TRATAMENTO FORMAL DA SINTAXE PORTUGUESA

Johann Haller *

Resumo

Este artigo mostra as partes e funções de um programa para análise morfológica e sintática, aplicado a uma frase em língua portuguesa. A análise serve para fins de correção gramatical, para sistemas de informação e de tradução automática. Ao mesmo tempo, é utilizada como instrumento didático em cursos de Linguística Computacional no Mestrado da UFC.

***Palavras-chave**: Análise morfológica; análise sintática; gramáticas formalizadas; tradução automática; linguistica computacional.*

Abstract

This article features a system for morphological and syntactic analysis, with an example for a sentence in Portuguese language. The analysis serves for spelling and grammar checking, for informations systems and machine translation. At the same time, it is used forA didactical purposes in courses of Computational Linguisticas for Post-graduates at the Federal University of Ceará.

***Key words**: morphological analisys; syntatic analisys; formal grammars; machine translation; computational linguistics.*

INTRODUÇÃO: AS GRAMÁTICAS FORMALIZADAS

As gramáticas formalizadas começaram a ganhar fama mundial nos anos 50 do nosso século, embora já houvesse tentativas de usar métodos distributivos e matemáticos bem antes, sendo os mais famosos os chamados "Junggrammatiker" na Alemanha, e os distribucionalistas americanos e ingleses dos anos 30. Coincidentemente com as regras de expansão sintática de Chomsky, mas muitas vezes sem conhecimento deles, os especialistas da nova era da computação começaram a formar regras em todos os níveis lingüísticos de maneira que pudessem ser interpretadas por programas de computador. Uma destacada qualidade deste procedimento consiste no fato de que estas regras podiam ser verificadas (ou falsificadas) em grandes quantias de textos, levando a discussões diferentes das tradicionais, orientadas sempre num punhado de exemplos que podiam facilmente ser contestados por outro punhado de exemplos novos. Assim sendo, ganharam valor as regras que se aplicam com sucesso a uma maior porcentagem de exemplos vindas de um "corpus" mais ou menos representativo. Não queremos, neste pequeno artigo, discutir os pros e contras deste procedimento – existem lingüistas, hoje, que afirmam ser ele o único modo de se fazer pesquisa lingüística no mundo. Não queremos assumir esta posição embora deva se reconhecer que a obrigação de escrever regras logicamente corretas e detalhadas sempre leva a uma compreensão melhor dos fatos lingüísticos existentes.

Queremos, no entanto, ilustrar o método com exemplos tomados da língua portuguesa. Esta língua somente foi tratada formalmente dentro de alguns projetos de tradução automática, como os da IBM, ou então no EUROTRA, projeto da Comunidade Européia, que, por ser puramente acadêmico, não produziu nenhum sistema comercial, mas muitas páginas valiosas contemplando fatos lingüísticos de todas as línguas européias (e não só do inglês e de algumas outras línguas ditas "maiores"). Os recursos computacionais do português acham-se reunidos numa página da Internet, e constam também algumas (não todas) instituições brasileiras onde se trabalha no processamento automático de textos da língua vernácula (http://www.portugues.mct.pt/recursos.html#ferr). Em alguns lugares no Brasil onde achamos esta atividade pode-se dizer que estes trabalhos foram iniciados por ex-alunos do autor deste artigo.

* Universidade de Saarbrücken/RFA, Email: hans@iai.uni-sb.de

Também, o autor continua trabalhando com esta língua na universidade alemã de Saarbrücken onde é diretor do IAI, um instituto de pesquisa que trabalha muito com a própria língua alemã com inglês e o francês, porque são estas as línguas em que o processamento de documentação técnica acha mais aplicação. Projetos de corretores automáticos, de extração de terminologia etc. fornecem a motivação (e a necessária base econômica) para continuar com os trabalhos de dicionários, gramáticas e regras de tradução entre uma boa parte das línguas da Comunidade Européia e outras. Como há vários projetos de grande investimento no Brasil, por parte da indústria automobilística e de outros ramos, espera-se que isto aconteça também com a inclusão do português do Brasil. (veja www.iai.uni-sb.de).

Do ponto de vista brasileiro, a pesquisa em lingüística computacional seria perfeitamente adequada à situação dada no país: não precisa de investimentos altos em tecnologia (já que existem hoje laboratórios modernos em muitas universidades), há uma grande tradição em letras e lingüística, e, com isso, uma boa oferta de mão de obra, e as aplicações existem em grande número.

