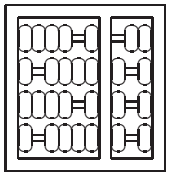

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Algoritmos Baseados em Teoria das Veias para Resolução de Pronomes

F. J. V. Silva *A. M. B. R. Carvalho*
N. T. Roman

Technical Report - IC-12-24 - Relatório Técnico

October - 2012 - Outubro

Algoritmos Baseados em Teoria das Veias para Resolução de Pronomes

Fernando José Vieira da Silva Ariadne Maria Brito Rizzoni Carvalho*
Norton Trevisan Roman†

Resumo

A resolução automática de anáforas pronominais é uma tarefa bastante árdua, apresentando inúmeras dificuldades, especialmente quando existe mais de um candidato a referente. Ao longo dos anos, diversas estratégias foram propostas para lidar com esse desafio, muitas delas baseadas exclusivamente em informações sintáticas presentes no texto. Em constraste com essas estratégias, a Teoria das Veias provê base para restringir o domínio de referência de um pronome, ao identificar "veias" sobre a árvore de estrutura de discurso. Entretanto, essa teoria é pouco explorada para resolução automática de pronomes. Neste trabalho adaptamos três algoritmos, baseados em Teoria da Centralização, para utilizar os conceitos de Teoria das Veias. Como resultado, avaliamos cada um desses algoritmos, comparando-os com seus originais (baseados em Teoria da Centralização), verificando as contribuições de cada uma das teorias para a tarefa de resolução automática de pronomes em língua portuguesa.

1 Introdução

Anáfora é um fenômeno linguístico que ocorre quando há uma referência abreviada a outro elemento prévio do texto. Essa abreviação é chamada de *anáfora*, enquanto o elemento referenciado é chamado de *referente* [12]. O foco deste trabalho são as anáforas pronominais, que ocorrem quando a anáfora é um pronome. Considere, por exemplo, a seguinte sentença: "João e **Maria** foram passear no zoológico, apesar de **ela** preferir o shopping". Nesse caso, o pronome "ela" é facilmente identificado como se referindo a "Maria", devido à concordância de gênero e número entre o nome próprio e o pronome. Contudo, a complexidade da tarefa de encontrar um referente para uma anáfora aumenta consideravelmente na presença de mais de um possível referente, como na sentença: "O **cachorro** de João fugiu esta manhã. **Ele** latiu e assustou o carteiro". Nessa sentença, tanto "cachorro" quanto "João" são candidatos a referentes da anáfora "Ele", pois ambos concordam em gênero e número com o pronome, sendo tal ambiguidade resolvida somente no nível semântico.

A resolução automática de anáforas é essencial para diversos tipos de sistemas de processamento de língua natural, como sistemas de extração de informação, sumarização e

*Instituto de Computação, Unicamp, Brasil
†Escola de Artes, Ciências e Humanidades, USP, Brasil

geração automática de textos. Considere, por exemplo, o sistema apresentado na Figura 1, extraído de um conhecido sistema *online* de tradução automática de textos em português para o inglês. Nesse exemplo, pode-se verificar que o sistema não foi capaz de encontrar o referente correto para "Ele" na sentença "O cachorro de João fugiu esta manhã. Ele latiu e assustou o carteiro", que foi traduzida como *"John's dog ran away this morning. He barked and scared the mailman"*. Em consequência disso, "Ele" foi traduzido para "*he*", levando o leitor da sentença em inglês a acreditar que João latiu.

Figura 1: Tradução onde o referente para a anáfora não foi encontrado corretamente.

Na busca por soluções para a resolução automática de anáforas pronominais, diversas estratégias foram exploradas ao longo dos anos. Algumas delas, como o algoritmo original de *Hobbs* [13] e o algoritmo de *Lappin e Leass* [14], por exemplo, utilizam somente informações sintáticas contidas no texto, enquanto outras abordagens utilizam técnicas de aprendizado de máquina (*e.g.* [8]), ou se baseiam em teorias linguísticas para a resolução de anáforas pronominais (*e.g.* [1]).

Nesse último grupo, encontram-se também as abordagens com base na Teoria da Centralização (*e.g.* [1]) e na Teoria das Veias (*e.g.* [22]) que, dentre outras coisas, trabalham com a noção de coerência do discurso. Assim, essas teorias fornecem subsídios para a resolução de pronomes, ao dar preferência a referentes que mantenham a coerênca do discurso, restando, quando isso não for possível, a intuição de que o segmento do discurso é menos coerente.

As vantagens e desvantagens do uso de Centralização para resolução pronominal foram exploradas em [9]. O objetivo do trabalho aqui proposto é identificar se a aplicação da Teoria das Veias representa algum ganho para esta tarefa e comparar as duas abordagens. O restante do relatório está organizado como segue: a Seção 2 apresenta a Teoria da Veias, os algoritmos utilizados para identificar as veias e os conceitos utilizados na resolução de pronomes; as Seções 3, 4 e 5 descrevem os algoritmos adaptados para Teoria das Veias; a Seção 6 traz a análise da complexidade desses algoritmos; a Seção 7, por sua vez, traz a avaliação dos resultados e comparação entre as teorias, enquanto que a Seção 8 apresenta as conclusões.

2 Teoria das Veias

A Teoria da Estrutura Retórica (RST[1]) fornece um modelo para a representação do discurso através de relacionamentos entre suas unidades discursivas [20]. Segundo essa teoria, o discurso é composto por uma série de unidades discursivas, sendo que duas unidades discursivas adjacentes podem estar relacionadas de tal forma que uma delas desempenhe um papel específico em relação à outra. Relacionamentos podem ocorrer entre uma unidade discursiva e outro relacionamento, ou entre dois relacionamentos, gerando assim uma árvore com a estrutura do discurso (existindo, em algumas situações, várias árvores para o mesmo discurso).

Os relacionamentos podem ser de diversos tipos, como Elaboração, Avaliação, Parentético, Mesma Unidade, Circunstância, Fundo, Concessão, entre outros[2]. Na maioria dos relacionamentos, um dos nós representa o núcleo e o outro seu satélite. O núcleo é uma unidade de texto indispensável para a compreensão do segmento do discurso, enquanto que a função do satélite é complementar o significado do segmento do discurso. A Figura 2 mostra um exemplo de árvore RST. Nessa figura, as setas apontam para os núcleos, como ocorre na relação de Avaliação entre a primeira e a segunda unidade discursiva, sendo que o núcleo apresenta uma situação ("Cientistas britânicos detectaram, em adultos, a produção de células hepáticas a partir de células tronco da medula óssea"), enquanto o satélite faz um comentário que avalia essa situação ("Foi dado o primeiro passo para a diminuição das filas de espera para transplante de fígado"). Algumas vezes há relacionamentos entre dois núcleos, em que nenhuma seta é mostrada, como por exemplo na relação de Mesma Unidade, encontrada entre a última unidade discursiva ("dentro do organismo humano") e o relacionamento Parentético, formado pela penúltima e ante-penúltima unidades discursivas ("mostra que além disso elas são capazes de originar outro tipo de células" e "células hepáticas"), o que indica que essas unidades discursivas compõem uma única unidade com o mesmo significado no discurso.

Originalmente elaborada com o intuito de expandir o conceito de coerência de discurso, estabelecido pela Teoria da Centralização, do nível local para o global [7], a Teoria das Veias considera que as entidades presentes nas unidades discursivas RST podem ser agrupadas na forma de uma veia sobre a árvore de estrutura do discurso, de tal modo que a veia de um nó na árvore contenha todas as entidades indispensáveis para a compreensão do trecho do discurso coberto por esse nó [7]. Esse reagrupamento, por sua vez, deveria ser suficiente para o referenciamento de qualquer pronome (pois, do contrário, haveria uma entidade indispensável fora da veia), reduzindo assim o conjunto de possíveis candidatos.

2.1 Identificação das Veias

Segundo a Teoria das Veias, a cada nó da árvore RST são associados **cabeçalhos** e **veias**. Adicionalmente, cada nó terminal (representando uma **unidade discursiva RST**) possui um **rótulo**. Em nossa implementação, esse rótulo é obtido pela concatenação de todas as entidades nele realizadas. O cabeçalho de um nó terminal é seu próprio rótulo, enquanto o

[1] Em inglês: *Rethorical Structure Theory*.

[2] Para uma lista completa sobre os possíveis relacionamentos, consulte [20].

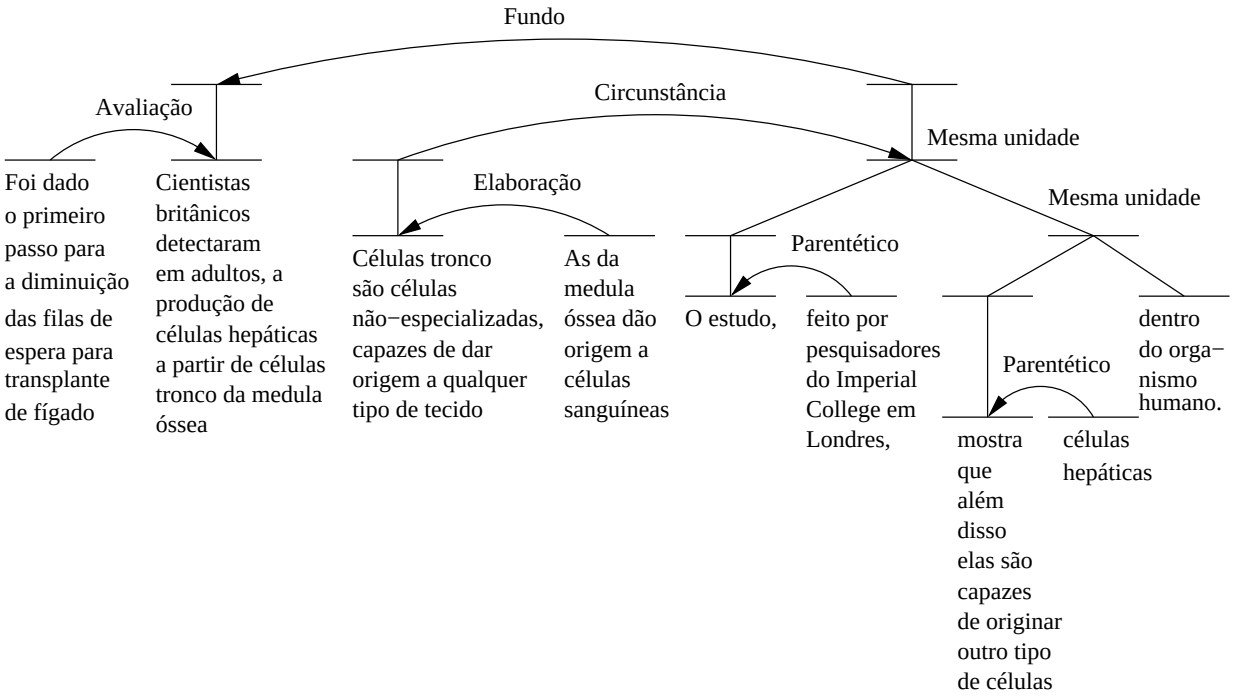

Figura 2: Árvore RST representando a estrutura do discurso.

