Utilização de XML para Desenvolvimento Rápido de Analisadores Morfológicos Flexíveis

Bruno Oliveira[1], Carlos Pona[1], Ricardo Ribeiro[2], and David Martins de Matos[1]

[1] L[2]F/INESC-ID Lisboa/IST
[2] L[2]F/INESC-ID Lisboa/ISCTE
Rua Alves Redol 9, 1000-029 Lisboa, Portugal
{bfso,cmpo,rdmr,david}@inesc-id.pt
http://www.l2f.inesc-id.pt/

Resumo Descreve-se a utilização de um repositório lexical e de uma ferramenta de geração morfológica na produção rápida de ferramentas de análise morfológica. A ferramenta criada a partir dos dados XML, denominada XISPA (XML fInite-State-Powered Analyzer), utiliza a tecnologia de autómatos de estados finitos para a realização do processo de análise morfológica. O processo de criação do XISPA beneficiou da estrutura clara e bem definida dos dados produzidos pelo Monge. O uso de XML garantiu a independência relativamente ao processo de gestão dos dados linguísticos, permitindo um elevado nível de flexibilidade na decisão de construção de ferramentas seguindo este processo. Outro aspecto positivo a relevar é o curto intervalo de tempo necessário à produção de um protótipo (o XISPA foi completamente implementado em menos de um dia).

1 Introdução

O uso de ferramentas de processamento de língua natural é cada vez mais ubíquo, seja num simples corrector ortográfico, seja num sistema de diálogo. As aplicações existentes actualmente trabalham, cada uma, sobre um formato de dados específico, havendo a necessidade de introduzir ferramentas de conversão que facilitem a portabilidade da informação linguística. Coloca-se, assim, o problema da manutenção de colecções de dados adequados às diferentes necessidades daquelas aplicações. O Repositório Lexical do Laboratório de Sistemas de Língua Falada do INESC ID Lisboa foi criado para facilitar a gestão e manutenção de dados linguísticos, permitindo a representação de uma informação rica sobre vários níveis de descrição da língua. Este repositório foi definido para ser independente das aplicações que dele fazem uso. Este aspecto levanta, contudo, o problema da adaptabilidade dos dados em utilizações concretas: é necessária a definição de formas e linguagens de interoperação. A ferramenta Monge (Morphological Generator) surge neste contexto: esta ferramenta interage com o Repositório Lexical ao nível da produção e extracção de informação morfológica. O Monge extrai informação morfológica do Repositório e produz um documento XML. A representação em XML garante a independência de processamentos subsequentes e, devidamente validada, garante, em simultâneo, a coerência da informação resultante.

Neste documento descreve-se a utilização de um repositório lexical e de uma ferramenta de geração morfológica (Monge) na produção rápida de ferramentas de análise

morfológica (ferramentas que expressam o radical e conjunto de traços da forma de entrada). A ferramenta criada a partir dos dados XML, denominada XISPA (XML fInite-State-Powered Analyzer), utiliza a tecnologia de autómatos de estados finitos para a realização do processo de análise morfológica.

O documento tem a seguinte estrutura: primeiro apresenta-se o repositório lexical, fonte de dados para as ferramentas descritas; de seguida, descreve-se o gerador morfológico Monge, produtor da informação para a construção do XISPA. A descrição do XISPA, da sua construção, assim como conclusões acerca dos resultados obtidos e comparação com outras ferramentas semelhantes, terminam o texto.

2 Repositório Lexical

A área de processamento da língua apresenta frequentemente o problema bizarro de haver recursos para utilizar, mas não ser possível efectuar cabalmente esse uso. Tal acontece, ou porque os dados codificam informação que não corresponde exactamente às necessidades, ou porque, mesmo havendo essa correspondência, o formato dos dados não é o que seria útil. Estas variações conduzem a situações de incompatibilidade de dados entre ferramentas que realizam a mesma função e impede que os dados de uma possam ser reutilizados noutra, ou até mesmo a simples combinação desses dados para enriquecimento mútuo das diferentes aplicações.

