SEÇÃO DA REVISTA
ID – NÚMERO DE IDENTIFICAÇÃO DO ARTIGO ORIGINAL

Parser Sintático para o Português Brasileiro: desafios e soluções

Syntactic Parser for Brazilian Portuguese: challenges and solutions
Analizador sintáctico para portugués brasileño: desafíos y soluciones

[1]
Faculdade de Tecnologia do Estado de São Paulo (FATEC), São Paulo, SP, Brasil.

Manoel Francisco Guaranha[2]
Faculdade de Tecnologia do Estado de São Paulo (FATEC), São Paulo, SP, Brasil.
Universidade Santo Amaro (UNISA), São Paulo, SP, Brasil.

Recebido em: _____.
Aprovado em: _____.

Resumo
Este artigo tem como objetivo apresentar o *Parser* Sintático para o Português Brasileiro *Parsero*, desenvolvido a parir das ideias da Gramática Gerativa (CHOMSKY, 2015) aperfeiçoada pela Teoria X-Barra (CHOMSKY, 2014). Para tanto, foram utilizadas as regras desenvolvidas por Othero (2009), especialmente para o Português Brasileiro, que seguem o padrão X-Barra, e adaptadas pelo nosso projeto de pesquisa para atender às necessidades de processamento surgidas ao longo do desenvolvimento do *Parser*. A pesquisa utilizou como *corpus* o recurso DELAF_PB - *Dicionário de Palavras Simples Flexionadas para o Português Brasileiro* para povoar um Banco de Dados *Structured Query Language* (SQL). O DELAF_PB foi disponibilizado pelo Projeto Unitex-PB, desenvolvido pelo NILC – Núcleo Interinstitucional de Linguística Computacional e pelo ICMC - Instituto de Ciências Matemáticas e de Computação (USP; NILC; ICMC, 2004) e foi construído com base no formalismo francês DELA - *Dictionnarie Electronique du LADL* (MUNIZ, 2004). Como resultado, disponibilizamos para futuros pesquisadores a Base de Dados SQL com 1.193.295 unidades léxicas classificadas, o endereço com o código aberto do *Parsero* e um link para execução do aplicativo.
Palavras-chave: Linguística Computacional. Processamento de Linguagem Natural. Gramática Gerativa. *Parser* Sintático para o Português Brasileiro.

Abstract
This article aims to present the Syntactic Parser for Brazilian Portuguese Parsero, developed from the ideas of Generative Grammar (CHOMSKY, 2015) improved by the X-Barra Theory (CHOMSKY, 2014). Therefore, the rules developed by Othero (2009) were used, especially for Brazilian Portuguese, which follow the X-Barra pattern, and adapted by our research project to meet the processing needs that emerged during the development of Parser. The research used as corpus the resource DELAF_PB - Dictionary of Simple Flexed Words for Brazilian Portuguese to populate a Structured Query Language (SQL) Database. DELAF_PB was made available by the Unitex-PB Project, developed by NILC - Interinstitutional Nucleus of Computational Linguistics and by ICMC - Institute of Mathematical and Computer Science (USP; NILC; ICMC, 2004) and was built based on the French formalism DELA - Dictionnarie Electronique du LADL (MUNIZ, 2004). As a result, we have made available to researchers interested in the topic the SQL Database with 1,193,295 classified lexical units, the address with the open source of Parsero and a link to run the application.
Keywords: Computational Linguistics. Natural Language Processing. Generative Grammar. Syntactic Parser for Brazilian Portuguese.

[1] Orcid: https://orcid.org/0000-0003-0395-0303
E-mail: willian.pacheco@fatec.sp.gov.br.
[2] Orcid: https://orcid.org/0000-0002-8676-601X
E-mail: manoel.guaranha@gmail.com.

Resumen
Este artículo tiene como objetivo presentar el Analizador sintáctico Parsero para portugués brasileño, desarrollado a partir de las ideas de Gramática Generativa (CHOMSKY, 2015) mejoradas por la Teoría X-Barra (CHOMSKY, 2014). Para eso, utilizamos las reglas desarrolladas por Othero (2009), especialmente para el portugués brasileño, que siguen el patrón X-Barra, y adaptadas por nuestro proyecto de investigación para satisfacer las necesidades de procesamiento que surgieron durante el desarrollo de Parser. La investigación utilizó como corpus el recurso DELAF_PB - Diccionario de Palabras Simples Inflexionadas para Portugués Brasileño para llenar una Base de Datos en Lenguaje Estructurado de Consulta (SQL). DELAF_PB fue puesto a disposición por el Proyecto Unitex-PB, desarrollado por NILC - Núcleo Interinstitucional de Lingüística Computacional y por ICMC - Instituto de Matemáticas e Informática (USP; NILC; ICMC, 2004) y fue construido en base al formalismo francés DELA - Dictionnarie Electronique du LADL (MUNIZ, 2004). Como resultado, ponemos a disposición de los investigadores interesados en el tema la Base de Datos SQL con 1,193,295 unidades léxicas clasificadas, la dirección con código fuente abierto de Parsero y un enlace para ejecutar la aplicación.
Palabras-clave: Ligüística computacional. Procesamiento del lenguaje natural. Gramática generativa. Analizador sintáctico para portugués brasileño.

Introdução

Um dos problemas mais instigantes na área de processamento é o de construção de *Parses* sintáticos, programas que processam textos. Os desafios que se colocam na construção desses leitores eletrônicos compreendem as seguintes etapas: a construção e manutenção de um léxico organizado de modo a facilitar o processamento, já que o conjunto de palavras de uma língua é extenso e, pode-se dizer, aberto a novos termos que surgem a partir dos diferentes usos; a construção de algoritmos morfossintáticos, capazes de rotular os léxicos segundo sua função, uma vez que o valor posicional de cada unidade ou locução varia de acordo com a posição que ocupa nos enunciados; a construção de algoritmos capazes de agrupar os léxicos em sintagmas atribuindo-lhes as funções sintáticas e validando-os segundo as regras de combinação estabelecidas pela gramática. Em cursos tecnológicos de Análise e Desenvolvimento de Sistemas, a investigação de processos de construção de *Parses* sintáticos constitui uma tarefa interdisciplinar que oferece oportunidade de aprendizagem aos pesquisadores tanto os que se dedicam ao estudo das estruturas da língua, português falado no Brasil, no nosso caso, quanto àqueles que investigam estruturas de dados e de linguagens de programação que sejam adequadas para o Processamento de Linguagem Natural (PLN) que, diferente do processamento de números, sujeitos às regras rígidas da Matemática, requerem processos mais complexos de tratamento. Além disso, este projeto disponibilizou, para pesquisadores interessados no tema, recursos pouco frequentes na Internet: o aplicativo de PNL, *Parsero*, e o Banco de Dados com o léxico do Português Brasileiro.