PROCESSAMENTO DE DOCUMENTOS E ANÁLISE MORFOLÓGICA

Para o computador, um texto representa nada mais do que uma cadeia de caracteres alfanuméricos e outros, por exemplo, espaços em branco que podem separar uma palavra da próxima. Queda imediatamente claro que isto não é o único critério porque pontuações costumam ser ´coladas´ diretamente nas palavras. Da mesma maneira, não é somente o ponto (ou ponto de interrogação ou de exclamação) que separa uma frase da outra; títulos, muitas vezes, costumam ser escritos sem nenhuma pontuação. Encontrando um ponto, porém, não garante o fim de uma frase: pode este ponto pertencer a um número, ser um ponto de suspensão ou ainda ter outras interpretações. Em muitas línguas, ademais, existem abreviações que se caracterizam pela terminação em ponto: “etc.” e outras. Nestes casos, torna-se necessário incluir uma análise morfológica e consulta ao dicionário para saber se esta palavra e uma abreviação, se a palavra seguinte pode ser o começo de uma nova frase e assim em diante. Por conseqüência, os programas que fazem esta tarefa (isolar as unidades de análise dentro do texto, interligado com análise morfológica) já foram uma das partes mais complicadas nos programas comerciais de tradução (o mais antigo deles, o SYSTRAN, inclui já desde uma década a língua portuguesa, um mais moderno, o LOGOS, acaba de lançar o par inglês-português no começo do ano 2000).

Segue-se um exemplo do resultado do programa correspondente do IAI, aplicado ao seguinte parágrafo:

Somos originários de mais 2100 grupos étnicos, raciais e tribais e somamos em torno de cinco milhões de pessoas no mundo inteiro.

{ori=Somos,wnra=26,wnrr=1,snr=2,string=somos,lu=ser,vtyp=fiv,c=verb,tns=pres,mode=ind,per=1,nb=plu,pctr=no,last=no,pctl=no,gra=cap}
{ori=originários,wnra=27,wnrr=2,snr=2,pctr=no,last=no,pctl=no,gra=small,nb=plu,g=m,lu=originário,c=adj}
{ori=de,wnra=28,wnrr=3,snr=2,string=de,c=p,lu=de,pcomp=no,pctr=no,last=no,pctl=no,gra=small}
{ori=mais,wnra=29,wnrr=4,snr=2,string=mais,lu=mais,c=adv,pctr=no,last=no,pctl=no,gra=small}
{ori=mais,wnra=29,wnrr=4,snr=2,string=mais,lu=mais,c=part,sc=card;z,pctr=no,last=no,pctl=no,gra=small}
{ori=2100,wnra=30,wnrr=5,snr=2,c=z,lu=2100,s=integer,gra=digits,pctr=no,last=no,pctl=no}
{ori=grupos,wnra=31,wnrr=6,snr=2,pctr=no,last=no,pctl=no,gra=small,lu=grupo,c=noun,ehead={nb=plu,g=m}
{ori=étnicos,wnra=32,wnrr=7,snr=2,pctr=yes,last=no,pctl=no,gra=small,nb=plu,g=m,lu=étnico,c=adj}
{ori=&cm,wnra=33,wnrr=8,snr=2,string=&cm,lu=comma,c=punct,pctr=no,last=no,pctl=no,gra=other}
{ori=raciais,wnra=34,wnrr=9,snr=2,pctr=no,last=no,pctl=yes,gra=small,nb=plu,lu=racial,c=adj}
{ori=e,wnra=35,wnrr=10,snr=2,string=e,c=w,sc=coord,pctr=no,last=no,pctl=no,gra=small,lu=e}
{ori=tribais,wnra=36,wnrr=11,snr=2,pctr=no,last=no,pctl=no,gra=small,nb=plu,lu=tribal,c=adj}
{ori=e,wnra=37,wnrr=12,snr=2,string=e,c=w,sc=coord,pctr=no,last=no,pctl=no,gra=small,lu=e}
{ori=somamos,wnra=38,wnrr=13,snr=2,pctr=no,last=no,pctl=no,gra=small,vtyp=fiv,tns=pres,mode=ind,per=1,nb=plu,lu=somar,c=verb}
{ori=em_torno_de,wnra=39,wnrr=14,snr=2,string=em_torno_de,lu=em_torno_de,c=part,s=card,pctr=no,last=no,pctl=no,gra=small}
{ori=em_torno_de,wnra=39,wnrr=14,snr=2,string=em_torno_de,lu=em_torno_de,c=p,pctr=no,last=no,pctl=no,gra=small}
{ori=cinco,wnra=40,wnrr=15,snr=2,string=cinco,lu=5,c=card,nb=plu,pctr=no,last=no,pctl=no,gra=small}
{ori=milhões,wnra=41,wnrr=16,snr=2,pctr=no,last=no,pctl=no,gra=small,lu=milhão,s=quant,c=noun,ehead={nb=plu,g=f}
{ori=de,wnra=42,wnrr=17,snr=2,string=de,c=p,lu=de,pcomp=no,pctr=no,last=no,pctl=no,gra=small}
{ori=pessoas,wnra=43,wnrr=18,snr=2,pctr=no,last=no,pctl=no,gra=small,lu=pessoa,c=noun,ehead={nb=plu,g=f}}
{ori=no,wnra=44,wnrr=19,snr=2,string=no,pcomp=yes,c=p,lu=em,nb=sg,g=m,pctr=no,last=no,pctl=no,gra=small}
{ori=mundo,wnra=45,wnrr=20,snr=2,pctr=no,last=no,pctl=no,gra=small,lu=mundo,c=noun,ehead={nb=sg,g=m}}
{ori=inteiro,wnra=46,wnrr=21,snr=2,pctr=yes,last=no,pctl=no,gra=small,nb=sg,g=m,lu=inteiro,c=adj}
{ori=.,wnra=47,wnrr=22,snr=2,string=.,lu=.,c=w,sc=punct,pctr=no,last=yes,pctl=no,gra=other}
(MAAS 98)