O repositório de dados linguísticos multiusos do L^2F [1,2], é uma solução para o problema da compatibilização e reutilização de dados linguísticos: é capaz de armazenar dados pertencentes a diferentes paradigmas e expressos originalmente em diferentes formatos e cobrindo vários níveis (e.g., morfologia, sintaxe e semântica). Permite ainda expandir a cobertura ou a capacidade de descrição com impacto mínimo nos dados representados. O repositório está descrito em UML/XMI [3], sendo o código a ele associado gerado de forma inteiramente automática. A figura 1 mostra vários exemplos de dados.

Além da capacidade básica de armazenamento, o repositório funciona como ponte entre várias representações de dados, indo além da mera definição de um modelo canónico, definindo também transformações de dados entre vários modelos externos. Estes agentes de tradução constituem um factor positivo na consideração de utilização do repositório numa aplicação. As ferramentas especializadas, das quais as traduções são exemplos, podem ser utilizadas para preparar dados para uso por aplicações previamente existentes e que podem passar a utilizar dados enriquecidos; as aplicações podem ainda recorrer sem intermediários à representação nativa dos dados no repositório, fazendo uso directo dos dados nele armazenados. A figura 2 apresenta os vários grupos de modelos que coexistem para governar o repositório.

O repositório e o seu conteúdo contribuem, assim, tanto para o enriquecimento dos dados utilizados por aplicações existentes (enriquecendo, deste modo, as próprias aplicações), como para a possibilidade de reutilização de dados em aplicações que, de outra forma, não lhes teriam acesso.

PAROLE [4]

```
<mus id="r592" naming="algo" gramcat="adverb"
     autonomy="yes" synulist="usyn23987 usyn23988">
  <gmu range="0" reference="yes" inp="mfgr1">
    <spelling>algo</spelling>
  </gmu>
</mus>
<mus id="pi1" naming="algo" autonomy="yes"
     gramcat="pronoun" gramsubcat="indefinite"
     synulist="usyn23320">
  <gmu range="0" reference="yes" inp="mfgempty">
    <spelling>algo</spelling>
  </gmu>
</mus>
<ginp id="mfgr1" example="abaixo">
  <combmfcif combmf="combtm0">
    <cif stemind="0">
      <removal/><addedbefore/><addedafter/>
    </cif>
  </combmfcif>
</ginp>
<ginp id="mfgempty" comment="empty Mfg">
  <combmfcif combmf="combtmempty">
    <cif stemind="0">
      <removal/><addedbefore/><addedafter/>
    </cif>
  </combmfcif>
</ginp>
<combmf id="combtmempty"/>
<combmf id="combtm0" degree="positive"/>
```

SMorph [5]

```
algo            /pr_i/s/GEN:*/pri .
```

LUSOlex [6]

```
Adv191 <algo> ADVÉRBIO - FlAdv2 <algo>
Pi1 <algo> PRONOME INDEFINIDO - <algo>
FlAdv2  <abaixo>
        __P____ 0       <><>
$
```

EPLexIC [7]

```
algo/R=p/"al~gu/algo
algo/Pi=nn/"al~gu/algo
```

Figura 1. Diferenças entre léxicos na descrição de *algo*.

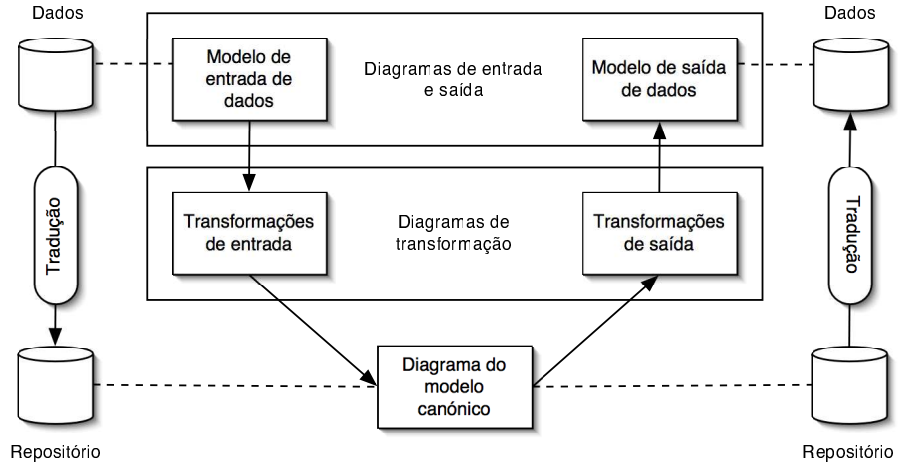

Figura 2. Modelos e fluxos de dados no repositório de recursos linguísticos.

