Problema da Correspondência de Post

Jerusa Marchi
jerusa.marchi@ufsc.br

Universidade Federal de Santa Catarina (UFSC)
Laboratório de Inteligência Artificial e Tecnologia Educacional (IATE)

18 de outubro de 2019

Problema da Correspondência de Post

O problema da correspondência de Post, introduzido e provado ser indecidível, em 1946, pelo matemático polonês Emil Post, trata da manipulação de palavras.

Ele é descrito como um queba-cabeça. Inicia-se com uma coleção de dominós, cada um contendo duas palavras, uma em cada lado, e deseja-se que as palavras obtidas lendo-se ambos os lados sejam iguais.

Definições Básicas

Definição (Dominó)

Um dominó é representado graficamente como sendo:

$$\left[\frac{a}{ab}\right]$$

Definição (Coleção de Dominós)

Uma coleção de dominós é representada graficamente como sendo:

$$\left\{\left[\frac{b}{ca}\right], \left[\frac{a}{ab}\right], \left[\frac{ca}{a}\right], \left[\frac{abc}{c}\right]\right\}$$

Definição (Combinação)

Uma lista de dominós onde as palavras formadas lendo os símbolos da parte superior e inferior sejam iguais é denominada combinação.

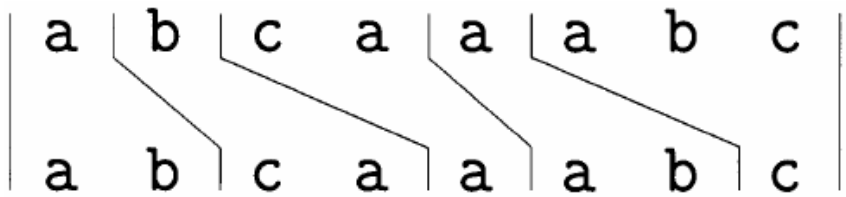
Exemplo de Combinação

Exemplo

A lista a seguir é considerada uma combinação. Lendo-se a parte superior da lista obtem-se a palavra abcaaabc, que é a mesma obtida ao ler a parte inferior.

$$\left[\frac{a}{ab}\right]\left[\frac{b}{ca}\right]\left[\frac{ca}{a}\right]\left[\frac{a}{ab}\right]\left[\frac{abc}{c}\right]$$

A combinação pode ser descrita também deformando os dominós de modo que os símbolos correspondentes da parte superior e inferior se alinhem.



Indecidibilidade do Problema

Para algumas coleções de dominós pode não ser possível encontrar uma combinação. Por exemplo, a lista a seguir não pode conter uma combinação porque cada palavra superior é mais longa do que a palavra inferior correspondente.

$$\left\{ \left[\frac{\text{abc}}{\text{ab}}\right], \left[\frac{\text{ca}}{\text{a}}\right], \left[\frac{\text{acc}}{\text{ba}}\right] \right\}$$

O problema da correspondência de Post consiste em determinar se uma coleção de dominós possui uma combinação. Este problema não pode ser resolvido por nenhum algoritmo.

Definição Formal do Problema

Definição (Problema da Correspondência de Post)

Uma instância do problema da correspondência de Post é uma coleção P de dominós:

$$P = \left\{ \left[\frac{t_1}{b_1}\right], \left[\frac{t_2}{b_2}\right], \cdots, \left[\frac{t_k}{b_k}\right] \right\},$$

e uma combinação é uma sequência $i_1, i_2, \cdots, i_l$, onde $t_{i_1}, t_{i_2} \cdots t_{i_l} = b_{i_1}, b_{i_2} \cdots b_{i_l}$. O problema consiste em determinar se P possui uma combinação. Sendo

$$PCP = \{\langle P \rangle \mid P \text{ possui uma combinação}\}.$$

Teorema

Teorema (SIPSER, 2006)

PCP é indecidível.