cabeçalho de um nó não terminal é a concatenação dos cabeçalhos de seus filhos nucleares. Dessa forma, a identificação das veias em uma árvore RST é feita em duas etapas: 1) identificação dos cabeçalhos, em um processo *bottom-up*, descrito pelo Algoritmo 1; e 2) identificação das veias, em um processo *top-down*, descrito pelo Algoritmo 2 [7].

```
   Entrada: node = Nó da árvore RST
   Saída: node
 1 início
 2     se node é terminal então
 3         node.head ← node.label;
 4     senão
 5         para child ∈ node.children faça
 6             Identificar cabeçalhos em child;
 7             se child é Núcleo do relacionamento RST então
 8                 para en ∈ child faça
 9                     node.head ← node.head + en;
10                 fim
11             fim
12         fim
13     fim
14     retorna node
15 fim
```

Algoritmo 1: Algoritmo para identificação dos cabeçalhos.

A análise da complexidade dos algoritmos para identificação de cabeçalhos e veias considera como entrada o corpus Sum-it, formado por artigos jornalísticos retirados da seção

de ciências e saúde do jornal "Folha de São Paulo"[3], anotados sintaticamente pelo parser "PALAVRAS"[2][4], possuindo também anotações de relacionamentos retóricos segundo a Teoria RST, feitas manualmente por especialistas [4]. Assume-se que há O(S) nós na árvore RST, onde S é o número de unidades discursivas RST do texto segmentado. A identificação dos cabeçalhos, descrita pelo Algoritmo 1, é executada a partir do nó raiz, sendo repetida para cada nó descendente, recursivamente em pré-ordem. Supondo-se que cada unidade discursiva RST possui um número constante de entidades em seu rótulo, temos a complexidade O(Sh), onde h é a altura da árvore RST. Como h = O(S), temos a complexidade O(S^2) no pior caso.

Uma vez identificados os cabeçalhos, a etapa seguinte é identificar as veias dos nós. A veia do nó raiz é representada pelo seu próprio cabeçalho, enquanto as veias dos outros nós são definidas da seguinte forma:

1. Se o nó for nuclear e possuir um irmão satélite à esquerda cujo cabeçalho é h, então sua veia é a concatenação das palavras que forman h e a veia de seu pai. Nessa concatenação, as palavras presentes em h são marcadas por parênteses. Todavia, se não houver irmão satélite à esquerda, então a veia será a mesma de seu pai;

2. Se o nó for satélite e for o filho mais à esquerda de seu pai, então a veia é a concatenação de seu cabeçalho com a veia de seu pai. Do contrário, a veia é a concatenação de seu cabeçalho com cada entidade não marcada contida na veia de seu pai. O conjunto de entidades não marcadas é obtido através da função *simpl(v)*.

De forma semelhante à concatenação de cabeçalhos, o processo de identificação de veias, descrito pelo Algoritmo 2, também é executado a partir do nó raiz e depois para cada nó decendente, mas em pós-ordem. Como a veia do nó raiz é seu próprio cabeçalho, essa referência (linha 3) leva tempo O(*1*). Assim como na identificação de cabeçalhos, supomos que cada nó da árvore RST possui um número constante de entidades em sua veia. Dessa forma, a identificação das veias também possui complexidade O(S^2).

2.2 Veias e Resolução de Pronomes

A Teoria das Veias também introduz o conceito de **domínio de acesso**, que restringe os candidatos a referente de um determinado pronome àquelas entidades encontradas na veia do nó da árvore RST em que o pronome em questão se encontra. Com isso, a Teoria das Veias parte de alguns princípios que podem auxiliar na resolução da referência pronominal. São eles:

1. **Um satélite ou núcleo pode se referir a um irmão à esquerda**: A Figura 3 ilustra esse tipo de acessibilidade. Nessa figura, as duas unidades discursivas RST estão ligadas por uma relação de Concessão, sendo a unidade discursiva 1 o núcleo.

[3] www.folha.uol.com.br

[4] Essas anotações são estruturadas de acordo com o modelo especificado em [10].

```
   Entrada: node = Nó da árvore RST
   Saída: node
 1 início
 2     se node é raiz da árvore então
 3         node.vein ← node.head;
 4     senão
 5         se node é Núcleo do relacionamento RST então
 6             Seja sibling_node = node.parent.left_child;
 7             se (sibling_node ≠ node) ∧ ¬ (sibling_node é Núcleo do relacionamento) então
 8                 node.vein ← node.vein + mark(sibling_node.head);
 9             fim
10             para en ∈ node.parent.vein faça
11                 node.vein ← node.vein + en;
12             fim
13         senão
14             para en ∈ node.head faça
15                 node.vein ← node.vein + en;
16             fim
17             se node = node.parent.left_child então
18                 para en ∈ node.parent.vein faça
19                     node.vein ← node.vein + en;
20                 fim
21             senão
22                 node.vein ← node.vein + simpl(node.parent.vein);
23             fim
24         fim
25     fim
26     se ¬ (node é terminal) então
27         para child ∈ node.children faça
28             Identificar veias em child;
29         fim
30     fim
31     retorna node
32 fim
```

Algoritmo 2: Algoritmo para identificação das veias.

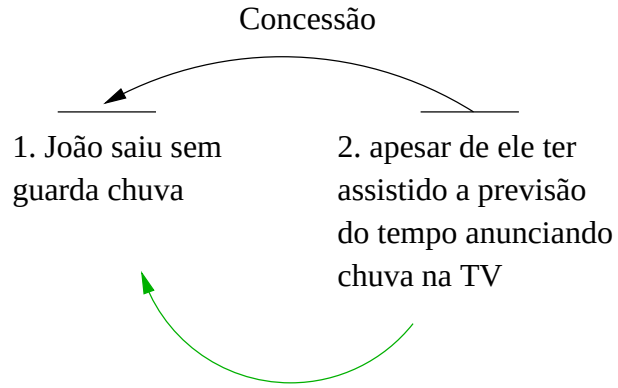

Figura 3: Exemplo de acessibilidade pela Teoria das Veias.

2. **Um núcleo pode se referir a um satélite à esquerda**: A Figura 4 também mostra um exemplo de relação de Concessão, onde esse tipo de acessibilidade ocorre. Entretanto, diferentemente da Figura 3, o núcleo está à direita do satélite.

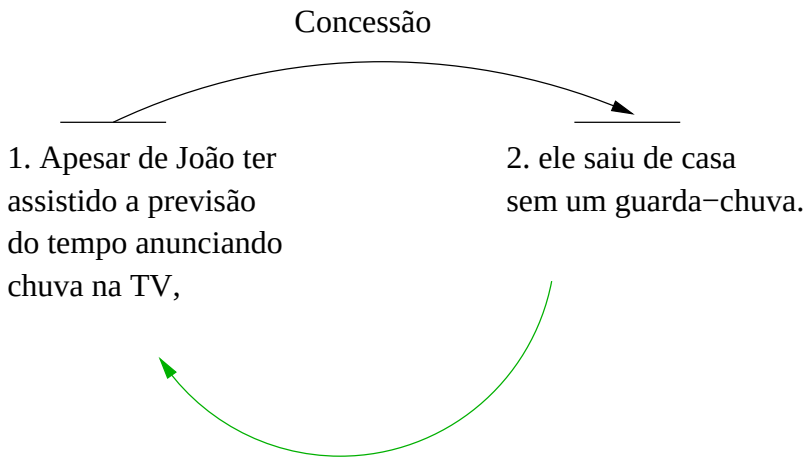

Figura 4: Exemplo de acessibilidade de um nó núcleo na árvore RST.

3. **Um satélite à direita de um núcleo x não pode ser acessado por um irmão à sua direita, seja ele núcleo ou satélite**: Na Figura 5, o pronome "Ela", na unidade discursiva 4, a princípio possui duas candidatas a referente, "Maria" e "sua mãe". Entretanto, de acordo com este princípio, as unidades discursivas 3 e 4 são adjacentes e, portanto, "sua mãe" não está acessível, sendo descartada como referente.

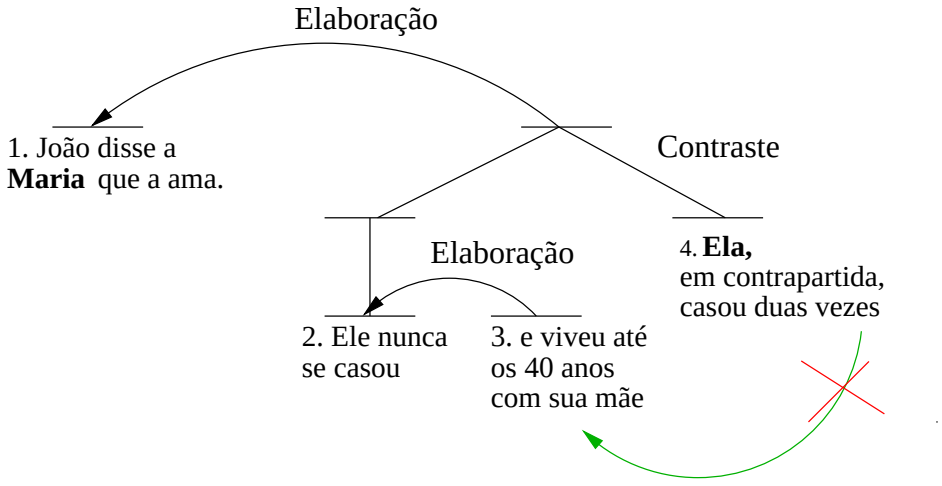

Figura 5: Exemplo de restrição de acessibilidade.