Torna-se necessária uma administração rigorosa dos resultados desta análise. Principalmente, isto se faz através dos traços `snr` (*sentence number*, aqui sempre igual a 2, dizendo que é a segunda frase do texto), `wnra` (*word number absolute*, dizendo a posição absoluta da palavra no texto) e `wnrr` (*word number relative*, dizendo a posição da palavra na frase). Observe-se que os dois últimos têm igual valor quando uma palavra pode ser interpretada de duas maneiras (exemplos: `mais` como *advérbio* ou *partitivo*: duas vezes 29 e. 4, e `em_torno_de` como *preposição local* ou *partitivo*: duas vezes 39 e. 14.). O segundo exemplo mostra que unidades de análise podem consistir de várias palavras quando estas formam uma unidade lexical indissolúvel – como é o caso das locuções preposicionais do português.

`Pctr` e `pctl` servem para indicar se o programa encontrou uma pontuação imediatamente do lado esquerdo ou direito da palavra, `last` marca a última palavra da frase – uma qualidade muitas vezes útil para tomar decisões em casos de homografía. `Gra` indica se a palavra começa com uma maiúscula, ou então se ela contém uma mistura de maiúsculas, minúsculas ou dígitos (como em `2100`).

`Lu` significa lexical unit e contem a forma lematizada da palavra tal como ela aparece num dicionário: o singular dos substantivos, o singular masculino dos adjetivos e o infinitivo dos verbos.

Passamos então a contemplar as indicações sintáticas, sendo `c` a categoria (`noun, verb, adj, card, adv, w` – este último servindo para preposições, coordenações etc.) e `sc` a possível subcategoria (`card, punct, vtyp,` etc.). Alguns traços especiais necessários para a posterior análise sintática completam as informações obtidas pela análise gráfica e morfológica de frase e palavra: número, pessoa, modo e tempo verbal, gênero e número para adjetivos e substantivos (nestes últimos aparecem agrupados no `ehead`, de acordo com as teorias gramaticais de projeção). Finalmente, nas proposições do português (como em algumas outras línguas românicas) distingue-se o fato da preposição ser composta com o artigo (`pcomp`).

Para obter-se todas estas informações, é necessário um dicionário dos morfemas da língua portuguesa que contenha (entre outras) as seguintes entradas:

```
{string=originári,lu=originário,c=a,t=def}
{string=som,lu=somar,c=v,t={mt=a,s=normal}}
{string=da,c=p,lu=de,nb=sg,g=f,pcomp=yes}

{string=os,c=flex,l=def,m={nb=plu,g=m}}
{string=a,c=flex,l={mt=a,s=normal},m={vtyp=fiv,tns=pres,mode=ind,per=3,nb=sg}}
{string=s,c=flex,l=e;normal,m={nb=plu},mod={ão>õe,em>en,il>i}}
{string=milhão,lu=milhão,c=n,g=f,t=normal,s=quant}
```

Em combinação com um programa, escrito na (eficiente) linguagem C, que tenta identificar as unidades listadas no dicionário, consegue-se atribuir a uma boa porcentagem das palavras de um texto comum a (ou as) classificação correspondente – geralmente a mais de 90 por cento.

No caso de uma palavra desconhecida, é fácil para um lingüista adicionar os morfemas que faltem. Casos como plurais que mudam o ditongo final são tratadas com entradas como as duas últimas.

Por razões sistemáticas, as expressões que consistem de várias palavras são incluídas nos dicionários bilíngües. Isto é o caso de `em_torno_de,` a ser traduzido para o alemão como `um.` Para as línguas que têm longas listas de locuções seria talvez mais econômico tratá-las como entradas monolinguais mas há que se considerar que muitas vezes estas expressões se traduzem ao pé da letra, por exemplo, a outras línguas românicas. De qualquer forma, aqui se vê claramente a origem e a motivação dos procedimentos aplicados que era a finalidade de um sistema de tradução multilingual:

```
{string=em_torno_de,c=part,de=um}
{string=em_torno_de,c=p,de=um}
```

RESOLUÇÃO PARCIAL DE HOMOGRAFÍAS E AGRUPAMENTO SUPERFICIAL

Mediante a aplicação de uma gramática (porém organizada de uma maneira procedural, veja a seguir), obtêm-se os grupos sintáticos da frase, neste caso até o nível supremo de `hs (head sentence node)`. Quando isto não for possível, agrupam-se no mínimo os grupos nominais e verbais. Com isto, fica também resolvida, na maioria das vezes, a questão dos homógrafos; foi decidido aqui que **mais** não é advérbio senão um partitivo diante de um número e dentro de um grupo nominal.

