Universidade de São Paulo - USP
Universidade Federal de São Carlos - UFSCar
Universidade Estadual Paulista – UNESP

Criação de um grande repositório público de Entidades Nomeadas Abreviadas extraídas de um Corpus Histórico do Português do Brasil: automatizando a extração de padrões

Kátia Tiemi Hirotsu
Sandra Maria Aluísio

NILC-TR-08-16
Dezembro, 2008

Série de Relatórios do Núcleo Interinstitucional de Lingüística Computacional
NILC - ICMC-USP, Caixa Postal 668, 13560-970 São Carlos, SP, Brasil

Agradecimentos

A minha família pelo apoio durante a minha vida e principalmente durante os períodos mais difíceis.

Aos amigos que fiz durante a decorrer este curso, agradeço por toda ajuda durante essa etapa da minha vida.

E a minha orientadora que me ajudou durante a realização deste trabalho.

Índice

Índice de Figuras

Resumo

O principal termo relacionado a esta monografia é **Entidade Nomeada** (EN), que é definida como nome próprio de pessoa, organização, local, acontecimento, coisa (objeto nomeado), obra (artefato e construção humana), conceito abstrato, além de data e valor. O contexto em que uma EN está inserida não é considerado. A tarefa de reconhecer as Entidades Nomeadas de um texto é muito importante não só para a área de Lingüística Computacional, mas também para muitas outras disciplinas, uma vez que através dessas entidades é possível compreender melhor o documento dos quais elas foram extraídas. Este trabalho de conclusão de curso está inserido no escopo do projeto *Dicionário Histórico do Português do Brasil* (DHPB), um projeto do programa Institutos do Milênio do CNPq, cujo principal objetivo é a criação de um dicionário de Português do Brasil referente aos mesmos documentos utilizados neste projeto. O objetivo deste projeto de conclusão é construir um repositório de acesso público com EN abreviadas e seus atributos encontrados no corpus de documentos históricos brasileiros do século XVI ao XIX do projeto DHPB. Entre esses atributos estão: a categoria da abreviatura, sua expansão, a oração da qual ela foi extraída, entre outros. Porém, foi observada a dificuldade em realizar o processo de construção desse repositório e por esse motivo é proposta a criação de uma aplicação que tem como meta facilitar essa atividade. Dessa forma, a criação de um repositório deste tipo se torna menos dispendioso em termos de tempo e trabalho humano. No final espera-se que as informações das EN abreviadas sejam úteis, de forma a evitar expansões erradas das abreviações e assim comprometer a compreensão do texto no qual elas estão inseridas.

1. Introdução

1.1 Contextualização e motivação

Este projeto é uma continuação do Projeto de Graduação I ("Criação de um grande repositório público de Entidades Nomeadas Abreviadas extraídas de um Corpus Histórico do Português do Brasil") apresentado no primeiro semestre de 2008 ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP. Dessa forma, a área em que esse estudo está inserido continua sendo a Lingüística Computacional, mais especificamente relaciona-se à subárea de Lingüística de Corpus e está inserido em um projeto mais amplo, o Dicionário Histórico do Português do Brasil (DHPB), cujo objetivo é a elaboração de um dicionário composto por palavras encontradas em documentos históricos brasileiros.

O DHPB é um projeto do programa Institutos do Milênio, do CNPq, composto pelos grupos NILC[1] (Núcleo Interinstitucional de Lingüística Computacional – Instituto de Ciências Matemáticas e de Computação da USP de São Carlos), DL[2] (Departamento de Letras da Universidade Federal de São Carlos), Faculdade de Ciências e Letras da Unesp de Araraquara[3] entre outras instituições nacionais e internacionais. O corpus utilizado no DHPB é constituído por aproximadamente 7,5 milhões de palavras de textos históricos escritos entre os séculos XVI e o começo do século XIX, compondo-se de cartas de jesuítas missionários, documentos de bandeirantes, reportagens de sertanistas e documentos da Inquisição. Todos esses documentos foram digitalizados, além de um dicionário de abreviaturas composto por abreviaturas dos séculos XVI ao século XIX [Flexor, 1991], para que o processamento computacional pudesse ser realizado. Contudo, como descrito em Vale et al. [2008], o dicionário Flexor exibe muitas abreviações não encontradas no corpus do projeto DHPB – somente 16% delas estão no corpus. Numa tentativa de estimar o número de abreviaturas do corpus DHPB diferentes das do dicionário Flexor, Candido (2008) criou heurísticas para extrair abreviaturas num total de 7.045 com os seguintes padrões e porcentagens de contribuição dos padrões:

- Presença de sobrescrito: ant.^{o} , cid.^{e} , p.^{a} (61%)
- Ponto interno sucedido por até 4 símbolos: cid.e , embg.e , ex.mo (24%)
- Palavras terminadas por algumas consoantes: cap , reg , liv , v (15%)

Somente 35% destas abreviaturas levantadas estão em Flexor. Desta forma, concluímos que o dicionário Flexor poderia ser melhorado com o uso de heurísticas como as apresentadas aqui. Este projeto de conclusão trabalha neste sentido.

[1] http://www.nilc.icmc.usp.br/nilc/index.html
[2] http://www.ufscar.br/~letras/index.php
[3] http://www.fclar.unesp.br/

No projeto de Graduação I foram construídos padrões para recuperar EN da categoria "local" (seguindo a divisão em categorias adotadas pela HAREM[4]). O processador de corpus utilizado foi o UNITEX-MILENIO[5], que possui uma modificação do alfabeto do UNITEX para atender as necessidades do projeto DHPB, desenvolvido em um projeto de mestrado do ICMC [Candido, 2008]. Também foram utilizados o dicionário de variantes (que segue a metodologia SIACONF - Sistema de Apoio à Contagem de Freqüência em Corpus - proposta por Giusti et. al. [2007]) e o Philologic[6] (um processador de corpus baseado na Web) que auxiliaram na apresentação de possíveis variações de palavras presentes nos documentos.

A experiência adquirida durante o primeiro semestre de 2008 foi importante porque indicou quais são os problemas reais enfrentados pelos pesquisadores da área de lingüística de corpus que manipulam textos históricos. O maior obstáculo encontrado no trabalho anterior foi a ausência de uma ortografia nesses documentos antigos, que dá origem a uma grande quantidade de variações na escrita de uma mesma palavra. Esse problema é ainda mais grave quando a palavra é uma abreviação, pois uma vez que a abreviação é expandida de maneira errada, pode levar a uma interpretação equivocada do documento. Essa dificuldade em lidar com muitas variantes e formas abreviadas de vocábulos, além das conseqüências geradas por uma expansão errada da abreviação, é mais bem explicada no relatório técnico resultante da monografia do projeto anterior [Hirotsu e Aluisio, 2008].

Em razão da dificuldade apresentada pela ausência de uma ortografia vigente na época e considerando uma adaptação da metodologia empregada no reconhecimento de EN de textos contemporâneos para a criação do repositório REPENTINO[7] pode-se observar que a parte de validação manual dos resultados obtidos pela busca de padrões nos documentos históricos, ou seja, a última fase do método é a que consome mais tempo e torna o trabalho mais difícil. Essa é uma das motivações para este estudo, que tem como objetivo a criação de uma aplicação para facilitar a ação de recuperação de EN, poupando tempo e esforço dos pesquisadores que trabalham nesse tipo de atividade.

1.2 Objetivos do trabalho

Nesta monografia é proposta a criação de uma aplicação que tem a finalidade de facilitar o trabalho de pesquisadores da área de lingüística de corpus na tarefa de Reconhecimento de Entidades Nomeadas (REN) presentes em corpus históricos brasileiros. Além disso, outra meta é acrescentar mais EN abreviadas (tanto da categoria "local" como das

[4] Avaliação de Reconhecedores de Entidades Mencionadas – organizada pela Linguateca (http://www.linguateca.pt/)
[5] http://moodle.icmc.usp.br/milenio/
[6] http://moodle.icmc.usp.br/philologic-milenio/
[7] http://poloclup.linguateca.pt/repentino/

demais categorias do HAREM) retiradas do corpus do projeto e conseqüentemente tornar o repositório mais rico e completo.

1.3 Organização da monografia

Esta monografia está dividida em quatro capítulos. Na introdução apontamos os desafios impostos durante a manipulação de textos do português histórico do Brasil e a motivação para a realização da aplicação de REN. No segundo são apresentados os conceitos importantes para entender o conteúdo dessa monografia, além de uma apresentação sucinta dos estudos que auxiliaram na realização deste projeto. O Capítulo 3 fornece as etapas executadas, além dos resultados obtidos. No capítulo quatro é estabelecida uma conclusão com base nos resultados obtidos ao projeto.

2. Revisão bibliográfica

2.1 Conceitualização e terminologia

Perl é uma linguagem estável e multiplataforma criada por Larry Wall, sendo utilizada em projetos de missão crítica. Além de ser "open source" (código aberto) e possuidora das melhores características de outras linguagens como: C, awk, sed, sh, BASIC, entre outras, também possui muitos módulos prontos, como Perl/Tk para criação de interface gráfica, módulos para executar a conexão com bancos de dados (como PostgreSQL e MySQL), além de muitos outros disponíveis no site do CPAN[8] (Comprehensive Perl Archive Network).

