acesso à informação relevante relativa aos vários domínios de interesse. Por este motivo surgem os sistemas de extracção de informação de forma a que seja seleccionada apenas a informação mais importante para um determinado domínio. Várias técnicas têm sido usadas neste âmbito, sendo as mais usadas a utilização de regras e descoberta de padrões, uso de análise sintáctica e baseada em aprendizagem, usando uma variedade de métodos estatísticos.

Nesta tese apresenta-se um sistema que permite extrair informação do domínio da culinária, identificando os elementos relevantes (ingredientes e o conjunto de acções que constituem a preparação), inserindo-os numa base de dados preparada para o efeito, tendo como base uma ontologia que caracteriza os conceitos deste domínio.

No capítulo 2 é feita uma abordagem aos sistemas existentes que realizam extracção de informação em domínios específicos, mas diferentes do domínio da culinária, usando diversas abordagens na sua elaboração.

O capítulo 3 contém a arquitectura do sistema criado: o pré-processamento, o processamento das receitas e a transformação da saída. Apresenta-se também a cadeia de processamento XIP, que é usada no processamento das receitas para obter a informação sintáctica e as dependências relevantes para este domínio da culinária. Está também presente neste capítulo uma descrição do corpus usado para testar e avaliar o sistema.

O capítulo 4 descreve a ontologia usada como base para este trabalho e que contém todos os conceitos necessários para o domínio. É igualmente apresentada a base de dados criada tendo em conta a estrutura da ontologia e para onde é carregada a informação obtida das receitas processadas.

No capítulo 5 é apresentado o módulo de processamento de receitas, que utiliza a cadeia de processamento XIP para extrair a informação através da aplicação de regras.

O capítulo 6 contém a descrição do módulo de transformação de saída, que realiza a a transformação da saída normalizada em XML obtida no módulo de processamento de receitas para código SQL e consequente introdução da informação na base de dados.

Para finalizar, o capítulo 7 contém a avaliação realizada para os módulos de processamento de receitas e transformação da saída, considerando os ingredientes e as acções.

2 Estado da Arte

Ao longo do tempo têm sido vários os sistemas de extracção de informação criados e diversas as abordagens usadas. Segundo Blaschke et al. (Blaschke et al., 2002), o interesse nestes sistemas começou na década de 50, com os trabalhos de Zellig Harris (Harris, 1952), e, mais tarde, de Noam Chomsky (Chomsky, 1965). Até aos anos 80, os sistemas que eram desenvolvidos baseavam-se no preenchimento de simples templates, embora se tenha tentado induzir os campos dessas templates, como afirmam Gaizauskas e Wilks (Gaizauskas & Wilks, 1998). Esta última abordagem foi sendo progressivamente abandonada até ao ínicio dos anos 90 por ser demasiado complexa.

A partir desta altura a tecnologia de extracção de informação fez rápidos progressos, devido a uma série de conferências focadas na avaliação destes sistemas, as *Message Understanding Conferences*. Até à data realizaram-se sete conferências, sendo os temas diversificados, tal como mensagens de operações navais, ataques terroristas, lançamento de satélites, entre outros, como descritas por Gaizauskas e Wilks (Gaizauskas & Wilks, 1998). Desde a sexta edição que os sistemas concorrentes podem escolher a tarefa a que se propõem resolver, como por exemplo, reconhecimento de entidades nomeadas ou resolução de co-referências. Ao longo das edições os sistemas foram melhorando a sua prestação, tanto na precisão como na cobertura, no entanto, a barreira dos 60% definida por Hobbs (Hobbs, 2002) ainda não foi ultrapassada para reconhecimento de cenários, devido, principalmente, a limitações da tecnologia e ao facto de se extrair apenas o que está explícito no texto, necessitando de realizar inferências para resolver este problema.

Como refere Blaschke et al. (Blaschke et al., 2002), os sistemas de extracção de informação têm evoluído, mudando também os seus métodos, passando de sistemas construídos com base em regras manuais para regras aprendidas automaticamente, e também passaram a ser compostos apenas por técnicas estatísticas (como os *Hidden Markov Models*). Estas técnicas têm sido usadas com grande sucesso pelos biólogos na análise de genoma, e quando o corpus disponível é grande. De igual modo, tem havido uma mudança da análise sintáctica baseada em regras para técnicas de análise parcial e de segmentação aproximada.

Com a expansão da Internet nos últimos anos têm igualmente surgido sistemas para extrair algum tipo de informação de páginas web, que contém informação estruturada ou semi-estruturada, usando para o efeito parsers de HTML, como indica Ying (Ying, 2006). Novas áreas de investigação estão a

aparecer, tal como os sistemas de pergunta-resposta, também impulsionadas por conferências, como o TREC e o CLEF.

Na secção 2.1 descrevem-se alguns dos sistemas que existem para realizar extracção de informação no domínio da biologia e biomedicina, e na secção 2.2 alguns sistemas referentes ao domínio dos artigos desportivos, nomeadamente, que realizam extracção de resultados de jogos desportivos.

2.1 Extracção de Informação em artigos de Biologia e Biomedicina

Tem havido um crescente interesse em certas áreas como a biomedicina, devido ao enorme crescimento da literatura disponível online da área (estimado em cerca de 500 mil artigos por ano), à necessidade urgente dos biológos de encontrar informação relevante, e também ao crescente sucesso da aplicação de técnicas de processamento de língua natural a conteúdo baseado na Internet, como refere Ying (Ying, 2006).

Segundo Blaschke et al. (Blaschke et al., 2002), os objectivos dos sistemas são muito variados, entre os quais estão a descrição de proteínas e das suas funções, detecção de nomes de proteínas e relações entre proteínas, localização de proteínas em células ou tecidos, interacções proteína-medicamento e proteína-proteína e análise de sequências de DNA.

Conforme indicam Rindflesh et al. (Rindflesch et al., 2000), existem várias abordagens para a extracção de informação de textos biomédicos, que são a utilização de parsing sintáctico, processamento de estatísticas relativas à frequência da informação, sistemas baseados em regras e sistemas baseados em aprendizagem. Outros sistemas juntam mais do que uma abordagem na sua realização. Os sistemas que se descrevem nesta secção são, na sua maioria, constituídos por dois módulos, um módulo inicial de análise sintáctica, seguido de uma análise contextual. Outras diferenças que poderão ocorrer dizem respeito à utilização de dicionários.

A extracção de informação neste domínio é uma tarefa estimulante devido às dificuldades que traz. Algumas dessas dificuldades dizem respeito à não existência de uma convenção consistente para os nomes, que é agravada pela constante criação de novos termos, o que implica que os dicionários e ontologias existentes não estejam actualizados com os últimos avanços no domínio. Para além deste facto, existe também ambiguidade entre os termos, na medida em que um termo pode ser usado para denotar mais do que uma classe semântica, tal como *p53*, que é usado para identificar tanto um gene como uma proteína, como indicam Mukherjea et al. (Mukherjea et al., 2004). Proux et al. (Proux et al., 1998) referem que pode também acontecer que o nome de genes possa pertencer a uma categoria gramatical, sendo assim anotadas com a a categoria respectiva, como *boss*, ou *vamp*.

Nas secções 2.1.1 a 2.1.7 são apresentados sistemas que realizam extracção de informação neste domínio da biologia e biomedicina. Os sistemas das secções 2.1.1 a 2.1.6 são baseados em regras, enquanto que o sistema da sub-secção 2.1.7 é baseado em aprendizagem.

2.1.1 BioAnnotator

O sistema BioAnnotator (Mukherjea et al., 2004) foi desenvolvido para realizar a identificação de termos biológicos da literatura científica e anotar estes termos com a sua classe correspondente. Usa dicionários e propriedades das palavras para realizar a identificação de novos termos.

O BioAnnotator usa como corpus as publicações disponíveis na base de dados da MedLine (*MedLine*, 2006). Esta base de dados contém vários artigos de investigação relativos a esta área da biomedicina.

O sistema é constituído por dois módulos, sendo o primeiro a aplicação de um parser superficial ao documento com o objectivo de identificar apenas sintagmas nominais (SN), pois é nestes sintagmas que está a informação relevante para o sistema.

O segundo módulo consiste na extracção de termos, usando para isso dicionários e um conjunto de regras. São usados três dicionários, UMLS (*Unified Medical Language System*, 2006), LocusLink (*LocusLink: Gene Database*, 2006), e GeneAlias. O UMLS é um repositório de termos médicos e relações entre eles, que contém também a classe semântica de cada termo, tal como gene, proteína ou aminoácido. O LocusLink contém uma listagem de nomes de genes de vários organismos, sendo a principal fonte para a anotação dos nomes de genes. O último, GeneAlias, é uma lista local de nomes de genes que não estão presentes no dicionário anterior.

O módulo de extracção de termos inicia-se com a procura nos dicionários de cada um dos SN detectados anteriormente. Caso não encontre informação relevante, remove as palavras funcionais (designadas por *stop-words*) do SN, recorrendo a uma lista de palavras funcionais. É feita nova procura nos dicionários. Se após esta procura não houver nenhum emparelhamento, então é realizada outra procura, desta vez com cada uma das palavras do SN.

Outro mecanismo que este sistema usa para extrair nomes é a utilização de regras. Estas são usadas pois os dicionários não contêm todos os termos existentes, e também porque os termos biológicos, na sua maioria, respeitam um padrão específico. Foram criados três tipos de regras (em ficheiros XML contendo expressões regulares) para extrair termos e para classificar esses termos:

- O primeiro tipo considera que muitos termos biológicos são constituídos por maiúsculas, números e caracteres não alfabéticos, conseguindo assim anotar termos como "*PTEN*", "*c-N-ras*" ou "*T-Lymphocytes*". Usa outra regra para excluir termos que foram anotados incorrectamente, tal como

10000, "*H.D.Smith*", ou outros nomes próprios.

Os dois últimos tipos de regras dizem respeito à classificação dos termos:

- Detecção de algumas palavras que precedem ou que se seguem aos termos biológicos, dando assim uma indicação da sua classe. Exemplos de frases detectadas por esta regra são "*p16 tumor supressor gene*", "*proteins Ra1 and Cdc42*".
- Utilização de uma lista de prefixos e sufixos que dão uma indicação muito provável da classe do termo. Por exemplo, a maior parte das proteínas têm o sufixo "-ase".

Na classificação dos termos, o sistema tem em conta os dicionários utilizados. Se o termo é identificado através do UMLS, então a classe semântica obtida é associada ao termo. Se é identificado através do LocusLink ou GeneAlias, o termo obtém a classe semântica gene, pois estes dicionários só contêm genes.

Uma outra característica deste sistema é a capacidade para realizar a aprendizagem de afixos, que permite construir a lista de afixos para a última das regras descritas anteriormentes. A ideia base é que os termos biológicos normalmente contém prefixos e sufixos que indicam a sua classe, como é o exemplo das proteínas, que normalmente terminam em "-ase".

A aprendizagem é feita através do UMLS. Os termos do UMLS são agrupados em classes (proteína, gene, etc.) e a detecção de afixos é feita em quatro passos. Inicialmente há um pré-processamento dos nomes, em que é removida a pontuação, números e palavras duplicadas. De seguida, realiza-se a identificação de prefixos, em que a lista de palavras criadas no passo anterior é colocada numa árvore de prefixos (*trie*), como a da figura 2.1, e através desta são retirados os prefixos comuns. A identificação de sufixos decorre de uma forma semelhante, mas colocando as palavras na árvore de uma forma inversa. No último passo são removidos os prefixos e sufixos que são comuns ao inglês e que não têm importância no domínio da biologia, usando para tal uma lista de afixos de palavras inglesas com base num corpus, e removem-se estes afixos da lista de afixos descobertos nos passos anteriores. Desta forma, obtêm-se os afixos necessários para uma das regras aplicadas neste sistema.

2.1.2 PROPER

O objectivo do sistema PROPER (Fukuda et al., 1998) é detectar e identificar nomes de genes e proteínas em documentos biológicos e médicos. O sistema tem como características a não utilização de dicionários, e ser unicamente baseado em informação contextual.

É feita uma distinção entre alguns termos, os chamados *core-terms*, que contêm uma grande quantidade de informação, e que podem ser considerados como candidatos a termos biológicos. Estes *core-*

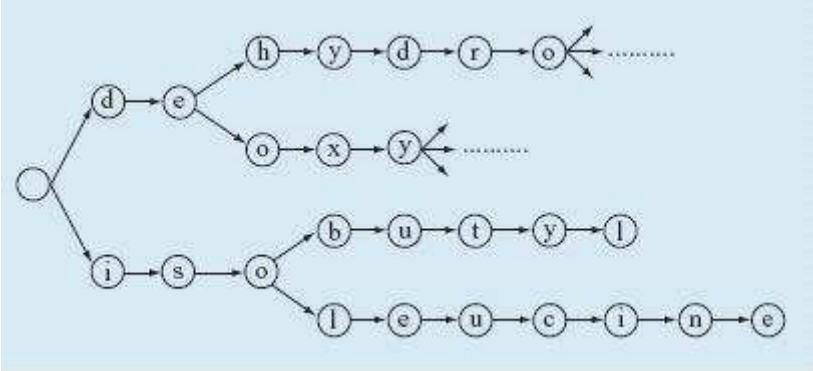

Figura 2.1: Exemplo de uma árvore de prefixos para os prefixos *dehydro* e *deoxy*, e para as palavras *isobutyl* e *isoleucine*.

terms são apresentados de seguida, entre ≪≫:

- "*≪Src≫ homology (≪SH≫) 2 and ≪SH3≫ domains*";
- "*≪p54≫ ≪SAP≫ kinase*".

Existem também os *feature-terms* (adiante designado por *f-term*), que são expressões que descrevem as funções e características dos core-terms.

- "*EGF ≪receptor≫*";
- "*Ras GTPas-activating ≪protein≫*".

O sistema é constituído por dois módulos, o primeiro módulo que é o da extracção de *core-terms*, e o segundo que realiza a concatenação de *core-terms*. O primeiro módulo tem uma sequência de processamento com cinco passos, correspondentes à aplicação de cinco filtros diferentes:

- Filtro que extrai palavras com maiúsculas, números e símbolos especiais e anota-os como candidatas a *core-terms*;
- Filtro que remove palavras com comprimento maior que 9 caracteres com um "-" e minúsculas;
- Filtro que remove palavras que mais de metade do comprimento são símbolos especiais;
- Filtro que remove palavras referentes a unidades;
- Filtro que remove palavras que são nomes de pessoas ou artigos nas referências.

As palavras candidatas que passarem estes filtros são anotadas como *core-terms*.

No segundo módulo é feita uma concatenação entre *core-terms*, ou seja, a anotação realizada no módulo anterior é estendida para palavras adjacentes ou concatenada com outras anotações. Os sintagmas nominais sem conjunções nem preposições (nomeados como *core-blocks*) são reconstituídos através das seguintes regras de concatenação, que consistem em pistas contextuais:

- Ligação de anotações quando há *core-terms* e *f-terms* adjacentes:
  - "*≪Src≫ ≪SH3≫ ≪domain≫*" passa a "*≪Src SH3 domain≫*";
- Anotação de parênteses nas seguintes situações:
  - "*(≪SH3≫)*" passa a "*≪(SH3)≫*";
  - "*(≪SH2≫ (and|or) ≪SH3≫)*" passa a "*≪(SH2 (and|or) SH3)≫*".

Os *core-blocks* são concatenados usando também um etiquetador POS com três regras:

- Ligação de anotações não adjacentes, se todas as palavras entre elas forem nomes, adjectivos ou números:
  - "*≪Ras≫ guanine nucleotide exchange ≪factor≫ ≪Sos≫*" passa a "*≪Ras guanine nucleotine exchange factor Sos≫*";
- Extensão da anotação para a esquerda se há um determinante ou preposição:
  - "*The focal adhesion ≪kinase≫ (≪FAK≫)*" passa a "*The ≪focal adhesion kinase (FAK)≫*";
- Extensão da anotação para a direita se houver uma letra maiúscula ou uma letra grega:
  - "*≪p85≫ alpha*" passa a "*≪p85 alpha≫*";

De seguida são reconstruídas as dependências entre *core-blocks* usando para o efeito um conjunto de regras que estendem a anotação entre *core-blocks*, como, por exemplo, na presença de "*A, B, ... C and D f-term*", em que A, B, C e D são *core-terms*, toda a frase é anotada.

O último passo consiste na desmarcação de algumas anotações realizadas no passo anterior e que podem ser consideradas não necessárias. Mais uma vez, são usadas regras, neste caso para quando um *f-term* anotado não foi estendido e permanece como uma palavra única, e para quando a última palavra da frase obtida na fase de extensão e concatenação não for um nome, pois pode acontecer anotar um *core-term* que não seja nome, por exemplo "*Src-related*". Nestas situações, a anotação é removida.

2.1.3 EDGAR

O sistema EDGAR (Rindflesch et al., 2000) tem como objectivo extrair informação relativa a medicamentos e genes relevantes para o cancro partindo da literatura biomédica, possibilitando a identificação de nomes de genes, células e medicamentos, e eventualmente determinar as relações entre estes. No final, o sistema produz uma lista de predicados contendo as relações e os termos identificados.

O corpus utilizado pelo sistema EDGAR consiste nos resumos dos artigos que se encontram na base de dados MedLine (*MedLine*, 2006), que dizem respeito a relações entre genes, medicamentos e células.

O sistema é constituído por quatro módulos, iniciando-se com uma análise sintáctica de cada frase, recorrendo a um etiquetador estocástico, de forma a resolver possíveis ambiguidades. Desta forma, segmentos das frases ficam identificadas como sintagmas nominais, verbais, sintagmas preposicionais, entre outros.

De seguida, identificam-se os sintagmas nominais que são referentes a genes e células, através da detecção de algumas palavras sinalizadoras, tal como *cell*, *clone*, *line* para células, e *activated*, *expression* e *gene* para genes. Ao encontrar um SN com uma destas palavras, esse SN é marcado como sendo referente a células ou genes. Esta forma de identificação é possível, visto que a estrutura dos sintagmas nominais presentes nos textos é muito regular.

No terceiro módulo é feita uma identificação dos nomes de genes e células em cada um dos sintagmas nominais, usando uma sequência de dicionários. Os dicionários usados são:

- UMLS (*Unified Medical Language System*, 2006), que contém uma compilação de termos usados no domínio biomédico, onde cada termo está associado a um tipo semântico, tal como produto farmacêutico, gene ou genoma, e célula;
- lista auxiliar de genes e células que não estão presentes no UMLS;
- lista local de genes e células, determinados no módulo anterior através da análise contextual.

Cada sintagma nominal é passado sequencialmente por estes três dicionários até emparelhar com alguma entrada. Um exemplo da saída gerado por este módulo está na figura 2.2, tendo em conta a frase "*This effect of cyclosporin A or herbimycin A on the down-regulation of ERCC-1 correlates with enhanced cytotoxicity of cisplatin in this system*".

Por fim, o sistema tenta identificar características das células e possíveis relações entre estas, extraindo dados sobre o tipo de orgão, tipo de cancro e organismo, bem como características específicas do domínio. Estes dados e características são obtidos com técnicas de análise contextual, em que algumas palavras fornecem esta informação (tal como *transfected* e *resistant*). Exemplos de predicados extraídos pelo sistema estão na figura 2.3.

```
[of cyclosporin A] - Cyclosporine (Pharmacologic Substance) - UMLS
[herbimycin A] - herbimycin (Pharmacologic Substance) - UMLS
[of ERRCC-1] - ERCC-1 (Gene) - Lista auxiliar
```

Figura 2.2: Tipos semânticos detectados no sistema EDGAR e dicionário de origem.

```
gene_np(['v-src', transfected, cells])
gene_np([activated, src])
cell_np(['hag/src3-1', cells])
drug('99140404', 'Cisplatin')
gene('99140404', 'h-ras')
cell('99140404', 'HAG-1', 'Gallbladder', (...), 'Human')
```

Figura 2.3: Predicados extraídos do sistema EDGAR.

2.1.4 FASTUS

O sistema FASTUS (Hobbs, 2002) tem como objectivo detectar nomes de enzimas, as suas propriedades e as relações entre enzimas. O sistema usa a tecnologia de transdutores de estados finitos em cascata, em que cada uma das fases é alimentada pela fase anterior, e, no estado final (quando a sequência de entrada corresponder a um determinado padrão) de cada fase é construída uma estrutura que representa a informação presente numa frase. A representação final do sistema consiste em estruturas semânticas que contêm informação sobre as entidades e relações entre enzimas. O sistema é constituído por quatro módulos:

- Reconhecimento de palavras compostas, onde se detectam palavras complexas (correspondentes a nomes ou expressões) tal como "*gamma-Glutamyl proline*", "*3,4-dehydroproline*" ou "*molecular weight*";
- Análise sintáctica, que está dividida em dois passos:
  - Detecção de sintagmas básicos, ou seja, as frases são segmentadas (emphchunking) em sintagmas nominais, verbais, etc.;
  - Identificação de grupos complexos, onde se identificam grupos de sintagmas simples detectados na fase anterior, tendo em conta o domínio que se está a tratar. Um exemplo é a junção de dois SN num único grupo ("≪*the proline analog*≫ ≪*3,4-dehydroproline*≫"), sendo cada um deles representado entre ≪ e ≫. Neste módulo já se começam a construir as estruturas de saída, contendo as entidades e algumas relações. Por exemplo, para o grupo complexo *"gamma-Glutamyl kinase, the first enzyme of the proline biosynthetic pathway"* são construídas as estruturas representadas na figura 2.4.

```
Reaction:
  ID: R1
  Pathway: proline
  Enzyme: E1

Enzyme:
  ID: E1
  Name: gamma-Glutamyl kinase
  Molecular-Weight: -
  Subunit-Component: -
  Subunit-Number: -
```

Figura 2.4: Estruturas criadas no segundo módulo do sistema FASTUS.

- Aplicação de padrões do domínio ao texto de forma a tentar obter mais informação sobre as enzimas. Um exemplo de um padrão é "*<Compound> have <Measure> of <values>*", que permite preencher o campo "Molecular-Weight" das estruturas. Também tem padrões que consideram a forma passiva das frases, associando-as à forma activa, tal como "*<Protein> inhibits <Reaction>*", e "*<Reaction> is inhibited by <Protein>*". Exemplificando para a frase "*The enzyme had a native molecular weight of 236,000 and was apparently comprised of six identical 40,000-dalton subunits.*", são criadas as estruturas da figura 2.5;

- Fusão de todas as estruturas criadas, tendo em conta todo o texto de entrada. A informação que diz respeito à mesma entidade é associada na mesma estrutura semântica. Os critérios que são tidos em conta para determinar se duas estruturas podem ser ligadas são a estrutura interna dos sintagmas nominais e a consistência entre as estruturas. Aplicando esta colapsação às estruturas das figuras 2.4 e 2.5, obtém-se uma nova estrutura, representada na figura 2.6, juntamente com a estrutura com ID igual a E4 (na figura 2.5, que não pode ser ligada, por ser inconsistente com as restantes).

