O TRATAMENTO AUTOMATIZADO DA LINGUAGEM NATURAL

Carla Alexandra EZARQUI*

RESUMO

Este trabalho apresenta as modificações realizadas no Grammar Play, um analisador sintático, capaz de compreender sentenças simples e afirmativas do português brasileiro, que possuam apenas uma oração, identificando sua estrutura de constituintes. As alterações contribuíram para minimizar sua limitação, quanto à análise de determinadas frases, permitindo-o classificar como gramaticais sentenças que apresentam uma locução verbal com até dois verbos. Além disso, foram adicionados ao léxico desse *parser* dois pronomes pessoais, verbos conjugados em três tempos diferentes e vários termos em inglês e português, utilizados na área da informática. Portanto, as mudanças fizeram com que a gramática e o léxico do Grammar Play abrangessem um número maior de expressões da língua portuguesa, inclusive aquelas que apresentam termos emprestados do inglês.

PALAVRAS-CHAVE: Linguística. Inteligência Artificial. Prolog. Processamento de Linguagem Natural. *Parser*.

ABSTRACT

This paper presents the modification accomplished to Grammar Play, a parser that analyses simple and affirmative sentences of Brazilian Portuguese, which have only a clause identifying its constituent structure. The alterations contribute to reduce its limitation in relation to the analysis of specific phrases, allowing it to classify as grammatical phrases which have a verbal locution with up to two verbs. Besides, two personal pronouns, verbs conjugated in three different tenses and many terms in English and Portuguese used in informatics area were added to the lexicon of this parser. Therefore, the changes made the grammar and the lexicon of the Grammar Play embrace a bigger number of expressions of Portuguese language, including those that show terms which were borrowed from English.

KEYWORDS: *Linguistic. Artificial Intelligence. Prolog. Natural Language Processing. Parser.*

Contribuições para o Grammar Play

A comunicação tem se tornado cada vez mais sofisticada, possibilitando ao homem relacionar-se com as máquinas utilizando interfaces muito próximas à linguagem natural. A humanidade caminha para uma era em que o computador pensará tão rápido e racionalmente quanto o homem, agindo através do mesmo meio para realizar a comunicação, ou seja, a fala.

* Faculdade de Tecnologia de Taquaritinga – FATEC-Tq - Av. Dr. Flávio H. Lemos, 585. CEP 15900-000 - Taquaritinga - SP - e-mail: carlaezarqui@yahoo.com.br

Possibilitar que um software dialogue com o usuário é um trabalho que tem motivado tanto linguistas quanto informatas a se dedicarem à formalização e aplicação computacional da língua. Essa necessidade de automatizar uma linguagem deu origem à área da Linguística Computacional, que sendo considerada uma subárea da Inteligência Artificial, trabalha com muitos dos seus conceitos. (OTHERO; MENUZZI, 2005)

O desenvolvimento das tecnologias na área da Linguística Computacional, bem como nas demais subáreas da Inteligência Artificial certamente continuará até que os computadores atinjam uma complexidade semelhante à do cérebro humano. No momento em que um robô se comportar de forma semelhante ao ser humano, vivendo com ele em sociedade, será atingido um dos objetivos mais almejados do homem, uma vez que ele sempre se ocupou em criar máquinas que facilitassem a sua vida, como as que realizam complexos cálculos matemáticos em apenas alguns segundos. Desta vez, o computador em forma de um robô não apenas servirá ao homem, facilitando a sua vida, mas, principalmente, ao compartilhar as mesmas vivências, será o seu maior aliado.

A metodologia adotada para o desenvolvimento deste trabalho foi baseada principalmente na análise de dois livros: Linguística Computacional - Teoria e Prática (OTHERO; MENUZZI, 2005) e Teoria X-Barra - Descrição do português e aplicação computacional (OTHERO, 2006), uma vez que o objetivo do trabalho é tratar computacionalmente uma língua, tendo como objeto de estudo o analisador sintático Grammar Play. Os livros citados fundamentam o desenvolvimento desse analisador e, por isso, foram a base para a descrição das regras gramaticais que o constituem, bem como para o processo de implementação dessas regras no Prolog. É apresentada a implementação realizada ao código do Grammar Play, que objetivou contribuir para melhorar sua compreensão da língua portuguesa. Uma vez que é função dele classificar como gramaticais ou agramaticais sentenças sugeridas pelo usuário, quanto mais extenso for o seu léxico e maior o número de regras, que compondo a gramática de uma língua, preveem as possíveis formações de frases, mais abrangente é sua compreensão. Por isso, foram adicionadas à gramática do Grammar Play regras que o permite analisar sentenças que apresentam uma locução verbal com até dois verbos e ao léxico adicionou-se verbos conjugados nos tempos verbais: pretérito imperfeito, pretérito perfeito e futuro do presente do modo indicativo, além de dois pronomes pessoais: eu e nós e dezesseis substantivos em inglês e português relacionados à área da informática.