Por causa dessas propriedades e da sua versatilidade para busca de padrões em textos através de expressões regulares, o Perl foi escolhido como a linguagem de programação para construir a aplicação de Reconhecimento de Entidades Nomeadas (ARENA) construída neste projeto.

Um outro termo importante citado na Introdução e que aparecerá no decorrer desta monografia é o HAREM que consiste de uma avaliação conjunta de reconhecimento de Entidades Mencionadas. Chama-se "avaliação conjunta" porque é promovida uma comparação entre os sistemas dos participantes, com base em um conjunto de tarefas consensuais numa determinada área, usando um grupo de recursos em comum e uma métrica consensual. O julgamento do desempenho de sistemas de identificação e classificação de Entidades Mencionadas em textos é baseado em uma coleção de documentos de Língua Portuguesa, e é realizado pela Linguateca.

A metodologia aplicada neste estudo para recuperar EN abreviadas é a mesma empregada pelo é REPENTINO[9] (repositório de Entidades Nomeadas do português moderno)

[8] http://cpan.perl.org/
[9] http://www.linguateca.pt/repentino/

[Sarmento et. al., 2006], um repositório público que contém EN localizadas em textos contemporâneos. Sua construção é coletiva, contando com diversos pesquisadores. Atualmente, conta com 450.181 exemplos de entidades nomeadas e sua principal função é auxiliar no desenvolvimento de sistemas de Reconhecimento de EN da língua portuguesa, pois os exemplos armazenados ajudam a construir regras para identificar as EN.

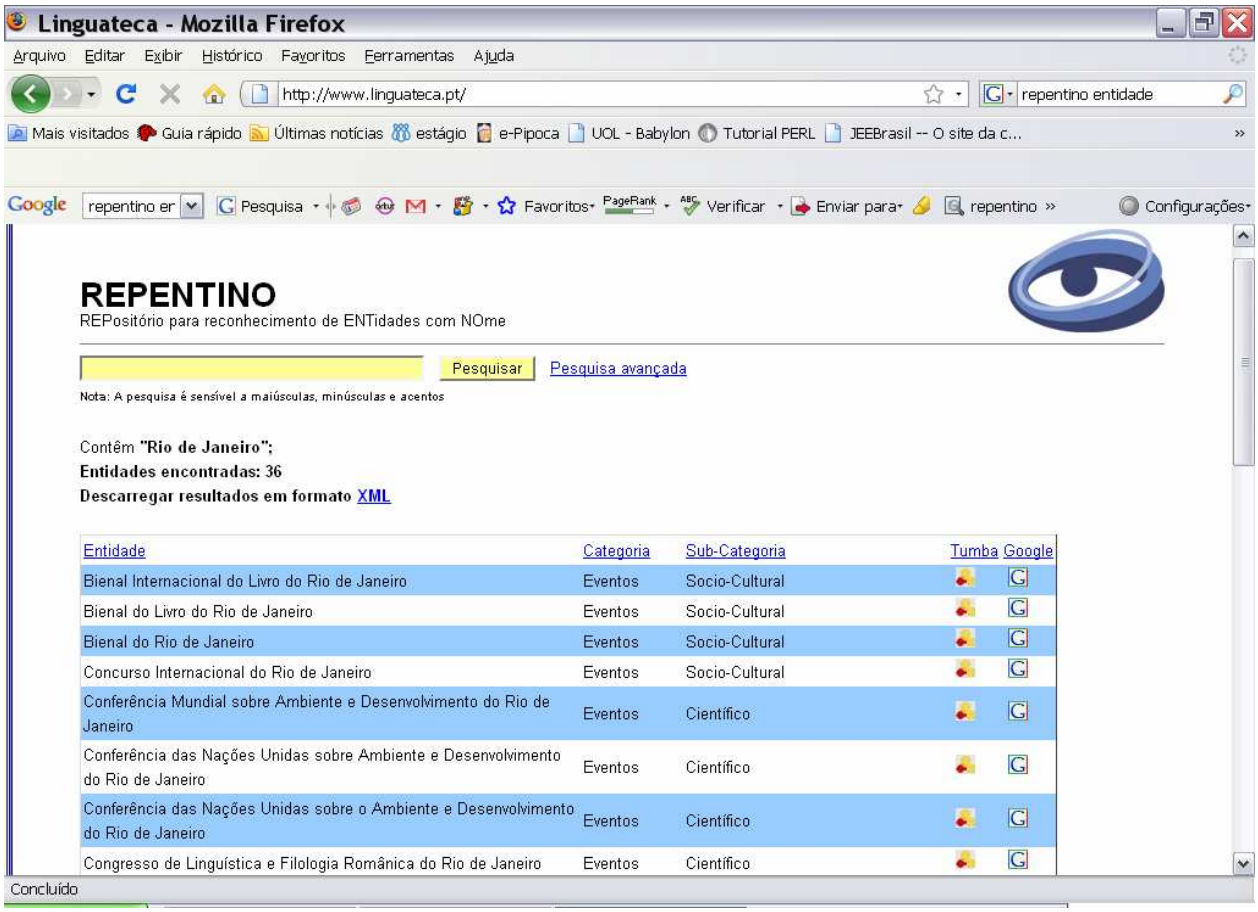

Figura 1: saída do REPENTINO para o termo "Rio de Janeiro"

2.2 Trabalhos relacionados

Existem diversos trabalhos relacionados à tarefa de REN, porém a sua grande maioria é construída para um corpus formado por textos contemporâneos, que possuem atributos diferentes do corpus deste projeto, que apresenta variações de escrita de um mesmo vocábulo. Todavia, estudos relacionados ao REN foram pesquisados para dar uma visão geral de como ocorre o reconhecimento de Entidades, além de mostrar alternativas para realizar essa atividade. Nas próximas seções são apresentados, resumidamente, alguns desses estudos.

2.2.1 Sistema Malinche

Inicialmente, o sistema Malinche [Solorio, 2007] foi desenvolvido para o Espanhol, mas provou-se que não seriam necessárias adaptações para que o mesmo sistema trabalhasse com REN em textos da língua portuguesa. Neste estudo, é empregada a estratégia de aprendizado de

máquina para realizar a atividade de REN. Para isso, foi escolhido o algoritmo de aprendizagem Support Vector Machine (SVM) ([Vapnik, 1995]; [Stitson et al., 1996]), mais especificamente a implementação do SVM inclusa no software livre WEKA[10].

O problema de REN é dividido em duas etapas, a primeira é chamada de delimitação das EN, e é responsável pela determinação de quais palavras podem ser consideradas EN. Já a segunda etapa denomina-se classificação de EN e encarrega-se da classificação da entidade em uma das seguintes classes: pessoa, organização, localização e variado.

Cada palavra possui cinco atributos para que o reconhecimento seja feito. Estes atributos são:

1. Informação ortográfica (se a palavra começa ou não com maiúscula);
2. Posição da palavra na sentença;
3. A própria palavra;
4. e 5. Correspondem ao contexto da palavra (conjunto de rótulos que indicam se a palavra está no começo de uma entidade, se pertence à entidade ou se não estão em nenhuma das duas opções anteriores, além de anotações PoS).

Estes atributos são mais bem descritos no trabalho de Solorio [2007].

Os resultados apresentados no artigo citado acima mostram que a utilização de algoritmos de aprendizado de máquina é uma boa estratégia para o reconhecimento de entidades. No HAREM de 2005, a medida P[11] para a tarefa de identificação é de 49,68% [Cardoso, 2007].

2.2.2 Sistema Cortex[12]

Foi desenvolvido por Christian Nunes Aranha em sua tese de doutorado. O reconhecimento dos termos ocorre com ajuda de um autômato para identificar padrões de formação de entidades compostas com base num repertório de regras. São várias etapas até o término do reconhecimento das entidades. Quanto mais textos ele processa mais conhecimento lingüístico é acumulado, pois aprende a partir deles.

Ele tem quatro fontes de dados:

- Almanaque: lista de entidades de uma determinada categoria obtida de enciclopédica
- Metapalavras: lista de termos que aparecem nas vizinhanças das entidades
- Adivinhação: conjunto de termos que constituem as entidades mencionadas (por exemplo, Prof., Dr.)

[10] http://www.cs.waikato.ac.nz/ml/weka/

[11] Mede a eficiência do sistema para delimitar corretamente as entidades.

[12] www.cortex-intelligence.com

- Léxico: armazena todo o conhecimento aprendido através de textos já processados pelo Cortex

No HAREM os valores das medidas foram: precisão (65,57%), abrangência (86,69%), medida f (0,7466) [Aranha, 2007].