A gramática é constituída por grupos de regras numeradas (por exemplo, 336 para um grupo preposicional simples) que se aplicam segundo um procedimento bem determinado. Analisam-se primeiro umas construções especiais, como datas etc., em segundo lugar grupos mais longos (por serem provavelmente a solução certa), e finalmente as palavras restantes em grupos mínimos.

```
<1> to <22>:
 {c=hs,r=325,coord=yes,snr=2}
    {c=hs,r=447,snr=2}
       {ori=Somos,lu=ser,vtyp=fiv,c=verb,tns=pres,mode=ind,per=1,nb=plu}
       {ori=originários,nb=plu,g=m,lu=originário,c=adj}
       {c=pp,r=336,prplu=de,snr=2}
          {ori=de,c=p,lu=de,pcomp=no}
          {c=np,r=331,snr=2}
             {c=adj,r=119a,snr=2}
                {ori=mais,lu=mais,c=part,sc=card;z}
                {ori=2100,c=z,lu=2100,sc=integer}
             {ori=grupos,lu=grupo,c=noun, ehead={nb=plu,g=m}}
             {c=adj,r=20,adk=complex,snr=2}
                {ori=étnicos,pctr=yes,nb=plu,g=m,lu=étnico,c=adj}
                {ori=&cm,lu=comma,c=punct}
                {ori=raciais,pctl=yes,nb=plu,lu=racial,c=adj}
                {ori=e,c=w,sc=coord,lu=e}
                {ori=tribais,nb=plu,lu=tribal,c=adj}
    {ori=e,c=w,sc=coord,lu=e}
    {c=hs,r=342,snr=2}
       ori=somamos,vtyp=fiv,tns=pres,mode=ind,per=1,nb=plu,lu=somar,c=verb}
       {c=np,r=303,fu=np,s=quant,snr=2}
          {c=adj,r=119a,snr=2}
             {ori=em_torno_de,lu=em_torno_de,c=part,s=card}
             {ori=cinco,lu=5,c=card,nb=plu}
          {ori=milhões,lu=milhão,s=quant,c=noun,ehead={nb=plu,g=f}}
       {c=pp,r=336,prplu=de,snr=2}
          {ori=de,c=p,lu=de,pcomp=no}
          {ori=pessoas,lu=pessoa,c=noun, ehead={nb=plu,g=f}}
       {c=pp,r=214,prplu=em,snr=2}
          {ori=no,pcomp=yes,c=p,lu=em,nb=sg,g=m,ds=em,ls=em}
          {c=np,r=28,fu=np,ehead={nb=sg,g=m},nb=sg,g=m,snr=2}
             {ori=mundo,lu=mundo,c=noun, ehead={nb=sg,g=m}
             {ori=inteiro,nb=sg,g=m,lu=inteiro,c=adj}
<22> to <23>:
 {ori=.,lu=.,c=w,sc=punct}
```

Não tem muito risco de erro numa regra como a de `horas`:

```
{c=pp,r=001,prplu=a}[{c=p,lu=a},{c=card}, {c=noun,ori=horas}]
```

O mesmo vale para uma regra que determina um grupo nominal com um contexto na esquerda de uma coordenação, uma preposição ou uma vírgula, com tanto que seja examinada a flexão do grupo anterior que tem que "unificar" (não contradizer) a flexão do grupo analisado:

```
{c=np,r=28,fu=np}[{c=noun},{c=adj;attr)]
 {lc={c=coord;p;punct},check=d_n}
```

A gramática e o programa de interpretação servem-se em abundância desta nova técnica da unificação sem deixar de lado a eficiência das gramáticas procedurais. A desvantagem destas gramáticas procedurais costuma ser a manutenção que precisa de muito cuidado e familiarização, neste caso ainda dificultado pelo fato das regras não serem recursivas nem conter elementos facultativos ou reiterativos.

O preço que se paga pela passagem aos últimos formalismos gramaticais (que oferecem todos estes meios) é alto: em geral, o programa (parser) que aplica a gramática ao texto tem que seguir todos os caminhos e sempre armazenar os resultados intermediários como veremos mais adiante. Houve tentativas de superar ou pelo menos aliviar estas dificuldades por meio da introdução de regras estatísticas que determinam o caminho mais provável, mas como é sabido na lingüística, a estatística nem sempre dá certo, sendo a linguagem natural o uso infinito de meios finitos.

