INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Reconhecimento de Entidades Mencionadas (Obra, Valor, Relações de Parentesco e Tempo) e Normalização de Expressões Temporais

João Miguel Sanches Loureiro

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

Juri

Presidente:	Doutor Ernesto José Marques Morgado
Orientador:	Doutor Nuno João Neves Mamede
Co-orientador:	Doutora Maria Luísa Torres Ribeiro Marques da Silva Coheur
Vogal:	Doutora Irene Pimenta Rodrigues

Novembro de 2007

Agradecimentos

Gostaria de agradecer a dedicação, incentivo e colaboração dos meus orientador – Professor Nuno Mamede e co-orientadora – Professora Luísa Coheur.

A disponibilidade e contribuição de Jorge Baptista, Cristina Mota, Caroline Hagège e Xavier Tannier foram também factores importantes no desenvolvimento deste trabalho.

Por último, gostaria de agradecer aos colegas Ana Guimarães, Ana Mendes, Luís Romão e Telmo Machado pela sua paciência e colaboração, assim como a todos os outros elementos do L^2F que de uma forma de outra contribuíram para a realização deste trabalho.

Lisboa, 17 de Novembro de 2007
João Miguel Sanches Loureiro

Resumo

Este trabalho trata do Reconhecimento de Entidades Mencionadas para a língua portuguesa, relativamente às categorias Obra, Valor, Relações de Parentesco e Tempo. O Reconhecimento de Entidades Mencionadas é uma tarefa da área do processamento de língua natural que pode ser útil no desenvolvimento de sistemas de Pergunta-Resposta, Extracção de Informação e de Resumo (ou Sumarização), disponibilizando informação linguística e de forma estruturada. Este documento aborda também a normalização de expressões temporais, tais como *24 de Novembro de 2005* ou *próximo dia*. A normalização do tempo tem como objectivo a conversão dos valores referentes a expressões temporais, para um formato padrão, facilitando a partilha desta mesma informação entre diferentes sistemas. São assim apresentados neste documento os procedimentos adoptados, tanto para o Reconhecimento de Entidades Mencionadas, como para a normalização do tempo.

Abstract

This document stands for Named Entity Recognition under the categories Work of Art (*Obra*), Value (*Valor*), Relatives (*Relações de Parentesco*) and Time (*Tempo*), for portuguese language. Named Entity Recognition is a Natural Language processing task. It can help Question-Answering, Information Extraction and Sumarization systems development by providing useful and structured linguistic information. We also attempt to normalize time expressions such as *24 de Novembro de 2005* (*24th November, 2005*) and *próximo dia* (*next day*). Time normalization is about converting time expressions' values to a standard format allowing this information to be shared between different systems. This document presents the adopted proceedings for the recognition of named entities and for the normalization of time expressions.

Palavras Chave
Keywords

Palavras Chave

Processamento de Língua Natural

Reconhecimento de Entidades Mencionadas

Normalização de Expressões Temporais

Keywords

Natural Language Processing

Named Entity Recognition

Time Normalization

Índice

Lista de Figuras

Lista de Tabelas

1 Introdução

1.1 Motivação

O Reconhecimento de Entidades Mencionadas (do inglês: NER – *Named Entity Recognition*) é visto como uma sub-tarefa da extracção de informação que tem como objectivo localizar e classificar elementos atómicos num texto, de acordo com um conjunto predefinido de categorias. Desse conjunto de categorias fazem parte as categorias tempo (e.g., datas, durações), valores (e.g., valores monetários), entre outras.

O exemplo seguinte mostra a saída (segundo as regras de etiquetação do MUC (*MUC - Message Understanding Conferences*, n.d.)) resultante do processamento de uma frase, num sistema de Reconhecimento de Entidades Mencionadas:

O Pedro comprou 200 acções da PT em 2006.

```
O <ENAMEX TYPE="PERSON">Pedro</ENAMEX>comprou
<NUMEX TYPE="QUANTITY">200</NUMEX>acções da
<ENAMEX TYPE="ORGANIZATION">PT</ENAMEX>em
<TIMEX TYPE="DATE">2006</TIMEX>.
```

Note-se que as entidades encontram-se delimitadas por etiquetas que designam a sua categoria e tipo: <CATEGORIA TIPO="SUBTIPO">*entidade*</CATEGORIA>.

Este trabalho incide sobre o reconhecimento de entidades mencionadas referentes às categorias Obra, Valor, Relações de Parentesco e Tempo, sendo dado particular destaque a esta última.

As expressões de tempo indicam *quando* algo ocorreu (e.g., dia 2 de Maio de 1990), *quanto tempo* algo durou (e.g., 2 horas), ou *com que frequência* algo aconteceu (e.g., 2 vezes por semana).

O ser humano está sempre consciente da sua localização temporal, pelo que quando quer referir um determinado dia ou hora, usa expressões como *a dois de Janeiro*, *Quarta-Feira*, *na próxima semana*, ou *daqui a dois dias*. Para se interpretar de forma correcta e precisa este tipo de expressões é frequentemente necessário recorrer ao conhecimento do contexto temporal.

Pretende-se desenvolver um sistema capaz de efectuar a obtenção automática e de forma estruturada de informação sobre as entidades mencionadas presentes no texto, com os objectivos de participar no HAREM (Avaliação de Reconhecedores de Entidades Mencionadas) e auxiliar o desenvolvimento de um sistema de pergunta-resposta (Mendes, 2007).

Após efectuado o Reconhecimento de Entidades Mencionadas, o valor das entidades com referências temporais deve ser interpretado e normalizado de forma estruturada, facilitando a sua disponibilização para aplicações futuras.

1.2 Estratégia

Para o Reconhecimento de Entidades Mencionadas são usadas técnicas de processamento de língua natural, sendo que uma das principais ferramentas usadas é o XIP (Xerox, 2003), que é parte de uma cadeia de processamento disponível no L^2F.

O XIP tem a capacidade de suportar a execução de funções em código Python, o que permite efectuar a normalização de expressões temporais.

1.3 Roteiro

O capítulo 2 aborda o Tempo segundo várias perspectivas: envereda-se pelo domínio da lógica, passando depois pelos esquemas de anotação de expressões temporais e o modo como estes permitem situar de forma absoluta os eventos descritos no texto segundo um esquema temporal ou em relação a outros eventos.

Segue-se o capítulo 3 referente ao Reconhecimento de Entidades Mencionadas, onde são mencionadas as directivas, assim como a arquitectura utilizadas para esse fim. Ainda no mesmo capítulo é feita uma descrição do processo de reconhecimento das várias entidades propostas.

A arquitectura e os procedimentos usados, referentes à normalização de expressões temporais, são descritos no capítulo 4.

O capítulo 5 apresenta os resultados obtidos no Reconhecimento de Entidades Mencionadas, segundo os critérios de avaliação do HAREM.

Por fim, no capítulo 6 são descritos os problemas detectados durante o decorrer do trabalho e são propostas algumas soluções no sentido de melhor o desempenho do sistema (tanto a nível do reconhecimento de entidades como ao nível da normalização).

2 Estado da Arte

Este capítulo descreve vários métodos de representação de expressões temporais. Na secção 2.1 é feita uma abordagem às teorias de tempo no domínio da lógica, nomeadamente a *Tense Logic* e a *Interval-based Logic*. Segue-se uma descrição dos padrões TIMEX2 e TimeML definidos para anotação de expressões temporais, na secção 2.2.

2.1 Lógica Temporal

Como é referido na Stanford Encyclopedia of Philosophy (*Stanford Encyclopedia of Philosophy - Temporal Logic*, n.d.), o termo Lógica Temporal tem sido usado de uma forma geral, para designar qualquer tipo de representação de expressões temporais que tenha como princípio um ambiente baseado em lógica e, mais concretamente, para se referir à abordagem introduzida em 1960 por Arthur Prior sob o nome de *Tense Logic*, tendo sido posteriormente desenvolvida por lógicos e cientistas na área da informática. Uma abordagem alternativa é a *Interval-based Logic*. Estas duas lógicas distinguem-se essencialmente pela forma como consideram a decomposição do tempo em unidades.

As aplicações da Lógica Temporal incluem o uso da mesma como formalismo para esclarecer questões filosóficas relacionadas com tempo ou como um ambiente através do qual é definida uma semântica para as expressões temporais. Na área da Inteligência Artificial funciona como uma linguagem para codificar conhecimento acerca do tempo.

Nesta secção é feita uma descrição dos aspectos esseciais das lógicas mencionadas – *Tense Logic* e *Interval-based Logic* – comparando as vantagens e desvantagens de cada uma.

2.1.1 Tense Logic

De acordo com os autores Daniel Jurafsky e James H. Martin (Jurafsky & Martin, 2000), a *Tense Logic* interessa-se pela forma como os tempos verbais transmitem informação temporal. A teoria de tempo mais directa defende que o tempo flui inexoravelmente para a frente e que os eventos se encontram associados a pontos ou intervalos de tempo que, por sua vez, estão associados a uma linha de tempo. Dadas estas noções, pode-se então impor uma ordem a eventos distintos, colocando-os sobre uma linha de tempo. Mais especificamente, pode-se dizer que um evento precede um outro, se o tempo flui do

primeiro evento em direcção ao segundo. A acompanhar estas noções está ainda a ideia do momento presente. Combinando esta noção com a de uma relação de ordem temporal, surgem as noções familiares de passado, presente e futuro.

Como é de supor, existe um grande número de esquemas com o objectivo de representar este tipo de informação temporal. Considere-se o seguinte exemplo:

I arrived in New York.

I am arriving in New York.

I will arrive in New York.

Todas estas frases referem-se ao mesmo tipo de evento, sendo apenas distintas no tempo do verbo. Se as frases em questão forem representadas, recorrendo ao ambiente de FOPC (First Order Predicate Calculus), sem qualquer informação temporal, todas partilham a seguinte representação, em que w é a variável do evento:

$$\exists w\ ISA(w, Arriving) \wedge Arriver(w, Speaker) \wedge Destination(w, NewYork)$$

No entanto, a informação temporal que provém dos tempos verbais pode ser explorada, representado a informação adicional sobre a variável w do evento, sob a forma de predicados. Mais concretamente, podem ser adicionadas variáveis temporais que representam o intervalo correspondente ao evento, o ponto final (*endpoint*) do evento, e predicados temporais relacionando este ponto final com o tempo presente, como indicado pelo tempo do verbo. Esta abordagem leva à seguinte representação para os exemplos mencionados anteriormente, em que as variáveis i e e representam, respectivamente, o intervalo de tempo associado ao evento e o fim desse intervalo:

$$\exists i, e, w\ ISA(w, Arriving)$$
$$\wedge Arriver(w, Speaker) \wedge Destination(w, NewYork)$$
$$IntervalOf(w, i) \wedge EndPoint(i, e) \wedge Precedes(e, Now)$$

$$\exists i, e, w\ ISA(w, Arriving)$$
$$\wedge Arriver(w, Speaker) \wedge Destination(w, NewYork)$$
$$IntervalOf(w, i) \wedge MemberOf(i, Now)$$

$$\exists i, e, w\ ISA(w, Arriving)$$

$$\wedge Arriver(w, Speaker) \wedge Destination(w, NewYork)$$
$$IntervalOf(w, i) \wedge EndPoint(i, e) \wedge Precedes(Now, e)$$

O predicado *Preceeds* representa a noção de que o primeiro ponto de tempo (primeiro argumento) precede o segundo no tempo; a constante *Now* refere-se ao momento presente. Para eventos passados, o ponto final do intervalo precede o momento presente. De forma semelhante, para eventos futuros o momento presente tem que preceder o fim do evento. Para eventos presentes, o momento presente está contido no intervalo do evento.

No entanto, algumas complicações ocorrem ao serem considerados os exemplos que se seguem. Ambos referem eventos passados, mas não é correcto representá-los da mesma forma.

Flight 1902 arrived late.

Flight 1902 had arrived late.

Na frase *Flight 1902 had arrived late* parece haver um outro evento subjacente – o voo 1902 já tinha chegado atrasado *quando* uma outra coisa aconteceu. Tendo este problema em consideração, Reichenbach (Reichenbach, 1947) introduziu a noção de ponto de referência (*reference point*). Neste simples esquema temporal, o momento presente é equiparado ao tempo da elocução e é usado como ponto de referência para quando o evento ocorre (e.g., *before, at, after*). Na abordagem de Reichenbach, a noção de ponto de referência é diferente do tempo da elocução e do tempo do evento. Os próximos exemplos ilustram a ideia base desta abordagem:

When Mary's flight departed, I ate lunch.

When Mary's flight departed, I had eaten lunch.

Em ambos estes exemplos, o evento de comer aconteceu no passado, ou seja, antes da elocução. No entanto, o tempo verbal no primeiro exemplo, indica que o evento de comer começou na altura em que o voo teve início, enquanto que o segundo exemplo indica que o evento de comer terminou antes do início do voo. Portanto, nos termos de Reichenbach o evento da partida especifica o ponto de referência. Estes factos podem ser ajustados introduzindo restrições adicionais, relacionando os eventos de comer e da partida. Na frase *When Mary's flight departed, I ate lunch*, o ponto de referência precede o evento de comer, enquanto que no exemplo *When Mary's flight departed, I had eaten lunch*, a acção de comer precede o ponto de referência. A figura 2.1 ilustra a abordagem de Reichenbach para os tempos verbais Ingleses.

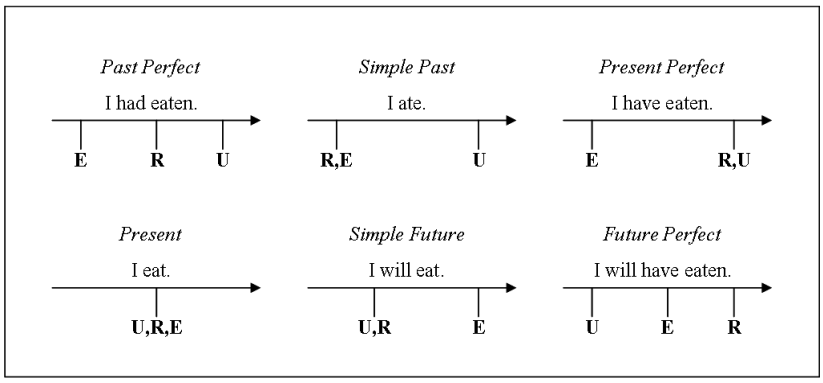


Figura 2.1: A abordagem de Reichenbach aplicada a vários tempos verbais Ingleses. Nestes diagramas o tempo flui da esquerda para a direita, o **E** denota o tempo do evento, o **R** denota o tempo de referência e o **U** denota o tempo da elocução.

2.1.2 Interval-based Logic

Segundo Endriss e Gabbay (Endriss & Gabbay, n.d.), em muitas aplicações, uma visão simplificada do tempo como uma sequência de pontos acaba por ser uma abstracção adequada da realidade. No entanto, na área do raciocínio em Inteligência Artificial, alguns autores têm dado preferência a sistemas baseados em intervalos em vez de pontos. Segundo os mesmos autores, os intervalos são um formalismo de representação temporal mais rico do que as lógicas temporais baseadas em pontos.

Dados dois pontos temporais distintos, *t1* e *t2*, o primeiro pode-se encontrar antes ou depois do segundo. Além disso, se se assumir o tempo como sendo discreto, os dois podem até encontrar-se, isto é, um deles pode estar exactamente antes do outro.

Em contraste, um intervalo pode não estar apenas antes de outro intervalo, como os dois podem também encontrar-se ou até sobrepor-se, ou o primeiro pode dar início ou finalizar o segundo, ou ter lugar dentro do outro. De facto, existem 13 relações diferentes entre intervalos (os seis anteriormente mencionados, os seus inversos e a equivalência). Estas relações são frequentemente conhecidas como as *relações de Allen*, devido ao trabalho de Allen (Allen, 1983) relativo aos intervalos de tempo. Um exemplo de uma lógica para intervalos é a lógica modal de intervalos proposta por Halpern e Shoham (Halpern & Shoham, 1991), que é uma lógica multi-modal equipada com operadores modais para cada uma das 13 relações de Allen. A figura 2.2 ilustra estas 13 relações segundo a representação proposta por Gibbs (Gibbs, 2004).

Ambas as lógicas, *point-base* e *interval-based*, permitem modelar situações onde um evento ocorre antes (e possivelmente imediatamente antes) de outro evento. No entanto, uma lógica onde as unidades de tempo primitivas são intervalos, consegue ir mais além do que uma lógica baseada em pontos temporais, nomeadamente nos seguintes dois casos cruciais:

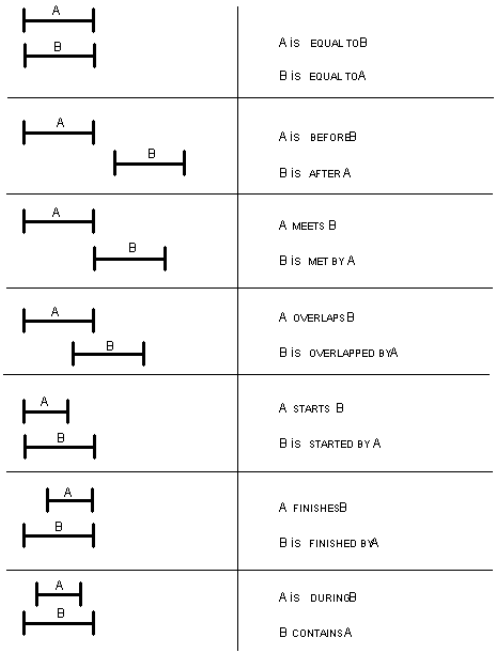


Figura 2.2: As 13 relações de Allen.

- consegue expressar que duas unidades de tempo se sobrepõem;
- consegue expressar que uma unidade de tempo pode ser decomposta em diversas outras unidades.

As várias relações entre intervalos tornam ainda possível a distinção, por exemplo, entre sub-intervalos que começam o intervalo dominante e sub-intervalos que se encontram estritamente no espaço de duração do intervalo dominante.

No entanto, a lógica de Halpern e Shoham (baseada em intervalos) é fortemente indecidível; portanto, não se pode esperar que uma lógica modal genérica com todas estas características seja computacionalmente tratável. Contudo, uma lógica de intervalos restrita pode ser de algum interesse. Em vez de se usar uma lógica que suporte todas as 13 relações de Allen, para certas aplicações pode ser suficiente considerar uma lógica que permita expressar pelo menos: (i) eventos passados e futuros (da mesma forma que a lógica baseada em pontos temporais permite) e (ii) eventos que ocorrem em sub-intervalos do intervalo de referência.

2.2 Esquemas de Anotação de Expressões Temporais

Na anotação de textos, interessam as expressões que denotam conteúdo temporal, isto é, expressões que referem tempo (e.g., *July 1, 1987*), durações (e.g., *three months*) ou frequências (e.g., *monthly*). Em

The Language of Time (Mani et al., n.d.), Pustejovsky refere que a identificação e distinção destes tipos de expressões é essencial, de modo a permitir situar de forma absoluta os eventos descritos no texto segundo um esquema temporal ou situá-los relativamente a outros eventos.

O resultado de um texto anotado pode ter diversas aplicações no desenvolvimento de um sistema computacional. Exemplos dessas aplicações são: o processamento sintáctico e semântico de texto; recuperação de informação; sistemas de pergunta-resposta; síntese de fala; tradução automática ou ainda geração de fala.

Esta secção descreve os aspectos essenciais de dois dos esquemas propostos para a etiquetação de expressões temporais: o TIMEX2 e o TimeML.

2.2.1 TIDES - Standard for the Annotation of Temporal Expressions

O TIDES (*Translingual Information Detection, Extraction and Summarization*) (*TIDES - DARPA Translingual Information Detection, Extraction, and Summarization*, n.d.) é um projecto tecnológico desenvolvido pela DARPA (*Defense Advanced Research Projects Agency*) (*DARPA - Defense Advanced Research Projects Agency*, n.d.). O seu principal objectivo é tornar possível que os falantes de inglês possam encontrar e interpretar informação necessária, independentemente da língua em que se encontra. De forma a possibilitar tais capacidades, o TIDES visa desenvolver um sistema tecnológico que integre várias potencialidades e que depois possa ser testado em problemas do mundo real, nomeadamente:

- Detecção - Procurar ou descobrir informação relevante;
- Extracção - Identificar os factos principais;
- Resumo - Reduzir o número de palavras que uma pessoa precisa ler;
- Tradução - Converter para inglês um texto numa outra língua.