3 Implementação dos Algoritmos

Os algoritmos implementados neste trabalho se baseiam no conceito de restrição de acessibilidade, sendo que a estratégia adotada consiste em definir métodos para escolher o referente de um pronome dentre as entidades situadas nas veias. O método de escolha do referente

é inspirado nos métodos de ordenação utilizados pelos algoritmos baseados em Teoria da Centralização: BFP (Brennan, Friedman e Pollard) [3], *S-List* [18], e LRC (*Left-Right Centering*) [21], conforme apresentado a seguir. Porém, antes de apresentar os algoritmos, são discutidos os métodos para verificação de concordância e restrições de ligação, utilizados por todos os algoritmos.

3.1 Verificação de Concordância de Gênero e Número

A verificação de concordância entre um pronome e um candidato a referente é uma operação comum a todos os algoritmos implementados neste trabalho. Pronome e sintagma nominal devem concordar em gênero e número. Essas verificações são feitas como descrito a seguir.

- **Concordância de número**: O número do pronome e do candidato a referente são comparados;
- **Concordância de gênero**: Verifica-se se tanto pronome quanto candidato a referente são do mesmo gênero. Se o candidato a referente possuir forma comum para ambos os gêneros, como por exemplo "cliente", "cientista", "artista", ou for um nome próprio – pois os nomes próprios não tem seu gênero identificado no corpus utilizado neste trabalho – então assume-se que há concordância entre pronome e candidato a referente.

A fim de tratar anáforas conceituais, a verificação também leva em consideração a concordância com substantivos coletivos. Nesse caso, a concordância em gênero e número é tratada da seguinte forma:

- **Concordância de número**: Se o pronome estiver no plural e o candidato a referente for um substantivo coletivo, então considera-se que há concordância;
- **Concordância de gênero**: Se o candidato a referente for um substantivo coletivo e o pronome estiver no plural, então considera-se o gênero do elemento cujo conjunto é representado por esse substantivo, como por exemplo, em "O **time** joga a final do campeonato neste domingo. **Elas** estão concentradas e focadas na partida". Como o substantivo coletivo "time" pode se tratar de um grupo de atletas masculino ou feminino, o pronome "Elas" pode se referir às integrantes desse coletivo; logo, considera-se que há concordância.

3.2 Restrições de Ligação

Em alguns casos, a ligação entre um pronome e seu referente pode ser restrita por certas regras sintáticas, como em: "**João** acha que Felipe vai ajudá-**lo**". Nesse exemplo, apesar de ser evidente que o pronome "lo" se refere a "João", ele ainda concorda em gênero e número com "Felipe", o que pode dificultar a tarefa de resolução pronominal. Entretanto, a Teoria de Ligação de Chomsky ([17] *apud* [5]) define princípios que indicam que a referência a "Felipe" não seria correta.

No caso de pronomes pessoais – alvo de nosso estudo – a restrição de ligação definida pelo princípio B da Teoria da Ligação de Chomsky [17] diz que um pronome deve estar livre

em sua categoria de ligação, ou seja, dados pronome e candidato a referente em uma mesma oração, o pronome não deve c-comandar esse candidato. Uma entidade A c-comanda uma outra entidade B se, e somente se:

- A não domina[5] B e B não domina A na árvore sintática;
- O primeiro nó da árvore sintática que se ramifica e que domina A também domina B.

Em outras palavras, o primeiro nó da árvore sintática que domina o pronome também deve dominar seu candidato a referente, mas um não deve dominar o outro. O conceito de restrição de ligação também permite identificar se pronome e referente são contra-indexados em uma sentença. Isso ocorre quando eles não podem se referir à mesma entidade por não respeitarem o princípio B da Teoria da Ligação de Chomsky.

A verificação de restrições de ligação é utilizada pelos diversos algoritmos implementados neste trabalho. Na implementação descrita pelo Algoritmo 3, dadas duas entidades realizadas na mesma oração (sintagma nominal e pronome, dois pronomes, ou dois sintagmas nominais), considera-se que as restrições de ligação são respeitadas se uma entidade não c-comanda a outra ([17] *apud* [5]). Esse processo é descrito pelo Algoritmo 4.

```
   Entrada: a = entidade.
   Entrada: b = entidade.
   Saída: Indicador se a referência entre entidades a e b respeita (Verdadeiro) ou não (Falso) as restrições de
          ligação.
 1 início
 2 |   oracao_a = obtêm oração que contém a entidade a;
 3 |   oracao_b = obtêm oração que contém a entidade b;
 4 |   se oracao_a ≠ oracao_b então
 5 |   |   retorna Verdadeiro
 6 |   fim
 7 |   senão
 8 |   |   retorna ¬ (a c-comanda b)
 9 |   fim
10 fim
```

Algoritmo 3: Algoritmo para verificar as restrições de ligação.

```
   Entrada: nod = nó na árvore sintática.
   Saída: nó da oração que contém a entidade contida em nod.
 1 início
 2 |   se nod é uma oração então
 3 |   |   retorna nod
 4 |   fim
 5 |   senão
 6 |   |   branch_nod = obtêm oração que contém a entidade coberta por nod;
 7 |   |   retorna branch_nod
 8 |   fim
 9 fim
```

Algoritmo 4: Algoritmo para obter a oração à qual uma entidade pertence.

[5] A domina B se existe um caminho de A, até as folhas, através de B.

A complexidade computacional do Algoritmo 3, que verifica as restrições de ligação, é dominada pelas etapas de comparação das orações onde as entidades se encontram (linhas 2 a 4) e verificação de c-comando (linha 8) – sendo a verificação de c-comando descrita pelos Algoritmos 5 e 6. Como ambas as etapas percorrem a árvore de derivação, que possui $O(\mathrm{N})$ folhas – onde N é o número de entidades no discurso, então a complexidade dessas etapas, no pior caso e, portanto, desse algoritmo, é $O(\mathrm{N})$.

```
   Entrada: a = entidade.
   Entrada: b = entidade.
   Saída: Indicador se a entidade a c-comanda a entidade b (Verdadeiro) ou não (Falso).
 1 início
 2 |  retorna (¬ a domina b) ∧ (¬ b domina a) ∧ a.parent domina b
 3 fim
```

Algoritmo 5: Algoritmo para verificação de c-comando.

```
   Entrada: nod_a = entidade.
   Entrada: nod_b = entidade.
   Saída: Indicador se a entidade nod_a domina a entidade nod_b (Verdadeiro) ou não (Falso).
 1 início
 2 |  se nod_b.parent = NULL então
 3 |  |  retorna Falso
 4 |  fim
 5 |  senão se nod_b = nod_a então
 6 |  |  retorna Verdadeiro
 7 |  senão
 8 |  |  retorna (nod_a domina nod_b.parent)
 9 |  fim
10 fim
```

Algoritmo 6: Algoritmo para verificação de domínio.

3.3 Algoritmo VT-BFP

O algoritmo VT-BFP leva esse nome em razão de ordenar as entidades contidas nas veias através de método semelhante ao usado pelo algoritmo BFP para ordenar os centros de um determinado enunciado. Dessa forma, o algoritmo primeiramente identifica as veias na árvore RST e, em seguida, para cada pronome, ordena as entidades contidas na veia do nó em que ele se encontra, de acordo com dois critérios: 1) a posição do enunciado onde essa entidade se encontra dentro do discurso; e 2) a função do sintagma nominal, segundo a ordem sujeito > objeto direto > objeto indireto > outros > adjunto. Finalmente, o algoritmo simplesmente procura por um referente nessa mesma veia, escolhendo a primeira entidade encontrada que concorde em gênero e número com o pronome. Esse processo é descrito pelo Algoritmo 7, e se repete para cada enunciado do discurso. Para isso, considera-se que as veias de todas as unidades discursivas RST já foram identificadas previamente, pelo Algoritmo 2.

O exemplo apresentado na Figura 6[6] ilustra o funcionamento desse algoritmo. Nele podemos ver as unidades discursivas RST apresentadas na árvore RST da Figura 7 – que se trata de uma sub-árvore de um discurso maior – e também os elementos contidos na veia da

[6]Retirado do texto CIENCIA_2000_17082 do corpus Sum-it (descrito na seção 5.1 deste trabalho).

unidade discursiva 2, onde se encontra o pronome "ele". Percebe-se que nessa veia há tanto entidades mencionadas na própria unidade discursiva 2, quanto entidades mencionadas na unidade discursiva 1 (por se tratar de um irmão núcleo à esquerda). Também há entidades mencionadas nas unidades 4 e 5, além de entidades mencionadas previamente no discurso, em ramos da árvore RST que não são apresentados na sub-árvore do exemplo[7]. A veia apresentada no exemplo foi então ordenada de acordo com os critérios do algoritmo VT-BFP e, ao buscar pelo referente do pronome "ele" da unidade discursiva 2, escolhe-se "Adalberto Veríssimo" como referente do pronome "Ele", uma vez que "Adalberto Veríssimo" é o elemento mais bem classificado na veia por exercer a função de sujeito.

1. Adalberto Veríssimo de a ONG Imazon apresentou estudo segundo o qual
as cidades em regiões amazônicas ocupadas de forma predatória duram por volta de 23 anos
2. ele citou como exemplo as cidades de Paragominas

veia = {ele=Adalberto Veríssimo,cidades,paragominas,humaitá,açailândia,forma,
regiões,cidades,Adalberto Veríssimo,anos,estudo,ONG Imazon,queda,tendência,
gás carbônico,efeito estufa,causador,fenômenos,combinados,desertificação,áreas}

3. (PA)
4. Açailândia
5. (MA)
6. e Humaitá
7. (AM)

Figura 6: Exemplo de execução do Algoritmo VT-BFP.