2.2.3 Sistema SIEMÊS

SIEMÊS[13] (Sistema de Identificação de Entidades Mencionadas com Estratégia Siamesa) [Sarmento, 2006] é um sistema híbrido apoiado por regras e por uma Base de dados com exemplos de entidades mencionadas já classificadas. É desenvolvido pela Linguateca e utiliza regras de forma e de semelhança para identificar e classificar as entidades mencionadas em texto livre. O cálculo de semelhanças do SIEMÊS é feito com base no REPENTINO. Sua medida P no HAREM de 2005 é 76,75% na tarefa de identificação [Cardoso, 2007].

3. Estado atual do trabalho

3.1 Projeto

A seqüência de passos que o usuário deve seguir na ARENA é baseada nas etapas da metodologia aplicada na criação do Repositório REPENTINO. Ela consiste de seis fases:

1. Escolher uma categoria para a qual se pretende pesquisar exemplos de entidades nomeadas.

2. Decidir uma estratégia apropriada para a pesquisa desses exemplos, que podem ser divididos em três formas:

   a) busca por palavras abreviadas colocadas ao lado esquerdo de certos tipos de Entidades Nomeadas (palavras que recebem o rótulo <INIT>).

   b) busca utilizando o contexto “local”, ou seja, palavras que indicam a presença de Entidades Nomeadas próximas a elas (exemplo: “localizado na XXX”, “próximo da XXX”, cuja probabilidade de que “XXX” indique uma Entidade Nomeada da categoria local).

   c) sufixos discriminatórios (exemplo: atualmente existem as partículas “Ltda.”, “S.A.”, indicando a presença de nomes de organizações próximos a essas partículas).

3. Construção de um padrão para ser usado em algum programa para a realização da busca no corpus.

4. Validação manual dos resultados obtidos. É necessário verificar se o resultado se enquadra na categoria pesquisada.

[13] http://poloclup.linguateca.pt/repentino/faq.html?#q14

5. Inserção dos resultados validados na etapa anterior no repositório.

6. Criação de uma nova categoria ou subcategoria, aumentando o sistema de classificação taxonômico. Essa etapa é opcional.

A última etapa não é executada porque se tomou como modelo a taxonomia definida no

O rótulo <INIT>, que foi citado anteriormente, é um rótulo para simbolizar um atributo da palavra. Entre os rótulos mais importantes neste projeto, estão:

- <ENT>: representa Entidade Nomeada;
- <INIT>: representa uma colocação presente ao lado esquerdo de alguns tipos de EN;
- <ABREV>: representa palavras abreviadas;

A referência Vale et al. [2008] possui uma explicação mais minuciosa sobre os atributos criados para o projeto DHPB.

Como o corpus do projeto é constituído de textos com um grande número de variações na sua escrita, uma adaptação na metodologia descrita anteriormente é importante para recuperar o maior número possível de EN abreviadas. A adaptação sugerida é retornar também as variações de grafia de um padrão de busca. Essas variantes podem ser encontradas no dicionário de variação de grafia, desenvolvido no projeto DHPB e que segue a metodologia do SIACONF proposta por Giusti et al. [2007]. São empregadas 43 regras de transformações para agrupar variações sob uma forma ortográfica. O Philologic utiliza o AGREP e é uma ferramenta que exibe resultados cuja grafia é similar à palavra digitada pelo usuário, portanto é mais um recurso que ajuda na obtenção de variações de termos. No trabalho Vale et al. [2008] encontra-se um maior detalhamento sobre este assunto. Essa adaptação só não foi realizada neste projeto porque o tempo não foi suficiente para realizá-la, porém é uma ótima sugestão para trabalhos futuros.

3.2 Descrição das atividades realizadas

3.2.1 Mudança do Gerenciador de Banco de Dados

Houve a necessidade de se mudar o gerenciador de Banco de Dados do PostgreSQL para o MySQL, uma vez que o servidor, na qual a aplicação web criada no projeto anterior será instalada, já possui o MySQL funcionado corretamente. Portanto, essa migração de gerenciador de Banco de Dados torna o trabalho de instalação da aplicação web mais fácil.

A seguir está o código SQL para a construção da tabela que armazena os dados da EN validadas manualmente pelo ser humano.

```
create table entidadeNomeada(
     idAbreviacao integer not null AUTO_INCREMENT,
     abreviacao LONGTEXT not null,
```

```
            expansao LONGTEXT not null,
            categoria TEXT not null,
            texto LONGTEXT,
            primary key (idAbreviacao));
```

O driver utilizado na aplicação web para se conectar com o Banco de Dados teve que ser modificado, e essa foi a única variação que ocorreu no seu código fonte.

3.2.2 Aprendizagem e familiarização da linguagem Perl

Um tempo de aprendizagem foi necessário devido à falta de experiência e conhecimento em programação usando a linguagem Perl. Durante esse período, a leitura de tutoriais[14] disponíveis na internet, além de livros específicos da linguagem[15] em questão foram muito importantes e ajudaram a adquirir noções básicas para a implementação da ARENA.

3.2.3 Criação da ARENA (Aplicação para Reconhecimento de Entidade Nomeada)

O desenvolvimento da ARENA é dividido em vários ciclos, e em cada ciclo é adicionada uma nova função (no caso do Perl ela é chamada de sub-rotina) que depois é testada para verificar se a resposta obtida é a mesma que a esperada pelo desenvolvedor. Primeiro são implementados os principais requisitos (com maior prioridade), como a busca por um padrão nos textos, a janela para exibir os resultados obtidos, entre outros. Dado o tempo limitado para o projeto, os requisitos escolhidos para implementação são aqueles que possuem a maior precedência entre o restante, ou seja, aqueles que são básicos para a tarefa de Reconhecimento de EN.

A implementação da aplicação foi baseada na mesma metodologia aplicada durante a construção do REPENTINO, dessa forma existem três estratégias para reconhecer EN, a busca por termos rotulados como <INIT>, palavras que indicam a presença de EN próximos a ela ou sufixos discriminatórios. Para realizar buscas por abreviaturas considerados <INIT> ou EN já manipuladas pelo projeto DHPB, é utilizada uma planilha Excel que por enquanto possuem apenas abreviações que começam com “a”, “b” e “c”. Essa planilha é o dicionário Flexor

---

[14] http://www.ime.usp.br/~glauber/perl/perl.htm
http://www.numaboa.com.br/informatica/tutor/linguagens/perlExprReg.php
http://perldoc.perl.org/index-tutorials.html

[15] Learning Perl 5ª edição – autores: Randal L. Schwartz, Tom Phoenix & Brian D. Foy, publicado pela O'Reilly
Mastering Perl/TK 1ª edição – autores: by Steve Lidie and Nancy Walsh, publicado pela O'Reilly
Perl in a Nutshell 2ª edição – autores: Nathan Patwardhan, Ellen Siever and Stephen Spainhour, publicado pela O'Reilly

digitalizado com anotações de EN do HAREM e foi construído por um bolsista lingüista do projeto DHPB.

	A	B	C	D	E	F	G	H
1	Entidade?	Abreviatura	Expansão	Forma Canônica	Flexão	Categoria Gramatical	Atributos	Tipo de Entidade
2	+	ã	ano	ano	ms	N	INIT	TEMPO
3	+	Ã	anos	ano	mp	N	INIT	TEMPO
4	+	ã	anos	ano	mp	N	INIT	TEMPO
5		ã	hão	haver	P3p	V		
6	+	Å	Afonso	Afonso	ms	N	ENT	PESSOA
7		a barracam^to	abarracamento	abarracamento	ms	N		
8		a Costam^to	acostamento	acostamento	ms	N		
9	+	a Gregd^o	agregado	agregado	ms	N	INIT	PARENTE
10		a Gregd^o	agregado	agregado	ms	A		
11		a Gregd^o	agregado	agregar	K	V		
12	+	a Remat^e	arrematante	arrematante	ms:fs	N	INIT	TITULO
13	+	A^a	Aranha	Aranha	ms	N	ENT	PESSOA+AMB
14	+	A^a	Aranha	Aranha	fs	N	ENT	PESSOA+AMB
15		A^a	Aranha	aranhar	P3s	V		
16		A^a	Aranha	aranhar	Y2s	V		
17		A^al	auxiliar	auxiliar	ms	A		
18		A^al	auxiliar	auxiliar	fs	A		
19	+	A^al	auxiliar	auxiliar	ms	N	INIT	TITULO
20	+	A^al	auxiliar	auxiliar	fs	N	INIT	TITULO

Figura 2: Planilha que contém abreviações do dicionário Flexor com anotações de EN do HAREM

As colunas da planilha Excel verificadas pela aplicação construída nesse estudo são: “atributos” e “tipo de entidade”. Dependendo da escolha que o usuário fizer na interface ilustrada pela figura 4, o tipo de atributo procurado é diferente. No caso em que a busca for por EN presentes no dicionário (primeira opção), retorna-se as abreviações que possuem “atributo” semelhante a “ENT” e “tipo de entidade” igual à selecionada no menu inicial (figura 3). Já no caso em que a escolha da busca é por INIT presente nos textos (segunda opção), as abreviações exibidas são as que possuem no campo “atributo” a sigla “INIT” e em “tipo de entidade” a categoria optada pelo usuário na primeira interface da aplicação (figura 3).