No âmbito do projecto TIDES foi desenvolvido um conjunto de princípios (*guidelines*) para anotação de expressões temporais e normalização da informação temporal que estas expressões denotam. Não é suposto os princípios de etiquetação representarem todo o tipo de informação temporal proveniente da comunicação em língua natural (embora seja esse o objectivo a longo prazo). Estes princípios destinam-se a dois tipos de utilizadores:

1. Anotadores humanos que pretendem anotar expressões temporais com a finalidade de construir uma corpora, consistindo de dados temporais anotados, a ser usada pela comunidade de língua natural;

2. Programadores de sistemas que tentam construir programas de etiquetação para extracção de informação temporal de documentos.

O TIDES é, portanto, um padrão de anotação que tem como finalidade a identificação e normalização de expressões temporais de forma a poderem ser usadas em sistemas de pergunta-resposta (e.g., questões do tipo *quando*, ou questões com dependências temporais), tradução automática e resumo.

Este padrão de anotação especifica mais detalhes para a representação semântica do que as tarefas de reconhecimento do TIMEX usado em fóruns de avaliação anteriores (MUC7 1998). O conjunto de expressões suportadas pelo TIDES inclui todas as que eram suportadas pela etiqueta TIMEX das entidades nomeadas do MUC-7, com extensões que incluem, por exemplo, expressões ancoradas em eventos (e.g., *2 days before departure*), conjuntos de tempos (e.g., *every week*) e tempos não específicos (e.g., *[the traffic report comes on at] quarter past the hour* [1]). Assim, de acordo com o *Tern Evaluation Plan* (Ferro et al., 2004) e o documento de Robyn Kozierok (Ferro et al., n.d.), o TIDES estende a definição original da categoria TIMEX das entidades mencionadas do MUC, por fim a incluir uma maior variedade de expressões e a possibilitar a sua normalização e é também a base da etiqueta TIMEX3 do TimeML (Markup Language for Temporal and Event Expressions) (*TimeML - Markup Language for Temporal and Event Expressions*, n.d.), que é abordado na secção 2.2.2.

O esquema de anotação baseia-se numa única etiqueta XML (eXtensible Markup Language) (*Extensible Markup Language (XML)*, n.d.), `<TIMEX2>`, a ser aplicada a expressões temporais. Portanto, este padrão é frequentemente designado como o padrão TIMEX2 (*TIMEX2 standard*). A anotação de expressões temporais consiste em delimitar essas mesmas expressões com uma etiqueta especial. No início da expressão é colocada a etiqueta `<TIMEX2>` e no fim a etiqueta `</TIMEX2>`:

```
<TIMEX2>Halloween</TIMEX2>
```

As expressões temporais podem referir datas de calendário, partes do dia, ou durações (e.g., períodos de horas, dias, séculos). Basicamente, se uma frase ou palavra refere uma determinada área numa linha do tempo, então, esse significado deve ser capturado. É através de palavras-chave (*lexical triggers*) que são determinadas as expressões que devem ser anotadas.

Cada palavra-chave é uma palavra ou expressão numérica que transmite uma ideia de unidade temporal, ou um conceito como *day* ou *monthly*. São consideradas palavras-chave, aquelas que podem ser orientadas segundo uma linha do tempo, ou pelo menos em relação a um tempo (passado, presente,

[1] Ao longo deste documento, há por vezes a necessidade de referir uma determinada entidade e o(s) seu(s) respectivo(s) contexto(s). Para tal optou-se por delimitar o(s) contexto(s) com parenteses rectos, isolando assim a entidade: `[contexto]` *entidade* `[contexto]`.

futuro). Alguns exemplos de palavras-chave são: *minute, afternoon, Monday, New Year's Eve, 8:00, 1994, recent, ago*.

Segundo os princípios de etiquetação do TIDES 2005 (Ferro et al., 2005), as expressões de tempo podem ser classificadas segundo vários tipos:

Precisas: são expressões que denotam uma data de calendário ou uma duração (e.g., *July 15, 1999*);

Imprecisas: são expressões vagas ou de limites imprecisos (e.g., *a year ago, almost two o'clock*);

Modificadas: são expressões de certa forma quantificadas ou modificadas através de modificadores (e.g., *approximately, no more than*);

Frequências: são expressões que expressam a frequência com que algo ocorre (e.g., *every Tuesday, every Deceber 31*);

Não-Específicas: são expressões que não referenciam um tempo específico. Subdividem-se noutros dois tipos:

1. Genéricas: especificam uma classe de entidades temporais em vez de um tempo específico (e.g., *I love December, Winters are cold*);
2. Indefinidas: são expressões indefinidas (e.g., *on a Tuesday*).

De forma a permitir a caracterização de tais expressões, a etiqueta TIMEX2 contempla ainda vários atributos opcionais. Segundo o *TIMEX2 Quick Guide* (Ferro et al., 2003), estes atributos são:

VAL – Este atributo corresponde à normalização de expressões relativas a datas de calendário ou horas, tempo geológico ou BCE[2], e durações. Usa uma extensão do esquema ISO 8601 que:

- captura um maior número de unidades de tempo (e.g., 196 significa *the 1960s*, P1C significa *for one century*, MA210 significa *210 million years ago*);
- permite que *tokens* representem todo ou parte de um valor (e.g., PRESENT_REF (para *now*), SU (para *summer*), WE (para *weekend*), NI (para *night*));
- faz um uso adequado da variável livre "X" para expressões mal especificadas (e.g., *the past few months*), incluindo as que denotam apenas granularidade (e.g., *monthly*);
- permite misturar formatos baseados em meses e formatos baseados em semanas (para uma expressão como *one Friday night in October 1998*).

[2]BCE - Before The Common/Christian Era

Exemplos da aplicação deste atributo são:

```
<TIMEX2 VAL="1991-10-06">October 6, 1991</TIMEX2>
<TIMEX2 VAL="1992-SU">last summer</TIMEX2>
<TIMEX2 VAL="1993-12-02">five days ago</TIMEX2>
```

MOD – Este atributo é usado para capturar a semântica de certos modificadores, tais como *no latter than* ou *late*, que podem aparecer no ambito de uma expressão TIMEX2. Surge combinado com o atributo VAL. Os possíveis valores para MOD são:

- BEFORE, AFTER, ON_OR_BEFORE, ON_OR_AFTER para expressões que designam pontos no tempo;
- LESS_THAN, MORE_THAN, EQUAL_OR_LESS, EQUAL_OR_MORE para expressões referentes a durações;
- START, MID, END, APPROX para ambos os casos anteriores.

Exemplos da aplicação deste atributo são:

```
<TIMEX2 VAL="P1Y" MOD="LESS_THAN">less than a year</TIMEX2>
<TIMEX2 VAL="1996-08" MOD="END">the end of August</TIMEX2>
<TIMEX2 VAL="199" MOD="START">the early 1990s</TIMEX2>
```

SET – Este atributo designa expressões que denotam conjuntos de tempo, assim como *every year, nearly every week, the past two days*. Quase sempre aparece associado com o atributo VAL. O seu único valor possível é `YES`. Exemplos da aplicação deste atributo são:

```
<TIMEX2 SET="YES" VAL="XXXX-XX-XX">daily</TIMEX2>
<TIMEX2 SET="YES" VAL="XXXX-HX">semiannual</TIMEX2>
```

ANCHOR_VAL – Este atributo indica a normalização (de acordo com o formato ISO) da referência a horas ou datas de calendário para interpretação de uma expressão vaga ou parcialmente ancorada, como *now* ou *the past few years*. Surge associado com o atributo VAL para tais expressões. O tempo de referência pode ser o tempo narrativo ou o tempo do documento. Os atributos ANCHOR_VAL e ANCHOR_DIR surgem sempre em conjunto.

ANCHOR_DIR – Este atributo diz respeito à normalização da direccionalidade. Pode ter associados os valores: WITHIN, STARTING, ENDING, AS_OF, BEFORE ou AFTER. Por exemplo, *now* ancora a direccionalidade de AS_OF e *the past few years* ancora a direccionalidade de BEFORE. Exemplos da aplicação deste atributo são:

```
<TIMEX2 VAL="PRESENT_REF" ANCHOR_VAL="1994-01-21T08:29"
ANCHOR_DIR="AS_OF">now</TIMEX2>
<TIMEX2 VAL="P9M" ANCHOR_VAL="1993-08"
ANCHOR_DIR="ENDING">the last nine months</TIMEX2>
<TIMEX2 VAL="PXY" ANCHOR_VAL="1993"
ANCHOR_DIR="BEFORE">recent years</TIMEX2>
```

NON_SPECIFIC – Este é um atributo genérico que designa essencialmente expressões temporais não referenciáveis (*I love December, It's a sunny day, He will decide at the last minute, We work 9 to 5*). Exemplo da aplicação deste atributo é:

```
<TIMEX2 VAL="P1Y" NON_SPECIFIC="YES"
ANCHOR_DIR="ENDING">per year</TIMEX2>
```

Atributo	Função	Exemplo
VAL	Contém uma forma normalizada do tempo/data.	VAL="1964-10-16"
MOD	Captura os modificadores temporais.	MOD="APPROX"
SET	Identifica expressões que denotam conjuntos de tempo.	SET="YES"
ANCHOR_VAL	Contém uma forma normalizada do tempo/data "ancorado".	ANCHOR_VAL="1964-10-16"
ANCHOR_DIR	Captura a direccionalidade entre VAL e ANCHOR_VAL.	ANCHOR_DIR="BEFORE"
NON_SPECIFIC	Identifica expressões de tempo não referenciáveis.	NON_SPECIFIC="YES"

Tabela 2.1: Tabela de atributos do TIMEX2.

A Tabela 2.1 descreve, sumariamente, os diversos atributos do TIMEX2 tal como proposto por Pustejovsky (Pustejovsky et al., 2003).

Para além dos tipos de expressões anteriormente referidos, existe ainda um número de outros tipos que podem ser problemáticos para os anotadores de texto. Nessa categoria incluem-se as expressões

temporais ancoradas em eventos, expressões determinadas culturalmente e expressões cujos valores podem mudar.

Uma expressão temporal ancorada num evento é uma expressão, que para que o seu valor possa ser totalmente resolvido, torna-se necessário saber o tempo do evento. Por exemplo, para determinar o valor de *the day* na expressão *the day after our meeting*, é preciso saber quando é que ocorreu a reunião. Expressões determinadas culturalmente, são aquelas cuja interpretação requer, tipicamente, conhecimento a nível cultural ou de um domínio específico (e.g., *baseball season, prime time*).

Por fim, existe o tipo de expressões cujos valores podem mudar, onde se enquadram expressões do tipo *the stock price fell from $4.02 to $3.85*, em que o valor do *stock* varia com o tempo; ou expressões em que é usada uma entidade para referir outra, como é o caso de *September 11*, que se refere ao ataque terrorista e não simplesmente à data em si.

2.2.2 TimeML - Markup Language for Temporal and Event Expressions

Um outro trabalho semelhante ao TIDES na anotação de expressões temporais é o TimeML (*TimeML - Markup Language for Temporal and Event Expressions*, n.d.). O TimeML é uma linguagem de especificação de eventos e expressões temporais em língua natural, desenvolvido no âmbito do projecto TERQAS (*Time and Event Recognition for Question Answering Systems*) e que tem também associado um conjunto de princípios de anotação (Pustejovsky et al., 2006).

O TimeML foi desenvolvido tendo em consideração os seguintes problemas na marcação de eventos e expressões temporais:

- *Time stamping* de eventos (identificar um evento e ordená-lo cronologicamente);
- Ordenação de uns eventos em relação aos outros;
- Raciocínio sobre expressões temporais contextualmente mal especificadas (funções temporais tais como *a semana passada* ou *duas semanas antes*);
- Raciocínio sobre a persistência de eventos (quanto tempo dura um evento ou o seu resultado).

Em resumo, o TimeML consiste num esquema de anotação para identificação de eventos mencionados num documento de texto, permitindo orientá-los segundo uma linha de tempo.

No centro de qualquer esquema cuja finalidade é a percepção de informação temporal, existe um método para representar expressões temporais tais como: *1961*, ou *today*. Para modelar este tipo de expressões, o TimeML defininiu a etiqueta TIMEX3, que é uma variação do esquema TIMEX2. O processo de identificação de informação temporal é semelhante nos dois esquemas de anotação, no entanto, contrariamente ao que acontece no TIMEX2, o TimeML foca-se no conteúdo em vez de nas palavras-chave.

De acordo com Pustejovsky (Pustejovsky et al., 2005), existem quatro tipos de expressões temporais que são capturados pela etiqueta TIMEX3: TIME, DATE, DURATION e SET. Uma expressão do tipo TIME, é uma expressão que se refere a uma parte do dia, mesmo que de uma forma muito indefinida:

Mr. Smith left at 9 a.m. Friday, October 1, 1999;

Mr. Smith left ten minutes to three;

Mr. Smith left late last night.

Note-se que, com excepção do primeiro exemplo, as expressões referidas não especificam completamente a hora/data em que o evento ocorreu. Estas expressões necessitam de mais informação para representarem o que realmente pretendem. Este é um problema frequente e que o TimeML trata recorrendo às funções temporais.

O tipo DATE[3] pode ser definido como uma expressão que se refere a uma data de calendário. Alguns exemplos que se enquadram nesta categoria são:

Mr. Smith left Friday, October 1, 1999;

Mr. Smith left in October of 1963;

Mr. Smith left last week.

Uma expressão é do tipo DURATION se descreve explicitamente uma extensão de tempo. Alguns exemplos são:

Mr. Smith stayed 2 months in Boston;

Mr. Smith stayed 3 weeks in Boston;

Mr. Smith stayed 3 hours last Monday;

Finalmente, o tipo SET (tal como no TIMEX2) designa expressões que referem tempos que ocorrem regularmente, ou seja, a frequência com que algo acontece. Estas são expressões como:

John swims twice a week;

John swims every two days.

[3]Segundo Pustejovsky, o método mais fácil para distinguir um TIME de um DATE é olhar para a granularidade da expressão. Se a granularidade da expressão for mais pequena que um dia, então enquadra-se na categoria TIME.

Tal como acontece no TIMEX2, quando uma data ou hora não se encontra completamente especificada, podem ser introduzidas variáveis no atributo `value` de forma a normalizar a expressão em causa. Por exemplo, uma expressão como, *January 12* não transmite qualquer informação sobre o ano. Assim sendo, o seu valor seria especificado como XXXX-01-12.

Uma outra característica que provém do TIMEX2 é a possibilidade de ser identificadas durações de tempo. As durações têm um formato especial para o atributo `value`, pois representam um período de tempo. Exemplo disto é:

```
<TIMEX3 tid="t1" type="DURATION"
value="P3D">three days</TIMEX3>
```

O TimeML tenta capturar todas as expressões temporais através da etiqueta TIMEX3. No entanto, muitas são as expressões às quais falta informação crucial para a sua completa especificação. De facto, a análise de um corpus revela que geralmente existem muito poucas expressões completamente especificadas. As funções temporais têm como objectivo capturar a informação que está em falta, através das outras expressões. Portanto, quando uma expressão TIMEX3 não está completamente especificada, esta é associada a uma outra expressão através do atributo `temporalFunction`. Tome-se como exemplo um artigo de jornal que tem uma data associada. Se no corpo desse artigo surgir a expressão *today*, esta é associada à data do documento, de modo a ficar completamente determinada:

```
<TIMEX3 tid="t1" type="DATE" value="PRESENT_REF"
temporalFunction="true" valueFromFunction="tf1"
anchorTimeID="t0">
today
</TIMEX3>
```

Os atributos `valueFromFunction` e `anchorTimeID` referem-se, respectivamente, à identificação da função temporal que completa a especificação e à identificação de outra expressão TIMEX3 que disponibiliza informação à função temporal.

Segundo a equipa responsável pelo desenvolvimento do TimeML (Pustejovsky et al., 2004), ao contrário dos trabalhos anteriores desenvolvidos na área da especificação de eventos e de tempo, o TimeML separa a representação de eventos e expressões temporais das dependências de ligação ou ordenação que possam existir num determinado texto. Uma das mais importantes inovações introduzidas no TimeML são as etiquetas LINK. O conjunto de etiquetas LINK codifica as várias relações que existem entre os elementos temporais num documento, assim como estabelece uma ordem entre os eventos. O TimeML especifica quatro principais estruturas de dados: EVENT, TIMEX3, SIGNAL e LINK (que se decompõe nas etiquetas TLINK, SLINK e ALINK).

A etiqueta EVENT captura os eventos temporais. A etiqueta TIMEX3 é usada para capturar todas as expressões de tempo. Palavras funcionais como *at* e *from*, que indicam a forma como os objectos temporais se relacionam entre si, são etiquetadas com a etiqueta SIGNAL. Finalmente, as relações entre eventos ou entre eventos e tempo são capturadas pelas etiquetas TLINK, SLINK e ALINK. Segue-se uma descrição de cada uma destas etiquetas.

<**EVENT**> – Eventos são situações que ocorrem pontualmente ou durante um determinado período de tempo. São também considerados eventos, predicados que descrevem *estados* ou *circunstâncias* em que algo é verdadeiro. Os eventos são expressos através de verbos ou tempos verbais, nominalização, adjectivos, cláusulas predicativas ou sintagmas preposicionais. Os eventos podem ser de vários tipos:

Occurrence: *die, crash, build, merge, sell*;

Perception: *see, hear, watch, feel*;

Reporting: *say, report, announce*;

Aspectual: *begin, finish, stop, continue*;

State: *on board, kidnapped, love*;

I_State: *belive, intend, want*;

I_Action: *attempt, try, promise, offer*.

Exemplo da aplicação da etiqueta EVENT:

```
All 75 passengers <EVENT eid="1" class="OCCURRENCE"
tense="past" aspect="NONE">died</EVENT>
```

Os eventos são frequentemente identificados através da anotação da cabeça da sintagma verbal. Um identificador único (*identifier*), a classe do evento (*event class*), o tempo (*tense*) e o aspecto (*aspect*), são os atributos que classificam um *event*. Os atributos *tense* e *aspect* são preenchidos numa fase de pré-processamento de acordo com a especificação do TimeML (*TimeML - A Formal Specification Language for Events and Temporal Expressions*, n.d.).

<**TIMEX3**> – A etiqueta TIMEX3 é utilizada para marcar expressões temporais explícitas, tais como, horas, datas ou durações e é modelada segundo as etiquetas TIMEX2 (Ferro et al., 2005) do TIDES e TIMEX da linguagem de anotação temporal proposta por Setzer (Setzer, 2001). As expressões temporais identificadas pelo TIMEX3 subdividem-se em três classes:

1. Expressões Temporais Concretas: *June 11, 1989, Summer, 2002*;
2. Expressões Temporais Relativas: *Monday, Next year, Two days ago*;
3. Durações: *Three months, Two years*.

Exemplo da aplicação da etiqueta TIMEX3:

```
<TIMEX3 tid="1" type="DATE" value="2004-11-22">
November 22, 2004</TIMEX3>
```

Atributos adicionais são usados, por exemplo, para ancorar expressões temporais relativas a outras expressões, de modo a permitir a computação dos valores de tempo absolutos (e.g., *last week*).

<**SIGNAL**> – A etiqueta SIGNAL é usada para anotar secções de texto, tipicamente palavras funcionais, que indicam como é que os objectos temporais devem ser relacionados entre si. As expressões marcadas como SIGNAL constituem vários tipos de elementos linguísticos: indicadores de relações temporais, como por exemplo preposições temporais (e.g. *on, during*) e outras conectivas temporais (e.g. *when*) e subordinações (e.g. *if*). A funcionalidade base da etiqueta SIGNAL foi introduzida por Setzer (2001).

Um exemplo da aplicação da etiqueta SIGNAL é:

```
Two days <SIGNAL sid="1">before</SIGNAL> the attack...
```

<**TLINK**> – TLINK (*Temporal Link*) representa a relação temporal que existe entre eventos ou entre um evento e um tempo e estabelece uma ligação entre as entidades envolvidas, tornando explícito se a sua relação é do tipo *before, after, includes, is_included, holds, simultaneous, immediately after, immediately before, identity, begins, ends, begun by* ou *ended by*. Estas são as 13 relações de Allen referidas na secção 2.1.

