Engenharia do Léxico Computacional: aspectos seleccionados e visão

Mário Amado Alves
Laboratório de Inteligência Artificial e Ciências da Computação

O projecto dum léxico computacional requer a definição e realização dum conjunto não trivial de bases e decisões de engenharia informática. Em Alves (2002) desenvolvi um possível tal conjunto, para o caso dum léxico de elevada usabilidade e extensibilidade. Esse trabalho está publicado na íntegra na Internet (www.liacc.up.pt/~maa/ELC). No presente artigo exponho os resultados principais daquele estudo, incorrendo em certos aspectos que me parecem mais interessantes no contexto dos correntes Encontros da APL.

Primeiro descrevo de modo sucinto alguns conceitos de engenharia informática e de software. Seguidamente apresento os principais resultados da investigação reportada. Finalmente, discuto algumas ideias associadas – por vezes em oposição – àqueles resultados, e ainda em estado de 'visão'.

Engenharias

"O desenvolvedor de software profissional segue procedimentos de engenharia. Num ambiente de linguística computacional, o desenvolvimento é frequentemente mais informal" (Gibbon (2000), p. 28). Proponho contrariar esta posição 'artesanal' das ferramentas de processamento da língua – não por 'birra', mas por *requisito*, como veremos.

A *engenharia informática* é a aplicação da *ciência* ao desenvolvimento de sistemas informáticos. Assumindo a plataforma informática habitual, nomeadamente computadores pessoais correndo sistemas operativos padrão (Linux, Windows, juntar símbolos © e ® a gosto), a focagem volta-se para o processo de desenvolvimento de software, ou simplesmente *processo de software* – aquilo a que Gibbon (2000) aludia. O processo de software tem habitualmente a seguinte composição:

1. Requisitos**
2. Especificação**
3. Projecto (design)*
4. Implementação*
5. Integração
6. Teste

Actas do XVIII Encontro Nacional da Associação Portuguesa de Linguística, Lisboa, APL, 2002, pp. 91-102.

7. Exploração (deployment)
8. Manutenção

Os asteriscos marcam a importância relativa do componente no projecto POLLUX (vd. secção dedicada). O processo de software produz, e reutiliza, *itens de software*: documentos, planos, programas, etc. A *engenharia de software* fornece métodos, técnicas, e ferramentas, para efectuar o processo de software. O desenvolvimento propriamente dito centra-se nos componentes de *projecto*, *implementação* e *integração*. Alguns métodos e técnicas desta região do processo:

- COTS (Commercial Off-The-Shelf)
- Megaprogramação, ou programação por componentes
- Linguagens 'universais'
- DSL (Domain-Specific Languages)
- Modelo de dados relacional
- RTM (Reduced Technology Mix)*

O asterisco marca uma contribuição do presente autor para a panorâmica, definitivamente com aquele nome em Alves (2003), mas já conceptualmente presente em Alves (2001) e Alves (2002).

POLLUX: Portuguese Lexicon Largely Usable and eXtensible

Alves (2002) aborda o desenvolvimento do léxico computacional enquanto componente de software, nomeadamente de sistemas de processamento da língua. O trabalho foi motivado pela percepção de certas limitações dos sistemas existentes, principalmente quanto à *usabilidade* e *extensibilidade* destes. Assim, o problema abordado pode formular-se como

(1) *aumento da usabilidade e extensibilidade do léxico computacional.*

Um dos sistemas existentes em particular recebeu crucialidade, por ser o léxico computacional de 'serviço' no meio onde se realizou a investigação. Trata-se do sistema *POLARIS: POrtuguese Lexicon Acquisition and Retrieval System*, desenvolvido e mantido pelo GLINT[1]. A actividade deste grupo impunha sobre aquele sistema requisitos que o mesmo não podia suportar elegantemente – ou de todo. Por exemplo, requeria-se que o POLARIS acomodasse novo conhecimento, incluindo a rectificação de certos erros entretanto encontrados no conhecimento representado. Este requisito, aparentemente natural, era na prática impedido por um esquema de dados não normalizado e altamente comprometido com codificações não documentadas e de difícil, se não impossível, engenharia reversa.

[1] Grupo de Língua Natural (Depart. Informát. Fac. Ciênc. Tecnol. Univ. Nova Lisboa).