O código-fonte da ARENA utiliza alguns módulos prontos em Perl que podem ser encontrados no site do CPAN. Entre esses módulos estão:

- Tk: utilizada para construir a interface gráfica vista pelos usuários da aplicação;
- DBI: realiza a conexão da aplicação com o banco de dados (MySQL);
- Spreadsheet::ParseExcel: contém métodos para manipular e extrair informações contidas em planilhas Excel.

Todos os documentos históricos foram copiados para um único arquivo texto (com extensão txt). A cada interação é lida apenas uma linha do arquivo texto e procura-se nela o padrão procurado através da seguinte expressão regular:

```
($var =~/(.*)($palavra)(.*)/i
```

Onde:

- "$var" é uma variável que recebe a linha do arquivo texto;
- "$palavra" é o padrão procurado e "(. *) ($palavra) (.*)" significa qualquer coisa antes ou depois de "$palavra";
- e /i busca sem diferenciar maiúsculas de minúsculas;

Quando a linha lida possuir o padrão desejado, seu número é armazenado em um vetor ("@vetorEN"). Dessa forma, para exibir os resultados, o arquivo "txt" é manipulado como um vetor e mostram-se apenas as linhas cujo número é igual a variável armazenada em cada um dos elementos de "@vetorEN".

Informações mais detalhadas podem ser obtidas através da cópia do código-fonte da ARENA com documentação interna. Ela está disponível como apêndice dessa monografia.

3.3 Resultados obtidos

A seguir, são mostradas as interfaces da ARENA criadas para facilitar o trabalho de Reconhecimento de Entidades Nomeadas. Primeiro, o usuário deve selecionar em qual categoria se enquadra a EN abreviada procurada (Figura 3). As categorias empregadas neste projeto e disponibilizadas no menu da tela inicial são as mesmas que as utilizadas no HAREM.

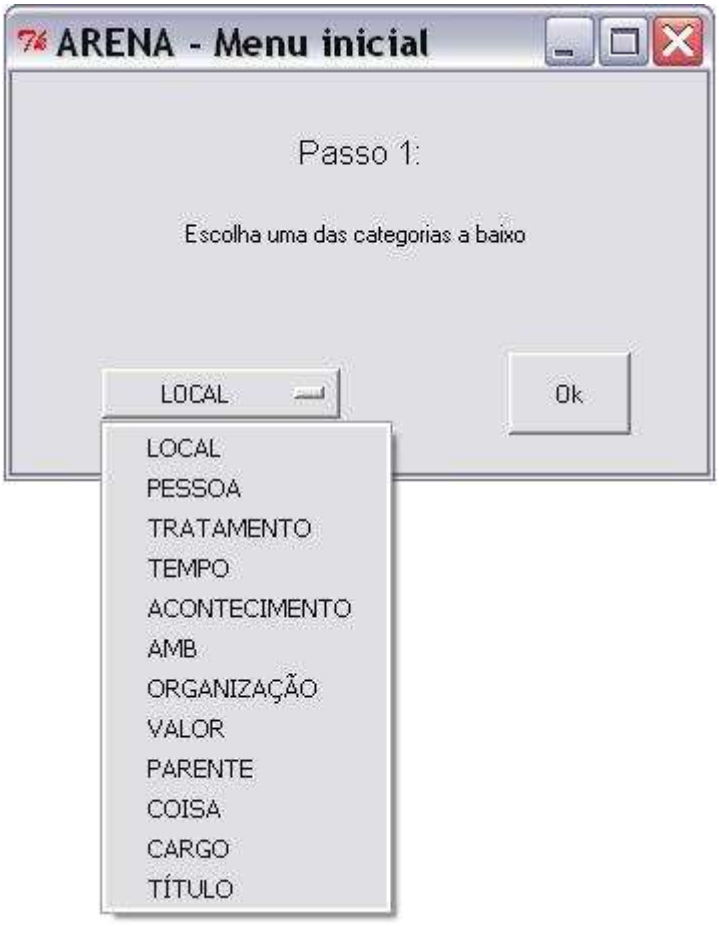

Figura 3: Tela inicial da ARENA - escolha da categoria da EN

Depois é solicitado ao usuário a escolha de uma das opções de busca de padrões (Figura 4).

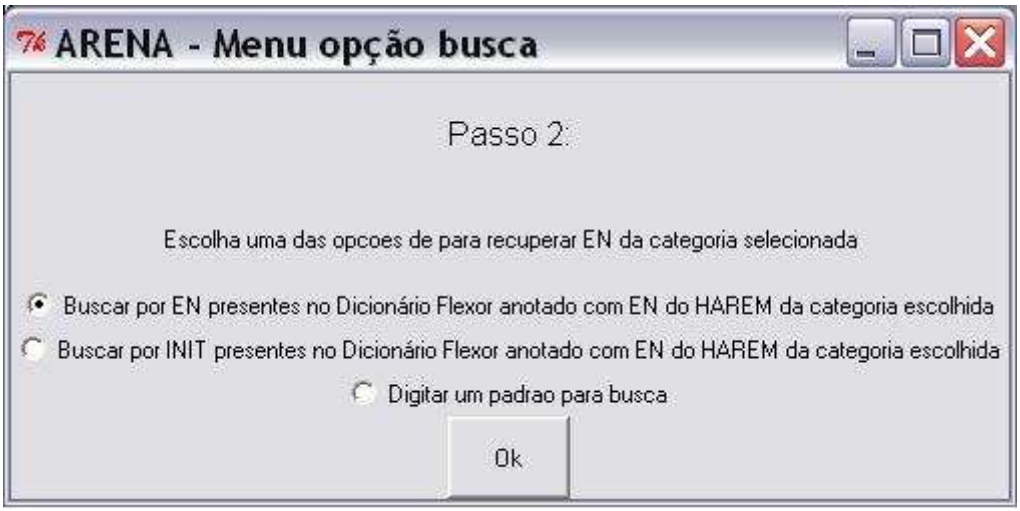

Figura 4: Opções de busca disponibilizadas pela ARENA para realização da busca de EN

Caso a opção do usuário seja a primeira ou a segunda, a próxima janela exibida (figura 5) será uma lista contendo todas as abreviações que possuem na coluna “atributo” da planilha Excel os seguintes valores:

- <ENT> (título dada as Entidades Nomeadas abreviadas) ou
- <INIT> (antecedem Entidades Nomeadas), de acordo com a explicação da seção anterior.

Dessa forma, as buscas por rótulo retornam uma janela com um grande número de vocábulos. Se o usuário clicar duas vezes sobre uma dessas abreviações, uma nova janela aparecerá exibindo a linha do texto em que a abreviação é encontrada (ver Figura 6).