Ideia da Prova

Ideia da Prova

Conceitualmente essa prova é simples, ainda que ela envolva muitos detalhes técnicos. A principal técnica é uma redução da linguagem A_{TM} através da história de computação. Nós mostramos que a partir de qualquer Máquina de Turing M e uma entrada w nós podemos construir uma instância P onde uma combinação é um histórico de computação aceitação de M sobre$w$. Se pudéssemos determinar se a instância possui uma combinação, seríamos capazes de determinar se M aceita w.

Ideia da Prova

Ideia da Prova

Como nós podemos construir P de modo que uma combinação seja uma história de computação de M sobre w? Nós escolhemos os dominós em P de modo que construir uma combinação force a simulação de M. Nesta combinação, cada dominó liga uma posição ou posições em uma configuração com os estados correspondentes na próxima configuração.

Ideia da Prova

Ideia da Prova

Antes de iniciar a construção, é necessário lidar com três pequenos pontos técnicos:

- Ao construir P, assumimos que M ao computar w, nunca tenta mover a cabeça de leitura além do final esquerdo da fita.
- Se $w = \varepsilon$, nós usamos a palavra $\sqcup$ no lugar de w na construção.
- Modificamos PCP para exigir que a combinação começe com o primeiro dominó,

$$\left[\frac{t_1}{b_1}\right].$$

Ideia da Prova

Ideia da Prova

Futuramente, nós mostraremos como eliminar essa exigência. Nós chamamos esse problema de problema da correspondência de Post modificado. Sendo

$MPCP = \{\langle P \rangle \mid P \text{ é a combinação que inicia no primeiro dominó}\}$.

Demonstração do Teorema

Prova

Deixamos a Máquina de Turing R decidir PCP e construímos S para decidir A_{TM}. Sendo

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject}),$$

onde Q, Σ, Γ e δ são o conjunto de estados, o alfabeto da entrada, o alfabeto da fita, e a função de transição de M, respectivamente.

Neste caso, S constrói uma instância de PCP P que possui uma combinação se e sómente se M aceita w. Para fazer isso, S primeiramente constrói uma instância P' de MPCP. Nós descreverêmos a construção em sete partes, cada uma corresponde a um aspecto particular de simular M sobre w. Para explicar o que está acontecendo, nós intercalamos a construção com um exemplo da construção em ação.

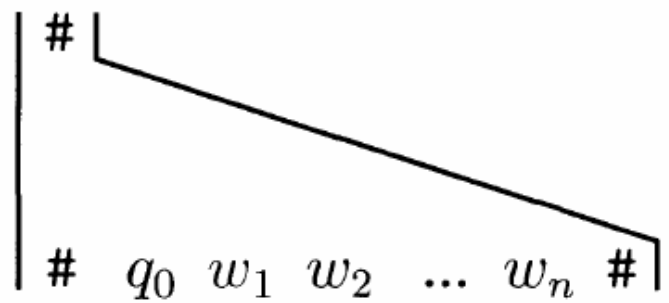
Parte 1

Prova

A construção começa da seguinte maneira.

Coloque $\left[\frac{\#}{\#q_0 w_1 w_2 \cdots w_n \#}\right]$ em P' como o primeiro dominó $\left[\frac{t_1}{b_1}\right]$.

Como P' é uma instância de MPCP, a combinação deve iniciar com este dominó. Assim, a palavra inferior começa corretamente com $C_1 = q_0 w_1 w_2 \cdots w_n$, a primeira configuração na história de computação para M sobre w, como mostrado na figura a seguir.



Parte 1

Prova

Nesta representação da combinação parcial realizada até agora, a palavra inferior consiste em $\#q_0 w_1 w_2 \cdots w_n \#$ e a palavra superior somente de $\#$. Para conseguir uma combinação é necessário estender a palavra superior para combinar com a inferior. Nós provemos dominós adicionais para permitir essa extensão. Os dominós adicionais fazem com que a próxima configuração de M apareça na extensão da palavra inferior, forçando a simulação de um passo de M.

Nas partes 2, 3 e 4, nós adicionamos em P' dominós que executam a parte principal da simulação. Parte 2 lida com o movimento da cabeça para a direita, parte 3 lida com a movimentação para a esquerda e a parte 4 lida com as células da fita que não são adjacentes a cabeça de leitura.

