Maximização de Lucro em Radares por Máquina de Turing

Bernardo Flores Salmeron Denilson das Mercês Amorim
Lucas Natanael Brito Prates Thalles Yan Santos Medrado

26/11/2018

1 Introdução

Máquina de Turing é um modelo matemático de computação definindo uma máquina abstrata operando em símbolos de uma fita de acordo com uma tabela de regras. Introduzido em 1936 por Alan Turing, o modelo é capaz de expressar qualquer algoritmo de computador.

Neste trabalho é apresentado uma máquina de Turing capaz de resolver o problema da maximização de lucro em radares. O problema foi adaptado do URI Online Judge[1].

2 Problema

Dado o lucro obtido ao se instalar um radar em cada ponto de uma rodovia, qual lucro máximo pode-se obter dado que quaisquer dois radares devem estar separados por no mínimo K posições?

2.1 Entrada

A entrada contém dois números N e K $(1 \leq K \leq N)$ — o tamanho da rodovia e a distância mínima entre dois radares, respectivamente. Em seguida, a entrada contem N números $lucro_1, lucro_2, \ldots, lucro_N$ $(lucro_i \geq 1)$ correspondentes ao lucro de instalação de um radar na i-ésima posição.

Os números da entrada são representados no sistema unário e são separados por um espaço em branco.

2.2 Saída

A saída deve ser a arrecadação máxima correspondente a melhor distribuição de radares.

2.3 Exemplo

Entrada	Saída
11111_11_111_11_11111_111_111	11111111111

3 Algoritmo

O problema apresenta subestrutura ótima e, portanto, pode ser resolvido por programação dinâmica. O subproblema ótimo usando na resolução de problemas maiores é: Qual melhor lucro pode-se obter (dada a restrição) até uma certa posição i $(0 \leq i \leq N)$ da rodovia?

A seguinte recorrência é capaz de solucionar o problema:

$$\mathrm{rad}_k(i) = \begin{cases} 0 & \text{se } i = 0, \\ \max \begin{cases} \mathrm{rad}_k(i-1) \\ \mathrm{rad}_k(\max(0, i-k)) + lucro_i \end{cases} & \text{caso contrário.} \end{cases}$$

Em palavras simples:

- Na posição zero (i.e. não existe rodovia), não é possível obter lucro.
- Nas demais posições, maximiza-se a escolher entre não inserir o radar i e manter o lucro até o momento, ou inserir o radar i e acumular o $lucro_i$ ao melhor lucro até a k-ésima posição anterior.

Essa recorrência pode ser convertida em um algoritmo iterativo com o auxílio de uma tabela de recorrência $dp[0...N]$.

1. Inicialize $dp[0]$ com 0.
2. Para cada i entre 1 e N (inclusivo), inicialize $dp[i]$ com o máximo entre $dp[i-1]$ e $dp[\max(0, i-k)] + lucro_i$.
3. Feito isso, a solução para o problema encontra-se em $dp[N]$.

4 Implementação

A implementação da máquina final foi dividida pela equipe em três etapas.

4.1 Abstração

Nesta etapa a equipe abstraiu a máquina final como um conjunto de Máquinas de Turing. Para isso, a implementação da máquina foi dividida em subproblemas de modo a simplificar a ideia e cada tarefa foi implementada individualmente em diferentes estados. Para esse problema, foi utilizada uma máquina de sete fitas e 35 estados. As fitas foram distribuídas da seguinte maneira:

1. Fita I/O: Recebe o valor de entrada e imprime o valor de saída
2. Fita N: Contém o tamanho da rodovia
3. Fita K: Contém a distância máxima permitida entre rodovias
4. Fita I: Controla o iterador I do laço utilizado na solução
5. Fita DP: Contém o vetor DP e seus respectivos valores em ordem crescente de índice
6. Fita X1: Fita auxiliar para realizar operações
7. Fita X2: Fita auxiliar para realizar operações

Um diagrama da maquina pode ser encontrado na Figura 1.