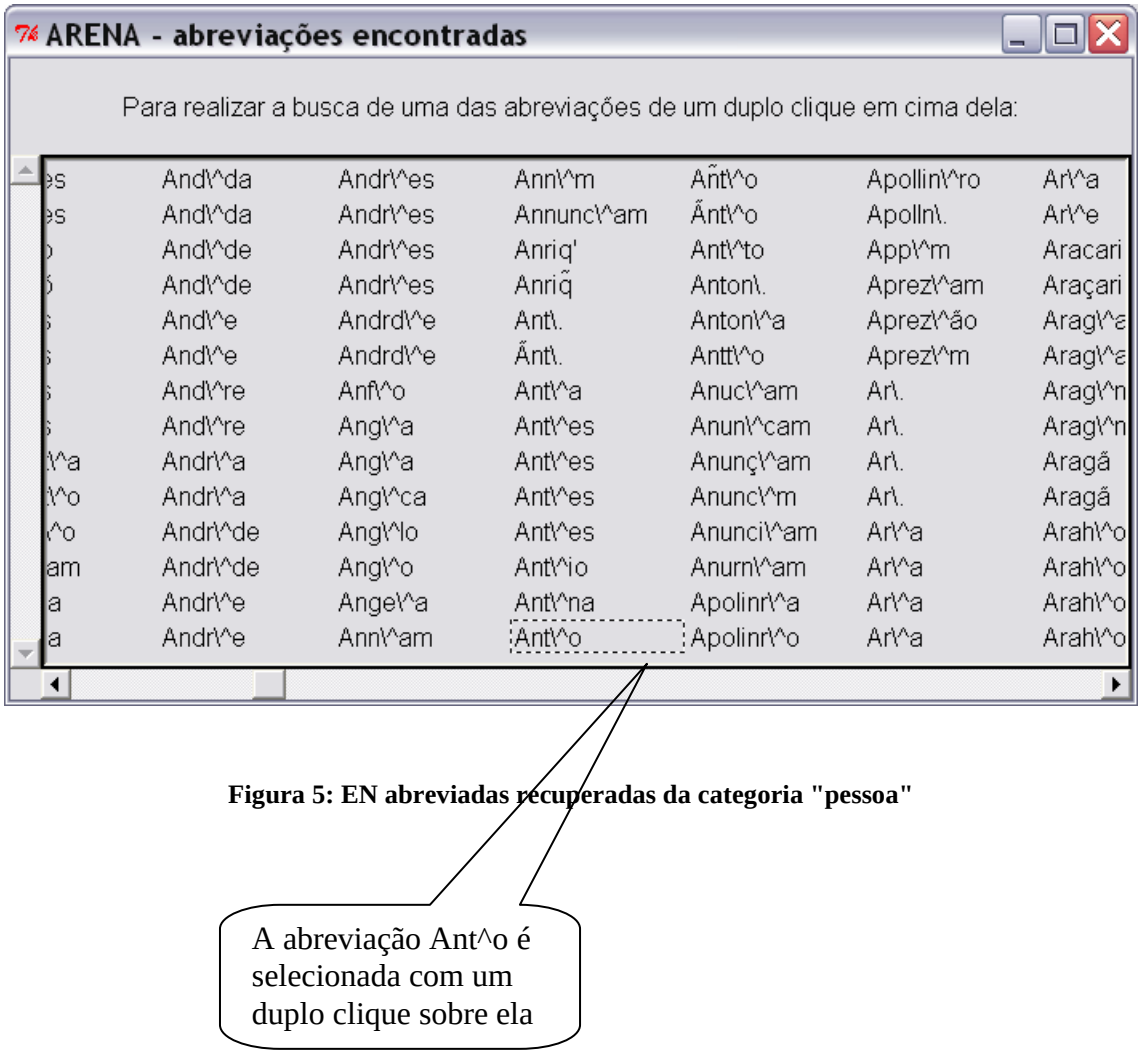

Figura 5: EN abreviadas recuperadas da categoria "pessoa"

A Figura 6 mostra os resultados da busca por “Ant^o”. Cada linha apresentada é uma ocorrência do padrão procurado. Se o usuário clicar duas vezes sobre um dos resultados, uma nova tela aparecerá com as sete linhas anteriores e sete linhas posteriores a aquela que possui o padrão (figura 7).

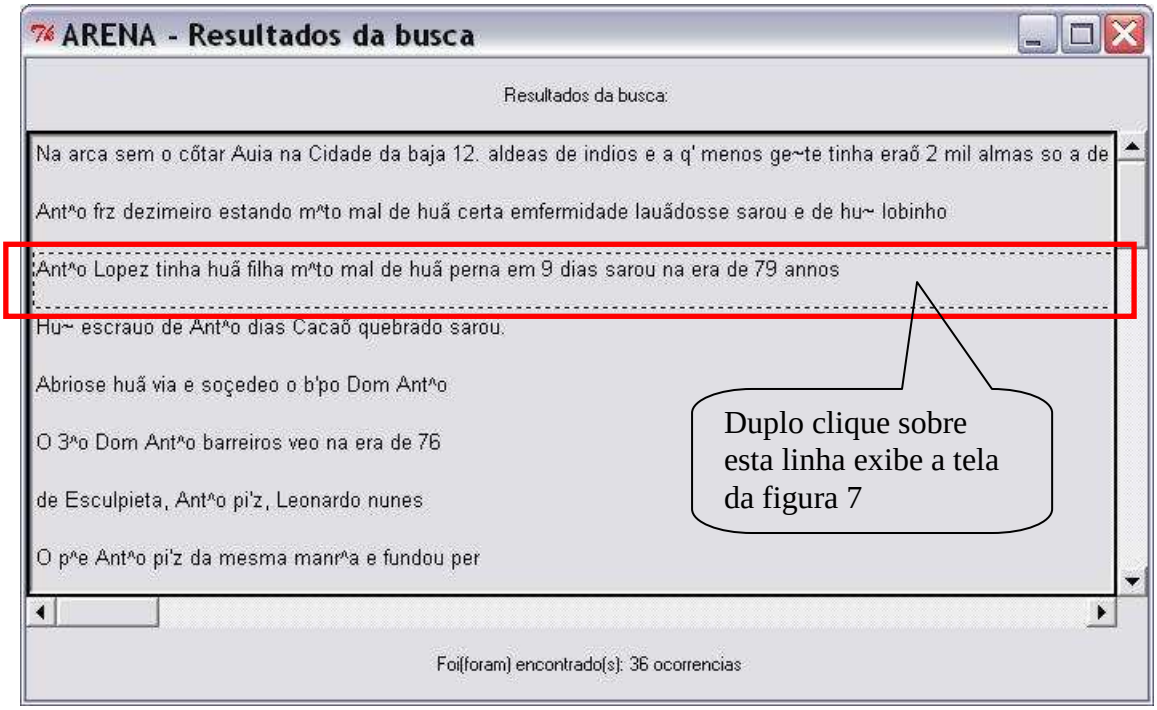

Figura 6: Tela que exibe o resultado da busca por um padrão

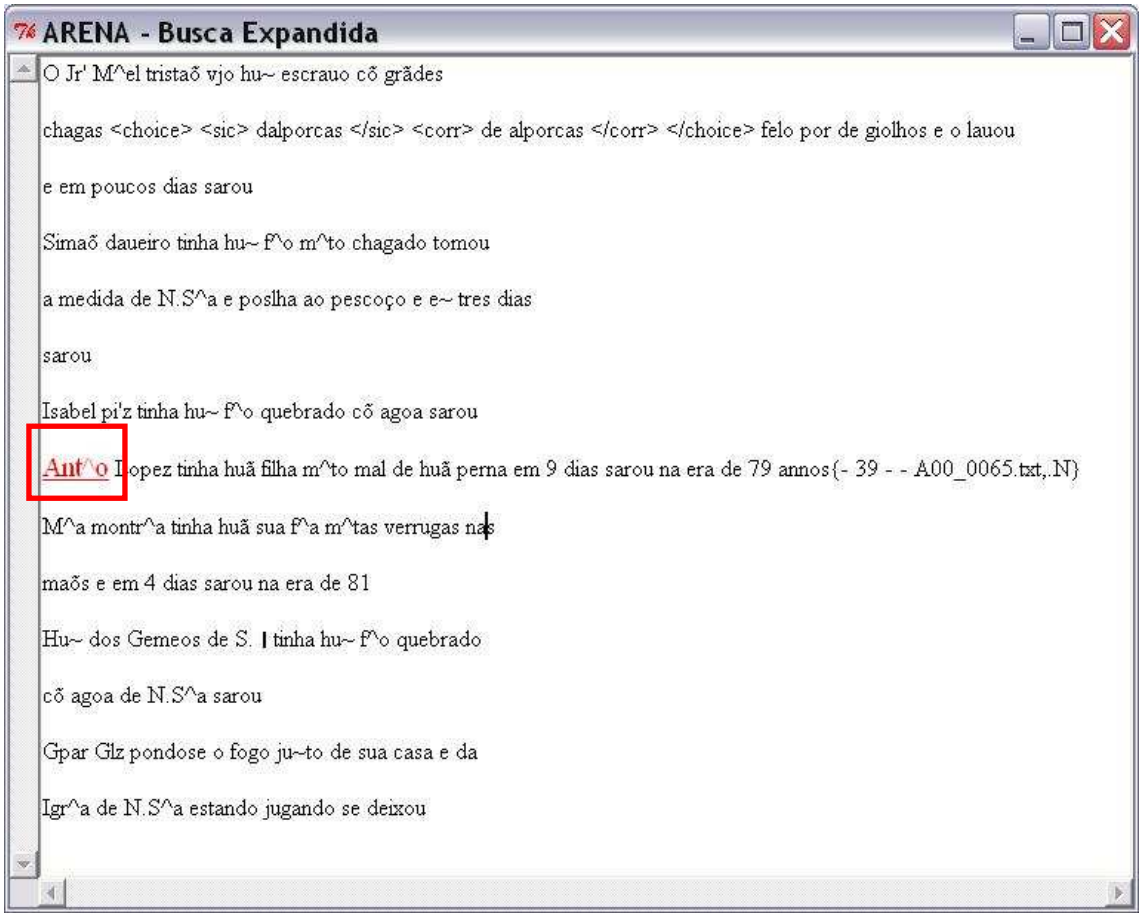

Figura 7: Tela que exibe as linhas adjacentes a aquela que possui o padrão

Se o resultado retornado se enquadrar nas características desejadas pelo usuário, a inserção dos dados da EN no repositório pode ser feita através de um duplo clique sobre o padrão em vermelho que aparece na tela da busca expandida (figura 7). A inserção no banco de dados ocorre depois do preenchimento do formulário que aparece na figura 8.

Figura 8: Formulário para inserção dos dados da EN abreviada no banco de dados do projeto

Caso o usuário queira digitar um padrão para a busca (terceira opção da Figura 4) a tela a seguir será exibida (figura 9).