O Grammar Play representa um grande passo para o desenvolvimento brasileiro direcionado à área da Linguística Computacional, contudo apresenta limitações no que diz respeito à aceitação de determinadas expressões do português brasileiro, considerando-as como sendo expressões gramaticais.

A gramática do Grammar Play é apresentada com todas as alterações realizadas. As regras adicionadas estão numeradas e destacadas em negrito.

```
% Inicialização do analisador
s(S):-s(EC,S, []),append('resultado.pl'),write('S'),writeln(EC),
told,write('Estrutura de constituintes: '),write(EC),nl;
append('resultado.pl'), write('Frase agramatical ou léxi-
co desconhecido.'), told,write('Frase agramatical ou léxico
```

```
desconhecido'),nl.
%%%%%%%%%%%%%%%%%%%%%%%%
% Regras Sintagmáticas %
%%%%%%%%%%%%%%%%%%%%%%%%
% Sentença (S)
s([sn, SN, sv, SV]) --> sn([_,Num], SN), sv(Num, SV).
s([sv, SV, sn, SN]) --> sv(Num, SV), sn([_,Num], SN).
s([sv, SVimp]) --> svimp(_, SVimp).
s([sadv, SAdv, s, S]) --> sadv(SAdv), s(S).
s([conj,[CONJ],s,S]) --> conj(CONJ),s(S).

% Sintagmas Nominais (SN)
sn(Conc, [det, [Det], n_bar, N_Barra]) --> det(Conc, Det), n_
barra(Conc, N_Barra).
sn(Conc, [n_bar, N_Barra]) --> n_barra(Conc, N_Barra).
sn(Conc, [pre_det, [Pre_Det], sn, SN]) --> pre_det(Conc, Pre_
Det), sn(Conc, SN).
n_barra(Conc, [pro, [Pro]]) --> pro(Conc, Pro).
n_barra(Conc, [n, [N]]) --> n(Conc, N).

n_barra(Conc, [sadj, SAdj, n_bar, N_Barra]) --> sadj(Conc, SAdj),
n_barra(Conc, N_Barra).
n_barra(Conc, [n_bar, X, sadj, SAdj]) --> x(Conc, X), sadj(Conc,
SAdj).
x(Conc, [n, [N]]) --> n(Conc, N).
x(Conc, [n_bar, Y, sadj, SAdj]) --> y(Conc, Y), sadj(Conc, SAdj).
y(Conc, [n, [N]]) --> n(Conc, N).
y(Conc, [n_bar, Z, sadj, SAdj]) --> z(Conc, Z), sadj(Conc, SAdj).
z(Conc, [n, [N]]) --> n(Conc, N).
n_barra(Conc, [nbar, X, sp, SP]) --> x(Conc, X), sp(P, SP).

% Sintagmas Adjetivais (SAdj)
sadj(Conc, [adj_bar, Adj_Barra, sadv, SAdv]) --> adj_barra(Conc,
Adj_Barra), sadv(SAdv).
sadj(Conc, [adj_bar, Adj_Barra, sp, SP]) --> adj_barra(Conc, Adj_
Barra), sp(P, SP).
sadj(Conc, [sadv, SAdv, adj_bar, Adj_Barra]) --> sadv(SAdv), adj_
barra(Conc, Adj_Barra).
sadj(Conc, [adj_bar, Adj_Barra]) --> adj_barra(Conc, Adj_Barra).
adj_barra(Conc, [adj, [Adj]]) --> adj(Conc, Adj).
adj_barra(Conc, [sadv, SAdv, adj_bar, Adj_Barra]) --> sadv(SAdv),
adj_barra(Conc, Adj_Barra).
adj_barra(Conc, [adj_bar, A, sadv, SAdv]) --> a(Conc, A),
sadv(SAdv).
```

```
adj_barra(Conc, [adj_bar, A, sp, SP]) --> a(Conc, A), sp(P,
SP).
a(Conc, [adj, [Adj]]) --> adj(Conc, Adj).
% Sintagmas Preposicionais (SP)
sp(Lex, [p, [P], sn, SN]) --> p(Lex, inf, P), sn(Conc, SN).
sp(Lex, [p, [P], sn, SN]) --> p(Lex, Conc, P), sn(Conc, SN).
1) sp(Lex, [p, [P], svi, Svi]) --> p(Lex, inf, P), svi(Svi).
2) sp(Lex, [p, [P], svi, Svi]) --> p(Lex, Conc, P), svi(Svi).