3 Monge

A ferramenta Monge é um gerador morfológico que permite, com base numa raiz e num conjunto de características a ela associadas, obter uma forma (ver figura 3).

3.1 Relação com os dados

O Monge depende intrinsecamente do modelo de dados do Repositório Lexical. Não está, portanto, condicionado pela estrutura de uma base de dados particular. Pode, assim, funcionar com diferentes bases de dados, desde que estas sigam o modelo canónico. A utilização do Monge e da saída XML por ele produzida, como intermediária da representação constante na base de dados, faz com que as aplicações clientes dos dados sejam independentes do modelo do repositório.

Salienta-se também o facto de o Monge transformar os dados em formato relacional, sem estruturação explícita (como tal, de difícil utilização), num modelo estruturado (XML), em que estão patentes as definições subjacentes ao modelo de objectos representado. A utilização de XSD permite, simultaneamente, descrever e validar a estrutura dos dados de saída, facilitando o processo de exportação dos dados.

3.2 Modo de Funcionamento

O Monge recebe como entradas uma raiz e um conjunto opcional de categorizações e/ou traços – categoria, subcategoria, ou qualquer característica definida no dicionário de tipos. Com base nessa raiz, o Monge obtém todas as formas gráficas correspondentes, de modo a poder obter as bases de flexão regular da raiz: são estas bases de flexão, juntamente com a raiz que darão origem à forma flexionada. A forma gráfica da raiz

é também usada para recolher informação acerca da sua categoria e sub-categoria gramaticais. Por fim, o Monge selecciona os paradigmas de flexão apropriados para as formas em causa e aplica as transformações neles descritas. No final do processo, as palavras flexionadas a partir das respectivas raízes são agrupadas pelas suas categorias gramaticais. Posteriormente, esta informação é transformada numa árvore XML, em que às palavras são associados os traços morfológicos correspondentes. Para efeitos de validação e de organização da estrutura, um esquema XML (XSD) é associado ao documento produzido. Actualmente, o Monge está implementado na linguagem Perl e utiliza os módulos DBI, para acesso à base de dados; utiliza a biblioteca Xerces-p para representar e gerar informação XML.

```
$ ./monge.pl -x ser number singular gender masculine
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
 <monge:lemma
    xmlns:monge="http://www.l2f.inesc-id.pt/~bfso/monge.xsd"
    value="ser">
   <morph cat="NOUN" subcat="COMMON">
     <form value="ser">
       <feature given="yes" name="number" value="SINGULAR"/>
       <feature given="yes" name="gender" value="MASCULINE"/>
     </form>
   </morph>
   <morph cat="VERB" subcat="MAIN">
     <form value="sido">
       <feature given="yes" name="number" value="SINGULAR"/>
       <feature given="no" name="mood" value="PARTICIPLE"/>
       <feature given="yes" name="gender" value="MASCULINE"/>
     </form>
   </morph>
 </monge:lemma>
```

Figura 3. Exemplo de execução do Monge para a raiz *ser* sem restrições à categorização. Note-se a produção de todas as formas para todas as categorias e subcategorias.

4 XISPA

O XISPA é um analisador morfológico baseado em tecnologia de estados finitos. Dada uma palavra, o XISPA verifica se ela existe e, caso tal aconteça, devolve o conjunto de pares traço/valor que lhe está associado. Assim, dada a palavra *anestesiando*, o resultado devolvido pelo XISPA seria como indicado abaixo.

lemma	**cat**	**subcat**	**mood**
anestesiar	VERB	MAIN	GERUND

Para realizar o processo de análise, o XISPA usa a informação gerada pelo Monge, na forma de uma máquina de estados finitos. A ideia subjacente a esta realização é a de facilitar a construção deste tipo de ferramentas a partir de dicionários de larga cobertura.