O programa do IAI, chamado *mpro*, possui uma alta velocidade e é um dos mais robustos que existem no mercado; consegue também trabalhar com a maioria dos formatos, do simples DOS ao HTML, o formato das páginas internet (MAAS 98).

AS GRAMÁTICAS CAT2

O formalismo CAT2 está dentro dos padrões do projeto EUROTRA, o grande projeto de pesquisa em tradução automática realizado por todos os estados membros da Comunidade Européia de 1985 a 1992. Pode ser considerado uma linha paralela (por isto o 2!) que continuou seguindo as diretrizes dos dois primeiros anos quando a chefia do projeto decidiu abandonar o primeiro formalismo (relativamente puro de unificação e declarativo) e passar a outra linha bem mais procedural, o chamado E-framework (HALLER 90,93; SHARP 94).

Assim que as letras CAT lembram os primeiros conceitos de EUROTRA, o C significando `constructor` (o nome que o EUROTRA deu às regras de expansão), o A `átomo` (as entradas de base) e T `translator`, as regras de manipulação de árvores etiquetadas. O CAT2 também guardou o conceito de representações em níveis diferentes, por exemplo, o morfológico, o sintático e ou semântico, fato muito criticado na época pelos formalismos concorrentes como LFG (lexical functional grammar, BRESNAN 82) ou então HPSG (head driven phrase structure grammar) ou TAG (Tree adjoining grammar). Uma descrição geral destes formalismos é fornecida na página http://cslu.cse.ogi.edu/HLTsurvey/ch3node10.html#SECTION33. Como o EUROTRA, tampouco o CAT2 chegou a ser aplicado em escala comercial embora fossem construídos gramáticas e dicionários relativamente grandes. Hoje, este formalismo e o programa de interpretação escrito em PROLOG, está ainda sendo usado no projeto UNL (http://www.iai.uni-sb.de/UNL-iai.html) e em várias universidades para fins didáticos. Somente nos últimos anos, a interface do usuário chegou a um ponto de permitir uma velocidade suficiente para o desenvolvimento, e os interpretadores PROLOG a uma eficiência semelhante embora a própria linguagem de programação não chegasse tampouco a uma exploração comercial significante.

Além do mais, o CAT2 somente mostra as grandes vantagens dele num sistema multilingual em que as gramáticas e dicionários sempre servem tanto para análise e geração – e por isto, devem ser elaborados com o máximo cuidado e testados exaustivamente. Foram feitas algumas gramáticas de grande porte, junto com dicionários de várias dezenas de mil entradas lexicais, estes últimos nas línguas alemã, inglesa e francesa. Gramáticas experimentais existem para uma dezena de outras línguas, incluindo línguas ´exóticas´como o árabe, o koreano, o russo e o japonês, na maioria dos casos com componentes de tradução para o alemão ou o inglês. Para isto, tornou-se necessário desenvolver uma sistemática de atributos e valores de grande abrangência a qual está exaustivamente descrita em (STREITER 96).

Explicaremos o CAT2 com alguns exemplos da língua portuguesa.

Morfologia no CAT2

Toda gramática do CAT2 começa com um comando que indica o nível de processamento e a língua tratada:

```
@level(mspt/morph/portuguese).
```

`Mspt` significa ´estrutura morfológica do português´, `morph` se refere em geral a partes da gramática que trabalham com operações de caracteres (e não de árvores).

Pode servir como exemplo simples a morfologia do adjetivo. No dicionário existem então entradas similares às observadas na gramática anterior, um pouco mais voltadas para ser lidas pelo usuário, estando elas logo na forma lematizada:

```
feio          = {role=gov,cat=a,lex=feio,flex=o}.[].
interessante  = {role=gov,cat=a,lex=interessante,flex=e}.[].
```

As regras C para construir as formas flexionadas podem então ser escritas da seguinte forma, descrevendo a operação de cortar e colar necessária para se obter a raiz e a nova forma flexionada:

```
adjsingf = {cat=a,lemma=STEM+a,agr={num=sing,gen=fem},max=yes}
     .[{cat=a,flex=o,lemma=STEM+o, agr={num=sing,gen=masc},max=no}].
adjplurm = {cat=a, lemma=STEM+os, agr={num=plu,gen=masc},max=yes}
     .[{cat=a,lemma=STEM+o,agr={num=sing,gen=masc},flex=o,max=no}].
adjplurf = {cat=a, lemma=STEM+as, agr={num=plu,gen=fem},max=yes}
     .[{cat=a,lemma=STEM+o,agr={num=sing,gen=masc},flex=o,max=no}].
adjplurs = {cat=a, lemma=STEM+s, agr={num=plu},max=yes,flex=e}
     .[{cat=a,lemma=STEM,flex=e,max=no}].
```

O traço `max` torna-se necessário para a geração, indicando que esta regra se aplica somente uma vez.