Para ilustrar a funcionalidade desta etiqueta, tome-se como exemplo a frase seguinte, adicionando-lhe a anotação de TLINK, que ordena os dois eventos mencionados, com uma magnitude denotada pelo valor da expressão temporal:

John left 2 days before the attack.

```
<TLINK eventInstanceID="ei1" signalID="s1"
relatedToEvent="ei2" relType="BEFORE" magnitude="t1"/>
```

Note-se que, esta ligação compõe duas asserções: (i) a partida de John, e_{i1}, precede o ataque, e_{i2}; e (ii) o intervalo que separa estes eventos tem uma magnitude igual ao valor da expressão temporal t_1.

<**SLINK**> – SLINK (*Subordination Link*) é usada para identificar relações entre dois eventos, ou um evento e um sinal. Podem ser de vários tipos:

1. **Modal**: Relação introduzida, principalmente, pelos verbos modais (*should, could, would*, etc.) e eventos que introduzem uma referência a um possível mundo; estes são na sua maioria do tipo I_STATE:
   (a) *John should have bought some wine;*
   (b) *Mary wanted John to buy some wine.*
2. **Factive**: Certos verbos pressupõem a veracidade do argumento. Incluem *forget, regret, manage*:
   (a) *John forgot that he was in Boston last year;*
   (b) *Mary regrets that she didn't marry John;*
   (c) *John managed to leave the party.*
3. **Counterfactive**: O evento introduz uma pressuposição sobre a não veracidade do seu argumento. Por exemplo: *forget (to), unable to* (no passado), *prevent, cancel, avoid, decline*:
   (a) *John forgot to buy some wine;*
   (b) *Mary was unable to marry John;*
   (c) *John prevented the divorce.*
4. **Evidential**: Relações evidenciais são introduzidas por eventos do tipo REPORTING ou PERCEPTION:
   (a) *John said he bought some wine;*
   (b) *Mary saw John carrying only beer.*
5. **Negative Evidential**: Relações introduzidas por eventos REPORTING e alguns eventos do tipo PERCEPTION que denotam polaridade negativa:
   (a) *John denied he bought only beer.*
6. **Negative**: Relações introduzidas por partículas negativas (*not, nor, neither*, etc.), que são marcadas como SIGNAL, a respeito dos eventos que estão a modificar:
   (a) *John didn't forgot to buy some wine;*

(b) *John did not wanted to marry Mary.*

Um predicado modal como *want* introduz uma ligação SLINK como se verifica no exemplo seguinte em que os eventos e_{i1} e e_2 são, respectivamente, o *wants* e o *teach*, e s_1 tem o valor *to*:

Bill wants to teach on Monday.

```
<SLINK eventInstanceID="ei1" signalID="s1"
SubordinatedEvent="e2" relType="MODAL"/>
```

<**ALINK**> – ALINK (*Aspectual Link*) representa a relação entre um evento *aspectual* e o seu argumento evento. São exemplos das possíveis relações aspectuais:

1. **Initiation**: *John started to read*;
2. **Culmination**: *John finished assembling the table*;
3. **Termination**: *John stopped talking*;
4. **Continuation**: *John kept talking*.

Para ilustrar o comportamento do ALINK, note-se que o predicado aspectual *begin* é tratado como um evento isolado, independente do evento logicamente modificado; a mudança é introduzida como uma relação dentro do ALINK. Esta relação é expressa no exemplo seguinte, onde os eventos e_1 e e_2 são, respectivamente, o *began* e o *sink*, e s_1 tem o valor *to*:

The boat began to sink.

```
<ALINK eventInstance="e1" signalID="s1" relatedToEvent="e2"
relType="INITIATES"/>
```

Para além das etiquetas já mencionadas, existe ainda uma outra que se dá pelo nome de MAKEINSTANCE e que permite distinguir diferentes instâncias de um determinado evento. Se se considerar a frase *John teaches on Monday and Tuesday*, verifica-se que existe apenas um verbo que representa dois eventos. Para se proceder à anotação de tais casos, é necessário criar duas instâncias de *teaches*, de forma a representar os dois diferentes eventos. Deve ser criada uma MAKEINSTANCE por cada instância de um evento presente no texto. No entanto, quando a cardinalidade é elevada, o anotador pode optar

por criar apenas uma MAKEINSTANCE, associando-lhe a cardinalidade apropriada. Considere-se a frase *John teaches on Monday and Tuesday*. A aplicação da etiqueta MAKEINSTANCE a este exemplo, corresponde à criação das instâncias e_{i1} e e_{i2} do evento e_1 (*teaches*):

```
John
<EVENT> eid="e1" class="OCCURRENCE">
teaches
</EVENT>
on Monday and Tuesday.
<MAKEINSTANCE eiid="ei1" eventID="e1" tense="PRESENT"
aspect="NONE"/>
<MAKEINSTANCE eiid="ei2" eventID="e1" tense="PRESENT"
aspect="NONE"/>
```

Concluindo, as características que distinguem o TimeML dos outros projectos são:

1. Extensão dos atributos do TIMEX2;
2. Introdução de funções temporais que permitem expressões especificadas intencionalmente: *three years ago*, *last month*;
3. Identificação de sinais que determinam a interpretação de expressões temporais:
   (a) Preposições temporais: *for*, *during on*, *at*;
   (b) Conectivas temporais: *before*, *after*, *while*.
4. Identificação de todas as classes de expressões de eventos:
   (a) Tempos verbais: *has left*, *was captured*, *will resign*;
   (b) Adjectivos de estado e outros modificadores: *sunken*, *stalled*, *on board*;
   (c) Eventos nominais: *merger*, *Military Operation*, *Gulf War*.
5. Criação de dependências entre tempo e eventos:
   (a) Dependências de ligação (*anchoring*): *John left on Monday*;
   (b) Dependências de ordem (*ordering*): *The party happened after Midnight*;
   (c) Dependências embutidas (*embedding*): *John said Mary left*.

2.3 Conclusão

Neste capítulo foi feita uma abordagem a duas lógicas temporais cuja principal diferença reside na forma em como decompõem o tempo. As lógicas referidas são a Tense Logic e a Lógica Baseada em Intervalos que assumem como unidades primitivas de tempo, o ponto e o intervalo, respectivamente.

No que concerne aos esquemas de anotação – TIDES e TimeML – foram referidas as várias questões envolvidas na anotação de conteúdo temporal de um texto.

Podem ainda ser estabelecidas relações entre os esquemas de anotação e as ontologias [4] de tempo. No entanto, este tema não é abordado uma vez que já se encontra fora do âmbito deste trabalho.

Contudo, deixa-se como referência alguns trabalhos desenvolvidos na área das ontologias de tempo, nomeadamente as ontologias KSL (Zhou & Fikes, n.d.) e a DAML-Time (*DAML-Time Homepage*, n.d.) que têm como particularidade o relacionamento com os projectos TIDES e TimeML, respectivamente, descritos na secção 2.2 deste documento.

[4] Uma ontologia é um conjunto de categorias ou distinções através das quais é possível conceptualizar o mundo (Beule, n.d.).

3 Reconhecimento de Entidades Mencionadas

O Reconhecimento de Entidades Mencionadas é feito de acordo com os critérios de delimitação, estabelecidos para cada uma das categorias de entidades, descritos na secção 3.1 deste capítulo. A tarefa de reconhecimento ou identificação é feita através de uma cadeia de processamento da qual faz parte o analisador sintáctico da XEROX (secção 3.2). Na secção 3.3 encontra-se descrito o processo de reconhecimento das entidades segundo as várias categorias.

3.1 Entidades e Critérios de Delimitação

Nesta secção apresentam-se os critérios de delimitação das entidades mencionadas, nomeadamente os que dizem respeito às categorias Obra, Valor, Ralações de Parentesco e Tempo.

3.1.1 Obra

A categoria Obra refere-se a títulos ou nomes de obras. Entende-se por obra qualquer coisa feita pelo Homem e que tenha um nome próprio (não comum).

As directivas do HAREM (Cardoso & Santos, 2005) subdividem a categoria Obra em quatro tipos distintos: tipo Produto, tipo Reproduzida, tipo Arte e tipo Publicação.

Produto Esta subcategoria refere um produto concreto. Muitas vezes o nome de um produto é idêntico ao nome de uma marca, a diferença está no contexto: uma obra do tipo Produto é algo que se faz em série e tem um nome concreto, enquanto que a marca é algo mais abstracto (e.g., Fiat Punto, Microsoft Word). A marca ou empresa, só por si, não deve ser etiquetada.

Reproduzida Esta subcategoria refere-se a obras das quais há muitos exemplares. O nome representa o original a partir do qual se fazem as reproduções (e.g., «*E tudo o Vento Levou*», «*Sinfonia em si bemol*», de Carlos Seixas).

Arte Esta subcategoria refere-se a obras ou objectos dos quais há um exemplar único, como por exemplo *Torre Eiffel*, *Cristo-Rei*, *Capela Sistina*, ou *Ponte da Arrábida*. Repare-se que quando a entidade mencionada se refere a certos edifícios ou monumentos, este podem ser, simultaneamente, classificados como entidades do tipo Local.

Publicação Esta subcategoria abrange obras escritas não referidas pelo nome, tais como citações de livros, artigos, decretos, directivas, entre outros. A etiqueta deve abranger todas as palavras relacionadas com a publicação, inclusive nomes de editoras e/ou locais da publicação (e.g., *Maia et al. (2004)*, *Santos & Sarmento (2003:114)*, *Mota (op.cit.)*, *Decreto Lei 254/94*).

3.1.2 Valor

Um valor pode referir-se a quantidades absolutas ou relativas, designar dinheiro ou classificações desportivas, ordinais normais e outras. Dada esta definição, podem-se extrair três subcategorias:

Classificação Esta subcategoria engloba valores que traduzem classificações, ordenações ou pontuações (e.g., 2-1, 86 pontos, 6ª [Exposição Mundial de Cinema]). São consideradas excepções enumerações de parágrafos, tópicos e graus escolares e académicos.

Moeda Na subcategoria Moeda enquadram-se entidades mencionadas que consistem num número associado a uma moeda. A etiquetação deve, portanto, abranger não só o número que representa a quantidade como também a respectiva moeda (e.g., 300$00, £40, 50 contos, 120 milhões de euros).

Quantidade Uma quantidade não é mais que um número quantificado por uma unidade (e.g., 3 kg, 50mm, 17ºC, 20 Hz). Por unidades entendem-se as usadas para medir propriedades como distância, tempo, luz, área, volume, peso, massa, etc). Objectos que podem ser contáveis não são considerados unidades (e.g., pessoas, folhas de papel, cadernos). As unidades monetárias (e.g., euros) são já abrangidas pela subcategoria Moeda, enquanto que as unidades de tempo (e.g., anos) dizem respeito à categoria Tempo.

3.1.3 Relações de Parentesco

De acordo com o HAREM, os graus de parentesco são considerados entidades do tipo Individual, da categoria Pessoa, mas apenas se precederem um nome próprio (e.g., avô João). No entanto, o desenvolvimento de um sistema de pergunta-resposta, no L^2F (Laboratório de Sistemas de Língua Falada), requer que sejam identificadas todas as relações de parentesco, independentemente de estarem, ou não, associadas a um nome próprio, pelo que se optou por classificar tais entidades segundo uma categoria individual. Assim, esta categoria abrange todo o tipo de relações de parentesco (e.g., pai, primo, ex-namorada, tio-avô) que podem, ou não, preceder um nome.

3.1.4 Tempo

A categoria Tempo refere-se a todo o tipo de entidades que denotam conteúdo temporal. Assim sendo, as entidades mencionadas desta categoria podem ser do tipo Hora, Data, Duração ou Frequência.

Hora Esta subcategoria refere-se às expressões temporais cuja unidade de tempo é mais pequena que o dia (e.g., 10 horas, meio-dia, 10 da noite, 10:30, 20 para as 10).

Data Nesta subcategoria enquadram-se as expressões temporais cuja unidade de tempo é igual ou maior que o dia (e.g., dia 2 de Janeiro, Janeiro de 1990, século XX). Uma data pode também ser qualquer combinação de expressões temporais cuja unidade é igual ou maior que o dia. Como exemplo dessas várias expressões tem-se o dia da semana (e.g., quarta-feira), o dia do mês (e.g., dia 2), a semana (e.g., próxima semana), o mês (e.g., Janeiro), o ano (e.g., 2002), o século (e.g., século XXI). Podem ainda ser consideradas datas, expressões de tempo relativas (e.g., próximo dia 2, há 2 anos, mês anterior).

Duração Uma duração é uma qualquer referência a um período de tempo contínuo no qual pode, ou não, decorrer uma acção (e.g., [durante] 2 anos [pratiquei atletismo], [há] 2 anos [participei na meia-maratona]).

Frequência Uma frequência designa o número de vezes que uma determinada acção se repete num dado período de tempo (e.g., 2 vezes por semana, de 2 em 2 dias, mensalmente).

O HAREM define quatro subtipos para as entidades da categoria Tempo: Data, Hora, Período e Cíclico. A definição das entidades do tipo Data referente ao HAREM diverge da definição adoptada para este trabalho, dado que não permite a inclusão na entidade de palavras que não referem explicitamente a data (e.g., [final de] 1999). A inclusão deste tipo de palavras é importante, na medida em que permite obter mais informação acerca das datas.

No que diz respeito às entidades do tipo Hora, as directivas adoptadas são idênticas às propostas para o HAREM.

De acordo com o HAREM, um período refere-se a um intervalo de tempo contínuo e não repetido, com apenas um início e um fim (e.g., Inverno, anos 80, século XIX, 1984, pós-25 de Abril). Uma entidade temporal do tipo Cíclico compreende períodos recorrentes, quando empregues como tal (e.g., Natal, 1º de Maio, Carnaval). Considerem-se as as três frases seguintes:

Vou três vezes a Londres no próximo Inverno.

O Inverno em Oslo costuma ser frio.

A Joana nasceu no Inverno passado.

Segundo as directivas do HAREM, a mesma entidade mencionada pode referir um período único, um tempo cíclico, ou uma data. No primeiro caso *Inverno* é considerado um tempo periódico. Na

segunda frase, *Inverno* é classificado como um tempo cíclico. Por fim, no último exemplo, *Inverno* é uma entidade temporal do tipo Data.

A minuciosidade destas directivas pode dar origem a diversas ambiguidades dificultando o reconhecimento de entidades, até porque uma data não deve deixar de ser considerada como tal, apesar de referenciar um tempo cíclico (e.g., [no dia] 25 de Abril [comemora-se...]).

Estas directivas são um pouco contraditórias ao permitirem classificar como Período entidades que não reflectem explicitamente conteúdo temporal (e.g., [depois da] IBM [fui para a SUN]).

Por último, devido ao facto do HAREM não suportar expressões de tempo incertas ou relativas (e.g., o próximo mês) optou-se por seguir as directivas inicialmente apresentadas, que resultam de uma adaptação das directivas do TimeML (Pustejovsky et al., 2006) para a língua portuguesa, originalmente estabelecidas para o inglês.

3.2 Cadeia de Processamento e XIP

O XIP é uma ferramenta central no desenvolvimento deste trabalho. Nesta secção encontra-se uma descrição da arquitectura desta ferramenta, assim como da sua gramática. É ainda feita uma breve abordagem à cadeia de processamento utilizada, da qual o XIP é parte integrante.

3.2.1 Cadeia de Processamento

Como se pode observar na figura 3.1, a cadeia de processamento é um conjunto de vários módulos entre os quais existem conversores de XML (eXtensible Markup Language) que fazem a conversão de dados entre módulos.

De acordo com a cadeia de processamento em Maio de 2007 (Mamede, 2007), o módulo de Segmentação (*Tokenize*) é responsável pela identificação de determinadas entidades, tais como:

- endereços de e-mail;
- números IP e endereços HTTP;
- números ordinais terminados em º ou ª (e.g., `12º`);
- números com `.` e `,` (e.g., `1.234,50`);
- números inteiros (e.g., `1234`);
- diversos tipos de abreviaturas (e.g., `a.c.`, `V.Exa.`);
- sequências de `!`, `?` e `...` (e.g., `!!!`, `?!?!?!?`);

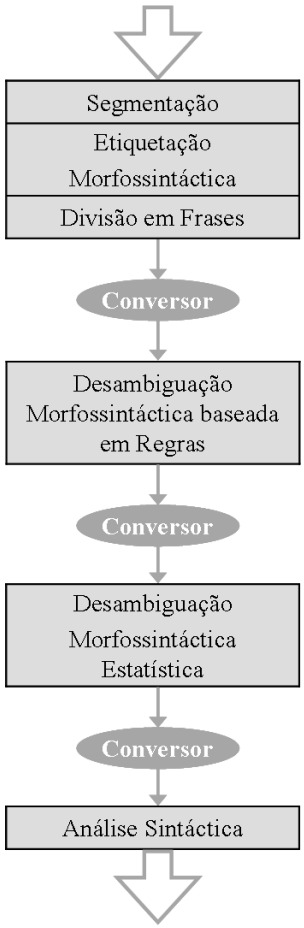


Figura 3.1: A cadeia de processamento.

- sinais de pontuação (e.g., `!`, `?`, `.`, `:`, `;`, `(`, `)`, `[`, `]`, `-`, `_`);
- símbolos (e.g., `«`, `»`, `#`, `$`, `%`, `&`, `+`, `*`, `<`, `=`, `@`);
- números romanos (e.g., MMVII);
- sequência de caracteres não aceites pelo *Palavroso*;
- palavras (e.g., `alface`, `fim-de-semana`).

O módulo de Segmentação pode ser parametrizado com ficheiros com listas de abreviaturas.

Em seguida, no módulo de Etiquetação Morfossintáctica, o Palavroso (Medeiros, 1995) codifica os segmentos identificados anteriormente segundo 10 campos (categoria, subcategoria, modo, tempo, pessoa, número, género, grau, caso, formação), tendo a categoria 13 valores possíveis (nome, verbo, adjectivo, pronome, artigo, advérbio, preposição, conjunção, numeral, interjeição, marcador passiva e pontuação). Este módulo aceita um dicionário como entrada.

O módulo Divisão em Frases faz a divisão do texto em frases, considerando como terminadores de frase todos os segmentos constituídos unicamente por `.`, `!`, ou `?`.

Posteriormente, no módulo de Desambiguação Morfossintáctica baseada em Regras, é executado o RuDriCo (Rule Driven Converter) (Pardal, 2007) que tem como primeiro objectivo a adaptação dos resultados produzidos pelo analisador morfológico às necessidades específicas de cada analisador sintáctico. Para tal, modifica a segmentação feita pelo analisador morfológico. Nesta chamada, o RudDriCo faz correcções à saída do Palavroso (e.g., `sida`, que poderia, erradamente, ser classificada como um particípio passado do verbo *ser*), efectua alterações dos lemas dos pronomes, advérbios, artigos e outras categorias gramaticais (e.g., `quaisquer`, dado que o plural de *qualquer* é irregular), faz a descontracção de palavras (e.g., `nas = em + as`), executa regras de desambiguação morfossintáctica e identifica locuções prepositivas e adverbiais, entre outras (e.g., `à esquerda de`). Nesta fase, o RuDriCo pode ser parametrizado com regras de descontracção, de locuções e de desambiguação.

Segue-se a execução do módulo de Desambiguação Morfossintáctica Estatística. A funcionalidade deste módulo é desempenhada pelo Marv (Ribeiro et al., 2003) que, recorrendo ao algoritmo de Viterbi (Jurafsky & Martin, 2000), selecciona uma das etiquetas de cada palavra, sendo essa escolha efectuada apenas com base na categoria e subcategoria. No caso dos verbos, se a palavra tiver associadas várias etiquetas de verbo após a selecção da categoria, o Marv opta pela primeira delas.

Numa última fase, é executado o analisador sintáctico (XIP) que tem como finalidade introduzir informação léxica, aplicar regras de desambiguação morfossintáctica, aplicar gramáticas locais, ignorar as etiquetas preteridas pelo Marv, fazer a segmentação em *chunks* e calcular as dependências entre os mesmos. O XIP é parametrizável através de gramáticas e de léxicos.