Actualmente, as ferramentas de análise morfológica implementam esse processo através autómatos ou transdutores de estados finitos, sendo uma das referências fundamentais o trabalho de Koskenniemi [8,9]. Tal acontece porque o tempo de análise é $O(n)$ (n é o comprimento da palavra a analisar), independentemente do número de palavras do dicionário. A desvantagem a sublinhar é a quantidade de memória ocupada. Quanto ao tipo de informação usada pelos sistemas de estados finitos, a escolha entre uma abordagem baseada em paradigmas e uma baseada em regras morfológicas está relacionada com a língua que se pretende tratar: para o Português é comum a abordagem paradigmática.

Os sistemas mais comuns são sistemas especializados, i.e., seguem o processo de desenvolvimento tradicional e apresentam uma complexidade de desenvolvimento consideravelmente superior à que aqui se descreve. Acrescem a estas dificuldades, a manutenção, quer do código desenvolvido, quer dos dados usados, sendo especialmente difícil a adaptação a contextos variados. Como exemplos de analisadores baseados em estados finitos, podemos citar o sistema SMORPH [5] e o utilizado no Xerox Research Center Europe (XRCE)[3].

Mokhtar [5] descreve um sistema de análise morfológica baseado num autómato de estados finitos determinista denominado SMORPH, independente da língua. O ficheiro de informação necessária para efectuar a análise do Português tem cerca de 1 MByte, cobrindo cerca de 900 mil formas. No XRCE, o sistema de processamento morfológico (disponível para várias línguas) é baseado em transdutores de estados finitos. Apesar de não estarem disponíveis dados concretos sobre o sistema, o XRCE salienta a rapidez e a (reduzida) dimensão dos dados usados.

4.1 Construção

O processo de construção do XISPA pode ser observado na figura 4: os dados XML produzidos pelo Monge são processados por forma a especificar um autómato finito contendo todas as formas conhecidas para uma dada língua. Note-se que este autómato não contém em si mais do que a selecção do átomo (*token*) a apresentar (um número inteiro): a transposição da saída do autómato para o conjunto de características correspondentes à forma de entrada é realizado separadamente. Note-se que o autómato não codifica nenhum tipo de ambiguidade relativamente às formas reconhecidas: a ambiguidade é codificada nos átomos apresentados na saída, o que produz resultados equivalentes. O facto de o autómato não codificar ambiguidade significa, no entanto, que a sua utilização é mais simples.

O processo de construção do XISPA ignora ainda outros aspectos tradicionalmente relacionados com a análise morfológica, nomeadamente, a segmentação da entrada em átomos a analisar e a captura de elementos regulares (e.g., números). Estes aspectos não se reflectem, no entanto, negativamente, uma vez que é possível delegar estas tarefas

[3] `http://www.xrce.xerox.com`

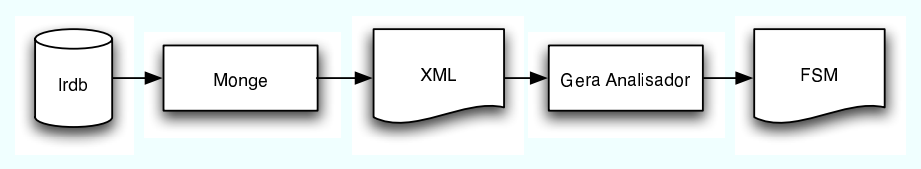

Figura 4. Processo de construção do XISPA (autómato finito).

noutras ferramentas e, assim, conseguir também distribuir as responsabilidades (ver figura 5).

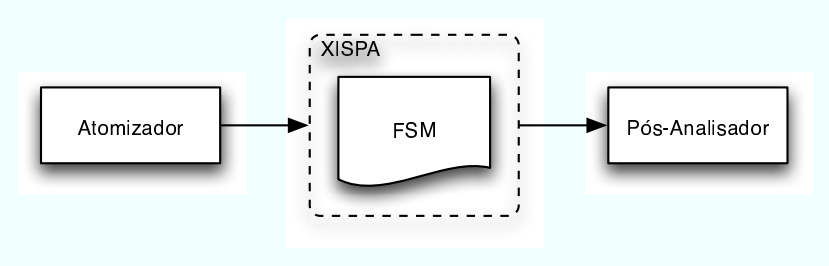

Figura 5. Processo de utilização do XISPA numa cadeia de análise morfológica: o atomizador segmenta a entrada em elementos a analisar; o módulo de pós-análise permite complementar a análise produzida.