Observe-se a forma geral das regras: começando com o nome da regra (que serve no momento do teste para parar o programa nesta regra), continuando com o signo "igual" os traços do lado esquerdo da regra entre colchetes, a flecha das regras de expansão substituída pelo ponto, e o lado direito da regra definido com [].

Finalmente, uma regra termina com um ponto, o qual separa ela da próxima regra. Esta forma é válida para todo tipo de regras – as entradas terminais simplesmente têm a mão direita vazia.

Sintaxe

Como na morfologia, temos o comando inicial da gramática parcial que trabalhará com árvores ou estruturas de atributos e valores (`syntactic`):

```
@level(cspt/syntactic/portuguese).
```

Como exemplo de uma regra sintática, mostramos aqui em primeiro lugar a regra que analisa e gera grupos nominais. A forma geral corresponde ao que foi descrito no capítulo anterior. Usa-se em abundância a unificação para passar todos os traços necessários para a cabeça da regra, `np`. Ao mesmo tempo, garante-se que somente se construirá um grupo nominal se artigo, adjetivo e substantivo não contiverem informações contraditórias relativas ao "agreement" (número e gênero). Um grupo como *"a casa bonito" simplesmente não seria analisado.

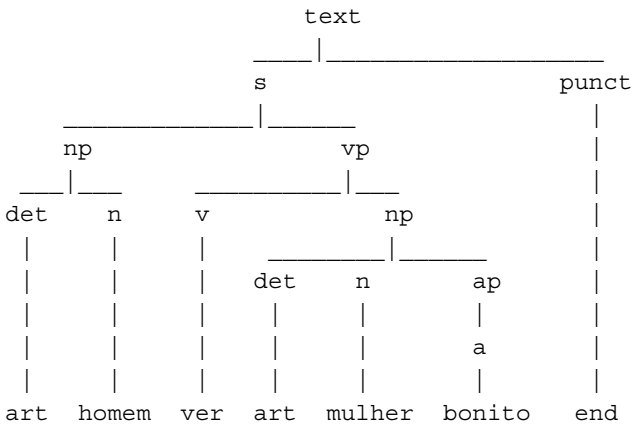
O adjetivo na geração será colocado no devido lugar (antes ou depois do substantivo) se ele está com classificação correta no dicionário.

```
np   = {cat=np,agr=A,semf=S,type=T,d=D,dtype=R}.[
            ^{cat=det,agr=A,dtype=R},
            ^{cat=ap,agr=A,pos=pre},
             {cat=n,agr=A,semf=S,type=T,d=D} ,
            ^{cat=ap,agr=A,pos=post},
            ^{cat=pp},
            ^{cat=s,type=rel,semf=S,agr=A}].
```

Elementos opcionais são marcados com ^, e as regras podem ser recursivas como se vê na primeira regra do grupo preposicional, válida para preposições simples ("de") que não incluem o artigo definido (`d~=def`):

```
pp    =  {cat=pp,sem=S,semf=E,d=D,dtype=R,agr=A}.[
                  {cat=p,sem=S,d~=def},
                  {cat=np,semf=E,d=D,agr=A,dtype=R}].
```

Como análise de uma frase simples como "O homem vê a mulher bonita." obtemos então o seguinte resultado em forma de árvore:



Existe também a possibilidade de representar o resultado de uma forma semelhante da gramática anterior, quer dizer, com todos os traços e colunas diferentes, representando o agrupamento dentro da frase. A representação em árvore, porém, é de muito valor para a didática já que muitos alunos de lingüística aprenderam a desenhar estas árvores nas aulas. É fácil de achar erros nas análises, e somente especialistas preferem a forma de traços e parênteses – a qual se torna claramente necessária quando se quer achar a causa de um eventual erro de análise. Para isto, também é valioso a possibilidade de observar o procedimento do programa passo a passo.

Primeiro, o programa constrói o primeiro grupo nominal, copiando tantos elementos quantos são necessários para caber dentro da regra:

```
cspt:b:<art>   COPYING SOURCE ATOM >
cspt:b:<homem>   COPYING SOURCE ATOM >
cspt:b:np   FOUND > g
       0
   ___|___
  1        2
  |        |
art  homem
```

Depois, copia-se o próximo elemento da frase:

```
cspt:b:<ver>   COPYING SOURCE ATOM >
```

e vê se isto já pode integrar uma nova regra sintática, no caso aquela da frase “O homem vê” que é correta seguindo as regras puramente sintáticas:

```
cspt:b:s   FOUND >
cspt:f:np_vp_agr   ok >
cspt:f:declarative   ok >
```

Podem-se observar aqui as regras tipo “f” (feature – traço) que servem para completar a informação de vários nós de uma manera geral: se aplicam a todas as frases que não sejam explicitamente marcadas como imperativas ou exclamativas contanto que o primeiro elemento seja um grupo nominal. Por isto, chama-se também “default rules”:

```
declarative  = {cat=s}>>{mode=decl}.[{cat=np}, * ].
np_vp_agr = {cat=s,tense=T}.[ {cat=np,agr=A}, {}>>{agr=A,tense=T}, * ].
```