A formulação (1), e particularmente as categorias de *usabilidade* e *extensibilidade*, são a síntese do seguinte conjunto de requisitos, cuja compilação foi parte muito importante do trabalho:

1. herança do POLARIS
2. formas alternativas
3. palavras compostas
4. subcategorização verbal
5. equivalentes de tradução
6. aquisição automática
7. problema do OOV
8. extensibilidade nomenclatural
9. extensibilidade estrutural
10. usabilidade operacional
11. usabilidade integracional
12. eficiência temporal

O conjunto é descrito pormenorizadamente em Alves (2002).

A procura da resolução do problema passou por experiências de extensão e reengenharia do POLARIS, e tomou eventualmente a forma de projecto dum novo sistema, POLLUX (*Portuguese Lexicon Largely Usable and eXtensible*). A metodologia geral adoptada consistiu na utilização de técnicas e tecnologias informáticas *padrão*, nomeadamente modelo, sistema e linguagem de base de dados relacional (SQL), arquitectura modular, linguagem de programação de propósito geral (Ada) e tecnologias de integração comuns (GCC, linha de comando, web) – em combinação com um princípio geral de *simplificação*.

O princípio de simplificação abrange quer a interface quer a arquitectura interna do sistema. Na interface a necessidade de simplicidade advém primordialmente dos requisitos de usabilidade. É sabido que ninguém, e particularmente não o investigador em processamento da língua, adopta componentes complicados e/ou difíceis de usar. Verdade óbvia, decerto, mas nem sempre honrada. Mas porventura a determinação mais importante do princípio de simplificação é a *desinflacção do modelo de classes*, nomeadamente um modelo de dados com poucas classes, ou relações. Tal consegue-se por concentração de atributos em poucas classes primárias – o que por sua vez decorre do enriquecimento semântico destas classes.

POLLUX. Conceitos

O projecto POLLUX clarificou importantes conceitos da engenharia do léxico computacional. Começando pelo próprio conceito de léxico computacional, passámos duma situação de noção difusa, e em confusão com *base de conhecimento lexical*, para a seguinte formulação:

um determinado léxico computacional	=	uma determinada BCL	+	um determinado (sub)conjunto de módulos de acesso

Outros conceitos definidos e clarificados em Alves (2002): palavra, forma, categoria, corpus, ocorrência, tipo, palavra composta. Particularmente a palavra *palavra* é altamente polissémica, conforme usada na literatura. Lexema, forma-tipo, ocorrência de forma, forma gramatical, forma (orto)gráfica – qualquer destes conceitos é dito *palavra*, no devido contexto – e frequentemente fora dele. Por causa da vastidão denotacional que acompanha a sua vaguesa, a palavra *palavra* é utilizada no título e passagens mais gerais de Alves (2002), bem como do corrente artigo. Pela mesma razão não a utilizaremos nas formulações estritamente técnicas, mas sim aos termos inambíguos definidos, particularmente *forma*, definida como uma sequência de símbolos pertencentes a um determinado alfabeto, representante duma porção de texto e/ou discurso numa dada língua.

POLLUX. Princípios

Identificaram-se e adoptaram-se quatro princípios para o desenvolvimento de POLLUX: morfocentrismo, politeorismo, dinamismo e engenhismo.

O morfocentrismo advém do desiderato de alimentar o sistema com conhecimento adquirido *automaticamente* e *incrementalmente*, o qual sugere um esquema conceptual morfocêntrico. De facto, na origem da aquisição incremental está sempre a *forma*. Pode mesmo não haver mais nada além disso, como no caso das palavras compostas identificadas por processos meramente estatísticos sobre texto em bruto. O conceito de *Item de Conhecimento Lexical (morfocêntrico)* (ICL) e um esquema de dados baseado neste conceito foi a minha resposta a este problema. O léxico é essencialmente uma colecção de formas, mais ou menos (no limite inferior, nada) analisadas. O conceito (difuso) tradicional de *lexema* ("forma-base", *head-word*, ...) é tratado – satisfatoriamente – como um *grupo de formas*, geralmente uma *família flexional*. Um membro do grupo desempenha o papel de seu *representante*, e pode servir de âncora a propriedades que se queiram atribuir ao grupo como entidade e/ou a todos os seus membros. O conceito de equivalente de tradução é tratado – também satisfatoriamente – como um conjunto de ICLs. Ao longo de toda a tese, particularmente a partir do capítulo ICL, mostra-se como outros conceitos e problemas são também convenientemente tratados morfocentricamente. Em suma, o morfocentrismo é proposto como um princípio conveniente à solução do problema. Um resultado particularmente interessante é a morfologia unificada, descrita adiante.