% Sintagmas Verbais (SV)
sv(Num, [v_bar, V_Barra, sadv, SAdv ]) --> v_barra(Num, i, V_Bar-
ra), sadv(SAdv).
sv(Num, [v_bar, V_Barra, sp, SP]) --> v_barra(Num, i, V_Barra),
sp(_, SP).
3)sv(Num, [v_bar, V_Barra, svi, SVI]) --> v_barra(Num, _, V_Bar-
ra), svi(SVI).
4)sv(Num, [v_bar, V_Barra, svg, SVG]) --> v_barra(Num, _, V_Bar-
ra), svg(SVG).
5)sv(Num, [v_bar, V_Barra, svtc, SVTC]) --> v_barra(Num, _, V_
Barra), svtc(SVTC).
6)sv(Num, [v_bar, V_Barra, svvp, SVVP]) --> v_barra(Num, _, V_
Barra), svvp(SVVP).
sv(Num, [v_bar, V_Barra]) --> v_barra(Num, _, V_Barra).
svimp(Num, [v_bar, V_Barra_i]) --> v_barra_i(Num, _, V_Barra_i).
sv(Num, [sadv, SAdv, v_bar, V_Barra]) --> sadv(SAdv), v_barra(Num,
_, V_Barra).
svimp(Num, [sadv, SAdv, v_bar, V_Barra_i]) --> sadv(SAdv), v_
barra_i(Num, _, V_Barra_i).

sv(Num, [v_bar, V_Barra, sadv, SAdv ]) --> v_barra(Num, _, V_Bar-
ra), sadv(SAdv).
svimp(Num, [v_bar, V_Barra_i, sadv, SAdv]) --> v_barra_i(Num, _,
V_Barra_i), sadv(SAdv).
sv(Num, [v_bar, V_Barra, sn, SN]) --> v_barra(Num, _, V_Barra),
sn(_, SN).
sv(Num, [v_bar, V_Barra, sp, SP]) --> v_barra(Num, _, V_Barra),
sp(_, SP).
svimp(Num, [v_bar, V_Barra_i, sp, SP]) --> v_barra_i(Num, i, V_
Barra_i), sp(_, SP).

v_barra(Num, _, [sadv, SAdv, v_bar, V_Barra]) --> sadv(SAdv), v_
barra(Num, _, V_Barra).
v_barra_i(Num, _, [sadv, SAdv, v_bar, V_Barra_i]) --> sadv(SAdv),
v_barra_i(Num, _, V_Barra_i).
```

```
    v_barra(Num, i, [v, [V]]) --> v(Num, i, V).
    v_barra(Num, td, [v, [V], sn, SN]) --> v(Num, td, V), sn(_, SN).
    v_barra(Num, ti, [v, [V], sp, SP]) --> v(Num, ti(P), V), sp(P,
SP).
    7) v_barra(Num, td, [v, [V], svi, SVI]) --> v(Num, td, V),
svi(SVI).
    8) v_barra(Num, ti, [v, [V], svi, SVI]) --> v(Num, ti(P), V),
svi(SVI).
    9) v_barra(Num, ad, [v, [V], svg, SVG]) --> v(Num, ad, V),
svg(SVG).
    10) v_barra(Num, ap, [v, [V], svg, SVG]) --> v(Num, ap, V),
svg(SVG).
    11) v_barra(Num, tc, [v, [V], svtc, SVTC]) --> v(Num, tc, V),
svtc(SVTC).
    12) v_barra(Num, vp, [v, [V], svvp, SVVP]) --> v(Num, vp, V),
svvp(SVVP).
    v_barra(Num, vl, [vl, [V], sn, SN]) --> v(Num, vl, V), sn(Conc,
SN).
    v_barra(Num, vl, [vl, [V], sadj, SAdj]) --> v(Num, vl, V),
sadj(Conc, SAdj).
    v_barra(Num, vl, [v, [V], sp, SP]) --> v(Num, vl, V), sp(_, SP).
    v_barra(Num, vl, [v, [V], sadv, SAdv]) --> v(Num, vl, V),
sadv(SAdv).
    v_barra_i(_, i, [v, [Vimp]]) --> vimp(_, i, Vimp).
    v_barra_i(_, td, [v, [Vimp], sn, SN]) --> vimp(_, td, Vimp),
sn(_, SN).
    v_barra_i(_, _, [v_bar, T, sadv, SAdv]) --> t(_, _, T), sadv(SAdv).
    v_barra(Num, _, [v_bar, T, sadv, SAdv]) --> t(Num, _, T),
sadv(SAdv).
    v_barra(Num, _, [v_bar, T, sp, SP]) --> t(Num, _, T), sp(_, SP).
    v_barra(Num, _, [v_bar, T, sn, SN]) --> t(Num, _, T), sn(_, SN).