O mesmo procedimento se repete com o grupo “a mulher bonita”, intercalando-se a formação do grupo adjetival (ap), que sería mais complicado se estivesse acompanhado de um adverbio etc. (“mais bonita”):

```
cspt:b:<art>   COPYING SOURCE ATOM >
cspt:b:<mulher>   COPYING SOURCE ATOM >
cspt:b:<bonito>   COPYING SOURCE ATOM >
cspt:f:adjpos   ok >
cspt:b:ap   FOUND >
cspt:b:np   FOUND >
```

Quando existe a sigla >>, como também na segunda regra em questão, isto significa que um grupo np somente será válido se número e pessoa (no caso de pronome) nao contêm informaçao contraditória; uma frase como “O homem vêem” sería excluída da análise. No caso da frase ser “O homem tem”, o verbo recebería a informação compartilhada com o grupo nominal que o número é o singular.

```
cspt:b:s   FOUND > g
       0
   ___|_____
  1         4
  |         |
__|___      |
2     3     5
|     |     |
art  homem  ver
```

Finalmente, se copia o último elemento da frase e se forma o texto, consistindo da frase sintaticamente correta mais a pontuação:

```
cspt:b:<end>   COPYING SOURCE ATOM >
cspt:b:text   FOUND > g
```

É também possível escrever gramáticas compreendendo várias frases ou parágrafos, por exemplo, para analisar as referências de pronomes ou outros elementos dêiticos.

AS REGRAS DE MANIPULAÇÃO DE ÁRVORES E A REPRESENTAÇÃO SEMÂNTICA

As regras de manipulação de árvores são precedidas por um comando indicando os dois níveis entre os quais se farão as transposições de árvores, o nível `ispt` sendo um nível semântico ou de interface entre duas línguas, por exemplo, o português e o inglês – lembramos que este formalismo foi desenvolvido para a tradução automática.

```
@level(cspt<=>ispt).
```

Aqui entra em jogo um novo tipo de regras chamado `t`, de transformação, uma operação muitas vezes necessária para a tradução. Os defensores dos últimos formalismos alegam não necessitar de regras deste tipo porque resolvem tudo somente com análise e geração sofisticada, mas terminam com regras muito complicadas porque é dificil de abstrair e de levar somente em conta o que fica invariável entre duas línguas diferentes.

```
@rule(t).
```

O nosso exemplo será a regra que transforma uma representação sintática de uma frase na sua contrapartida semântica, a qual se assemelha a uma árvore de dependência.

```
s    = {cat=s}.[ np, {cat=vp}.[v:{vf=fin},
*rest ]] <=>
               {}.[ v:{cat=v},
               np:{type~=rel},
               *rest:{focus~=front,
               type~=rel} ].
```

Formalmente, são duas regras do tipo `b`, separadas por uma flecha simples ou dupla. A flecha dupla significa que esta regra é válida tanto para análise quanto para geração, o que é o caso na maioria das regras. A flecha única somente costuma usar-se quanto uma língua possui várias variantes da mesma forma gramatical, por exemplo, em português as duas formas do pronome relativo (“que” e suas equivalências combinadas com “qual”). Neste caso, pode-se formular uma gramática que é capaz de analisar as duas formas, mais no momento da geração somente gera somente as formas explícitas, precedidas do artigo:

```
np_rel = {cat=np,type=rel}>>{d=def}.[*]
<= {}.[*].
```

Nas partes da direita e da esquerda ocorrem, ainda, marcadores (`v:, np:,rest`), que marcam elementos a serem transportados ao outro nível; elementos não marcados são excluídos. Isto serve na regra atual para gerar uma árvore mais simples, no caso de um grupo nominal serve por exemplo, para não transportar os artigos que quase nunca serão traduzidos à outra língua como lexemas (veremos mais adiante).

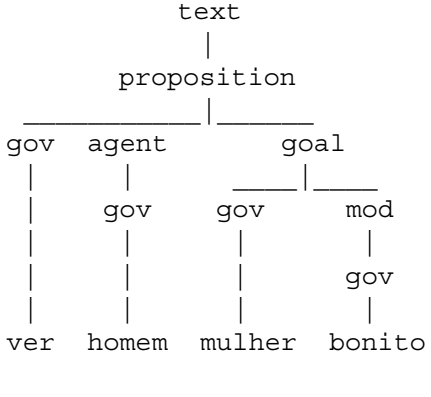
O nível semântico tem as suas próprias regras, usando o resto da informação do dicionário, como, por exemplo, a valência dos verbos:

```
pred = {cat=C}.[{role=gov,cat=C,frame=
          {arg1=A1,arg2=A2}},^A1, ^A2,
          *{role=mod} ].
```

Todo elemento tem que concordar com os outros que estão agrupados com ele; isto se faz à través das variáveis `A` que têm que unificar uma com a outra para o predicado semântico ser válido. A entrada lexical correspondente do verbo “ver” e a seguinte:

```
ver = {role=gov,cat=v,lex=ver,flex=irreg,
      frame={arg1={role=agent,cat=n},arg2=
      {role=goal,cat=n}}
```

Inserida esta regra na representação semântica, obtemos a seguinte árvore:



O processo pode ser invertido, é dizer que com esta representação é possível gerar a mesma frase portuguesa novamente, e ao trocar, por exemplo, todos os traços ´número´ pelo plural, poderíamos gerar a mesma frase com todos os membros no plural: “Os homens vêem as mulheres bonitas.”