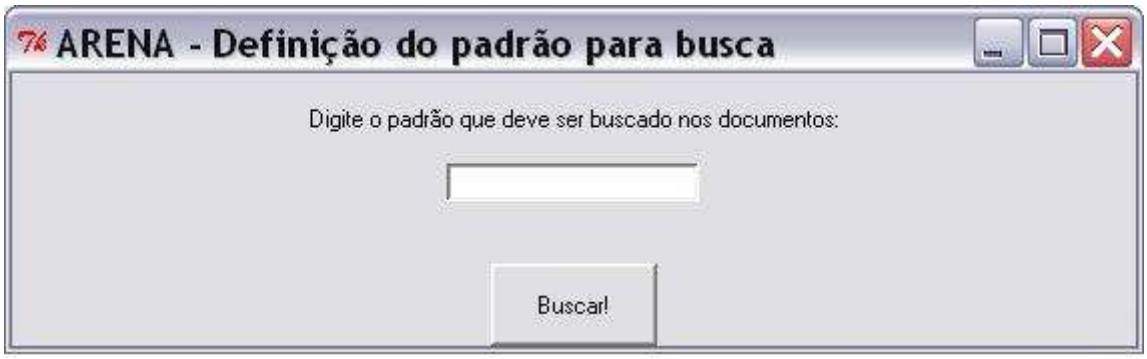

Figura 9: Busca por um padrão digitado pelo usuário

As linhas contendo o padrão investigado são apresentados em uma janela semelhante a figura 6. A partir desse ponto, a seqüência de apresentação dos resultados é a mesma que a explicada anteriormente para a procura por abreviações com rótulo “ENT” ou “INIT” na planilha Excel.

Com o auxílio da ARENA, o repositório possui atualmente informações de 106 diferentes EN da categoria “local” e “pessoa”. A aplicação web (construída no projeto de Graduação I) que exibi as informações de EN armazenadas no repositório possui a seguinte interface:

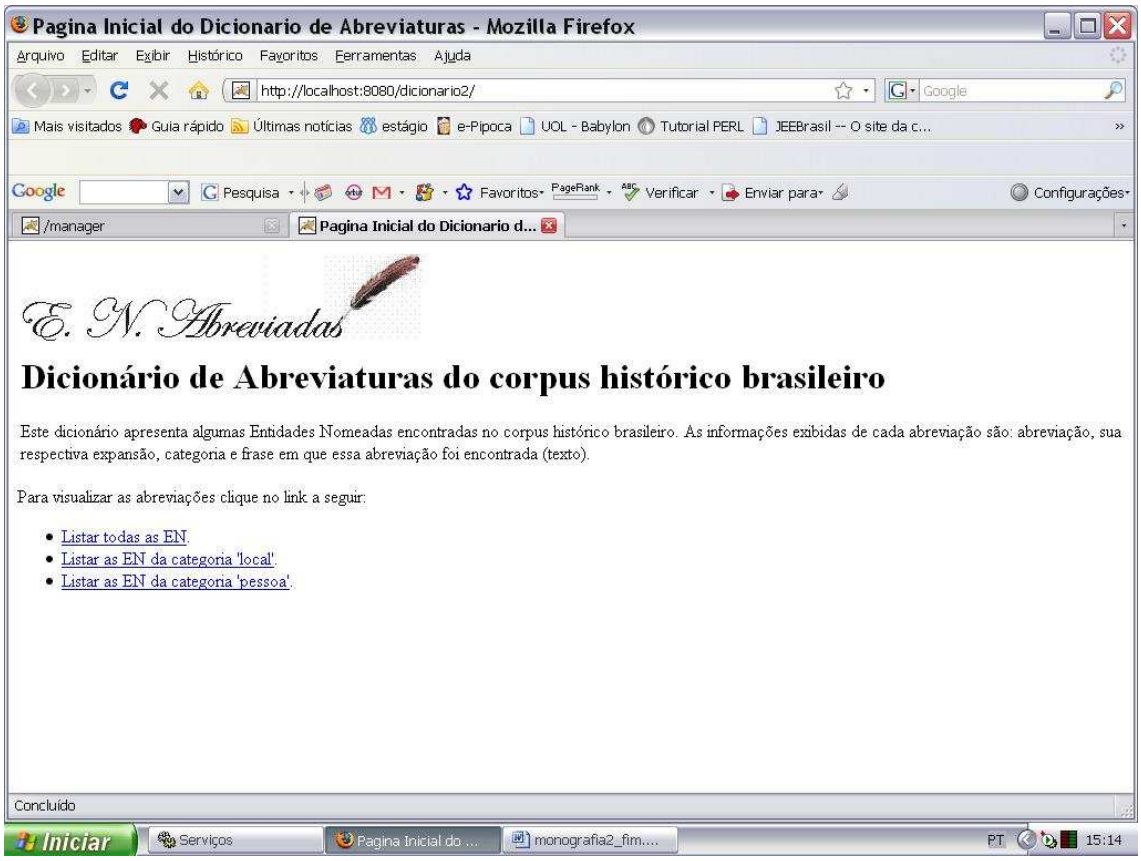

Figura 10: Página inicial da aplicação web

Figura 11: Exibição das informações relativas a EN da categoria "local"

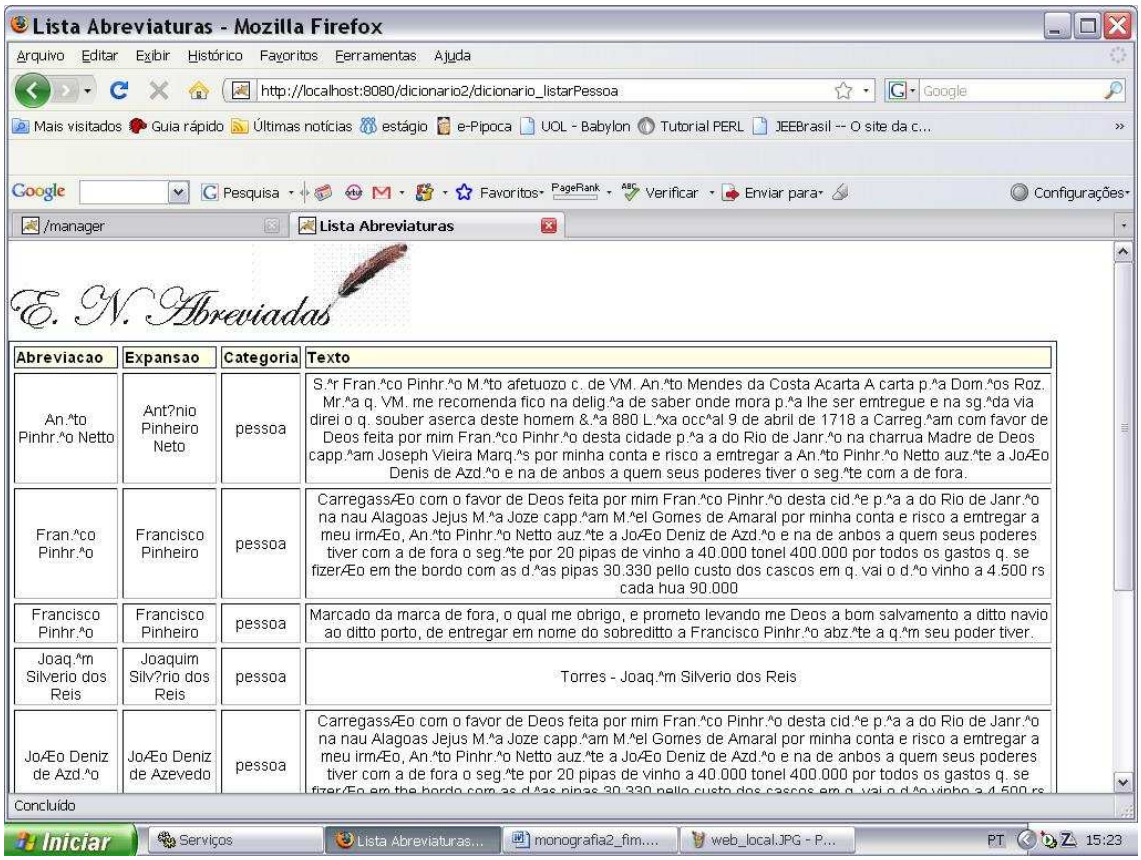

Figura 12: Exibição das informações relativas a EN da categoria "pessoa"

4. Conclusões e trabalhos futuros

O estudo realizado durante as disciplinas de Projeto de Graduação I e II foi muito importante porque abordou um assunto que não consta na grade curricular do curso de graduação em Ciências da Computação, além de oferecer a oportunidade de participar de um projeto tão amplo como é o DHPB. Mais especificamente, neste semestre, a implementação de uma aplicação baseada em uma linguagem sem conhecimento prévio foi uma tarefa muito enriquecedora em experiência e em ganho de informações. Trabalhar com os documentos do século XVI ao XIX é muito interessante, pois é uma ocasião de conhecer melhor a história do Brasil, além de apresentar algumas curiosidades da época em que eles foram escritos.

Existem algumas sugestões para aperfeiçoar a ARENA. Entre elas estão a introdução de um botão de ajuda nas telas da aplicação para auxiliar o usuário durante o uso da ferramenta, adicionar uma barra de progresso para informar o estado da busca do padrão nos documentos, tornar a busca mais rápida, adicionar uma janela para ler o caminho da planilha Excel que contem os EN e INIT abreviados e do arquivo que contem os textos do corpus, além da implementação de uma sub-rotina que retorna as variações de grafia de um padrão de busca.