Os outros princípios estão associados ao problema do OOV (Out-Of-Vocabulary), à adopção dum sistema de bases de dados relacional como componente da arquitectura, etc. Vide Alves (2002) para detalhes.

POLLUX. Esquema de dados

A base de conhecimento lexical de POLLUX está estruturada em quatro classes:

ICL	Item de Conhecimento Lexical
FTC	Fonte de conhecimento
EQT	Equivalência de tradução
SEQ	Sequência de constituintes

Sendo ICL a classe primária e mais rica de atributos, e as restantes essencialmente relacionais, isto é, representando relações entre itens da ICL. Esquema da classe ICL:

Atributos identificantes	*Ort*	Forma ortográfica
	Tip	Tipo de forma
	Cat	Categoria morfossintáctica
	Lng	Língua
	Ftc	Fonte de conhecimento
	Dtc	Data de criação
	Id	Identificador
Atributos morfológicos	*Tem*	Tempo, ou *tense and mood*
	Pes	Pessoa
	Num	Número
	Gen	Género
	Dgr	(*Degree*) Grau dos adjectivos e advérbios
Conhecimento linguístico adicional	*Mor*	Informação morfológica adicional
	Sin	Classe sintáctica
	Sem	Classe semiológica
	Snx	Informação sintáctica adicional
	Smx	Informação semiológica adicional
	Inx	Outra informação extensa
	Xφα	Atributos extraordinários
Atributos relacionais	*Seq*	Sequência constituinte
	Can	Forma canónica
	Bas	Forma básica
	Inv	Forma inversa
Registo de uso	*Dte*	Data da última escrita
	Dtl	Data da última leitura
	Nrl	Número de leituras

Um ICL concreto consiste num subconjunto de atributos valorados, ou propriedades, por exemplo:

$$\begin{bmatrix} \text{Ort} \Rightarrow \langle \text{amor} \rangle \\ \text{Tip} \Rightarrow \text{x} \\ \text{Cat} \Rightarrow \text{n} \\ \text{Num} \Rightarrow \text{s} \\ \text{Gen} \Rightarrow \text{m} \end{bmatrix}$$

POLLUX. Morfologia Unificada

Sob este nome exploro a hipótese dum sistema de composição de formas abrangedor de *todas* as classes de "construção de palavras" tradicionais, nomeadamente *flexão*, *derivação*, e *composição* propriamente dita (justaposição, etc.) Esta hipótese é motivada por:

- A observação de que todas aquelas classes partilham o mesmo mecanismo geral: concatenação.
- Desiderato de tratar as palavras compostas ao mesmo nível das simples.

Flexão

Virtualmente toda a forma flexionada "simples" – prótotipo da difusa noção de palavra – é uma composição concatenativa dum "radical" lexical com um afixo flexional (geralmente um sufixo). Eis como um ICL representa este fenómeno quanto à forma verbal ⟨escrevo⟩:

$$\begin{bmatrix} Ort \Rightarrow \langle \text{escrevo} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{v} \\ Tem \Rightarrow \text{s} \\ Pes \Rightarrow \text{1} \\ Num \Rightarrow \text{s} \\ Seq \Rightarrow \left\langle \begin{bmatrix} Ort \Rightarrow \langle \text{escrev} \rangle \\ Tip \Rightarrow \text{r} \\ Cat \Rightarrow \text{v} \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{o} \rangle \\ Tip \Rightarrow \text{s} \\ Cat \Rightarrow \text{v} \\ Tem \Rightarrow \text{s} \\ Pes \Rightarrow \text{1} \\ Num \Rightarrow \text{s} \end{bmatrix} \right\rangle \end{bmatrix}$$

As excepções notórias a este esquema são as formas (motivadamente) ditas *irregulares*, tipicamente associadas a verbos de alta frequência e.g. ⟨ser⟩ (vide fórmula abaixo, à esquerda). Neste caso *Seq* é nulo, pois ⟨fui⟩ é uma forma inanalisável. Este item distingue-se do seu homógrafo do verbo "ir" pelo valor de *Can* (fórmula à direita).