COMANDOS DE SISTEMA DO CAT2

Para fazer estas manipulações, o CAT2 dispõe de uma serie de comandos e ajudas que tornam o uso muito simples, que são indicados na tela quando se da o comando `help`:

```
Object Commands      |Database Commands    | System Commands
---------------+----------------+-----------------------------------------
compare  delete      |open   close         |comment/% count edit  freeze  help/?
keep     show        |get    remove        |history  input lex    load    quit
translate            |select upload        |repeat   set   shell/! source  status
                     |unlock               |trace    unload untrace version  xi
                     |                     |xl
```

Objetos são sempre representações lingüísticas que podem ser administradas (delete), mostradas na tela (show) ou guardadas num arquivo (keep) ou então manipuladas para análise lingüística (translate). Quando se obtêm várias representações de uma frase, se podem comparar (compare) as representações de um mesmo nível e ver em qual atributo ou nó elas diferem.

Os comandos que se referem à base de dados servem quando se usam grandes dicionários; o acesso é mais rápido, e a inserção de novas entradas é facilitada.

Já os comandos do próprio sistema servem para carregar novas gramáticas (load), indicar a língua fonte e a língua objeto da tradução (set) num sistema multilingual ou repetir uma série de comandos executados anteriormente (repeat).

O comando ´input´ serve para teclar uma frase na tela ou pegar um arquivo inteiro para análise.

Todas as operações estão também descritas no manual CAT2 (SHARP 94).

Para rodar o sistema CAT2, é necessário instalar um PROLOG (versão SICSTUS ou SWI), e compilar as bases de PROLOG correspondentes.

O sistema CAT2 (com algumas gramáticas e dicionários de exemplo) pode ser obtido livremente pelo IAI (www.iai.uni-sb.de) por universidades e outros órgãos de pesquisa, para fins não-comerciais.

CONCLUSÃO

Mostramos exemplos de sistemas baseados na unificaçao para análise morfológica e sintática, tomando o português como língua de exemplificação. Discutimos as vantagens e obstáculos de vários procedimentos, e chegamos à hipótese que o formalismo CAT2 é muito apto para fins didáticos. Outras ferramentas para análise do português acham-se na página internet http://www.portugues.mct.pt/recursos.html#ferr.

BIBLIOGRAFIA

Gramáticas formais em geral:

http://cslu.cse.ogi.edu/HLTsurveych3node5.html#SECTION33

BRESNAN, Joan (editor): The Mental Representation of Grammatical Relations. MIT Press, Cambridge, Massachusetts, 1982.

HALLER, Johann: Die semantische Interface-Struktur von EUROTRA im Transfer zwischen Deutsch und Portugiesisch. In: Lüdtke, H. und Schmidt-Radefeldt, J. (Hrsg.): LINGUISTICA CONTRASTIVA: DEUTSCH VERSUS PORTUGIESISCH UND SPANISCH. Akten des 2. kontrastiven Kolloquiums an der Christian-Albrechts-Universität Kiel vom 15.-17. November 1990. Reihe: Acta Romanica 11., G. Narr Verlag, Tübingen.

_____. EUROTRA - O projeto de pesquisa e desenvolvimento em tradução automática da Commissão Europeia. Exemplos de Tradução Português-Alemão. In: Coletânea da revista Letras de Hoje, PUC Porto Alegre, 1991

_____. CAT2 - Vom Forschungssystem zum präindustriellen Prototyp. In: Pütz, Horst P. und Haller, Johann (Hrsg.): Sprachtechnologie: Methoden, Werkzeuge, Perspektiven. Sprache und Computer, Band 13, S. 282 - 303. Georg Olms Verlag, Hildesheim, 1993.

MAAS, Dieter. (1998) Multilinguale Textverarbeitung mit MPRO In: G. Lobin et al. (eds): Europäische

Kommunikationskybernetik heute und morgen, KoPäd, München.

SHARP, Randall. (1994) CAT2 Reference Manual Version 3.6 Saarbrücken, IAI.

STREITER, Oliver. (1996) Linguistic Modeling for Multilingual Machine Translation Frankfurt, Shaker Verlag.

_____. Oliver et alii. (1992) Getting Started with the CAT2 MT-System Saarbrücken, IAI.