Este artigo pretende, portanto, apresentar e discutir os resultados do processo de elaboração de *Parser Sintático* para o Português Brasileiro *Parsero*, pesquisa realizada no Curso Superior de *Tecnologia em Análise e Desenvolvimento de Sistemas* da FATEC – *Faculdade de Tecnologia do Estado de São Paulo*, no âmbito da CEPE - *Câmara De Ensino, Pesquisa e Extensão* – Fatec Ipiranga. Para tanto, foram utilizados os conceitos da Gramática Gerativa de Chomsky (2014; 2015), com adaptações propostas por Othero (2009) para atender às especificidades do Português Brasileiro e, além disso, com adaptações que foram necessárias para atender às especificidades deste projeto. A pesquisa utilizou como *corpus* para a construção de um banco de dados do léxico o recurso DELAF_PB - *Dicionário de Palavras Simples Flexionadas para o Português Brasileiro*, arquivo com aproximadamente 880.000 palavras. Esse recurso, cuja documentação detalhada pode ser encontrada em Muniz (2004), foi disponibilizado pelo *Projeto Unitex-PB*, desenvolvido pelo NILC – *Núcleo Interinstitucional de Linguística Computacional* e pelo ICMC - *Instituto de Ciências Matemáticas e de Computação* (USP; NILC; ICMC, 2004) e foi construído com base no formalismo francês DELA - *Dictionnarie Electronique du LADL* (MUNIZ, 2004).

A pesquisa desenvolveu, documentou e disponibilizou em repositórios virtuais três recursos linguísticos-computacionais para o Português Brasileiro: o Banco de Dados SQL (*Structured Query Language)* ParseroDB (PACHECO; GUARANHA, 2021); o código aberto do processador sintático *Parsero* (PACHECO; GUARANHA, 2021b); e uma interface visual para execução do *software* (PARSERO, 2021). Espera-se que esses recursos possam ser utilizados e, eventualmente, aperfeiçoados por pesquisadores que se interessem em avançar nos trabalhos de desenvolvimento de sistemas na área de Processamento de Linguagem Natural (PLN) no Brasil.

Para dar conta de nosso objetivo, este artigo será desenvolvido em três seções. Na primeira delas, discutiremos aspectos gerais da Gramática Gerativa proposta por Chomsky (2015) e aperfeiçoada pela Teoria X-Barra, conforme afirma o próprio Chomsky (2014, p. 392). Na segunda seção, apresentaremos o Banco de Dados SQL que serviu de *corpus* para o nosso Processador de Linguagem Natural, detalhando o processo de construção desse recurso. Na terceira seção, apresentaremos as características de nosso *Parser*, o *Parsero*, e as regras da Gramática gerativa adaptadas para o Português Brasileiro por Othero (2009), que foram utilizadas no algoritmo, justificando, ainda, algumas adaptações que fizemos para atender às nossas necessidades. Nesta última seção também apresentaremos exemplos de sentenças processadas pelo aplicativo.

Gramática Gerativa: potencialidades para o Processamento de Linguagem Natural (PNL)

Para este estudo, consideramos a língua como "um conjunto (finito ou infinito) de sentenças, cada sentença sendo finita em extensão e construída a partir de um conjunto finito de elementos" (CHOMSKY, 2015, p. 18). Essa concepção possibilita-nos estabelecer regras que possam separar sentenças gramaticais, que são reconhecidas como tal em uma língua, que parecem "aceitáveis a um falante nativo" (CHOMSKY, 2015, p. 18), das sentenças agramaticais, as que não são aceitáveis. Assim, podemos entender a gramática de uma língua como um "mecanismo que gera todas as sequências gramaticais" (CHOMSKY, 2015, p. 18) dessa língua. Dado que "[q]ualquer gramática de uma língua irá *projetar* o *corpus* de enunciados observados, finito e mais ou menos acidental, em um conjunto (presumivelmente infinito) de enunciados gramaticais" (CHOMSKY, 2015, p. 19, grifos do autor), pode-se dizer que "uma gramática reflete o comportamento do falante, que, baseado em uma experiência finita e acidental com a língua, pode produzir ou compreender um número indefinido de novas sentenças." (CHOMSKY, 2015, p. 19).

Quando falamos em sequências gramaticais de uma língua, neste ponto, falamos da perspectiva morfossintática, e não semântica. Podemos parafrasear Chomsky construindo sentenças como (1) "infinitos pássaros dormem em espelhos de açúcar" e (2) "pássaros os acima voam cabeças das dos homens". No caso de (1), ainda que a sequência não seja "dotada de sentido" ou "significativa" em um contexto não literário, pode ser considerada gramatical do ponto de vista da organização dos seus constituintes. No caso de (2), ainda que possamos, com alguma dificuldade, reorganizar os constituintes e compreender o significado do conjunto, temos uma estrutura não aceitável como está, agramatical portanto.

Chegamos neste ponto a um conceito geral de Gramática, em que cada sentença pode ser decomposta em pequenas unidades de sentido denominadas sintagmas. Essas unidades têm como núcleo as categorias gramaticais da língua e se relacionam entre si em uma estrutura de sentido que pode ser expressa em formato que se assemelha a uma árvore (Figura 1). Essas estruturas formais permitem que possamos estabelecer regras gerais para uma língua.

Para uma descrição mais detalhada dessa teoria, remetemos o leitor a Chomsky (2015) e, para um estudo dessas estruturas no português, para Souza e Silva; Koch (1993) e Othero (2009). O exemplo de decomposição e de sentença que apresentaremos a seguir foi adaptado de Chomsky (2015, p. 38-39). Dada a sentença (3) "O menino comeu o doce",

percebemos que as palavras não são agrupadas de modo aleatório, mas seguem regras de agrupamento que podem ser representadas do seguinte modo:

(1) 1. Sentença → SN (Sintagma nominal) + SV (Sintagma verbal)

2. SN → Artigo + Nome

3. SV → Verbo + SN

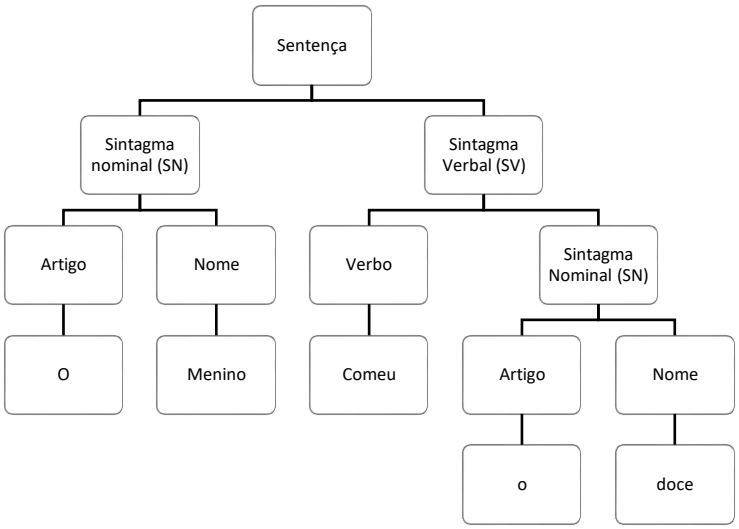
4. Artigo → o, a, os, as...