$$\begin{bmatrix} Ort \Rightarrow \langle \text{fui} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{v} \\ Tem \Rightarrow \text{s} \\ Pes \Rightarrow \text{1} \\ Num \Rightarrow \text{s} \\ Can \Rightarrow \begin{bmatrix} Ort \Rightarrow \langle \text{ser} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{v} \\ Tem \Rightarrow \text{n} \end{bmatrix} \end{bmatrix} \qquad \begin{bmatrix} Ort \Rightarrow \langle \text{fui} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{v} \\ Tem \Rightarrow \text{s} \\ Pes \Rightarrow \text{1} \\ Num \Rightarrow \text{s} \\ Can \Rightarrow \begin{bmatrix} Ort \Rightarrow \langle \text{ir} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{v} \\ Tem \Rightarrow \text{n} \end{bmatrix} \end{bmatrix}$$

Há também casos envolvendo mais do que um afixo flexional e.g. o adjectivo "escritas" (por exemplo em "palavras escritas"). Este caso pode ser analisado quer como (re)aplicação da flexão a uma forma já flexionada, quer como 'verdadeira' flexão múltipla; respectivamente:

$$\begin{bmatrix} Ort \Rightarrow \langle \text{escritas} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{a} \\ Num \Rightarrow \text{p} \\ Gen \Rightarrow \text{f} \\ Seq \Rightarrow \left\langle \begin{bmatrix} Ort \Rightarrow \langle \text{escrita} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{a} \\ Gen \Rightarrow \text{f} \\ Seq \Rightarrow \left\langle \begin{bmatrix} Ort \Rightarrow \langle \text{escrit} \rangle \\ Tip \Rightarrow \text{r} \\ Cat \Rightarrow \text{a} \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{a} \rangle \\ Tip \Rightarrow \text{s} \\ Gen \Rightarrow \text{f} \end{bmatrix} \right\rangle \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{s} \rangle \\ Tip \Rightarrow \text{s} \\ Num \Rightarrow \text{p} \end{bmatrix} \right\rangle \end{bmatrix}$$

$$\begin{bmatrix} Ort \Rightarrow \langle \text{escritas} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{a} \\ Num \Rightarrow \text{p} \\ Gen \Rightarrow \text{f} \\ Seq \Rightarrow \left\langle \begin{bmatrix} Ort \Rightarrow \langle \text{escrit} \rangle \\ Tip \Rightarrow \text{r} \\ Cat \Rightarrow \text{a} \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{a} \rangle \\ Tip \Rightarrow \text{s} \\ Gen \Rightarrow \text{f} \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{s} \rangle \\ Tip \Rightarrow \text{s} \\ Num \Rightarrow \text{p} \end{bmatrix} \right\rangle \end{bmatrix}$$

Note-se que as duas análises podem coexistir na mesma BCL. Normalmente terão diferentes valores de Ftc. Nesta tese preferimos a primeira.

Concatenação ajustada

Até aqui vimos casos de concatenação pura e simples i.e. em que

$$Ort(i) = Ort(Seq(i)(1)) \ \& \ \ldots \ \& \ Ort(Seq(i)(n))$$

onde & representa a operação de concatenação de cadeias de caracteres usualmente encontrada nas linguagens de programação (espaço branco entre literais e função *strcat* em C, & em Ada, etc.)

Contudo a realidade é bem diferente. São inúmeros os casos em que a afixação determina uma mudança da forma afixada. Por exemplo: A sufixação de ⟨mente⟩ a uma forma adjectival feminina (para a formação do advérbio) resulta no apagamento do eventual acento agudo existente nesta, e.g. ⟨típica⟩ vs. ⟨tipicamente⟩. Certos sufixos verbais apagam mais caracteres finais do radical a que se apensam, do que outros, e.g. ⟨amo⟩ (apagamento até à vogal temática inclusivé) vs. ⟨amarei⟩ (total preservação dos caracteres da forma infinitiva). Certos radicais nominais determinam a inclusão do caracter ⟨e⟩ entre si e o sufixo ⟨s⟩ de flexão de plural e.g. ⟨feliz⟩ vs. ⟨felizes⟩ (cf. *⟨felizs⟩). Estes fenómenos são convenientemente tratados com o auxílio de valores especiais do domínio do `Ort` e.g.