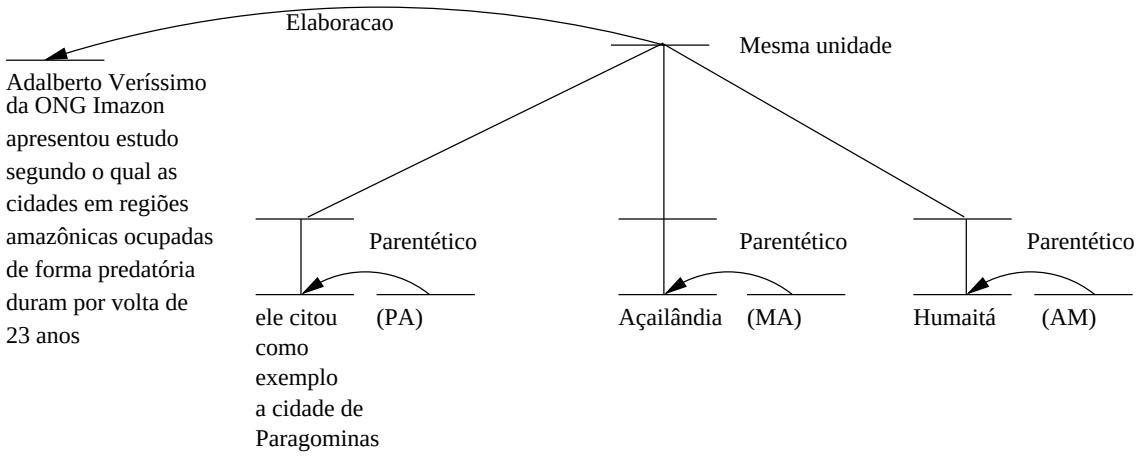

Figura 7: Exemplo de árvore RST utilizada pelo algoritmo VT-BFP.

[7]O restante da árvore RST do texto exemplo foi omitido por questão de simplificação.

```
   Entrada: U_n = Conjunto de entidades no enunciado atual
   Entrada: RST = Nó raíz da árvore RST
   Saída: U_n
 1 início
 2     para en ∈ U_n faça
 3         se en é um pronome então
 4             Seja en.R uma unidade discursiva RST sendo en ∈ R;
 5             Seja en.R.vein o conjunto das entidades contidas na veia de R;
 6             Ordenar en.R.vein;
 7             para ref ∈ en.R.vein faça
 8                 se ref concorda com en ∧ ¬ há restrição de ligação entre ref e en então
 9                     en.ref ← ref;
10                     Encerra loop;
11                 fim
12             fim
13         fim
14     fim
15     retorna U_n
16 fim
```

Algoritmo 7: Algoritmo VT-BFP.

Ao contrário do BFP original, o VT-BFP é capaz de encontrar referentes no mesmo enunciado do discurso do pronome, uma vez que esses referentes ainda estão presentes na veia da unidade discursiva RST onde o pronome se encontra. Como o algoritmo VT-BFP procura por toda a veia da unidade discursiva RST em que o pronome se encontra – diferentemente do algoritmo BFP, que só procura no centro do enunciado anterior – o algoritmo VT-BFP pode escolher um referente que se encontra no mesmo enunciado do pronome, o que lhe permite resolver anáforas intra-sentenciais, mesmo que os enunciados do discurso sejam representados por sentenças, o que não ocorre com o BFP. Entretanto, o algoritmo VT-BFP não utiliza as transições de centro definidas pelo BFP [3], mas se utiliza da mesma lógica usada pelo BFP para ordenar seus enunciados, porém o VT-BFP a aplica para a ordenação das veias.

3.4 Algoritmo VT-SL

Inspirado no algoritmo S-List [18], o algoritmo VT-SL ordena as entidades contidas nas veias de forma semelhante à utilizada pelo algoritmo S-List para ordenar sua estrutura de dados principal (a S-List propriamente dita). Dessa forma, o algoritmo VT-SL ordena as veias conforme sua:

1. **Posição do enunciado no discurso**: Quanto mais próxima do final do discurso a entidade se encontrar, melhor classificada neste quesito;

2. **Estado da informação (*Information status*)** – O estado da informação de uma entidade indica se ela foi recém introduzida no contexto, ou se ela já foi mencionada anteriormente. Esse conceito tem origem na Escala de Familiaridade de Prince [16], e em sua extensão proposta em [19]. Segundo esse critério, as classes são agrupadas como *OLD* (entidades antigas), *MED* (entidades intermediárias) ou *NEW* (entidades novas), e a ordem de preferência na resolução de pronomes é a seguinte: entidades

pertencentes a classes do grupo *OLD* precedem as entidades pertencentes a classes do grupo *MED*, que por sua vez precedem as do grupo *NEW* (ou seja, *OLD* > *MED* > *NEW*). Segue abaixo uma breve descrição desses grupos e das classes que os compõem [18, 19]:

- ***OLD***: Composto por entidades classificadas como *EVOKED* (evocadas, referenciadas) ou *UNUSED* (não utilizadas). As entidades *EVOKED* são aquelas já mencionadas no texto e que são referenciadas por pronomes. Por sua vez, entidades *UNUSED* são realizadas somente por substantivos próprios;
- ***NEW***: Somente a classe *BRAND NEW* (nova) pertence a este grupo. As entidades que são mencionadas pela primeira vez no texto são classificadas como *BRAND NEW*, podendo ser identificadas por serem acompanhadas de artigos indefinidos.
- ***MED***: Introduzido em [19], este grupo contém classes que, segundo a Escala de Familiaridade de Prince [16], pertenciam ao grupo *NEW*, mas foram separadas neste novo grupo com o intuito de tratar referências anafóricas em textos com pouca incidência de pronomes. Este grupo é formado por entidades classificadas como *INFERRABLE* (inferível), *CONTAINING INFERRABLE* (recipiente de entidade inferível) e *ANCHORED BRAND NEW* (nova ancorada). Uma entidade classificada como *INFERRABLE* possui ligação com outra entidade do discurso, mas de forma indireta e não anafórica. Essa ligação pode ser inferível através de informações do contexto. Já as entidades classificadas como *ANCHORED BRAND NEW* são relacionadas (ancoradas) ao contexto por entidades *OLD*. Assume-se que a diferença entre *CONTAINING INFERRABLE* e *ANCHORED BRAND NEW* é irrelevante, sendo ambas consideradas equivalentes na prática.

3. **Posição da entidade dentro do enunciado**: Quanto mais próxima do início do enunciado, mais bem colocada neste quesito.

O Algoritmo 8 descreve o mecanismo de ordenação de entidades em uma veia, utilizado pelo algoritmo VT-SL. Antes de ordenar a veia, esse algoritmo define o estado da informação, como descrito anteriormente. Nesta implementação, todo pronome é marcado como EVOKED (linhas 10 e 11). Isso ocorre porque qualquer pronome está de fato realizando alguma entidade mencionada anteriormente no discurso – ou posteriormente, no caso de catáforas[8]. Substantivos precedidos por artigos indefinidos são marcados como BRAND NEW (linhas 12 e 13), enquanto substantivos próprios não precedidos por determinantes são marcados como UNUSED (linhas 14 e 15). Por fim, entidades que não atendem à nenhuma dessas condições são marcadas como ANCHORED BRAND NEW. Após definir o estado da informação de todas as entidades contidas na veia (linhas 2 a 31), o algoritmo definitivamente ordena a veia considerando três critérios: posição do enunciado no discurso, estado da informação e posição da entidade no enunciado (linha 32).

[8]Todavia, catáforas não são tratadas neste trabalho.

```
   Entrada: vein = Veia a ser ordenada
   Saída: vein = Veia ordenada
 1 início
 2     para w ∈ vein faça
 3         se ¬ (w é sintagma nominal) então
 4             se ¬ (w é adjetivo) então
 5                 ant ← w;
 6             fim
 7             Segue para a próxima iteração;
 8         fim
 9         en ← w;
10         se en é um pronome então
11             is ← EVOKED;
12         senão se ant é um artigo indefinido então
13             is ← BRAND_NEW;
14         senão se ¬ (ant é um determinante) ∧ (en é um substantivo próprio) então
15             is ← UNUSED;
16         senão
17             is ← ANCHORED_BRAND_NEW;
18         fim
19         se is ∈ {EVOKED, UNUSED} então
20             Marcar en como OLD;
21             en.info_status ← 1;
22         senão se is ∈ {ANCHORED_BRAND_NEW} então
23             Marcar en como MED;
24             en.info_status ← 2;
25         senão se is ∈ {BRAND_NEW} então
26             Marcar en como NEW;
27             en.info_status ← 3;
28         fim
29         Seja en.disc_index o índice do enunciado do discurso;
30         Seja en.utt_index o índice de en no enunciado do discurso;
31     fim
32     Ordenar vein por disc_index, info_status, utt_index;
33     retorna vein
34 fim
```

Algoritmo 8: Algoritmo utilizado por VT-SL para ordenação das veias.

Dessa forma, o algoritmo VT-SL – descrito pelo Algoritmo 9 – torna-se semelhante ao VT-BFP, ou seja, após ordenar as entidades contidas na veia (linha 6), o algoritmo busca pela primeira entidade encontrada que concorde em gênero e número com o pronome e que respeite as restrições de ligação de Chomsky [17] (linhas 7 a 12), repetindo esse procedimento para cada pronome do enunciado (linhas 2 a 14).

```
   Entrada: U_n = Conjunto de entidades no enunciado atual
   Entrada: RST = Nó raíz da árvore RST
   Saída: U_n
 1 início
 2     para en ∈ U_n faça
 3         se en é um pronome então
 4             Seja en.R uma unidade discursiva RST sendo en ∈ R;
 5             Seja en.R.vein o conjunto das entidades contidas na veia de R;
 6             Ordenar en.R.vein;
 7             para ref ∈ en.R.vein faça
 8                 se ref concorda com en ∧ ¬ há restrição de ligação entre ref e en então
 9                     en.ref ← ref;
10                     Encerra loop;
11                 fim
12             fim
13         fim
14     fim
15     retorna U_n
16 fim
```

Algoritmo 9: Algoritmo VT-SL.