3.2.2 Arquitectura XIP

O XIP (Xerox Incremental Parser) (Xerox, 2003) é uma ferramenta que recebe um texto como entrada e fornece informação linguística acerca do mesmo. Este conversor pode desambiguar a informação lexical, segmentar o texto em nós (*chunks*) ou em outros tipos de agrupamento e criar dependências entre eles. Uma dependência é uma relação linguística entre duas unidades linguísticas.

O XIP é composto por três módulos principais e dois módulos opcionais de pré-processamento 3.2. Os módulos principais são:

- Módulo de desambiguação contextual: o XIP recorre a este módulo para atribuir traços (*features*) ou categorias às palavras de acordo com o contexto em que se encontram;
- Módulo de agrupamento (*chunking*): este módulo segmenta as unidades linguísticas em nós ou em outros tipos de agrupamento;

- Módulo de dependências: este módulo usa regras para identificar dependências entre unidades linguísticas.

Os módulos opcionais de pré-processamento são:

- Normalização, Tokenização e Morfologia (NTM): este módulo providencia a forma normalizada e toda a potencial informação léxica para cada palavra identificada;
- Desambiguador Hidden Markov Model (HMM): este módulo recorre ao algoritmo *Hidden Markov Model* para determinar qual a categoria gramatical mais provável a atribuir a uma palavra tendo em conta o seu contexto imediato.

Os módulos de pré-processamento NTM e HMM não se encontram disponíveis no L^2F, sendo a funcionalidade do NTM subsituída pelos módulos de Segmentação e Etiquetação Morfossintáctica e o HMM substistuído pelo módulo de Desambiguação Morfossintáctica Estatística.

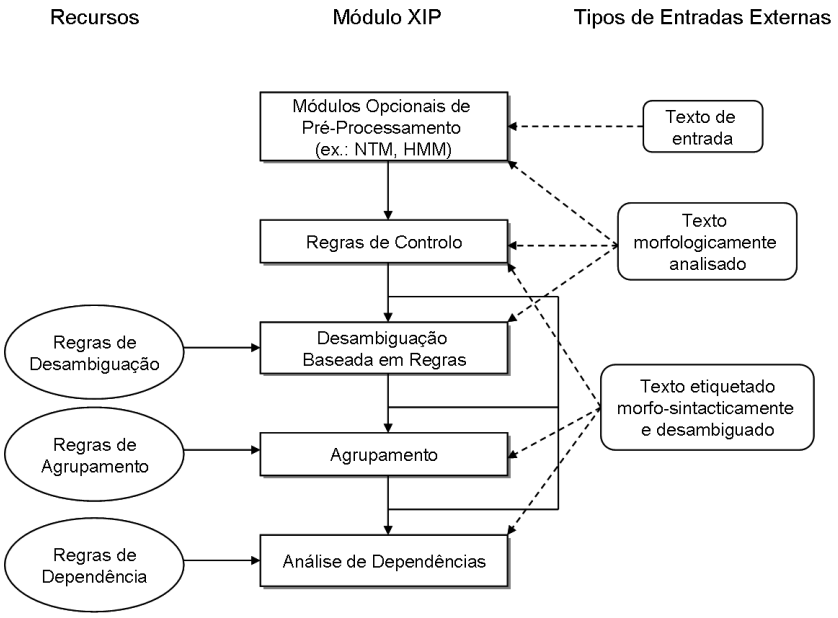


Figura 3.2: A arquitectura do XIP.

O mecanismo de desambiguação baseado em regras faz a leitura mais provável de uma palavra dado o seu contexto. Por exemplo, a palavra *comida* pode ser interpretada como um nome ou um particípio passado do verbo comer, dependendo da situação em que é empregue:

```
A comida estava boa. //nome
A cenoura foi comida pelo coelho. //particípio passado
```

As regras de desambiguação podem também ser utilizadas para redefinir valores de traços com base no contexto do nó.

O módulo de agrupamento (*chunking*) usa regras de agrupamento para agregar sequências de categorias em estruturas que possam ser depois processadas pelo módulo de dependências. Estas regras encontram-se organizadas em camadas e são aplicadas a sequências de categorias, uma após a outra, no texto desambiguado. Após o processamento estar terminado, o XIP cria uma árvore de nós (*chunks*).

Considere-se como exemplo a frase: *A Joana comprou um carro*. A árvore correspondente a esta frase é composta por três nós – um sintagma nominal (NP) e um sintagma verbal (VP) seguido de um outro sintagma nominal (NP):

```
NP[A Joana] VP[comprou] NP[um carro]
```

O módulo de dependências estabelece relações de dependência entre palavras ou nós. Estas relações de dependência são especificadas na gramática e podem incluir dependências, tais como `SUBJ` ou `CDIR`, que relacionam o *sintagma nominal* com o *verbo* e o *verbo* com o *complemento directo*, respectivamente.

As relações de dependência são definidas por um conjunto de regras que tomam como entrada um conjunto de nós ou dependências previamente calculadas. Os nós podem ser criados pelo módulo de agrupamento ou por outros módulos de agrupamento externos. As regras de dependência são aplicadas sequencialmente e baseiam-se na informação armazenada na árvore de nós e no conjunto de dependências.

Considere-se a seguinte frase: *O coelho comeu a cenoura*. Duas dependências, entre outras, podem ser encontradas neste exemplo:

```
SUBJ(comeu,coelho)
CDIR(comeu,cenoura)
```

No XIP, os nós lexicais representam uma única leitura lexical de uma unidade linguística. Cada nó tem traços associados. Um traço expressa uma propriedade de um nó, como por exemplo o género (masculino ou feminino), ou o número (singular ou plural). Um traço consiste num par nome-valor. Assim, os traços associados a nós, permitem especificar categorias. O exemplo seguinte ilustra os traços associados à palavra *borracha*, correspondendo `fem` e `sg` a feminino e singular, respectivemante:

```
NOUN[fem:+,sg:+]
```

3.2.3 Gramática XIP

Uma gramática XIP consiste num conjunto de ficheiros de texto, que contêm regras que permitem fazer a desambiguação, o *chunking* e encontrar relações de dependência num texto.

A gramática é composta pelos seguintes tipos de ficheiros:

- Declarações das etiquetas usadas para descrever traços, categorias e dependências nas regras do XIP;
- Diferentes tipos de regras que recorrem a operadores e expressões regulares para testar os traços de um nó;
- Um ficheiro de configuração onde se encontram declarados todos os ficheiros constituintes da gramática.

As regras podem especificar uma determinada expressão, um contexto, ou restrições que são aplicadas aos nós. As regras são utilizadas para desambiguar as várias leituras possíveis associadas a um nó, para construir uma árvore de nós e para criar dependências entre os nós dessas mesmas árvores.

O XIP suporta três tipos de regras: Regras de Dominância Imediata (*Immediate Dominance Rules*), Regras de Sequência (*Sequence Rules*) e Regras de Dependência (*Dependency Rules*). Tanto as regras de dominância imediata como as de sequência são regras de agrupamento. Ao nível da sintaxe, a única diferença entre estes dois tipos de regras é o operador de atribuição. No caso das regras de dominância imdediata este operador consiste numa seta `->` enquanto que nas regras de sequência é designado pelo sinal `=`:

Regra de Dominância Imediata:

```
REGRA 1: NP -> DET, (PRON), NOUN[masc].
```

Regra de Sequência:

```
REGRA 2: NP = DET, (PRON), NOUN[masc].
```

Ao nível semântico estas regras são interpretadas de forma diferente. Ambas as regras criam um nó `NP` ao detectarem um determinante, seguido de um nome do género masculino, podendo ainda existir um pronome entre os dois. No entanto, as regras de dominância imediata, ao contrário das regras de sequência, são aplicadas independentemente da ordem pela qual os nós surgem no lado direito da regra. No caso das regras de sequência, é estritamente necessário que os nós no texto de entrada surjam

exactamente pela ordem na qual se econtram no lado direito das regras, para que estas possam ser aplicadas. No caso de ser possível aplicar várias regras de dominância imediata, o factor de escolha é baseado na maior sequência possível (da direita para a esquerda). No caso das regras de sequência, estas são aplicadas sequencialmente, pela ordem definida pelo programador e o texto de entrada é lido da esquerda para a direita.

As regras de dependência, são úteis na criação de dependência entre entidades, como por exemplo entre o *sujeito* e o *verbo*, ou podem ainda ser utilizadas para adicionar ou remover traços a um nó.

Criação de uma dependência:

```
REGRA 3: |NP{?*, #1[last]}, VP{?*, #2[last]}|   Subj(#2, #1).
```

Adição de um traço a um nó:

```
REGRA 4: |NP[mass=+] {(?), num,
                      (?[lemma:de]), ?[mass, meas], (num)}| ~
```

A regra 3 cria uma dependência `Subj` entre o último segmento de um nó `NP` e o último segmento de um nó `VP`, sendo o `NP` e `VP` nós adjacentes.

A regra 4 acrescenta um traço `mass` ao nó `NP` caso este obedeça à estrutura especificada pela regra, ou seja, o `NP` deve ser constituído pelas seguintes entidades: uma entidade não especificada seguida de um número, uma entidade opcional com o lema *de*, uma entidade que contenha os traços `mass` e `meas` e, por último, uma outra entidade numérica, também opcional.

3.3 Processo de Reconhecimento

Nesta secção encontra-se uma descrição dos procedimentos adoptados para efectuar o reconhecimento das entidades referentes às várias categorias e subcategorias.

Ao longo deste capítulo, diversas regras acompanham a descrição do processo de reconhecimento de entidades, sendo que estas são apenas alguns exemplos (por vezes simplificados), extraídos de um total de cerca de 400 regras, que têm como função ilustrar os vários procedimentos.

Cada categoria de entidades é caracterizada por um traço específico. Depois, conforme os tipos de cada categoria, as entidades podem ser classificadas com outros traços.

Durante o processo de implementação, recorreu-se várias vezes ao AC/DC (*AC/DC – Acesso a corpora, disponibilização de corpora*, n.d.) da Linguateca como ferramenta auxiliar, permitindo o acesso a

corpora [1] anotada do Público, de forma a melhorar o desempenho do sistema ao nível do Reconhecimento de Entidades Mencionadas.

3.3.1 Obra

O reconhecimento de entidades mencionadas referentes à categoria Obra, é efectuado recorrendo a uma gramática com regras de sequência específicas para este fim. Estas entidades são caracterizadas pelo traço `culture`.

3.3.1.1 Produto

Para auxiliar o reconhecimento de obras do tipo Produto, criou-se um léxico com nomes de diversos tipos de marcas (e.g., automóveis, computadores, telemóveis). Estes nomes são identificados pelo traço `brand`.

Um produto é definido pelo nome de uma marca seguido de um número (do tipo dígito), ou de uma palavra cuja primeira letra é maiúscula, ou ainda de combinações entre estes dois últimos tipos de entidades.

Os produtos são então identificados por regras que detectam nomes de marcas seguidos de diversas entidades. Por exemplo, a regra 5 identifica nomes de produtos cujo nome da marca antecede um número do tipo dígito (e.g., Nokia 6300), atribuindo-lhes os traços `culture` e `product`.

```
REGRA 5: noun[culture=+, product=+] = ?[brand], num[dig].
```

3.3.1.2 Reproduzida

Para se proceder ao reconhecimento de obras do tipo Reproduzida implementou-se regras de sequência com contexto associado. Considere-se, como exemplo, a seguinte regra:

```
REGRA 6: NOUN[example=+] = |ART[lemma:o], ADJ| NOUN.
```

A regra de sequência 6 tem associado um contexto à esquerda. O contexto é um sequência de nós que é definida entre duas barras verticais (e.g., |*contexto*|) que tanto se pode encontrar à esquerda como à direita da regra, ou simultaneamente à esquerda e direita. Esta regra apenas é aplicada se o XIP detectar na entrada uma entidade `NOUN` antecedida de um contexto que consiste de um artigo com

[1] *Corpora* é uma colecção de dados seleccionados e organizados segundo critérios linguísticos explícitos para cumprir determinadas funções (Andrade, 2003)

o lema *o*, seguido de um adjectivo. Note-se que o contexto não faz parte do nó NOUN criado após a execução da regra.

Após analisar vários títulos de obras, concluiu-se que existe um indeterminado número de padrões possíveis para a sua detecção. Isto é, existem títulos que podem ser compostos apenas por números (e.g., 300), ou títulos cujo nome pode conter símbolos de pontuação (e.g., George - O Rei da Selva, Mandela: Meu Prisioneiro, Meu Amigo). Também não é garantido que os vários nomes que compõem o título de uma obra, comecem todos por maiúsculas.

Portanto, em algumas situações, por uma questão de precisão, assume-se que os nomes ou títulos de obras são sequências de caracteres delimitados por aspas, cuja primeira palavra começa por uma letra maiúscula (e.g., ≪*Nome*≫). De acordo com esta definição uma uma das regras implementadas para a identificação de obras do tipo Reproduzida é:

```
REGRA 7: NOUN[culture=+,inquote=+] = |PUNCT[bracket,left]|
                                     ?[maj], ?*[bracket:~]
                                     |PUNCT[bracket,right]
                                     ?[lemma:ser];PUNCT[comma],
                                     ?[lemma:escrever]|.
```

A regra 7 identifica obras escritas cujo nome ou título antecede um contexto que consiste de umas aspas à direita, seguidas de vírgula ou uma palavra com o lema *ser* e de uma outra palavra cujo o lema é *escrever*. O nome ou título da obra tem que ser também precedido por umas aspas à esquerda. Por outras palavras, a regra identifica nomes ou títulos que surgem entre aspas e que antecedem expressões como *foi escrito por* ou *, escrito por*. As obras do tipo Reproduzido têm associado o traço inquote. Vários outras regras semelhantes contemplam o reconhecimento de títulos de obras inseridos noutros contextos (e.g., escreveu autor em *Título*, *Título* o mais recente filme, etc.).

A regra 8 efectua também o reconhecimento de obras do tipo Reproduzida, sendo o título da obra um padrão definido por uma sequência de nós, seguido do ano de edição (um número do tipo dígito) que se encontra entre parêntesis (e.g., Braveheart(1995)).

```
REGRA 8: NOUN[culture=+, inquote=+] =
                    ?+[maj], (prep, (art), ?+[maj])
                    |punct[left, paren], num[dig], punct[right, paren]|.
```

3.3.1.3 Arte

O reconhecimento de obras do tipo Arte é feito com base em regras de sequência, atribuindo-lhes o traço `monument`. Tome-se como exemplo a regra 9 que faz o reconhecimento de igrejas:

```
REGRA 9: NOUN[culture=+, monument=+] = ?[lemma:igreja],
                                       (?[lemma:paroquial]),
                                       (prep[lemma:de], (?[lemma:o])),
                                       ?+[maj], (prep, (art), ?+[maj])*.
```

A regra 9 reconhece uma palavra com o lema *igreja*, seguida de uma palavra opcional, com o lema *paroquial*, seguida de uma possível preposição *de* e de uma palavra com o lema o, seguidos de uma sequência de nós que representam o nome da igreja. Outras regras semelhantes fazem o reconhecimento de mais obras, como por exemplo palácios e castelos, entre outros.

Dado que algumas destas entidades podem ser consideradas simultaneamente obras e locais, e que está a ser desenvolvido um outro trabalho (Romão, 2007) (no L^2F), no âmbito do Reconhecimento de Entidades Mencionadas, que abrange outras categorias (e.g, Local), foram definidas regras de dependência que adicionam os traços `culture` e `monument` a entidades do tipo Arte que se encontrem classificadas como locais.

```
REGRA 10: |NOUN[location:+, culture=+, monument=+]
                                        {?[lemma:mosteiro], ?*}| ~
```

Por exemplo, a regra 10 atribui os traços característicos das obras do tipo Arte, a nomes de mosteiros que contenham o traço `location`.

Alguns nomes de obras específicas do tipo Arte são também reconhecidos por regras de sequência, caso sejam nomes compostos (e.g., Cristo Redentor), ou pelo léxico no caso de nomes simples (e.g., Petra).

3.3.1.4 Publicação

As entidades do tipo Publicação identificadas são Decretos-Lei (e.g., Decreto Lei 234/94, Decreto-Lei nº454/99 de Junho de 1999) e obras referidas pelo ano de edição (e.g., Camões(1554)). Estas entidades contêm o traço `publication`.

O reconhecimento de entidades que referem Decretos-Lei é feito através de regras de dependência, como por exemplo:

```
REGRA 11: NOUN[culture=+, publication=+] = ?[lemma:decreto-lei],
                                           (?[surface:n], ?[surface:º]),
                                           NUM[dig],
                                           SYMBOL[slash],
                                           NUM[dig].
```

A regra 11 detecta Decretos-Lei do tipo *Decreto-Lei 234/94*. O uso do traço `surface` permite especificar que determinadas expressões só são reconhecidas caso surjam escritas tal e qual como se encontram definidas na regra. O reconhecimento de outros padrões, como por exemplo *Decreto-Lei nº454/99 de Junho de 1999*, está a cargo de outras regras semelhantes.

3.3.2 Valor

O reconhecimento de números é essencial, uma vez que são parte integrante de outras entidades mencionadas, como por exemplo, o tempo (e.g., 5 dias, ano 1999, 22 horas), valores monetários (e.g., $20, 50 euros), ou quantidades (e.g., um quarto de litro). Os números são actualmente classificados no XIP segundo as subcategorias:

Números Cardinais: dez, cento e vinte e três;

Números Cardinais Dígitos: 10, 123;

Números Cardinais Fraccionários: 22,5;

Números Ordinais: 1º, primeiro;

Números Fraccionários: dois quintos;

Números Incertos (*Fuzzy*): vinte e muitos, vinte e pouco;

Números Romanos: XXI, DC.

Os números romanos são identificados no módulo de Segmentação (da cadeia de processamento), através de um *script* PERL (Practical Extraction and Report Language) que faz o seu reconhecimento com base em expressões regulares. Note-se que, no caso de ser necessário considerar o contexto de uma expressão para identificar a numeração romana, estes números surgem, geralmente, associados a nomes (e.g., século XXI, João VI de Portugal).

As entidades da categoria Valor são caracterizadas pelo traço `quant`.

3.3.2.1 Classificação

Para o reconhecimento de valores do tipo Classificação recorreu-se a regras de sequência com contexto associado. O contexto aplica-se no caso de classificações (e.g., [ficou em] 2º [lugar]), resultados (e.g., [ganhou por] 2-0), tempos de classificação (e.g., [aos] 18'), graus académicos (e.g., [frequentou o] 12º [ano]). Estes exemplos são reconhecidos pelas regras 12, 13, 14 e 15, respectivamente.

```
REGRA 12: ?[quant=+, classific=+] = |?[lemma:ficar], ?[lemma:em]|
                                    ?
                                    |?[lemma:lugar]|.

REGRA 13: ?[quant=+, classific=+] = |?[lemma:ganhar], (?[lemma:por])|
                                    num[dig], punct[dash], num[dig].

REGRA 14: NOUN[quant=+, classific=+] = |?[lemma:a], ?[lemma:o]|
                                       num[dig], symbol[quote].

REGRA 15: NOUN[quant=+, classific=+] = |?[lemma:frequentar], ?[lemma:o]|
                                       ?
                                       |?[lemma:ano]|.
```

Note-se que o ponto de interrogação (`?`) que refere a entidade a ser reconhecida nas regras 12 e 15, designa um número ordinal (e.g., 14º). Isto acontece uma vez que não é possível referir explicitamente este tipo de entidades numéricas, dado que correspondem a único segmento.

Como se pode verificar, este tipo de entidades são marcadas pelo traço `classific`.

3.3.2.2 Quantidade

As quantidades são entidades compostas por um nome, que representa uma unidade de medida, e uma entidade numérica que permite quantificar essa unidade de medida (e.g., 10 quilos, vinte litros). O XIP identifica unidades de medida e classifica-as segundo um tipo específico. Dos tipos de unidades identificados pelo XIP constam, por exemplo, as unidades de volume (e.g., quilolitro, litro, mililitro), unidades de comprimento (e.g., quilómetro, metro, milímetro), unidades de tempo (e.g., ano, dia, segundo) e unidades de massa (e.g., quilograma, grama, miligrama).