Referências

ALUÍSIO, S. M. E CANDIDO JR., A. Córpus históricos para a construção de dicionários. A ser publicado no Livro Introdução à Lingüística Computacional: 1ª Escola Brasileira, 2008.

ARANHA, C. N. O Cortex e a sua participação no HAREM, 2007. Disponível no site: http://acdc.linguateca.pt/aval_conjunta/LivroHAREM/Cap09-SantosCardoso2007-Aranha.pdf

CANDIDO JR., A. Criação de um ambiente para o processamento de córpus de Português Histórico. Dissertação de Mestrado do ICMC-USP, 143 p., 2008.

CARDOSO, D. S. N. HAREM, a primeira avaliação conjunta de sistemas de reconhecimento de entidades mencionadas para português, 2007. Disponível em: http://xldb.di.fc.ul.pt/linguateca/prefacio.pdf

FLEXOR, MARIA HELENA M. O. Abreviaturas - Manuscritos dos Séculos XVI Ao XIX. 2nd ed. São Paulo: UNESP. 468 p., 1991.

GIUSTI, R.; CANDIDO JR, A.; MUNIZ, M. C. M.; CUCATTO, L. A.; ALUÍSIO, S. M. Automatic detection of spelling variation in historical corpus: An application to build a Brazilian Portuguese spelling variants dictionary. In: **Corpus Linguistics**, 2007, Londres. Corpus Linguistics, 2007.

HIROTSU, K. T. e ALUÍSIO, S.M. Criação de um grande repositório público de Entidades Nomeadas Abreviadas extraídas de um Corpus Histórico do Português do Brasil. Relatórios Técnicos do NILC – NILC-TR-XX-08, YY p., 2008.

SARMENTO, L. (2006). SIEMÊS: a named entity recognizer for Portuguese, In: Proceedings of PROPOR´2006, LNCS Volume 3960/2006, p. 90-99, 2006.

SARMENTO, L., PINTO, A. S., CABRAL, L. (2006). REPENTINO: A wide-scope gazetteer for Entity Recognition in Portuguese, In: Proceedings of PROPOR´2006, LNCS Volume 3960/2006, p. 31-40, 2006.

SOLORIO, T. MALINCHE: A NER system for Portuguese that reuses knowledge from Spanish. Reconhecimento de entidades mencionadas em português: Documentação e actas do HAREM, a primeira avaliação conjunta na área, Capítulo 10, p. 123–136, 2007. Disponível em http://acdc.linguateca.pt/aval_conjunta/LivroHAREM/Cap10-SantosCardoso2007-Solorio.pdf

STITSON, M.O., WESTON, J.A.E., GAMMERMAN, A., VOVK, V., VAPNIK, V. Theory of Support Vector Machines (Dept. Comp. Sci. Tech. Rep. CSD-TR-96-17; London: Univ. London Royal Holloway College), 1996

VALE, O.; CANDIDO Jr. A.; MUNIZ, M. ;BENGTSON, C.; CUCATTO, L.; ALMEIDA, G.;BATISTA, A.;PARREIRA, M.C.; BIDERMAN, M.T. ALUÍSIO, S. Building a large dictionary of abbreviations for named entity recognition in Portuguese historical corpora. In the Proceedings of the LREC 2008 Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2008), p. 1-10.

VAPNIK, V. N. The Nature of Statistical Learning Theory (New York: Springer), 1995.

