UNIVERSIDADE DE SÃO PAULO
Núcleo Interinstitucional de Lingüística Computacional

siaconf

Relatório Técnico

Rafael Giusti
Orientadora: Profª Drª Sandra Maria Aluísio

Índice

1. Introdução

Este Relatório Técnico descreve o **siaconf**, o Sistema de Apoio à Contagem de Freqüência em Corpus. O sistema em questão foi desenvolvido dentro do escopo do Projeto Milenio, com o objetivo de apoiar a etapa de contagem de freqüência de palavras, a qual é bastante prejudicada pela variação de grafia inerente ao português brasileiro histórico.

O sistema foi desenvolvido em GNU/Linux na linguagem de programação PERL. O uso de expressões regulares é intenso, tanto para programadores que desejem modificar o sistema quanto para usuários do sistema. O anexo 1 contém uma muito breve descrição de expressões regulares que deve ser suficiente para somente utilizar o sistema.

No que diz respeito a internacionalização, o **siaconf** utiliza o português como interface e codifica texto em Unicode/UTF-8. O sistema espera ser executado em terminais com codificação UTF-8, de forma que a interface pode ser visualizada de forma estranha em terminais que utilizem outra codificação. É possível também que algumas tarefas do sistema sejam prejudicadas, mas isso é bem menos provável.

O resto deste Relatório Técnico está organizado como se segue. Na Seção 2 é descrita a estrutura do sistema e introduzidos as ferramentas que o compõem. A Seção 3 explica como utilizar o **siaconf** para realizar várias tarefas de interesse. Cada uma das ferramentas que compem o **siaconf** sao descritas nesta Seçao.

2. Estrutura

O **siaconf** é composto de várias ferramentas que desempenham papéis distintos. Sucintamente, essas ferramentas são implementadas pelos seguintes programas:

- **importar.plx**: utilizado para adicionar um corpus ao sistema. O programa pode reunir vários arquivos de texto como um único corpus e ainda tentar codificar os arquivos para UTF-8, mas essa tarefa é insegura. A adição de corpora é explicada na Seção 3.1.
- **alfabetizar.plx**: este programa auxilia a definição do alfabeto utilizado no corpus. Ele busca o texto por caracteres não-listados no alfabeto do corpus. O uso deste programa é explicado na Seção 3.2.
- **preparar.plx**: remove a formatação do corpus e os símbolos não-importantes. O uso deste programa é explicado na Seção 3.3.
- **regras.plx**: este programa aplica as regras no corpus e gera os relatórios. O uso deste programa é explicado na Seção 3.4. A adição de regras é explicada na Seção 3.4.1. Os relatórios são explicados na Seção 3.4.2.

Cada corpus é mantido num diretório separado. Esse diretório e o arquivo do corpus são criados automaticamente pelo sistema no momento em que o corpus é importado. Arquivos de configuração (alfabeto e regras) podem ser mantidos no diretório do corpus (específico) ou no diretório raiz do **siaconf** (global).

3. Utilização do sistema

A invocação dos programas que compõem o sistema deve ser feita por linha de comando. Alguns programas necessitam argumentos para operar. O **siaconf** depende de várias ferramentas nativas do SO GNU/Linux. Portanto, é recomendável que o **siaconf** seja utilizado nesse sistema. No entanto, caso as ferramentas necessárias sejam instaladas em outro SO, é possível que o **siaconf** funcione corretamente nesse ambiente.

Uma utilização corriqueira do siaconf consiste em:

1. Importar um novo corpus com a ferramenta `importar.plx`.
2. Definir um arquivo de alfabeto inicial.
3. Incrementar o alfabeto, se necessário, com a ferramenta `alfabetizar.plx`.
4. Remover a formatação do corpus com a ferramenta `preparar.plx`.
5. Definir um conjunto inicial de regras.
6. Aplicar as regras com a ferramenta `regras.plx`.
7. Analisar resultados experimentais e refinar regras, repetindo os passos 6 e 7 indefinidamente.

