Anaísa Forti da Fonseca
11811ECP012

Trabalho 03 – Máquinas de Turing

Neste terceiro laboratório, foi solicitada a implementação de um Simulador Universal de Máquinas de Turing (MTs). O programa aceita a especificação de uma MT e, dentre uma lista de cadeias de caracteres dada, diz quais pertencem e quais não pertencem à linguagem reconhecida pela máquina.

As entradas consistem nas informações necessárias para montar a máquina, tais como o número de estados, os símbolos terminais, os símbolos de fita, o estado de aceitação e as transições. Juntamente às especificações da máquina, são fornecidas as cadeias de entrada. A saída trata de dizer se tais cadeias pertencem ou não à linguagem da máquina, imprimindo “aceita” ou “rejeita” para cada cadeia que é analisada.

Por sua simplicidade e rapidez em desenvolvimento, a solução foi implementada em python. O código interpreta as entradas fornecidas pelo usuário com as informações para montar a máquina e armazena estas informações em variáveis e em estruturas de dados como a lista. Cada transição da máquina é uma lista com 5 elementos: o primeiro é o estado de origem, o segundo é o símbolo da transição, o terceiro é o estado de destino, o quarto é o símbolo a ser escrito na fita e o quinto e último elemento é o movimento a ser feito pelo cursor (direita ou esquerda). Todas as transições são armazenadas em uma única lista. O mesmo raciocínio vale para as cadeias: cada uma é uma lista e todas são armazenadas em uma outra lista (lista de listas).

O programa possui a função “percorreCadeia” que recebe como parâmetros uma fita (que consiste na cadeia sendo analisada concatenada com o símbolo bem branco “B”), a posição do cursor na fita (inicialmente 0) e o estado atual da máquina (inicialmente 0 também). Esta é uma função que, para cada caractere da fita em que o cursor estiver, verifica qual transição é possível por meio de um laço “for” e, através da recursividade, consegue percorrer toda a lista de transições para cada posição do cursor. Neste trabalho, apenas máquinas determinísticas foram testadas. Então, para cada símbolo analisado existirá apenas uma transição possível.

A cada transição realizada, um símbolo é escrito na fita e o cursor é movido em uma unidade para a esquerda ou para a direita. Quando a recursividade da função “percorreCadeia” é chamada, em seus parâmetros são passados a nova fita (com o novo

símbolo escrito pela transição realizada), a nova posição do cursor e o novo estado atual da máquina.

Mesmo que a solução tenha sido implementada em python, não foi necessário fazer cópias da fita para realizar a sobrescrita de um símbolo na mesma. Por exemplo, suponha uma fita f = [‘a’, ’b’, ’c’] e uma transição t = [‘0’, ’c’, ’1’, ’d’, ’D’] realizada (ambas definidas como listas). Para substituir o ‘c’ da fita pelo ‘d’ (como diz na transição), basta fazer o comando: fita[2] = transicao[3]. Com isso, a substituição do símbolo será feita e a fita será f = [‘a’, ’b’, ’d’].

Voltando à função “percorreCadeia”, uma máquina de Turing neste código para em dois casos: ou quando não há mais nenhuma transição possível, ou quando chegar no estado de aceitação (não importando o momento em que ele foi atingido). No primeiro caso, a função retornará “false”, já que o motivo de ter parado foi a falta de transições possíveis (e não atingiu o estado de aceitação antes disso). No segundo caso, como a máquina atingiu o estado de aceitação, a função retornará “true” para a cadeia analisada.

Para interpretar o retorno da função “percorreCadeia”, foi utilizado um laço de repetição “for” verificando se, para cada cadeia, o retorno foi “true” ou “false”. Para os retornos “true” o laço imprimirá “aceita”, mostrando que a cadeia pertence à linguagem reconhecida pela máquina. Analogamente, o laço imprimirá “rejeita” para os retornos “false”, revelando que a cadeia não pertence à linguagem da máquina. Com isso, nosso formato de saída é montado.

Com relação ao funcionamento e qualidade, a solução apresentou-se eficiente retornando o resultado esperado para 100% dos casos de teste, sendo estes alguns retirados da internet e os do próprio sistema de correção.

Para que fosse possível analisar a eficiência do programa em termos de tempo, foi utilizada a notação Big-O. Considerando que a função principal possui um laço “for” que percorre a lista de t transições e é feita baseada em recursividade, a cada símbolo lido da fita esse laço “for” é feito novamente. Assim, considerando o pior caso possível, baseando-se na notação Big-O, pode-se definir o tempo como O(n), sendo n o número de símbolos da cadeia/fita. A notação é feita desta maneira já que apenas uma transição é possível para o estado e símbolo de fita atual. Isso se aplica às máquinas de Turing determinísticas.

Conclui-se, então, que quanto maior a cadeia (e, portanto, a fita), mais tempo será necessário para percorrer ela completamente. Entretanto, é importante lembrar que, diferente de um autômato, uma máquina de Turing para ao chegar no estado de aceitação a qualquer momento. Então, dependendo da máquina descrita, a cadeia/fita nem sempre

será percorrida por completo. Para melhor análise de espaço e tempo, neste relatório serão levadas em conta máquinas que, para chegar ao estado de aceitação, precisam percorrer a fita por completo. Também, serão consideradas apenas máquinas que ou não tenham mais transições possíveis em um certo ponto, ou que atinjam o estado de aceitação para determinada cadeia (ou seja, máquinas que sempre param em um certo ponto). Caso esses dois fatores não sejam levados em conta, a máquina pode entrar em loop e percorrer a fita infinitamente.

Já para análise da eficiência do programa em termos de espaço, observou-se o tamanho máximo da fita atingido, levando em conta máquinas com apenas uma fita. Como a máquina é determinística, há apenas uma transição possível para cada símbolo da fita. Em um primeiro momento considera-se que há apenas uma escrita na fita a cada símbolo lido. Portanto, o tamanho máximo da fita atingida nesta máquina considerada é n+1, sendo n o número de símbolos da cadeia analisada (que foi inserida na fita) e o +1 referente ao símbolo em branco (B) que é adicionado logo depois da cadeia na criação da fita.

Fitas de máquinas de Turing podem ser infinitas para a direita. Então, existirão casos em que a máquina adicionará símbolos além dos n+1 que são inicializados na fita (cadeia analisada + símbolo em branco). Nestes casos, não há como definir ao certo o tamanho máximo da fita, pois dependerá das possíveis transições e escritas na fita que compõem a máquina.

Portanto, seguindo o raciocínio de que há apenas uma escrita na fita a cada símbolo lido e que a fita possuirá tamanho máximo de n+1, pode-se concluir de que o espaço ocupado é diretamente proporcional ao tamanho da cadeia analisada. Ou seja, quanto maior a cadeia percorrida, maior será o tamanho máximo da fita da respectiva máquina de Turing. Novamente, aqui foram consideradas máquinas determinísticas que precisam percorrer a fita por completo para chegar ao estado de aceitação e, também, foram consideradas apenas linguagens Turing-decidíveis (linguagens para as quais a MT sempre para).

Com isso, examinando os dados de espaço e tempo descritos, encontra-se um ponto muito importante em comum: quanto maior a cadeia dada como entrada para máquina de Turing, maior o tempo e o espaço necessários para percorrer essa cadeia e para armazenar a fita, respectivamente.