2.1.5 Sistema de Proux, Rechenmann e Julliard

O sistema desenvolvido por Proux et al. (Proux et al., 1998) pretende fazer uma identificação de genes da *Drosophila*. Tal como o sistema FASTUS da secção 2.1.4, é também baseado na tecnologia de estados finitos em cascata. O corpus que o sistema utiliza é extraído de uma base de dados com informação molecular e genética respeitante unicamente à Drosophila, a FlyBase (The FlyBase Consortium, 1997; *FlyBase Database*, 2006). Mais concretamente, o sistema usa um campo da base de dados que contém pequenos resumos de artigos biológicos relativos a genes específicos da Drosophila.

O sistema é constituído por dois módulos, tendo cada uma destas sub-módulos. O primeiro módulo consiste numa análise sintáctica com o objectivo de anotar candidatos a nomes de genes, sendo constituído pelos seguintes sub-módulos:

```
Enzyme:
  ID: E2
  Name: -
  Molecular-Weight: 236,000
  Subunit-Component: -
  Subunit-Number: -

Enzyme:
  ID: E3
  Name: -
  Molecular-Weight: -
  Subunit-Component: E4
  Subunit-Number: 6

Enzyme:
  ID: E4
  Name: -
  Molecular-Weight: 40,000
  Subunit-Component: -
  Subunit-Number: -
```

Figura 2.5: Estruturas criadas no terceiro módulo do sistema FASTUS.

```
Enzyme:
  ID: E1
  Name: gamma-Glutamyl kinase
  Molecular-Weight: 40,000
  Subunit-Component: E4
  Subunit-Number: 6
```

Figura 2.6: Estrutura final do sistema FASTUS.

- Pré-processamento do texto para facilitar a tarefa de tokenização, tal como a expressão "*20-30ºC*", que é reescrita como "*20 to 30 Celsius degrees*";
- Aplicação de um etiquetador, que é um autómato de estados finitos não determinístico que realiza três funções, tokenização, pesquisa lexical e desambiguação:
  - Na tokenização, o texto é separado em símbolos;
  - A pesquisa lexical usa um transdutor de estados finitos para realizar a análise morfológica, atribuindo assim uma ou mais categorias gramaticais aos símbolos encontrados;
  - O desambiguador, como o próprio nome indica, pretende realizar a desambiguação morfológica, ou seja, escolher a categoria gramatical correcta para o símbolo, tendo em conta as palavras que estão à sua volta. Este desambiguador é um programa baseado nos *Hidden Markov Models*;
- Aplicação de regras de recuperação, usando dicionários e regras de prefixos e sufixos, com o objectivo de recuperar erros gerados pelo etiquetador. O uso do dicionário neste módulo tem como objectivo remover alguns termos como nomes de espécies, unidades, ou nomes de proteínas, que foram incorrectamente anotados. São também removidos nomes de genes que não pertecem ao domínio considerado. Relativamente à utilização de regras, estas consistem em regras especiais, de detecção de sufixos e de detecção de prefixos, sendo aplicadas por esta ordem:
  - As regras especiais detectam expressões complexas que envolvem sequências de caracteres como DNA ("*TCAATTAGT*") ou notação de péptidos ("*VGIDLGTYSC*");
  - As regras de sufixos têm como objectivo a remoção de nomes incorrectamente anotados como genes. São removidas as palavras que tenham sufixos de verbos ("*-ed*" ou "*-ing*"), advérbios ("*-ly*"), nomes ("*-ation*"), proteínas ("*-ase*" ou "*-one*"), etc;
  - As regras de prefixos pretendem recuperar certos termos que não foram anotados nos módulos anteriores com base em prefixos. Os termos removidos são, na maioria, compostos químicos ("*phosphoribosylaminoimidazole*") e biológicos ("*orthologue*"). As regras de sufixos são aplicadas antes das regras de prefixos pois um sufixo tem mais informação gramatical associada que um prefixo, ou seja, a categoria gramatical depende dos sufixos e não dos prefixos.

No final deste módulo é criada uma representação das frases, com a indicação dos candidatos a nomes de genes.

O segundo módulo consiste numa análise contextual, que pretende validar os candidatos como nomes de genes, usando para isso padrões de validação. Os candidatos que estejam antes ou depois de expressões específicas podem ser aceites ou rejeitados. Por exemplo, na frase "*Antp and esp genes*

activate..." as palavras "*antp*" e "*esp*" estão juntas à palavra "*gene*", e são então validadas como nomes de genes.

De seguida é feita uma reformatação da representação de entrada, contemplando as validações efectuadas, de forma a poder ser aplicada posteriormente a mais análises semânticas ou gramaticais.

2.1.6 Sistema de Hagège e Brun

O sistema construído por Hagège e Brun (Hagège & Brun, 2003) tem como objectivo a identificação de produtos tóxicos, bem como as suas propriedades. Resulta deste sistema uma listagem de predicados referentes às propriedades deste produtos. Estes predicados são:

- SUBSTANCE - indica o nome do produto;
- PHYS_FORM - indica o estado físico do produto;
- DESCRIPTION_COLOR - indica a cor do produto;
- DESCRIPTION_SMELL - indica o cheiro do produto;
- SYNONYM - estabelece a relação de sinonímia entre dois nomes;
- PROPERTY - indica propriedades físicas ou químicas do produto;
- ORIGIN - indica se o produto é natural ou não;
- USE - indica a utilização que o produto pode ter.

A estes predicados pode ser adicionado o sufixo "_NEG", para indicar a negação do predicado.

O sistema efectua o processamento de um corpus que consiste num conjunto de textos disponibilizados na Internet pela ATSDR (*Agency for Toxic Substances and Disease Registry*, 2006). Estes textos contêm a apresentação de produtos tóxicos, bem como das características que possuem.

O sistema é constituído por três módulos, um analisador geral de inglês, um normalizador morfo-sintáctico, e um normalizador específico do domínio.

O primeiro módulo, o analisador de inglês, extrai dependências entre nós lexicais, usando para o efeito o sistema XIP (Xerox Research Centre Europe, 2003). As acções realizadas incluem a tokenização, análise morfo-sintáctica, etiquetação realizada com regras e *Hidden Markov Models*, segmentação e a extracção de dependências. Estas dependências são obtidas através de regras. As regras consistem em três partes, o contexto (expressão regular que tem de ser emparelhada para se aplicar a regra), a condição (comparação das características associadas a nós), e a extracção (lista de dependências a criar

```
| SN{?*, #1[last:+]}, ?*[verb:~], SV{?*, #2[last:+]} |
if (~SUBJ(#2, #1))
 SUBJ(#2, #1)
```

Figura 2.7: Regra de dependência usada no sistema de Hagège e Brun.

caso se verifiquem as descrições do contexto e da condição). Na figura 2.7 está um exemplo de uma regra.

A regra da figura 2.7 associa uma dependência SUBJ (sujeito) entre a cabeça do SN e o verbo. A primeira linha, o contexto, contém a expressão regular que tem de ser emparelhada, neste caso é um SN (sintagma nominal) seguido de ou ou mais segmentos que não são verbos, e de um SV (sintagma verbal). A segunda linha tem a condição, que testa se já existe uma dependência SUBJ entre as duas palavras (o verbo e a cabeça do SN). A última linha realiza a extracção, neste caso, cria a dependência SUBJ.

No segundo módulo, normalizador morfo-sintáctico, é feita uma análise sintáctica profunda, obtendo assim novas relações sintácticas através da aplicação de regras. Estas regras são aplicadas à representação produzida pelo analisador anterior. Neste módulo é realizada também uma normalização entre a forma passiva e a forma activa, dando origem às relações OBJ-N (objecto normalizado) e SUBJ-N (sujeito normalizado).

O último módulo é o único que é específico ao domínio (os dois anteriores são gerais ao inglês). Este módulo aplica um conjunto de regras à representação obtida no módulo anterior, de forma a retirar as propriedades dos produtos, e contém certas características que melhoram o desempenho geral do sistema, que são:

- Utilização de regras de tokenização específicas, para considerar como um único símbolo produtos do género de *2,3-Benzofurano*;
- Utilização de regras de desambiguação, para remover categorias gramaticais associadas a palavras, no caso do seu significado não fazer sentido no domínio (como "*sharp*", que é simultaneamente um verbo e um nome, mas o nome pertence ao domínio da música);
- Utilização de regras para melhorar o tratamento das coordenações, de forma a tratar as situações em que o último termo da coordenação é precedido pela vírgula e pelo coordenador;
- Adição de semântica lexical, em que se adicionam características a certas palavras que facilitam a detecção de parafrases (tal como a adição do traço "*colour : +*" às palavras que designem cores);
- Resolução de anáforas simples, relativa às palavras "*it*" e "*its*", que vão sempre referir-se ao produto tóxico que está a ser descrito no texto, havendo apenas um produto descrito em cada texto.

```
if ( SUBSTANCE(#1) & ATTRIB(#1, #8[adj_property])
                  & ISAJ(#9, #10)
                  & #8[lemme]:#9[lemme] )
  PROPERTY(#1, #10, ##Pron[lemme=NONE],
                    ##Pron[lemme=NONE],
                    ##Pron[lemme=NONE])
```

Figura 2.8: Regra usada no sistema de Hagège e Brun.

Acetone is a manufactured chemical that is also found naturally in the environment. It is a colorless liquid with a distinct smell and taste. It evaporates easily, is flammable, and dissolves in water. It is also called dimethyl ketone, 2-propanone, and beta-ketopropane. Acetone is used to make plastic, fibers, drugs, and other chemicals. It is also used to dissolve other substances. It occurs naturally in plants, trees, volcanic gases, forest fires, and as a product of the breakdown of body fat. It is present in vehicle exhaust, tobacco smoke, and landfill sites. Industrial processes contribute more acetone to the environment than natural processes.

Figura 2.9: Exemplo de texto de entrada do sistema de Hagège e Brun.

De seguida realiza-se a detecção de parafrases. Este aspecto tem uma elevada importância pois os textos não estão todos normalizados e as maneiras de descrever os produtos não são sempre iguais nos textos de entrada, sendo, portanto muito variadas. São usadas regras que necessitam de informação lexical e estrutural. As regras permitem ligar palavras pertencentes a categorias morfológicas diferentes, mas que conduzem ao mesmo significado, tal como o adjectivo "*flammable*" e o verbo "*burn*". Para isso são criadas algumas relações:

- A relação ISAJ, que liga um adjectivo e um verbo quando o verbo pode ser descrito como be+adjectivo (como "*flammable*" e "*burn*");
- A relação TURNTO, que liga um nome a um verbo quando o verbo pode ser descrito como turn to+nome (como "*liquid*" e "*liquefy*");
- A relação HASN, que liga um nome a um verbo quando o verbo pode ser descrito como have+noun (como "*volatility*" e "*evaporate*");
- A relação SYNO, que liga 2 palavras com o mesmo significado (como "*call*" e "*name*").

A partir destas relações constroem-se as regras para produzir os predicados, tal como a regra da figura 2.8, que permite atribuir a uma substância (na variável #1) uma propriedade (em #10).

Tendo em conta o texto de entrada da figura 2.9, o sistema extrai a lista de predicados apresentada na figura 2.10.

A maioria da informação presente no texto de entrada da figura 2.9 foi extraída correctamente, no entanto, há situações em que só é extraída parte da informação, e outras em que se extrai a informação incorrecta.

```
SUBSTANCE(acetone)
PHYS_FORM(acetone,chemical)
PHYS_FORM(acetone,liquid)
DESCRIPTION_COLOUR(acetone,colorless)
DESCRIPTION_SMELL(acetone,distinct)
PROPERTY(acetone,burn,NONE,NONE,easily)
PROPERTY(acetone,evaporate,NONE,NONE,easily)
PROPERTY(acetone,dissolve,water,in,NONE)
ORIGIN(acetone,natural,vehicle exhaust,in)
ORIGIN(acetone,natural,tobacco smoke,in)
ORIGIN(acetone,natural,landfill site,in)
ORIGIN(acetone,natural,plant,in)
ORIGIN(acetone,natural,the environment,in)
ORIGIN(acetone,man-made,NONE,NONE)
ORIGIN(acetone,natural,tree,in)
ORIGIN(acetone,natural,volcanic gas,in)
ORIGIN(acetone,natural,forest fire,in)
ORIGIN(acetone,natural,a product,in)
SYNONYM(acetone,dimethyl ketone)
SYNONYM(acetone,beta-ketopropane)
SYNONYM(acetone,2-propanone)
USE(acetone,NONE,NONE,make,plastic,present)
USE(acetone,NONE,NONE,make,fiber,present)
USE(acetone,NONE,NONE,make,drug,present)
USE(acetone,NONE,NONE,make,other chemical, present)
USE(acetone,NONE,NONE,dissolve, other substance,present)
```

Figura 2.10: Exemplo de uma lista de predicados criada pelo sistema de Hagège e Brun.

2.1.7 Sistema de Collier, Nobata e Tsujii

O sistema desenvolvido por Collier et al. (Collier et al., 2000) na Universidade de Tóquio tem o objectivo de extrair os nomes de genes, usando para tal *Hidden Markov Models*. Usa um corpus anotado de 100 artigos da MEDLINE (*MedLine*, 2006) para o teste, com as classes proteína, DNA, RNA, *source* (com diversas subclasses) e desconhecido, que são consideradas as classes de interesse para o domínio.

O módulo inicial do sistema consiste no cálculo da sequência mais provável das classes dos nomes, ou seja, encontrar a probabilidade de cada classe dadas as palavras, $P(C|W)$, usando para o efeito uma implementação de *Hidden Markov Model*, e assumindo que a probabilidade pode ser encontrada através de bigramas das classes dos nomes.

Este sistema considera um par constituído pela palavra e por uma característica da palavra (representados por <W,F>). A característica pode ser uma entre várias, tal como a presença de números, letras gregas, combinações de letras e números, minúsculas e caracteres de pontuação. Estas características das palavras fornecem evidências que ajudam a distinguir as classes de nomes das palavras, podendo ajudar a encontrar semelhanças entre palavras conhecidas e desconhecidas. Por exemplo, sabendo que *LMP - 1* é uma proteína, e, encontrando *AP - 1* no corpus de teste, pode-se fazer uma suposição de que *AP - 1* também é uma proteína, dado que partilham a mesma característica relativa aos caracteres maiúsculos.

De seguida calcula-se a probabilidade das classes de nomes para uma sequência de palavras, tendo em conta que se usam bigramas, portanto, a primeira palavra da sequência terá uma forma de cálculo diferente das restantes. A equação 2.1 calcula a probabilidade da classe da palavra inicial.

$$\begin{aligned} &P(C_t \mid < W_{first}, F_{first} >) = \\ &\quad \sigma_0 f(C_{first} \mid < W_{first}, F_{first} >) + \\ &\quad \sigma_1 f(C_{first} \mid < -, F_{first} >) + \\ &\quad \sigma_2 f(C_{first}) \end{aligned} \tag{2.1}$$

Para as restantes palavras, o sistema usa a equação 2.2.

$$\begin{aligned} &P(C_t \mid < W_t, F_t >, < W_{t-1}, F_{t-1} >, C_{t-1}) = \\ &\quad \lambda_0 f(C_t \mid < W_t, F_t >, < W_{t-1}, F_{t-1} >, C_{t-1}) + \\ &\quad \lambda_1 f(C_t \mid < -, F_t >, < W_{t-1}, F_{t-1} >, C_{t-1}) + \\ &\quad \lambda_2 f(C_t \mid < W_t, F_t >, < -, F_{t-1} >, C_{t-1}) + \\ &\quad \lambda_3 f(C_t \mid < -, F_t >, < -, F_{t-1} >, C_{t-1}) + \\ &\quad \lambda_4 f(C_t \mid C_{t-1}) + \\ &\quad \lambda_5 f(C_t) \end{aligned} \tag{2.2}$$

A probabilidade de uma palavra com uma certa característica ter a classe C_t é dada pela função f com as combinações de bigramas possíveis. Por exemplo, a parte da equação 2.2 que se inicia com λ_1, refere-se a um conjunto de duas palavras em que à palavra de ordem t só interessa a sua característica (F_t), podendo ser uma palavra qualquer. O mesmo se passa nas outras partes da equação 2.2. A função f é simplesmente uma estimativa de probabilidade partindo das contagens no corpus de treino. Nas equações 2.1 e 2.2 σ e λ são constantes, e $\sigma_0 \geq \sigma_1 \geq \sigma_2$, $\lambda_0 \geq \lambda_1 \ldots \geq \lambda_5$. Ou seja, é dada mais evidência aos factos (palavras) que são "visíveis" no corpus de treino que aos que são obtidos quando não se tem em conta a palavra original.

Depois de se ter calculado estas probabilidades, o sistema usa o *algoritmo de Viterbi* para procurar no espaço de estados as possíveis atribuições de classes aos nomes. Isto é, procura a maior probabilidade, maximizando assim $P(W, C)$.

Após a etiquetação dos nomes com as classes, o sistema efectua uma limpeza dos documentos, que consiste em criar uma lista de frequência de palavras e classes de um documento e, de seguida, etiquetar de novo o documento usando a classe mais frequente para a palavra descoberta pelo HMM. Esta limpeza permite melhorar o desempenho do sistema em geral.

2.1.8 Avaliação

Como se pode ver na tabela 2.1, que contém os resultados reportados pelos autores dos sistemas, três dos sistemas analisados conseguiram bons resultados, com precisão acima dos 90%, e dois destes com cobertura acima de 90%. O sistema PROPER (descrito na secção 2.1.2) consegue ter bons resultados, apesar de apenas realizar análise sintáctica. No entanto, estes valores são muito subjectivos, visto que todos os sistemas usam corpus diferentes (apesar da maioria usar a mesma fonte), pelo que não é possível efectuar comparações entre eles.

Sistema	Precisão	Cobertura
BioAnnotator	60.3%	68.6%
PROPER	94.7%	98.84%
EDGAR	n/d	n/d
FASTUS	60%	60%
Sistema de Proux et al.	91.4%	94.4%
Sistema de Hagège e Brun	96%	77%
Sistema de Collier et al.	n/d	n/d

Tabela 2.1: Avaliação dos Sistemas de Biologia

2.2 Extracção de Informação em Textos Desportivos

A quantidade de sistemas que pretendem extrair informação de textos desportivos é muito menor do que os realizados para o domínio da biologia, mas tem também algum interesse no domínio da extracção de informação.

Nesta secção (nas secções 2.2.1 e 2.2.2) descrevem-se dois sistemas que extraem os resultados de jogos de futebol e hóquei na língua inglesa, espanhola e sueca, tendo como base apenas uma pesquisa de padrões que ocorrem normalmente nos textos. Uma das maiores dificuldades é a variedade de formas como o conjunto resultado/nome das equipas pode ser apresentado nos textos, tendo os sistemas de lidar com essa situação.

2.2.1 Sistema de Extracção em Futebol

O sistema desenvolvido por Llobera et al. (Llobera et al., 2003) tem como objectivo a extracção do vencedor, do derrotado, do resultado e do local de encontro de textos jornalísticos sobre futebol, permitindo não só textos na língua inglesa, mas também de espanhol. Este sistema é constituído por cinco módulos, sendo estas tokenização, separador de frases, analisador de frases, pesquisa de padrões e pesquisa de dados relevantes (análise de uma tabela).

O primeiro módulo consiste na detecção de símbolos do texto de entrada, usando para isso os mecanismos de tokenização existentes no Java (pertencentes à classe *StringTokenizer*). Adicionalmente, para o espanhol, são substituídos todos os caracteres acentuados pelos equivalentes não acentuados.

De seguida, todas as frases são separadas, tendo em conta que uma frase são todos os caracteres presentes entre dois delimitadores. Por delimitadores compreendem-se os seguintes caracteres: “.”, “,”, “?”, “!”, “;” e “:”. Cada frase é depois guardada numa lista de frases.

No terceiro módulo, o analisador de frases é chamado para cada elemento da lista preenchida no módulo anterior, que contém uma representação de todas as frases do texto de entrada. Para cada linguagem tem uma lista de verbos, preposições e conjunções, as quais são designadas por palavras delimitadoras. O objectivo é, então, dividir a frase em grupos, sendo a separação feita por estas palavras. Em cada grupo é identificado o verbo, o sujeito, o objecto e sintagmas preposicionais. Para o caso de se analisar a frase “*10 November 2003 Blackburn Rovers FC ended a run of five consecutive Premiership defeats with a nervy 2-1 win against Everton FC at Ewood Park tonight.*”, obtém-se os grupos da figura 2.11. A informação presente na figura 2.11 é guardada num objecto *Phrase*.

Depois de cada frase ser analisada e dividida segundo os critérios referidos, realiza-se uma pesquisa de padrões relevantes. Estes padrões consistem numa lista de palavras ou frases que são prováveis de

- Subject: “10 November 2003 Blackburn Rovers FC ended a run”
- Verb: “win”
- Object: [NULL]
- PP:
  - “of five consecutive Premiership”
  - “with a nervy 2-1”
  - “against Everton FC at Ewood Park tonight”

Figura 2.11: Grupos obtidos no terceiro módulo do sistema de extracção em futebol.

ocorrer, e são compostos por vários tipos de padrões, contendo padrões para verbos e palavras relevantes, para locais e para resultados, usando para tal a classe *Pattern* do Java:

- Para detectar o nome das equipas são usados os seguintes padrões, entre outros,

  ```
  [A-Z]\\w*, [A-Z]\\w* [A-Z][A-Z][A-Z]*;
  ```

- A determinação do vencedor é mais complexa, pois são considerados cinco tipos de verbos: verbos acerca de ganhar e perder na forma activa e na passiva, e também verbos acerca do empate. Normalmente, o nome do vencedor ou do derrotado está no grupo anterior ou no posterior ao que contém o verbo, dependendo da forma passiva ou activa. Por exemplo, na presença de uma frase na passiva que contém um verbo relativo a uma vitória (“*has been won*”), é muito provável encontrar o nome do vencedor após o verbo;

- Para determinar onde o jogo foi realizado, o sistema apenas tem em conta os grupos que contém a presença de duas preposições relativas à localização (“*at*” e “*in*”) no caso do inglês, e apenas uma preposição no caso do espanhol (“*en*”). Permite encontrar frases como “*The match was won by Barcelona in Camp Nou*”, mas, por outro lado, também detecta frases como “ *The match was won by Barcelona in September*”. Por esta razão, existe uma lista de certas palavras (com meses, dias da semana, etc.) de forma a rejeitar as frases detectadas contendo estas palavras;

- O padrão para determinar os resultados é muito simples. Consiste na pesquisa de padrões da forma “Number Union Number”, onde *Number* é qualquer número, e *Union* é uma palavra ou caracter que aparece entre os dois números, tal como “-”, “*to*”, “*against*”, “*by*” e “:”. De forma a obter apenas o resultado final, pois no texto podem estar descritos resultados intermédios, só se considera o resultado que tenha os maiores números.