Logicamente, nem todos os passos devem ser sempre executados. Essa é apenas uma descrição genérica para uso rápido do **`siaconf`**.

3.1. Importação de corpora

O **`siaconf`** é capaz de trabalhar com vários corpora independentes, utilizando para cada um deles um conjunto de arquivos de configuração distinto ou utilizando o mesmo conjunto de arquivos de configuração de experimento para todos os corpora instalados.

A adição de um corpora deve ser feita pela criação de um diretório no diretório raiz do sistema **`siaconf`**. O corpus deve ser armazeado num único arquivo de texto puro, com a codificação UTF-8. Um arquivo de nome *alfabeto* (Seção 3.2) e um arquivo de nome *regras* (Seção 3.4) podem ser criados no diretório do corpus.

Para auxiliar a adição de corpora, existe a ferramenta importar.plx. Essa ferramenta serve para converter arquivos para a codificação UTF-8, concatenar vários arquivos num único arquivo de texto e construir a estrutura de diretórios para o novo corpus.

A sintaxe da ferramenta de importação é a seguinte:

```
./importar.plx <nome> <arquivos>
```

Sendo que `<nome>` é o nome do novo corpus e `<arquivos>` é uma que deve conter os nomes dos arquivos a serem concatenados para compor o corpus.

Num primeiro momento, a ferramenta analisa a lista de `<arquivos>`, checando se todos os nomes fornecidos conferem com arquivos ou diretórios acessíveis. Caso um dos nomes da lista seja um diretório, o programa irá acessar o diretório e processar todos os seus arquivos recursivamente.

Os arquivos de texto são concatenados na ordem em que os seus arquivos são encontrados. Quando os arquivos tem seus nomes listados na lista de `<arquivos>`, a ordem é especificada pelos argumentos da linha de comando. Quando os arquivos são encontrados pela busca recursiva, a ordem é determinada pelo *shell* – normalmente, no UNIX, essa ordem é alfabética.

A ferramenta de importação tenta utilizar o aplicativo `file` para reconhecer a codificação dos arquivos. Em seguida, o sistema tenta realizar a conversão dos arquivos para UTF-8 através do aplicativo `iconv`. A eficiência do processo depende da capacidade de `file` em reconhecer de forma correta a codificação dos arquivos original. A confiabilidade é bastante alta em se tratando de arquivos com codificação retrocompatível com ASCII, como ISO-8859-9 (latino, árabe) e UTF-8. No entanto, o aplicativo `file` pode ser enganado por outras codificações, como UTF-16. Se você souber que um ou mais dos seus arquivos possui essa codificação, prefira convertê-los manualmente com o aplicativo `iconv` ou outro software de sua preferência (Word, Gnome Editor etc).

Após executada, a ferramenta importar.plx cria, no diretório `<corpus>` um arquivo de nome `<corpus>-sujo.txt`. A "limpza" deve ser feita pela ferramenta `preparar.plx`.

3.2. Definição do alfabeto

No que concerne o **`siaconf`**, o português histórico é um idioma diferente do português

falado atualmente no Brasil: ele não conhece o conjunto de caracteres utilizado no corpus e, menos ainda, as regras para convertê-los de caixa alta para caixa baixa ou para ordenar o texto. Por isso, é necessário que você ensine a ele como realizar essas tarefas.

O programa irá procurar, no diretório do corpus, pelo arquivo de nome `alfa`. Caso esse arquivo não seja encontrado, o sistema irá procurar por ele no diretório "raiz" do **`siaconf`**. Isso permite que você mantenha um alfabeto global para todos os corpora do idioma que utiliza predominantemente e defina alfabetos específicos a alguns corpora. No entanto, se nem o alfabeto global, nem o específico forem encontrados, o sistema não poderá operar de forma correta.

