Controle de Tráfego Urbano Inteligente

Trabalho Final do Curso de Teoria da Computação

João Roberto - 222217111
Caio Mello - XXXX
Vinicius - YYYYYYYYY
Ciclano - ZZZZZZZZ

Período Letivo
2025.2

Índice

1. Introdução

1.1. Enunciado do problema

A Companhia Turing Traffic Systems está desenvolvendo um sistema de controle de tráfego urbano baseado em Máquinas de Turing para gerenciar os semáforos de uma cidade inteligente. O objetivo é determinar qual deve ser o próximo estado do semáforo principal de um cruzamento, considerando informações sobre o fluxo de veículos.

O cruzamento possui três avenidas que se encontram no mesmo ponto, e cada avenida possui sensores que contam o número de veículos aguardando em cada direção.

O sistema deve decidir a próxima cor do semáforo principal de acordo com a seguinte política:

***I** - Se uma avenida tiver mais da metade do total de veículos somados das três avenidas, ela terá prioridade, e o semáforo para essa avenida ficará verde, enquanto os outros ficarão vermelho.*

***II** Caso nenhuma avenida tenha mais da metade dos veículos, o sistema escolhe a avenida com maior número de veículos.*

***III** Em caso de empate, a avenida A tem prioridade sobre B, e B sobre C (ordem A > B > C).*

1.2. Contextualização e importância na área de cidades inteligentes.

lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero lero

1.3. Abstrações, notações específicas e corolários significativos

Como parte desse trabalho, temos a missão de modular Máquinas de Turing que solucionem o problema preposto. Para tal, teremos como entrada uma string representando a distribuição de carros dentre as avenidas A, B e C, seguindo o formato:

$$\text{Seja } w \text{ uma palavra e } n \in \mathbb{N}$$

$$w = \varphi_0 * ... * \varphi_{n-1} * \varphi_n$$

$$\varphi_k \in \{a, b, c\} \text{ tal que } 0 \leq k \leq n$$

Nesta representação, cada carro na avenida A será uma letra 'a', na avenida B, a letra 'b', e na C, 'c'.

Será útil também a definição de uma função para cálculo da cardinalidade de uma letra, construída segundo o descrito abaixo:

$$\text{Seja } w \text{ uma palavra e } x \text{ uma letra}$$

$$\gamma_x(w) : \{a, b, c\}^* \to \mathbb{N}$$

$$\gamma_x(w) = |\{i \in \{1, ..., |w|\} \land i = x\}|$$

Onde γ é a função de cardinalidade.

De forma direta, $\gamma_x(w)$ nos retorna o número de ocorrências da letra x na palavra w.

A partir do enunciado da questão, abstraímos duas propriedades que nosso automômato deve computar, as chamaremos de A e B.

I. Determinar a avenida com quantidade de carros superior a metade do total de automóveis em todas avenidas.
   - Ou seja, **determinar a letra x cuja cardinalidade é superior ao tamanho total da string w de entrada**:

$$x \in \{a, b, c\} \mid \gamma_x(w) > \frac{\gamma_a(w) + \gamma_b(w) + \gamma_c(w)}{2} \tag{A}$$

   - Caso mais de uma avenida se qualifique, devemos respeitar a A sobre B, B sobre C e, por transitividade, A sobre C.

II. Determinar a avenida com a maior quantidade de carros.
   - Ou seja, **determinar a letra x de maior cardinalidade na palavra w**:

$$\begin{gathered} x, y, z \in \{a, b, c\} \land y \neq z \land z \neq x \land x \neq z \\ \gamma_x(w) > \gamma_y(w) \land \gamma_x(w) > \gamma_z(w) \end{gathered} \tag{B}$$

   - Em caso de empate, devemos respeitar a ordem de prioridade definida em 1.2.

Também é importante considerar que a computação da propriedade 2 só deve ocorrer caso não haja parada por estado final na computação da condição 1.

Vamos considerar também algumas equivalências para facilitar a modelagem e validação do automômato.