Foi efectuada uma pesquisa no sentido de complementar o léxico com unidades do Sistema Internacional. A tabela 3.1 contém as 28 unidades de medida actualmente identificadas pelo XIP:

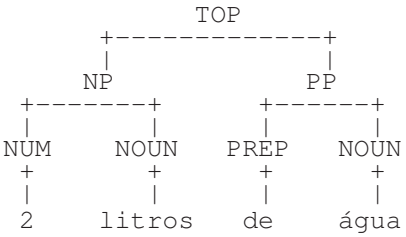
Unidade	Traço	Exemplo
Moeda	curr	Euro (€)
Tempo	time	Segundo (s)
Comprimento	length	Metro (m)
Massa	mass	Quilo (Kg)
Volume	volume	Litro (l)
Área	area	Metros Quadrados (m^2)
Corrente Eléctrica	electcurrent	Ampère (A)
Carga Eléctrica	electcurrent	Coulomb (C)
Capacitância Eléctrica	electcapacitance	Farad (F)
Resistência Eléctrica	electresistance	Ohm (Ω)
Condutância Eléctrica	electconductance	Siemens (S)
Tensão Eléctrica	electpotencial	Volt (V)
Pressão	pressure	Pascal (Pa)
Energia	energy	Joule (J)
Temperatura	temperature	Kelvin (K)
Intensidade Luminosa	luminointensity	Candela (cd)
Fluxo Luminoso	luminoflux	Lúmen (lm)
Iluminânca	illuminance	Lux (lx)
Substância	substance	Mole (mol)
Força	force	Newton (N)
Velocidade	speed	Rotações por Minuto (rpm)
Potência	power	Watt (W)
Ângulo	angle	Radianos (rad)
Armazenamento de Informação	infostorage	Byte (B)
Densidade	density	Quilograma por Metro Cúbico (kg/m^3)
Taxa de Fluxo Volumétrico	flowrate	Metro Cúbico por Segundo (m^3/s)
Inércia	inercia	Quilograma Metro Quadrado ($kg.m^2$)
Quilograma-Força	massforce	Quilograma Força (kgf)

Tabela 3.1: Unidades de medida identificadas pelo XIP.

As quantidades são reconhecidas no módulo de Agrupamento do XIP como sintagmas nominais (*noun phrase*). A regra 16 agrupa uma entidade numérica seguida de uma unidade de medida, entre as quais poderá existir uma preposição *de*.

```
REGRA 16: NP[quant=+] = num, (prep[lemma:de]), noun[meas].
```

O aplicação da regra 16 à expressão *2 litros de água* resulta na seguinte árvore:



Posteriormente, recorrendo a regras de dependência, os sintagmas criados são restruturados, sendo-lhes adicionado um traço de acordo com o tipo da unidade de medida em questão. A regra 17 adiciona o traço `length` ao sintagma nominal quando este representa uma quantidade cuja unidade

de medida é do tipo comprimento (*length*).

```
REGRA 17: |NP[length=+] {(?), num, (?[lemma:de]),
                         ?[length, meas], (num)}| ~
```

No caso das quantidades cuja unidade não reflecte uma propriedade, apenas o valor numérico deve fazer parte da entidade mencionada (e.g., 20 [carros]). Nesta situação, apenas a entidade numérica possui o traço `quant`. Para além deste traço, estas quantidades genéricas são também identificadas pelo traço `generic`. A atribuição destes traços é conseguida através de regras de dependência.

Se se considerar agora a expressão *10 metros e 20 centímetros*, constata-se que a expressão, na sua totalidade, poderia representar uma única entidade do tipo Quantidade, embora composta por duas sub-quantidades. Mas, como nem sempre é garantido que as várias unidades presentes na expressão (e.g., metro, centímetro) pertençam ao mesmo tipo, optou-se por considerar as sub-quantidades como entidades distintas, o que não está errado de todo, uma vez que o valor da quantidade associada à expressão *10 metros e 20 centímetros* não é mais que o resultado da soma das sub-quantidades *10 metros* e *20 centímetros*.

Contudo, são ainda estabelecidas várias relações (dependências) entre as várias funções sintácticas de uma frase. Na frase *A Maria comprou 10 metros e 20 centímetros de tecido.* estão presentes, entre outras, relações de quantificação, relações de coordenação e relações de complemento directo:

```
QUANTD(metros,10)
QUANTD(centimetros,20)
COORD(e,metros)
COORD(e,centímetros)
CDIR_POST(comprou,metros)
CDIR_POST(comprou,centímetros)
```

A dependência `QUANTD` permite quantificar as unidades de medida, relacionando-as com os respectivos valores numéricos. A dependência `COORD` relaciona as unidades de medida que se encontram ligadas por uma conjunção. As unidades de medida, que se encontram na função de complemento directo, são relacionadas com o verbo através da dependência `CDIR_POST`.

3.3.2.3 Moeda

São vários os formatos em que podem surgir os valores monetários, como por exemplo: *20 euros*, *20$*, *$20*, *20 euros e 50*, *20 euros e 50 cêntimos*, *20 milhões de euros*. Repare-se que estes valores monetários

são quantidades que resultam da associação de um número a um nome que representa a unidade monetária. Para se proceder ao reconhecimento de tais quantidades é necessário, antes de mais, efectuar o reconhecimento das várias moedas. Os nomes das moedas podem ser considerados segundo o número de palavras que os constitui: nomes simples e nomes compostos. Os nomes simples são constituídos por uma palavra apenas (e.g., escudo, libra, dólar). Por sua vez, os nomes compostos referem-se a moedas cujo nome é composto por várias palavras (e.g., escudo de cabo verde, libra libanesa, dólar americano).

Os nomes simples são identificados através de um léxico de moedas contendo esses mesmos nomes. No entanto, se apenas se adicionar estes nomes ao léxico das moedas, vão ocorrer situações, como por exemplo o adjectivo *marroquino* ser reconhecido como uma forma do verbo "marroquinar", ou a moeda *dirham* ser classificada como uma forma do verbo "dirhar". Isto deve-se ao facto de o Palavroso tentar "adivinhar" a classificação adequada à palavra em questão quando não há informação presente. Portanto, estas palavras encontram-se também definidas no léxico do Palavroso, permitindo uma correcta atribuição das categorias aos vários nomes de moedas.

Isto evita também o tipo de situação em que o lema da palavra escrita em minúsculas difere do lema da mesma quando escrita com maiúsculas. Considere-se então a moeda *dólar das Caraíbas Orientais*. Se a palavra *Caraíbas* não estivesse correctamente lexicalizada no XIP, o seu lema iria ser diferente consoante a palavra surgisse escrita com minúsculas (e.g., caraíba) ou com maiúsculas (e.g., Caraíbas).

Relativamente aos nomes compostos, a sua identificação é efectuada mediante regras de sequência. Tome-se como exemplo a regra 18:

```
REGRA 18: NOUN[curr=+, meas=+] = ?[lemma:dólar], ADJ[masc].
```

A regra de sequência 18, ao identificar na entrada do XIP, uma entidade com o lema *dólar* (independentemente da categoria), seguida de um adjectivo do género masculino, agrupa-as, classificando essa nova entidade como um `NOUN` com os traços `curr` e `meas`. Nomes de moedas, tais como *dólar canadiano* ou *dólar americano*, são reconhecidos por esta regra.

Se se considerar a estrutura de nomes de moedas como *escudo de Cabo Verde*, pode ser definido um outro tipo de regras gerais. Note-se que, este tipo de estrutura corresponde ao nome (simples) de uma moeda (*escudo*), seguido de uma preposição (*de*) e do nome de uma localidade (*Cabo Verde*). Assim, podem-se definir regras do tipo:

```
REGRA 19: NOUN[curr=+, meas=+] = ?[lemma:escudo],
                                 ?[lemma:de],
                                 NOUN[location].
```

A regra 19 permite reconhecer nomes de moedas como *escudo de Cabo Verde*.

Contudo, a generalidade destas regras, pode eventualmente permitir que expressões, tais como *dólar português* ou *escudo de Itália*, fossem reconhecidas como nomes referentes a moedas válidas, quando não o são. Logo, dado que existe um número limitado e conhecido de moedas cujo nome se enquadra na categoria de nomes compostos, é preferível adoptar uma segunda alternativa, que consiste na definição de regras individuais para cada um destes nomes. Assim, garante-se uma maior eficiência e que apenas são reconhecidos nomes de moedas que realmente existem. Exemplo disso é regra 20 que serve, exclusivamente, para identificar a moeda *coroa dinamarquesa*.

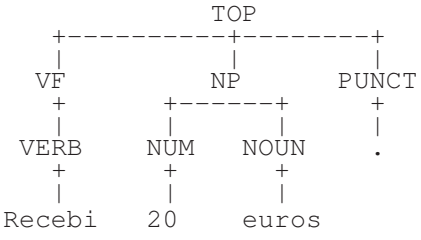
```
REGRA 20: NOUN[curr=+, meas=+] = ?[lemma:coroa],
                                 ADJ[lemma:dinamarquês].
```

A regra de sequência 20, ao identificar na entrada do XIP uma entidade com o lema *coroa*, seguida de um adjectivo com o lema *dinamarquês*, agrupa-as, classificando essa nova entidade como um `NOUN` com os traços `curr` e `meas`.

Após o reconhecimento das moedas, é então possível reconhecer as quantidades monetárias. Tome-se como exemplo a seguinte regra:

```
REGRA 21: NP[curr=+, quant=+] = NUM; ?[lemma:um],
                                ?[curr, meas].
```

A regra de sequência 21, ao identificar na entrada do XIP uma entidade da categoria `NUM`, ou uma palavra com o lema *um*, seguida de uma outra entidade com os traços `curr` e `meas`, agrupa-as numa nova entidade, atribuindo-lhe a categoria `NP` e os traços `curr` e `quant`. A aplicação desta regra à frase *Recebi 20 euros.* tem como resultado a seguinte árvore sintáctica:



Outras regras efectuam o reconhecimento de outros padrões de valores monetários, como por exemplo, *2 euros e 50 cêntimos*. No caso de ser omitida a moeda (e.g., 2,50), o valor é identificado por regras de contexto, como por exemplo a regra 22, que reconhece um número como sendo um valor monetário, caso este seja precedido por uma forma do verbo *custar*.

REGRA 22: `?[curr=+, quant=+] = |verb[lemma:custar]| num.`

3.3.3 Relações de Parentesco

O reconhecimento das relações de parentesco é feito, em grande parte, através do léxico. Criou-se também uma gramática cujo objectivo é o reconhecimento de relações expressas através de nomes compostos (e.g., tio avô, irmão gémeo). Esses nomes são assim reconhecidos através de regras de sequência, como é o caso da regra 23, que identifica uma palavra com o lema *irmão*, seguida de outra palavra com o lema *gémeo* e agrupa-as num único nó, atribuindo-lhe a categoria `NOUN` e o traço `relative`.

REGRA 23: `NOUN[relative=+] = ?[lemma:irmão], ?[lemma:gémeo].`

No caso de o nome desta relação surgir com um hífen (e.g., meio-irmão), é considerado um único segmento, sendo portanto reconhecido pelo léxico.

3.3.4 Tempo

Para o reconhecimento de entidades temporais começaram por ser introduzidas no léxico as várias unidades de medida de tempo, tais como *segundo*, *hora*, *dia*, *semestre*, *ano*, etc. Foram também introduzidos os meses do ano (e.g., Janeiro) e dias da semana (e.g., segunda-feira), assim como as respectivas abreviaturas (e.g., Jan, seg.). Por uma questão de coerência, as abreviaturas possuem o mesmo lema que as respectivas palavras, permitindo que tenham comportamentos idênticos quando aplicadas em regras.

De modo a facilitar uma posterior normalização do tempo, foi definido o traço `month` para os meses do ano, que define um conjunto de possíveis valores (e.g., 1, 2, 3, ... 12) que representam os vários meses do ano. Segue-se um exemplo da definição do mês de Janeiro no léxico relativo ao tempo:

```
janeiro: NOUN +=[month=1, time=+].
```

Este exemplo mostra a adição dos traços `month` e `time` e os respectivos valores ao nome *janeiro*.

Estas entidades podem também aparecer sob a forma de nomes compostos (e.g., 2ª feira, segunda feira), tornando-se mais uma vez necessário recorrer a regras de sequência para efectuar o seu reconhecimento.

3.3.4.1 Hora

No que diz respeito ao reconhecimento de entidades temporais do tipo Hora, este é realizado por um conjunto de regras de sequência, que abrangem as várias formas que as entidades deste tipo podem

assumir (e.g., 2 horas e 20 minutos, 2:20, um quatro para as três da tarde, etc.).

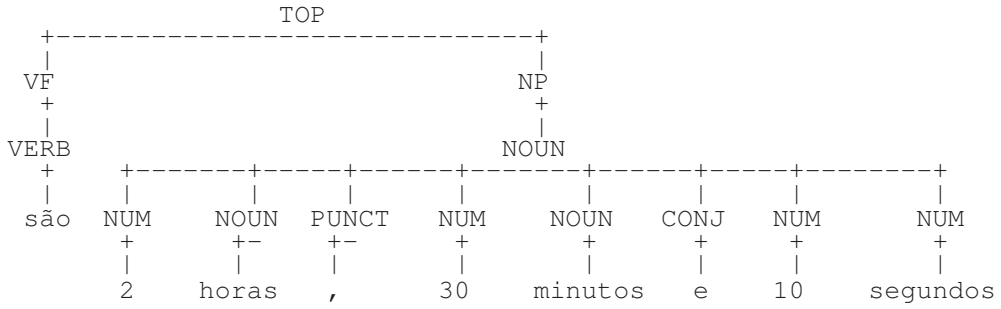
Considere-se, por exemplo, a regra 24 que faz o reconhecimento de expressões do tipo *2 horas*, que consistem de um número seguido de uma unidade de tempo:

```
REGRA 24: NOUN[hour=+, time=+] = |prep[lemma:a], art[lemma:o]|
                                 NUM[frac:~, ord:~],
                                 ?[lemma:hora].
```

Note-se que a expressão *2 horas*, conforme o contexto em que surge, tanto pode fazer referência a uma hora específica como a um período de tempo ou duração. Torna-se, portanto, necessário associar um contexto a esta regra, de forma a permitir distinguir as diferentes situações.

Neste exemplo particular (regra 24), o contexto refere-se à expressão *à* ou *às* que normalmente antecede uma hora específica. Existem outras expressões que permitem contextualizar um entidade temporal do tipo Hora, como por exemplo formas do verbo *ser* (e.g., são 2 horas da tarde) ou a expressão *por volta de* (e.g., por volta das 14 horas), que são abrangidas por outras regras semelhantes.

Expressões mais extensas como *2 horas, 30 minutos e 10 segundos* são identificadas como um todo, através de um único nó `NOUN` que abrange toda a expressão:



3.3.4.2 Data

Relativamente às entidades de tempo do tipo Data, o processo de reconhecimento foi orientado de forma a facilitar a normalização de expressões. Para tal, optou-se por repartir a data em várias componentes: dia da semana, dia do mês, mês e ano. O reconhecimento destas várias componentes é feito em paralelo através de regras de sequência.

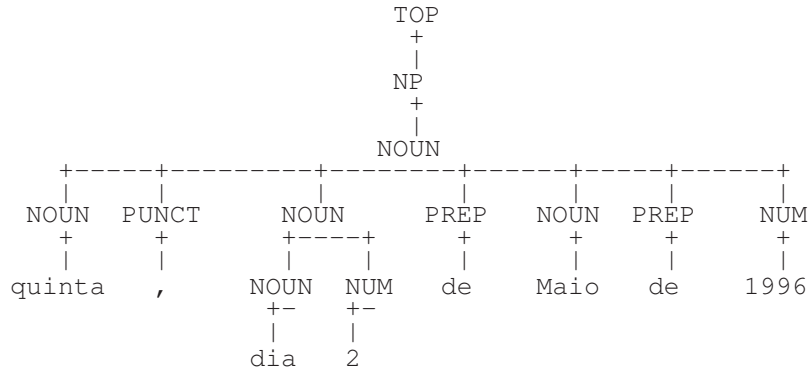
Procedeu-se à inserção de várias palavras, que complementam as datas, no léxico do tempo, às quais foi atribuído um traço `timeref`, permitindo suportar as diversas alternâncias das várias componentes da data (e.g., próximo dia 2, passado mês).

A agregação destas palavras nos nós resultantes do reconhecimento das várias componentes de uma data, é uma estratégia que visa facilitar a normalização de expressões temporais, nomeadamente datas relativas (e.g., próximo dia 2), permitindo ancorar tais expressões no tempo. A palavra *próximo* na expressão *próximo dia 2* permite atribuir um valor temporal concreto a essa expressão relativamente a um tempo ou data de referência, como por exemplo a data de discurso. Todas as componentes de uma data que contenham palavras caracterizadas pelo traço `timeref` são classificadas com o traço `uncertain`, indicando que se tratam de expressões relativas ou incertas.

As várias formas de expressão que permitem exprimir componentes de uma data (e.g., dia 2, próximo dia, mês de Maio, ano anterior) foram convertidas em regras de sequência, no sentido de efectuar o seu reconhecimento. Exemplo disso é a regra 25 que atribui os traços `time` e `monthday` a um número inteiro caso este anteceda uma expressão composta por uma palavra com o lema *de* seguida de um mês do ano (e.g., 2 [de Janeiro]).

```
REGRA 25: ?[time=+, monthday=+] = num[dig];num[card, frac:~]
                                  |?[surface:de], ?[month]|.
```

Após a fase de reconhecimento das várias componentes, estas são concatenadas, por várias regras de sequência (que contemplam as várias combinações entre as várias componentes de uma data), numa nova entidade que representa a data final. A árvore seguinte ilustra o reconhecimento da data *quinta-feira, dia 2 de Maio de 1996*:



O nó `NOUN` superior abrange toda a data e é identificado pelo traço `date`. Os sub-nós que se referem às várias componentes da data são previamente reconhecidos e classificados com os respectivos traços: ao nó que corresponde ao dia da semana (e.g., *quinta*) é atribuído o traço `weekday`; ao nó que abrange a palavra *dia* e o valor numérico 2 é atribuído o traço `monthday`; ao nó que representa o mês (e.g., Maio) é atribuído o traço `month`; ao nó que corresponde ao valor numérico que representa o ano (e.g., *1996*), é atribuído o traço `year`.

3.3.4.3 Duração

Uma duração reflecte um intervalo ou período de tempo (e.g., 2 dias, 3 horas e meia). Na perspectiva deste trabalho, uma duração não é mais que uma quantidade de tempo. Assim, tal com as entidades do tipo Quantidade, da categoria Valor, as durações não são mais que sintagmas nominais com o traço `quant`, que agregam um valor numérico e uma medida de tempo. O traço que permite caracterizar a quantidade como sendo do tipo tempo (`time`) é posteriormente adicionado através de regras de dependência, como por exemplo a regra 26.

```
REGRA 26: |NP[length=+] {(?), num, (?[lemma:de]),
                        ?[length, meas], (num)}| ~
```

No entanto, se se considerar a expressão de *5 a 6 de Maio*, verifica-se que também esta denota um período de tempo. Este período de tempo é delimitado por duas datas (5 de Maio e 6 de Maio), sendo reconhecido pela regra 27.

```
REGRA 27: noun[time=+, duration=+] = prep[lemma:de], ?[time, date],
                        ?[surface:a], ?[time, date].
```

Uma outra regra semelhante reconhece durações, também delimitadas por duas datas, da forma *entre 5 e 6 de Maio*.

Note-se que estas durações são classificadas com o traço `duration` em vez de `quant` de modo a poderem ser diferenciadas das durações que são sintagmas nominais.

3.3.4.4 Frequência

Expressões como *2 vezes ao dia* ou *de 2 em 2 semanas* designam entidades temporais do tipo Frequência, em que há um período de tempo que se repete.

O reconhecimento destas entidades é efectuado recorrendo a regras de sequência, como por exemplo a regra 28, que identifica frequências do tipo *2 vezes por dia*.

```
REGRA 28: NOUN[time=+, frequency=+] = NUM[dig];num[card, frac:~],
                                      ?[lemma:vez],
                                      ?[lemma:por],
                                      ?[time, meas].
```

A regra 28 identifica uma expressão como sendo do tipo Frequência (atribuindo-lhe os traços `time` e `frequency`) caso detecte um número dígito ou cardinal não-fraccionário, seguido das palavras com os lemas *vez* e *por* e uma unidade de tempo (e.g., dia).