    t(Num, _, [v, [V]]) --> v(Num, _, V).
    t(Num, _, [v, [V], sp, SP]) --> v(Num, _, V), sp(_, SP).
    t(Num, _, [v, [V], sn, SN]) --> v(Num, _, V), sn(_, SN).
    t(Num, _, [v_bar, TT, sadv, SAdv]) --> tt(Num, _, TT), sadv(SAdv).
    t(_, _, [v, [Vimp], sn, SN]) --> vimp(_, td, Vimp), sn(_, SN).
    tt(Num, _, [v, [V]]) --> v(Num, _, V).
    % Locução Verbal
    %Infinitivo
    13) svi([vi_bar, VI_Barra]) --> vi_barra(VI_Barra).
    14) vi_barra([vi, [VI]]) --> vi(VI).
    15) vi_barra([vi, VI, sn, SN]) --> vi(VI), sn(_, SN).
    16) vi_barra([vi, Vi, sadj, SAdj]) --> vi(VI), sadj(_, SAdj).
    17) vi_barra([vi, VI, sadv, SAdv]) --> vi(VI), sadv(SAdv).
```

```
    18) vi_barra([vi, VI, sp, SP]) --> vi(VI), sp(_, SP).

    % Gerúndio
    19) svg([vg_bar, VG_Barra]) --> vg_barra(VG_Barra).
    20) vg_barra([vg, [VG]]) --> vg(VG).
    21) vg_barra([vg, VG, sn, SN]) --> vg(VG), sn(_, SN).
    22) vg_barra([vg, VG, sadv, SAdv]) --> vg(VG), sadv(SAdv).
    23) vg_barra([vg, VG, sp, SP]) --> vg(VG), sp(_, SP).

    % Tempo Composto
    % Particípio
    24) svtc([vtc_bar, VTC_Barra]) --> vtc_barra(VTC_Barra).
    25) vtc_barra([vtc, VTC, sn, SN]) --> vtc(VTC), sn(_, SN).
    26) vtc_barra([vtc, VTC, sadj, SAdj]) --> vtc(VTC), sadj(_, SAdj).
    27) vtc_barra([vtc, VTC, sadv, SAdv]) --> vtc(VTC), sadv(SAdv).
    28) vtc_barra([vtc, VTC, sp, SP]) --> vtc(VTC), sp(_, SP).

    % Voz Passiva Analítica
    %Particípio
    29) svvp([vvp_bar, VVP_Barra]) --> vvp_barra(_, VVP_Barra).
    30) vvp_barra(_,[vvp, [VVP]]) --> vvp(_, VVP).
    31)  vvp_barra([vvp,  VVP,  sadv,  SAdv]) -->  vvp(_,  VVP),
sadv(_,SAdv).
    32) vvp_barra([vvp, VVP, sp, SP]) --> vvp(_, VVP), sp(_, SP).

    % Sintagmas Adverbiais (SAdv)
    sadv([adv_bar, Adv_Barra]) --> adv_barra(Adv_Barra).
    sadv([b, B, adv_bar, Adv_Barra]) --> b(B), adv_barra(Adv_Barra).
    sadv([adv_bar, Adv_Barra, sp, SP]) --> adv_barra(Adv_Barra),
sp(P, SP).
    adv_barra([adv, [Adv]]) --> adv(Adv).
    adv_barra([adv_bar, C, sp, SP]) --> c(C), sp(P, SP).
    b([adv_bar, Adv_Barra]) --> adv_barra(Adv_Barra).
    c([adv, [Adv]]) --> adv(Adv).

    %%%%%%%%%%%%%%%%%%%
    % Regras Lexicais %
    %%%%%%%%%%%%%%%%%%%
    pre_det(Conc, Pre_Det) --> [Pre_Det], {pre_det(Conc, Pre_Det)}.
    det(Conc, Det) --> [Det], {det(Conc, Det)}.
    n(Conc, N) --> [N], {n(Conc, N)}.
    pro(Conc,Pro) --> [Pro], {pro(Conc, Pro)}.
    v(Num, Val, V) --> [V], {v(Num, Val, V)}.
    vimp(Num,Val, Vimp) --> [Vimp], {vimp(Num,Val, Vimp)}.
```

```
33) vi(VI) --> [VI], {vi(VI)}.
34) vg(VG) --> [VG], {vg(VG)}.
35) vtc(VTC) --> [VTC], {vtc(VTC)}.
36) vvp(Conc, VVP) --> [VVP], {vvp(Conc, VVP)}.
p(Lex, Conc, P) --> [P], {p(Lex, Conc, P)}.
adj(Conc, Adj) --> [Adj], {adj(Conc, Adj)}.
adv(Adv) --> [Adv], {adv(Adv)}.
conj(CONJ) --> [CONJ], {conj(CONJ)}.
```