No último módulo é preparada a informação final sobre os jogos através de uma análise de uma tabela de jogos (classe *Match*). Cada instância de *Match* contém apenas a informação acerca do jogo que

está presente em cada frase, daí a necessidade deste módulo, cujo objectivo é unir estas instâncias, de forma a obter uma instância por jogo. Este módulo consiste em dois passos, sendo o primeiro a extracção de todos os nomes de equipas, e o segundo é unir os jogos tendo em conta os nomes das equipas. No primeiro passo, são pesquisados os campos vencedor e derrotado dos jogos para obter os nomes das equipas. No segundo passo unem-se jogos que não tenham contradições entre si. Portanto, um jogo com vencedor "Barcelona" e outro com vencedor "Real Madrid" não podem ser unidos.

No final, obtem-se uma instância da classe *Match* por cada jogo que o texto relate, com informações sobre o vencedor, o derrotado, ou empate, o resultado e o local onde se realizou o jogo.

2.2.2 Sistema de Extracção em Futebol e Hóquei

O sistema desenvolvido por Lindvall e Nilssonn (Lindvall & Nilsson, 2002) pretende extrair informação de notícias desportivas de jornais suecos, mais concretamente, reportagens de jogos de futebol e de hóquei no gelo. Foca-se sobretudo em obter a informação relevante ao invés de construir uma gramática que resultaria numa compreensão mais profunda dos textos. Este sistema extrai o nome das equipas, o resultado final, e o nome da equipa vencedora, caso haja. A estratégia passa pelo reconhecimento de padrões simples.

Este sistema é composto por quatro módulos, que são a detecção de padrões chave, o módulo da lógica local, o módulo da lógica global e um último módulo de limpeza.

No primeiro módulo é feita uma detecção de padrões relevantes para este domínio, havendo especialmente quatro padrões relevantes, relativos aos resultados, nomes das equipas e palavras que referenciam vitória, empate ou derrota:

- O padrão mais simples é o do empate, pois consiste apenas na detecção de um verbo ("*oavjgort*"), nas suas diferentes flexões verbais;
- O padrão das equipas é composto simplesmente pelo nome da equipa, que pode consistir em:
  - um prefixo, uma localização e um sufixo;
  - apenas o prefixo e sufixo;
  - apenas a localização.

  Um exemplo de uma equipa com prefixo e localização é "*IFK Göteborg*", e uma com localização e sufixo é "*Örgryte IS*";
- O padrão dos resultados é também relativamente simples, consistindo na forma "Number1 - Number2", sendo *Number1* e *Number 2* valores inteiros, separados por um traço. Esta é a forma usada para reportar os resultados dos jogos;

- O padrão mais complicado é o da vitória/derrota, e tem a forma "*Team1 [...] Key [...] Team2 [...] Team3*"; *Team1*, *Team2* e *Team3* são padrões de equipas (referidos acima), e *Key* é uma palavra que diz algo sobre o resultado do jogo. Exemplificando para a frase "*IFK Göteborg lyckades till slut besegra ett svagt Örgryte efter spännande match.*" ("O IFK finalmente conseguiu derrotar um fraco Örgryte num jogo emocionante."), *Team1* é "*IFK Göteborg*", *Key* é "*besegra*" e *Team2* é "*Örgryte*". *Team 3* fica sem valor atribuído.

  Existem quatro variações deste padrão, de forma a contemplar a forma activa e passiva para cada uma das palavras de vitória e derrota. Esta distinção é importante para determinar a equipa vencedora, que, por exemplo, pode estar em *Team1* ou *Team2*, conforme a forma activa ou passiva.

  Pode acontecer que uma frase emparelhe com vários padrões. Para evitar ambiguidades na escolha do padrão, o sistema define uma ordem pela qual os padrões são aplicados, sendo o que tem maior prioridade "*Team1 Key Team2*", e o que tem menos é simplesmente "*Key*".

Após a conclusão deste módulo, é construída uma lista de equipas e resultados ("*scores*") mencionados, e uma lista dos padrões de resultados detectados (vitória, derrota ou empate).

No módulo seguinte, o da lógica local, é extraída informação que está disponível em cada frase. Deixa de existir a distinção entre a forma activa/passiva e das palavras de vitória/derrota. O resultado dos jogos passam para, por exemplo, "*IFK wins*", ou "*IFK defeats AIK*", com base em regras de transformação de padrões, que passam pela detecção de preposições que actuam sobre o nome das equipas. Por exemplo, se aparecer no texto "*mot IFK*" ("contra o IFK"), é uma indicação de que a frase é sobre outra equipa (o "sujeito" da frase). No final é criada uma lista de equipas, indicando as preprosições que as precedem ou sucedem, uma lista de factos da forma "*Team1 wins*", ou "*Team1 defeats Team2*", e a lista de resultados inalterada do módulo anterior.

No módulo da lógica global, é feita uma detecção do sujeito (a equipa de que o texto é sobre) para cada frase, seguindo uma série de regras. Este passo é realizado para completar factos que estejam incompletos, tal como "*Team1 loses*", baseando-se numa lista de prioridades acerca da credibilidade dos factos (um facto com maior prioridade corresponde a um facto mais certo). Neste módulo é também atribuído um resultado a todos os factos encontrados. Existe também uma prioridade associada aos resultados, o resultado associado a um facto que tiver maior prioridade é atribuído ao facto em questão. A saída deste módulo compreende uma lista de factos juntamente com os resultados para cada facto, e as prioridades de ambos.

O último módulo consiste na limpeza da lista de factos, que é processada e, eventualmente, na remoção de alguns factos. Desenrola-se através de dois passos:

- Unificação: os factos são comparados e, se possível, são unificados. Mais uma vez, são usadas prioridades para os factos usando para tal a fórmula 2.3. Ordena-se a lista de factos por ordem

decrescente, e vai-se removendo o facto de maior prioridade e compara-se este com os restantes. Cria-se assim uma nova lista com os factos unificados.

- Remoção de inconsistências: as entradas da lista que não são consistentes são apagadas da lista. Duas entradas são consideradas inconsistentes se as equipas forem as mesmas, mas o resultado for diferente. O facto que tiver menor prioridade é removido. Existe também um valor mínimo para esta prioridade, abaixo do qual as entradas da lista não são consideradas. No sistema, este valor é 0.

$$10 \cdot (prioridade\ do\ facto) + (prioridade\ do\ resultado) \tag{2.3}$$

Após estes quatro módulos são obtidos os resultados finais, que consistem numa lista de jogos, com referência às equipas, se houve vitória/empate, e o resultado final. Cada entrada da lista de jogos consiste em quatro campos, "[Team1, wins/draw, Team2, Score]". Por exemplo, pode ser construída a seguinte entrada: "[['IFK', 'Goteborg'], wins, ['AIK'], [2, '-', 1]]".

2.2.3 Avaliação

Os resultados obtidos por estes dois sistemas (presentes na tabela 2.2) são bastante semelhantes, pois têm basicamente a mesma estrutura, tendo o primeiro sistema sido construído retirando ideias do segundo sistema. Existem algumas variações nestes valores, como a cobertura do primeiro sistema que cai para 30% quando analisa textos muito grandes, e o segundo sistema sobe os seus valores de precisao e cobertura para 85% e 75%, respectivamente, quando o artigo que está a analisar apenas contém a descrição de um jogo.

De referir que estes valores são bastante semelhantes aos valores de referência para sistemas de extracção de informação, definidos por Hobbs (Hobbs, 2002).

Sistema	Precisão	Cobertura
S. Futebol	60%	40%
S. Futebol e Hóquei	66%	56%

Tabela 2.2: Avaliação dos Sistemas de Notícias Desportivas

2.3 Comparação Entre os Sistemas

A tabela 2.3 mostra a comparação entre os sistemas apresentados, no que diz respeito às abordagens utilizadas. De uma forma geral, os sistemas apresentados pertencentes ao domínio da biologia e bio-

Sistema	Análise Sintáctica	Análise Contextual	Aprendizagem	Uso de Dicionários
BioAnnotator	Sim	Sim	Sim	Sim
PROPER	Não	Sim	Não	Não
EDGAR	Sim	Sim	Não	Sim
FASTUS	Sim	Sim	Não	Não
Sistema de Proux et al.	Sim	Sim	Não	Sim
Sistema de Hagège e Brun	Sim	Sim	Não	Não
Sistema de Collier et al.	Não	Não	Sim	Não
Sistema de Futebol	Não	Sim	Não	Não
Sistem de Futebol e Hóquei	Não	Sim	Não	Não

Tabela 2.3: Comparação entre os Sistemas

medicina são baseados em análise sintáctica e análise contextual, havendo dois que usam em parte (o BioAnnotator) ou totalmente (Sistema de Collier et al.) métodos de aprendizagem. Já relativamente aos sistema de extracção no domínio desportivo, apenas usam a abordagem de análise contextual.

3 Arquitectura do Sistema

3.1 Arquitectura

O sistema de carregamento de receitas é constituído por três módulos, sendo o primeiro um módulo de pré-processamento onde são efectuadas algumas alterações às receitas; o segundo módulo é responsável pelo processamento das receitas, usando para tal a cadeia de processamento XIP (Xerox Research Centre Europe, 2003; Aït-Mokhtar et al., 2002), realizando também a extracção da informação necessária; por fim, o último módulo transforma os dados obtidos de forma a serem carregados numa base de dados.

3.2 Pré-processamento

O módulo de pré-processamento consiste na transformação das receitas de forma a facilitar a sua análise no módulo seguinte. As alterações efectuadas aos ficheiros consistem em adicionar um ponto final (".") no final de cada linha do ficheiro, uniformizar a medida relativa às colheres e algumas unidades de medida, e dividir a receita em partes.

A adição de um ponto final no final de cada linha dos ficheiros que contêm as receitas é devida ao sistema XIP, que efectua o processamento de um texto frase a frase. Esta alteração tem mais impacto na parte dos ingredientes, pois normalmente as receitas não têm um ponto final a seguir a cada ingrediente. Desta forma, cada frase vai corresponder a uma entrada na listagem dos ingredientes numa receita.

Outra alteração consiste em uniformizar a medida relativa às colheres (que podem ser de sopa, de sobremesa, de café e de chá), pois existem receitas que contêm, por exemplo, "2 colheres, das de sopa, de açúcar", ou "2 colheres (de sopa) de açúcar". Nestas situações a uniformização transforma ambas as expressões em "2 colheres de sopa de açúcar". Tal como a alteração anterior, esta também vai facilitar o processamento no módulo seguinte, neste caso, auxiliando na identificação da quantidade de um alimento, e é também uma forma para ficar em acordo com as medidas presentes na ontologia.

É realizada uma alteração a nível de algumas unidades de medida que são incorrectamente usadas, tal como "gr" e "grs", usadas para denotar a unidade "grama", bem como alguns alimentos que não estão correctamente escritos, como "açucar", sendo necessário colocar o acento, de forma a que seja reconhecido como um alimento. Nestas situações, as palavras incorrectas são transformadas para a

palavra correcta, ou seja, “g” e “açúcar”.

Cada receita é também dividida em três partes, o título, os ingredientes e a preparação, originando cada uma destas três ficheiros diferentes. Esta separação é realizada através da detecção de uma linha em branco entre cada uma das partes, não podendo haver outras linhas em branco no ficheiro que contém a receita. Sendo assim, a receita terá que ter obrigatoriamente estas linhas em branco de modo a que o sistema funcione correctamente. Esta alteração é útil para saber, nos módulos seguintes, qual das partes está a ser analisada, tendo especial impacto no módulo de transformação da saída (ver secção 3.4), onde é mesmo necessário saber qual das partes está a ser analisada.

As alterações realizadas são feitas usando a consola, utilizando expressões regulares juntamente com substituições com o comando “sed”.

3.3 Processamento das Receitas

O segundo módulo, responsável pelo processamento das receitas, tem como finalidade extrair a informação necessária das receitas, tanto a nível dos ingredientes como das acções presentes na preparação. Para tal usa-se a cadeia de processamento XIP, efectuando uma análise sintáctica frase a frase, e extraindo dependências que contêm tanto os ingredientes como as acções identificadas. Estas dependências são obtidas através de regras, analisando o contexto das frases e testando a existência de determinadas características das frases (padrões).

Este módulo está mais detalhado no capítulo 5.

Cadeia de Processamento XIP

A cadeia de processamento XIP permite realizar a análise sintáctica de frases, obtendo uma divisão das frases em segmentos (sintagma nominal, preposicional, etc.) e extraindo dependências entre elementos da frase. A figura 3.1 apresenta a arquitectura da cadeia de processamento XIP.

A análise inicia-se com uma segmentação da frase, onde se separa a frase nos seus constituintes mais simples (as palavras) e identificam-se alguns símbolos especiais, como endereços de e-mail, números, abreviaturas e números romanos. De seguida realiza-se a etiquetação morfossintáctica, atribuindo a cada palavra as suas categorias gramaticais, subcategorias, número e género, entre outras. Para realizar esta função é usada a ferramenta Palavroso (Medeiros, José C., 1995). Após a etiquetação é realizada uma divisão em frases, considerando separadores de frases todos os segmentos constituídos por “.”, “!” e “?”. Depois usa-se a ferramente RuDriCo (Pardal & Mamede, 2004) para realizar algumas correcções e modificações, tal como correcções ao Palavroso e descontracção de palavras (como “do”, que passa

a “de” + “o”). O passo seguinte consiste na aplicação de um desambiguador morfossintáctico, o Marv (Ribeiro et al., 2002), que escolhe a categoria gramatical mais adequada para cada palavra entre as categorias atribuídas pelo Palavroso. Depois o resultado é enviado para o XIP (Xerox Research Centre Europe, 2003; Aït-Mokhtar et al., 2002), que tem como funções dividir a frase em segmentos, calcular dependências entre esses segmentos, introduzir informação léxica e aplicar gramáticas locais.

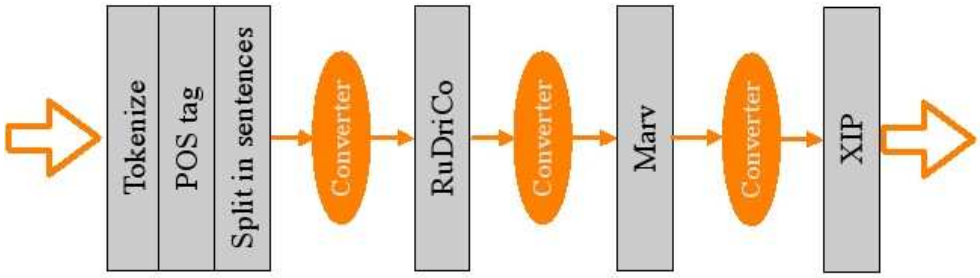

Figura 3.1: Arquitectura da cadeia de processamento XIP.

O XIP permite a criação de léxicos ao associar determinados traços às palavras. Os léxicos são construídos por listas onde se associam os traços às palavras. No exemplo da figura 3.2, adicionam-se os traços “meas” (medida) e “time” (tempo) às palavras com lema “hora”, desde que tenha a classe gramatical “noun” (nome).

```
hora:  noun += [meas=+, time=+].
```

Figura 3.2: Exemplo de entrada no léxico.

Este método para construir um léxico só permite lidar com palavras, pelo que tem que se recorrer a outro tipo de configuração para suportar palavras compostas. Normalmente, esta configuração vai agrupar num único segmento palavras que, até então, estavam em segmentos separados. Na figura 3.3 está um exemplo desta situação. Neste caso, pretende-se que a expressão “quilómetros por hora” seja considerada como uma medida. Como tal, ao encontrar as palavras com lema “quilómetro”, “por” e “hora” nesta ordem, forma-se um segmento com a categoria “noun”, atribuindo também o traço “meas”. O sinal “?” indica que a categoria gramatical da palavra em questão é indiferente, interessando apenas o conteúdo dentro dos parêntesis rectos (ou seja, em “?[lemma:quilómetro]” pretende-se emparelhar com uma palavra que tenha o lema “quilómetro”, independentemente da sua categoria gramatical).

```
1> noun[meas=+] = ?[lemma:quilómetro], ?[lemma:por], ?[lemma:hora].
```

Figura 3.3: Exemplo de entrada na gramática local.

As regras são também um mecanismo importante na medida em que estas permitem estabelecer relações entre nós lexicais, quer sejam simples palavras ou segmentos. Uma regra é constituída por três

partes (ver figura 3.4). A primeira, o contexto, consiste num padrão que tem de ser emparelhado, e que descreve propriedades estruturais que o texto de entrada deve ter. A segunda parte, a condição, consiste numa expressão booleana construída através de dependências já descobertas e com operadores booleanos ("&", "|" e "^"). A última parte, a extracção, é uma lista de dependências na forma $nome[lista\_t](a_1, a_2, ..., a_3)$, em que "nome" é o nome da dependência, "lista_t" é uma lista de traços associados à dependência, e "$a_1, a_2, ..., a_n$" são os argumentos da dependência. Ou seja, para obter as dependências é necessário que determinada expressão emparelhe com o padrão da parte Contexto, e que as dependências da parte Condição sejam todas satisfeitas. A figura 3.5 contém um exemplo de uma regra usada para determinar a dependência CDIR (complemento directo), para a frase "Os amigos comeram duas bananas". Esta regra contém as três partes constituintes de uma regra.

```
| Contexto |
Condição
Extracção
```

Figura 3.4: Estrutura de uma regra.

```
| #2[verb:+];sc{?*,#2[verb:+]}, (ADVP), NP#3 |
 if ( HEAD(#1,#2) & HEAD(#4,#3) & VDOMAIN(?,#1) & ~CDIR(#1,?) &
      ~PREDSUBJ(?,#4) )
   CDIR[post=+](#1,#4)
```

Figura 3.5: Exemplo de uma regra para determinar uma relação de complemento directo.

A utilização de regras é a base da extracção deste trabalho e nas secções 5.2 e 5.3 serão apresentadas as que estão no âmbito deste trabalho, e que permitem extrair dependências como INGR (que indica um ingrediente da receita) ou ADICIONAR (que indica a acção "Adicionar", colocar um alimento dentro de um utensílio).

3.4 Pós-processamento

O módulo de pós-processamento é responsável pela transformação da saída do XIP em código SQL de forma a ser carregado numa base de dados (descrita na secção 4.3), e assim ter a informação da receita colocada na base de dados.

Supondo que é extraída pelo XIP a dependência INGR(azeite), este módulo produz o código SQL "INSERT INTO Ingrediente (IDReceita, IDMedida, IDAlComEstado, nome) VALUES("1", "13", "24", "azeite")" referente apenas à tabela "Ingrediente". O campo "IDReceita" indica qual a receita a que este ingrediente pertence, "IDMedida" indica o ID da medida do ingrediente, neste caso, como não tem quantidades, o módulo associa como quantidade "0" e como unidade de medida "unidade". O campo

"IDAlComEstado" refere, ainda que indirectamente, o alimento a que o ingrediente se refere, e o "nome" indica o nome do ingrediente, com as suas quantidades associadas, ou seja, se o ingrediente for "200 g de açúcar", é "200 g de açúcar" que será colocado neste campo.

Para realizar esta transformação é usada a linguagem Java, efectuando manipulação de XML e interagindo com a base de dados. No capítulo 6 é apresentadoo módulo de transformação de saída.

3.5 Corpus

Procedeu-se à recolha de um corpus de receitas culinárias de diversos tipos (contemplando receitas de carne, peixe, sobremesa e sopa, 12 de cada, escolhidas aleatoriamente) retiradas de dois livros (Maria de Lourdes Modesto, 1981, 1982) e de várias páginas da Internet (*Gastronomia*, 2006; *Roteiro Gastronómico de Portugal*, 2006; *Kitchenet*, 2006; *WikiLivros*, 2006). De cada fonte foram retiradas duas receitas de cada tipo, consistindo o corpus em 48 receitas no total, das quais 40 constituem o corpus de treino, de onde se derivaram as regras para a extracção de ingredientes e acções, e as restantes 8 servem para realizar uma avaliação final.

As receitas são normalmente compostas por três partes, o nome da receita, os ingredientes e a preparação. Em alguns casos, a indicação do início dos ingredientes ou da preparação (como, por exemplo, "Ingredientes:") está omisso. A parte correspondente aos ingredientes consiste numa listagem de alimentos juntamente com a sua quantidade, tal como "25 g de açúcar". A parte da preparação contém um conjunto de frases que descrevem uma ou mais tarefas que constituem a execução da receita.

4 Ontologia e Base de Dados

4.1 A Ontologia

A ontologia (Ribeiro et al., 2006) disponível para o domínio da culinária contém um descrição de conceitos e relações relativos a este domínio. Como tal, os conceitos constituem uma hierarquia de classes, sendo as principais Alimento, Utensílio e Acção. Em cada uma das classes, os conceitos vão sendo mais específicos até chegar, por exemplo, ao animal (no caso dos alimentos), ou ao utensílio.

4.1.1 Alimentos

O conceito *Alimento* diz respeito às substâncias que são digeríveis e podem ser consumidos por humanos de forma a que possam viver, crescer e ter energia. Inclui as subclasse carne, peixe, derivados de leite, cereais, frutas, entre outros. Para o alimento "lombo de vaca", é possível encontrar as super-classes "Vaca", "Carne de Açougue", "Carne" e "Alimento", nesta ordem. Como os alimentos normalmente não são usados como um todo, existe também o conceito "Parte de Alimento Sólido", de forma a representar as partes que o alimento pode ter. Tendo em conta o exemplo anterior, "lombo de vaca" é um subclasse de "Vaca" (Alimento) e também de "Lombo" (Parte de Alimento). Associado também aos alimentos está o conceito de "Estado de Alimento", que indica o estado que um alimento tem. O estado pode ser alimentar (e.g. verde, maduro), físico (e.g. sólido, líquido), culinário (e.g. cozido, assado), e de segmentação (e.g. aos cubos, às rodelas).

4.1.2 Utensílios

O conceito *Utensílio* contém uma hierarquia de utensílios que podem ser usados na cozinha. Estes utensílios estão divididos nas subclasses "Utensílio manual" (e.g. faca, tacho), "Electrodoméstico" (e.g. frigorífico, fogão) e "Acessório" (e.g. tampa).

4.1.3 Acções

O conceito Acção consiste nas operações que podem ser executadas com utensílios e alimentos, e que indicam os passos necessários para realizar as receitas. Inerente a este conceito estão os conceitos "Acção

Primitiva", "Acção Ordenada" e "Acção Composta". As acções primitivas são as acções mais simples da culinária, tal como partir um ovo, untar uma forma ou adicionar um alimento a um recipiente. Existem três tipos de acções primitivas, Acções primitivas de espera (esperar 10 minutos, esperar que alimento esteja cozido), de utensílios (ligar o fogão, tapar o tacho), de alimentos (partir um ovo), e de alimentos e utensílios (cortar uma maçã com faca). As acções compostas são tarefas mais complexas e que são definidas a partir de um conjunto de acções primitivas, tal como Cozer e Assar. As acções ordenadas associam um número a uma acção primitiva ou composta, de forma a definir sequências de acções.

A figura 4.1 contém as relações existentes entre os diversos conceitos da ontologia.