4 Normalização de Expressões Temporais

Neste capítulo é feita uma descrição da arquitectura XIP-Python que permite a integração de funções Python no XIP e refere ainda a forma como esta arquitectura permite a normalização de expressões temporais.

4.1 Arquitectura XIP-Python

O XIP (analisador sintáctico descrito no capítulo 3) disponibiliza uma interface que permite executar funções Python [1].

Para se usar um programa Python via XIP (Xerox, 2001) é necessário declarar uma interface específica entre o XIP e o Python. Esta interface descreve os nomes e argumentos das funções Python definidas no programa. A declaração dos nomes das funções deve ser aberta e fechada por uma sequência de três aspas e devem ser antecedidos da etiqueta `Python`:

```
"""
Python:
clean_dictionary(),
write_to_file(string value)
"""
```

Após a inclusão do ficheiro do programa em Python na lista de ficheiros a serem carregados pelo XIP, a invocação de funções Python pode ser feita através de um qualquer ficheiro da gramática desde que seja antecedida pela etiqueta `Script`. As funções Python podem também ser invocadas mediante regras:

```
REGRA 29: |NOUN#1{num, ?[lemma:vez], ?, ?[time, meas]}|
        {
            normalize_freq(#1, @sentencenumber);
        }
```

[1] Python é uma linguagem de programação orientada a objectos.

A regra 29 permite especificar em que condições é disparada a função `normaliza_freq`, isto é, a função só é invocada caso o XIP detecte um nó da categoria `NOUN` composta por um número seguido de uma palavra com o lema *vez*, seguido de um outra qualquer entidade e de uma entidade que contenha os traços `time` e `meas`. O argumento `@sentencenumber` representa o número da frase na qual se encontra o nó em questão.

Estas regras possibilitam ainda a passagem de um nó como argumento de uma função. Os nós são designados pelo símbolo cardinal (`#`) seguido de um número. No caso da regra 29 o nó em questão é referenciado por `#1`.

Após um função Python receber um nó como argumento, este pode ser manipulado através dos métodos disponibilizados pela classe XipNode. Ao ser passado um nó como argumento, o que a função recebe não é mais que um número identificador do nó em questão. Com esse identificador é criada uma estrutura XipNode que é composta pelos seguintes campos:

```
[POS, surface, lemma, features, parent, daughter,
 last, next, previous, left, right, leftoff, rightoff]
```

Para aceder ao valor de cada um destes campos invoca-se o nome do respectivo campo sobre o nome da estrutura que representa o nó:

```
node = XipNode(node_id)
lemma = node.lemma
print 'O lema do nó é:', lemma
```

Considere-se a seguinte frase à entrada do XIP: *Viajo de transportes duas vezes por dia.* Supondo que este excerto de código corresponde à função `normaliza_freq` e que esta função é invocada de acordo com a regra 29, o resultado esperado é:

```
O lema do nó é: duas vezes por dia
```

4.2 Processo de Normalização

A normalização de expressões é conseguida recorrendo a funções em código Python.

Tal como foi referido na secção 4.1 as funções Python podem ser invocadas mediante regras, em qualquer ficheiro da gramática. Isto significa que a normalização das expressões temporais pode ser feita em simultâneo com o reconhecimento de entidades. Contudo, optou-se por realizar a normalização durante a execução das regras de dependência, após o reconhecimento de entidades. A vantagem desta

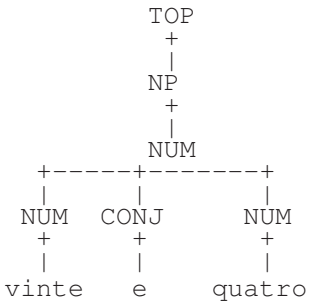
abordagem é a simplificação do código da gramática do XIP (responsável pela identificação de entidades da categoria Tempo), uma vez que a normalização e o reconhecimento são executados em fases distintas.

Para proceder à normalização de expressões com referências temporais tornou-se necessário implementar algumas funções auxiliares para manipular os nós.

De entre as várias funções auxiliares definidas destacam-se:

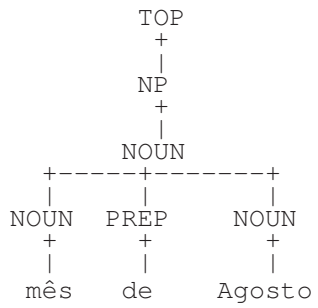
- a função `has_feature(features, feat)` que dados o conjunto de traços de um nó e um traço, indica se esse traço faz parte do nó em questão;
- a função `get_feature_val(features, feat)` que dados o conjunto de traços de um nó e um determinado traço, devolve o valor correspondente a esse traço;
- a função `month_norm(month)` que traduz o valor numérico correspondente a um mês do ano para a sua abreviatura correspondente em português (e.g., 4 → ABR);
- a função `ext2num(number)` que traduz números inteiros por extenso para o seu respectivo valor numérico;
- a função `search_node(#1, feat)` que dados o identificador de um nó e um determinado traço, devolve o próprio nó ou o primeiro sub-nó que apresentar com esse traço, que seja composto por uma palavra apenas;
- a função `search_num(#1)` que devolve o próprio nó ou o primeiro sub-nó que encontrar com o traço `NUM`.

À primeira vista poder-se-ia pensar que a implementação da função `search_num(#1)` seria desnecessária quando se tem uma outra função que permite procurar um nó com um determinado traço. No entanto, há uma pequena particularidade que obriga a esta necessidade. Considere-se o número *vinte e quatro* e a sua respectiva representação em forma de árvore:



Como se pode verificar uma entidade numérica pode ser constituída por outras entidades do mesmo tipo. Por esta razão, quando se procura um nó com o traço `NUM`, o nó que se quer é o que representa o número na sua totalidade, sendo este o nó `NUM` superior.

Considere-se agora a expressão *mês de Agosto* e a sua respectiva árvore:

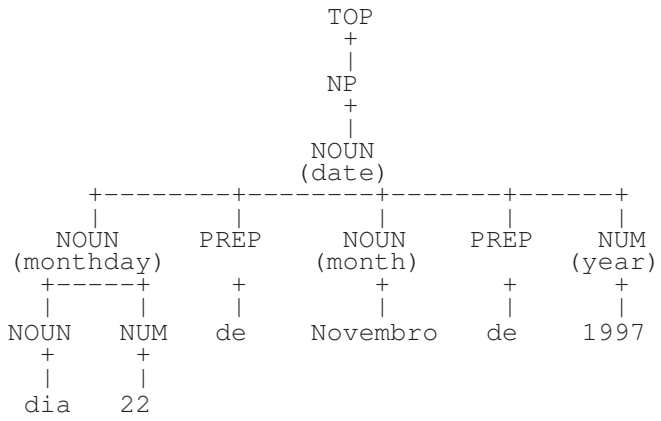


Nesta árvore ambos os nós `NOUN` possuem o traço `month`. No entanto, para o processo de normalização interessa apenas o nó `NOUN` inferior. Assim, a função `search_node(#1, feat)` tem como critério de selecção o primeiro nó ou sub-nó que possuir o traço em questão (traço `month` neste exemplo em particular) e se for constituído apenas por uma palavra. Note-se que a procura é sempre efectuada no sentido de cima para baixo e da esquerda para a direita.

Estas funções fazem parte de um conjunto de 13 funções auxiliares definidas conjuntamente com outras 5 funções principais, que perfazem um total de cerca de 700 linhas de código. As funções principais são:

- a função `normalize_date(#1, int sentence_id)` que recebe como argumentos um identificador de um nó e o número da frase, procedendo à normalização de datas (e.g., dia de Janeiro, próximo ano);
- a função `normalize_date2(#1, int sentence_id, string op)` que realiza a normalização de datas (e.g., daqui a 2 dias, há 3 dias atrás), recebendo como argumentos um identificador de um nó, o número da frase e um operador (+ ou −) que indica o tipo de operação que deve ser realizada para ancorar a data no tempo;
- a função `normalize_freq(#1, int sentence_id)` que recebe como argumentos um identificador de um nó e o número da frase, efectuando a normalização de entidades Tempo do tipo Frequência (e.g., 2 dias por semana);
- a função `normalize_freq2(#1, int sentence_id)` que recebe como argumentos um identificador de um nó e o número da frase, realizando a normalização de entidades Tempo do tipo Frequência (e.g., de 2 em 2 dias);
- a função `normalize_duration(#1, int sentence_id)` que recebe um identificador de um nó e o número da frase como argumentos e efectua a normalização de durações (e.g., 2 anos).

Para compreender o processo de normalização tome-se agora como exemplo a *data 22 de Novembro de 1997* e a sua representação em forma de árvore:



O processo de normalização tem início durante a execução das dependências. Aquando da detecção de uma data é invocada uma função responsável pela sua normalização: `normalize_date(#1, int sentence_id)`

Uma data é composta por várias componentes como descrito no capítulo 3. Neste exemplo em particular surge um nó superior com o traço `date` que abrange toda a data. Este nó é constituído por outros nós, entre os quais existe um nó com o traço `monthday` que representa o dia da semana (e.g., dia 22), um nó com o traço `month` que representa o mês (e.g., Novembro) e um nó que representa o ano (e.g., 1997).

A normalização da data é feita através da exploração dos nós da árvore. São pesquisados nós que contenham um dos traços: `weekday`, `monthday`, `month`, ou `year`.

No caso de ser identificado um nó contendo o traço `weekday` (e.g., Sábado), é extraído o valor associado ao traço `weekday` que contém o valor numérico referente ao dia da semana (e.g., 7). Este valor é posteriormente convertido para a correspondente abreviatura que designa o dia da semana em causa (e.g., SAB) e depois guardado numa variável.

Ao ser encontrado um nó com o traço `monthday` (e.g., dia 22), é efectuada uma nova exploração (com início neste nó), tentando-se extrair o valor numérico referente ao dia do mês (e.g., 22). Este valor é posteriormente guardado numa variável. No entanto, se o dia se encontrar escrito por extenso (ex: vinte e dois) o seu valor é previamente convertido para o respectivo valor numérico, recorrendo à função auxiliar `ext2num(int number)`.

O mês pode ser designado pelo seu próprio nome (e.g., Novembro) ou sob a forma de valor numérico (e.g.,11). Ao ser detectado um nó com o traço `month` (e.g., mês de Novembro), inicia-se uma

nova exploração, tentado-se encontrar o valor concreto do mês (e.g., Novembro). Caso o mês surja designado pelo nome, é pesquisado o valor do traço `month` que contém o valor numérico referente ao mês em questão (e.g., 11). Finalmente, este valor é convertido para a respectiva abreviatura em Português (e.g., NOV) e guardado numa variável.

Por último, o nó referente ao ano, ao qual pertence o traço `year`, recebe um tratamento semelhante ao do dia do mês, sendo o seu valor posteriormente guardado numa variável.

Após concluída a normalização da data, é necessário guardar o seu valor normalizado de forma a depois poder ser consultado para outros fins. Para tal, decidiu-se criar um dicionário de tempo.

O dicionário de tempo consiste num ficheiro que é criado cada vez que o XIP é executado. Este ficheiro tem entradas (linhas) quantas as expressões normalizadas de um determinado texto processado. Por cada entrada é apresentado o número da frase à qual pertence a expressão normalizada e o número do nó de topo referente a essa mesma expressão. Para além destes dois identificadores são ainda apresentados os valores resultantes da normalização de expressões, que variam consoante o tipo de expressão temporal em questão (e.g., data, hora, duração, frequência).

Uma possível [2] entrada no dicionário de tempo para a expressão *dia 22 de Novembro de 1997* é:

```
SENTENCE: 0, NODE: 14, VALUE: SAB-22-NOV-1997
```

Note-se que o valor correspondente ao dia da semana só surge no dicionário de tempo quando é explicitamente referido na expressão, ou quando pode ser calculado em função dos outros componentes da data – dia do mês, mês e ano. Para tal recorre-se aos métodos e calendário das bibliotecas *time* e *datetime* do Python. Contudo, não é possível calcular o dia da semana para datas cujo ano é inferior a 1900, devido ao facto do o calendário usado não abranger tais datas.

Da mesma forma que para os dias da semana, as outras componentes da data também só estão presentes no dicionário de tempo quando são directamente explicitadas na expressão ou quando o seu valor pode ser deduzido da expressão em causa.

Um possível entrada no dicionário de tempo para a expressão *Novembro de 1997* é:

```
SENTENCE: 0, NODE: 14, VALUE: NOV-1997
```

Uma outra alternativa, seria substituir por variáveis a informação que não está disponível:

```
SENTENCE: 0, NODE: 14, VALUE: XXX-XX-NOV-1997
```

[2] O número da frase e o número do nó podem variar conforme o texto onde se encontra a expressão.

Todas as expressões que referem datas imprecisas ou relativas encontram-se identificadas com o traço `uncertain`, visto requererem um tratamento diferente das que referem datas concretas. A normalização de expressões relativas (e.g., próxima semana) implica o conhecimento de uma data de referência a partir da qual é possível ancorar a expressão temporal no tempo. Podem ser considerados três tipos de tempo como referência: o tempo do discurso, o tempo do tópico e o tempo do documento.

Para além do tempo de referência, tornou-se necessário complementar o reconhecimento de diversas palavras ou expressões, que surgem no contexto das expressões temporais, com informação adicional. Certas palavras ou expressões como *anterior* ou *próximo* indicam o sentido (passado ou futuro, respectivamente) no qual deve ser calculado o tempo relativamente ao tempo de referência. Este tipo de palavras ou expressões é caracterizado pelo traço `timeref`. Este traço indica que se tratam de modificadores temporais e pode ter associados os valores `-`, `0` ou `+`, conforme a expressão refira passado, presente, ou futuro, respectivamente. Houve ainda necessidade lhes adicionar um outro traço – `timeval` – que indica o valor unitário associado a estas palavras ou expressões. Este traço é atribuído pelo XIP na fase do reconhecimento de entidades.

É nesta fase que se tira partido da repartição do reconhecimento de datas em componentes (exemplificado no capítulo 3). Como já foi demonstrado, a normalização de datas é feita componente a componente. Assim, quando surge um modificador temporal (e.g., próximo) é possível saber qual a unidade de tempo que lhe está associada. Tome-se como exemplo os traços e respectivos valores associados à palavra *próximo*:

```
[timeval:1,timeref:+,adv:+]
```

No caso concreto da normalização da expressão *no próximo dia*, estes traços indicam que para se ancorar a expressão no tempo, a data de referência deve ser incrementada no valor de 1 e a palavra *dia* indica a unidade do incremento.

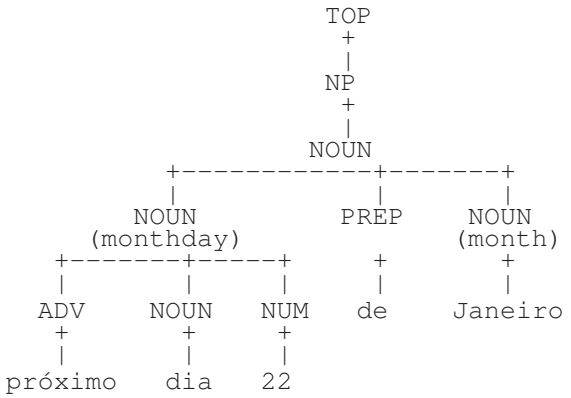
Estes cálculos são efectuados com o auxílio dos métodos disponibilizados pelas bibliotecas `time` e `datetime` do Python, tendo por base uma data de referência, os valores dos traços `timeref` e `timeval` e uma unidade de tempo – dia da semana, dia do mês, mês e ano.)

Da mesma forma é calculado o valor de expressões como *mês passado*, *próximo ano* ou *dia anterior*, em que o valor do `timeval` se mantém (com o valor 1), variando a unidade de tempo e o valor associado ao `timeref`.

Estes são os casos mais simples da normalização de expressões temporais relativas. Tome-se agora como exemplo a expressão *próximo dia 22*. Neste caso não basta incrementar 1 dia à data de referência. Tem que ser calculado o próximo dia que apresente o valor *22*, o que poderá dar a origem a um novo

mês e talvez um novo ano, caso o dia 22 do mês corrente (da data de referência) já tenha passado e esse mês seja o último do ano.

O processo de normalização torna-se ainda mais complicado se se considerar a expressão próximo dia *22 de Janeiro*. Este exemplo revela-se num problema para a normalização, tendo em conta a abordagem da decomposição da data em componentes. Assuma-se que a data de referência é 22 de Janeiro de 2007.



Analisando a expressão por partes, a normalização de *próximo dia 22* tem como valor *22 de Fevereiro*. No entanto, quando é efectuada a normalização do mês, o valor deste é sobreposto por *Janeiro*. Assim, a normalização desta expressão tem como valor final *22 de Janeiro de 2007*, o que está incorrecto dado que só haverá um próximo dia 22 de Janeiro em 2008. Este problema é resolvido efectuando um pós-processamento que detecta este tipo de situação fazendo a correcção do ano.

Expressões como *ontem*, *hoje* ou *amanhã* encontram-se também classificadas com os traços `uncertain`, `timeref` e `timeval`, porque para além de serem datas relativas podem também surgir como um segundo tempo de referência em expressões do tipo *de amanhã a duas semanas*.

Considere-se a expressão *de amanhã a duas semanas*. O processo de normalização deste tipo de expressões subdivide-se em duas partes. Uma primeira parte consiste no cálculo de uma segunda data de referência (e.g., amanhã) face à primeira. Numa segunda fase é calculado o valor final da expressão em relação à data calculada anteriormente.

A normalização deste tipo de expressões é efectuado pela função `normalize_date2(#1, int sentence_id, string op)`. A mesma função é responsável pela normalização da expressão *há 2 dias atrás*. Embora o processo de normalização destas duas expressões seja semelhante, neste último caso, a diferença reside no facto de não existir uma segunda data de referência, sendo os cálculos efectuados em relação à data de referência principal. Note-se ainda que no caso da expressão *de amanhã a duas semanas* é passado o operador + como argumento da função, enquanto que no caso da expressão *há 2 dias atrás* é passado o operador −.

A função `normalize_freq(#1, int sentence_id)` realiza a normalização de entidades temporais do tipo Frequência (e.g., 2 dias por mês, 2 vezes por semana). O príncipio do processo de normalização é o mesmo utilizado anteriormente para as datas: consiste na exploração dos nós da árvore referente à expressão em causa, avaliando o seu valor e efectuando as transformações necessárias. A normalização da expressão *2 dias por mês* tem como resultado o seguinte tipo de entrada no dicionário de tempo:

```
SENTENCE: 0, NODE: 9, VALUE: P1M, FREQ: 2D
```

O campo `FREQ` refere uma frequência de dois dias num período de um mês, especificado pelo campo `VALUE`.

Expressões que denotam frequências do tipo *de 2 em 2 dias* são processadas pela função `normalize_freq2(#1, int sentence_id)`. Este tipo de expressão requer um tratamento diferente das outras frequências já referidas apenas porque apresenta um padrão diferente. No entanto, a base procedimental é a mesma. A normalização da expressão *de 2 em 2 dias* tem como resultado o seguinte tipo de entrada no dicionário de tempo:

```
SENTENCE: 0, NODE: 11, VALUE: P2D, FREQ: 1X
```

O campo `FREQ` refere uma frequência de uma vez num período de 2 dias, especificado pelo campo `VALUE`.

A normalização de expressões que referem durações de tempo (e.g., 20 dias) está a cargo da função `normalize_duration(#1, int sentence_id)`. Tal como para as frequências, são explorados os nós da árvore referente à expressão em causa, sendo avaliados os seus valores e posteriormente manipulados de forma a obter-se o valor final. A normalização da expressão (e.g., 3 anos) tem como resultado o seguinte tipo de entrada no dicionário de tempo:

```
SENTENCE: 0, NODE: 7, VALUE: P3A
```

5 Avaliação

Para medir o desempenho de um sistema, torna-se necessário efectuar uma avaliação dos seus resultados. Nesse sentido, a tarefa de Reconhecimento de Entidades Mencionadas foi submetida a uma avaliação de acordo com os critérios de avaliação do HAREM. Avaliou-se também o desempenho da tarefa de normalização de expressões temporais. Este capítulo apresenta uma descrição do processo de avaliação, assim como os resultados obtidos.