5. Nome → menino, menina, meninos, meninas, bolo, bolos, doce, doces...

6. Verbo → comeu, comeram, pegou, pegaram, comprou, compraram...

Cada regra deve ser interpretada na forma X → Y como a instrução "reescreva X como Y", de modo que teríamos como gerar, por substituição, sentenças como (5) "O menino comeu o bolo"; (6) "A menina comprou o doce"; (7) "Os meninos pegaram os bolos" e assim por diante. Poderíamos, igualmente, gerar sentenças como (8) "O bolo comeu o doce"; (9) "Os doces comeram as meninas", entre outras que seriam gramaticalmente aceitáveis, embora não o fossem semanticamente, pelo menos em contextos não literários. Podemos, ainda, determinar regras que regulam a organização dos termos no interior dos sintagmas em uma língua como o Português: artigos femininos no plural só poderiam estar ligados a substantivos femininos no plural, por exemplo. Também podemos determinar regras que regulam o relacionamento entre os sintagmas: caso o sintagma nominal fosse composto por um artigo e um substantivo no plural, o verbo teria de estar conjugado na terceira pessoa do plural, e assim por diante. O diagrama da Figura 1 seria um modo de representar uma gramática que analisa os constituintes a partir de um conjunto de regras que especificamos. Uma sequência de palavras é um constituinte se pudermos ligá-la "a um único ponto de origem" ao qual atribuímos um rótulo. Percebe-se pelo diagrama arbóreo da Figura 1 que o Sintagma Nominal aparece no nível abaixo da Sentença, ao lado do Sintagma Verbal, e que também aparece como componente do Sintagma Verbal. No primeiro caso, ele desempenha a função de sujeito: "O menino"; no segundo, de objeto direto: "o bolo".

Figura 1 – Diagrama arbóreo de (3)



Fonte: Autoria própria.

Com a finalidade de atender ao Português Brasileiro, as regras originais propostas por Chomsky (2015) precisam ser adaptadas, ainda que não alteradas em sua essência. Um excelente trabalho nesse aspecto foi desenvolvido por Othero (2009). Trata-se de uma pesquisa detalhada e consistente que serviu de base para a construção de nosso algoritmo. Na seção seguinte, vamos apresentar o Base de Dados que é utilizado por nosso PLN e, na última seção as características técnicas e de processamento segundo as regras estudadas em Othero (2009), com algumas adaptações e justificativas.

Modelagem e construção do Banco de Dados do Parser Parsero

Para o Processamento de Linguagem Natural, o primeiro problema é ter um léxico categorizado e atualizado, trabalho que requer pesquisa criteriosa e custosa. Encontramos esse material disponível no banco de dados DELAF-PB (Muniz, 2004; 2015). O DELAF-PB é um arquivo em formato *dic* que contém 9072146 milhões de linhas com palavras simples e flexionadas, conforme o padrão: palavra, canônica. Classe+traços: flexão.Cada linha é uma entrada de texto terminada com um caractere de quebra de linha (*\n*) e composta por até cinco campos delimitados por pontuação de acordo com o formato do *Dictionnarie Electronique du LADL (DELA)* (MUNIZ, 2004, p. 17).

Neste formato, uma vírgula (“,”) foi utilizada para separar as duas formas de cada palavra, a flexionada e a canônica, não flexionada. Para substantivos e adjetivos, a forma canônica está atribuída ao gênero masculino, quando não se trata de palavra de dois gêneros e, para verbos, está no infinitivo. Um ponto (“.”) à frente de uma palavra

representa uma categoria correspondente às classes gramaticais: substantivos, adjetivos, artigos, preposições, conjunções, numerais, pronomes, verbos, advérbios e interjeições. Também há entradas para prefixos, siglas e abreviaturas. Cada verbete pode estar categorizado em mais de uma classe gramatical, neste caso há uma entrada para cada classe ou, no caso de verbos, a entrada pode, ainda, repetir-se para cada uma das flexões.

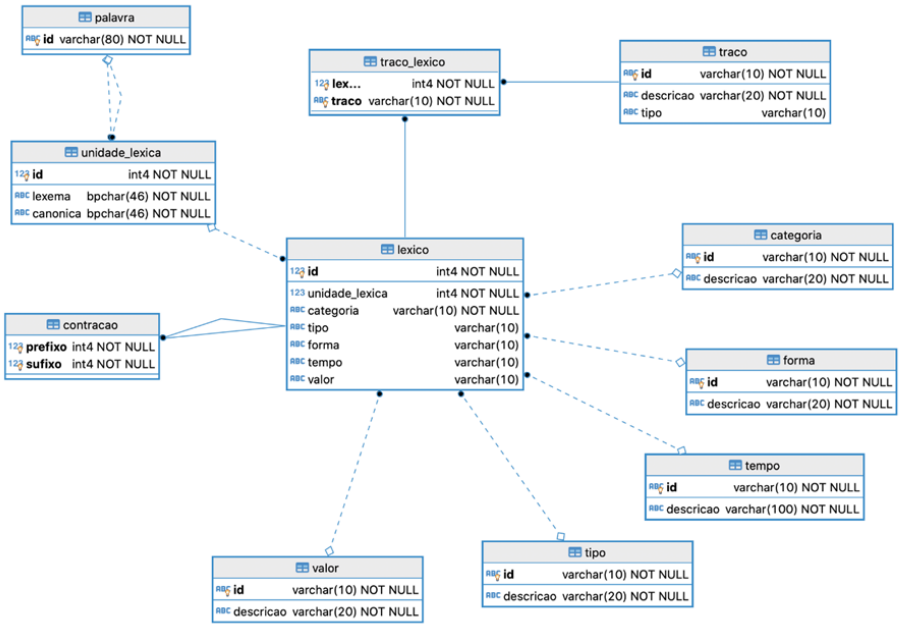
Um sinal de adição ("+") foi utilizado para conectar classes gramaticais a subtipos ou traços, que podem trazer informações semânticas. Por exemplo, "ART+**Def**", trata-se de um artigo definido; "ART+Ind", artigo indefinido; "PRO+Dem", pronome demonstrativo etc. Cada classe pode ser associada a traços de acordo com sua categoria. Os substantivos podem ser próprios ou coletivos; os numerais podem ser cardinais, ordinais, multiplicativos ou fracionários; os artigos podem ser definidos ou indefinidos; os pronomes podem ser demonstrativos, relativos, interrogativos, de tratamento, possessivos ou pessoais; as conjunções podem ser coordenativas, subordinativas ou correlativas. Esses traços das conjunções podem permitir, no futuro, construir algoritmos que possam detectar as relações de sentido entre orações.

Ao fim de cada linha do arquivo *dic*, podia ou não haver um sinal de dois pontos (":"), que foi utilizado para delimitar informações de flexão, tempo, forma, valor, gênero, número, grau, pessoa. No dicionário, um verbete podia conter muitas flexões.

