



F U N D A Ç ã O
GETULIO VARGAS

EMAp

Escola de
Matemática Aplicada

Gramáticas computacionais no formalismo HPSG utilizando a *Grammar Matrix*

Leonel Figueiredo de Alencar (UFC e
Visitante EMAp/FGV)

Alexandre Rademaker (IBM Research Lab
e EMAp/FGV)



Estrutura de constituintes

- flat S
- NP VP
- NP VP com modificação adjetival



1º encontro 12 de abril de 2021

Gramática Livre de Contexto

- CFG1

$S \rightarrow D N V D N$

- CFG2

$S \rightarrow NP VP$

$NP \rightarrow D A^* N$

$VP \rightarrow V (NP) (VP)$

$CS \rightarrow C S$

$C \rightarrow \text{"that"}$

Teoria X-barra

$XP \rightarrow YP X'$

$X' \rightarrow X ZP$

YP especificador

X núcleo

ZP complemento

X' X-barra



Paralelismo entre categorias

- Sujeitos e determinantes
 - **We** created a monster.
 - **our** creation of a monster
- (Sag; Wasow; Bender, 2003, p. 64)

Gramática Universal

$XP \rightarrow YP, X'$

$X' \rightarrow X, ZP$

- Parâmetro da Ordem
 - Inglês, Português etc.: +
 - Japonês: -

(Mioto; Silva; Lopes, 2005, p. 35)

Exemplos

- the cat sleeps under the bridge
- Neko wa hashi no shita de nemuru
- cat bridge under sleeps
- The dog chases the cat.
- Inu wa neko o oikakemasu.
- cat dog chases



Construção de uma minigramática

- Gramática do inglês gerada pela **Grammar Matrix** disponível em:
<https://github.com/LR-POR/tutorial>

Algumas referências

- KLENK, Ursula. *Generative Syntax*. Tübingen: Narr, 2003.

Este livro apresenta a evolução da gramática gerativa a partir do trabalho de Chomsky nos anos de 1950, desembocando nos formalismos não transformacionais LFG, GPSG e HPSG.


Algumas referências

- BENDER, E. M. Grammar Engineering for Linguistic Hypothesis Testing. In: GAYLORD, N. et al. (Org.). *The Proceedings of the Texas Linguistics Society 10: Computational Linguistics for Less-Studied Languages*. Stanford: CSLI, 2008. p. 16-36.

Este artigo mostra como a implementação computacional pode servir para testar hipóteses linguísticas.



2º encontro 19 de abril de 2021



Grammar Matrix → Gramática
computacional (TDL + Lisp) ← HPSG
(teoria gramatical e formalismo para
descrição das estruturas gramaticais
de uma língua)

Gramática computacional

- Gramática de “papel e lápis”

“FRASE é um enunciado de sentido completo, a unidade mínima de comunicação.” (Cunha; Cintra, 1985, p. 116)
- Gramática computacional
 - CFG
$$S \rightarrow NP VP$$
 - HPSG: signos



Por que gramática computacional?

- Do ponto de vista da linguística: possibilidade de verificar automaticamente a consistência interna e a plausibilidade empírica de um modelo da linguagem humana ou de uma análise específica de uma língua particular



Por que gramática computacional?

- Do ponto de vista da ciência da computação:
 - motivação teórica: a gramática, como parte da linguagem (natural), constitui uma faculdade da mente humana e, portanto, integra o domínio da inteligência artificial
 - aplicações: extração de informações, resolução de perguntas, tradução automática do tipo FAHQT etc.



Q&A

Watson da IBM

FERRUCI, D. et al. Building Watson: An Overview of the DeepQA Project. *AI Magazine*, v. 31, n. 3, 2010.

MCCORD, M. C.; MURDOCK, J. W.; BOGURAEV, B. K. Deep parsing in Watson. *IBM Journal of Research and Development*, Armonk, v. 56, n. 3/4, p. 1-15, 2012.



Extração de informações

NOVICHKOVA, S; EGOROV, S.; DARASELIA, N. MedScan: a natural language processing engine for MEDLINE abstracts. *Bioinformatics*, Oxford, v. 19, n. 13, p. 1699-1706, 2003.