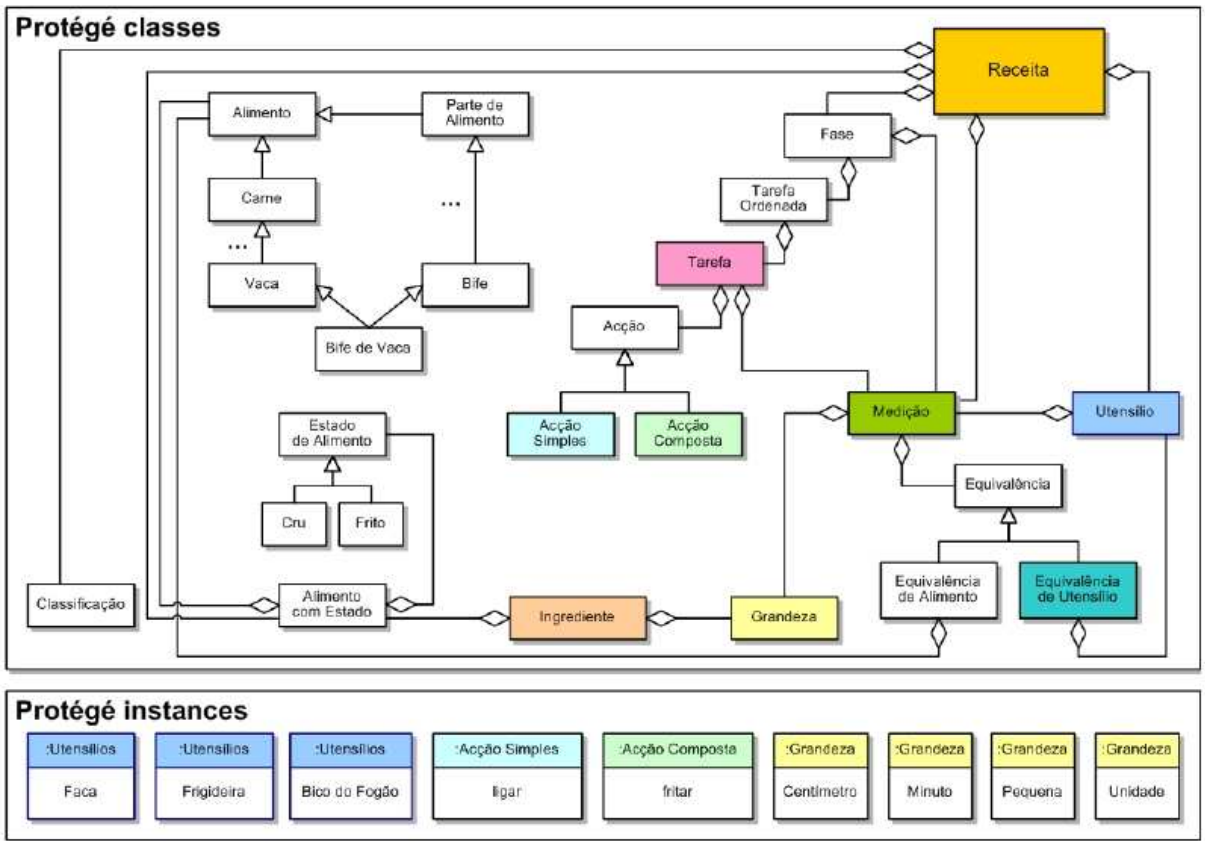

Figura 4.1: Conceitos principais da ontologia.

4.2 Alterações à Ontologia

Tendo-se verificado que a ontologia não estava completa e que faltavam conceitos que eram relevantes para o domínio, foram adicionadas mais algumas acções e mais uma dimensão para o Estado. As adições relativas às acções foram:

- Acções Primitivas:
  - Acções de Alimentos:
    - Manter: Conservar um alimento:
      - Manter Quente: Conservar um alimento quente;
      - Manter Frio: Conservar um alimento frio;
    - Espremer: Extrair líquido de alimentos, nomeadamente, citrinos;
    - Verificar: Certificar que alimento está a gosto (Exemplo: "Rectifique o sal");
    - Tirar: Remover algo a um alimento:
      - Tirar Ossos: Remover os ossos do alimento;
      - Tirar Espinhas: Remover as espinhas do alimento;
      - Tirar Sementes: Remover as sementes ou caroços do alimento;
    - Coser: Coser um alimento com agulha e linha;
    - Juntar: Adicionar um alimento a outro:
      - Juntar até Meio: Adicionar um alimento até meio do outro alimento;
      - Juntar em Cima: Adicionar um alimento em cima de outro;
      - Juntar ao Lado: Colocar um alimento ao lado de outro;
      - Juntar à Volta: Rodear um alimento com outro alimento;
  - Acções de Alimentos e Utensílios:
    - Moldar: Dar forma a um alimento recorrendo a um utensílio;
  - Acções de Utensílios:
    - Retirar Cima: Retirar um utensílio de cima do outro;
    - Retirar Dentro: Retirar um utensílio de dentro do outro;
    - Agitar: Abanar um utensílio;
- Acções Compostas:
  - Arranjar: Amanhar o peixe, arranjar pimentos;
  - Temperar: Adicionar vários alimentos para realizar um tempero;
  - Fazer: Produzir determinado alimento:
    - Fazer Caldo: Produzir um caldo;
    - Fazer Gelatina: Produzir uma gelatina;

Na análise às acções primitivas de utensílios verificou-se que faltavam algumas acções inversas, pois existem as acções "Colocar Dentro" e "Colocar Cima", que indicam que um utensílio deve ser

colocado dentro/em cima de outro utensílio. No entanto não era possível representar as acções inversas, pelo que se criaram as acções "Retirar Cima" e "Retirar Dentro", de forma a poder tratar expressões como "Tire a panela do forno.".

Relativamente à acção "Juntar", esta foi considerada pois, embora exista a acção "Adicionar", que diz respeito à adição de um alimento a um utensílio, existem situações em que não há referências a utensílios, ou situações que dizem respeito à apresentação final da receita, pelo que se criou esta acção para poder representar esses casos.

As restantes acções foram criadas devido à existência de algumas frases no corpus de treino que não tinham correspondência com nenhuma das acções da ontologia, sendo necessária a sua existência para representar de uma forma mais concreta as acções das receitas, como as acções compostas adicionadas. No entanto, faltam muitas acções compostas pois estas acções resultam da combinação de acções primitivas, logo, é possível criar múltiplas acções compostas.

A adição efectuada no Estado é relativa à dimensão "Tamanho", que pode ser "Grande", "Pequeno" ou "Médio". A ontologia já possui o conceito de "Medida", mas que apenas considera medidas exactas (como 20 gramas). Para este tipo de medidas imprecisas e que variam de alimento para alimento, e da percepção de cada pessoa, não havia uma representação possível. Como tal, adicionou-se esta dimensão ao conceito "Estado" e assim podem-se tratar expressões como "2 cebolas grandes".

4.3 Relação Entre a Ontologia e a Base de Dados

Foi necessário construir uma base de dados de raiz, tendo como base a ontologia já descrita, devido à inexistência de uma base de dados preparada para este domínio e para este efeito.

O diagrama que representa a base de dados assim criada está presente na figura 4.2. Quase todas as tabelas têm uma correspondência directa com as classes da ontologia, exceptuado as relações existentes para as classes "Alimento com Estado" e "Tarefa".

A classe "Alimento com Estado", que relaciona um alimento e vários estados, necessita de um mapeamento diferente devido à multiplicidade do campo "Estado" (por exemplo, carne assada fatiada). Desta forma, a tabela "EstadosAlComEstado" relaciona uma instância da tabela "AlimComEstado" (que tem apenas uma referência para o alimento) e um estado possível (na tabela "Estado"), permitindo assim ter o mesmo alimento associado a vários estados.

Da mesma forma, a classe Tarefa relaciona uma acção com vários alimentos, relação presente na tabela "AlimentosTarefa", associando a uma tarefa todos os alimentos (com estado) que dela fazem parte.

Relativamente à classe "Medição" da ontologia (representada na tabela "Medida"), esta pode consistir numa medição fixa (por exemplo "20 g") ou numa medição delimitada (por exemplo "7 a 8 ovos"). Como só pode ser uma destas duas, criaram-se duas tabelas ("MedicaoFixa" e "MedicaoDelimitada") com uma chave estrangeira para a tabela "Medida", de forma a que, quando se procura por uma medida, ou está na tabela "MedicaoFixa" ou na "MedicaoDelimitada".

As tabelas "Categoria", "CatAlimento" e "PaiAlimento" têm como objectivo representar conhecimento presente na ontologia referente à hierarquia de categorias dos alimento. A tabela "Categoria" contém todas as categorias presentes nesta hierarquia, a tabela "CatAlimento" indica a primeira categoria de um determinado alimento (por exemplo, o alimento "salmão" tem como primeira categoria "Peixe de água doce"), e a tabela PaiCategoria estabelece as restantes relações presentes acima na hierarquia ("Peixe de água doce" tem como "pai" a categoria "Peixe").

Quanto à forma de representação das acções envolvidas na preparação de uma receita, na ontologia cada receita tem três fases (preparação, confecção e apresentação), uma fase é um conjunto de tarefas (ordenadas ou opcionais), e uma tarefa associa uma acção a alimentos e utensílios. Como neste trabalho só se vai considerar uma fase (a da confecção), não há necessidade de ter uma tabela para esta classe, havendo a tabela "Tarefa Ordenada" que relaciona uma receita com as suas diversas tarefas, e um número que indica a ordem da tarefa. A tabela "Tarefa" inclui a acção correspondente, os utensílios necessários (se aplicável) e o tempo ou temperatura relativa ao alimento ou ao utensílio, dependente da acção (se aplicável). Desta forma é possível representar na base de dados os passos por que passa a confecção de uma receita.

As tabelas "Alimento", "Estado", "Utensílio", "Grandeza" e "Acção" têm dados carregados previamente, pelo que os valores dessas tabelas estão restritos (a um léxico definido) e não serão colocados novos dados durante o carregamento de uma receita.

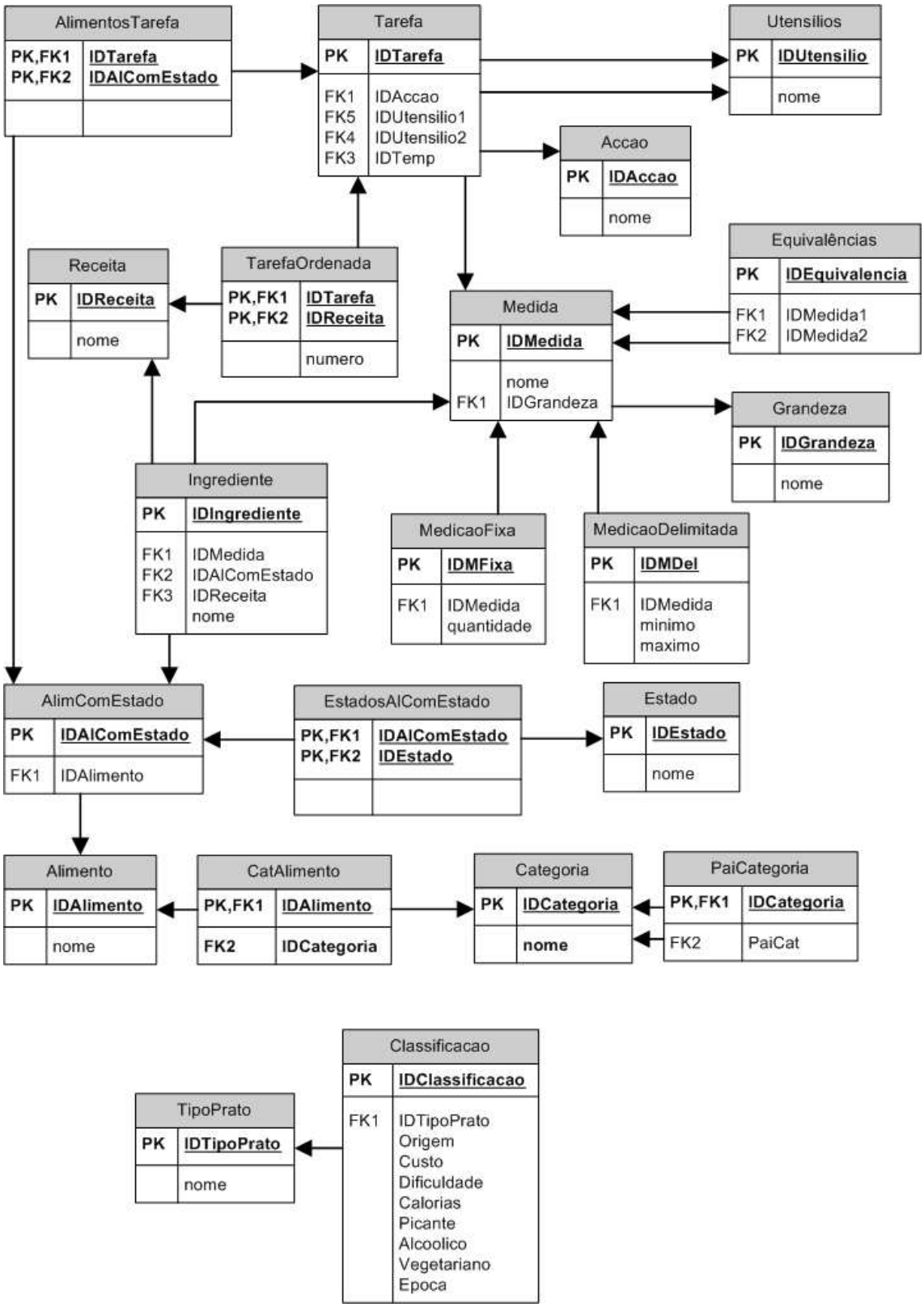

Figura 4.2: Diagrama da base de dados desenvolvida.

5 Processamento de Receitas

Neste capítulo, descreve-se o módulo que realiza o processamento das receitas, usando para tal o sistema XIP de forma a realizar a análise sintáctica das frases e também obter as dependências que incluem a informação extraída das receitas.

A secção 5.1 contém o léxico criado que engloba as palavras pertencentes ao domínio da culinária, divididos em ingredientes, estados e utensílios. Nas secções 5.2 e 5.3 está presente a representação usada para os ingredientes e acções, respectivamente, bem como as regras que permitem extrair a informação necessária na forma de dependências.

5.1 Léxico de Ingredientes, Utensílios e Estados

Os traços que foram introduzidos para caracterizar as palavras do domínio da culinária são:

- alimento - Indica um alimento, tal como, "vaca", "truta" ou "esparguete";
- partealimento - Indica a parte de um alimento, tal como "gema", "acém" ou "polpa";
- estado - Indica o estado de um alimento, tal como "cozido", "maduro" ou "fresco";
- utensilio - Indica um utensílio de cozinha, tal como "tacho", "faca", ou "chávena";
- material - Indica o material de que é feito o utensílio, tal como "vidro", "pirex" ou "porcelana".

Dados estes traços, é necessário agora aplicá-los, associando-os às palavras respectivas. Como a ontologia contém uma listagem de alimentos, parte de alimentos, estados e utensílios, pode-se proceder a uma passagem de cada um dos elementos referidos para o léxico. No entanto, devido à ontologia não estar completa, foi necessário ir a outras fontes, tal como a uma página da Internet (*Receitas e Menus*, 2006), para ter um léxico mais completo e rico. No total existem 561 entradas para alimentos, 68 para partes de alimento, 75 para estados, 160 para utensílios e 17 para materiais.

Na figura 5.1 estão representados três alimentos com a sua associação ao traço respectivo. Quando é encontrada uma palavra com lema "lebre", e esta seja da categoria "noun", então é-lhe adicionado o traço "alimento", mantendo os outros que eventualmente possa ter. De notar que, embora a ontologia

tenha bem definidas todas as categorias de alimentos (carne, peixe, fruta, etc.), nesta altura é suficiente saber apenas que é um alimento. Futuramente poder-se-á considerar completar este léxico com a informação referente às categorias dos alimentos.

O traço "partealimento" diz respeito às partes de um alimento. Esta parte do léxico tem mais componentes de carne do que das outras, pois é mais comum o facto da carne dos animais ser dividida em partes do que noutras categorias de alimentos. Algumas partes de alimento que fazem parte do léxico estão representadas na figura 5.2, onde se associa o traço "partealimento" a essas palavras.

O estado de um alimento refere-se a certas características que o alimento apresenta. Alguns desses estados estão representados na figura 5.3. A atribuição do traço "estado" é semelhante aos casos anteriores, no entanto, desta vez a categoria gramatical não interessa, visto que, tanto adjectivos (como "maduro") como particípios passados (como "grelhado") podem ser considerados como estado de alimento. No caso de particípios passados, o seu lema é o verbo ("grelhado" é particípio passado com lema "grelhar"), pelo que apenas se vai associar o traço "estado" ao particípio passado do verbo em questão.

"Utensílio" é um conceito de igual modo importante neste domínio, pelo que também faz parte do léxico. A figura 5.4 contém alguns desses utensílios usados na cozinha, bem como a associação com o traço "utensilio".

```
camarão:    noun +=[alimento=+].
lebre:      noun +=[alimento=+].
feijão:     noun +=[alimento=+].
```

Figura 5.1: Exemplos de entradas do léxico de culinária (alimentos).

```
alcatra:    noun +=[partealimento=+].
costeleta:  noun +=[partealimento=+].
filete:     noun +=[partealimento=+].
gema:       noun +=[partealimento=+].
polpa:      noun +=[partealimento=+].
```

Figura 5.2: Exemplos de entradas do léxico de culinária (partes de alimento).

```
maduro:         adj +=[estado=+].
grelhar:   pastpart +=[estado=+].
ralar:     pastpart +=[estado=+].
```

Figura 5.3: Exemplos de entradas do léxico de culinária (estado).

Como foi explicado na secção 3.3, existe uma outra forma de construir o léxico para conceitos com mais de uma palavra, como "feijão verde". Esta forma é designada de gramática local, e altera os segmentos de uma frase, agrupando várias palavras no mesmo segmento. Como tal, todos os alimentos e

```
tacho:      noun +=[utensilio=+].
garfo:      noun +=[utensilio=+].
balança:    noun +=[utensilio=+].
```

Figura 5.4: Exemplos de entradas do léxico de culinária (utensílios).

utensílios que têm mais que duas palavras foram colocadas nesta gramática local. Foram também colocadas algumas variedades de alimentos, como as variedades de queijo ("Queijo Emmental", "Queijo Parmesão", etc.) e de várias frutas ("Maçã Golden", "Pera Rocha", etc.), como representado na figura 5.5. Por exemplo, ao encontrar duas palavras seguidas cujo lema é "queijo" e "Emmental", estas duas palavras são agrupadas no mesmo segmento, que fica com o traço "alimento". O número presente no início diz respeito à ordem da aplicação destas regras, sendo, neste caso o mesmo, visto que é indiferente aplicar uma regra antes ou depois de outra.

Foram também definidas regras relativas ao nome dos utensílios. De facto, o nome de bastantes utensílios tem a forma "≪Utensílio≫ de cozinha", o que permite derivar uma regra para aplicar este padrão. Da mesma forma, foi também determinado o padrão "≪Utensílio≫ de ≪Material≫", daí a utilização do traço "material", para palavras como vidro, barro, ferro, alumínio, entre outras. Na figura 5.6 está presente a implementação destes padrões.

De referir também a adição de algumas medidas que não estão na ontologia, e que por vezes surgem em algumas receitas, como "lata" ("1 lata de leite condensado"), "pacote" ("1 pacote de natas"), ou "cálice" ("1 cálice de vinho do Porto"). Estas palavras ficam anotadas com o traço "meas" (medida), traço este que não é específico deste domínio.

```
1> noun[alimento=+] = ?[lemma:"queijo"], ?[surface:"Emmental"];
1> noun[alimento=+] = ?[lemma:"feijão"], ?[lemma:"verde"].
1> noun[alimento=+] = ?[lemma:"maçã"], ?[surface:"Golden"];
```

Figura 5.5: Exemplos de entradas do léxico com mais de uma palavra.

```
20> noun[utensilio=+] = ?[utensilio:+], ?[lemma:"de"], ?[lemma:"cozinha"].
20> noun[utensilio=+] = ?[utensilio:+], ?[lemma:"de"], ?[material:+].
```

Figura 5.6: Padrões determinados para os utensílios.

5.2 Extracção dos Ingredientes

Após a passagem da receita pelo XIP, devem ser obtidas as dependências que identificam os ingredientes que essa receita tem. A secção 5.2.1 apresenta a representação usada para os ingredientes e a secção 5.2.2 contém as regras usadas para realizar a extracção dos ingredientes.

5.2.1 Representação dos Ingredientes

Para ser possível extrair a informação relativa aos ingredientes, é necessário criar algumas dependências de forma a representar os ingredientes de uma receita, que são:

- INGR - indica o ingrediente, bem como a sua quantidade e a unidade de medida;
  - parâmetros:
    - Com aridade unária, o único argumento é o alimento (por exemplo INGR(sal));
    - Com aridade binária, o primeiro argumento é um número, que indica a quantidade em termos de unidades do alimento, e o segundo argumento é o alimento (por exemplo, INGR(5, limões));
    - Com aridade ternária, o primeiro argumento é um número que denota a quantidade, o segundo argumento é a unidade de medida utilizada, e o terceiro é o alimento (por exemplo, INGR(7, dl, água)).
- PARTEDE - indica que um determinado nome é parte de um alimento;
  - parâmetros:
    - Apenas tem aridade binária, sendo o primeiro argumento a parte do alimento, e o segundo argumento é o alimento (por exemplo, PARTEDE(gema, ovo));
- ESTADO - indica o estado em que um alimento está.
  - parâmetros:
    - Apenas tem aridade binária, o primeiro argumento é o estado do alimento e o segundo é o alimento (por exemplo, ESTADO(grande, cebola));

Exemplificando para a frase "200 g de bife do lombo de vaca.", é possível obter as dependências INGR(200, g, vaca), PARTEDE(bife, vaca) e PARTEDE(lombo, vaca). Já para a frase "Queijo ralado.", obtém-se INGR(queijo) e ESTADO(ralado, queijo).

5.2.2 Regras para a Extracção de Ingredientes

São várias as formas que um ingrediente tem no corpus de receitas. O ingrediente pode ter ou não uma quantidade associada, pode ser ou não constituído por partes de alimento, pode ter a indicação de duas quantidades associadas, entre outras. Deste modo, são necessárias várias regras para conseguir detectar todas estas situações. Para os ingredientes foram escritas quinze regras e para as dependências PARTEDE e ESTADO foram duas as regras escritas, para cada dependência.

Em relação à detecção da dependência INGR, foram identificados vários padrões:

- «Número» «quantidade» de («parte de alimento»)+ de «alimento»:

  São exemplos da aplicação deste padrão as frases "200 g de bife do lombo de vaca" e "200 g de lombo de vaca". Ou seja, estas frases têm um sintagma nominal que corresponde à quantidade e dois ou três sintagmas preposicionais que indicam a parte do alimento ou o alimento, como se pode ver na figura 5.7 (saída obtida pelo XIP com a opção para visualizar a árvore sintáctica).