A implementação do Banco de Dados a partir do DELAF_PB foi feita com o Sistema Gerenciador de Banco de Dados (SGBD) PostgreSQL 12.7 (THE POSTGRESQL GLOBAL DEVELOPMENT GROUP, 2021) e apoiada pela ferramenta de administração Dbeaver (DBEAVER, 2021). A operação de alimentar o Banco foi realizada por meio de algoritmos de conversão das entradas do DELAF-PB (MUNIZ, 2015) e foi codificada em linguagem Go 1.16.4 (GOLANG, 2021).

Inicialmente, as entradas do dicionário foram submetidas a uma minuciosa análise em busca de padrões que pudessem permitir transformar a estrutura do DELAF_PB para a modelagem de um banco de dados. Com o SGBD e por meio de scripts em Structured Query Language (SQL), foram criadas 11 tabelas (Esquema 1). A tabela principal (Léxico) contém 6 atributos. Uma unidade léxica foi formada pela união de uma palavra em sua forma flexionada e não flexionada (canônica). Cada classe gramatical foi associada a um núcleo funcional descrito em Othero (2009, p. 141), que funciona como chave de identificação primária. O produto final, o Banco de Dados populado, foi documentado e disponibilizado em (PACHECO; GUARANHA, 2021)

Esquema 1 - Modelo entidade-relacionamento do banco de dados



Fonte: Autoria própria.

No processo de construção do Banco de Dados, foram atendidas algumas especificidades quanto à natureza do nosso *corpus*. Uma unidade do léxico como "casa", por exemplo, pode pertencer a duas categorias gramaticais, verbo e substantivo. Caso seja verbo, a forma canônica corresponde ao verbo no infinitivo "casar", caso seja substantivo, a forma canônica corresponde ao substantivo feminino "casa". Para dar conta dessa ambiguidade, foram criadas três tabelas: *Palavra*, *Unidade Léxica e Léxico*. A duplicação dentro de uma base pode ser um desperdício de memória e tomar tempo maior para a consulta de dados. É importante por isso que ela seja evitada. A estratégia de criar essas três tabelas permitiu que a unidade lexical "casa" fosse armazenada apenas uma vez na tabela *Palavra* e que sua chave primária fosse herdada pela tabela *Unidade Léxica* a qual, por sua vez, transmite a chave para a tabela *Léxico* (Esquema 1).

Consideramos o lexema associado à sua forma-canônica como uma unidade lexical que embasa a estrutura do nosso Banco de Dados assim como ocorre no *corpus* do DELAF_PB. Para as categorias de pessoa, gênero, grau e número manteve-se o padrão linguístico de reuni-las dentro da tabela de traço semântico (*Traço*). Isso está de acordo também com o que propôs Bryce-Codd (1972 *apud* LAKE e CROWTHER, 2013, p. 75) em sua "Quarta Forma normal" para padronização de dados. Essa solução permitiu contemplar a característica do nosso Banco de Dados em que uma classe pode conter muitos traços e um traço semântico pode ser atribuído a várias classes, conforme Esquema 1.

Finalmente, adotamos a estratégia para dar conta de combinações e contrações de palavras representadas no DELAF pela união de duas classes pela letra "X" (como em *PREPXART* para o lexema "da", preposição "de" mais artigo "a"). Nem combinações e nem contrações envolvem diretamente categorias gramaticais e, por definição, têm independência em relação a elas. Sem perda de seu conceito original, foi possível simplesmente agrupá-las em uma nova entidade com cada entrada identificada pelo prefixo e sufixo correspondente. São essas novas entidades as seguintes: PROXPRO, contração de pronome com pronome como "aqueloutra", por exemplo; PREPXADV, contração de preposição mais advérbio, como "dali"; PREPXART, contração de preposição mais artigo, como nos casos de crase "à" e "do", no sentido de posse, como em "carro do homem"; PREPXPRO, contração de preposição mais pronome, como "nalguma", com sentido de em alguma; e PREXPREP, como em "dentre".

O Banco de Dados povoado contém 860.778 palavras, 879.155 unidades léxicas e um total de 1.193.295 unidades léxicas classificadas (uma unidade léxica pode pertencer a mais de uma categoria lexical). A descrição de todas as categorias gramaticais encontra-se documentada em Pacheco e Guaranha (2021).

Aspectos técnicos e regras utilizadas na construção do *Parser Parsero*, um algoritmo de Processamento de Linguagem Natural (PNL)

O processador sintático *Parsero* foi desenvolvido em linguagem GO (GOLANG, 2021) sob o padrão MVC (Modelo-Visão-Controlador), que é uma base comum para muitos modelos baseados em *Web* (SOMMERVILLE, 2011, p.108). O programa é uma API (*Application Programming Interface*) criada com a biblioteca *Gorilla Mux* (GORILLA, 2021). Esse padrão de projetos MVC tem três partes, assim utilizadas em nosso *Parser*: no Modelo, primeira parte, ficam as funções para consulta e inserção dos dados na base léxica, a qual foi descrita na seção anterior. A Visão, segunda parte, abriga um único endereço que, ao ser chamado, aciona o Controlador, terceira parte, em que estão abrigadas as instruções do algoritmo. O Controlador, então, executa as seguintes tarefas: recebe a sentença, consulta a base de dados e retorna as possibilidades de classificação para cada palavra. Depois disso, recebe esse resultado e percorre as regras de produção, as quais detalharemos nesta seção (Quadro 1) até encontrar uma válida para a sequência. No final, retorna um arquivo de texto com as unidades lexicais rotuladas de acordo com suas categorias, agrupadas em sintagmas e encapsuladas em formato (JSON) *JavaScript Object Notation* (ECMA-404, 2017).

O método escolhido para percorrer as regras foi a Análise Sintática Descendente Recursiva (ASDR) com derivação à esquerda (THAIM, 2020, p. 37). As regras de produção formam uma árvore que pode ser percorrida da raiz, a sentença, até as folhas, os núcleos funcionais. Nessa estrutura, os sintagmas são os ramos da árvore e os nós que ficam localizados onde seriam as folhas são representados pelos núcleos sintáticos definidos nas regras (Quadro 1). A ASDR caminha pela árvore da raiz até as folhas sempre executando primeiro o nó à esquerda. Em nossa abordagem, ela recebe um vetor com as classes possíveis para cada palavra e, ao atingir uma folha da árvore, compara cada possibilidade de classificação com o núcleo funcional da regra vigente. O ciclo é repetido até não haver mais possibilidades.