Originalmente, o léxico apresentava somente verbos no tempo presente, conjugados nas terceiras pessoas do singular e plural. Uma vez adicionados os tempos pretérito perfeito, pretérito imperfeito e futuro do presente, além das pessoas Eu e Nós, foi necessário controlar não apenas o número, mas a pessoa em que o verbo é conjugado na sentença. Portanto, foram substituídos o sing (singular) e o plur (plural) do léxico, responsáveis pela concordância nominal e verbal, por ppsi (primeira pessoa singular pretérito imperfeito), pppi (primeira pessoa plural pretérito imperfeito), ppsp (primeira pessoa singular pretérito perfeito), pppp (primeira pessoa plural pretérito perfeito), ppst (primeira pessoa singular presente), pppt (primeira pessoa plural presente), ppsf (primeira pessoa singular futuro), pppf (primeira pessoa plural futuro), tpsi (terceira pessoa singular pretérito imperfeito), tppi (terceira pessoa plural pretérito imperfeito), tpsp (terceira pessoa singular pretérito perfeito), tppp (terceira pessoa plural pretérito perfeito), tpst (terceira pessoa singular presente), tppt (terceira pessoa plural presente), tpsf (terceira pessoa singular futuro) e tppf (terceira pessoa plural futuro).

As regras adicionadas à gramática permitem sentenças com até dois verbos, pois o processo verbal pode ser indicado por mais de uma palavra: o primeiro no pretérito perfeito, pretérito imperfeito, presente ou futuro do presente do modo indicativo, e o segundo nas formas nominais infinitivo impessoal, gerúndio ou particípio.

Exemplos: “Eu sei **formatar** (infinitivo impessoal) computadores”; “Os alunos estavam **estudando** (gerúndio) na biblioteca”; “Eles têm **conversado** (particípio) frequentemente”.

Seguindo os conceitos da língua portuguesa, este fenômeno é chamado de locução verbal, denominada como sendo o conjunto formado por verbos que exprimem um processo.

Portanto, nas locuções verbais, um verbo, denominado auxiliar, junta-se a um outro, denominado principal, estendendo-lhe a significação. O verbo principal estará sempre conjugado no infinitivo ou na forma nominal gerúndio e o auxiliar receberá as flexões de tempo, modo, pessoa e número. Se houver mais de um auxiliar, a flexão ocorrerá somente no primeiro, o que não se enquadra no *parser* estudado. (TERRA, 2007)

Todos os verbos pertencentes ao léxico do Grammar Play poderão formar uma locução verbal, com qualquer verbo no infinitivo impessoal do mesmo léxico, podendo apresentar como complemento um SN (Sintagma Nominal), um SAdj (Sintagma Adjetival), um SAdv (Sintagma Adverbial) e/ou um SP (Sintagma Preposicional) ou, ainda, quando o verbo for intransitivo, não apresentar complemento.