Apesar da alusão ao algoritmo S-List, o algoritmo VT-SL não mantém uma estrutura única, como a S-List. A principal semelhança está no fato desse algoritmo usar os mesmos critérios de ordenação para as veias, porém com pesos diferentes. A marcação de entidades EVOKED também é diferente no VT-SL, onde todos os pronomes são assim marcados, possibilitando o mesmo resultado obtido pelo S-List, por contar com referência evocativa [6], uma vez que uma referência a um pronome também referencia indiretamente a entidade por ele referenciada.

3.5 Algoritmo VT-LRC

Inspirado no algoritmo LRC (*Left-Right Centering*), o algoritmo VT-LRC (Algoritmo 10) utiliza o mesmo mecanismo de ordenação de entidades na veia que o algoritmo VT-BFP. Entretanto, quando ele encontra um pronome, ele primeiro procura pelo referente entre as entidades realizadas previamente no mesmo enunciado do discurso, da esquerda para a direita, buscando por entidades em enunciados anteriores somente se não encontrar nenhum candidato que concorde com o pronome em gênero e número e respeite as restrições de ligação de Chomsky [17].

4 Complexidade dos Algoritmos

A análise da complexidade é feita em relação à operação de encontrar os referentes para todos os pronomes do discurso. Como todos os algoritmos consistem em identificar as veias,

```
   Entrada: U_n = Conjunto de entidades no enunciado atual
   Entrada: RST = Nó raíz da árvore RST
   Saída: U_n
 1 início
 2 |   para en ∈ U_n faça
 3 |   |   se en é um pronome então
 4 |   |   |   para ref ∈ U_n faça
 5 |   |   |   |   se ref concorda com en ∧ ¬ há restrição de ligação entre ref e en então
 6 |   |   |   |   |   en.ref ← ref;
 7 |   |   |   |   |   Segue para a próxima iteração;
 8 |   |   |   |   fim
 9 |   |   |   fim
10 |   |   |   Seja en.R uma unidade discursiva RST sendo en ∈ R;
11 |   |   |   Seja en.R.vein o conjunto das entidades contidas na veia de R;
12 |   |   |   Ordenar en.R.vein;
13 |   |   |   para ref ∈ en.R.vein faça
14 |   |   |   |   se ref concorda com en ∧ ¬ há restrição de ligação entre ref e en então
15 |   |   |   |   |   en.ref ← ref;
16 |   |   |   |   |   Encerra loop;
17 |   |   |   |   fim
18 |   |   |   fim
19 |   |   fim
20 |   fim
21 |   retorna U_n
22 fim
```

Algoritmo 10: Algoritmo VT-LRC.

ordenar as entidades contidas nelas e buscar por referentes nesse domínio, eles possuem a mesma complexidade.

A identificação das veias é necessária antes da primeira execução de qualquer algoritmo. Como já mostrado, essa etapa leva o tempo O(S^2), onde S representa o número de nós na árvore RST. Já a ordenação das entidades contidas na veia de um nó da árvore RST leva o tempo O($N\ log\ N$), sendo N o número de entidades realizadas em todo o discurso, uma vez que também há O(N) entidades na veia. Como essa tarefa é repetida para as veias de todas as S unidades discursivas RST, essa etapa leva o tempo O($S(N\ log\ N)$). Adicionando-se o tempo da identificação das veias inicial, temos O(S^2 + $S(N\ log\ N)$).

Já a etapa de busca pelo referente correto de um dado pronome é feita pela veia do nó da árvore RST em que ele se encontra. Como há O(N) entidades nessa veia, considerando também que há O(N) pronomes no discurso, e que se leva tempo O(N) para verificar as restrições de ligação entre o pronome e cada um de seus candidatos a referente, então essa etapa leva o tempo O(N^3) e, portanto, a complexidade dos algoritmos é O(S^2 + $S(N\ log\ N)$ + N^3). Assumindo que $S <= N$ neste trabalho, uma vez que não analisamos textos sem entidades, então a complexidade de todos os algoritmos baseados em Teoria das Veias aqui implementados é sempre dominada pela tarefa de busca pelo referente, sendo portanto O(N^3).

5 Comparação e Avaliação dos Algoritmos

A Tabela 1 mostra uma comparação entre os algoritmos aqui implementados e os algoritmos baseados em Teoria da Centralização. Nessa tabela, a coluna "Referentes distantes" indica se o algoritmo é capaz de encontrar referentes a mais de um enunciado de distância, enquanto a coluna "Intra-sentencial" indica se o algoritmo é capaz de encontrar referentes no mesmo enunciado do discurso. Como pode ser visto, os algoritmos baseados em Teoria das Veias possuem complexidade assintótica semelhante à maioria dos algoritmos baseados em Teoria da Centralização, isto é, $O(N^3)$[9]. Entretanto, a maior vantagem dos algoritmos baseados em Teoria das Veias está na capacidade de encontrar referentes tanto intra-sentenciais quanto a mais de um enunciado de distância.

Tabela 1: Comparação entre as características dos algoritmos.

Algoritmo	Referentes distantes	Intra-sentencial	Complexidade
Conceitual	Não	Não	$O(N^3)$
BFP	Não	Não	$O(N^6)$
LRC	Não	Sim	$O(N^3)$
S-List	Não	Sim	$O(N^3)$
VT-BFP	Sim	Sim	$O(N^3)$
VT-SL	Sim	Sim	$O(N^3)$
VT-LRC	Sim	Sim	$O(N^3)$

5.1 Avaliação dos Algoritmos

Para teste e avaliação dos algoritmos aqui descritos foi utilizado o corpus Sum-it, sendo que as anotações desse corpus estão divididas em diversos arquivos que acompanham cada texto do corpus. Os arquivos utilizados neste trabalho são os seguintes:

- **Conteúdo do Texto**: O conteúdo do texto jornalístico de forma inalterada e plana (sem anotações) é armazenado em um arquivo texto comum;

- **Lista de Palavras**: As palavras do texto são separadas em um arquivo em formato XML, e definidas por *tags* do tipo "*word*". Cada *tag* no arquivo possui um atributo "id", que o identifica de forma única no texto, sendo seu conteúdo a palavra propriamente dita. Para acomodar as anotações de referentes corretos das anáforas, foi necessária a edição desses arquivos, de modo a acrescentar o atributo "*ref*" à *tag* "*word*" de cada pronome, para que contivesse o "id" de seu referente, ou dos vários referentes considerados corretos. A Figura 8 exemplifica o conteúdo desses arquivos. Nesse exemplo, pode-se ver que a anáfora "Ele" – representada pela palavra "word_182" – refere-se tanto a "presidente" (palavra "word_160"), quanto a "Alberto_Portugal" (nome do presidente, na palavra "word_166"). Assim, se um algoritmo escolher qualquer uma dessas duas palavras como referente, considera-se que ele obteve sucesso.

```
<word id="word_159">O</word>
<word id="word_160">presidente</word>
<word id="word_161">de</word>
<word id="word_162">a</word>
<word id="word_163">Embrapa</word>
<word id="word_164">(Empresa_Brasileira_de_Pesquisa_Agropecuária)</word>
<word id="word_165">,</word>
<word id="word_166">Alberto_Portugal</word>
<word id="word_167">,</word>
<word id="word_168">salientou</word>
<word id="word_169">que</word>
<word id="word_170">a</word>
<word id="word_171">empresa</word>
<word id="word_172">busca</word>
<word id="word_173">soluções</word>
<word id="word_174">para</word>
<word id="word_175">os</word>
<word id="word_176">problemas</word>
<word id="word_177">de</word>
<word id="word_178">a</word>
<word id="word_179">agricultura</word>
<word id="word_180">nacional</word>
<word id="word_181">.</word>
<word id="word_182"ref="word_160,word_166">Ele</word>
<word id="word_183">citou</word>
<word id="word_184">o</word>
<word id="word_185">exemplo</word>
<word id="word_186">de</word>
<word id="word_187">pesquisas</word>
```

Figura 8: Lista de palavras, retirada do corpus Sum-it.

- **Anotações Gramaticais**: A classificação gramatical de cada palavra é armazenada em um arquivo separado. Esse arquivo também possui *tags* do tipo *word*, com os mesmos identificadores da lista de palavras, mas com a adição de uma *tag* interna à primeira, que representa sua classe gramatical – essa *tag* recebe o mesmo nome da marcação utilizada pelo parser PALAVRAS. A Figura 9 apresenta um exemplo de um trecho desse arquivo[9].

```
<word id="word_1">
<art canon="o"gender="F"number="S">
<secondary_art tag="artd"/>
</art>
</word>
<word id="word_2">
<n canon="discussão"gender="F"number="S"/>
</word>
<word id="word_3">
<prp canon="sobre"/>
</word>
<word id="word_4">
<art canon="o"gender="F"number="S">
<secondary_art tag="artd"/>
</art>
</word>
<word id="word_5">
<n canon="biotecnologia"gender="F"number="S"/>
</word>
<word id="word_6">
<adj canon="nacional"gender="M-F"number="S"/>
</word>
```

Figura 9: Anotações gramaticais retiradas do corpus Sum-it.

- **Árvore Sintática**: Cada texto do corpus também é acompanhado de um arquivo de *chunks*, responsável por especificar a árvore sintática de cada sentença do texto. Nesse contexto, uma *tag chunk* representa um nó da árvore. Dentro dela, o atributo "ext" indica sua função sintática, como "sta" para afirmação, "subj" para sujeito ou "n" para nomes, entre outros, enquanto que o atributo "form" representa a forma sintática, como "fcl" para oração finita, "np" para sintagma nominal, ou "art" para artigo definido. Por sua vez, o atributo "span" indica o intervalo de palavras que compõem esse nó. A Figura 10 mostra um trecho de um desses arquivos.