<#>	apagamento dum caracter à esquerda
<'#>	apagamento de todos os acentos à esquerda

em conjunto com uma definição apropriada das regras de afixação, e.g. definir que o radical verbal sujeito a afixação é a forma infinitiva, passando os sufixos a incluir o número necesário de <#>s na sua representação. Esta foi a abordagem, bem sucedida, adoptada em Rocio et al. (2000).

ICL regra

Interessa poder representar formalmente, na BCL, as próprias regras de constituição. Tal é exequível por meio do *ICL regra*, caracterizado enquanto tal por conter valores especiais, principalmente em `Seq` e `Ort`.

Por exemplo, a regra de formação do plural dos adjectivos:

$$\left[\begin{array}{l} Gen \Rightarrow \langle 1 \rangle \\ Seq \Rightarrow \left\langle \left[\begin{array}{l} Tip \Rightarrow x \\ Cat \Rightarrow a \\ Num \Rightarrow s \end{array}\right], \left[\begin{array}{l} Ort \Rightarrow \langle s \rangle \\ Tip \Rightarrow s \\ Num \Rightarrow p \end{array}\right] \right\rangle \end{array}\right]$$

Note-se a ausência de `Ort`, tanto no item principal com no seu primeiro elemento constituinte. Tal significa que `Ort` é nulo. Este é um dos valores especiais referidos acima. Convenciona-se que um ICL com `Ort` nulo é um *ICL variável*. Um ICL variável representa todos ICLs consigo compatíveis i.e. com as mesmas propriedades não nulas, armazenados ou deriváveis na BCL. A própria regra acima, sendo um ICL variável significa adicionalmente que *o valor de `Ort` dos itens resultantes é a concatenação ajustada das `Orts` dos elementos de `Seq` da regra.* Note-se igualmente a ausência de outros atributos morfossintácticos comuns, nomeadamente `Tip`, `Cat` e `Num`. Convenciona-se que *o seu valor é igual ao do afixo, ou, na falta deste, do radical.* Naturalmente, porque o resultado duma afixa-

ção é sempre uma forma flexionada, *no caso do atributo* `Tip`, *dá-se preferência ao valor* x. O valor especial <1> em *Gen* refere o 1° item de *Seq*, com o significado óbvio de que para cada item resultante *x*, *Gen*(*x*) = *Gen*(*Seq*(*x*)(1)). Note-se que o ICL regra é o modo de expressar restrições contextuais de aplicação dos afixos. O afixo acima (2° elemento de *Seq*) é um item da BCL (referenciado pela regra). O item afixo só por si não define o contexto da sua aplicação, nomeadamente concatenação a uma forma flexionada (adjectivo). Tal é dito pela regra.

Derivação

Tradicionalmente, a derivação opõem-se à de flexão por envolver mudança da categoria morfossintáctica: formação de advérbios por sufixação de ⟨`mente`⟩ à forma adjectival feminina e.g. ⟨`tipicamente`⟩, formação deverbal de nomes e.g. ⟨`corredor`⟩, formação denominal de verbos e.g. ⟨`encadernar`⟩, etc. No presente formalismo a descrição de tais fenómenos não se diferencia estruturalmente da flexão. Basta valorar adequadamente os atributos da regra, nomeadamente definindo a nova categoria.

$$\left[Seq \Rightarrow \left\langle \begin{bmatrix} Tip \Rightarrow x \\ Cat \Rightarrow a \\ Num \Rightarrow s \\ Gen \Rightarrow f \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{<\#>mente} \rangle \\ Tip \Rightarrow s \\ Cat \Rightarrow V \end{bmatrix} \right\rangle \right]$$

Note-se a total ausência de atributos morfossintácticos (Ort, Tip, Cat) do item principal. Os seus valores são determinados pelos dos constituintes conforme as convenções acima.

$$\left[Seq \Rightarrow \left\langle \begin{bmatrix} Tip \Rightarrow x \\ Cat \Rightarrow v \\ Tem \Rightarrow n \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{<\#>dor} \rangle \\ Tip \Rightarrow s \\ Cat \Rightarrow n \end{bmatrix} \right\rangle \right]$$

$$\left[Seq \Rightarrow \left\langle \begin{bmatrix} Ort \Rightarrow \langle \text{en} \rangle \\ Tip \Rightarrow p \end{bmatrix}, \begin{bmatrix} Tip \Rightarrow x \\ Cat \Rightarrow n \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{<\#>ar} \rangle \\ Tip \Rightarrow s \\ Cat \Rightarrow v \\ Tem \Rightarrow n \end{bmatrix} \right\rangle \right]$$

Composição

O presente formalismo permite expressar de modo semelhante a composição propriamente dita, nas suas várias formas, nomeadamente: justaposição com e sem intervenção do hífen ou do espaço e aglutinação.