O algoritmo completo para um conjunto de regras de produção de tamanho *n* em que cada possibilidade de produção é representada pelo símbolo βi é o seguinte:

```
Função (sentença)
{
        1) escolher uma produção α tal que α → β1 | β2 | … | βn
                1.1) para (cada i de 1 até n)
                1.1.1) se (βi é um não terminal NT)
                        a) invocar procedimento βi( )
                1.1.2) senão se (βi é um terminal Σ) e (βi == símbolo atual)
                        b) avance na sentença para o próximo símbolo
                1.1.3) senão
                        c) retorna regra inválida
                1.1.4) retorna regra válida
}
```

Como cada regra de produção pode evocar a si mesma a cada momento, o algoritmo tem natureza recursiva. Uma sentença é considerada inválida se não pode ser expressa por nenhuma das regras de produção estabelecidas e, se há um erro, ele é classificado em erro de léxico (quando não encontra palavra no Banco de Dados), erro sintático (sentenças agramaticais) ou erro interno (evento inesperado como queda da rede, problema de acesso ao Banco de Dados etc). A interface gráfica final foi construída com a biblioteca *React* em linguagem EcmaScript 6 (ECMA, 2015).

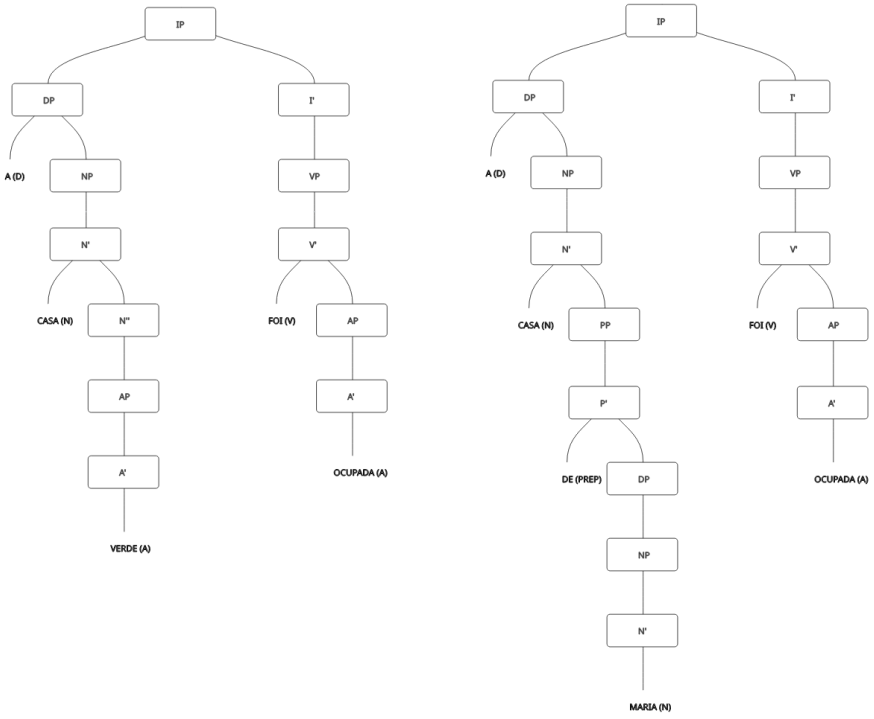
Para a construção das regras de produção, utilizamos como base aquelas propostas por Othero (2009), apresentadas no Quadro 1 que, por sua vez, seguem a Teoria X-Barra que é, no dizer de Chomsky (2014, p. 389-398), uma segunda fase, para além dos "primórdios da gramática da estrutura sintagmática" (p. 390). Um dos pontos destacados por Chomsky acerca dessa teoria é que

> ao invés de aparecer como nas gramáticas de estrutura sintagmática tentando construir a estrutura de cima para baixo ('*top-down*'), aqui ela se move dos

> itens lexicais para cima. Assume que os itens lexicais já vêm colocados em um pequeno conjunto de categorias possíveis, em que cada item em cada categoria possui um conjunto de traços que na verdade dizem o que o item lexical relevante pode fazer em uma derivação/computação. Assim, na medida em que um item lexical se 'projeta' por meio da estrutura, ele carrega seus traços com ele, e estes determinam como eles podem se combinar, onde, e como eles irão ser lidos. (CHOMSKY, 2014, p. 392)

Desse modo, é possível elaborar macrorregras gramaticais que prevejam estruturas com complementos e adjuntos, por exemplo. As regras são orientadas para conseguir representar estruturas sintáticas cujo núcleo exige ou aceita complementos e regras gramaticais que possibilitem analisar sintagmas cujo núcleo não aceita ou não exige complementos. Empregando os conceitos da Teoria X-Barra, nosso *Parser* consegue lidar com sentenças com em (10) "A casa verde" e (11) "A casa de Maria", conforme Figura 2:

Figura 2 – Diagramas arbóreos de Sintagmas Nominais – Teoria X-Barra



Fonte: Autoria própria. Interface gráfica do *Parsero* 1.0.

O diagrama à esquerda representa um adjunto adnominal e o da direita um complemento nominal. Não contemplamos, neste projeto, a distinção entre complementos e adjuntos, uma vez que os sintagmas preposicionais podem assumir tanto as funções de complemento nominal quanto de adjunto adnominal e que, para distinguir

essas funções, teríamos de recorrer a critérios de ordem "predominantemente semântico-pragmática" (SOUZA e SILVA; KOCH, 1983, p. 21).

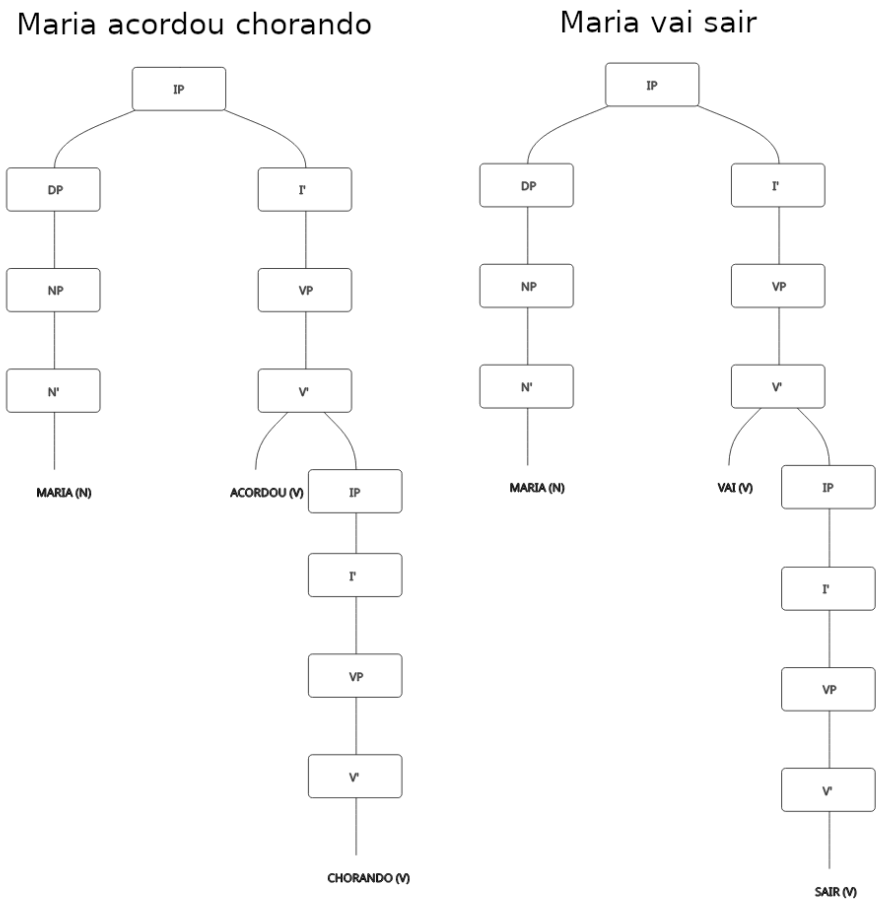
Em Othero (2009, p. 142), as regras de produção são formadas por um núcleo funcional único que pode ser: Determinante (D), Substantivo (N), Verbo (V), Adjetivo (A), Advérbio (Adv), Preposição (P), Quantificador (Q), Numeral (Num). Também poder ser formado por um sintagma ou por um nó X-Barra, que pode conter ou não um complemento, por isso a notação (') aparece ao lado dele.