O Sintagma Verbal (SV) foi aprimorado passando a ser denominado como: SVI (Sintagma Verbal com Infinitivo Impessoal), SVG (Sintagma Verbal com Gerúndio), SVTC (Sintagma Verbal - Tempo Composto) e SVVP (Sintagma Verbal - Voz Passiva). Nas regras referentes a esses sintagmas verbais as categorias sintagmáticas como SAdj, SAdv, SN e SP que modificam o núcleo são representadas como complemento, ocupando a respectiva posição na árvore sintática. Portanto, todas as regras se encerram no V ao invés de V'. Isso estabelece que determinado sintagma seja o complemento do verbo, aceitando um adjunto que agirá como modificador desse mesmo verbo.

Embora, semelhante, a formação dos quatro sintagmas citados acima: SVI, SVG, SVTC e SVVP não é idêntica, isto é, nem todos se combinam com os sintagmas citados anteriormente: SN, SAdj, SAdv, SP e SV, sendo estes sistematizados pela gramática de estrutura sintagmática.

Somente as regras condizentes à constituição do SVI são apresentadas. As regras implementadas foram as seguintes:

3) SV → V' SVI - 13) SVI → VI' - 14) VI' → VI - 15) VI' → VI SN - 16) VI' → VI SAdj 17) VI' → VI SAdv - 18) VI' → VI SP

A regra 3 expressa que um sintagma verbal pode ser constituído por um V e um SVI , isto é, um verbo junto de outro, em que o segundo deverá, obrigatoriamente, estar no infinitivo impessoal, este por sua vez pode ser acompanhado pelas categorias sintagmáticas expressas nas regras 15 a 18.

As regras 13 e 14 se complementam e permitem que uma sentença se encerre com um VI. Nesse caso, os verbos mais ocorrentes são os verbos intransitivos. Exemplo: "Ela adora **digitar** (verbo intransitivo)."

Nas sentenças em que há uma locução verbal, pode haver também um SN, um SAdj, um SAdv e um SP, e as regras 15, 16, 17 e 18 tratam essas possibilidades.

A regra 16 vai ocorrer quando o verbo no infinitivo for um verbo de ligação.

Assim, verificam-se os respectivos exemplos: "Nós precisaremos **desenvolver** (infinitivo impessoal) **um sistema** (sintagma nominal)."; "Eu quero **ser** (infinitivo impessoal) **inovador** (sintagma adjetival)."; "Eles sabiam **programar** (infinitivo impessoal) **otimizadamente** (sintagma adverbial)." e "Eu prefiro **estudar** (infinitivo impessoal) **em casa** (sintagma preposicional)."

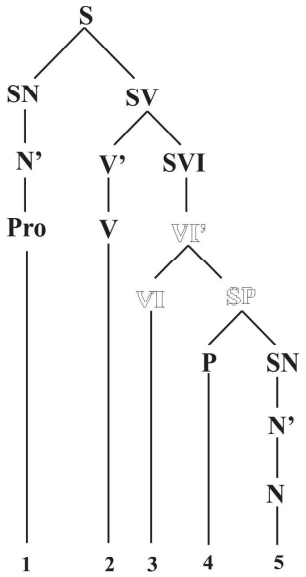

Ilustração 1 – Árvore Sintática da sentença: Eu prefiro estudar em casa
Fonte: Autoria Própria (2009)
Onde: **1:** Eu; **2:** prefiro; **3:** estudar; **4:** em; **5:** casa.

Segundo Othero (2006, p. 100) "a implementação das regras sintagmáticas em Prolog precisa ser mais detalhada do que prevista no modelo da Teoria X-Barra." Logo, as regras 7 e 8 especificam que um verbo no infinitivo pode ser sucedido por qualquer verbo transitivo direto ou indireto, onde a regra 8 deverá trabalhar em conjunto com as regras 1 e 2, que por sua vez tratam da inserção de uma preposição na sentença. Ambas são as mesmas de Othero (2006), apenas foram adaptadas para o SVI. Exemplo: "Você **gosta** (verbo transitivo indireto) **de** (preposição) **aprender** (infinitivo impessoal) coisas novas."