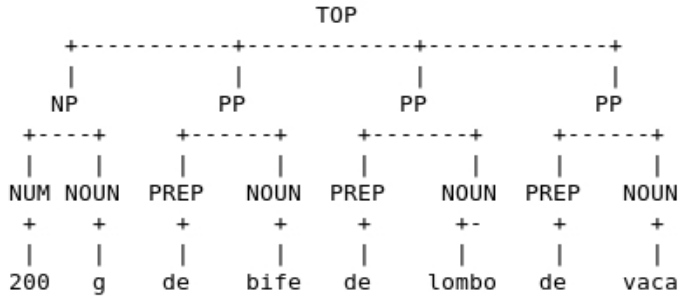

Figura 5.7: Saída do XIP para a frase "200 g de bife de lombo de vaca".

  A regra que determina a dependência a ser extraída está presente na figura 5.8. O contexto é composto por um sintagma nominal (NP), sendo depois necessário verificar se a cabeça desse sintagma (determinado através da dependência "HEAD") tem um traço "meas" (ou seja, é uma quantidade) e se tem uma relação de quantidade com outra palavra (neste caso a cabeça é "g", e existe uma relação QUANTD entre "g" e "200"), e se também há uma relação MOD entre a quantidade e uma dada palavra que é um alimento (neste caso a palavra "vaca"). Se tudo se verificar, é criada a dependência INGR, que, neste caso, fica com os argumentos "200", "g" e "vaca", respectivamente.

- «Número» «quantidade» de «alimento»:

  Este padrão é usado para frases como "100 g de açúcar", não havendo partes de alimento e a quantidade especifica directamente o alimento.

```
| #4NP |
 if( HEAD(#1,#4) & QUANTD(#1[meas:+], #2) & MOD(#1, #3[alimento:+]) )
  INGR(#2, #1, #3)
```

Figura 5.8: Regra para determinar um ingrediente que possui partes.

Na figura 5.9 está uma regra que permite aplicar este padrão. Nesta regra, o contexto é composto por um sintagma nominal seguido de um sintagma preposicional. De seguida, verifica-se se a cabeça dos sintagmas têm, respectivamente, o traço "meas" e o traço "alimento", e se existe uma relação de quantidade entre a cabeça do sintagma nominal ("g", neste caso), e outra palavra, que é "100" para esta frase. No final é criada a dependência INGR(100, g, açúcar).

```
| #1NP, #2PP |
 if( HEAD(#3[meas:+],#1) & HEAD(#4[alimento:+],#2) & QUANTD(#3, #5) )
  INGR(#5,#3,#4)
```

Figura 5.9: Regra para determinar um ingrediente sem partes.

- «Número» «parte de alimento» (de «parte de alimento»)? de «alimento»:

  Expressões como "4 bifes do lombo de vaca" ou "2 dentes de alho" são emparelhadas por este padrão. A diferença deste para o padrão acima apresentado reside na forma da quantidade. Neste caso, a quantidade não tem unidades de medida, como grama, sendo considerada apenas as unidades da parte do alimento como quantidade.

  A regra da figura 5.10 tem em conta este padrão para extrair a dependência INGR. Tal como a regra construída para o primeio padrão, o contexto considera também apenas um sintagma nominal, que, para a frase exemplificada, é "4 bifes". A condição verifica se a cabeça ("bifes") deste sintagma nominal é uma parte de alimento, se há uma relação de quantidade entre este último e outra palavra (neste caso é "4"), e se há uma relação MOD entre a cabeça já referida e outra palavra que seja um alimento(neste caso, "vaca"). Depois, é criada a dependência INGR(4, vaca).

```
| NP#1 |
if( HEAD(#2[partealimento:+], #1) & QUANTD(#2, #4)
    & MOD(#2, #3[alimento:+]) )
  INGR(#4, #3)
```

Figura 5.10: Regra para determinar um ingrediente que possui partes.

- «Número» «alimento»:

  Este padrão aplica-se a frases como "2 limões" ou "3 tomates", sendo semelhante ao anterior,

com a diferença de este não considerar partes de alimento, ou seja, refere-se a unidades de um determinado alimento.

A regra que usa este padrão para extrair a dependência INGR está presente na figura 5.11. O contexto é constituído apenas por um sintagma nominal, verificando-se se a cabeça deste sintagma é um alimento, e se há uma relação de quantidade entre as duas palavras que compõem o sintagma. Em caso afirmativo, é criada a dependência INGR(2, limões).

```
| NP#1 |
if( HEAD(#2[alimento:+], #1) & QUANTD(#2[alimento:+], #3) )
  INGR(#3, #2)
```

Figura 5.11: Regra para determinar um ingrediente com unidades como quantidade.

- «Alimento»:

  Este é o padrão mais simples, visto que apenas considera o nome do alimento. Expressões como "sal" ou "sal e pimenta" são consideradas neste padrão.

  Para este padrão é usada a regra da figura 5.12. Nesta situação apenas existe um sintagma nominal com uma palavra, que vai corresponder à cabeça do sintagma. Esta palavra tem que ser um alimento para a dependência INGR(sal) poder ser criada. É verificado ainda se não existe nenhuma relação INGR já criada anteriormente, pois, ao considerar apenas como contexto um sintagma nominal, a regra fica demasiado abrangente, e, consequentemente, consideraria também frases como "2 limões", em que também apenas existe um sintagma nominal e sua cabeça é também um alimento. Assim sendo, é colocada na condição a expressão ~INGR(?,#2) (o "~" indica a negação) exactamente para descartar a utilização desta regra na situação em que a expressão contém uma quantidade unitária, situação que já é tratada com outra regra.

```
| NP#1|
if( HEAD(#2[alimento:+], #1) & ~INGR(?,#2) )
  INGR(#2)
```

Figura 5.12: Regra para determinar um ingrediente numa expressão que contém apenas o ingrediente.

- «Parte de alimento» de «alimento»:

  Este padrão aplica-se a expressões como "lombo de vaca", onde não há nenhuma quantidade e existe uma parte de alimento seguida de um alimento.

  A regra que usa este padrão está na figura 5.13. O contexto contém um sintagma nominal ("lombo") e um sintagma preposicional ("de vaca"), e é verificado se as cabeças dos sintagmas

("lombo" e "vaca") são, respectivamente, uma parte de alimento e um alimento, e se estão ligadas através de uma relação MOD. Se isto se verificar, é criada a dependência INGR(vaca). Mais uma vez, como na regra anterior, é usada uma expressão de negação da dependência INGR, pois verificou-se que esta regra detectava também a frase "4 fatias de pão", já tratada numa das regras anteriores. Como tal, a negação é usada de forma a não ter dependências repetidas.

```
| NP#1, PP#2 |
if( HEAD(#3,#2) & HEAD(#4[partealimento:+],#1) &
   MOD[post:+](#4,#3[alimento:+]) & ~INGR(?*,#3) )
  INGR(#3)
```

Figura 5.13: Regra para determinar um ingrediente numa expressão com parte de alimento e alimento.

- «Número» «quantidade» «parte de alimento» de «parte de alimento» de «alimento»:

  Este padrão é semelhante ao primeiro apresentado, contudo, este não tem a preposição "de" entre a quantidade e a parte de alimento (como se pode ver na figura 5.14), sendo "200 g bife do lombo de vaca" uma expressão onde este padrão se aplica. É necessário efectuar esta diferenciação visto que o sintagma preposicional "de bife" presente no exemplo do primeiro padrão, passa neste caso a um sintagma nominal "bife", portanto, a regra do primeiro padrão não se pode aplicar a estas situações.

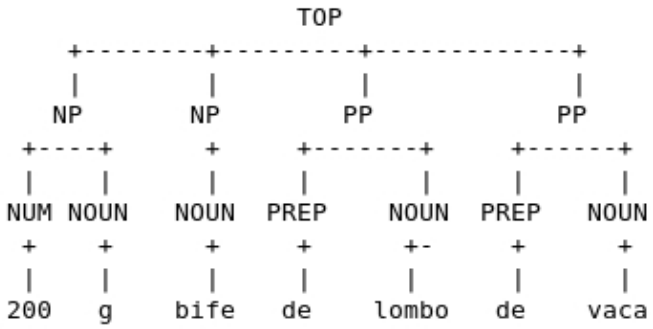

Figura 5.14: Saída do XIP para a expressão "200 g bife de lombo de vaca".

Na figura 5.15 está a regra que é usada para detectar este padrão. O contexto é mais complexo que os anteriores, considerando todos os sintagmas que constituem um exemplo do padrão. Assim, e como se pode ver uma representação na figura 5.14, o contexto consiste em dois sintagmas nominais, sendo que o primeiro tem associado um traço "quant" (de quantidade), seguido de dois sintagmas preposicionais, que contêm uma parte de alimento e o alimento. É depois verificado se a cabeça do primeiro SN tem o traço "meas", se há uma relação de quantidade entre esta cabeça e outra palavra ("200" neste caso) e se as cabeças dos restantes sintagmas são parte de alimento ou alimento (no caso do último sintagma). Caso isto se verifique, a dependência INGR(200, g, vaca) é

criada.

```
| NP#1[quant:+], NP#2, PP#3, PP#4 |
if( HEAD(#5[meas:+],#1) & QUANTD(#5,#6) & HEAD(#7[partealimento:+],#2) &
   HEAD(#8[partealimento:+],#3) & HEAD(#9[alimento:+],#4) )
  INGR(#6,#5,#9)
```

Figura 5.15: Regra para determinar um ingrediente que possui partes sem a preposição "de".

- «Número» «quantidade» «parte de alimento» de «alimento»:

  Este é um caso mais específico do que o anterior, onde também não existe a preposição "de" antes da parte de alimento, mas o ingrediente apenas tem uma parte de alimento, tal como em "200 g lombo de vaca". Estes dois últimos casos não podem ser engolbados apenas numa regra, como o primeiro padrão apresentado, que considerava a preposição "de", devido à ausência da relação MOD usada na regra do primeiro padrão, logo, tem que se considerar como uma situação diferente.

  Na figura 5.16 está a regra usada para este padrão. Comparando com a regra anterior, o contexto tem menos um sintagma (preposicional), a condição é semelhante, no entanto, esta regra tem mais uma expressão (ˆINGR(#3)), que indica que, se já existir alguma dependência INGR com um argumento, e este seja a variável #3, então a dependência é apagada. Isto é necessário pois uma regra anterior é aplicada, extraindo menos informação que esta regra que se está a usar, portanto, é apagada. Quanto à dependência criada, vai ser igual à da regra anterior, INGR(200, g, vaca).

```
| NP#1[quant:+], NP#2, PP#4 |
if( HEAD(#5[meas:+],#1) & QUANTD(#5,#6) & HEAD(#7[partealimento:+],#2) &
   HEAD(#9[alimento:+],#4) & ˆINGR(#3) )
  INGR(#6,#5,#9)
```

Figura 5.16: Regra para determinar um ingrediente com uma parte de alimento e sem a preposição "de".

- «Número» «quantidade» «alimento»:

  Este padrão é usado em frases como "150 g cebolas" ou "2 kg cabrito", onde não há a preposição "de" antes do alimento.

  Para este padrão é usada a regra da figura 5.17. O contexto consiste em dois sintagmas nominais, o primeiro tem o traço "quant", e o segundo é relativo ao alimento. De seguida é preciso verificar se a cabeça do primeiro sintagma tem o traço "meas" e se tem uma relação QUANTD com outra palavra (nesta situação é "150"), e também se a cabeça do segundo sintagma é um alimento. Necessita também de apagar uma dependência criada por uma regra anterior, precisamente por esta

que vai ser criada ter mais informação (relativa às quantidades). No final é criada a dependência INGR(150, g, cebolas).

```
| NP#1[quant:+], NP#2 |
if( HEAD(#3[meas:+],#1) & QUANTD(#3,#4) & HEAD(#5[alimento:+],#2) &
   ^INGR(#5) )
  INGR(#4,#3,#2)
```

Figura 5.17: Regra para determinar um ingrediente sem partes e sem a preposição "de".

- «Número» «alimento» de|com «número» «quantidade»:

  São exemplos da utilização deste padrão as frases "1 sável de 1 kg" ou "1 frango com 1,5 kg". Corresponde a ingredientes que possuem 2 quantidades, e que lhe deverá ser associada a quantidade com unidades de medida, que aparece depois do alimento. Ou seja, deve ser usado o "1 kg" para quantificar o alimento ao invés de "1 sável".

  A regra que implementa este padrão está na figura 5.18. O contexto tem um sintagma nominal ("1 sável") seguido de um sintagma preposicional ("de 1 kg"). Depois de verificar se a cabeça dos dois sintagmas são, respectivamente, um alimento e uma medida, e de verificar se existe uma relação de quantidade entre a cabeça do SP e outra palavra, é removida a dependência criada anteriormente que indicava a quantidade do lado esquerdo do alimento (neste exemplo é removido INGR(1, sável)), pois parte do padrão usado nesta regra emparelha com o padrão «número» «alimento». Como a informação dada por esta regra é mais correcta que a dada pela outra regra, apenas se mantém a dependência criada pela regra actual (neste caso, INGR(1, kg, sável)).

```
| NP#1, PP#2 |
if( HEAD(#3[alimento:+],#1) & HEAD(#4[meas:+],#2) & QUANTD(#4,#5) &
   ^INGR(*?,#3) )
  INGR(#5,#4,#3)
```

Figura 5.18: Regra para determinar um ingrediente numa frase cujo alimento tem duas quantidades.

- «Número» («parte de alimento»)+ de «alimento» de «número» «quantidade»:

  Tal como o padrão anterior, este padrão também tem como objectivo identificar frases que tenham associadas duas quantidades, mas, neste caso, quando o ingrediente é composto por partes de alimento, como "1 bife de lombo de vaca de 300 g".

  Na figura 5.19 está a regra que é usada para este padrão. O contexto contém um sintagmas nominal (correspondente a "1 bife"), seguido de qualquer sintagma (a expressão "?*"), e terminando com dois sintagmas preposicionais, tendo o primeiro a referência ao alimento e o segundo à quantidade e unidade de medida. Faz-se de seguida a verificação de que a cabeça do sintagma nominal

é uma parte de alimento (“bife”), a cabeça do primeiro sintagma preposicional é um alimento (“vaca”), a cabeça do último sintagma preposicional é uma medida ‘(“g”) e se existe uma relação de quantidade entre esta medida e um outro termo. De seguida é removida a dependência INGR que foi criada por outra regra anterior. No final, obtem-se a dependência INGR(300, g, vaca).

```
| NP#1, ?*, PP#2, PP#3 |
if( HEAD(#4[partealimento:+],#1) & HEAD(#5[alimento:+],#2)
   & HEAD(#6[meas:+],#3) & QUANTD(#6,#7) & ^INGR(*?,#5) )
  INGR(#7,#6,#5)
```

Figura 5.19: Regra para determinar um ingrediente numa frase com partes de alimento e duas quantidades.

- «Parte de alimento» de «número» «alimento»:

  Este padrão é usado em frases como “Sumo de 10 limões”, em que se pretende obter a parte de alimento de várias unidades de um alimento.

  Para este padrão é usada a regra presente na figura 5.20. O contexto tem um sintagma nominal e um preposicional seguidos, sendo depois necessário verificar se as cabeças de cada um dos sintagmas são, respectivamente, uma parte de alimento e um alimento, e se existe uma relação QUANTD entre a cabeça do SP e uma outra palavra que seja um número (daí a utilização do traço “num”). Caso estas condições se verifiquem, são criadas duas dependências, INGR(10, limões) e PARTEDE(sumo, limões). Esta última dependência é criada juntamente com a do ingrediente porque tem uma forma específica para este padrão, pelo que não pode ser tratada separadamente, tal como as regras que se apresentarão a seguir para obter as partes de alimento.

```
| NP#1, PP#2 |
if( HEAD(#3[partealimento:+],#1) & HEAD(#4[alimento:+],#2) &
    QUANTD(#4,#5[num:+]) )
  INGR(#5,#4),
  PARTEDE(#3,#4)
```

Figura 5.20: Regra para determinar um ingrediente e parte de alimento de várias unidades do alimento.

- q.b. «alimento» , «alimento» q.b. e «parte de alimento» de «alimento» q.b.:

  A utilização destes padrões acontece em situações em que a quantidade associada ao alimento é “q.b.” (quanto baste), situações que ocorrem diversas vezes na descrição dos ingredientes das receitas, como “Sal q.b.”, “q.b. açúcar” ou “raspa de limão q.b.”.

  As regras das figuras 5.21, 5.22 e 5.23 mostram a utilização destes padrões, bem como a extracção da dependência adequada. As primeiras duas regras são muito semelhantes, apenas muda a posição do advérbio “q.b.” (na primeira fica depois do alimento, e na segunda antes do alimento)

que tem como lema “quanto baste”. Na condição apenas se verifica se já existe uma dependência INGR apenas com o alimento, sendo removida em caso afirmativo para obter uma dependência com mais informação, INGR(q.b., açúcar).

A regra da figura 5.23 (que diz respeito à associação da quantidade “q.b.” a um alimento com partes) verifica ainda se uma das palavras é uma parte de alimento.

```
| NP#1, ADVP#2[lemma:"quanto baste"] |
if( HEAD(#3,#1) & ^INGR(#3) )
 INGR(#2,#3)
```

Figura 5.21: Regra para determinar um ingrediente cuja quantidade é “q.b.” e a quantidade aparece depois do alimento.

```
| ADVP#2[lemma:"quanto baste"], NP#1 |
if( HEAD(#3,#1) & ^INGR(#3) )
 INGR(#2,#3)
```

Figura 5.22: Regra para determinar um ingrediente cuja quantidade é “q.b.” e a quantidade aparece antes do alimento.

```
| NP#1, PP#4, ADVP#2[lemma:"quanto baste"] |
if( HEAD(#3[partealimento:+],#1) & HEAD(#5,#4) & ^INGR(#5) )
 INGR(#2,#5)
```

Figura 5.23: Regra para determinar um ingrediente cuja quantidade é “q.b.” e que contém uma parte de alimento.

Para detectar a dependência PARTEDE, que indica a existência da parte de um alimento, são necessários dois padrões:

- «Número» «quantidade» de («parte de alimento»)+ de «alimento»:

  Este padrão é o mesmo do primeiro padrão para a dependência INGR, e serve para as mesmas frases de exemplo (“200 g de bife de lombo de vaca” e “200 g de lombo de vaca”). Permite obter as dependências PARTEDE(bife, vaca) e PARTEDE(lombo, vaca), de forma a representar as partes que o alimento tem.

  A figura 5.24 contém a regra que aplica este padrão. O contexto tem dois sintagmas preposicionais, podendo ter outros sintagmas entre eles. Relativamente ao segundo sintagma, deve ser iniciado pela preposição “de”, para evitar que esta regra seja aplicada, por exemplo, com um SP iniciado com a preposição “com”. O primeiro sintagma preposicional tem na cabeça uma parte de alimento e o segundo tem um alimento. Depois disto, é necessário verificar se existe uma relação MOD

entre as duas cabeças dos sintagmas. Em caso afirmativo, cria-se a dependência PARTEDE. O contexto é usado desta forma para ter apenas uma só regra de forma a possibilitar a criação das duas dependências na frase "200 g de bife de lombo de vaca". Se o contexto não tivesse a expressão "?*", apenas era possível com esta regra obter PARTEDE(lombo, vaca).

```
| PP#1, ?*, PP#2{prep[lemma:"de"],?*} |
if( HEAD(#3, #1) & HEAD(#4, #2) &
    MOD[post:+](#3[partealimento:+], #4[alimento:+]) )
 PARTEDE(#3, #4)
```

Figura 5.24: Regra para determinar as partes de um alimento com a preposição "de" entre quantidade e alimento.

- «Número» «quantidade» («parte de alimento»)+ de «alimento»:

  Este padrão é semelhante ao anterior, no entanto este não tem a preposição "de" antes da primeira parte de alimento (como em "200 g lombo de vaca"). Tal como já foi explicado para a dependência INGR, isto deve-se à mudança do sintagma que contém a primeira parte de alimento, sendo que, no caso anterior, o sintagma é preposicional, e neste é nominal.

  A regra que implementa este padrão está na figura 5.25. A única diferença em relação à anterior diz respeito ao contexto, visto que, neste caso, o primeiro sintagma é um sintagma nominal e não preposicional.

```
| NP#1, ?*, PP#2{prep[lemma:"de"],?*} |
if( HEAD(#3, #1) & HEAD(#4, #2) &
    MOD[post:+](#3[partealimento:+], #4[alimento:+]) )
 PARTEDE(#3, #4)
```

Figura 5.25: Regra para determinar as partes de um alimento sem a preposição "de" entre quantidade e alimento.

Estas duas regras, juntamente com as dos ingredientes, permitem obter toda a informação acerca de um ingrediente, especificando as suas partes e o alimento em si.

Quanto ao estado do alimento, são também dois os padrões que permitem deduzir o estado em que se encontra um ingrediente:

- «Número» «quantidade» de «alimento» «estado»:

  Um exemplo deste padrão é "100 g de cenoura ralada", e o objectivo é extrair a dependência ESTADO(ralada, cenoura), de forma a indicar o estado em que o alimento se encontra. Este padrão corresponde a situações onde existe uma quantidade com unidades de medida associada ao alimento.

A figura 5.26 contém a regra usada para obter esta dependência. O contexto inclui dois sintagmas, um preposicional e um adjectival, podendo ter entre eles outros sintagmas. O sintagma preposicional vai conter o nome do alimento ("de cenoura") na sua cabeça, e o adjectival contém o estado que o alimento tem. É necessário verificar se existe uma relação MOD entre o alimento e o seu estado, e, caso isto se verifique, é então criada a dependência ESTADO(ralada, cenoura).

```
| PP#1 , ?*, AP#4|
if( HEAD(#2, #1) & HEAD(#3, #4) &
    MOD[post:+](#2[alimento:+], #3[estado:+]) )
 ESTADO(#3, #2)
```

Figura 5.26: Regra para deteminar o estado de um alimento com quantidades associadas.

- «alimento» «estado»:

  Este padrão pode-se aplicar a situações em que não há quantidades associadas (como em "pão ralado"), ou, se houver, dizem respeito às unidades do alimento, e não a unidades de medida (como em "2 tomates maduros").

  A regra que aplica este padrão está na figura 5.27. Relativamente à regra anterior, apenas muda o primeiro sintagma, que, agora é um sintagma nominal, pelo facto de não ter nenhuma preposição antes do nome do alimento. A condição é a mesma, pelo que no final é obtida a dependência ESTADO(ralado, pão).

```
| NP#1 , ?*, AP#4|
if( HEAD(#2, #1) & HEAD(#3, #4) &
    MOD[post:+](#2[alimento:+], #3[estado:+]) )
 ESTADO(#3, #2)
```

Figura 5.27: Regra para deteminar o estado de um alimento sem quantidades associadas.

5.3 Extracção das Acções

Da mesma forma que, para a extracção dos ingredientes, são também obtidas as dependências que contêm a informação sobre as tarefas que envolvem a preparação da receita. Nas secções 5.3.1 e 5.3.2 estão presentes, respectivamente, a representação e as regras que permitem extrair as dependencias relativas às acções.