Nas regras definidas por Othero (2009, p. 142)[3], cada um dos Sintagmas é composto por um núcleo funcional que representa o símbolo central do sintagma. Foram definidos os seguintes sintagmas: a) Nominais (SN): têm como núcleo os Substantivos (N); b) Determinantes (SD): podem conter como núcleo Artigos, Pronomes Demonstrativos, Pronomes Pessoais ou estarem ocultos na sentença. São representados como determinantes (D); c) Numerais (SNum): têm como núcleo os Numerais (Num); d) Possessivos (SPoss): têm como núcleo os Pronomes Possessivos (Poss); e) Quantificadores (SQ): têm como núcleo os quantificadores como algum, todos, cada, nenhum; f) Adjetivais (SA): têm como núcleo os Adjetivos (A); g) Adverbiais (SAdv): têm como núcleo os Advérbios (Adv); e h) Verbais (SV): têm como núcleo os Verbos (V).

Além destes sintagmas, o autor propõe, ainda, dois sintagmas para lidar com sentenças complexas. O primeiro, o Sintagma Inflexional (SI), dá conta das sentenças que contêm mais de um verbo em sua estrutura. Quando isso ocorre, esses dois verbos são genericamente entendidos como locuções verbais pelo nosso *Parser*, independente de serem essas locuções constituídas ou não por um verbo principal, que não se flexiona, mais um auxiliar, como em (12) "Maria vai sair" (Figura 3, à esquerda), ou por dois verbos principais, que são passíveis de serem flexionados, como em (13) "Maria acordou chorando" (Figura 2, à direita). Para uma discussão mais profunda sobre locuções verbais, encaminhamos o leitor para Othero (2009, p. 126-131).

Figura 3 – Sentença com verbo auxiliar e com dois verbos principais

[3] Optamos por adaptar a terminologia das regras de Othero (2009), que usa a terminologia inglesa *Phrasal*. Preferimos, como estamos trabalhando com o Português Brasileiro, a expressão Sintagma. Desse modo, usaremos SN, SV, SA, Sadv, SP e assim por diante, como Sintagma Nominal, Sintagma Verbal, Sintagma Adjetival, Sintagma Adverbial, Sintagma Preposicionado etc.



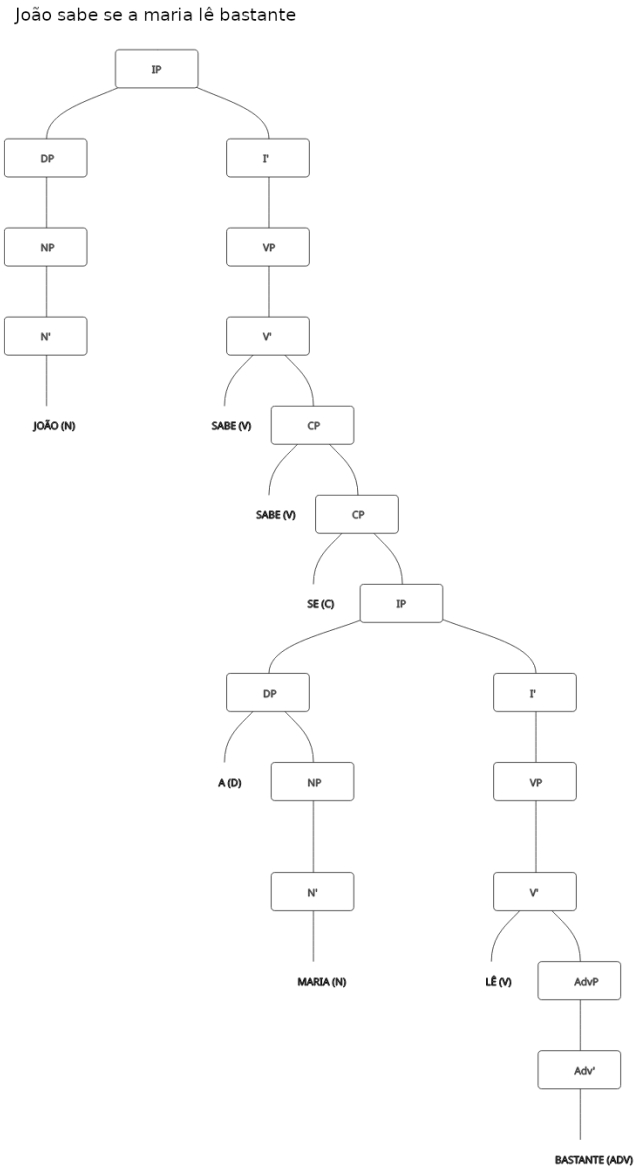
Fonte: Autoria própria. Interface gráfica do *Parsero 1.0*.

Além disso, recorremos à solução de Othero (2009) para tratar estruturas com orações subordinadas conectadas por conjunções, acrescentando o SC (Sintagma Complementador), conforme descrito pelo autor:

> A posição de núcleo do CP [S em nosso *Parser*] será ocupada por **que**, **se** ou **quando** (como em João sabe se a Maria lê bastante; João sabe quando a Maria chegou). Com essas mesmas regras, podemos analisar também frases que contenham uma estrutura verbo auxiliar + verbo principal dentro do IP complemento (A Maria disse que iria chegar mais tarde) (Othero, 2009, p. 139, grifos do autor)

Nosso *Parser* executou corretamente essas estruturas, conforme exemplo da Figura 4, (14) "João sabe se a Maria lê bastante".