Tradução automática

- Sistema KANT da Carnegie Mellon University (LFG)
- Sistema VERBMOBIL (HPSG)
- Sistema MOLTO (Grammatical Framework)

Engenharia da gramática

- **Abordagem baseada em conhecimento:** elaboração manual de uma gramática num formalismo computacionalmente implementado → *parser* → análise sintática automática (*parsing*)
- **Abordagem baseada em dados:** manual de anotação (gramática) → construção de um *treebank* → algoritmo de aprendizagem de máquina → modelo estatístico → *parsing*



Indicações de leitura

DUCHIER, D.; PARMENTIER, Y. High-level Methodologies for Grammar Engineering, Introduction to the Special Issue. *Journal of Language Modelling*, Warszawa, Poland, v. 3, n. 1, p. 5-19, 2015.

DOI:

<https://doi.org/10.15398/jlm.v3i1.117>

Indicações de leitura

- BENDER, E. M. Grammar Engineering for Linguistic Hypothesis Testing. In: GAYLORD, N. et al. (Org.). *The Proceedings of the Texas Linguistics Society 10: Computational Linguistics for Less-Studied Languages*. Stanford: CSLI, 2008. p. 16-36.



3º encontro

26 de abril de 2021


Recapitulação

- Versão 03 da minigramática do inglês (sen = *simple English*, arquivo [choices03.txt](#))
 - Fenômenos implementados:
 - Concordância sujeito-verbo na 3ª pessoa
 - Sentença simples com verbos intransitivos e verbos transitivos
 - Flexão nominal regular (plural)

Testfile output [out03.txt](#)

Redundâncias na codificação

- A atual hierarquia verbal mistura valência (propriedade dos radicais) com propriedades expressas pelas flexões verbais, como tempo, modo, pessoa, número etc.
- Para cada forma verbal, são necessárias 15 escolhas.
- Por exemplo, repete-se a especificação de -s em *chases* e *sleeps*
- Desse modo, a codificação do léxico verbal torna-se extremamente cansativa.

- 
- **Agrupamento de propriedades associadas em tipos**
 - Todo verbo no modo indicativo ou subjuntivo é finito.
 - A codificação da pessoa e número é independente da codificação do tempo e do modo (especialmente relevante para línguas com flexão verbal mais rica como latim ou português)

Classes posicionais

- Decomposição das formas flexionadas em classes posicionais

radical (stem)	vogal temática	tempo- modo	pessoa- número
cant	á	va	mos
part	í	a	mos

Implementação na Matriz Gramatical

- Cada classe posicional recebe um determinado *input* e comporta um determinado número de regras lexicais.
- A classe pode ser prefixal ou sufixal.
- Regras lexicais operam sobre o *input*, acrescentando algum material ortográfico (segmentos) e propriedades (*features* 'traços').
- Exemplo em [choices11d.txt](#)

Modelação prévia

- Desenhar fluxograma:
 - ClassePosicional 1 => ClassePosicional 2
=> ... => ClassePosicional n

Verbos auxiliares

- Dados
 - the dog **is** following the cat
 - the dog **has** followed the cat
 - the dog **will** follow the cat
- Cada auxiliar exige uma forma própria do verbo principal
- Implementação na Matriz Gramatical: **Word Order** e **Lexicon/Auxiliary** types no questionário

Alçamento (*raising*)

- Movimento do sujeito do verbo principal para sujeito do auxiliar

is (VP the dog following the cat) =>
the dog is (VP following the cat)

Cf. verbete *raising* na Wikipedia.

Auxiliar: com ou sem predicado?

- Os verbos *be*, *have* e *will*, como auxiliares do presente contínuo (progressivo), presente perfeito e do futuro, contribuem para a sentença apenas com especificações de tempo, modo, aspecto, pessoa e número (POULSON, 2011).
- Outros auxiliares, como os modais, possuem uma especificação semântica própria, devendo receber, na codificação, um predicado.

Implementação do auxiliar *be* progressivo

progr (aux1)

Auxiliary type 1:

Type name: progr

This auxiliary type contributes:

(X) No predicate.