Parte 2

Prova

Para cada $a, b \in \Gamma$ e cada $q, r \in Q$ onde $q \neq q_{reject}$,

$$\text{se } \delta(q, a) = (r, b, \text{R}) \text{ coloque } \left[\frac{qa}{br}\right] \text{ em } P'.$$

Parte 3

Prova

Para cada $a, b, c \in \Gamma$ e cada $q, r \in Q$ onde $q \neq q_{reject}$,

$$\text{se } \delta(q, a) = (r, b, \mathrm{L}) \text{ coloque } \left[\frac{cqa}{rcb}\right] \text{ em } P'.$$

Parte 4

Prova

Para cada $a \in \Gamma$,

$$\text{coloque } \left[\frac{a}{a}\right] \text{ em } P'.$$

Agora nós iremos construir um exemplo hipotético para ilustrar o que nós construímos até agora. Seja $\Gamma = \{0, 1, 2, \sqcup\}$. Suponha que w é a palavra 0100 e que o estado inicial de M seja q_0. No estado q_0, após ler um 0 suponha que a função de transição diga que M entra no estado q_7, escreve um 2 na fita e move a cabeça à direita, ou seja $\delta(q_0, 0) = (q_7, 2, \mathrm{R})$.

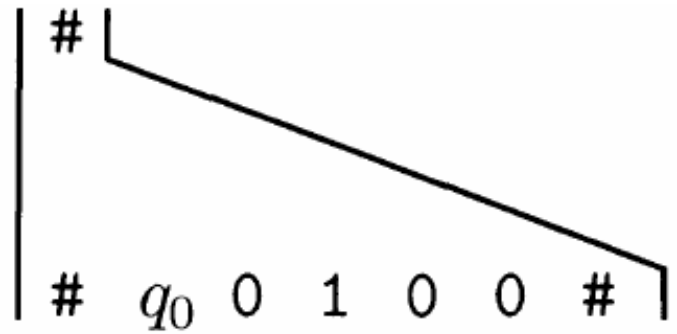
Parte 4

Prova

Parte 1 coloca o dominó

$$\left[\frac{\#}{\#q_0 0100\#}\right] = \left[\frac{t_1}{b_1}\right]$$

em P', e a combinação começa:



Parte 4

Prova

Adicionalmente, a parte 2 coloca o dominó

$$\left[\frac{q_0 0}{2 q_7}\right]$$

como $\delta(q_0, 0) = (q_7, 2, \mathrm{R})$ e a parte 4 coloca os dominós

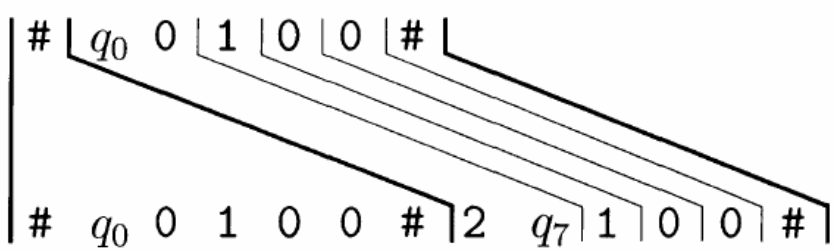
$$\left[\frac{0}{0}\right], \left[\frac{1}{1}\right], \left[\frac{2}{2}\right], e \left[\frac{\sqcup}{\sqcup}\right]$$

em P', pois 0,1,2 e $\sqcup$ são os membros de Γ.

Parte 4

Prova

A parte 4, juntamente com a parte 5 permite estender a combinação para



Assim, os dominós das partes 2, 3 e 4 estendem a combinação adicionando uma segunda configuração depois da primeira. Nós queremos que esse processo continue, adicionando a terceira configuração, a quarta, e assim por diante. Para isso acontecer, nós precisamos adicionar mais um dominó para copiar o símbolo #.

Parte 5

Prova

Coloque $\left[\frac{\#}{\#}\right]$ e $\left[\frac{\#}{\sqcup\#}\right]$ em P'.