O objetivo do arquivo `alfa` é fornecer ao sistema informações sobre o conjunto de símbolos (caracteres) do corpus sob análise. Ele é composto pelos campos "regras de caixa baixa", "ordenação" e "símbolos a ignorar".

O campo "regras de caixa baixa" inicia com uma linha contendo o símbolo `::MINIMIZAR`. Cada linha que segue deve ser um par caixa alta/caixa baixa, como "A a", "B b", "Á á" ou "Ç ç". Quando o sistema precisar minimizar palavras, ele irá utilizar esses pares para efetuar substituição de símbolos no texto. Símbolos que não devem ser alterados podem ser omitidos.

O campo "ordenação" inicia com o símbolo `::ORDEM`. Esse campo deve conter todos as letras do alfabeto de forma ordenada. Se um símbolo A aparece numa linha anterior a de um símbolo B, o sistema ordena A,B. Se um símbolo A aparece na mesma linha que um símbolo B, o sistema não especifica ordem, e ambas as ordens A,B e B,A são aceitáveis. Símbolos que não aparecem no campo ordenação são desconsiderados no processo de ordenação.

O campo "separadores" inicia com o símbolo `::QUEBRAR`. Os símbolos contidos neste campo são utilizados para "quebrar" uma linha. Isto é, esses símbolos são delimitadores de palavras. Os símbolos listados no campo "separadores" são também removidos do texto.

O campo "limpar" inicia com o símbolo `::LIMPAR`. Após a separação, todos os símbolos listados nesse campo são removidos das palavras.

O campo "ignorar" inicia com o símbolo `::IGNORAR`. Os simbolos contidos neste campo são ignorados pelo sistema na busca por símbolos e na ordenação.

Caracteres Unicode podem ser especificados através de seus pontos de entrada ao invés do caracter em si. Por exemplo, o caracter 'A' tem ponto de entrada *u0065*. Ele pode ser representado, no arquivo de alfabeto, como a seqüência *\u{0065}*.

O símbolo `#` é utilizado para definir um comentário. Todo conteúdo da linha que apareça após o símbolo `#` é ignorado. Se necessário, o símbolo `#` pode ser representado no alfabeto como a seqüência Unicode *\u{0035}*.

O símbolo de espaço é reconhecido pelo sistema como indentação. Para adicionar o espaço ao alfabeto (por exemplo, como delimitador), deve-se utilizar a seqüência de escape `\s`. A seqüência de escape `\t` pode ser utiilizada para representar símbolos de tabulação (TAB). Uma nova linha é um delimitador padrão.

A seguir, um exemplo de arquivo de alfabeto:

```
# Arquivo de alfabeto para vogais não-acentuadas

::MINIMIZAR
A a
E e
I i
O o
U u

::ORDEM
A a
E e
I i
O o
```

```
U u

::IGNORAR
\u{0035}
[bcdfghjklmnpqrstvwxyz]
[BCDFGHJKLMNPQRSTVWXYZ]

::QUEBRAR
\s
'
.
```

Em trabalhos incrementais, o conjunto de símbolos do alfabeto pode necessitar alterações conforme novos textos vão sendo adicionados ao corpus. Para isso, existe a ferramenta `alfabetizar.plx`. Ela tem a finalidade de auxiliar na detecção de novos símbolos.

A sintaxe da ferramenta é a seguinte:

```
./alfabetizar.plx <corpus>
```

Sendo que `<corpus>` é o nome do corpus em questão.

A ferramenta analisa todo o conteúdo do corpus, exibindo as linhas nas quais símbolos desconhecidos são encontrados. A ferramenta mostra um excerto do corpus que contém o símbolo novo e fornece o ponto de entrada Unicode do símbolo. A adição do símbolo ao alfabeto deve ser feita pela edição do arquivo `alfa`. Não é necessário utilizar todos os campos. A ordem em que os campos se apresentam é irrelevante.