A primeira é centrada na propriedade A, a qual nos dá que:

$$\gamma_x(w) > \frac{\gamma_a(w) + \gamma_b(w) + \gamma_c(w)}{2}$$

$$2 * \gamma_x(w) > \gamma_a(w) + \gamma_b(w) + \gamma_c(w)$$

Dado que $x \in \{a, b, c\}$,

$$\gamma_x(w) \in \{\gamma_a(w), \quad \gamma_b(w), \quad \gamma_c(w)\}$$

Generalizando,

$$\text{Sejam} \quad x, y, z \in \{a, b, c\} \mid x \neq y \land y \neq z$$

$$2 * \gamma_x(w) > \gamma_x(w) + \gamma_y(w) + \gamma_z(w)$$

$$\gamma_x(w) > \gamma_y(w) + \gamma_z(w) \tag{C}$$

Ou seja,

$$\gamma_x(w) > \frac{\gamma_x(w) + \gamma_y(w) + \gamma_z(w)}{2} \leftrightarrow \gamma_x(w) > \gamma_y(w) + \gamma_z(w)$$

Portanto, basta que nosso automômato compute propriedade C para que também valha a condição 1 do enunciado.

2. Máquina de uma única fita

2.1. Módulos e submódulos

Nossa máquina de uma única fita, a qual iremos tratar por **MT1**, terá dois módulos principais: α e β.

Cada estado final da máquina representará a decisão quanto a qual o único sinal que deve ter prioridade.

O módulo α será responsável por verificar se a condição 1 do enunciado vale para alguma letra a, b ou c através da propriedade C.

— TODO ELABORAR NO MODULO β

Caso nenhum estado final seja alcançado dentro de A, o automômato recorrerá ao módulo B, onde avaliará a condição 2 do problema.

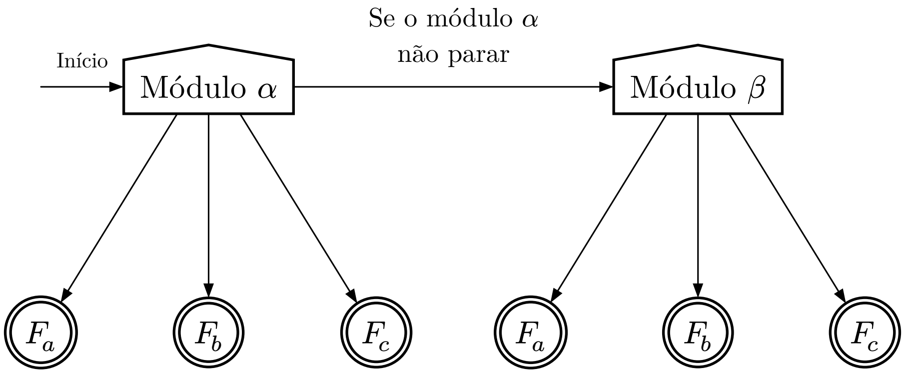


Diagrama 1: Arquitetura do automômato

2.1.1. Dentro do Módulo α

O módulo α irá analisar se a propriedade C vale para alguma letra em $\{a, b, c\}$. Essa operação será feita linearmente, um símbolo por vez.

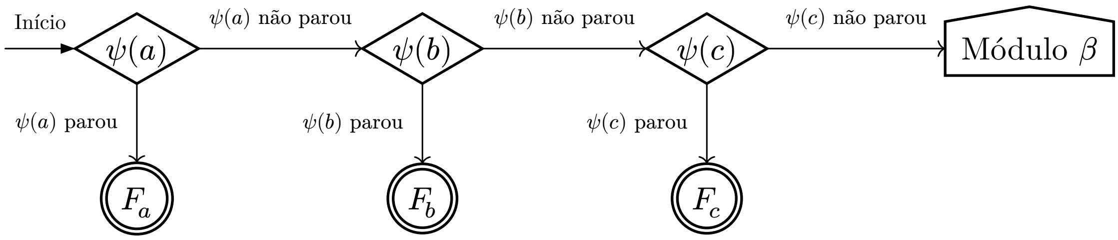
A análisa de um desses glifos pode implicar ou na parada total do automômato ao alcançar um estado final, ou, na avaliação da condição para outro glifo do alfabeto.

Caso nenhuma letra do alfabeto cumpra com uma propriedade alvo do módulo, a computação é encaminhada para o Módulo β do automômato.