- **Relacionamentos Retóricos**: O arquivo de anotações de relacionamentos retóricos

[9]Nesse exemplo, as *tags* utilizadas são: *art* (artigo); *secondary_art* (tipo do artigo, sendo *artd* para definido e *arti* para indefinido; *n* para substantivo; *prp* para preposição; e *adj* para adjetivo.

```
<text>
<paragraph id="paragraph_1">
<sentence id="sentence_1"span="word_1..word_18">
<chunk id="chunk_1"ext="sta"form="fcl"span="word_1..word_17">
<chunk id="chunk_2"ext="subj"form="np"span="word_1..word_6">
<chunk id="chunk_3"ext="n"form="art"span="word_1">
```

Figura 10: Árvore sintática retirada do corpus Sum-it.

contém o texto dividido em unidades discursivas RST, além de indicadores do relacionamento entre elas, formando a árvore RST. Essas unidades discursivas são delimitadas por *tags* do tipo *segment*, que por sua vez também contêm os atributos *parent* – indicando a unidade discursiva RST (ou o nó da árvore) que possui função de núcleo do relacionamento – e *relname*, que é um indicador do tipo de relacionamento retórico, sendo que os tipos de relacionamentos retóricos utilizados para a anotação desse corpus foram os mesmos definidos em [15], tratando-se de 32 tipos diferentes de relações. Os demais nós da árvore RST são representados por *tags* do tipo *group*. A Figura 11, retirada de [4] – já apresentada na Seção 2 e repetida aqui por conveniência – representa uma árvore RST do corpus Sum-it, definida conforme a Figura 12.

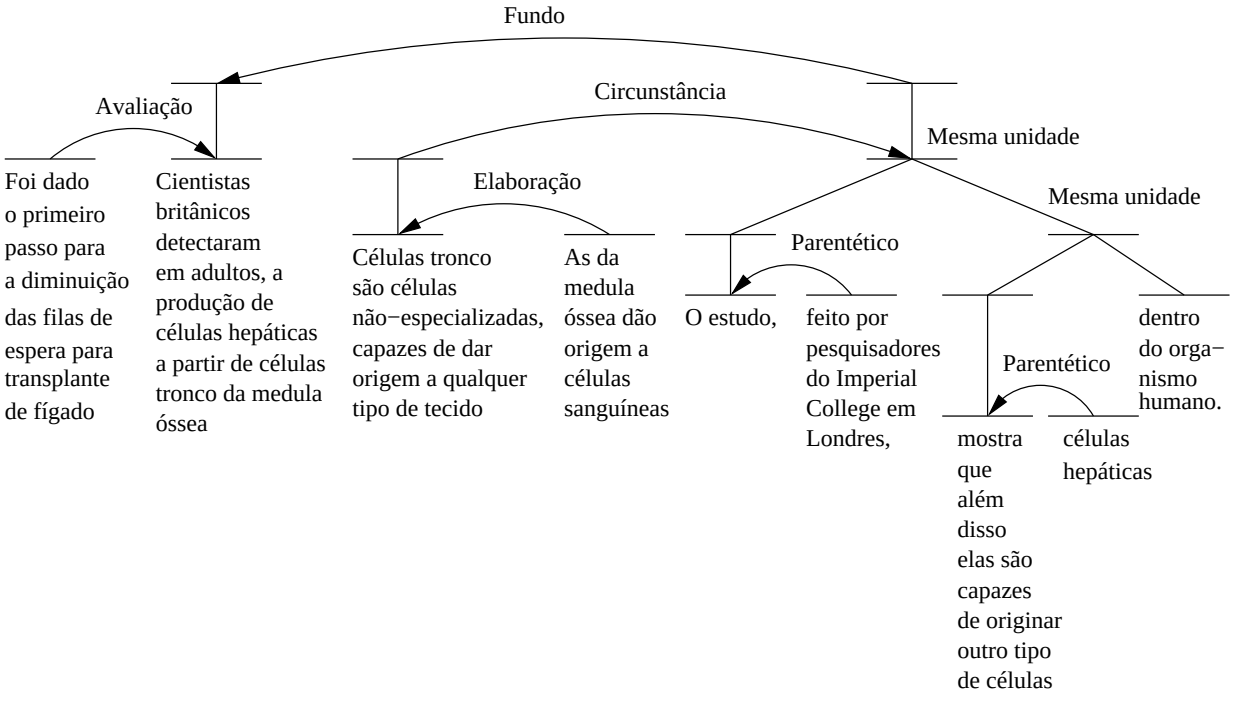

Figura 11: Árvore RST representando a estrutura do discurso.

```
<segment id="1"parent="2"relname="evaluation">Foi dado o primeiro passo para
a diminuição das filas de espera para transplante de fígado.</segment>
<segment id="2"parent="40"relname="span"> Cientistas britânicos detectaram,
em adultos, a produção de células hepáticas a partir de células-tronco da medula
óssea.</segment>
<segment id="3"parent="34"relname="span">
Células-tronco são células não-especializadas, capazes de dar origem a qualquer
tipo de tecido.</segment>
<segment id="4"parent="3"relname="elaboration"> As da medula óssea dão origem
a células sanguíneas.</segment>
<segment id="5"parent="26"relname="span"> O estudo,</segment>
<segment id="6"parent="5"relname="parenthetical"> feito por pesquisadores
do Imperial College, em Londres,</segment>
<segment id="7"parent="29"relname="span"> mostra que, além disso, elas são
capazes de originar outro tipo de célula</segment>
<segment id="8"parent="7"relname="parenthetical"> _células hepáticas_
</segment>
<segment id="9"parent="28"relname="same-unit"> dentro do organismo humano.
</segment>

<group id="25"type="multinuc"parent="41"relname="span"/>
<group id="26"type="span"parent="25"relname="same-unit"/>
<group id="28"type="multinuc"parent="25"relname="same-unit"/>
<group id="29"type="span"parent="28"relname="same-unit"/>
<group id="34"type="span"parent="25"relname="circumstance"/>
<group id="40"type="span"parent="44"relname="span"/>
<group id="41"type="span"parent="40"relname="background"/>
<group id="44"type="span"parent="46"relname="span"/>
<group id="46"type="span"parent="47"relname="span"/>
<group id="47"type="span"/>
```

Figura 12: Anotações de relacionamentos retóricos, retiradas do corpus Sum-it.

Com o objetivo de reduzir o escopo deste trabalho, nesta avaliação somente foram considerados pronomes pessoais do caso reto e oblíquo átonos, na terceira pessoa do singular e plural, totalizando 129 pronomes em todo o corpus, sendo 41% deles anáforas intra-sentenciais. Esses pronomes tem, em média, 24 candidatos a referente que concordam em gênero e número e aparecem antes do pronome no texto, sendo que a média de candidatos presentes na mesma sentença ou na sentença anterior é de 3 entidades, que são geralmente acessíveis por métodos baseados em Teoria da Centralização.

5.2 Resultados

A avaliação dos algoritmos foi feita através de experimentos com dois tipos de segmentação de texto diferente: sentenças, delineadas pela árvore sintática do corpus, onde cada enunciado do discurso é representado por uma frase; e os segmentos de texto que compõem os relacionamentos retóricos definidos pela Teoria RST – aqui chamadas de unidades discursivas RST – onde cada unidade discursiva representa um enunciado do discurso.

Os resultados obtidos pela execução dos algoritmos no corpus segmentado por unidades discursivas RST são exibidos na Tabela 2. Nelas, as colunas "Intra. Result." e "Inter. Result." indicam o percentual de acerto para anáforas intra- e inter-sentenciais, respectivamente, enquanto a coluna "Sem Candidato" indica o percentual de casos em que os algoritmos não foram capazes sequer de indicar um candidato a referente, mesmo que incorreto. Por fim, a coluna "Resultado" indica o percentual de acerto total. A Tabela 3, por sua vez, mostra os resultados ao executar os algoritmos sobre o corpus segmentado em sentenças.

Tabela 2: Corpus segmentado em unidades discursivas RST.

Algoritmo	Intra. Result.	Inter. Result.	Sem Candidato	Resultado
Conceitual	49%	28%	33%	36%
BFP	47%	28%	35%	36%
LRC	57%	28%	29%	39%
S-List	62%	28%	29%	42%
VT-BFP	47%	33%	20%	39%
VT-LRC	49%	33%	20%	40%
VT-SL	52%	29%	20%	39%

Tabela 3: Corpus segmentado em sentenças.

Algoritmo	Intra. Result.	Inter. Result.	Sem Candidato	Resultado
Conceitual	0%	43%	19%	26%
BFP	0%	43%	26%	26%
LRC	66%	43%	8%	53%
S-List	62%	43%	8%	51%
VT-BFP	49%	31%	20%	39%
VT-LRC	66%	31%	15%	46%
VT-SL	57%	30%	20%	41%

Ao compararmos os algoritmos baseados em Teoria das Veias com seus semelhantes baseados em Teoria da Centralização, podemos observar que, ao utilizar a segmentação por unidades discursivas RST, os algoritmos VT-BFP, VT-LRC e VT-SL apresentam resultados superiores para a resolução de anáforas inter-sentenciais, muito embora essa diferença não seja estatisticamente significativa ($\chi^2(df = 2) = 0,1618, p = 0,9223$). Os algoritmos BFP, LRC e *S-List*, por outro lado, levam vantagem em relação a anáforas intra-sentenciais, ainda que também de maneira não estatisticamente significativa ($\chi^2(df = 2) = 0,4506, p =$

$0,7983$, para o conjunto com verificação de restrições, e $\chi^2(df = 2) = 0,4741, p = 0,7890$ para o conjunto sem).