O primeiro destes dominós nos permite copiar o símbolo $\#$ que marca a separação entre as configurações. Adicionalmente, o segundo dominó nos permiteadicionar um símbolo em branco $\sqcup$ no final da configuração para simular os infinitos espaços em branco a direita que são suprimidos quando nós escrevemos a configuração.

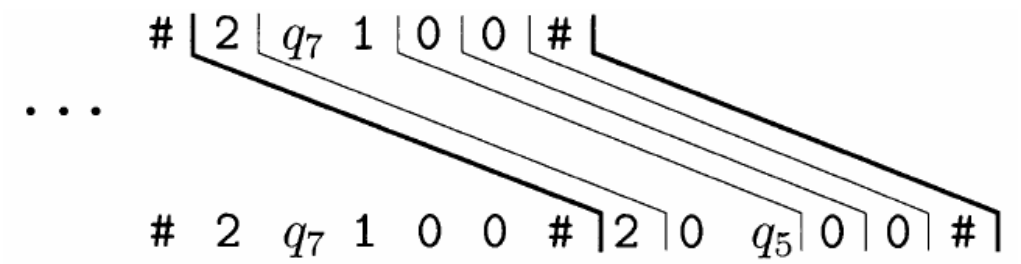
Parte 5

Prova

Continuando com o exemplo, digamos que no estado q_7, depois de ler um 1, M vai para o estado q_5, escreve um 0, e move a cabeç para a direita. Ou seja, $\delta(q_7, 1) = (q_5, 0, \mathrm{R})$. Então nós temos o dominó

$$\left[\frac{q_7 1}{0 q_5}\right] \text{ em } P'.$$

A última combinação parcial estende a combinação para



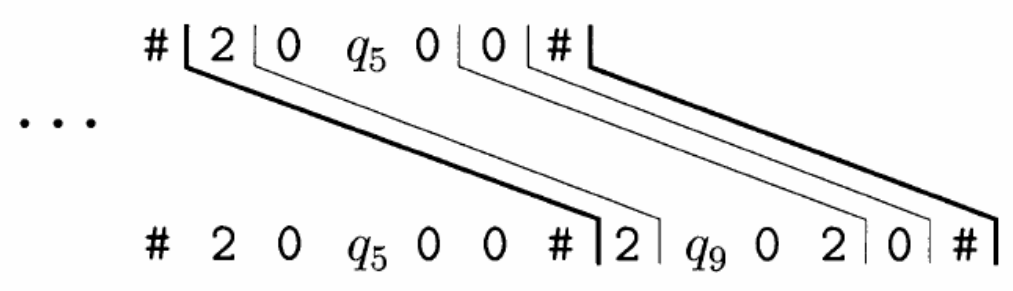
Parte 5

Prova

Então suponha que no estado q_5, depois de ler um 0, M vai para o estado q_9, escreve um 2, e move a cabeç para a esquerda. Ou seja, $\delta(q_5, 0) = (q_9, 2, \mathrm{L})$. Então nós temos os dominós

$$\left[\frac{0q_50}{q_902}\right], \left[\frac{1q_50}{q_912}\right], \left[\frac{2q_50}{q_922}\right], e \left[\frac{\sqcup q_50}{q_9 \sqcup 2}\right]$$

A primeira configuração é relevante porque o símbolo a esquerda da cabeça de leitura é 0. A combinação parcial precedente estende a combinação para



Parte 5

Prova

Note que, enquanto nós construímos uma combinação, somos forçados a simular M sobre w. Este processo continua enquanto até M alcançar um estado de parada. Se um estado de aceitação acontecer, nós queremos que a parte superior da combinação parcial "alcançe"a parte inferior de forma que a combinação esteja completa. Nós podemos fazer com que isso aconteça adicionando dominós adicionais.