() An independent predicate.

Verbos auxiliares modais

- Operador de modalidade \Box (ou M, do alemão *Möglichkeit* ‘possibilidade’):
 - the dog **can** bark
- Operador de modalidade \Diamond (ou N, do alemão *Notwendigkeit* ‘necessidade’):
 - the dog **must** sleep

Cf. lógica modal na [Encicl. Stanford de Fil.](#)

Implementação dos auxiliares modais

- Análoga à dos auxiliares *be*, *have* e *will*, exceto pela atribuição de um predicado aos modais

`possibility (aux4)`

Auxiliary type 4:

Type name: `possibility`

This auxiliary type contributes:

☐ No predicate.

☒ **An independent predicate.**

Hipergeração na versão choices11d

- *chase + ed = *chaseed* etc.

False positives

9 *the cat slept 1 21

14 *the dog chaseed the cat 1 34

[...]

Ver [results11d.txt](#)

Hipogeração na versão choices11c

- Formas verbais irregulares como *chased* e *slept* não são analisadas

False negatives

[...]

29 the dog has slept 0 12

31 the dog has chased the cat 0 20

Ver [results11d.txt](#)


Indicações de leitura

- ARONOFF, M. *Word formation in generative grammar*. Cambridge: MIT, 1976.
- DAVIS, A. R. *Linking by types in the hierarchical lexicon*. Stanford: CSLI, 2001.
- MONTEIRO, J. L. *Morfologia portuguesa*. 4. ed. Campinas: Pontes, 2017.



Indicações de leitura

- POULSON, L.
Meta-modeling of Tense and Aspect in a Cross-linguistic Grammar Engineering Platform.
UW Working Papers in Linguistics, v. 28, 2011.



4º encontro 3 de maio de 2021

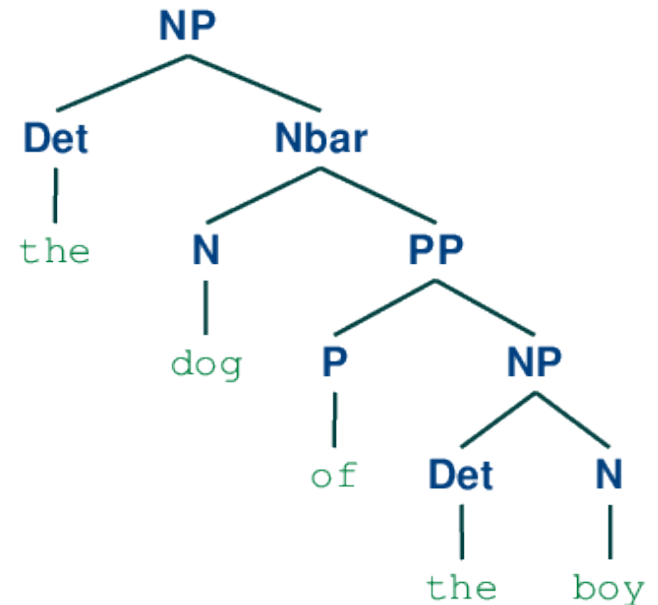
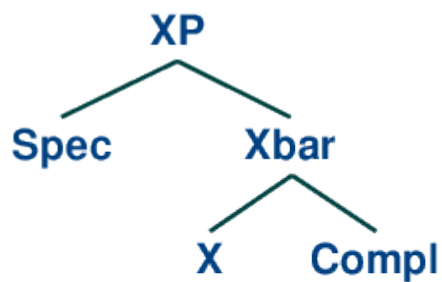
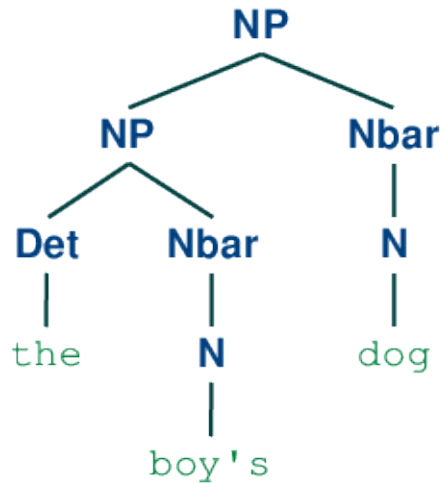
Roteiro