No experimento utilizando um corpus segmentado por sentenças, os algoritmos baseados em Teoria da Centralização tiveram um melhor desempenho, tanto para anáforas intra-sentenciais (com exceção do algoritmo BFP, que é incapaz de resolver anáforas nessas condições), quanto para anáforas inter-sentenciais. Nesse caso, ocorre uma diferença altamente significativa ($\chi^2(df = 2) = 43,6967, p < 0,001$, para o conjunto com verificação de restrições, e $\chi^2(df = 2) = 47,7368, p < 0,001$ para o conjunto sem) somente por conta do BFP que, ao contrário do VT-BFP, é incapaz de resolver anáforas intra-sentenciais nessas condições. Caso o algoritmo BFP seja removido, os resultados apontam para uma diferença não estatisticamente significativa.

Por fim, a maioria dos algoritmos apresentou alto índice de casos em que não foi possível sequer indicar um referente, mesmo que incorreto. Essa situação é amenizada apenas para os algoritmos LRC e *S-List* sobre um corpus segmentado em sentenças, fazendo com que a segmentação por sentenças demonstrasse uma diferença estatisticamente significativa ($\chi^2(df = 5) = 14,0405, p = 0,01535$, para o conjunto com verificação de restrição e $\chi^2(df = 5) = 13,0206, p = 0,02319$ para o conjunto sem) em relação à segmentação por unidades discursivas RST. Vale lembrar que nenhum algoritmo baseado em Teoria das Veias apresentou o melhor resultado em algum experimento.

5.3 Experimentos sem Verificação de Restrições de Ligação

Neste trabalho também foram conduzidos experimentos adicionais onde os algoritmos originais foram modificados de tal forma que as verificações de restrições de ligação não foram realizadas. Nesse caso, o algoritmo BFP não mais descarta conjuntos em que há dois pronomes contra-indexados ou um pronome e seu referente contra-indexados, assim como não há mais verificação de restrições de ligação durante a escolha de um referente para o pronome nos algoritmos LRC, *S-List*, VT-BFP, VT-LRC e VT-SL. Todavia, o Algoritmo Conceitual permanece inalterado, pois já não se utilizava desse tipo de verificação. O resultado obtido ao executar esses algoritmos modificados sobre o corpus segmentado por unidades discursivas RST é apresentado na Tabela 4, enquanto o resultado do experimento sobre o corpus segmentado por sentenças é mostrado na Tabela 5.

Tabela 4: Corpus segmentado em unidades discursivas RST, sem verificar restrições de ligação.

Algoritmo	Intra. Result.	Inter. Result.	Sem Candidato	Resultado
Conceitual	49%	28%	33%	36%
BFP	49%	28%	34%	36%
LRC	57%	28%	28%	39%
S-List	64%	28%	28%	43%
VT-BFP	49%	33%	19%	39%
VT-LRC	51%	33%	19%	40%
VT-SL	53%	29%	19%	39%

Tabela 5: Corpus segmentado em sentenças, sem restrições de ligação.

Algoritmo	Intra. Result.	Inter. Result.	Sem Candidato	Resultado
Conceitual	0%	43%	19%	26%
BFP	0%	43%	26%	26%
LRC	70%	43%	8%	54%
S-List	66%	43%	8%	53%
VT-BFP	53%	32%	19%	40%
VT-LRC	70%	32%	14%	47%
VT-SL	59%	30%	19%	42%

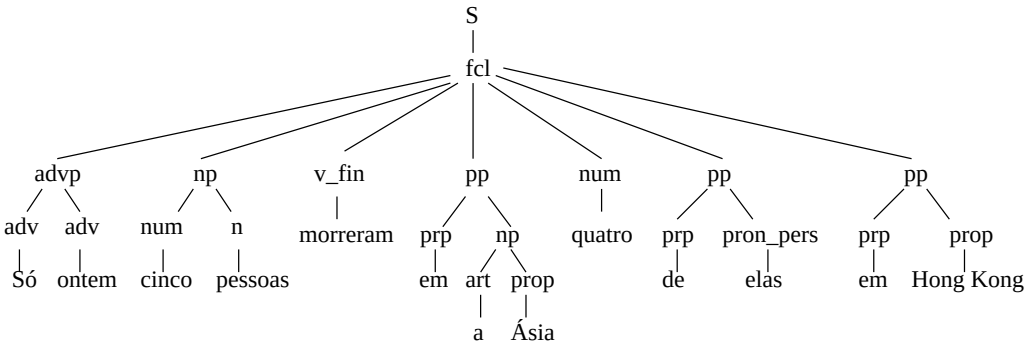

Figura 13: Árvore de derivação relativa à restrição de ligação encontrada em CIENCIA_2003_6457.

Ao compararmos com os resultados apresentados nos experimentos realizados com os algoritmos originais, podemos observar que os resultados dos experimentos com os algoritmos modificados para não verificar restrições de ligação são, em média, 1% superiores. Essa variação no resultado se deve ao fato de haver 2 casos em que as restrições de ligação impedem que seja feita a referência entre pronome e referente correto. No primeiro caso, encontrado no texto CIENCIA_2003_6457 e ilustrado pela árvore de derivação mostrada na Figura 13, pode-se ver que a referência entre o pronome "elas" e seu referente "pessoas" é impedida pelo princípio B da Teoria da Ligação de Chomsky, uma vez que o nó *np* ao qual "pessoas" pertence c-comanda o nó *pp* que contém o pronome "elas". O mesmo fenômeno também pode ser observado no segundo caso, ilustrado pela Figura 14, onde a referência entre o pronome "eles" e seu referente "cientistas" também é impedida, uma vez que o nó *np* c-comanda o nó *pron_pers* ao qual o pronome pertence.

Todavia, vale ressaltar que a diferença apontada nos resultados dos experimentos não é estatisticamente significativa; portanto, não podemos afirmar que a verificação de restrições de ligação é prejudicial. Porém, uma análise mais profunda deve ser realizada.

5.4 Comparação entre Teoria da Centralização e Teoria das Veias

Essa comparação é feita levando-se em conta os resultados apresentados no experimento que obteve melhores resultados utilizando o corpus segmentando por sentenças (Tabela

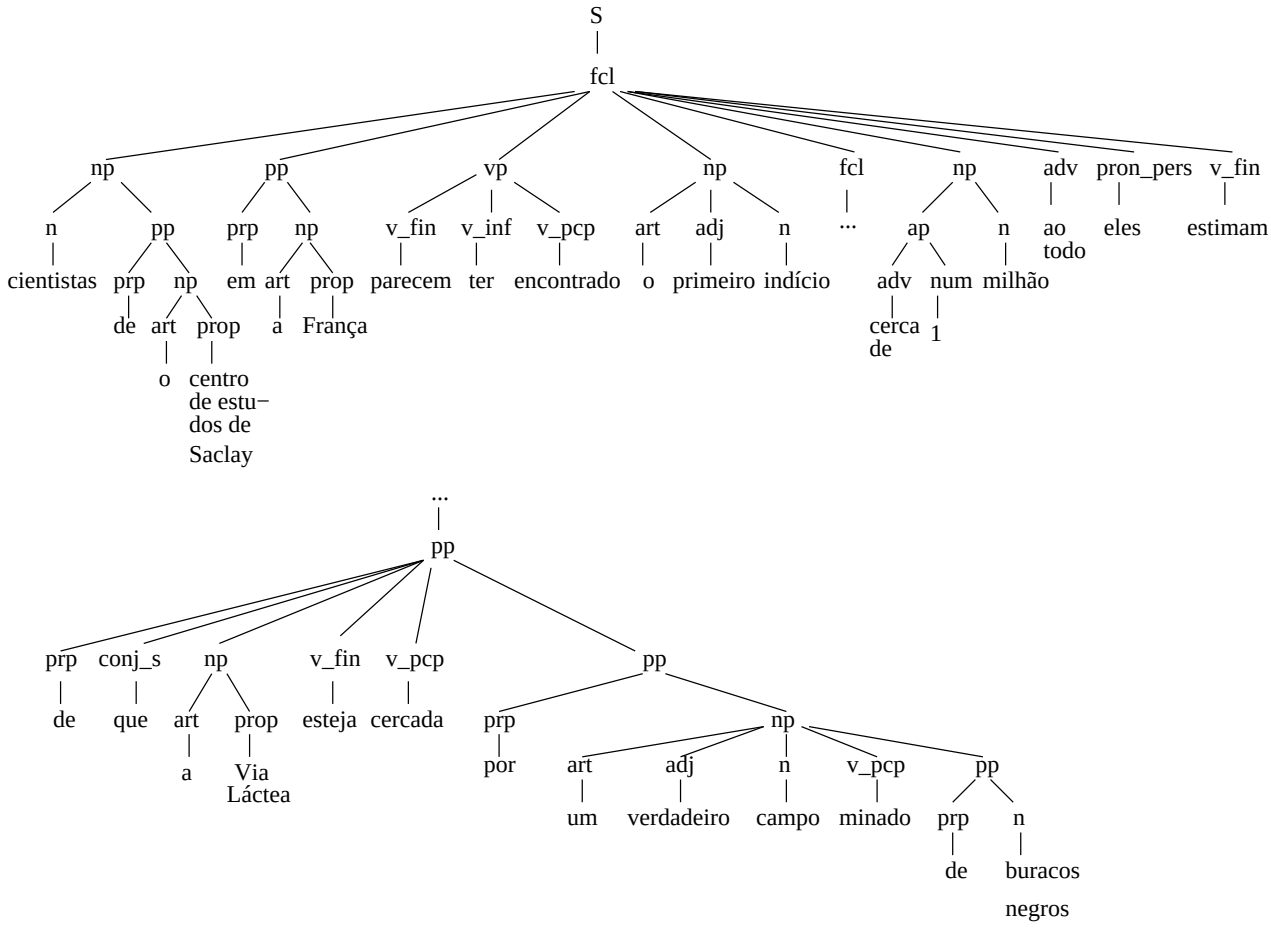

Figura 14: Árvore de derivação relativa à restrição de ligação encontrada em CIENCIA_2001_19858.