Por fim, a regra 33 que fazendo parte das regras lexicais, controla a forma com que um VI deve ser inserido em uma sentença.

Como os verbos no infinitivo não necessitam de nenhum controle, seja de número, seja de gênero, sua regra lexical não apresenta nenhuma variável, além do próprio VI.

No léxico também foi necessário uma adaptação, uma vez que uma nova categoria de verbos foi adicionada.

```
% Verbos no Infinitivo (vi)[1]
vi(amar).
vi(aprender).
vi(conversar).
vi(estudar)
vi(perdoar).
```

1 Esta é apenas uma amostra dos verbos no infinitivo impessoal adicionados ao léxico do Grammar Play. No total, foram adicionados 1.545 verbos.

Outras alterações foram realizadas no Grammar Play, são elas: a inserção do advérbio "não" ao léxico, o que possibilitou a construção de frases negativas e a implementação da regra sintagmática `s([conj,[CONJ],s,S]) --> conj(CONJ),s(S)`, bem como a regra de inserção lexical `conj(CONJ) --> [CONJ], {conj(CONJ)}` de Othero (2006) que não se apresentavam na gramática, além das conjunções no léxico do Grammar Play.

Em relação ao léxico, este necessitou de uma adaptação para que os verbos pudessem ser conjugados em diferentes pessoas e tempos, o que não requereu nenhuma mudança à gramática.

O léxico modificado do Grammar Play não é apresentado na íntegra, pois resultou em um documento com 6677 páginas, enquanto o original continha 964. Logo, apenas algumas categorias gramaticais são apresentadas com suas respectivas amostras.

```
% Léxico %
% Nomes (n)
% Nomes - Feminino Singular e Plural
n([fem,tpsp], alma).
n([fem,tppp], almas).
 n([fem,tpst], alma).
 n([fem,tppt], almas).

 % Adjetivos - Masculino Singular e Plural
 adj([masc,ppsi], belo).
 adj([masc,pppi], belos).
 adj([masc,ppsf], belo).
 adj([masc,pppf], belos).
 adj([masc,tpsi], belo).
 adj([masc,tppi], belos).
 adj([masc,tpsf], belo).
 adj([masc,tppf], belos).
```

A possibilidade dos substantivos constituírem o sujeito requer que eles concordem com os verbos, por isso devem ser cadastrados no pretérito imperfeito, pretérito perfeito, presente e futuro do presente e apenas em uma pessoa, pois correspondem à terceira pessoa do singular ou plural. Exemplo: "Os **alunos** (substantivo) desenvolveram um novo projeto" é equivalente a: "Eles desenvolveram um novo projeto".

O mesmo ocorre com os adjetivos e, apesar de ser uma alternativa que comprometeu o desempenho das consultas em Prolog, uma vez que cada um deles, feminino e masculino, são cadastrados oito vezes, foi uma forma de tornar possível o trabalho com vários tempos. No léxico original, que permitia apenas as terceiras pessoas do singular e plural, eles foram inseridos da seguinte maneira:

```
n([fem,sing], alma).
n([fem,plur], almas).
```

```
adj([masc,sing], belo).
adj([masc,plur], belos).
```

Os adjetivos, além de constituírem o sujeito juntamente de um substantivo, formam o predicativo do sujeito, devendo ser conjugados nas primeiras pessoas do singular e plural, por isso foram cadastrados dezesseis vezes. Exemplo: “Ela é bonita.”; “Nós somos bonitas.”

Os verbos foram cadastrados dezesseis vezes cada um, conjugados em quatro tempos verbais e quatro pessoas. Um exemplo é mostrado com o verbo escrever na primeira pessoa do singular no tempo futuro do presente sendo um verbo transitivo direto, visto que ele é intransitivo, transitivo direto, transitivo indireto e bitransitivo.

```
% Verbos (v)
% EU – FUTURO
v(ppsf, td, escreverei).
```

Os termos em inglês e português adicionados, relacionados à área da informática foram os seguintes, apresentados apenas para realizarem a concordância com os verbos no futuro do presente e na terceira pessoa do singular: `n([masc,tpsf], notebook)e n([masc,tpsf], servidor).`[2]

Considerando que o software em foco possui sua interface implementada em Delphi, com o intuito de testar as alterações realizadas no *parser*, foi necessário carregar a gramática e o léxico diretamente no SWI-Prolog, o que foi possível, devido ao fato de, tanto a gramática quanto o léxico estarem disponíveis no próprio Grammar Play.