3.3. Preparo do corpus

Uma vez que o corpus tenha sido importado e seu alfabeto esteja definido, é necessário prepará-lo para aplicação das regras. A ferramenta `preparar.plx` utiliza o arquivo de alfabeto para quebrar o corpus em palavras, gerando um arquivo que contém uma única palavra por linha.

A sintaxe desta ferramenta é a seguinte:

```
./preparar.plx <corpus>
```

O arquivo gerado pode também ser utilizado para a contagem de palavras ortográficas do corpus.

3.4. Especificação de regras

O **`siaconf`** trabalha detecta variação de grafia aplicando substituição de cadeias de caracteres no corpus. Quando duas palavras distintas são convertidas, através de regras de transformação, para uma grafia comum, o sistema forma um agrupamento.

Regras de transformação são explicadas no Anexo 1. Esta Seção trata da especificação de regras para o sistema.

O **`siaconf`** carrega as regras preferencialmente do arquivo `regras` localizado no diretório do corpus. Caso esse arquivo não seja encontrado, ele é procurado no diretório "raiz" do **`siaconf`**. Isso permite especificar um conjunto de regras genérico para a maioria dos corpora e um conjunto específico para alguns corpora. Entretatnto, se o arquivo `regras` não for encontrado em nenhum dos diretórios, o sistema não poderá operar corretamente.

Um arquivo de regras é uma lista de regras ordenadas. Essa é uma característica importante: algumas regras podem fazer substituições que afetam a aplicação de outras regras.

Como extensão à definição de regras dada no Anexo 1, o **`siaconf`** permite as expressões regularesm sejam definidas entre aspas. Isso permite substituições em "palavras" que contêm

espaços como caracteres. Útil se o alfabeto está configurado para quebrar o texto em padrões que contêm espaços.

Um exemplo de regras de transformação:

```
y y i
[^r][ãa]o .o am
“gr[aã]o rei” .+ grande-rei
```

3.5. Aplicação de regras

A aplicação de regras é feita pela ferramenta regras.plx. Sua sintaxe é a seguinte:

```
./regras.plx <corpus>
```

Onde `<corpus>` é o nome do corpus sobre o qual se deseja aplicar as regras.

A aplicação de regras é um processo demorado e consome grande quantiade de memória principal. É necessário testar todas as regras pelo menos uma vez contra cada palavra do corpus (certas regras são capazes de fazer várias alterações em uma mesma palavra) e ordenar o conjunto de palavras únicas por freqüência.

O **`siaconf`** utiliza uma léxico para apoiar a aplicação de regras. O léxico é carregado na memória principal e utilizado pela ferramentas regras.plx para diversas verificações. Por exemplo, palavras que constam do léxico não são transformadas pelas regras. Ao final do experimento, palavras transformadas em grafias presentes no léxico (ortografia) são agrupadas com a ortografia.

O arquivo do léxico deve ser uma lista de palavras contendo uma palavra por linha. Repetições no léxico não influenciam a transformação de palavras do corpus. O **`siaconf`** espera encontrar o arquivo `lexico` no diretório do corpus ou no diretório “raiz” do sistema.

3.6. Avaliação de relatórios

Após o término da aplicação das regras, a ferramenta `regras.plx` gera quatro relatórios que contêm uma grande quantidade de informação de apoio.

O relatório de transformações relizadas lista, em ordem de ocorrência, todas as transformações relizadas pelo programa, mostrando também as regras aplicadas em cada caso, na ordem aplicada.

O relatório de agrupamentos lista, em ordem de freqüência, todos os agrupamentos formados pela transformação de palavras em grafias semelhantes.

O relatório de transformaçãoes por regra lista todas as transformações nas quais cada regra teve participação. Muito útil para avaliar a eficácia de uma regra em específico.

Finalmente, o relatório de grafias perdidas lista todas as palavras que não foram contempladas por nenhuma regra. Útil para a criação de novas regras e seleção de candidatos para lexicalização.