5.1 Resultados da Avaliação do Reconhecimento de Entidades Nomeadas

A avaliação é efectuada com base na Colecção Dourada [1] do HAREM, segundo as directivas (Cardoso & Santos, 2005) usadas na primeira avaliação em 2005, recorrendo às ferramentas de avaliação (*Arquitectura dos programas de avaliação do HAREM*, 2005) disponibilizadas.

Estas ferramentas são apenas utilizadas na avaliação referente ao reconhecimento de entidades das categorias Obra e Valor, sendo as restantes categorias alvo de um processo de avaliação manual, uma vez que não se encontram conforme as directivas do HAREM.

Pretende-se avaliar o sistema segundo duas vertentes: a Identificação de entidades e a Classificação Semântica.

A avaliação da tarefa de identificação tem como objectivo medir a eficiência do sistema na delimitação correcta de entidades mencionadas.

A avaliação da classificação semântica tem como objectivo medir a eficácia do sistema na classificação de entidades mencionadas de acordo com uma hierarquia de categorias e subtipos definidos pelo HAREM, que foi exaustivamente criada e revista conjuntamente, de maneira a reflectir as diversas categorias e subtipos que as entidades mencionadas podem apresentar.

Para permitir obter mais informação sobre o sistema, a classificação semântica subdivide-se em quatro modalidades:

[1] A Colecção Dourada (CD) é um conjunto de textos previamente etiquetados manualmente de acordo com um conjunto de directivas.

- Classificação semântica por categorias, onde se pontua apenas a categoria da etiqueta;
- Classificação semântica por tipo, onde se pontuam apenas as entidades mencionadas que têm categoria(s) pontuada(s) como correcta(s) e onde se avalia o subtipo da etiqueta;
- Classificação semântica combinada, onde se avaliam as categorias e os tipos da EM, através de uma pontuação que combina as duas;
- Classificação semântica plana, onde se avaliam os pares categoria-tipo como folhas de uma classificação plana, considerando apenas como certos os casos que tenham categoria e tipo correctos.

A avaliação atribui a seguinte pontuação na tarefa de Identificação:

Correcta: quando o átomo inicial e o átomo final da EM são iguais na submissão e na Colecção Dourada, e o número de átomos da EM é o mesmo nas duas listas;

Parcialmente Correcta (por defeito): quando pelo menos um átomo da saída do sistema corresponde a um átomo de uma EM na Colecção Dourada, e o número total de átomos da EM do sistema participante é menor do que o número de átomos da respectiva EM da Colecção Dourada;

Parcialmente Correcta (por excesso): quando pelo menos um átomo da saída do sistema corresponde a um átomo de uma EM na Colecção Dourada, e o número total de átomos da EM do sistema participante é maior ou igual ao número de átomos da respectiva EM da Colecção Dourada;

Em Falta: quando o sistema do participante falha em detectar correctamente qualquer átomo de uma certa EM da Colecção Dourada;

Espúria: quando é delimitada uma alegada EM, que não consta na Colecção Dourada, parcialmente ou no total.

No caso das entidades correctamente identificadas é atribuída a pontuação 1. As entidades mencionadas identificadas como parcialmente correctas são pontuadas pela fórmula: $0,5 * (nc/nd)$, onde nc representa a cardinalidade da intersecção dos átomos de duas entidades mencionadas e nd a cardinalidade da reunião dos átomos de duas entidades mencionadas.

Na Classificação Semântica as entidades mencionadas são pontuadas como:

Correcta: quando a resposta do sistema coincide com a informação na Colecção Dourada;

Em Falta: quando o sistema não atribui uma dada classificação presente na Colecção Dourada;

Espúria: quando o sistema atribui uma classificação que não existe na Colecção Dourada.

No caso particular da Classificação Semântica Combinada, a avaliação atribui a seguinte pontuação:

- 0, se a categoria não estiver correcta;
- 1, se a categoria estiver correcta mas o tipo não;
- $1 + (1 - nc/n) - (ne/n)$, se a categoria estiver correcta e pelo menos um tipo também, sendo nc, ne e n, respectivamente, o número de tipos correctos, o número de tipos espúrios e o número de tipos possível na categoria.

Os resultados da avaliação são apresentados segundo vários critérios [2]:

Precisão: mede a proporção de respostas correctas em todas as respostas fornecidas pelo sistema;

Abrangência: mede a percentagem de entidades (neste caso, contidas na Colecção Dourada) que o sistema conseguiu identificar;

Medida F: A Medida F combina as medidas de precisão e de abrangência para cada tarefa, de acordo com a seguinte fórmula: $Medida\ F = (2 * Precisão * Abrangência)/(Precisão + Abrangência)$;

Sobre-geração: mede o excesso de resultados espúrios que um sistema produz, ou seja, quantas vezes produz resultados errados;

Sub-geração: mede o quanto faltou ao sistema analisar, dada a solução conhecida (a Colecção Dourada, neste caso).

Para participar na tarefa de Identificação as entidades devem ser abrangidas por uma etiqueta que contém o nome da categoria: `<CATEGORIA>`*entidade*`</CATEGORIA>`.

Para participar na tarefa de Classificação Semântica a etiqueta deve ser complementada com o nome do subtipo da categoria: `<CATEGORIA TIPO="SUBTIPO">`*entidade*`</CATEGORIA>`.

Nesse sentido, recorreu-se a um etiquetador (Romão, 2007) para converter as entidades no formato exigido pelas ferramentas de avaliação do HAREM.

Note-se que não são atribuídas quaisquer idenficações aos sistemas, pelo que os resultados obtidos nas várias tarefas são apresentados de forma destacada. Os resultados encontram-se organizados por ordem decrescente da Medida F.

[2] A fórmula para cada critério varia consoante a avaliação em questão (Identificação ou Classificação Semântica). Para detalhes acerca das fórmulas deve-se consultar o HAREM (*Avaliação no HAREM: Método e medidas*, 2005).

5.1.1 Resultados da Avaliação para a Categoria Obra

Estabelecendo uma comparação entre os resultados obtidos na tarefa de Identificação (tabela 5.1) e os resultdos obtidos pelos concorrentes da última avaliação do HAREM, na tarefa de Identificação, constata-se que o sistema obteve a melhor Precisão (48,41%), havendo uma diferença de cerca de 19% em relação à segunda melhor classificação. Em termos de Medida F foi conseguido o segundo melhor valor (0.1675). Em contrapartida, o valor da Abrangência foi o quarto melhor (10,13%), embora com apenas algumas décimas de diferença dos outros sistemas e tendo em conta que o melhor classificado só teve 18,85%. Note-se ainda que o sistema obteve o melhor desempenho na Sobre-geração (0,4390).

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
20.58	18.85	0.1968	0.7041	0.7336
48.41	**10.13**	**0.1675**	**0.4390**	**0.8827**
28.25	11.35	0.1620	0.6628	0.8645
25.05	10.42	0.1472	0.6966	0.8738
5.302	5.748	0.05516	0.8966	0.8972
9.375	0.1744	0.003425	0.7500	0.9953

Tabela 5.1: Resultados da identificação de entidades para a categoria Obra.

Relativamente à avaliação da Classificação Semântica Combinada, comparando os resultados obtidos (tabela 5.2) com os resultados da última avaliação do HAREM, verifica-se que o sistema obteve o melhor valor em termos de Precisão (51,75%) e de Medida F (0,1769) e um quarto lugar na Abrangência (10,67%).

Precisão Máxima do Sistema (%)	Abrangência Máxima na CD (%)	Medida F
51.75	**10.67**	**0.1769**
29.75	11.96	0.1706
17.30	15.85	0.1654
26.50	11.02	0.1557
5.097	5.526	0.05303
13.64	0.1993	0.003929

Tabela 5.2: Resultados da classificação semântica combinada para a categoria Obra.

Os resultados obtidos na Classificação Semântica Plana (tabela 5.3) revelam que o sistema alcançou o primeiro lugar na Precisão (51,75%) e também na Medida F (0,1769). O valor obtido na Abrangência (10,67%) encontra-se em terceiro lugar. Foi conseguido também o melhor desempenho na Sobre-geração (0,425).

No que diz respeito à Classificação Semântica por Categorias (tabela 5.4), verifica-se que o sistema obteve o melhor valor em termos de Precisão (51,75%), o segundo lugar na Medida F (0,1769) e quarto na Abrangência (10,67%). Mais uma vez foi conseguido o melhor desempenho na Sobre-geração (0,425).

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
51.75	**10.67**	**0.1769**	**0.425**	**0.8814**
29.75	11.96	0.1706	0.6628	0.8645
26.50	11.02	0.1557	0.6966	0.8738
10.66	9.761	0.1019	0.8673	0.8785
3.502	3.797	0.03643	0.9569	0.9533
0	0	0	0.7500	1.000

Tabela 5.3: Resultados da classificação semântica plana para a categoria Obra.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
22.29	20.41	0.2131	0.7041	0.7336
51.75	**10.67**	**0.1769**	**0.425**	**0.8814**
29.75	11.96	0.1706	0.6628	0.8645
26.50	11.02	0.1557	0.6966	0.8738
6.294	6.823	0.06548	0.8966	0.8972
18.75	0.3488	0.006849	0.7500	0.9953

Tabela 5.4: Resultados da classificação semântica por categorias para a categoria Obra.

Na Classificação Semântica por Tipos (tabela 5.5) conseguiu-se o primeiro lugar na Precisão (90%), Abrangência (90%) e Medida F (0,9). Não houve qualquer Sobre-geração ou Sub-geração, o que significa que o sistema atribuiu sempre o tipo correcto às entidades.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
90.0	**90.0**	**0.9**	**0**	**0**
88.24	88.24	0.8824	0	0
87.37	87.37	0.8737	0	0
36.01	36.65	0.3633	0.5517	0.5439
33.85	36.93	0.3533	0.5833	0.5455

Tabela 5.5: Resultados da classificação semântica por tipos para a categoria Obra.

5.1.2 Resultados da Avaliação para a Categoria Valor

A Precisão obtida (44,62%) na tarefa de Identificação (tabela 5.6) encontra-se em quarto lugar e a Abrangência (48,00%) em quinto lugar, o que resulta numa Medida F de 0,4625 que ocupa a quinta posição. Tentou-se ainda, através de uma regra, considerar como quantidades todos os números (dígitos) que não fazem parte de outras entidades. No entanto, apesar desta regra aumentar o valor da Abrangência (61,60%), diminui bastante a Precisão (29,98%), resultando numa Medida F inferior (0,4033), pelo que se optou por retirar a regra da gramática.

Alguns problemas ao nível do XIP impedem que sejam obtidos resultados mais elevados na

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
84.82	79.69	0.8217	0.1000	0.1628
44.33	84.50	0.5816	0.4611	0.04175
35.35	82.21	0.4944	0.5610	0.03549
35.24	81.58	0.4922	0.5618	0.04175
44.62	**48.00**	**0.4625**	**0.4853**	**0.4547**
34.44	68.46	0.4583	0.5536	0.1023
33.91	69.59	0.4560	0.5615	0.08977
55.72	38.16	0.4530	0.3415	0.5407
52.42	35.79	0.4254	0.3884	0.5720

Tabela 5.6: Resultados da identificação de entidades para a categoria Valor.

identificação de entidades da categoria Valor. A execução de regras de alisamento, no módulo de agrupamento (*chunker*) do XIP, impede o reconhecimento de entidades do tipo Duração, tais como *[aos] 6 anos*, *[cerca de] 6 anos*, ou *[com] 6 anos*, que são identificadas através de sintagmas nominais (NP). Ou seja, quando as regras de alisamento detectam determinados padrões (e.g., aos, cerca de, com, etc.) seguidos de sintagmas nominais (e.g., 6 anos), colapsam toda essa estrutura na forma de um sintagma preposicional (PP), perdendo-se a informação acerca do sintagma nominal.

Um outro problema prende-se com a definição de entidade do tipo Valor no HAREM. O sistema reconhece quantidades segundo as directivas apresentadas na secção 3.1. Ou seja, uma quantidade é um valor quantificado por uma unidade. No entanto, o sistema ignora a regra do HAREM que diz que uma entidade mencionada tem que conter um valor numérico (em forma de dígito) ou uma palavra começada por uma maiúscula. Isto dá origem a que entidades classificadas pelo sistema, como por exemplo *vinte metros*, sejam consideradas incorrectas pelas ferramentas de avaliação do HAREM.

Este último problema tem ainda mais relevância se se considerar a incerteza associada à classificação de expressões como *um carro*. Esta expressão pode reflectir uma quantidade ou apenas um sintagma nominal comum, conforme a palavra *um* se trate de um número ou de um artigo indefinido, respectivamente.

A única codificação actualmente suportada (ISO Latin-1) pelo analisador morfológico Palavroso representa também um obstáculo à identificação de valores monetários, dado que não contempla o símbolo do euro (€).

A concatenação de números é um problema actual do XIP que pode afectar a tarefa de Identificação de entidades. Isto é, o número *vinte e três* resulta do agrupamento do número *vinte* com uma conjunção e com o número *três*. No entanto, a forma como esta operação está a ser realizada no XIP, permite que a expressão *dois e três* seja também classificada como um único número, resultante do agrupamento do número *dois* com uma conjunção e com o número *três*.

Poderá eventualmente haver alguns problemas ao nível da Identificação, relacionados com o etiquetador responsável pela classificação das entidades, segundo o formato do HAREM, visto este não ter sido originalmente concebido para estas categorias específicas. Os problemas que poderão surgir estão relacionados com a delimitação de entidades, isto é, pode acontecer que a etiqueta seja fechada antes do fim da entidade (e.g., `<VALOR TIPO="QUANTIDADE">20 mil</VALOR> milhões de metros`).

Segundo os resultados obtidos na Classificação Semântica Combinada (tabela 5.7), constata-se que a Precisão Máxima do Sistema (46,27%) encontra-se em quarto lugar. A Abrangência obtida (49,40%) situa-se em sétimo lugar e a Medida F (0,4779) em quinto.

Precisão Máxima do Sistema (%)	Abrangência Máxima na CD (%)	Medida F
84.81	80.21	0.8245
44.94	85.75	0.5897
35.61	82.81	0.4980
35.57	82.35	0.4968
46.27	**49.40**	**0.4779**
35.30	71.12	0.4718
56.97	40.10	0.4707
34.75	72.26	0.4693
53.32	37.42	0.4398

Tabela 5.7: Resultados da classificação semântica combinada para categoria Valor.

Comparando os valores obtidos na Classificação Semântica Plana (tabela 5.8) com os resultados da última avaliação do HAREM, verifica-se que na Precisão (43,77%) o sistema classificou-se em quarto lugar. Na Abrangência (47,10%) obteve o sétimo lugar e na Medida F (0,4537) o quarto lugar.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
82.98	77.96	0.8039	0.1444	0.2025
41.24	78.60	0.5410	0.5586	0.1921
55.91	38.40	0.4553	0.4043	0.5762
43.77	**47.10**	**0.4537**	**0.5333**	**0.5021**
30.82	71.35	0.4304	0.6564	0.2463
30.77	71.56	0.4303	0.6571	0.2443
52.55	35.98	0.4271	0.4421	0.6013
31.03	61.67	0.4129	0.6502	0.2881
30.55	62.69	0.4108	0.6562	0.2777

Tabela 5.8: Resultados da classificação semântica plana para a categoria Valor.

A Precisão obtida (47,34%) pelo sistema situa-se na quinta posição da tabela da Classificação Semântica por Categorias (tabela 5.9). Na Abrangência (50,93%) foi obtido um sétimo lugar e na Medida F (0,4907) o sistema classificou-se na sétima posição.

Em oposição aos resultados obtidos na identificação e classificação das entidades da categoria Va-

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
86.98	81.71	0.8426	0.1000	0.1628
47.49	90.51	0.6229	0.4611	0.04175
38.83	90.31	0.5431	0.5610	0.03549
38.74	89.68	0.5410	0.5618	0.04175
38.95	77.42	0.5183	0.5536	0.1023
38.32	78.64	0.5153	0.5615	0.08977
47.34	**50.93**	**0.4907**	**0.4863**	**0.4557**
60.04	41.24	0.4890	0.3404	0.5407
56.05	38.38	0.4556	0.3872	0.5720

Tabela 5.9: Resultados da classificação semântica por categorias para a categoria Valor.

lor, a Classificação Semântica por Tipo (tabela 5.10) obteve o segundo melhor resultado na Abrangência (85,52%) e Medida F (0.8586), ficando em terceiro lugar na Precisão (85,20%). Ou seja, o sistema apresenta alguns problemas na identificação e classifcação das entidades quanto à categoria, mas uma vez identificadas, o sistema apresenta um bom desempenho (Medida F) na atribuição do tipo.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
92.20	93.12	0.9266	0.04938	0.04738
85.20	**86.52**	**0.8586**	**0.09160**	**0.08527**
86.18	84.07	0.8511	0.09000	0.06829
85.16	83.61	0.8437	0.09722	0.07727
76.53	82.03	0.7918	0.1809	0.1569
70.32	74.46	0.7233	0.2160	0.2135
70.09	74.19	0.7208	0.2188	0.2165
69.67	68.87	0.6927	0.2158	0.2064
69.51	68.70	0.6910	0.2165	0.2070

Tabela 5.10: Resultados da classificação semântica por tipos para a categoria Valor.

5.1.3 Resultados da Avaliação Global para as Categorias Obra e Valor

Os resultados globais obtidos nas categorias Obra e Valor são comparados com os resultados conseguidos pelos concorrentes na avaliação do HAREM em 2005, segundo o Cenário Selectivo. A comparação de resultados segundo este cenário poderá não ser a mais adequada, uma vez que nele estão presentes os resultados de sistemas que concorreram ao HAREM de forma discriminatória, ou seja, especificando quais as categorias de entidades mencionadas às quais pretendiam concorrer. Os sistemas concorrentes poderão ter concorrido em categorias diferentes, assim como o número de categorias nas quais participaram poderá também ter sido diferente.

Note-se que estes resultados são obtidos da combinação do reconhecimento de entidades de duas categorias apenas – Obra e Valor – e que por sinal a categoria Obra é aquela onde se consegue pior

desempenho ao nível da tarefa de Identificação (segundo a análise dos resultados obtidos pelos sistemas concorrentes).

Comparando os valores globais obtidos na tarefa de Identificação (tabela 5.11), para as categorias Obra e Valor, com os valores globais obtidos pelos sistemas concorrentes, verifica-se que o sistema obteve a décima segunda Medida F (0,4061). A Abrangência (37,01%) obtida situa-se também na décima segunda posição e a Precisão (44,98%) na décima quarta.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
78.50	82.84	0.8061	0.07913	0.07329
77.15	84.35	0.8059	0.09134	0.03575
76.85	83.56	0.8006	0.08966	0.04035
77.43	69.57	0.7329	0.09524	0.2079
76.31	58.40	0.6616	0.09725	0.3157
59.45	64.39	0.6182	0.2018	0.1607
56.95	64.39	0.6044	0.2353	0.1607
57.21	63.51	0.6020	0.2315	0.1707
58.57	52.12	0.5516	0.3408	0.4240
58.44	51.93	0.5499	0.3413	0.4240
47.32	54.50	0.5066	0.1119	0.1687
44.98	**37.01**	**0.4061**	**0.4764**	**0.5753**
36.89	35.03	0.3594	0.5013	0.5408
39.45	25.28	0.3082	0.4432	0.6472
57.40	17.72	0.2708	0.2330	0.7866
47.12	10.98	0.1781	0.1596	0.8101

Tabela 5.11: Resultados globais da identificação de entidades para as categorias Obra e Valor.

Os resultados obtidos na Classificação Semântica Combinada (tabela 5.12) revelam que o sistema classificou-se em sétimo lugar na Precisão (46,69%) e na Abrangência (37,76%) e obteve também a sétima melhor medida F (0,4175).

Na Classificação Semântica Plana (tabela 5.13), o sistema obteve a sexta melhor Precisão (44,35%) e um sétimo lugar na Abrangência (36,52%) e Medida F (0,4005).

Perante os resultados obtidos na Classificação Semântica por Categorias (tabela 5.14) constata-se que o sistema conseguiu a sétima melhor Medida F (0,4304), que resulta de um oitavo lugar na Precisão (47,66%) e um sétimo lugar na Abrangência (39,24%).