Dada a simplicidade de construção, foram feitas várias implementações do XISPA, desde as mais ingénuas e limitadas (em dimensão, mas não funcionalidade), utilizando a ferramenta "flex",[4] até às "fsmtools" [10] da AT&T.[5]

Outras são possíveis, desde que suportem o conceito de autómato finito.

5 Conclusões

O processo de criação do XISPA beneficiou da estrutura clara e bem definida dos dados produzidos pelo Monge. O uso de XML garantiu a independência relativamente ao processo de gestão dos dados linguísticos, permitindo um elevado nível de flexibilidade na decisão de construção de ferramentas seguindo este processo. Outro aspecto positivo a relevar é o curto intervalo de tempo necessário à produção de um protótipo: o XISPA foi completamente concebido e implementado em menos de um dia.

A utilização da tecnologia de estados finitos garante tempos de análise semelhantes aos de ferramentas especializadas. O processo de construção flexibiliza a definição

[4] `http://www.gnu.org/software/flex/`
[5] `http://www.research.att.com/sw/tools/fsm/`

dos dados para a ferramenta, sendo um factor muito importante na consideração de utilizações em novos contextos.

Em §4.1 já se mencionaram algumas diferenças entre o XISPA e as ferramentas às quais pode ser equiparado. Um aspecto notório é a completa ausência de suporte ao tratamento de palavras desconhecidas pelo dicionário. Este aspecto, como já foi referido, pode ser delegado em ferramentas auxiliares especializadas, ficando a gestão do reconhecimento e classificação livre desta responsabilidade.

Como nota final, referimos a facilidade de aplicação do XISPA a outras línguas, uma característica que partilha com outras ferramentas baseadas em estados finitos.

Agradecimentos

Este trabalho foi parcialmente financiado pelo Projecto NLE-GRID (POSC/PLP/60663/2004).

Referências

1. de Matos, D.M., Ribeiro, R., Mamede, N.J.: Rethinking Reusable Resources. In: Proceedings of the Fourth Language Resources and Evaluation Conference – LREC 2004, Lisboa, Portugal, ELRA – European Language Resources Association (2004) 357–360 ISBN 2-9517408-1-6.
2. Ribeiro, R., de Matos, D.M., Mamede, N.J.: How to Integrate Data from Different Sources. In: A Registry of Linguistic Data Categories within an Integrated Language Resources Repository Area (LREC 2004), Lisboa, Portugal, ELRA – European Language Resources Association (2004)
3. Booch, G., Rumbaugh, J., Jacobson, I.: The Unified Modeling Language User Guide. Addison-Wesley Longman, Inc. (1999) ISBN 0-201-57168-4.
4. PAROLE: Preparatory Action for Linguistic Resources Organisation for Language Engineering – PAROLE (1998) O projecto teve início em Abril de 1996 e a duração de 24 meses. Ver `http://www.hltcentral.org/projects/detail.php?acronym=PAROLE` para um sumário do projecto.
5. Aït-Mokhtar, S.: L'analyse présyntaxique en une seule étape. Thèse de doctorat, Université Blaise Pascal, GRIL, Clermont-Ferrand (1998)
6. Wittmann, L., Ribeiro, R., Pêgo, T., Batista, F.: Some Language Resources and Tools for Computational Processing of Portuguese at INESC. In: Proceedings of the Second International Conference on Language Resources and Evaluation (LREC 2000), Athens, Greece, ELRA – European Language Resources Association (2000) 347–350
7. de Oliveira, L.C.: EPLexIC – European Portuguese Pronunciation Lexicon of INESC-CLUL. documentation (n.d.)
8. Koskenniemi, K.: Two-level morphology: A general computational model for word-form recognition and production. Publications (11) (1983)
9. Koskenniemi, K.: Two-level model for morphological analysis. In Bundy, A., ed.: Proceedings of the Eighth International Joint Conference on Artificial Intelligence, EUA (1983) 683–685
10. Mohri, M.: Finite-State Transducers in Language and Speech Processing. Computational Linguistics **23**(2) (1997)