- Implementação da posse adnominal
- Correção da hipergeração (**chaseed*) e hipogeração (não reconhecimento de *chased*)
- Limitações da Matriz Gramatical e como superá-las
- Tarefas de casa:
 - ampliação da minigramática do inglês
 - implementação do latim

Posse adnominal

- Genitivo saxão:
(1) the **boy's** dog sleeps
- Genitivo normando:
(2) the dog **of** the boy sleeps
- Em (1), o possuidor (PSOR) funciona como especificador (Spec), mas em (2), como modificador:
(1b) *that the boy's dog sleeps

Esquema X-barra



Variação tipológica

- PSOR como especificador (Spec) ou modificador (Mod)?
 - Latim: *illa porta* [_{Mod} *scholae*] ‘aquela porta da escola’
- Marcação de PSOR ou PSUM (*possessum*)?
 - Nheengatu: *uka rukena*
casa porta ‘porta da casa’
- PSOR < PSUM ou PSUM < PSOR?

Implementação da posse adnominal em inglês

- Genitivo saxão e genitivo normando
 - Versão da minigramática choices14
 - Resultado do teste out14
 - Análise do resultado em result14.txt
- O que falta:
 - Pronomes possessivos
 - PSOR como pronome na construção genitiva normanda: *this dog of mine*

Morfologia computacional

- A formação de palavras (flexão, derivação etc.) pode ser modelada em dois componentes distintos (BEESLEY; KARTTUNEN, 2003):
 - Morfotática:
 - *bark + ed => barked*
 - *chase + ed => *chaseed*
 - Morfofonologia ou alterações ortográficas:
 - Ex. de regra em inglês: deletar e final antes de +ed

Correção da hipogeração e da hipergeração

- A Matriz Gramatical implementa atualmente apenas o nível morfológico.
- É possível implementar fenômenos morfofonológicos morfotaticamente por meio da criação de subtipos, porém, isso é deselegante e extremamente ineficiente.
- A melhor solução para lidar com a morfofonologia é alterar manualmente os arquivos **irules.tdl** e **irregs.tab**.



Estrutura de arquivos de gramática gerada pela Matriz Gramatical

abr 29 19:10 irules.tdl (morfologia)

abr 29 13:03 lexicon.tdl

abr 29 13:03 lrules.tdl (regras lexicais)

abr 29 13:03 roots.tdl

abr 29 13:03 rules.tdl

abr 29 13:03 simple english.tdl (tipos da língua)

jan 29 22:18 matrix.tdl (núcleo comum)

out 19 2020 mtr.tdl

jun 24 2020 head-types.tdl

jun 24 2020 labels.tdl

O arquivo *irules.tdl*

- Regras lexicais do questionário → regras de prefixação ou sufixação.

```
%suffix (* ed)
pst-lex-rule.

pres-part-suffix :=
%suffix (* ing)
pres-part-lex-rule.

pres-3sg-suffix :=
%suffix (* s)
pres-3sg-lex-rule.

past-part-suffix :=
%suffix (* ed)
past-part-lex-rule.
```

```
saxon-genitive-suffix :=
U:**- irules.tdl Top L9 Git-master (TDL)
```

Sintaxe do arquivo *irules.tdl*

Regra de formação do passado (*pst = past*)

```
pst-suffix :=  
%suffix (* ed)  
pst-lex-rule.
```

Esquema

```
NOME-AFIXO :=  
%TIPO_DE_AFIXO (* AFIXO)  
NOME_DA_REGRA_LEXICAL.
```

Definições dos demais tipos criados pelo usuário

- Arquivo *simple english.tdl* (nome da língua)

```
pst-lex-rule := ind-lex-rule & infl-lex-rule &
```

```
[ SYNSEM.LOCAL.CONT.HOOK.INDEX.E.TENSE  
past ] .
```