Na Classificação Semântica por Tipos (tabela 5.15) o sistema obteve os melhores resultados na Precisão (85,59%), Abrangência (86,81%) e Medida F (0,8620).

Precisão Máxima do Sistema (%)	Abrangência Máxima na CD (%)	Medida F
56.30	60.42	0.5829
65.10	51.13	0.5728
57.28	49.85	0.5330
56.79	48.73	0.5245
47.02	42.65	0.4473
46.57	42.25	0.4430
46.69	**37.76**	**0.4175**
36.37	26.81	0.3087
27.06	31.66	0.2918
32.20	24.64	0.2792
39.04	19.07	0.2563
31.66	19.66	0.2426
49.57	13.49	0.2121
38.76	7.025	0.1189

Tabela 5.12: Resultados globais da classificação semântica combinada para as categorias Obra e Valor.

5.1.4 Resultados da Avaliação para a Categoria Relações de Parentesco

Uma vez que a categoria Relações de Parentesco não se encontra de acordo com as directivas do HAREM, a tarefa de avaliação foi realizada manualmente, recorrendo-se a dez textos da Colecção Dourada do HAREM (escolhidos de forma aleatória).

Para este caso não se justifica a realização de uma avaliação semântica, dado que esta categoria não possui quaisquer tipos.

O valor obtido (tabela 5.16) na Precisão (89,74%) justifica-se pelo facto de estas entidades serem reconhecidas exclusivamente através do léxico. O valor obtido na Abrangência (68,63%) é mais baixo que o da Precisão, devido ao facto de nos textos da Colecção Dourada surgirem termos diminutivos, tanto portugueses (e.g., mamã, papá) como brasileiros (e.g., mamãe, papai), que não estão a ser considerados pelo léxico das relações de parentesco.

5.1.5 Resultados da Avaliação para a Categoria Tempo

Devido ao facto das directivas adoptadas para o reconhecimento de entidades da categoria Tempo não ser compatível com as directivas para a mesma categoria no HAREM, o processo de avaliação teve que ser efectuado manualmente, recorrendo-se a dez textos da Colecção Dourada do HAREM (os mesmos escolhidos para avaliar a categoria Relações de Parentesco).

A partir da análise das tabelas 5.17, 5.18, 5.19, 5.20 e 5.21 verifica-se que o sistema obteve uma Precisão de 82,36% e uma Abrangência de 72,53% na tarefa de Identificação, uma Precisão de 90,56% e uma Abrangência de 79,74% na Classificação Semântica Combinada, uma Precisão de 76,03% e uma

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
63.13	48.52	0.5487	0.3349	0.4843
51.50	54.45	0.5293	0.4569	0.4295
52.39	44.79	0.4830	0.4432	0.5292
52.16	43.94	0.4770	0.4489	0.5364
43.48	38.70	0.4095	0.5406	0.5944
42.86	38.11	0.4034	0.5467	0.6004
44.35	**36.52**	**0.4005**	**0.5255**	**0.6123**
25.03	27.94	0.2641	0.7290	0.6910
44.81	13.84	0.2114	0.4855	0.8585
13.41	12.72	0.1306	0.5762	0.8466
13.46	12.18	0.1279	0.7319	0.8724
11.00	11.97	0.1147	0.6551	0.8675
12.76	8.162	0.09956	0.5281	0.8954
12.82	2.877	0.04700	0.5270	0.9631

Tabela 5.13: Resultados globais da classificação semântica plana para as categorias Obra e Valor.

Abrangência de 66,95% na Classificação Semântica Plana, uma Precisão de 85,07% e uma Abrangência de 74.91% na Classificação Semântica por Categorias e uma Precisão de 80,10% e uma Abrangência de 66.95% na Classificação Semântica por Categorias.

A ausência de Sub-geração na Classificação Semântica deve-se ao facto de todas as entidade detectadas terem sido classificadas com uma categoria e com um tipo.

5.2 Resultados da Avaliação da Normalização de Expressões Temporais

Os resultados da normalização de expressões temporais foram avaliados manualmente, recorrendo aos resultados obtidos no processo de avaliação do reconhecimento de entidades mencionadas do tipo Tempo.

Todas as entidades da categoria Tempo, total ou parcialmente identificadas, e com a categoria e tipo correctamente atribuídos foram posteriormente submetidas a um processo de normalização. A tabela 5.22 mostra que a maior parte das expressões temporais foram correctamente normalizadas (84%) o que indica que, por vezes, embora algumas entidades não sejam identificadas no seu todo, ainda assim o sistema consegue normalizar o seu valor, total ou parcialmente.

Note-se que, uma vez que as entidades do tipo Hora não estão a ser consideradas na normalização, a percentagem de expressões correctamente normalizadas (84%) deve-se também ao facto de o número de entidades do tipo Hora, presentes nos textos utilizados na avaliação, ser bastante reduzido.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
61.30	64.81	0.6301	0.3416	0.3131
68.67	52.78	0.5968	0.2584	0.4286
62.31	53.27	0.5744	0.3383	0.4363
61.70	51.97	0.5642	0.3422	0.4475
50.54	44.99	0.4760	0.4579	0.5237
50.41	44.82	0.4745	0.4582	0.5244
47.66	**39.24**	**0.4304**	**0.4818**	**0.5793**
40.17	38.08	0.3910	0.5004	0.5399
43.45	27.80	0.3391	0.4419	0.6462
32.35	35.19	0.3371	0.6313	0.5989
31.26	34.88	0.3297	0.6418	0.5979
28.02	25.35	0.2662	0.6987	0.7264
50.11	15.47	0.2364	0.4694	0.8406
43.42	9.743	0.1591	0.4417	0.8758

Tabela 5.14: Resultados globais da classificação semântica por categorias para as categorias Obra e Valor.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
85.59	**86.81**	**0.8620**	**0.08421**	**0.07829**
85.08	84.86	0.8497	0.1036	0.09805
80.95	78.89	0.7991	0.1459	0.1931
80.21	79.53	0.7987	0.1598	0.1641
79.80	79.60	0.7970	0.1624	0.1602
79.13	80.14	0.7964	0.1636	0.1598
78.39	79.42	0.7890	0.1733	0.1676
78.05	79.05	0.7854	0.1744	0.1708
69.89	69.47	0.6968	0.2436	0.2316
77.85	44.51	0.5664	0.1918	0.5335
76.82	29.84	0.4299	0.1665	0.6697
57.28	27.60	0.3725	0.3273	0.6672
52.95	23.17	0.3224	0.3523	0.7029
52.68	23.07	0.3209	0.3557	0.7043

Tabela 5.15: Resultados globais da classificação semântica por tipos para as categorias Obra e Valor.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
89.74	68.63	0.7778	0.1026	0.2353

Tabela 5.16: Resultados da identificação de entidades para a categoria Relações de Parentesco.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
82.36	72.53	0.7714	0.05085	0.1642

Tabela 5.17: Resultados da identificação de entidades para a categoria Tempo.

Precisão Máxima do Sistema (%)	Abrangência Máxima na CD (%)	Medida F
76.03	66.95	0.7120

Tabela 5.18: Resultados da classificação semântica combinada para a categoria Tempo.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
76.03	66.95	0.7120	0.1525	0

Tabela 5.19: Resultados da classificação semântica plana para a categoria Tempo.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
85.07	74.91	0.7967	0.05085	0

Tabela 5.20: Resultados da classificação semântica por categorias para a categoria Tempo.

Precisão (%)	Abrangência (%)	Medida F	Sobre-geração	Sub-geração
80.10	95.44	0.8710	0.1017	0

Tabela 5.21: Resultados da classificação semântica por tipos para a categoria Tempo.

Certas (%)	Parcialmente Certas (%)	Não Normalizadas / Erradas (%)
84	2	14

Tabela 5.22: Resultados da avaliação da normalização de expressões temporais.

6 Conclusões e Trabalho Futuro

De acordo com os resultados apresentados no capítulo 5, pode-se afirmar que o sistema tem um bom desempenho no reconhecimento de entidades da categoria Obra, apresentando resultados (Precisão e Abrangência) superiores à maior parte dos concorrentes do HAREM 2005.

Quanto ao reconhecimento de entidades da categoria Valor, o sistema apresentou alguns problemas nesta tarefa. Estes problemas prendem-se com várias questões, nomeadamente com as regras de alisamento, responsáveis pela criação de sintagmas preposicionais (PP), que impossibilitam o reconhecimento de certos valores do tipo Quantidade; a codificação usada (ISO Latin-1) que não suporta o símbolo do euro (€); a forma como os números estão a ser reconhecidos ou concatenados, o que permite que expressões como *um e dois* sejam classificadas com um único número; o facto de se ter optado por considerar entidades do tipo Quantidade, expressões como *dois carros*, em que o número surge escrito por extenso, que estão incorrectas segundo as ferramentas de avaliação do HAREM; a dificuldade na classificação de quantidades que contêm o artigo indefinido *um* ou *uma*, que por vezes pode ser um número.

De modo a resolver o problema referido ao nível da codificação de letra, torna-se necessário que todos os componentes da cadeia de processamento passem a suportar a codificação de letra UTF-8.

No reconhecimento de entidades da categoria Relações de Parentesco verificou-se que o principal problema do sistema reside na identificação de nomes diminutivos, tanto em português (e.g., mamã, papá) como em brasileiro (e.g., mamãe, papai).

Este problema pode ser resolvido complementando as gramáticas locais e léxicos com regras e vocabulário, com o objectivo de aumentar o número de entidades actualmente reconhecidas. Esta solução aplica-se a todas as outras categorias de entidades também.

No que diz respeito ao reconhecimento de entidades Tempo, o sistema apresentou também um bom desempenho, tendo em consideração que os valores obtidos se encontram ao nível dos resultados conseguidos pelos melhores sistemas que participaram no HAREM 2005.

Neste contexto, a questão da concatenação dos números revela-se também um problema, uma vez que impossibilita o correcto reconhecimento de expressões como *dias 3 e 4 de Março*, onde estão subjacentes duas datas distintas (*3 de Março* e *4 de Março*). Actualmente não é possível distinguir estes dois

dias, dado que são classificados como uma única entidade numérica. A correcção deste problema passa por uma reestruturação das regras que fazem o reconhecimento de números.

Considerando ainda a expressão *dias 3 e 4 de Março* e sabendo que ao ser processada pelo sistema são reconhecidas duas entidades do tipo Data, nomeadamente a entidade *dia 3* e a entidade *4 de Março*, é possível estabelecer uma relação de coordenação entre as duas entidades (através de regras de dependência), permitindo saber que o mês referente à segunda entidade diz também respeito à primeira.

Relativamente à tarefa de normalização, embora se tenha obtido um bom desempenho, conseguindo-se normalizar a maior parte (84%) das entidades com referências temporais identificadas pelo sistema, há que passar também a contemplar as entidades do tipo Hora no processo de normalização.

Para melhorar o desempenho do sistema ao nível da normalização, deve-se recorrer à informação linguística associada aos verbos, no sentido de melhorar o ancoramento de expressões no tempo. As frases seguintes referem datas diferentes consoante o tempo verbal do verbo *partir*:

Ele partiu segunda-feira.

Ele parte segunda-feira.

O passado do verbo *partir* indica que se trata da segunda-feira anterior. O presente do verbo *partir* indica que se trata da próxima segunda-feira.

Quando se utiliza a expressão *há 2 dias*, sabe-se que para se obter a data referida é necessário recuar exactamente 2 dias no tempo. Porém, supondo que a data actual é 15 de Dezembro de 2000, a expressão *há 3 meses* não refere necessariamente o dia 15 de há precisamente 3 meses atrás (Setembro). A expressão pode referir uma data algures por volta do dia 15 de Setembro. Uma possível solução para abordar este tipo de expressões seria definir uma granularidade ou margem de incerteza para cada unidade de tempo (dia, mês, ano) (Caroline Hagege, 2007). Por exemplo, para o mês poder-se-ia definir uma granularidade de 50%, que corresponde a 15 dias. Isto aplicado à expressão *há 3 meses* significa que a data referenciada situa-se algures entre 1 de Setembro e 30 de Setembro de 2000 (- 3 meses $\pm$ 15 dias).

Deve ser desenvolvido trabalho no sentido de detectar e avaliar os diversos tempos de referência, passando o seu valor como argumento das funções já implementadas, de forma a permitir ancorar correctamente datas relativas no tempo.

Numa fase seguinte da normalização do tempo devem ser estabelecidas relações entre eventos e entre entidades temporais e eventos, como descrito na secção 2.2.2, e atribuir-lhes uma classificação de acordo com as 13 relações entre intervalos de Allen (secção 2.1). Neste sentido, torna-se necessário adicionar informação linguística extra a determinados verbos que permita resolver ambiguidades. Considerem-se as frases:

Ele descansou 2 dias depois do exame.

Ele chegou 2 dias depois do exame.

Como se pode constatar, a expressão *2 dias depois do exame* pode ter diferentes interpretações ao nível do tempo, conforme o verbo utilizado na frase. O verbo *descansar* está associado a um intervalo de tempo, enquanto que o verbo *chegar* refere um ponto no tempo.

O mesmo sucede com o exemplo seguinte:

Depois de amanhã vou ao médico.

Depois de amanhã sou um homem livre.

Nesta situação, consoante o verbo em questão, a expressão *depois de amanhã* pode corresponder a um ponto específico no tempo, ou ao ponto de inicial de um intervalo de tempo infinito.

Bibliografia

AC/DC – Acesso a corpora, disponibilização de corpora. (n.d.). http://acdc.linguateca.pt/index.html/.

Allen, J. F. (1983). *Maintaining Knowledge about Temporal Intervals*. Communications of the ACM.

Andrade, A. R. de. (2003). *Os Corpora Linguísticos: uma nova forma de «fazer Lexicografia»?* Actas do XVIII Encontro Nacional da Associação Portuguesa de Linguística.

Arquitectura dos programas de avaliação do HAREM. (2005). (http://poloxldb.linguateca.pt/harem.php?l=Arquitectura)

Avaliação no HAREM: Método e medidas. (2005). (http://poloxldb.linguateca.pt/harem.php?l=medidas)

Beule, J. D. (n.d.). *Creating Temporal Categories for an Ontology of Time.* (http://arti.vub.ac.be/~joachim/Bnaic04.pdf)

Cardoso, N., & Santos, D. (2005, Janeiro). *Directivas e categorias para identificação e classificação na colecção dourada do HAREM.* (http://poloxldb.linguateca.pt/harem.php?l=classificacao_v2)

Caroline Hagege, X. T. (2007). *Work on Temporal Processing*. Xerox Research Centre Europe.

DAML-Time Homepage. (n.d.). http://www.cs.rochester.edu/~ferguson/daml/.

DARPA - Defense Advanced Research Projects Agency. (n.d.). http://www.darpa.mil/.

Endriss, U., & Gabbay, D. (n.d.). *Halfway between Points and Intervals: A Temporal Logic Based on Ordered Trees.* (http://www.illc.uva.nl/~ulle/pubs/files/EndrissGabbayESSLLI2003.pdf)

Extensible Markup Language (XML). (n.d.). http://www.w3.org/XML/.

Ferro, L., Gerber, L., Mani, I., Sundheim, B., & Wilson, G. (2003, Novembro). *TIMEX2 Quick Guide.*

Ferro, L., Gerber, L., Mani, I., Sundheim, B., & Wilson, G. (2005, Setembro). *TIDES - 2005 Standard for the Annotation of Temporal Expressions.*

Ferro, L., Kozierok, R., Gerber, L., Mani, I., Sundheim, B., & Wilson, G. (n.d.). *Annotating Temporal Information - From Theory to Practice.* (http://timex2.mitre.org/articles/HLT2002-timex2.pdf)

Ferro, L., Kozierok, R., Gerber, L., Mani, I., Sundheim, B., & Wilson, G. (2004, Abril). *The TERN 2004 Evaluation Plan - Time Expression Recognition and Normalization.* (http://timex2.mitre.org/tern_evalplan-2004.29apr04.pdf)

Gibbs, A. (2004, Novembro). *Temporal Reasoning in Natural Language Processing.* (http://www.cs.umanitoba.ca/~ckemke/74.406-NLP/Notes-2005/Student-Presentations/Temporal-Reasoning.ppt)

Halpern, J. Y., & Shoham, Y. (1991). *A Propositional Modal Logic of Time Intervals.* Journal of the ACM.

Jurafsky, D., & Martin, J. H. (2000). *Speech and Language Processing*. Prentice-Hall.

Mamede, N. (2007, Maio). *A Cadeia de Processamento XIP.* L^{2}F – Laboratório de Sistemas de Língua Falada.

Mani, I., Pustejovsky, J., & Gaizauskas, R. (n.d.). *The Language of Time - A Reader.* http://fds.oup.com/www.oup.co.uk/pdf/0-19-926853-3.pdf. (Oxford University Press)

Medeiros, J. C. (1995). *Processamento Morfológico e Correcção Ortográfica do Português*. Unpublished master's thesis, Instituto Superior Técnico – Universidade Técnica de Lisboa, Portugal.

Mendes, A. (2007). *Clefomania - QA@L2F: Primeiros Passos*. Unpublished master's thesis, Instituto Superior Técnico – Universidade Técnica de Lisboa, Portugal.

MUC - Message Understanding Conferences. (n.d.). http://www.itl.nist.gov/iaui/894.02/related_projects/muc/.

Pardal, J. P. (2007, Maio). *Manual do Utilizador do RuDriCo.* L^{2}F – Laboratório de Sistemas de Língua Falada.

Pustejovsky, J., Katz, G., & Gaizauskas, R. (2003, Agosto). *Practical Applications of Temporal and Event Reasoning.* (www.cs.brandeis.edu/~jamesp/arda/time/documentation/Tuesday.ppt)

Pustejovsky, J., Katz, G., Gaizauskas, R., Setzer, A., Castaño, J., Ingria, R., et al. (2004, Janeiro). *TimeML: Robust Specification of Event and Temporal Expressions in Text.* (http://complingone.georgetown.edu/~linguist/papers/TimeML.pdf)

Pustejovsky, J., Knippen, R., Litman, J., & Saurí, R. (2005, Novembro). *Temporal and Event Information in Natural Language Text.* (http://timeml.org/site/publications/timeMLpubs/PustejovskyEtAl2.pdf)

Pustejovsky, J., Saurí, R., Littman, J., Knippen, B., Setzer, A., & Gaizauskas, R. (2006, Janeiro). *TimeML Annotation Guidelines.* (http://timeml.org/site/publications/timeMLdocs/annguide_1.2.1.pdf)

Reichenbach, H. (1947). *Elements of Symbolic Logic* (M. Co., Ed.).

Ribeiro, R., Mamede, N. J., & Trancoso, I. (2003). Using Morphossyntactic Information in TTS Systems: comparing strategies for European Portuguese. In *Computational Processing of the Portuguese Language: 6th International Workshop, PROPOR 2003, Faro, Portugal, June 26-27, 2003. Proceedings* (Vol. 2721). Springer.

Romão, L. (2007). *Reconhecimento de Entidades Mencionadas em Língua Portuguesa: Locais, Pessoas, Organizações e Acontecimentos*. Unpublished master's thesis, Instituto Superior Técnico – Universidade Técnica de Lisboa, Portugal.

Setzer, A. (2001, Setembro). *Temporal Information in Newswire Articles: an Annotation Scheme and Corpus Study.* (http://www.andrea-setzer.org.uk/PAPERS/thesis.ps)

Stanford Encyclopedia of Philosophy - Temporal Logic. (n.d.). http://plato.stanford.edu/entries/logic-temporal/.

TIDES - DARPA Translingual Information Detection, Extraction, and Summarization. (n.d.). http://www.darpa.mil/ipto/programs/tides/.

TimeML - A Formal Specification Language for Events and Temporal Expressions. (n.d.). http://www.ldc.upenn.edu/Catalog/docs/LDC20006T08/timeml_specs_1.2.1.html.

TimeML - Markup Language for Temporal and Event Expressions. (n.d.). http://www.timeml.org/site/index.html.

Xerox, R. C. E. (2001). *Xerox Incremental Parser – User's Guide (Scripting).*

Xerox, R. C. E. (2003). *XIP User Guide.*

Zhou, Q., & Fikes, R. (n.d.). *A Reusable Time Ontology.* (http://ksl.stanford.edu/KSL_Abstracts/KSL-00-01.html)