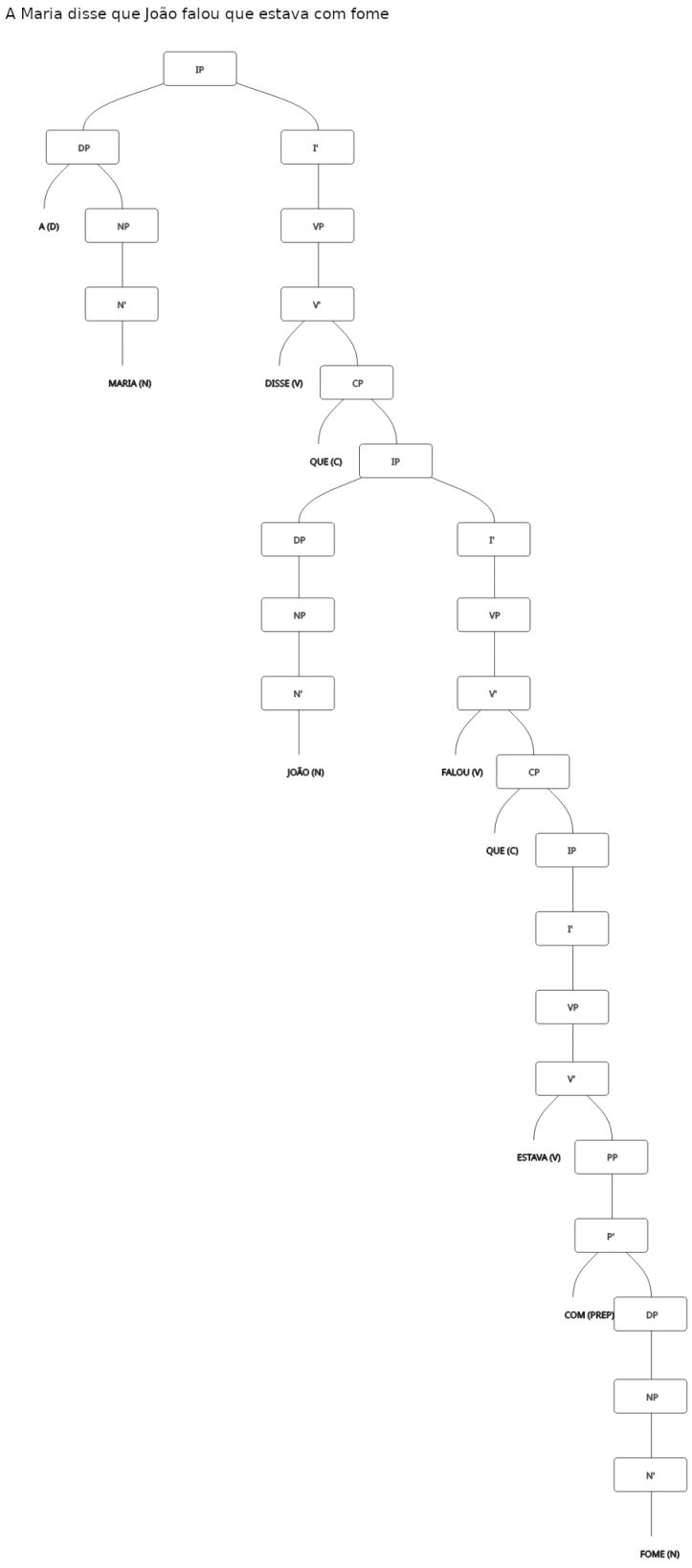
Figura 4 – Exemplo de sentença com Sintagma Complementador



Fonte: Autoria própria. Interface gráfica do *Parsero 1.0*.

Além disso, o SC permite que se processe frases com estruturas recursivas como em (15) "A Maria disse que João falou que estava com fome" (Figura 5).

Figura 5 – Exemplo de sentença com estrutura recursiva



Fonte: Autoria própria. Interface gráfica do *Parsero 1.0.*

Quanto à ordem de processamento das regras, nosso Parser adotou o método de

processar primeiro as mais complexas, em que há mais do que um núcleo funcional na regra. Por exemplo, em (16) "O carro azul" opta-se primeiro pelo processamento das regras em que o substantivo (N) é precedido ou sucedido de um adjetivo (A) (regras N' → N' SA e N' → SA N') e somente se essas regras não são válidas, parte-se para a regra mais simples em que só há um substantivo (N' → N). Não fosse desse modo, em estruturas como (16), o processamento terminaria antes que todas as palavras pudessem ser avaliadas porque a regra anterior (N' → N) já seria executada com sucesso, o adjetivo seria descartado e seguia-se para o restante da sentença. Com essa mudança foi necessário também adaptar regras que poderiam resultar em recursões infinitas à esquerda, conforme exemplo no Quadro 1.

Quadro 1 - Exemplo de eliminação da recursão à esquerda

Antes	**Depois**
SN → N'	SN → N'
N' → N' SP	N' → N N''
N' → N	N'' → SP N''
	N'' → nulo

Fonte: Autoria própria.

No Quadro 1, o Sintagma Nominal apresentava recursividade infinita à esquerda na regra (N' → N' SP) que foi substituída pelas regras (N' → N N'') e (N'' → SP N''). O símbolo N'' foi criado, podendo não conter valor nenhum. Quando aplicado a todas as regras, esse raciocínio levou à formação de novos símbolos X-Barra (Quadro 2). Em alguns casos, o elemento nulo foi explicitamente usado para identificar a possibilidade de ocultar o elemento.

Este raciocínio segue o método da fatoração à esquerda que consiste na reescrita de uma regra para adiar seu processamento até que ele seja possível (THAIN, 2020, p. 41). Por exemplo, pelas regras de Othero a frase (17) "O jogador de futebol caiu" geraria um ciclo infinito em nosso algoritmo ao tentar classificar a palavra "jogador" porque o nó N' retorna a si mesmo (na sequência N' → N'), se não fossem especificadas outras regras para que o processador não percorra novamente regras pelas quais já passou. Pelas novas regras, ao chegar ao nó N' é obrigatória a existência de um substantivo (N' → N N''). Se não for possível classificar a palavra como substantivo, o algoritmo não

prossegue. Feitas essas adaptações, a Gramática do *Parsero* ficou composta pelas regras apresentadas no Quadro 2.