Retiremos primeiro alguma poeira da frente da vista. A tradição gramatical só considera compostas as formas justapostas com ou sem intervenção do *hífen* e.g. ⟨`beija-flor`⟩, ⟨`madrepérola`⟩, e as aglutinadas e.g. ⟨`aguardente`⟩. A composicionalidade semiológica é muito difusa: um *beija-flor* é um *pássaro* caracterizado por 'beijar' flores; não um *qualquer ser* que o faça; *aguardente* é uma

metáfora; etc. Não parece ser possível tratar isto computacionalmente. Pelo menos sem recurso a um corpo de conhecimento enciclopédico e uma simulação robusta dos complexos processos de analogia, simbolismo, preferência, etc. da mente humana. Ora a inteligência artificial, pura e simplesmente, ainda não chegou a esse ponto. Portanto não é isto que nos interessa, mas sim os *aspectos estritamente morfossintácticos*. E aqui, parece que somente os casos de justaposição com intervenção de hífen são relevantes, ao exibirem *padrões de flexão não triviais* i.e. diferentes das formas simples.

As formas aglutinadas são, para os efeitos da presente tese, formas simples. As formas justapostas por intervenção do *espaço* e.g. ⟨greve de fome⟩ aparecem associadas à *terminologia* e a *exploração da informação*, e também têm interesse. Assim, abordamos (exclusivamente) as *formas justapostas com intervenção de hífen ou espaço*, por exemplo *beija-flor, greve de fome, todo-o-terreno*. Note-se que o armazenamento duma forma composta pode ser feito, inicialmente, na ausência de qualquer conhecimento estritamente morfossintáctico sobre a mesma, por exemplo:

$$\begin{bmatrix} Ort \Rightarrow \langle \text{greve de fome} \rangle \\ Ftc \Rightarrow x \\ Xxa \Rightarrow 5 \end{bmatrix}$$

onde *x* denota uma fonte de conhecimento que extraiu o termo a partir dum corpus de texto e representa em *Xxa* a frequência absoluta (número de ocorrências) do termo nesse corpus. Posteriormente, um agente *y* com capacidade de análise morfológica e/ou sintáctica pode identificar a categoria morfossintáctica do termo, gerando o item:

$$\begin{bmatrix} Ort \Rightarrow \langle \text{greve de fome} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{n} \\ Ftc \Rightarrow y \\ Xxa \Rightarrow 5 \end{bmatrix}$$

Chegamos à composição. Assumindo o facto (altamente provável) de o léxico conter os itens relativos aos óbvios termos constituintes ⟨greve⟩, ⟨de ⟩, ⟨fome⟩, o mesmo agente *y*, ou outro, será capaz de identificar as categorias dos mesmos. Este conhecimento representar-se-á, naturalmente, em *Seq*:

$$\begin{bmatrix} Ort \Rightarrow \langle \text{greve de fome} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{n} \\ Num \Rightarrow \text{s} \\ Gen \Rightarrow \text{f} \\ Ftc \Rightarrow y \\ Xxa \Rightarrow 5 \\ Seq \Rightarrow \left\langle \begin{bmatrix} Ort \Rightarrow \langle \text{greve} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{n} \\ Num \Rightarrow \text{s} \\ Gen \Rightarrow \text{f} \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{de} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{p} \end{bmatrix}, \begin{bmatrix} Ort \Rightarrow \langle \text{fome} \rangle \\ Tip \Rightarrow \text{x} \\ Cat \Rightarrow \text{n} \\ Num \Rightarrow \text{s} \\ Gen \Rightarrow \text{f} \end{bmatrix} \right\rangle \end{bmatrix}$$

Eis a morfologia unificada. A teoria volta a brilhar ao permitir uma definição extremamente simples de *palavra composta*: é um item com *mais de um* constituinte *flexionado*.