5.3.1 Representação das Acções

As acções têm uma forma mais complexa de serem representadas, devido a terem que considerar a utilização de um ou mais alimentos e utensílios. Inicialmente foram consideradas três abordagens diferentes para realizar a representação das acções, consistindo a primeira em apenas uma dependência chamada "ACCAO", tendo como argumentos o nome da acção, o alimento e outros parâmetros. Por exemplo, para a frase "Corte o bacalhau em quadrados", essa representação seria ACCAO(cortar, bacalhau, em quadrados). Esta forma de representação é muito limitativa, na medida em que só foram consideradas frases simples, não estando contemplados casos como a presença de mais do que um alimento, e também porque podia conduzir a uma grande complexidade de argumentos.

Depois da análise de frases mais complexas e que evidenciavam os problemas da representação anterior, surgiram outras três representações diferentes que permitiam resolver esses problemas. A primeira dessas representações consiste nas dependências *Acção*(argumento1), *ARG*(argumento2), ..., *ARG*(argumento$_n$), sendo *Acção* o nome da acção, e ARG um conjunto de dependências que indicavam os restantes argumentos da dependência relativa à acção. Desta forma, o terceiro módulo tem mais trabalho, pois tem que associar às acções as dependências ARG que lhe dizem respeito, ou seja, ao aparecer uma acção, todas as dependências ARG que surgem até haver outra acção são associadas à primeira acção. Esta representação produz mais dependências e não é uma representação ideal, podendo haver erros de associação entre as dependências ARG e a acção respectiva. Exemplificando, para a frase "Junte os ovos, a farinha e o açúcar", podem-se obter as dependências JUNTAR(ovos), ARG(farinha) e ARG(açúcar) com esta representação.

A segunda representação consiste em ter apenas um conjunto de dependências *Acção*(argumento) em que cada argumento (alimento ou utensílio) é colocado numa dependência diferente. Esta representação permite reduzir o número de regras a serem criadas, devido a ter uma estrutura simples, e por não necessitar, como no caso anterior, de mais uma dependência. Depois, no módulo de transformação de saída agrupam-se todas as dependências com o mesmo nome que estejam seguidas, que indicam todos os argumentos relativos a uma acção. Esta representação tem o problema de identificar como uma única acção, duas ou mais acções iguais que estejam seguidas na mesma frase, como em "Bata metade do açúcar e as gemas num alguidar, e bata o restante açúcar com as claras noutro alguidar", em que são obtidas quatro dependências "BATER", sendo consideradas pelo método descrito apenas como uma acção.

A última representação consiste em colocar todos os argumentos relativos a uma acção em apenas uma dependência, formando *Acção*(argumento1, argumento2, ..., argumento$_n$). Por exemplo, para a frase indicada na primeira representação, com esta representação obtém-se JUNTAR(ovos, farinha, açúcar). Esta forma é muito complexa e implica que se tenha um aumento de regras e da complexidade das mesmas, pois envolve a criação de uma regra para cada número de argumento da dependência.

A representação escolhida resulta de algumas alterações efectuadas à segunda representação referida, no que diz respeito às situações em que as acções têm mais que um alimento. Esta representação dá origem a diversos tipos de acções, consideradas na seguinte lista (sendo que *Acção* representa o nome de uma acção):

- Acção Primitiva de Alimentos e Utensílios e Acção Composta:
  - *Acção*(alimento) - Usado em situações em que há apenas a referência ao alimento (Ex: "Corte as cebolas em rodelas" origina a dependência CORTAR_RODELAS(cebolas));
  - *Acção*(alimento, e) - Usado quando a acção se refere a mais do que um alimento (Ex: "Bata os ovos e o açúcar" origina as dependências BATER(ovos, e) e BATER(açúcar, e));
  - *Acção*(alimento, utensílio) - Usado quando a acção se refere a um alimento e a um utensílio (Ex: "Deita-se o azeite na frigideira" origina a dependência ADICIONAR(azeite, frigideira));
  - *Acção*(alimento, utensílio, e) - Usado nas situações que envolvem um utensílio e mais do que um alimento (Ex: "Deitam-se os ovos e o açúcar num tacho" origina as dependências ADICIONAR(ovos, tacho, e) e ADICIONAR(açúcar, tacho, e));
  - *Acção*(utensílio) - Usado nas situações em que apenas existe a referência a um utensílio (Ex: "Deita-se o azeite na frigideira" origina a dependência ADICIONAR(azeite, frigideira)).
- Acção Primitiva de Alimentos:
  - *Acção*(alimento) - Para as situações em que a acção se refere a apenas um alimento (Ex: "Ponha o bacalhau de molho" dá origem à dependência DEMOLHAR(bacalhau));
  - *Acção*(alimento, e) - Para situações em que a acção se refere a mais do que um alimento (Ex: "Deixe alourar os alhos e as cebolas" dá origem às dependências ALOURAR(alhos, e) e ALOURAR(cebolas, e)).
- Acção Primitiva de Utensílios:
  - *Acção*(utensílio1, utensílio2) - Quando a acção só é possível com dois utensílios (Ex: "Leve o tacho ao lume" dá origem à dependência COLOCAR_CIMA(tacho, fogão));
  - *Acção*(utensílio) - Quando a acção apenas requer um utensílio (Ex: "Agitando o tacho de vez em quando" dá origem à dependência AGITAR(tacho)).
- Acção Primitiva de Espera:
  - *Acção*(tempo, unidade de tempo) - Usado na acção Espera Tempo (Ex: "... durante 30 minutos..." origina a dependência ESPERA_TEMPO(30, minutos));

– *Acção*(alimento, temperatura) - Usado na acção Espera Temperatura Alimento (Ex: "Espere que o frango atinja os 70ºC" origina a dependência ESPERA_TEMPERATURA_ALIMENTO(frango, 70ºC));

– *Acção*(utensílio, temperatura) - Usado na acção Espera Temperatura Utensílio (Ex: "Espere que o forno esteja a 200ºC" origina a dependência ESPERA_TEMPERATURA_UTENSILIO(forno, 200ºC));

– *Acção*(alimento com estado) - Usado na acção Espera Transformação (Ex: "Espere que o arroz fique cozido" origina as dependências ESPERA_TRANSFORMACAO(arroz) e ESTADO(cozido, arroz), acabando por ser semelhante a *Acção*(alimento)).

Existe ainda outra forma de representar as acções, presente em todas as variantes acima apresentadas, que diz respeito a acções que não têm nem utensílios nem alimentos associados, como nas frases "Deixe alourar" ou "Bate-se muito bem". Nesta situação, o ideal seria ter uma representação da forma ALOURAR() ou BATER(), no entanto, por limitações do sistema XIP, esta forma não é possível, pelo que tem que se recorrer a outra forma para representar esta situação. Assim, é colocado um argumento na dependência, o verbo que corresponde à acção, e associa-se o traço "semargumentos" à dependência, de forma a que, no módulo seguinte, esta dependência seja interpretada como uma dependência que não tem argumentos.

Em geral, cada acção vai ter um conjunto de verbos ou expressões que a identificam e diferenciam relativamente a outras acções. Esta associação entre as acções e os conjuntos de verbos ou expressões está presente nas tabelas do anexo B.

5.3.2 Regras para a Extracção das Acções

Como se viu na secção anterior, as dependências podem ter formas diferentes relativamente aos seus argumentos, que variam conforme o tipo de acção e com o número de alimentos a que a acção se refere. Nesta secção são apresentadas as regras que permitem criar cada um dos tipos de dependências, tendo sido criadas no total 328 regras.

- *Acção*(alimento):

  Esta forma é usada nas acções compostas, acções primitivas de alimentos, acções primitivas de alimentos e utensílios e em uma acção primitiva de espera. Exemplos de frases que originam esta dependência são "Corte as cebolas em rodelas", "Deixe arrefecer o leite" e "Descasque as cebolas", e as regras que tratam destas situações estão nas figuras 5.28, 5.29 e 5.30, respectivamente.

  Na primeira regra o contexto consiste no verbo "cortar" seguido de um sintagma nominal que contém um alimento ou parte de alimento (o ";" denota opcionalidade no contexto), terminando

com um adjectivo cujo lema da última palavra é "rodela". Desta forma é possível obter a dependência CORTAR_RODELAS com um alimento como argumento. Na segunda regra o contexto é semelhante, não tendo o adjectivo no final, e se houver já uma dependência ARREFECER, esta é apagada e cria-se uma nova com o alimento presente dentro do sintagma nominal. A última regra só necessita do verbo no contexto (neste caso "descascar"), verificando depois se existe uma relação CDIR entre o verbo e uma outra palavra que seja um alimento ou uma parte de alimento (a opcionalidade nesta parte é dada por "|").

```
| #1VERB[lemma:"cortar"], NP{?*, #2[alimento:+];#2[partealimento:+]},
    AP{ADJ{?*, noun[lemma:"rodela"]}} |
 CORTAR_RODELAS(#2)
```

Figura 5.28: Regra usada para determinar a dependência CORTAR_RODELAS.

```
| #1VERB[lemma:"arrefecer"], NP{?*, #2[alimento:+];#2[partealimento:+]} |
 if( ^ARREFECER(?) )
  ARREFECER(#2)
```

Figura 5.29: Regra usada para determinar a dependência ARREFECER.

```
| #1VERB[lemma:"descascar"] |
 if( CDIR(#1, #2[alimento:+])|CDIR(#1,#2[partealimento:+]) )
  DESCASCAR_FACA(#2)
```

Figura 5.30: Regra usada para determinar a dependência DESCASCAR_FACA.

- *Acção*(alimento, e):

  Esta forma é usada nas acções compostas, acções primitivas de alimentos e acções primitivas de alimentos e utensílios. Um exemplo de uma frase que origina uma dependência deste tipo é "Deixe alourar os alhos e as cebolas", sendo a regra da figura 5.31 usada para determinar as dependências. O contexto contém os diversos verbos que são utilizados para a dependência alourar ("alourar", "aloirar" e "dourar"), seguido de quaisquer palavras, e acabando com um NP que vai ter um alimento ou parte de alimento. Na condição verifica-se se existe uma relação de coordenação (COORD) entre uma palavra (neste caso a conjunção "e" na variável #3) e o alimento presente no contexto, e se também não existe já uma dependência ALOURAR criada anteriormente. Nesta situação são depois criadas as dependências ALOURAR(alhos, e) e ALOURAR(cebolas, e). Para as outras acções que originam o mesmo tipo de dependência, a estrutura da regra é semelhante.

```
| #1VERB[lemma:"alourar"];#1VERB[lemma:"aloirar"];#1VERB[lemma:"dourar"],
  ?*, NP{?*, #2[alimento:+];#2[partealimento:+]} |
 if( COORD(#3,#2) & ~ALOURAR(#2, ?*) )
  ALOURAR(#2,#3)
```

Figura 5.31: Regra usada para determinar a dependência ALOURAR.

- *Acção*(alimento, utensílio):

  Esta forma é usada nas acções primitivas de alimentos e utensílios e nas acções compostas, em que a acção envolve um alimento e um utensílio. Com as frases "Leve a gelatina ao frígorífico" e "Numa tigela bata as natas" é possível obter uma dependência deste tipo, sendo as regras presentes nas figuras 5.32 e 5.33 as que permitem extrair a dependência para ambos os casos. Na primeira regra, o contexto compõe-se de um verbo que indica a acção (neste caso são vários os que indicam) seguido de um sintagma nominal que contém o alimento ou parte de alimento, terminando com um sintagma preposicional contendo um utensílio, restringindo-se neste caso a um frigorífico. Não é necessário realizar nenhuma verificação, pelo que é criada a dependência com um alimento e um utensílio associados à acção. A segunda regra tem um contexto diferente, iniciando-se com um sintagma preposicional que contém o utensílio, seguido de qualquer segmento, tendo no final um verbo com lema "bater". Este contexto não tem nenhum sintagma que indica o alimento visto que na condição é apagada uma dependência já criada anteriormente, e que tem referência ao alimento em questão.

```
| #1VERB[lemma:"levar"];#1VERB[lemma:"guardar"];#1VERB[lemma:"pôr"],
  NP{?*, #3noun[alimento:+];#3noun[partealimento:+]},
  PP{?*, #2noun[lemma:"frigorífico"]} |
  GUARDAR(#3, #2)
```

Figura 5.32: Regra usada para determinar a dependência GUARDAR.

```
| PP{?*, #2[utensilio:+]}, ?*, #1VERB[lemma:"bater"] |
 if( ^BATER(#3) )
  BATER(#3,#2)
```

Figura 5.33: Regra usada para determinar a dependência BATER (1).

- *Acção*(alimento, utensílio, e):

  Esta forma, tal como a anterior, é usada nas acções primitivas de alimentos e utensílios e nas acções compostas, em frases como "Numa tigela, bata as natas e as claras", com um utensílio e mais de um alimento. Na figura 5.34 está presente a regra que permite extrair a dependência deste tipo.

Comparando com a regra da figura 5.33, as diferenças dizem respeito à adição de um sintagma nominal no contexto que contém o alimento ou parte de alimento, e na condição é verificado se existe uma coordenação entre alimentos, e é apagada a dependência criada com a regra da figura 5.33, devido a se ter mais conhecimento com esta regra que se está a aplicar. Nesta situação, no final são criadas duas dependências, BATER(claras, tigela, e) e BATER(natas, tigela, e).

```
| PP{?*, #2[utensilio:+]}, ?*, #1VERB[lemma:"bater"], ?*,
  NP{?*, #3[alimento:+];#3[partealimento:+]} |
 if( COORD(#4,#3) & ^BATER(#3,#4) )
  BATER(#3,#2,#4)
```

Figura 5.34: Regra usada para determinar a dependência BATER (2).

- *Acção*(utensílio1, utensílio2):

  Esta forma é usada nas acções primitivas de utensílios, mais concretamente, pelas acções "Colocar em Cima", "Colocar Dentro", "Tapar", "Ligar", "Retirar de Cima" e "Retirar de Dentro". Um exemplo de uma frase que origina este tipo de dependência é "Leve o tacho ao lume". A regra que cria a dependência para este caso está na figura 5.35. O contexto é apenas o verbo "levar", ficando a cargo da condição verificar se há uma relação de complemento directo entre o verbo e um utensílio, e se há uma relação MOD entre o verbo e uma palavra com lema "lume" ou "fogo" (esta última é mais utilizada em português brasileiro). Caso se verifique, cria-se a dependência, sendo o primeiro argumento o utensílio descoberto, e como segundo argumento o utensílio fogão, que é criado apenas para a dependência.

```
| #1VERB[lemma:"levar"] |
 if( CDIR(#1, #2[utensilio:+]) &
     (MOD(#1, #3[lemma:"lume"])|MOD(#1, #3[lemma:"fogo"])) )
  COLOCAR_CIMA(#2,##NOUN[lemma="fogão"])
```

Figura 5.35: Regra usada para determinar a dependência COLOCAR_CIMA.

- *Acção*(utensílio):

  Esta forma é usada nas acções primitivas de utensílios, mais concretamente, pelas acções "Abrir", "Agitar", "Colocar Pás" e "Desligar", e também nas acções compostas e acções primitivas de alimentos e utensílios. Por exemplo, uma frase que origina uma dependência deste tipo é "Agitando o tacho de vez em quando", sendo a dependência obtida através da regra presente na figura 5.36. Nesta regra o contexto é constituído por um verbo seguido de um sintagma nominal que contém o utensílio, sendo isto suficiente para determinar a dependência "AGITAR".

```
| #1VERB[lemma:"agitar"], NP{?*, #2[utensilio:+]} |
  AGITAR(#2)
```

Figura 5.36: Regra usada para determinar a dependência AGITAR.

- *Acção*(tempo, unidade de tempo):

  Esta forma é usada pela acção "Espera Tempo", em frases como "Passados 30 minutos, ...". A regra que permite extrair esta dependência, na figura 5.37, considera como contexto um sintagma nominal ou um nome, cuja primeira palavra é um número e a segunda é uma expressão temporal (anotada com traço "time"). Desta forma é possível criar a dependência ESPERA_TEMPO(30, minutos).

```
| NOUN{#1[num:+], #2[time:+]};NP{#1[num:+], #2[time:+]} |
  ESPERA_TEMPO(#1, #2)
```

Figura 5.37: Regra usada para determinar a dependência ESPERA_TEMPO.

- *Acção*(alimento, temperatura):

  A acção "Espera Temperatura Alimento" é a única acção que usa esta forma em frases como "Espere que a água esteja a 70ºC", para indicar um tempo de espera que termina quando o alimento atinge uma determinada temperatura. A figura 5.38 contém a regra que permite extrair a dependência para esta acção. O contexto é constituído pelo verbo "esperar", seguido de uma subcláusula (SC) iniciada por "que" e tendo no fim um alimento, seguindo-se outro verbo que tanto pode ser "estar", "atingir" ou "ficar" e terminando com um sintagma nominal que contém a temperatura.

```
| #1VERB[lemma:"esperar"],
  SC{pron[lemma:"que"], NP{?*, #2[alimento:+];#2[partealimento:+]},
  VF{VERB[lemma:"estar"];VERB[lemma:"atingir"];VERB[lemma:"ficar"]} },
  NP{?*, noun#3} |
ESPERA_TEMP_ALIMENTO(#2,#3)
```

Figura 5.38: Regra usada para determinar a dependêndia ESPERA_TEMP_ALIMENTO.

- *Acção*(utensílio, temperatura):

  Esta forma é apenas usadada pela acção "Espera Temperatura Alimento", e tem um objectivo semelhante ao da forma anterior, excepto que esta diz respeito à temperatura de um utensílio e não de um alimento. "Espere que o forno atinja os 200ºC" e "Deixe o forno aquecer até 180ºC" são dois exemplos de frases que estão incluídas nesta forma e que dão origem a uma dependência

"ESPERA_TEMP_UTENSILIO". Na figura 5.39 está presente a regra que permite extrair uma dependência para esta forma.

```
| #1VERB[lemma:"esperar"],
  SC{pron[lemma:"que"], NP{?*, #2[utensilio:+]},
  VF{VERB[lemma:"estar"];VERB[lemma:"atingir"];VERB[lemma:"ficar"]}},
  NP{?*, noun#3} |
ESPERA_TEMP_UTENSILIO(#2,#3)
```

Figura 5.39: Regra usada para determinar a dependência ESPERA_TEMP_UTENSILIO.

- *Acção*():

  Esta forma diz respeito a acções que não têm nem alimentos nem utensílios associados, como em "Bate-se muito bem", servindo a regra da figura 5.40 para extrair este tipo de dependência para a acção "BATER". O contexto é composto apenas pelo verbo, e para assegurar que esta regra não seja seleccionada em frases mais complexas como "Bata os ovos", devido ao simples contexto, é verificado se não existem outras dependências "BATER" já criadas, tendo tanto um, dois ou três argumentos. Verifica-se também se o verbo não está no particípio passado, como em "Junte as natas batidas", situação que não deve ser considerada como uma acção.

```
| #1VERB[lemma:"bater"] |
if( ~BATER(?) & ~BATER(?,?) & ~BATER(?,?,?) & HEAD(#2,#1)
    & #2[pastpart:~] )
  BATER[semargumentos=+](#2)
```

Figura 5.40: Regra usada para determinar a dependência BATER (3).

6 Transformação da Saída

O módulo de transformação de saída converte as dependências obtidas no módulo anterior pela cadeia de processamento XIP em código SQL de forma a que a informação respeitante às receitas seja colocada numa base de dados.

As alternativas consideradas para a construção deste módulo foram a utilização de XSLT e Java, por ambos possibilitarem a manipulação de ficheiros XML, contudo, como o XSLT não tem uma forma de interagir directamente com uma base de dados, esta hipótese foi abandonada. Note-se, contudo, que é possível introduzir dados numa base de dados através de XSLT, escrevendo o código SQL para a saída e correndo esse código na consola através do comando "mysql". Todavia, não era possível ir buscar informação necessária à base de dados (os identificadores de alimentos, por exemplo) e continuar o processamento do XML resultante do XIP. Devido a estas razões recorreu-se à linguagem Java, pelo facto de ser fácil a manipulação de ficheiros XML, mas, principalmente, de ser possível comunicar directamente com bases de dados.

Este módulo está dividido em três componentes, sendo o primeiro responsável por converter os estados obtidos nas dependências "ESTADO" para o nome que está presente na base de dados , visto que alguns dos estados são particípios passados, logo o seu lema vai corresponder ao verbo. Por exemplo, em "carne assada", a dependência criada é "ESTADO(assar, carne)" (usando a opção do XIP para mostrar apenas os lemas), então este componente vai converter "assar" para "assado", que é o estado correcto e o que está introduzido na base de dados.

O segundo componente tem como funções analisar o XML de saída do XIP e também aproveitar apenas as dependências que interessam para este domínio, para posteriormente serem processadas e colocadas na base de dados. O terceiro componente gere as comunicações com a base de dados, sendo aqui criado e executado o código SQL.

Existem ainda dois programas criados, um para os ingredientes e outro para as acções, que são responsáveis por, respectivamente, identificar os parâmetros das dependências INGR (ou seja, a quantidade, unidade e alimento, caso os tenham), e identificar o tipo de acção, pois cada tipo de acção referido na secção 5.3.1 tem um tratamento diferente. Na figura 6.1 apresenta-se a arquitectura deste módulo.

As secções 6.1 e 6.2 contêm uma descrição mais detalhada dos dois últimos componentes deste módulo.

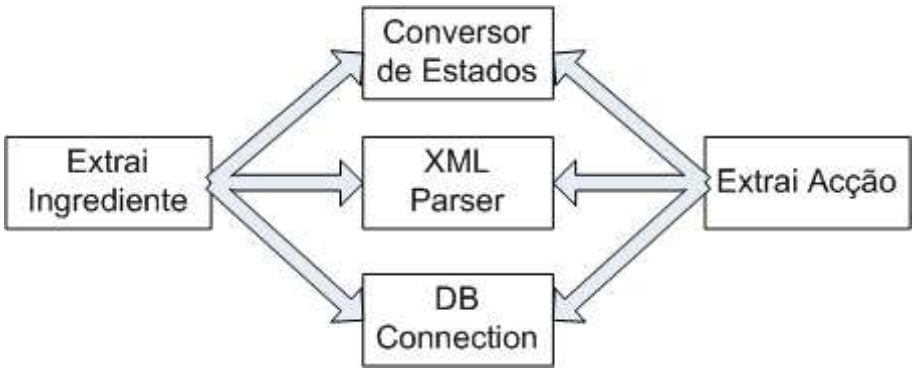

Figura 6.1: Componentes constituintes do terceiro módulo do sistema.

6.1 Análise do XML

A análise que é feita ao ficheiro XML proveniente do XIP consiste em extrair apenas os elementos do XML que dizem respeito às dependências necessárias. Na figura 6.2 está presente um excerto do ficheiro em XML criado pelo XIP contendo um elemento referente a uma dependência (neste caso "BATER"). Este ficheiro inicia-se com a descrição da frase e das suas palavras em termos de categorias gramaticais e traços, terminando com uma listagem das depedências dessa frase.

O atributo "name" do elemento "DEPENDENCY" indica o nome da dependência presente, e os elementos "PARAMETER" indicam os parâmetros que essa dependência possui, tendo os atributos desta dependência informação sobre a dependência. Desta forma, o atributo "ind" diz respeito ao índice do parâmetro, o atributo "num" referencia outro elemento no documento com informação sintáctica sobre a palavra, e o atributo "word" indica qual a palavra que é parâmetro da dependência. Neste caso, este elemento corresponde à dependência "BATER(pimento, copo misturador, e)".