- Sintaxe

```
NOME DO TIPO := SUPERTIPO_1 &  
SUPERTIPO_2 ... SUPERTIPO_N
```

```
MATRIZ_DE_ATRIBUTOS_E_VALORES.
```

Algoritmo de sufixação

- Dado $\%_{\text{suffix}}$ (* ed) :
 - Se o radical termina em “*”, substitua “*” por “ed”.
 - Como “*” representa o ε , i.e., a cadeia vazia (*empty string*), a flexão “ed” é sufixada ao radical:
 - bark ε → barked
 - chase ε → chased

Nova regra de formação do passado, versão 1

Morfologia concatenativa no LKB: Copestake (2002, p. 127-130)

```
;;; -*- Mode: TDL; Coding: utf-8 -*-
```

```
;;; Inflecting Lexical Rule Instances
```


```
pst-suffix :=
```

```
%suffix (* ed) (e ed) (sleep slept)
```

```
pst-lex-rule.
```

Lógica da regra na versão 1

- Assimetria entre análise e geração (COPESTAKE, 2003, p. 128-129): tanto *chased* quanto **chaseed* são analisadas, porém, apenas a primeira é gerada.
- Os pares (x,y) de *strings* da lista de sufixos são processados da direita para a esquerda.
- Na análise, para cada par, se o radical termina em x , x é removido e y , acrescentado. Na geração, é retornada uma forma ao primeiro *match*.
- Simulação em Python: *tools/MakePast.py*



Nova regra de formação do passado, versão 2

```
;;; -*- Mode: TDL; Coding: utf-8 -*-
```

```
;;; Inflecting Lexical Rule Instances
```

```
[...]
```

```
%(letter-set (!e abcd fghijklmnopqrstuvwxyz))
```

```
pst-suffix :=
```

```
%suffix (!e !eed) (e ed) (sleep slept) (give gave)
```

```
pst-lex-rule.
```


Lógica da regra 2

- Em vez de usar o *catch all* “*” na última regra (da direita para a esquerda), define-se o conjunto de caracteres *!e* com todos caracteres do inglês que não sejam “e”:

%(letter-set (!e abcd fghijklmnopqrstuvwxyz))

O último caso a ser verificado passa a ser então *(!e !eed)*, ou seja, se *x* está em *!e*, então substitua *x* por *x+y*. Agora não temos mais *chaseed*, pois *chase* não preenche essa condição.

Bloqueio de formas regulares

Conforme Copestake (2002, p. 200), a solução para evitar a geração de formas regulares como **sleeped* é listá-las no arquivo *irregs.tab* e especificar como verdadeiro o seguinte parâmetro no arquivo *globals.lsp*:

*(defparameter *irregular-forms-only-p* t)*

A Matriz Gramatical já faz essa especificação por defeito.

Arquivo *irregs.tab*

Todo o conteúdo deve estar entre aspas, i.e., constituir uma string de LISP (Copestake, 2002, p. 199-200). A Matriz Gramatical gera um arquivo *irregs.tab* vazio (apenas com comentários explicativos).

"

```
; irreg_form IRREG_TYPE reg_form
```

```
slept PST-LEX-RULE slept
```

```
slept PAST-PART-LEX-RULE slept
```

"

Incompatibilidade entre versões do LKB?

A estratégia de utilizar o arquivo *irregs.tab* não funcionou, provocando comportamento estranho do LKB (quebra do parser ao analisar a forma *slept*, acusando tipo de regra não definido).

Algumas limitações da Matriz Gramatical

- Apenas verbos divalentes? Não há como implementar verbos trivalentes com dois objetos? Ex.:
 - Abrams showed Browne the office. ([HP - NL Test Suite](#))
- Alçamento em orações completivas não suportado: *I asked who is sleeping.*
- Controle do sujeito e do objeto não suportados (verbos *prometer* e *persuadir*, por ex.)

Controle funcional obrigatório

- Controle do sujeito: o sujeito da oração encaixada, nucleada pelo verbo no infinitivo, é igual ao sujeito do verbo matriz:
 - **A Maria** prometeu [] fazer a tarefa.
- Controle do objeto: o sujeito da oração encaixada, nucleada pelo verbo no infinitivo, é igual ao objeto do verbo matriz:
 - A Maria persuadiu **o Pedro** [] a fazer a tarefa.
Ver Francez e Wintner (2002, p. 197).