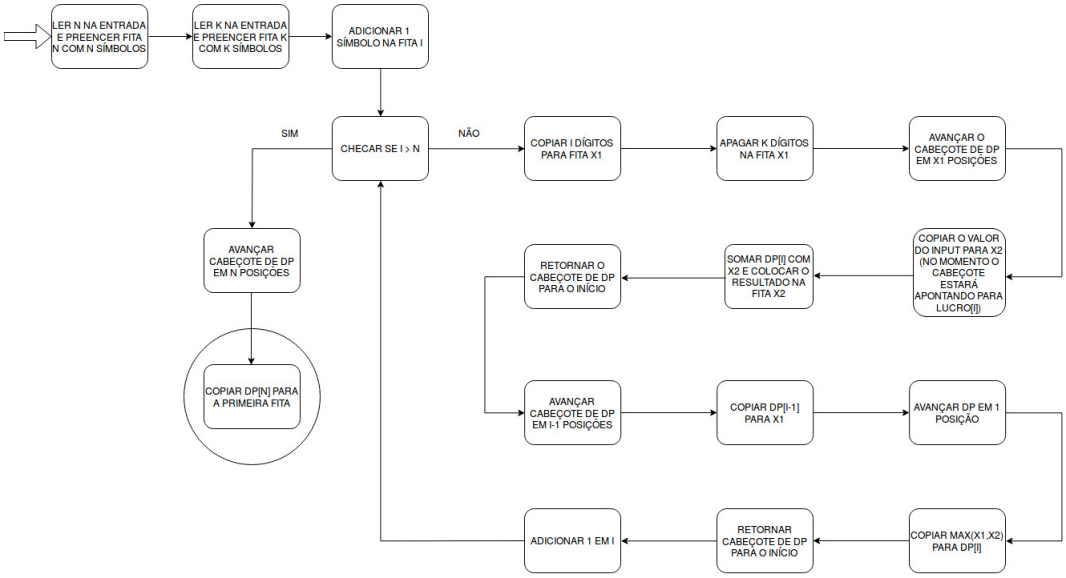

Figura 1: Diagrama da Máquina de Turing

4.2 Linguagem Simples

Como a máquina final tem sete fitas, notou-se que seria uma atividade complexa e exaustiva criar todas as permutações de transições requeridas pelo simulador de Máquinas de Turing quando a maioria das transições realizavam operações simples sobre no máximo três fitas.

Visando diminuir o trabalho manual desnecessário, a equipe optou por criar e utilizar uma linguagem mais simples baseada na linguagem do simulador, que é posteriormente convertida na linguagem base.

Nessa linguagem:

- As palavras chaves padrões do simulador (*name*, *init*, *accept*), bem como suas funções, foram mantidas.
- Uma palavras chaves *fitas* foi adicionada e sua função é definir a quantidade, os *id*s e a ordem das fitas. Isso significa que essa palavra chave deve ser seguida por N *id*s (representando N fitas) separados por vírgula, onde o i-ésimo *id* representa a i-ésima fita da máquina final.
- Houve uma leve, mas importante, modificação na definição de transições que agora seguem o seguinte padrão:

  $//[current\_tape]^M$

  $[current\_state], [read\_symbol]^M$

  $[new\_state], [write\_symbol]^M, [> \mid < \mid -]^M$

  Nessa nova definição, M é a quantidade de fitas nas quais se deseja realizar uma operação. $[current\_tape]^M$ define a ordem de leitura e escrita das fitas que são modificadas nessa transição, i.e. $[read\_symbol]^M[i]$ representa o símbolo que deve ser lido na fita de *id* $[current\_tape]^M[i]$ para $0 < i < M$ (o mesmo vale para os símbolos a serem escritos e as operações a serem realizadas). As demais $N - M$ fitas não serão modificadas, i.e. são geradas todas as permutações dessa transição adicionando as outras fitas sem modifica-las (lendo 1, escrevendo 1 e ficando ou lendo 0, escrevendo 0 e ficando).
- *?* foi definido como símbolo especial de leitura e escrita.

  Numa transição, caso uma fita tenha como $[read\_symbol]$ e $[write\_symbol]$ *?*, serão adicionadas permutações dessa transição (lendo 1 e escrevendo 1 ou lendo 0 e escrevendo 0) realizando a

operação definida na transição para a respectiva fita.

- Adotou-se $//--$ como símbolo que inicia uma linha de comentário (ignorada).

4.3 Parser

Nesta última etapa foi realizada a implementação da máquina na linguagem simples que foi posteriormente convertida na linguagem do simulador (através de um parser simples e objetivo escrito em Python 3 e baseado em expressões regulares).

5 Conclusão

Nesse trabalho avaliamos o poder computacional do modelo de Máquinas de Turing através da implementação de um algoritmo de programação dinâmica.

O modelo mostrou-se poderoso, porém complicado, produzindo maquinas com milhares de transições, necessitando de um passo intermediário na sua construção.

Referências

[1] Radares - URI Online Judge,
`https://www.urionlinejudge.com.br/judge/pt/problems/view/1689`