Quadro 2 - Regras de Produção para o *Parser Parsero*

Sintagma Inflexional SI → SD I' SI → I' I' → I" SV I" → I I" I" → SD I" I" → nulo Sintagma Complementizador SC → C SI; Sintagma Determinante SD → D SN SD → D SPoss SD → D SNum SD → SN SD → SPoss SD → SNum SD → D Sintagma Nominal SN → N' N' → N SP N' → N SC N' → N N" N' → AP N' N" → SC N" N" → SP N" N" → SA N" N" → nulo;	Sintagma Numeral SNum → Num SN SNum → Num SP; Sintagma Possesivo SPoss → Poss SN SPoss → SN Poss SPoss → Poss SNum; Sintagma Quantificador SQ → Q SD SQ → Q SP SQ → SD Q; Sintagma Adjetival SA → A' A' → SAdv A' A' → A SP A' → A SC A' → A A" A" → SAdv A" A" → SP A" A" → nulo;	Sintagma Preposicional SP → P' P' → SAdv P' P' → P SD P' → P SAdv P' → P SC P' → P SP P' → P; Sintagma Adverbial AdvP → Adv' Adv' → Adv SP Adv' → Adv Adv" Adv" → Adv' Adv" Adv" → nulo; Sintagma Verbal SV → V' V' → SAdv V' V' → V SD V' → V SP V' → V SC V' → V SA V' → V SAdv V' → V SI V' → V V" V" → SAdv V" V" → PP V" V" → nulo;

Fonte: Adaptação de Othero (2009, p. 142)

A única alteração para evitar ambuiguidades foi feita para o caso de precedência de adjetivos sobre substantivos, que pode ocorrer na Língua Portuguesa, como em (18) "O pobre homem de Roma morreu". Quando uma palavra pode pertencer a essas duas categorias, adotou-se a regra de prioridade para que as palavras que vêm em primeiro lugar sejam classificadas como substantivos e as que vêm em segundo como adjetivos,

pois é a forma mais comum no Português Brasileiro. Demais ambiguidades não foram tratadas. As Figuras 3, 4 e 5 são resultados gerados pelo aplicativo .

Considerações finais

Este artigo apresentou a construção de um *Parser* Sintático para o Português Brasileiro que utilizou o recurso disponível na Internet, o Dicionário DELAF_PB (MUNIZ, 2004) e as regras propostas por Othero (2009, p.142) estas, por sua vez, baseadas no conceito de Gramática Gerativa, de Chomsky (2015), e expandidas pela Teoria X-Barra (CHOMSKY, 2014; OTHERO, 2009).

A partir das regras, empreendemos um intenso processo de refatoração para adaptá-las ao algoritmo utilizado (Algoritmo Descendente Recursivo), o que nos levou à criação de novas regras que eliminaram os ciclos recursivos à esquerda facilitando o processamento.

A base léxica no formato DELA foi adaptada para o formato SQL e foi disponibilizada em código aberto. O projeto do Banco de Dados respeita o formalismo de Bryce-Codd (1972 apud LAKE e CROWTHER, 2013, p. 75) e evitou duplicações para poupar memória de processamento. A estrutura proposta permitiu também que novas palavras pudessem ser inseridas sem classificação o que priorizou a expansão do léxico. Ela deixa aberta a possibilidade no futuro de que as palavras possam ser classificadas manualmente por usuários ou por métodos matemáticos apreendidos a partir da base já existente posteriormente.

A tela de interação com o usuário foi desenvolvida em linguagem GO sob o padrão MVC e também está disponibilizada. Para processar os resultados, foi gerada uma API para análise de sentenças do Português Brasileiro que é capaz de processar sentenças simples e compostas sem pontuação; e uma interface gráfica capaz de apresentar os sintagmas de uma sentença e sua árvore sintática.

Além de ter propiciado reflexões sobre as potencialidades da Gramática Gerativa para o desenvolvimento de Processadores de Linguagem Natural, notadamente para atender às sofisticadas construções sintáticas do Português Brasileiro, este trabalho também constituiu uma oportunidade de pesquisa interdisciplinar que busca aproximar as ciências da linguagem e as ciências da computação. Além disso, disponibilizar os resultados, tanto a base de dados quanto o algoritmo, é um estímulo para que outros pesquisadores possam aperfeiçoar ou possam criar outros processadores sintáticos que serão a base para sistemas inteligentes os quais possam interagir com falantes do Português.

Referências

CHOMSKY, Noam. *Estruturas sintáticas*. São Paulo: Vozes, 2015.

_____. *A Ciência da linguagem*. São Paulo: Editora UNESP, 2014.

DBEAVER, Community. Dbeaver: Free Universal Database Manager. Versão 7.2.0. Disponível em: <https://dbeaver.io/download/>. Acesso em 16 Out. 2021

ECMA, Internacional. ECMAScript 2015 Language Specification. 2015. Disponível em: <https://262.ecma-international.org/6.0/>. Acesso em 16 Out. 2021.

ECMA-404, Internacional. The JSON data interchange syntax. 2017. Disponível em: <https://www.ecma-international.org/publications-and-standards/standards/ecma-404/>. Acesso em 16 Out. 2021.

GOLANG. GoLang. Versão 1.16.4. Disponível em: <https://pkg.go.dev/runtime>. Acesso em: 16 Out. 2021.

GORILLA, The Authors. Gorilla Web Toolkit. Versão 1.8.0 [S.I.]. 2021. Disponível em: <https://github.com/gorilla/mux. Acesso em 16 Out. 2021.

JÄGER G.; ROGERS J. Formal language theory: refining the Chomsky hierarchy. Phil. Trans. R. Soc. B3671956–1970. 2012. Disponível: <http://doi.org/10.1098/rstb.2012.0077>

LAKE, P.; CROWTHER P. Concise Guide to Databases. Springer. 307p. London, 2013

MUNIZ, M. C. M. A construção de recursos linguístico-computacionais para o português do Brasil: o projeto de Unitex-PB. Dissertação de Mestrado. Instituto de Ciências Matemáticas de São Carlos, USP. 72p. 2004. Disponível em: <http://ladl.univ-mlv.fr/brasil/bibliografia/oto/DissMuniz2004.pdf>. Acesso em 16 Out. 2021.

_____.DELAF-PB: Dicionário de Palavras Simples Flexionadas para o Português Brasileiro. Versão 2 [S.I.]. 2015. Disponível em: <http://www.nilc.icmc.usp.br/nilc/projects/unitex-pb/web/dicionarios.html>. Acesso em 16 Out. 2021.

OTHERO, Gabriel de Ávila. A gramática da frase em português: algumas reflexões para a formalização da estrutura frasal em português – Dados eletrônicos. – Porto Alegre : EDIPUCRS, 2009. 160 p.

PARSERO: Parser Sintático para o Português Brasileiro. 2021. Disponível em < https://parserov1.herokuapp.com>. Acesso em: 22/11/2021.

SOMMERVILLE, Ian. Engenharia de Software — 9. ed. — São Paulo: Pearson Prentice Hall, 2011.

SOUZA e SILVA, Cecília P. de; KOCH, Ingedore Villaça. *Linguística aplicada ao português*: sintaxe. São Paulo: Cortez, 1983.

THAIN, Douglas. Introduction to compilers and language design. Second Edition. Independently Published. 2020

The PostgreSQL Global Development Group. PostgreSQL Database Management System. Versão 12.7. Disponível em: <https://www.postgresql.org/download/>. Acesso em 16 Out. 2021.

USP, Universidade de São Paulo; NILC, Núcleo Interinstitucional de Linguística Computacional; ICMC, Instituto de Ciências Matemáticas e de Computação. Projeto UNITEX-PB, 2004. Disponível em: <http://www.nilc.icmc.usp.br/nilc/projects/unitex-pb/web/index.html>. Acessado em 12 out. 2021.