Apêndice

Código-fonte da ARENA

```
#--------------------------------pacotes necessários
use Tk;
use DBI;
use Spreadsheet::ParseExcel;
require Tk::TList;
#-----------------------------------------------------------------

#caminho dos arquivos
$caminho_excel = 'C:\Documents and Settings\All
Users\Documentos\ProjetoII\en_init_manipulado.xls';
$caminho_arquivoTxt = 'C:\Documents and Settings\All
Users\Documentos\ProjetoII\doc1.txt';

#categorias do HAREM
my @categorias = qw/LOCAL PESSOA TRATAMENTO TEMPO ACONTECIMENTO AMB
ORGANIZAÇÃO VALOR PARENTE COISA CARGO TÍTULO/;

#sub-rotina que constrói a tela com a escolha da categoria
$menuInicio = MainWindow->new;
$menuInicio->title("ARENA - Menu inicial");
$menuInicio->Label(-text=> "\n Passo 1: \n",
      -font => '9x15bold')->place(-rely => 0.2, -anchor => "center",
      -relx => 0.5);
$menuInicio->Label(-text=> "       Escolha uma das categorias a baixo
      ")->place(-rely => 0.4, -anchor => "center", -relx => 0.5);

#menu que mostra as opções de categorias
my $opcaoCategoria = $menuInicio->Optionmenu(
      #variável guarda o valor corrente da opção do menu
      -variable => \$categoria_escolhida,
      -options  => [@categorias],
      #comando é o callback
      -command  => [sub {print $tvar."\n"}, 'Categoria'],
      -textvariable => \$tvar
      );
$opcaoCategoria->place(-rely => 0.8, -anchor => "center",
      -relx => 0.3);
my $botaoOkCategoria = $menuInicio->Button(-text=>"Ok",
      -padx => 20, -pady => 10,
      -command => sub{
            menu_busca($categoria_escolhida);
      })->place(-rely => 0.8, -anchor => "center", -relx => 0.8);
#-----------------------------------------------------------------------
#loop
MainLoop;
#---------------------------------SUBROTINAS
#sub-rotina que abre e manipula a planilha Excel
sub abertura_arquivo{
      my $EN_INIT = $_[0];
      my $categoria_escolhida = $_[1];
      my $arquivoExcel;
      my @vetor_en;
      $arquivoExcel = Spreadsheet::ParseExcel::Workbook-
>Parse($caminho_excel);
```

```
      my $cont = 0;

      #para cada linha do arquivo xls
      foreach my $folha (@{$arquivoExcel->{Worksheet}})
      {
            #para cada linha = folha  mínima, se o valor máximo da
#linha estiver definida e a linha não for maior do que a linha máxima,
#FAÇA
            for (my $iR = $folha->{MinRow}; defined $folha->{MaxRow} &&
$iR <= $folha->{MaxRow}; $iR++)
            {
                  my $categoria_planilha = $folha->{Cells}[$iR][7]-
>Value;
                  if(($categoria_escolhida eq $folha->{Cells}[$iR][7]-
>Value && $folha->{Cells}[$iR][7]->Value ne "" && $folha-
>{Cells}[$iR][6]->Value eq $EN_INIT) || ($categoria_planilha =~
/.*$categoria_escolhida.*/ && $folha->{Cells}[$iR][7]->Value ne "" &&
$folha->{Cells}[$iR][6]->Value eq $EN_INIT)){
                        $cont++;
                        my $padrao = validandoPadrao($folha-
>{Cells}[$iR][1]->Value);
                        push(@vetor_en, $padrao);
                  }
            }
      }
      return @vetor_en; #retorna um vetor com as abreviações da
planilha
}

#sub-rotina que constrói a tela que contem as opções de busca de EN
sub menu_busca(){
      my $categoria = $_[0];
      $menuInicio->DESTROY;
      $menuBusca = MainWindow->new;
      $menuBusca->title("ARENA - Menu opção busca");
      $menuBusca->Label(-text=> "\n Passo 2: \n",
            -font => '9x15bold')->pack(-side =>'top');
      $menuBusca->Label(-text=> "\n Escolha uma das opcoes de para
recuperar EN da categoria selecionada \n")->pack(-side =>'top');

      my $opcaoBuscaBD = $menuBusca->Radiobutton(-text => "Buscar por
EN presentes no Dicionário Flexor anotado com EN do HAREM da categoria
escolhida",
            -variable => \$busca_escolhida,
            -value => 'BD')->pack(-side =>'top');

      my $opcaoBuscaExcel = $menuBusca->Radiobutton(-text => "Buscar
por INIT presentes no Dicionário Flexor anotado com EN do HAREM da
categoria escolhida",
                  -variable => \$busca_escolhida,
                  -value => 'planilha')->pack(-side =>'top');

      my $opcaoBuscaPadrao = $menuBusca->Radiobutton(-text => "Digitar
um padrao para busca",
                  -variable => \$busca_escolhida,
                  -value => 'padrao')->pack(-side =>'top');

      my $botaoOkBusca = $menuBusca->Button(-text=>"Ok",
                  -padx => 20, -pady => 10,
                  -command => sub{
```

```
                        print "\n\n busca escolhida foi:
".$busca_escolhida;
                        print "\n categoria: ".$categoria;
                        if ($busca_escolhida eq "padrao"){
                              tela_busca_padrao($categoria);
                        } else{
                              if($busca_escolhida eq "planilha"){
                                    @vetor = abertura_arquivo("INIT",
$categoria);
                              } else {
                                    @vetor = abertura_arquivo("ENT",
$categoria);
                              }
                        $menuBusca->DESTROY;
                        $mw = MainWindow->new;
                        mw->title("ARENA - abreviações encontradas");
                        $mw->Label(-text=> "\n Para realizar a busca
de uma das abreviações de um duplo clique em cima dela: \n",
                              -font => '9x15bold')->pack(-side
=>'top');
                        $mw->minsize(700,400);
                        my $tl = $mw->Scrolled("TList", -font =>
['Arial', '12'], -command => sub{
                                 my ($index) = @_;
                                 tela_resultado_busca("planilha",
@vetor[$index], $categoria)
                              })->pack(-fill => 'both', -expand => 1);
                              foreach(@vetor){
                                    $tl->insert('end', -itemtype =>
'text',
                                          -text => $_);
                              }
                           }
      })->pack(-side =>'bottom');
}

#sub-rotina que constrói a tela que solicita ao usuário a digitação do
padrão a ser procurado
sub tela_busca_padrao(){
      my $categoria = $_[0];

      $menuBusca->DESTROY;
      $tela_busca_padrao = MainWindow->new;
      $tela_busca_padrao->title("ARENA - Definição do padrão para
busca");
      $tela_busca_padrao->Label(-text=> "\n Digite o padrão que deve
ser buscado nos documentos: \n")->pack(-side =>'top');

      my $caixaTexto = $tela_busca_padrao->Entry( -textvariable =>
\$padrao )->pack(-side =>'top');

      my $botaoBusca = $tela_busca_padrao->Button(-text=>"Buscar!",
            -padx => 20, -pady => 10,
            -command => sub{
                  print "padrao: ".$padrao."\n";
                  $padrao = validandoPadrao($padrao);
                  tela_resultado_busca("padrão", $padrao, $categoria);
                  })->pack(-side =>'bottom');

}
```

```
#sub-rotina que contem a chamada para a rotina que constrói a tela que
exibi as linhas com o padrão procurado
sub tela_resultado_busca(){
      my $aux=$_[0];#verifica qual é o tipo de busca
      my $categoria= $_[2]; #terceiro parâmetro é a categoria que deve
ser buscada
      if($aux eq "padrão"){
            $tela_busca_padrao->DESTROY;
      }
      my $palavra= $_[1];#segundo parâmetro é a palavra que deve ser
buscada
      busca_palavra($palavra, $categoria, $aux);
}

#sub-rotina que constroi a tela que exibe todas as linhas que contem o
padrão procurado
sub busca_palavra(){
      my $palavra=$_[0];
      my $categoria=$_[1];
      my $tipo_busca = $_[2];
      open(arquivoTxt, "< $caminho_arquivoTxt");

      $menuBusca = MainWindow->new;
      $menuBusca->minsize(700,400);
      $menuBusca->title("ARENA - Resultados da busca");
      $menuBusca->Label(-text=> "\n Resultados da busca: \n")->pack(-
side =>'top');

      my $linha = 0;
      my @vetorEN;
      foreach $var (<arquivoTxt>){
            if($var =~/(.*)($palavra)(.*)/i){ #o problema é que lê
linha por linha
                  $cont++;
                  push @vetorEN, $linha;
            }
            $linha++;
      }
      close(arquivoTxt);

      my $tl = $menuBusca->Scrolled("TList", -orient=> 'horizontal' ,-
scrollbars => 'se', -width => 50, -height => 12,-font => ['Arial',
'10'], -command => sub{
                  my ($index) = @_;
                  $nroLinha = @vetorEN[$index];
                  mostra_texto($palavra,$nroLinha, $nroLinha-7,
$nroLinha+7);
                              })->pack(-fill => 'both', -expand => 1);

      open(arquivoTxt, "< $caminho_arquivoTxt");
      my @arquivo = <arquivoTxt>;
      my $posicaoVetor=0; #comeca mostrando a primeira linha
encontrada

      foreach $var (@vetorEN){
            $tl->insert('end', -itemtype => 'text',-text =>
@arquivo[$var]);
      }
      close(arquivoTxt);
      $menuBusca->Label(-text=> "\n Foi(foram) encontrado(s): $cont
ocorrencias\n")->pack(-side =>'bottom');
```

```
}

#tela que exibe a busca expandida, ou seja, 7 linhas antes e 7 linhas
#depois da linha que contem o padrão
sub mostra_texto(){
      $palavra = $_[0];
      $numeroLinha = $_[1];
      $inicio = $_[2];
      $fim = $_[3];

      $menuBusca = MainWindow->new;
      $menuBusca->title("ARENA - Busca Expandida");

      my $text = $menuBusca->Scrolled("Text", -wrap => 'none')
      ->pack(-expand => 1, -fill => 'both');

      $text->tagConfigure('nao', -font => [-family =>'Times',
                        -size =>'12',
                        -underline  => 'false']);

      $text->tagConfigure('abrev', -font => [-family =>'Times',
            -size =>'14',
            -underline  => 'true'], -foreground => 'red');

      $text->tagBind('abrev', "<Button-1>", sub {
                              #contrução da tela com o formulário de
inserção de dados da EN
                              $tela_insercao = MainWindow->new;
                              $tela_insercao->title("Insercao no Banco
de dados");
                              $tela_insercao->Label(-text=> "\n
Formulário para inserção no banco de dados \n")->pack(-side =>'top');

                              $tela_insercao->Label(-text=> "\n
Abreviação: \n")->place(-rely => 0.1, -anchor => "nw", -relx => 0.1);
                              $campoAbrev = $tela_insercao->Entry(-
textvariable => \$abrev,  -width => 30)->place(-rely => 0.13, -anchor
=> "nw", -relx => 0.3); # create Entry

                              $tela_insercao->Label(-text=> "\n
Expansão: \n")->place(-rely => 0.2, -anchor => "nw", -relx => 0.1);
                              $campoExp = $tela_insercao->Entry(-
textvariable => \$exp, -width => 60)->place(-rely => 0.23, -anchor =>
"nw", -relx => 0.3);

                              $tela_insercao->Label(-text=> "\n
Categoria: \n")->place(-rely => 0.3, -anchor => "nw", -relx => 0.1);
                              $campoExp = $tela_insercao->Entry(-
textvariable => \$cat, -width => 10)->place(-rely => 0.33, -anchor =>
"nw", -relx => 0.3);
                              $campoExp->insert('end', $categoria);

                              $tela_insercao->Label(-text=> "\n Oração
exemplo do corpus: \n")->place(-rely => 0.4, -anchor => "nw", -relx =>
0.1);
                              $scrollbar = $tela_insercao->Scrollbar(-
orient => 'horizontal');
                              $campoOracao = $tela_insercao->Entry(-
textvariable => \$oracao, -width => 60, -xscrollcommand => ['set' =>
$scrollbar]);
```

```
                              $scrollbar->configure(-command =>
['xview' => $campoOracao]);
                              $scrollbar->place(-relwidth=> 0.5, -rely
=> 0.5, -anchor => "nw", -relx => 0.3);
                              $campoOracao->place(-rely => 0.43, -
anchor => "nw", -relx => 0.3);

                              $tela_insercao->Button(-text=>"Inserir!",
                                    -padx => 20, -pady => 10,
                                    -command => sub{
                                          print "\nAbreviação inserida:
".$abrev."\n";
                                          if ($tipo_busca eq "padrão"){

      insereBD($abrev,$exp,$cat,$oracao,$palavra);
                                          }
                                          else{

      insereBD($abrev,$exp,$cat,$oracao,"xxx");
                                          }
                                    })->place(-rely => 0.9, -anchor =>
"s", -relx => 0.5);
                  });

      open(arquivoTxt, "< $caminho_arquivoTxt");
      my @arquivo = <arquivoTxt>;

      my $linhaAtual;
      foreach $linhaAtual ($inicio..$fim){
                  if(@arquivo[$linhaAtual] =~/(.*)($palavra)(.*)/i){
                        $text->insert('end', $1, 'nao');
                        $text->insert('end', $2, 'abrev');
                        $text->insert('end', $3, 'nao');
                  }
                  else{
                        $text->insert('end',
@arquivo[$linhaAtual]."\n", 'nao');
                  }
      }
      close(arquivoTxt);
}

#sub-rotina para inserir dados de uma EN no banco de dados
sub insereBD(){
            $abreviacao = $_[0];
            $expansao = $_[1];
            $categoria = $_[2];
            $oracao = $_[3];
            $padrao = $_[4]; #não é mais utilizado

            $database = "dicionario";
            $host = "localhost";
            $usuario = "usuario";
            $senha = "1234";
            my $dbh = DBI-
>connect("DBI:mysqlPP:$database;host=$host","$usuario",
"$senha",{'RaiseError' => 1});
            $dbh->do("INSERT INTO entidadeNomeada (abreviacao,
expansao, categoria, texto) VALUES ('$abreviacao',
'$expansao','$categoria','$oracao')");
```

```
            $tela_insercao->DESTROY;
}

#rotina para tornar o padrão válido para busca, uma vez que muitos
#padrões possuem “.” e “^” e estes símbolos possuem um significado
#próprio na busca por expressão regular
sub validandoPadrao(){
      my $padrao = $_[0];
      $padrao =~ s/\./\\\./g;
      $padrao =~ s/\^/\\\^/g;
      return $padrao;
}
```