Concepção e Análise de Algoritmos

Troca de Letras

Ângela Cardoso e Bruno Madeira

1 de Junho de 2015

Conteúdo

1 Introdução **2**

2 Reconhecimento de Palavras **3**

3 Procura de Palavras **5**

4 Inteligências Artificiais **6**

5 Principais Dificuldades **7**

6 Distribuição do Trabalho **8**

7 Conclusão **9**

Capítulo 1

Introdução

No âmbito da disciplina de Concepção e Análise de Algoritmos do Mestrado Integrado em Engenharia Informática e Computação, foi-nos proposta a implementação de um jogo de formação de palavras.

É disponibilizada uma cadeia aleatória de letras. Alternadamente, cada um de dois jogadores pode: escolher duas letras da cadeia e trocar a sua posição e/ou retirar uma subsequência que seja uma palavra. As palavras podem ser formadas no sentido direto ou inverso. Por cada palavra que forme, o jogador ganha um número de pontos igual ao número de letras da palavra. No final, vence o jogador que tiver a maior pontuação, ou seja, aquele com o maior conjunto total de letras nas palavras que encontrou.

Para decidir quais as palavras aceitáveis, utiliza-se um dicionário. Para o efeito, escolhemos um ficheiro .dic com todas as palavras em inglês americano, organizadas por ordem crescente de tamanho e por ordem alfabética. A escolha da língua prendeu-se essencialmente com a inexistência de acentos ortográficos e com o facto de ser um dicionário mais pequeno, comparado por exemplo com o de português. No entanto, tudo funcionaria da mesma forma com um dicionário de outra língua, desde que se faça o devido tratamento de todos os caracteres especiais.

Neste documento descreve-se a forma como implementamos o jogo, desde o tipo de algoritmos utilizados até às inteligências artificiais que permitem que um único jogador defronte o computador em diversos níveis de dificuldade.

Capítulo 2

Reconhecimento de Palavras

Ainda que não fosse implementada qualquer inteligência artificial, para que o jogo funcionasse, seria sempre necessário determinar se uma determinada sequência de caracteres escolhida pelo jogador é ou não uma palavra, isto é, se faz parte do dicionário ou não. Neste aspeto o jogo funciona como um corretor ortográfico, embora não haja necessidade de sugerir alternativas quando não se trata de uma palavra.

Uma vez que apenas possuímos um ficheiro com a lista das palavras aceitáveis, a abordagem ingénua a este problema seria bastante lenta. De facto, sempre que fosse selecionada uma não palavra, um algoritmo de força bruta, percorreria toda a lista até determinar que não funciona. Neste caso, tratando-se de uma lista ordenada por tamanho, o algoritmo poderia parar assim que ultrapassasse o tamanho da não palavra. Mas ainda assim, para não palavras grandes seria um processo moroso. Em inglês, por exemplo, existem quase dez mil palavras com comprimento até cinco.

Por forma a acelerar este processo, decidimos construir uma árvore de prefixos para representar o dicionário. Basicamente, cada nó da árvore possui 26 filhos, correspondentes às 26 letras do alfabeto, ordenadas da forma convencional. Se não existe nenhuma palavra que comece com a sequência de letras obtidas percorrendo a árvore desde a raiz até um nó, então esse nó é nulo e não tem filhos. Por exemplo, começando com a letra 'a' existem várias palavras, então o nó 'a', filho da raiz, terá 26 filhos. Um desses filhos é também 'a'. Mas não existem palavras começadas por 'aa', logo este nó 'a' filho do primeiro 'a' é nulo e não tem filhos.

Percorrendo a árvore desde a raiz até uma folha, obtém-se uma palavra. No entanto, algumas palavras são prefixos de outras. Logo, cada nó tem também um Booleano que indica se nesse nó termina alguma palavra ou não. Também podemos ver a árvore de prefixos como um autómato finito e determinístico, em que a raiz é o estado inicial e todos os nós em que podemos terminar uma palavra são estados finais.

Para determinar se uma determinava sequência de caracteres é uma palavra, basta

percorrer a árvore seguindo a ordem dos caracteres da sequência. Ou seja, uma vez construída a árvore de prefixos, este processo é linear no número de letras da sequência. Ainda assim, é necessário construir a árvore, sendo esse processo linear no número de letras de todas as palavras do dicionário. Ou seja, é um processo relativamente lento.

Por forma a evitar reconstruir a árvore de cada vez que se inicia a aplicação, decidimos guardar a árvore construída num ficheiro. Para tal, usamos índices (em lugar de apontadores) para os filhos de cada nó. Dessa forma, podemos gravar os índices num ficheiro de texto, que é lido no início da aplicação. O processo de construção da árvore precisa de ser feito uma única vez e demora cerca de cinco minutos. A partir daí, a leitura do ficheiro é instantânea.

Capítulo 3

Procura de Palavras

Para criar uma inteligência artificial, isto é, um programa de computador que seja capaz de jogar o jogo, cumprindo as regras e preferencialmente sendo capaz de ganhar, é necessário criar algum algoritmo que seja capaz de reconhecer palavras na cadeia de caracteres.

Poderíamos verificar se cada subsequência da cadeia de caracteres é uma palavra. Mas da forma como temos o dicionário carregado na árvore, há um algoritmo mais eficiente. De facto, dado um caracter inicial, podemos percorrer a árvore ao longo do caminho sugerido pela cadeia e assim obter todas as palavras da cadeia que comecem com esse caracter. Depois basta repetir o processo para o caracter inicial seguinte.

Com este algoritmo simples, já poderíamos construir uma inteligência artificial muito básica. No entanto, não estaríamos a tirar qualquer proveito da possibilidade de trocar duas letras da cadeia antes de escolher uma palavra. Para isso, basta fazer uma alteração ao algoritmo de busca de palavras. Primeiro trocam-se duas letras distintas da cadeia, depois procuram-se palavras na cadeia, mas apenas aquelas que incluam uma das letras trocadas.

Com estes dois algoritmos, já é possível criar um programa capaz de jogar o jogo com todo o potencial. Apenas falta a capacidade de encontrar palavras que necessitem de mais do que uma troca. Isto é, seria necessária uma generalização daquilo que é feito quando se procuram palavras com uma troca. Naturalmente, este tipo de busca é factorial no número de letras distintas na sequência, pois é necessário efetuar todas as trocas possíveis para garantir que não há mais palavras. Embora tal busca se torne lenta em cadeias com mais de 10 letras, é útil como condição de término do jogo. Isto é, se a dada momento (com 10 letras ou menos), o computador determina que não há mais palavras, independentemente do número de trocas, o jogo pode terminar.

Capítulo 4

Inteligências Artificiais

Capítulo 5

Principais Dificuldades

Capítulo 6

Distribuição do Trabalho

Capítulo 7

Conclusão