No caso de a análise ser feita para a detecção de ingredientes, apenas são aproveitados os elementos "DEPENDENCY" cujo atributo "name" seja "INGR", "PARTEDE" e "ESTADO". Estes elementos são colocados numa lista para serem posteriormente analisados. Já para a detecção de acções, os elementos "DEPENDENCY" aproveitados são "PARTEDE", "ESTADO" e todas as acções definidas.

Para as acções, é também adicionado ao elemento "PARAMETER" de cada acção um atributo que indica se esse parâmetro é um alimento ou um utensílio. Para tal, utiliza-se o atributo "num" deste elemento, e percorre-se o XML até encontrar o elemento "NODE" que tenha o número igual ao do atributo. Nesse elemento "NODE" está a informação sobre os traços associados à palavra, e, caso essa palavra tenha um dos traços "alimento" ou "utensilio", é adicionado ao elemento um atributo chamado "atributo" que indica qual dos dois traços se verifica. Esta adição tem como objectivo facilitar a detecção do tipo de acção (descritos na secção 5.3.1), pois sem isto era mais complicado saber se uma acção com um argumento seria uma acção referente a um alimento ou a um utensílio, visto que apenas é enviado

uma lista com elementos "DEPENDENCY" que não contêm essa informação.

No final é enviada uma lista com todos os elementos "DEPENDENCY" relevantes tanto para os ingredientes como para as acções.

```
...
<DEPENDENCY name="BATER">
<PARAMETER ind="0" num="26" word="pimento"/>
<PARAMETER ind="1" num="69" word="copo misturador"/>
<PARAMETER ind="2" num="32" word="e"/>
</DEPENDENCY>
...
```

Figura 6.2: Excerto do ficheiro XML com a dependência BATER.

6.2 Geração de SQL e Introdução na Base de Dados

Depois de realizar a análise do ficheiro XML e de obter as depedências relevantes, procede-se à criação de código SQL de forma a que estas dependências sejam colocadas na base de dados. Mas, antes disso, é necessário, no caso dos ingredientes, identificar o alimento, a quantidade, a unidade de medida e os estados do alimento, e, no caso das acções, identificar o tipo de acção, os alimentos e utensílios associados e os estados dos alimentos, de modo que a criação do SQL seja facilitada. Relativamente ao tipo de acção, para cada tipo é invocado um método diferente, diminuindo assim a complexidade comparativamente à situação de tratar uma acção por método, devido a serem mais de cem acções e, principalmente, porque uma acção pode ter mais do que um tipo de acção.

Para a parte dos ingredientes, o algoritmo que realiza a inserção na base de dados é o seguinte:

1. Obter *id* do alimento;
2. Obter *id* do alimento com estado;
3. Inserir estados do alimento;
4. Obter *id* da medida (quantidade e unidade);
5. Criar a *query*;
6. Inserir o ingrediente.

O algoritmo inicia-se com a obtenção do *id* do alimento que foi colocado previamente na tabela "Alimento", criando-se de seguida um alimento com estado, que necessita do *id* do alimento, e adicionam-se os estados que esse alimento tem. Prossegue-se obtendo o *id* da medida (que referencia a quantidade

do alimento), indo buscar à tabela "Grandeza" a unidade e verificando se já existe alguma medida igual que tenha sido colocado anteriormente. Nesta altura já se tem todos os dados necessários para introduzir o ingrediente e, portanto, cria-se e executa-se a *query*. De notar que existe um alimento com estado para cada ingrediente, mas uma medida pode estar associada a diversos ingredientes, pois o alimento com estado necessita de ser diferenciado de receita para receita, algo que não acontece com a medida.

Para a parte das acções, o algoritmo que insere as acções da preparação é o seguinte:

1. Obter *id* da acção;
2. Obter *id* dos utensílios (se necessário);
3. Obter *id* da medida (tempo ou temperatura, se necessário);
4. Adicionar a tarefa;
5. Para cada alimento (se necessário):
   (a) Obter *id* do alimento;
   (b) Obter *id* do alimento com estado;
   (c) Inserir estados do alimento;
   (d) Associar a tarefa ao alimento com estado.
6. Adicionar a tarefa ordenada.

Para inserir as acções primeiro tem que se obter o *id* da acção que, tal como os alimentos, também foi previamente colocado na base de dados. De seguida, dependendo do tipo de acção, são obtidos o *id* dos utensílios (para acções que tenham nos seus argumentos utensílios) e o *id* da medida, que tanto se pode referir a tempo (na acção ESPERA_TEMPO) como a temperatura (nas acções ESPERA_TEMP_ALIMENTO e ESPERA_TEMP_UTENSILIO). É adicionada a tarefa com a referência à acção e a possíveis utensílios e medida, e depois itera-se a lista dos alimentos realizando os mesmo passos que na parte de ingredientes, e associa-se esse alimento à tarefa. Por fim cria-se uma tarefa ordenada que relaciona a receita com o conjunto de tarefas que a constituem, e um número que indica a ordem da tarefa na receita.

Após estes passos, os ingredientes e as acções ficam introduzidas na base de dados conforme a ontologia.

Quando o sistema processa uma receita que tem como um dos ingredientes "200 g de açúcar", o módulo que realiza o processamento das receitas extrai a dependência INGR(200, g, açúcar). A partir desta dependência, o módulo de transformação da saída cria o SQL correspondente e que vai permitir

```
INSERT INTO Ingrediente(IDReceita, IDMedida, IDAlComEstado, nome)
       VALUES("2", "24", "11", "200 g de açúcar")
```

Figura 6.3: Código SQL resultante do processament de um ingrediente.

introduzir o ingrediente na base de dados. A figura 6.3 mostra o código SQL que resulta da análise desta dependência.

Este código insere na tabela "Ingrediente" o ingrediente processado. O conteúdo da tabela consiste em chaves estrangeiras para outras tabelas, que contêm a receita, a medida e o alimento com estado, respectivamente. Estas chaves são obtidas inquirindo as tabelas "Medida" por "200 g" e "AlimComEstado" por "açúcar", e, caso não tenham os valores, são criado durante o processamento. O ingrediente é também associado à receita, através do campo "IDReceita".

Relativamente às acções, caso a receita tenha a frase "Junte as cebolas", a dependência que é extraída é JUNTAR(cebolas), e o código SQL criado está presente na figura 6.4.

```
INSERT INTO Tarefa(IDAccao, IDUtensilio1, IDUtensilio2, IDTemp)
       VALUES("73", "0","0","0")

INSERT INTO AlimentosTarefa(IDTarefa,IDAlimComEstado)
       VALUES("44","19")

INSERT INTO TarefaOrdenada(IDReceita, numeroOrdem, IDTarefa)
       VALUES("2","3","44")
```

Figura 6.4: Código SQL resultante do processament de uma frase com uma acção.

Inicialmente é criada uma entrada na tabela "Tarefa", que contém, neste caso, apenas a referência à acção correspondente (com o valor 73). Os outros campos são usados quando a acção envolve um ou dois utensílios, uma temperatura ou um espaço de tempo, que não é o caso, portanto, esses campos são preenchidos com o valor "0". De seguida realiza-se a associação entre os alimentos pertencentes à acção e a própria acção. Como neste caso há só um alimento, a tabela "AlimentosTarefa" vai ter apenas uma entrada referente ao alimento "cebola", que tem o *id* 19 da tabela "AlimComEstado", e à tarefa criada na *query* anterior, cujo *id* é 44. Caso a acção tivesse mais ingredientes, mais *queries* semelhantes a esta seriam criadas. Para finalizar, tem que se atribuir um número de ordem à tarefa de forma a que seja possível saber a ordenação das acções na receita. Para isso, introduz-se uma entrada na tabela "TarefaOrdenada", indicando o *id* da receita, o número de ordem da tarefa e o *id* da tarefa.

7 Avaliação

7.1 Avaliação dos Ingredientes

A avaliação dos ingredientes consiste em verificar se as dependências extraídas pelo XIP relativas à parte dos ingredientes são iguais às esperadas. Desta forma, e como são três as dependências relacionadas com os ingredientes (INGR, ESTADO e PARTEDE), um ingrediente só é considerado correctamente extraído caso sejam extraídas todas as dependências que seja possível para cada ingrediente.

Por exemplo, para o ingrediente "4 tomates maduros", devem ser extraídas as dependências INGR(4, tomates) e ESTADO(maduro, tomate). Se uma das dependências não for extraída (mesmo que seja a do estado do alimento), o ingrediente não é considerado correcto para efeitos de avaliação.

Das 48 receitas constituintes do corpus de receitas, 40 foram usadas para esta avaliação, dando um total de 353 ingredientes. A tabela 7.1 contém os resultados obtidos nas três avaliações efectuadas.

Como se pode verificar, na primeira avaliação apenas cerca de um terço dos ingredientes foram correctamente identificados, valor que se deve a vários factores, tal como:

- Unidades de medida incorrectamente usadas como em "200 g. de açúcar", ou "400 grs de coelho";
- Formato de ingredientes diferente, como "200 g carne de vaca", em que não é usada a preposição "de";
- Utilização de valores fraccionários, como "1/2 chávena de chá de água";
- Medidas não consideradas, como "lata" ou "pacote" (por exemplo, "1 lata de leite condensado").

	1ª Avaliação	2ª Avaliação	3ª Avaliação
Data	Março 2007	Abril 2007	Julho 2007
Ingredientes Correctos	128	268	327
Ingredientes Dados	212	337	351
Total de Ingredientes	353	353	353
Precisão	60%	79%	93%
Cobertura	36%	76%	93%

Tabela 7.1: Avaliação dos Ingredientes

Na segunda avaliação, os resultados foram superiores (76% de cobertura) devido a ter-se considerado os factores que provocaram um baixo valor de cobertura na primeira avaliação para a construção de regras. No entanto, havia ainda algumas situações que não eram correctamente identificadas, tal como a utilização da expressão "q.b." associada aos ingredientes.

Para a terceira avaliação teve-se em conta o uso de "q.b." juntamente com os ingredientes, facto que permitiu subir a precisão e a cobertura para 93%. Após esta avaliação, e, apesar dos resultados obtidos serem bastante satisfatórios, ainda existem algumas situações que não são detectadas, situações que estão referidas no capítulo 8.

7.2 Avaliação das Acções

Da mesma forma que a avaliação dos ingredientes, a avaliação das acções tem por objectivo verificar se as dependências extraídas pelo XIP são iguais às que se esperam. Para se considerar uma acção correcta, devem ser extraídas todas as dependências que a ela estejam associadas, ou seja, para a frase "Junte o açúcar e a farinha", devem ser extraídas ambas as dependências "JUNTAR(açúcar, e)" e "JUNTAR(farinha, e)" para a acção ser considerada correctamente extraída.

Como para a avaliação dos ingredientes, foram usadas 40 receitas, o que dá um total de 510 acções. A tabela 7.2 inclui os resultados obtidos na avaliação realizada.

Os valores obtidos na avaliação são razoáveis, no entanto há que notar que estes valores se referem a um corpus de treino, onde as regras estão adaptadas a esse corpus e, portanto, no corpus de avaliação, os valores de precisão e cobertura deverão ser menores. Ainda assim é possível identificar alguns factores que impediram melhores resultados:

- Existência de quantidades associadas aos alimentos, como em "Junte 200 g de açúcar e 300 g de farinha", que origina apenas a dependência JUNTAR(açúcar);
- Coordenação entre alimentos que têm partes de alimentos, é uma limitação do XIP, que não indica correctamente todas as dependências COORD entre todos os alimentos, e assim não permite extrair correctamente as dependências que recorrem à dependência COORD;
- Acções que são facilmente compreensíveis em língua natural e que se associam a uma acção (das definidas na ontologia), mas que são difíceis de serem analisadas de uma forma genérica, como "Introduza meia casca de ovo, previamente lavada, numa pequena cavidade que abriu no centro de cada bife";

	1ª Avaliação
Data	Agosto 2007
Acções Correctas	385
Acções Dadas	453
Total de Acções	510
Precisão	85%
Cobertura	75%

Tabela 7.2: Avaliação das Acções

7.3 Avaliação da Introdução na Base de dados

Nesta secção é considerada a avaliação realizada ao módulo que realiza a transformação da saída, no qual os ingredientes e as acções são inseridos na base de dados. Como tal, aqui é considerada a correcta inserção de ambos na base de dados, seguindo os mesmos critérios que estão explicados nas secções anteriores. Sendo assim, o total de respostas correctas é dado pelo número de dependências extraídas no módulo anterior, quer de ingredientes, quer de acções, dando um total de 804 respostas, tendo que ser verificado na base de dados quantas entradas foram correcta e incorrectamente colocadas.

A tabela 7.3 contém os resultados obtidos para esta avaliação. Os resultados vão quase corresponder à média ponderada dos ingredientes e acções. Contudo, são inferiores devido a algumas acções e a alguns alimentos que não são inseridos por terem um lema não esperado, e que, portanto, não estão presentes na tabela dos alimentos, que tem os seus dados carregados antes do processamento de qualquer receita.

1ª Avaliação	
Data	Setembro 2007
Respostas Correctas	671
Respostas Dadas	798
Total de Respostas Certas	804
Precisão	84%
Cobertura	84%

Tabela 7.3: Avaliação da introdução na base de dados com base nas dependências extraídas

Caso se considere o sistema na sua totalidade e se se comparar as entradas da base de dados com cada uma das receitas iniciais, o número de respostas entre ingredientes e acções sobe para 863, provocando assim uma diminuição da cobertura para 78%, e evidenciando a acumulação de erros ao longo do sistema. Estes valores estão presentes na tabela 7.4.

1ª Avaliação	
Data	Setembro 2007
Respostas Correctas	671
Respostas Dadas	798
Total de Respostas Certas	863
Precisão	84%
Cobertura	78%

Tabela 7.4: Avaliação da introdução na base de dados com base na receita inicial

7.4 Avaliação Final

Na avaliação final são apenas consideradas as oito receitas restantes do corpus (duas de cada tipo: sopa, carne, peixe e sobremesa) que não foram usadas para desenvolver o sistema, ou seja, que não se usaram para a construção de regras, logo cujo conteúdo era desconhecido.

Os resultados obtidos nestas oito receitas podem ser vistos na figura 7.5. Como seria de esperar, os resultados estão abaixo dos obtidos com o corpus de treino, em especial o valor da cobertura, facto que se deve a situações não consideradas que não existiam no corpus de treino. Os resultados obtidos para a introdução na base de dados têm como base o número de dependências extraídas, incluíndo os ingredientes e as acções das receitas. Na tabela 7.6 são mostrados os valores finais para a situação em que é considerado como total todos os ingredientes e acções presentes nas receitas, implicando uma diminuição no valor da cobertura.

Avaliação Final			
	Ingredientes	Acções	Base de Dados
Respostas Correctas	64	87	139
Respostas Dadas	69	110	173
Total de Respostas Certas	72	135	179
Precisão	93%	79%	80%
Cobertura	88%	64%	78%

Tabela 7.5: Avaliação do corpus de avaliação com base nas dependências extraídas

Avaliação Final	
	Base de Dados
Respostas Correctas	139
Respostas Dadas	173
Total de Respostas Certas	207
Precisão	80%
Cobertura	67%

Tabela 7.6: Avaliação do corpus de avaliação com base na receita inicial

8 Conclusões e Trabalho Futuro

Esta tese apresenta um sistema que permite extrair a informação disponível em receitas de culinária, ou seja, a informação referente aos ingredientes e às acções pertencentes a cada receita, e colocar essa informação numa base de dados apropriada ao domínio.

Com este sistema é assim possível ter um repositório de informação sobre receitas, onde se podem realizar pesquisas das receitas existentes, obter os ingredientes que as compõem e as acções necessárias para a sua preparação, bem como a possibilidade de integração num sistema de diálogo.

Como em qualquer trabalho na área da língua natural, este trabalho não está completo, devido à evolução que a língua vai tendo, e também novos alimentos, novos utensílios e novas técnicas de culinária (novas acções) que podem surgir, o que implica que haja uma actualização dos conceitos presentes na ontologia, tal como dos léxicos do sistema e a forma como são tratadas as acções.

Desta forma, há vários aspectos do sistema que poderão ser futuramente melhorados e que se enumeram de seguida:

- Opcionalidade de ingredientes

  Ocorre quando existem dois ingredientes numa frase e estão ligados por uma coordenação "ou", indicando que pode ser usado um ou outro ingrediente, como em "30 g de manteiga ou margarina". Actualmente são extraídas duas dependências (INGR(30, g, manteiga) e INGR(margarina)), não havendo nenhuma referência à opcionalidade entre os ingredientes. Note-se que a informação extraída na segunda dependência está incompleta. Uma forma de resolver este problema passa por indicar que o alimento é opcional, usando mais um argumento na dependência, indicando precisamente esta opcionalidade.

- Opcionalidade das quantidades

  Esta situação refere-se a frases que tenham como opção a quantidade do alimento, como em "3 a/ou 4 colheres de sopa de água", indicando assim um intervalo para a quantidade do ingrediente. Com as regras apresentadas anteriormente, apenas o segundo valor fica representado na dependência (INGR(4, colheres de sopa, água)) e, tal como no caso anterior, não há referências à opcionalidade. Esta situação pode ser resolvida de uma forma semelhante à anterior, adicionando o argumento "ou" à dependência e criando outra para ter informação da opção (INGR(3, colheres

de sopa, água, ou) e INGR(4, colheres de sopa, água, ou)). Este problema é responsável pela não extracção de 1 ingrediente (em 8) na avaliação final.

- Ingredientes com partes de alimento sem referência a alimento

  As regras apresentadas na secção 5.2.2 têm em conta a utilização de partes de alimento apenas quando estão ligadas a um alimento. Mas há ingredientes que apenas têm a parte de alimento, como "5 gemas" ou "200 g de lombo". Ou seja, falta, no primeiro caso, dizer que é referente ao ovo, e no segundo caso, qual o animal do qual foi retirado o lombo. Na avaliação final são 2 os ingredientes não extraídos devido a este problema.

- Ambiguidade dos textos em língua natural

  A língua Portuguesa é uma língua muito rica em ambiguidades e o domínio da culinária não foge à regra. Frases como "pão e queijo ralado" e "1 molho de agriões" demonstram ambiguidades sintáctica e semântica, respectivamente, na medida em que, no primeiro caso, não se consegue determinar se o pão é ou não ralado, e no segundo caso, o facto da palavra "molho" ter dois sentidos (alimento e uma quantidade). Isto faz com que a dependência INGR(1, molho) seja reconhecida incorrectamente. Este problema provoca na avaliação final a não extracção de 2 ingredientes.

- Modificação de palavras pertencentes ao léxico

  Algumas frases têm o alimento escrito numa forma diferente da que está presente no léxico, sendo mais comum o uso de diminutivos, como em "4 pãezinhos redondos" e "100 g de batatinhas". Como estas palavras não têm o traço "alimento" (por não estarem presentes no léxico), não vão ser identificados como ingredientes. Uma possível solução seria adicionar estes diminutivos ao léxico, e no módulo de pós-processamento realizar uma substituição para a palavra que devia ser correcta, ou seja, caso se tivesse INGR(4, pãezinhos), podia-se passar para INGR(4, pão) e ESTADO(pequeno, pão).

- Enriquecimento do léxico

  Embora o léxico de alimentos e utensílios já seja bastante rico, há sempre um ou outro termo que pode faltar, variedades de alimentos (por exemplo, fruta) que não foram adicionadas ao léxico ou alimentos usados em receitas de outros países que não estão considerados no léxico. Desta forma a quantidade de palavras pertencentes a este domínio pode ser maior e assim melhorar o desempenho do sistema relativamente à detecção de ingredientes e de alimentos associados às acções.

- Adição de mais conhecimento da ontologia na base de dados

  Neste momento apenas os alimentos, utensílios, estados, acções e algumas unidades de medida são guardados na base de dados, ao qual se podiam juntar as categorias dos alimentos e as equivalências entre medidas de tal forma que fosse possível inquirir a base de dados acerca destes

dados (por exemplo, saber quais são as categorias a que pertece o bacalhau). Adicionalmente pode-se extrair informação das receitas relativamente à classificação das receitas, caso as receitas as tenham disponíveis, tendo assim mais informação referente a cada receita. De notar que as tabelas da base de dados correspondentes a estes conceitos já estão criadas, no entanto, não estão a ser usadas actualmente pelo sistema.

- Adição de mais regras para as acções

  As regras da secção 5.3.2 que permitem extrair as acções das receitas foram definidas tendo em conta o corpus de treino, que não considera todas as situações em que as acções são usadas. Deste modo, a construção de mais regras permite a detecção de mais situações em que essas acções estão presentes, reflectindo-se numa melhoria do desempenho do sistema.

- Ordenação das acções

  Actualmente, o sistema não realiza uma ordenação das acções presentes na mesma frase pois as acções surgem ordenadas pela ordem em que as regras foram declaradas no módulo de processamento de receitas, não respeitando a sua posição na frase. Este erro é criado no segundo módulo. Como tal, a ordenação presente na base de dados vai reflectir também a ordenação errada. Uma forma de corrigir este erro é, ao realizar a análise do XML de saída, verificar a posição em que o verbo correspondente à acção aparece na frase (informação que é dada pelo XIP) e comparar com as restantes acções da mesma frase. No entanto isto implicaria que se passasse o verbo da acção como argumento nas dependências, o que não acontece neste momento, de forma a que, quando se faz a análise das dependências no terceiro módulo, se tenha algum argumento que tenha uma referência ao verbo (logo, à posição que tem na frase) e que permita a comparação entre o mesmo argumento de outras acções.

- Detecção do tipo de receita

  Este era um dos objectivos propostos inicialmente, mas que não foi resolvido devido a ter-se dado mais ênfase à extracção dos ingredientes e das tarefas, sendo que não acabou por restar tempo para realizar este item. São vários os tipos de receitas existentes, passando pelos pratos de carne, peixe, sobremesa, entre outros. Uma forma de determinar o tipo de uma receita seria analisar o título da mesma e verificar que tipo de alimento estava presente. Por exemplo, a receita "Bacalhau à Brás" é uma receita de peixe porque contém no seu nome um alimento que é um peixe. Esta opção implicaria colocar no léxico do XIP a informação das categorias de cada alimento, de forma a determinar que, neste caso, bacalhau é um peixe. Pode-se usar também algumas regras para outro tipo de receitas, as sopas, pois a maioria do nome das receitas deste tipo contêm "Sopa de <alimento>".

Os resultados da avaliação ao sistema são bastante satisfatórios, em especial para a extracção dos

ingredientes, no entanto, a parte de extracção das acções pode ser melhorada de forma a considerar mais hipóteses, e assim obter melhores resultados. Relativamente à avaliação da introdução na base de dados, caso se considerasse apenas o número de ingredientes e acções extraídos no módulo de processamento das receitas, os valores de precisão e cobertura seriam perto dos 100% em ambos os casos, visto que são poucas as situações que não são introduzidas na base de dados. Este valor deve-se ao facto do módulo de transformação de saída não reconhecer alguns alimentos, provocando assim a não introdução dos ingredientes na base de dados e também levando a interpretações erradas do tipo de acção (pois para distiguir o tipo de acção é necessário saber se os argumentos da acção são alimentos ou utensílios).