Dessa forma, abstraímos um submódulo ψ tal que:

$$\psi(x) : \text{Computa se x cumpre a propriedade C, caso positivo, o automômato pára.}$$

Iremos avaliar, nesta ordem $\psi(a)$, $\psi(b)$, $\psi(c)$. Essa sequência garante a condição de desempate definida em I. Uma vez que caso alguma avaliação de ψ pare, as demais não serão computadas. Assim, caso valha $\psi(a)$ e $\psi(b)$ ou $\psi(c)$, iremos ainda parar no estado final representativo de a. A verdade análoga vale para b.



Fluxograma 1: Formato do Módulo A

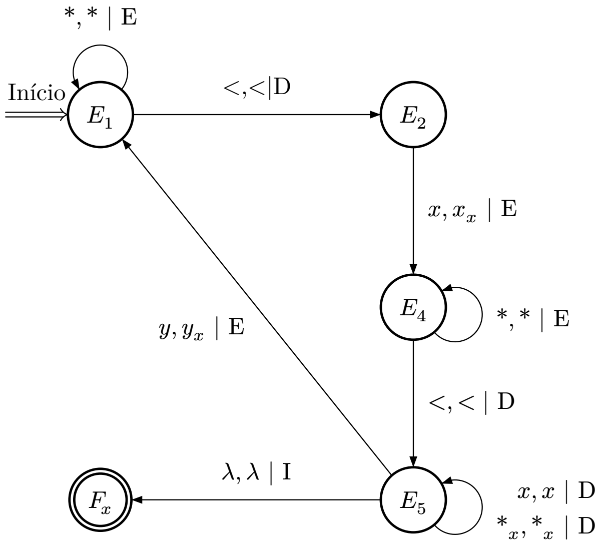
2.1.2. Submódulo: Função ψ

Definiremos os elementos do automômato que computa ψ da seguinte forma:

$$\mathbb{MT} = \{\mathbb{Q}, \Sigma, \Gamma, \delta, q_0, \mathbb{F}\} \ \text{ e } \ \Sigma = \{a, b, c\}$$

$$\Gamma = \Sigma \cup \left\{\forall x, y \in \Sigma \mid x_y\right\} \ \text{ e } \ q_0 = E_0 \ \text{ e } \ \mathbb{F} = \{F_x\}$$

$$\delta : \text{conjunto das transições contidas na imagem abaixo.}$$



Submódulo 1: Função ψ

A partir do automômato disposto, podemos discretizar as transições na forma do algoritmo abaixo:

$w =$ palavra de entrada
$\Sigma = \{a, b, c\}$
$x \in \Sigma$

1. Lê cada letra de w até encontrar uma ocorrência de x;
   a) Caso não encontre, consideramos que a propriedade C não vale para x e então executamos ψ para outro x;
   b) Caso encontre, sobreescreveremos o x na fita com a letra x_x;
2. Retornamos ao início da fita;
3. Buscamos uma letra y tal que $y \neq x$;
   a) Caso não encontre, consideramos que a propriedade C vale para x e paramos num estado final.
   b) Caso encontre, sobreescreveremos o y na fita com a letra y_x;
4. Retornamos ao início da fita e voltamos ao passo 1.

Vamos então provar que o algoritmo descrito verifica a propriedade C.

Dado que a palavra é finita, entendemos que o *loop* para em 1.a ou 3.a.

Tendo em mente que,

$$n \in \mathbb{N} \mid x, \gamma_n \in \{a, b, c\} \mid w \text{ é uma palavra.} \mid w = \gamma_0 * ... * \gamma_n$$

Temos em 1.a e 3.a, dois casos análogos

Os passos 1, 1.2, e 2 e 4 constroem o conjunto X:

$$X = \{\forall \gamma_i \in w \mid \gamma_i = x\}$$

Enquanto 3, 3.2 e 4 definem $\overline{X}$:

$$\overline{X} = \{\forall \gamma_i \in w \mid \gamma_i \neq x\}$$

Basta que verifiquemos:

$$|X| > |\overline{X}|$$

2.2. Máquina na íntegra

ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis

2.3. Análise de complexidade

ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis

2.4. Exemplos

ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis

3. Máquina de múltiplas fitas

3.1. Módulos

ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis

3.2. Máquina na íntegra

ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis

3.3. Análise de complexidade

ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis

3.4. Exemplos

ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis ipsi literis