Visão: grafos tipados com reificação

"O desenvolvedor de software profissional segue procedimentos de engenharia. Num ambiente de linguística computacional, o desenvolvimento é frequentemente mais informal" (Gibbon (2000), p. 28). Mostrei como é possível contrariar esta posição 'artesanal' das ferramentas de investigação em processamento da língua. Identifiquei e criei elementos de engenharia necessários e suficientes para tal: os resultados de trabalhos de grande fôlego (EAGLES), o software *open source* (PostgreSQL, MySQL), o conhecimento das boas práticas (CELEX). Ofereci uma 'colheita', um 'cabaz' desses frutos: POLLUX.

Os principais elementos de POLLUX parecem ser resistentes ao tempo: pude constatar recentemente – com agrado – que se encontram em desenvolvimento e utilização por várias pessoas. Contudo hoje projectaria certas coisas diferentemente. Utilizaria exclusivamente Unicode de 24 bits, por exemplo. E talvez abandonasse mesmo o SQL. De facto, uma nova visão de modelos de dados tem emergido, associada à *Semantic Web* (w3.org). Na minha visão trata-se de grafos tipados com reificação – e, na crista da onda, *via* reificação. Trata-se de representar um sistema não trivial de informação como um tal grafo. A reificação consiste na representação de tipos de ligações como subgrafos ou nós. Na versão *via* esta representação é necessária. As vantagens oferecidas por uma tal visão incluem a possibilidade de predicar sobre ligações, e tipos de ligação, predicando sobre nós, ou subgrafos.

Esta é uma visão informática, à frente dos olhos dum 'engenheiro lexical'. Quero explorar a estrutura geral do novo modelo, mas sei que o mesmo é apropriado à representação de conhecimento lexical.

Agradecimentos

O trabalho apresentado foi realizado, até 2001, no CENTRIA – Centro de Inteligência Artificial (Universidade Nova de Lisboa), com o apoio duma bolsa de mestrado da Fundação para a Ciência e Tecnologia (Ministério da Ciência e Tecnologia do Governo Português), e, desde 2001, no LIACC – Laboratório de Inteligência Artificial e Ciências da Computação (Universidade do Porto).

Referências[2]

Alves (2001)
Safe Web Forms and XML Processing with Ada / Mário Amado Alves. – 349--358 p. – In: Ada-Europe 2001, Leuven, Belgium, May 14-18. – Springer--Verlag, 2001. – (Lecture Notes in Computer Science; 2043)

Alves (2002)
Engenharia do Léxico Computacional: princípios, tecnologia, e o caso das palavras compostas / Mário Amado Alves. – Dissertação de mestrado, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, 2002-02-20. – 57 p. + anexos electrónicos – (http://www.liacc.up.pt/~maa/ELC)

Alves (2003)
The Use of Ada, GNAT.Spitbol and XML in the Sol-Eu-Net Project / Mário Amado Alves. – 12 p. – (Aceite na Ada-Europe 2003, com publicação na Springer-Verlag LNCS.)

EAGLES http://www.ilc.pi.cnr.it/EAGLES/home.html
EAGLES Online: Expert Advisory Group on Language Engineering Standards.

CELEX http://www.kun.nl/celex/
The CELEX Lexical Database: Dutch, English, German / CELEX, The Dutch Centre for Lexical Information.

Gibbon (2000)
Computational Lexicography / Dafydd Gibbon. – 1-42 p. – In: Lexicon Development for Speech and Language Processing / Edited by Frank Van Eynde and Dafydd Gibbon. – Dordrecht [...]: Kluwer Academic Publishers, 2000. – (Text, Speech and Language Technology; Volume 12) – (ISBN 0-7923-6368-8)

Rocio et al. (2000)
Automated Creation of a Partial Treebank of Medieval Portuguese / Vitor Rocio; Mário Amado Alves; José Gabriel P. Lopes; Maria Francisca Xavier; Graça Vicente. – 17 p. – In: Building and Using Syntactically Annotated Corpora / Anne Abeillé (ed.) – Kluwer Academic Publishers: Dordrecht, 2000. – (Language and Speech Series)

[2] Os items com um endereço emoldurado são sítios web. Os outros items podem ter endereços, mas são primariamente documentos tradicionais.