3). A principal diferença entre os algoritmos baseados na Teoria da Centralização e os algoritmos baseados em Teoria das Veias está na capacidade da segunda de identificar referentes situados a mais de um enunciado do pronome. Na maioria dos casos, os algoritmos baseados em Teoria da Centralização só são capazes de encontrar referentes há mais de um enunciado de distância quando há referência evocativa [6], ou seja, quando há outro pronome no enunciado anterior que também o referencia, como no exemplo da Figura 15, onde o pronome "ele" do enunciado 3 se refere a "mamífero"; porém, um algoritmo baseado em Teoria da Centralização só consegue escolher esse referente graças à referência feita pelo pronome "ele" no enunciado 2. Por outro lado, os algoritmos baseados na Teoria das Veias são capazes de escolher referentes em enunciados distantes do pronome sem nenhuma restrição, além do próprio conceito de acessibilidade imposto pela teoria, como descrito na seção 2.2 deste trabalho.

1. a coisa muda de figura com o novo mamífero ou o que sobrou de ele
Cf = {coisa, ele=mamífero, figura, mamífero}
Cb = Nenhum

2. ele foi achado em meio a sedimentos de origem marinha
Cf = {ele=ele=mamífero, meio, sedimentos, origem}
Cb = ele

3. pouco abaixo de ele em as camadas de rocha estão mariscos
fósseis que se extinguiram em o fim de o cretáceo enquanto lhe
faziam companhia moluscos típicos de o paleoceno
Cf = {ele=ele=ele=mamífero, paleoceno, moluscos, fim, companhia,
lhe=ele=ele=ele=mamífero, mariscos, camadas}
Cb = ele

Figura 15: Exemplo de referente encontrado a mais de um enunciado de distância pelo algoritmo BFP.

No corpus estudado, há 20 casos onde o referente está localizado a mais de um enunciado de distância do pronome, representando 15,5% do corpus. Em 6 casos (30%), nenhum algoritmo baseado em Teoria da Centralização foi capaz de encontrar o referente, enquanto algum algoritmo baseado em Teoria das Veias foi. Em 9 casos (45%), tanto algoritmos baseados em Teoria da Centralização quanto algoritmos baseados em Teoria das Veias foram capazes de encontrar os referentes. Por outro lado, em 5 casos (25%) nenhum algoritmo foi capaz de encontrar os referentes.

Quanto a anáforas intra-sentenciais, não foi observado nenhum caso em que algum algoritmo baseado em Teoria da Centralização encontrou o referente correto e que nenhum algoritmo baseado em Teoria das Veias foi capaz de encontrá-lo. Por outro lado, há casos em que somente o algoritmo VT-BFP foi capaz de encontrar o referente correto. Isso se deve ao fato de que esse algoritmo busca por entidades intra-sentenciais, usando a pre-

ferência sujeito > objeto direto > objeto indireto > outros > adjunto. Com isso, é possível encontrar referentes, como no exemplo da Figura 16, onde a escolha do referente "buracos negros" para o pronome "eles" é feita porque o referente ocupa a posição de sujeito. Outros algoritmos, como o VT-LRC, ou mesmo o LRC, escolheriam "quilômetros" por estar mais próximo do início do enunciado.

---

apesar disso esse objeto viajando por o espaço a 400 mil quilômetros por hora
(1/2703 de a velocidade de a luz) é um lembrete incômodo de que buracos negros
não têm as trajetórias comportadas e previsíveis que todos gostariam que eles tivessem

veia = {objeto,eles=buracos negros,buracos negros,trajetórias,lembrete,quilômetros,luz,
velocidade,espaço,anos,milhões,...}

Figura 16: Exemplo de referente intra-sentencial encontrado pelo algoritmo VT-BFP.

Por fim, em relação ao domínio de acesso, em 51 casos (39,5% do corpus), o referente correto não está contido na veia do nó da árvore RST em que o pronome se encontra. Nesses casos, nenhum algoritmo baseado em Teoria das Veias foi capaz de encontrar o referente. Dentre esses, em 17 casos (13,1% do corpus) algum algoritmo baseado na Teoria da Centralização foi capaz de resolver o pronome. De fato, isso se assemelha aos resultados apresentados em [22], onde o algoritmo LRC foi usado como "*backup*", promovendo um aumento de 14% no desempenho.

6 Conclusão

Neste trabalho apresentamos três novos algoritmos baseados em Teoria das Veias para a resolução de pronomes em língua portuguesa, inspirados em algoritmos já existentes para a mesma finalidade, mas baseados em Teoria da Centralização. Em complemento, também descrevemos suas implementações, analisamos a complexidade desses algoritmos e suas características principais, comparando-os com os algoritmos BFP, LRC e *S-List*.

Nenhum algoritmo baseado em Teoria das Veias obteve os melhores resultados em nenhum experimento. Esse resultado pode ser devido ao fato de 39,5% das referências do corpus estarem fora das veias. Nessas condições, o melhor resultado ainda é apresentado pelo algoritmo LRC original: 53% ao usar segmentação por sentenças, sem verificar restrições de ligação.

Para trabalhos futuros, será interessante elaborar algoritmos híbridos, baseados em Teoria das Veias e Teoria da Centralização que aproveitem melhor ambas as teorias. Uma hipótese seria utilizar um algoritmo baseado em Teoria da Centralização para encontrar referentes até o enunciado anterior e, se não encontrar, então procurar por enunciados mais distantes, através do domínio de acessibilidade imposto pela Teoria das Veias. Também podem ser feitas comparações entre algoritmos baseados em Teoria das Veias com outros baseados na pilha de estados atencionais [11], ou outro método que também considere entidades distantes dos pronomes.

Por fim, algumas variáveis consideradas pelos algoritmos aqui descritos podem também ser utilizadas como parte de métodos baseados em aprendizado de máquina para resolução de pronomes, como o fato de uma entidade pertencer ou não a veia onde o pronome está, sua classificação gramatical e distância em relação ao início do enunciado, entre outros.

Referências

[1] Ana Aires, Jorge Coelho, Sandra Collovini, Paulo Quaresma, and Renata Vieira. Avaliação de centering em resolução pronominal da língua portuguesa. In *5th International Workshop on Linguistically Interpreted Corpora of the Iberamia'2004*, pages 1–8, Puebla, Mexico, 2004.

[2] Eckhard Bick. *The Parsing System "Palavras". Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework*. PhD thesis, University of Arhus, 2000.

[3] Susan E. Brennan, Marilyn W. Friedman, and Carl Pollard. A centering approach to pronouns. In *ACL' 87 Proceedings of the 25th Annual Meeting on Association for Computational Linguistics*, pages 155–162, Stanford, CA, 1987.

[4] Thiago I. Carbonel, Sandra S. Collovini, Jorge C. Coelho, Juliana T. Fuchs, Lucia H. M. Rino, and Renata Vieira. Summ-it: Um corpus anotado com informações discursivas visando à sumarização automática. In *Proceedings of XXVII Congresso da SBC: V Workshop em Tecnologia da Informação e da Linguagem Humana TIL*, pages 1605–1614, Rio de Janeiro, Brazil, 2007.

[5] Noam Chomsky. *Lectures on Government and Binding*. Dordrecht: Foris, 1981.

[6] Dan Cristea. Motivations and implications of veins theory: a discussion of discourse cohesion. *International Journal of Speech Technology*, 12(2):83–94, 2009.

[7] Dan Cristea, Nancy Ide, and Laurent Romary. Veins theory: A model of global discourse cohesion and coherence. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and of the 17th International Conference on Computational Linguistics*, pages 281–285, Montreal, Canada, 1998.

[8] Ramon R. M. Cuevas, Willian Y. H., Diego J. de Lucena, Ivandré Paraboni, and Patrícia R. Oliveira. Portuguese pronoun resolution: Resources and evaluation. In *CICLing'08 Proceedings of the 9th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 344–350, Mexico City, Mexico, 2008.

[9] Fernando J. V. da Silva, Ariadne M. B. R. Carvalho, and Norton T. Roman. A comparative analysis of centering-based algorithms for pronoun resolution in portuguese. In *Proceedings of the 12th Ibero-American Conference on Advances in Artificial Intelligence*, pages 336–345, Bahia Blanca, Argentina, 2010.

[10] Caroline Gasperin, Renata Vieira, Rodrigo Goulart, and Paulo Quaresma. Extracting xml syntactic chunks from portuguese corpora. In *TALN, Workshop on Natural Language Processing of Minority Languages*, Batz-sur-Mer, France, 2003.

[11] Barbara J. Grosz and Candy Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, 1986.

[12] Graeme Hirst. *Anaphora in Natural Language Understanding: A Survey*, volume 119 of *Lecture Notes in Computer Science*. Springer, 1981.

[13] Jerry R. Hobbs. Pronoun resolution. Research Report 76-1, Department of Computer Sciences, City College, City University of New York, Artificial Inteligence Center SRI International 333 Ravenswood Ave Menlo Park, California 94025, 1976.

[14] Shallom Lappin and Herbert J. Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561, 1994.

[15] Thiago A. S. Pardo. *Métodos para Análise Discursiva Automática.* PhD thesis, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2005.

[16] Ellen Prince. *Radical Pragmatics*, chapter Toward a taxonomy of given-new information, pages 223–255. Academic Press, 1981.

[17] Beatrice Santorini and Anthony Kroch. *The Syntax of Natural Language: An Online Introduction Using the Trees Program.* <URL:http://www.ling.upenn.edu/~beatrice/syntax-textbook>, 2000. (Accessed 15/Dec/2010).

[18] Michael Strube. Never look back: An alternative to centering. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 1251–1257, Morristown, NJ, 1998.

[19] Michael Strube and Udo Hahn. Functional centering – grounding referential coherence in information structure. *Computational Linguistics*, 25:309–344, 1999.

[20] Maite Taboada and William C. Mann. *Introduction to RST (Rhetorical Structure Theory).* <URL:http://www.sfu.ca/rst/01intro/intro.html>, 2005. (Accessed 25/July/2010).

[21] Joel R. Tetreault. Analysis of syntax-based pronoun resolution methods. In *ACL '99 Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 602–605, College Park, MD, 1999.

[22] Joel R. Tetreault and James Allen. An empirical evaluation of pronoun resolution and clausal structure. In *2003 International Symposium on Reference Resolution and its Applications to Question Answering and Summarization*, pages 1–8, Venice, Italy, 2003.