Observa-se a seguir a consulta de uma sentença.

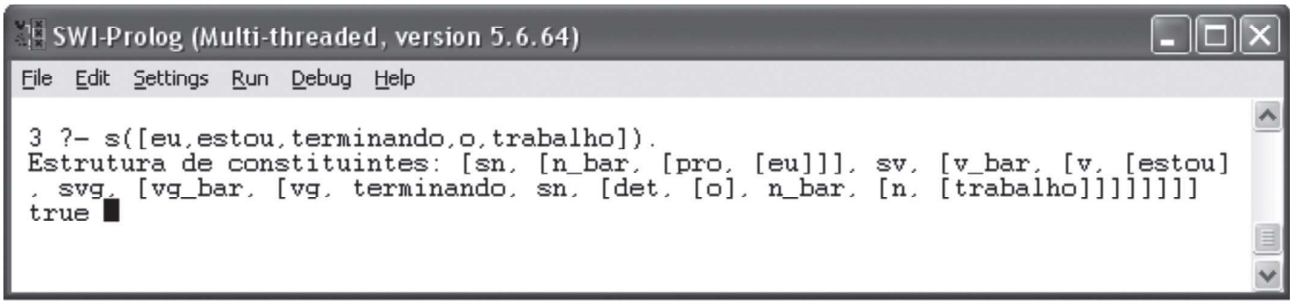

Ilustração 2 - Consulta em Prolog
Fonte: Autoria Própria (2009)

CONCLUSÃO

Ao analisar as regras sintagmáticas e a implementação das mesmas em Prolog, gerando um *parser* que é capaz de analisar frases com apenas uma oração, constata-se que a compreensão desse anali-

2 Esta é apenas uma amostra em que os termos estão adaptados para concordar com os verbos no singular e no tempo futuro do presente. Ao todo foram adicionados dezesseis substantivos ao léxico do Grammar Play.

sador é bastante restrita. Essa restrição se dá, não só por uma limitação apresentada pela linguagem Prolog, ao que se refere a tratar computacionalmente uma gramática ou até mesmo pela complexidade que a mesma pode apresentar. Mas, especificamente, porque não há documentação na íntegra, da gramática da língua portuguesa. Além disso, a linguagem é fruto de um fenômeno social e se modifica à medida que o tempo passa. Outro fator que contribui para o problema em questão é o fato de a comunicação verbal envolver a subjetividade proveniente do emissor ou receptor da mensagem. Incorporar a um software a capacidade de discernir só é possível se lhe for incorporado previamente, uma inteligência artificial.

As mudanças realizadas no *parser* foram conduzidas pela heurística, uma vez que as tentativas de implementar regras sintagmáticas, que aumentassem o nível de compreensão desse programa basearam-se na intuição, por ser um falante nativo da língua e nas circunstâncias, por já haver várias regras prontas. Portanto, foi necessário esquematizar o fenômeno linguístico que se objetivou implementar e adequar essa esquematização à ideia formalizada no *parser* Grammar Play. As soluções foram buscadas por aproximações sucessivas, avaliando-se os progressos alcançados, até que a regra em foco permitisse ao *parser* compreender determinada frase, embora, assim como as regras de Othero (2006), frases que não possuam sentido semântico, mas que apresentam uma estrutura gramatical correta, continuam sendo aceitas.

Ao término das pesquisas e dos estudos realizados através deste trabalho, obteve-se um parser "mais inteligente", visto que sua capacidade de compreender sentenças foi expandida.

REFERÊNCIAS

OTHERO, G. A.; MENUZZI, S. M. *Linguística Computacional* – Teoria e Prática. São Paulo: Parábola, 2005.
OTHERO, G. A. *Teoria X-barra: descrição do português e aplicação computacional*. São Paulo: Contexto, 2006.
TERRA, E. Curso Prático de Gramática. 5.ed. São Paulo: Scipione, 2007.