Exemplo de correção manual do arquivo *my language.tdl*

- choices54 da minha gramática do português:

False positives

17 *o cachorro quer que o gato late 1 89

27 *o cachorro afirmou que o gato lata 6 122

- Na Matriz Gramatical, o modo da oração completiva é determinado pelo complementador (no caso, *que*) e não pelo verbo.
- Alteração manual do TDL

Modo verbal em orações completivas

- Verbos declarativos (*verba dicendi*) e verbos de atividade mental exigem o modo indicativo (Mateus et al., 1989, p. 270-271):
 - o cachorro declarou que o gato late
 - *o cachorro afirmou que o gato lata
- Verbos volitivos exigem o modo subjuntivo (Mateus et al., 1989, p. 273):
 - *o cachorro quer que o gato late
 - o cachorro quer que o gato lata

Codificação da seleção de modo pelo verbo matriz

- TDL gerado pela Matriz Gramatical

volitive-verb-lex := clausal-nom-verb-lex & clausal-second-arg-trans-lex-item &

[SYNSEM [LOCAL.CAT.VAL.COMPS < [LOCAL [CAT [HEAD comp & [FORM finite], WH.BOOL -],

CONT.HOOK.INDEX.SF prop]] >, [...]

- TDL modificado

CONT.HOOK [INDEX.SF prop, **CLAUSE-KEY.E.MOOD subjunctive**]

Um problema não resolvido na gramática do inglês

- Não consegui implementar sentenças com mais de um auxiliar, embora isso esteja previsto no questionário (*sen/out33.txt*):

False negatives

31 the dog has been sleeping 0 288

32 the dogs have been sleeping 0 474

Exercícios

- Expandir a gramática do inglês:
 - Orações completivas
 - Perguntas QU (*wh-questions*)
 - Orações interrogativas encaixadas
 - Outras formas de expressão da posse adnominal (ver arquivo *sen/out33.txt*)
- Implementar gramática básica do latim: arquivo *lat/out19.txt*)

Importância do latim

Chegados ao Brasil, três eminentes matemáticos de renome internacional, Gleb Wataghin, professor de mecânica racional e mecânica celeste, Giacomo Albanese, professor de geometria, Luigi Fontapié, professor de análise matemática, que vieram contratados para lecionar na faculdade de Filosofia de S. Paulo — professor Wataghin é considerado, no mundo inteiro, um dos maiores pesquisadores de raios cósmicos — cuidaram, logo após os primeiros meses de aula, de enviar um ofício ao então ministro da Educação, que na época cogitava de reformar o ensino secundário. Vejamos o que, mais de esperança que de desânimo, continha esse ofício: "[...] Pedimos a vossa excelência que na reforma que se projeta se dê menos matemática e MAIS LATIM no curso secundário, para que possamos ensinar matemática no curso superior ". [...] **O professor Albanese costumava dizer — e muitas pessoas são disto prova — “ dêem-me um bom aluno de latim, que farei dele um grande matemático”.** (ALMEIDA, 1983, p. 7) (grifo meu)

Proposta de trabalho

- Criar um ou mais grupos de estudo para resolução das tarefas, por exemplo, um grupo coordenado por orientandos meus que estão trabalhando com HPSG ou esperam fazê-lo no futuro.
- Minhas soluções serão disponibilizadas no repositório <https://github.com/LR-POR/tutorial> em 10/06/2021.

Indicações de leitura

ALMEIDA, N. M. de. *Gramática latina*. 19. ed. São Paulo: Saraiva, 1983.

BEESELEY, K. R.; KARTTUNEN, L. *Finite state morphology*. Stanford: CSLI, 2003.

COPESTAKE, A. *Implementing typed feature structure grammars*. Stanford: CSLI, 2002.

FRANCEZ, N.; WINTNER, S. *Unification grammars*. Cambridge: CUP, 2012.

MATEUS, M. H. M. et al. (1989). *Gramática da língua portuguesa*. 2. ed. Lisboa: Caminho.