Bibliography

Agency for Toxic Substances and Disease Registry. (2006). (`http://www.atsdr.cdc.gov/`)

Aït-Mokhtar, S., Chanod, J.-P., & Roux, C. (2002). Robustness Beyond Shallowness: Incremental Dependency Parsing. *Natural Language Engineering*, *8*(3), 121–144.

Blaschke, C., Hirschman, L., & Valencia, A. (2002, Maio). Information Extraction in Molecular Biology. *Briefings in Bioinformatics*, *3*(2).

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. Cambridge: MIT Press.

Collier, N., Nobata, C., & Tsujii, J. (2000). Extracting The Names of Genes and Gene Products with a Hidden Markov Model. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING'2000), Saarbrucken, Germany.*

FlyBase Database. (2006). (`http://flybase.bio.indiana.edu/`)

Fukuda, K., Tsunoda, T., Tamura, A., & Takagi, T. (1998, Janeiro). Toward Information Extraction: Identifying Protein Names from Biological Papers. In *Proceedings of the Pacific Symposium on Biocomputing*.

Gaizauskas, R., & Wilks, Y. (1998). Information Extraction: Beyond Document Retrieval. *Journal of Documentation*, *54*(1), 70–105.

Gastronomia. (2006). (`http://www.gastronomia.madinfo.pt`)

Hagège, C., & Brun, C. (2003). Normalization and Paraphrasing using Symbolic Methods. In *Proceedings of the Second International Workshop on Paraphrasing* (pp. 41–48).

Harris, Z. (1952). Discourse Analysis: A Sample Text. *Language*, *28*(4), 474-494.

Hobbs, J. R. (2002). Information Extraction From Biomedical Text. *Journal of Biomedical Informatics*, *35*(4), 260–264.

Kitchenet. (2006). (`http://kitchenet.aeiou.pt/`)

Lindvall, E., & Nilsson, J. (2002). *Extracting Information from Sport Articles in Swedish Using Pattern Recognition.*

Llobera, E., Dalton, C., & Angulo, J. (2003). *Football Information Extraction System.*

LocusLink: Gene Database. (2006). (`http://www.ncbi.nih.gov/LocusLink`)

Maria de Lourdes Modesto. (1981). *Doze Meses de Cozinha*. Selecções do Reader's Digest.

Maria de Lourdes Modesto. (1982). *Cozinha Tradicional Portuguesa*. Editorial Verbo.

Medeiros, José C. (1995). *Processamento Morfológico e Correcção Ortográfica de Português.*

MedLine. (2006). (`http://www.ncbi.nlm.nih.gov/PubMed/`)

Mukherjea, S., Subramaniam, L. V., Chanda, G., Sankararaman, S., Kothari, R., Batra, V., et al. (2004, September/November). Enhancing a Biomedical Information Extraction System With Dictionary Mining and Context Disambiguation. *IBM Journal of Research and Development*, *48*(5/6), 693–702.

Pardal, J. P., & Mamede, N. J. (2004, Novembro). Terms Spotting with Linguistics and Statistics. In *Proceedings of the Herramientas y Recursos Lunguísticos para el Español y el Português Workshop* (pp. 298–304).

Proux, D., Rechenmann, F., & Julliard, L. (1998). Detecting Gene Symbols and Names in Biological Texts: A First Step Toward Pertinent Information Extraction. *Genome Informatics Workshop*, *9*, 72–80.

Receitas e Menus. (2006). (`http://www.receitasemenus.net`)

Ribeiro, R., Batista, F., Pardal, J. P., Mamede, N. J., & Pinto, H. S. (2006, September). Cooking an Ontology. In *The Twelfth International Conference on Artificial Intelligence: Methodology, Systems, Applications* (Vol. 4183, pp. 213–221).

Ribeiro, R., Oliveira, L., & Trancoso, I. (2002). Morphossyntactic Disambiguation for TTS Systems. In *Proc. of the 3rd Intl. Conf. on Language Resources and Evaluation* (Vol. V, p. 1427-1431).

Rindflesch, T. C., Tanabe, L., Weinstein, J. N., & Hunter, L. (2000). EDGAR: Extraction of Drugs, Genes and Relations From The Biomedical Literature. In *Proceedings of the Pacific Symposium on Biocomputing* (p. 514-525).

Roteiro Gastronómico de Portugal. (2006). (`http://www.gastronomias.com`)

The FlyBase Consortium. (1997). FlyBase: a Drosophila database. *Nucleic Acids Research*, *25*(1), 63-66.

Unified Medical Language System. (2006). (`http://umlsks.nlm.nih.gov/`)

WikiLivros. (2006). (`http://pt.wikibooks.org/wiki/Livro_de_Receitas`)

Xerox Research Centre Europe. (2003). *XIP User Guide.*

Ying, J. (2006, September). *Analysis and Comparison of Existent Information Extraction Methods.*

Author Index

I Anexos

A Lista de Acções

A.1 Acção Composta

- **Arranjar**: Amanhar o peixe ou arranjar pimentos.
- **Arrefecer**: Mergulhar um alimento em água gelada depois de escaldado para interromper o cozimento e manter a cor.
- **Assar**: Submeter um alimento à acção directa do fogo em seco.
- **Branquear**: Passar os alimentos em água a ferver para lhes tirar o ácido, o excesso de sal, a pele, para os tornar mais tenros, ou para os pré-cozer.
- **Ciclo**: Sequências de Acções que são repetidas até que uma condição de paragem se verifique, por exemplo, o alimento fica cozinhado.
- **Cozer**: Preparar alimentos submetendo-o à acção de um líquido não gordo.
- **Cozer em Líquido**: Cozer alimentos submergindo-os num líquido em ebulição.
- **Cozer a Vapor**: Cozer alimentos num recipiente furado, por cima de outro com água a ferver de maneira a que a água não toque nunca os alimentos e este cozinhem somente com a acção do vapor.
- **Cozinhar em banho-Maria**: Encher um recipiente com um terço de água e levar ao lume. Introduzir o alimento num segundo recipiente e colocar dentro do primeiro recipiente para que possa cozinhar.
- **Escaldar**: Mergulhar legumes ou frutas em água a ferver para interromper o cozimento, soltar as cascas, fixar a cor, tirar o gosto amargo e/ou reduzir a quantidade de sal das carnes curadas.
- **Estufar**: Cozinhar, em fogo lento, alimentos num recipiente fechado (criando um estufa).
- **Fazer**: Produzir determinado alimento.
  - **Fazer Caldo**: Produzir um caldo.
  - **Fazer Gelatina**: Produzir uma gelatina.
- **Flambear**: Regar um alimento com aguardente, rum ou licor, incendiá-lo com um fósforo e servir o alimento depois de apagada a chama.
- **Fritar**: Cozinhar os alimentos em gordura quente. Em inglês, "deep-fried" significa que o alimento é submerso em bastante óleo. "Sautée" e "pan-fried" referem-se a alimentos que são fritos em

pouco óleo, o suficiente para untar a frigideira e evitar que se peguem. "Stir-fried" é o acto de fritar pequenos pedaços de alimento em lume muito alto, tradicionalmente numa wok.

- **Gratinar**: Polvilhar com queijo ou pão ralado e levar ao forno a alourar alimentos pré-cozinhados e cobertos com um molho.
- **Grelhar**: Preparar os alimentos em chapa de metal colocada sobre brasas ou directamente no fogão.
- **Macerar**: Marinar um alimento em açúcar, álcool, licores ou numa mistura de óleo, vinho, ervas aromáticas e especiarias.
- **Marinar**: Introduzir o alimento num líquido para que fiquem mais tenros e saborosos.
- **Temperar**: Adicionar vários alimentos para realizar temperos.

A.2 Acção Primitiva

A.2.1 Acção Primitiva Alimentos

- **Albardar**: Envolver uma carne, ave ou caça numa delgada fatia de toucinho, para evitar que, ao cozinhar, seque ou pegue ao tacho.
- **Alourar**: Dar uma cor dourada a a qualquer alimento.
- **Amaciar**: Quebrar as fibras duras da carne, seja batendo com um martelo de carne ou usando marinadas ácidas.
  - **Amaciar Batendo**.
  - **Amaciar Marinada**.
- **Atar**: Passa-se um fio nas asas, unindo-as bem ao corpo, e faz-se o mesmo às coxas. Este dá às carnes e às aves, antes de se cozinharem, uma forma mais regular.
- **Caramelizar**: Processo de cozinhar o açúcar até ficar liquefeito e se transformar em calda, cuja cor vai do dourado ao castanho-escuro. O açúcar também pode ser caramelizado se for polvilhado sobre o alimento para derreter no forno como no crème brulée. Este termo também se aplica a cebolas e alho-porro quando sautée em gordura.
- **Coser**: Coser um alimento com agulha e linha.
- **Demolhar**: Introduzir um alimento em água fria, para lhe retirar determinadas impurezas ou sal.
- **Embeber**: Ensopar o bolo com calda de açúcar temperada ou licor; geralmente aplicada com pincel.
- **Engrossar**: Deitar qualquer porção de farinha num caldo ou molho para o tornar mais grosso.
- **Enrolar**: Dobrar um alimento sobre si mesmo de maneira tomar a forma de um rolo, por exemplo, tortas.

- **Envolver**: Revestir externamente um alimento com outro alimento.
- **Espremer**: Extrair líquido de alimentos, nomeadamente, citrinos.
- **Ferver**: Pôr um alimento líquido ao lume até que este entre em ebulição.
- **Glasear**: Cobrir os alimentos com um preparado como o açúcar, uma calda, ou um aspic para lhe dar brilho.
- **Impregnar**: Mergulhar o alimento seco em líquido quente para re-hidratar o alimento e/ou fazer o alimento ganhar o sabor do líquido.
- **Incendiar**: Acender um fósforo e deitar fogo a uma alimento. Utiliza-se para flambear alimentos ou chamuscar aves.
- **Juntar**: Adicionar um alimento a outro.
  - **Juntar até Meio**: Adicionar um alimento até meio do outro alimento.
  - **Juntar em Cima**: Adicionar um alimento em cima de outro.
  - **Juntar ao Lado**: Adicionar um alimento ao lado de outro.
  - **Juntar à volta**: Adicionar um alimento à volta de outro.
- **Lardear**: Inserir tiras de gordura (geralmente toucinho) em peças de carne magra, para deixar o prato mais suculento e saboroso.
- **Lavar**: Lavar em água corrente um ingrediente, normalmente fruta e legumes.
- **Manter**: Conservar um alimento.
  - **Manter Frio**: Conservar um alimento frio.
  - **Manter Quente**: Conservar um alimento quente.
- **Partir Ovo**: Partir a casca do ovo.
- **Pelar**: Tirar a pele (normalmente tomate ou pimentos) recorrendo a água quente.
- **Polvilhar**: Cobrir com um alimento ou utensílio com um alimento em pó ou ralado, por exemplo, farinha ou queijo ralado.
- **Rechear**: Introduzir um alimento dentro de outro.
- **Reduzir**: Diminuir qualquer molho, caldo ou calda deixando-o ferver.
- **Refogar**: Cozinhar os alimentos na mistura de temperos onde estão a cozinhar deixando-os depois em pequena quantidade de líquido temperado, num tacho bem tapado em lume brando, por longo tempo.
- **Tirar**: Remover algo a um alimento.
  - **Tirar Espinhas**: Remover as espinhas do alimento.
  - **Tirar Ossos**: Remover os ossos do alimento.
  - **Tirar Sementes**: Remover as sementes ou caroços do alimento.
- **Verificar**: Certificar que o alimento está a gosto.

A.2.2 Acção Primitiva Alimentos e Utensílio

- **Adicionar**: Colocar um alimento dentro de um recipiente.
- **Amassar**: Técnica de apertar e dobrar a massa para a deixar mais firme e macia. Amassar estica o glúten na farinha, dando mais elasticidade.
- **Aquecer**: Subir a temperatura a um alimento, fazendo uso de uma fonte de calor até que haja uma transformação do alimento.
- **Bater**: Incorporar ar a ingredientes, como as natas ou ovos, usando uma batedeira de metal.
- **Clarificar**: Retirar a gordura dos caldos e dos consomés. Também significa retirar as impurezas dos mesmos.
- **Cortar**: Cortar um alimento.
  - **Abrir ao Meio**: Dar um corte central no alimento e separar as duas metades.
  - **Aparar**: Cortar um alimento só nas pontas
  - **Cortar Bocados**: Cortar com uma faca em pedaços disformes.
  - **Cortar Borboleta**: Abrir um alimento (perna de carneiro, peito de frango, camarões) ao meio sem o separar. As metades ficam abertas, obtendo-se a aparência de uma borboleta.
  - **Cortar Cubos**: Cortar os alimento em cubinhos iguais, do mesmo tamanho.
  - **Cortar Rodelas**: Cortar um ingrediente às rodelas.
  - **Cortar Palitos**: Cortar um ingrediente aos palitos.
  - **Desfazer**: Separar o alimento, esmagando-o, fazendo uso de um garfo.
  - **Desfiar**: Separar o alimento cortando ou desfiando em tamanhos finos, usando faca, cutelo ou ralador. Pode-se usar também a picadora com o disco apropriado. A galinha cozida e o pato assado oriental são desfiados com dois garfos.
  - **Lascar**: Cortar em lascas, normalmente usando as mãos.
  - **Ralar**: Cortar com o ralador.
  - **Trinchar**: Cortar o alimento (normalmente uma ave) já cozinhado, para ser servido.
- **Descascar**: Tirar a pele ou casca a uma fruta ou legume.
  - **Descascar Descascador**: Tirar a pele ou casca a uma fruta ou legume usando um descascador mecânico.
  - **Descascar Descascador Legumes**: Tirar a pele ou casca a uma fruta ou legume usando um utensílio próprio.
  - **Descascar Faca**: Tirar a pele ou casca a uma fruta ou legume com uma faca.
  - **Descascar Ovo**: Tirar a casca ao ovo.
- **Enxugar**: Tirar o excesso de líquido a um alimento.

- **Furar**: Espetar os alimentos (a casca de frutas e legumes) para deixar que saia o ar ou a água durante o cozimento, Fura-se a pele do pato antes do cozimento para que escorra o excesso de gordura.
- **Guardar**: Colocar alimentos em recipientes de armazenamento.
- **Misturar**: Usa-se colher, batedeira ou liquidificadora para misturar homogeneamente dois ou mais alimentos. Ou combinar uma mistura mais leve com outra mais pesada. A mais leve é colocada sob a mais pesada, e em seguida usa-se uma colher ou espátula para mexer num delicado movimento em forma de oito, para misturar os ingredientes sem tirar o ar.
- **Moer**: Reduzir alimentos, em geral carne, a pedaços muito pequenos. Existem máquinas e facas próprias para isso. Ou reduzir o alimento a pó ou a pedacinhos usando pilão ou picadora. Há moedores especiais para temperos ou grãos de café.
- **Moldar**: Dar forma a um alimento recorrendo a um utensílio.
- **Peneirar**: Passar os ingredientes secos pela peneira para que as partículas maiores fiquem retidas e separadas do pó fino. Muito usado em massas assadas para arejar os ingredientes.
- **Pincelar**: Envolver um alimento com outro, no estado líquido, fazendo uso de um pincel ou trincha.
- **Raspar**: Desbastar a superfície de um alimento com um raspador.
- **Regar**: Derramar colheradas ou pinceladas do caldo ou gordura do recipiente sobre os alimentos durante o cozimento; dá sabor e humidade.
- **Retirar**: Esvaziar um recipiente dos alimentos que continha.
- **Revolver**: Mexer, com a ajuda de uma colher ou vara, alimentos contidos num recipiente.
- **Riscar**: Fazer incisões nas cascas, na polpa ou na gordura dos alimentos, como a carne, peixe ou legumes, antes de cozinhar.
- **Separa Ovo**: Separa a gema da clara, sendo cada uma das partes colocada num recipiente.
- **Untar**: Revestir uma forma com manteiga ou óleo e/ou farinha ou papel-manteiga para evitar que o conteúdo se pegue. Pode também forrar-se, com fatias de Bacon, folhas de espinafres e biscoito champanhe.
- **Virar**: Virar um alimento para que seja cozinhado dos dois lados sem queimar.

A.2.3 Acção Primitiva Utensílios

- **Abrir**: O contrário de tapar.
- **Agitar**: Abanar um utensílio.
- **Colocar Cima**: Colocar o "Utensílio1" em cima do "Utensílio2".
- **Colocar Dentro**: Colocar o "Utensílio1" dentro do "Utensílio2".

- **Colocar Pás**: Colocar pás misturadoras de líquidos ou massas na batedeira.
- **Desligar**: Desligar um electrodoméstico (micro-ondas) ou utensílio gerador de fonte de calor (fogão, forno, lamparina).
- **Ligar**: Ligar um electrodoméstico (micro-ondas) ou utensílio gerador de fonte de calor (fogão, forno, lamparina).
- **Retirar Cima**: Retirar um utensílio de cima de outro.
- **Retirar Dentro**: Retirar um utensílio de dentro de outro.
- **Tapar**: Tapar (normalmente uma tampa) uma panela ou tacho.

A.2.4 Acção Primitiva Espera

- **Espera temperatura alimento**: Espera que um alimento atinja uma determinada temperatura.
- **Espera temperatura utensílio**: Espera que um utensílio (normalmente o forno ou frigideira) atinja uma determinada temperatura.
- **Espera tempo**: Espera que decorra um instante de tempo.
- **Espera transformação**: Espera que um ingrediente se transforme e atinja um determinado estado de confecção.

B Associação Entre Acções e Verbos ou Expressões

Acção	**Lista de Verbos**
Acção Composta	
Arranjar	Arranjar, Amanhar
Arrefecer	Arrefecer
Assar	Levar a assar, Levar ao forno
Ciclo	
Cozer em Líquido	Levar ao lume, Cozer, Cozinhar
Cozer a Vapor	Cozer a vapor
Cozinhar em Banho-Maria	Levar ao lume em Banho-Maria
Branquear	Branquear
Fazer Caldo	Fazer Caldo
Fazer Gelatina	Fazer Gelatina
Macerar	Macerar
Marinar	Marinar
Flambear	Flambear
Fritar	Fritar
Escaldar	Escaldar
Estufar	Levar a estufar
Grelhar	Grelhar
Gratinar	Levar a gratinar
Temperar	Temperar

Tabela B.1: Verbos associados às acções compostas

Acção	**Lista de Verbos**
Acção Primitiva de Alimentos e Utensílios	
Abrir ao Meio	Abrir ao meio
Adicionar	Dispor, Colocar, Adicionar, Deitar, Introduzir, Despejar, Pôr, Juntar
Amaciar Batendo	Amaciar batendo
Amassar	Amassar
Aparar	Aparar
Aquecer	Aquecer
Bater	Bater
Clarificar	Clarificar, Retirar gordura
Cortar às Rodelas	Cortar às rodelas
Cortar em Cubos	Cortar em cubos
Cortar em Bocados	Cortar em bocados/pedaços, Partir, Dividir
Cortar em Borboleta	Cortar em borboleta
Cortar em Palitos	Cortar em palitos
Descascar com Faca	Descascar
Descascar com Descascador	Descascar com Descascador
Descascar com Descascador de Legumes	Descascar com Descascador de Legumes
Descascar Ovo	Descascar o ovo
Desfazer	Desfazer, Esmagar
Desfiar	Desfiar
Enxugar	Enxugar, Escorrer
Espremer	Espremer, Extrair
Furar	Furar
Guardar	Levar, Guardar, Pôr, Ir, Reservar
Lascar	Lascar
Misturar	Misturar, Dissolver, Incorporar
Moer	Passar, Picar, Triturar, Esmagar
Moldar	Moldar
Peneirar	Passar por peneira, Peneirar
Pincelar	Pincelar
Ralar	Ralar
Raspar	Raspar
Regar	Regar
Retirar	Retirar
Revolver	Mexer, Revolver
Riscar	Riscar
Separar Ovo	Separar o ovo
Trinchar	Trinchar
Untar	Untar
Virar	Virar

Tabela B.2: Verbos associados às acções primitivas de alimentos e utensílios

Acção	**Lista de Verbos**
Acção Primitiva de Alimentos	
Albardar	Albardar
Alourar	Alourar, Aloirar, Dourar
Amaciar Batendo	Amaciar batendo
Amaciar Marinada	Amaciar com uma marinada
Atar	Atar
Caramelizar	Caramelizar
Demolhar	Demolhar, Pôr de molho
Embeber	Embeber, Ensopar
Engrossar	Engrossar
Enrolar	Enrolar
Envolver	Envolver, Passar por, Esfregar
Ferver	Ferver
Glasear	Glasear
Impregnar	Mergulhar, Impregnar
Incendiar	Incendiar
Juntar	Juntar, Introduzir, Acrescentar, Adicionar, Deitar, Colocar
Juntar em Cima	Juntar, Introduzir, Acrescentar, Adicionar, Deitar, Colocar
Juntar à Volta	Juntar, Introduzir, Acrescentar, Adicionar, Deitar, Colocar
Juntar até Meio	Juntar, Introduzir, Acrescentar, Adicionar, Deitar, Colocar
Lardear	Lardear
Lavar	Lavar, Limpar
Manter Quente	Conservar, Manter
Manter Frio	Conservar, Manter
Partir Ovo	Partir o ovo
Pelar	Tirar pele, Pelar
Polvilhar	Polvilhar
Rechear	Rechear
Reduzir	Derreter
Refogar	Refogar, Preparar Refogado
Tirar Ossos	Tirar Ossos, Desossar
Tirar Espinhas	Tirar Espinhas
Tirar Sementes	Tirar sementes
Verificar	Verificar, Rectificar

Tabela B.3: Verbos associados às acções primitivas de alimentos

Acção	**Lista de Verbos**
Acção Primitiva de Utensílios	
Abrir	Abrir, Destapar, Destampar
Agitar	Agitar
Colocar em Cima	Levar ao lume, Colocar em cima, Pôr em cima
Colocar Dentro	Levar ao forno, Ir ao forno, Colocar Dentro, Pôr Dentro
Colocar Pás	Colocar pás
Desligar	Desligar
Ligar	Ligar
Retirar Cima	Retirar, Afastar, Tirar
Retirar Dentro	Retirar, Tirar
Tapar	Fechar, Tapar, Tampar

Tabela B.4: Verbos associados às acções primitivas de utensílios

Acção	**Lista de Expressões**
Acção Primitiva de Espera	
Espera Temperatura Alimento	Esperar que <alimento> esteja a x^{o}C / atinja x^{o}C
Espera Temperatura Utensílio	Esperar que <utensílio> esteja a x^{o}C / atinja x^{o}C
Espera Tempo	Passados x minutos, De véspera, Durante x minutos
Espera Transformação	Espere que <alimento> fique <estado>

Tabela B.5: Expressões associadas às acções primitivas de espera