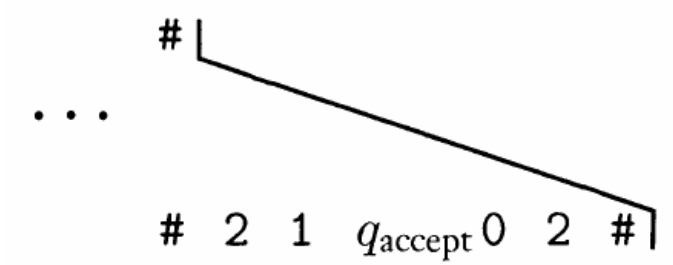
Parte 6

Prova

Para cada $a \in \Gamma$,

$$\text{coloque } \left[\frac{aq_{accept}}{q_{accept}}\right] e \left[\frac{q_{accept}a}{q_{accept}}\right] \text{ em } P'.$$

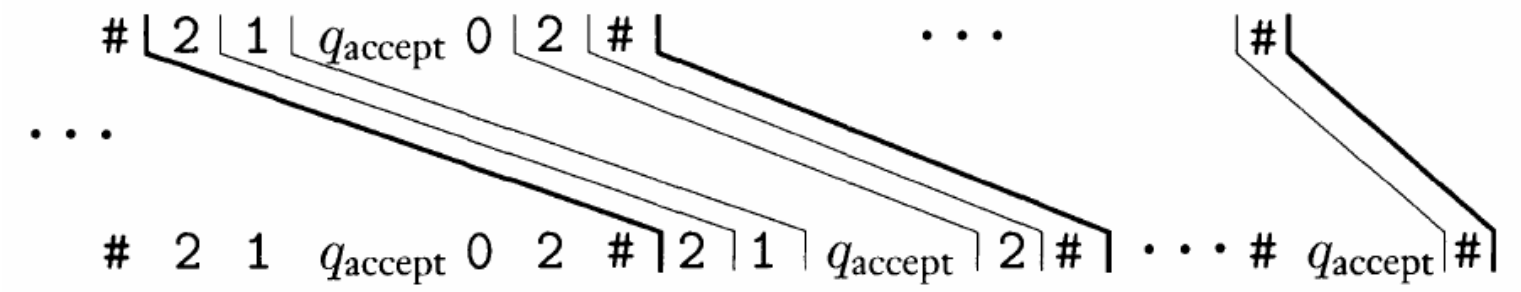
Esse passo tem o efeito de adicionar "falsos passos" na Máquina de Turing depois que ela para, onde a cabeça de leitura "come" símbolos adjacentes enquanto não restar nenhum. Continuando com o exemplo, se a combinação parcial se igualar em um ponto em qua a máquina para em um estado de aceitação for



Parte 6

Prova

Os dominós que nós acabamos de adicionar permitem que a combinação continue:



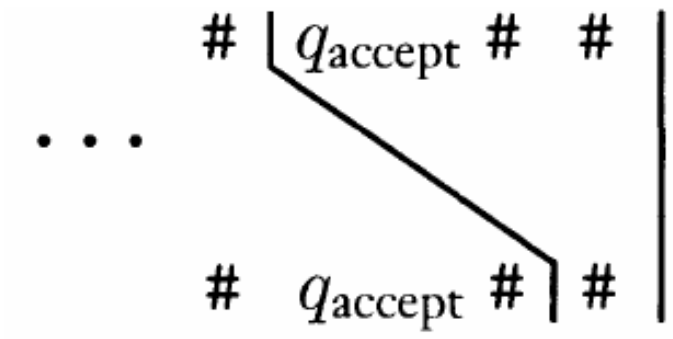
Parte 7

Prova

Finalmente nós adicionamos o dominó

$$\left[\frac{q_{accept}\#\#}{\#}\right]$$

e completamos a combinação:



Parte 7

Prova

Com isso concluímos a contrução de P'. Recorde que que P' é uma instância de MPCP onde a combinação simula a computação de M sobre w. Para finalizar a prova, nós recordamos que MPCP difere de PCP no fato de que é requerido que a combinação inicie no primeiro dominó da lista. Se nós virmos P' como uma instância de PCP ao invés de MPCP, ela, obviamente, tem uma combinação, independente de de onde M pare sobre w.

Nós mostraremos agora como converter P' para P, uma instância de PCP que continua simulando M sobre w. Nós fazemos isso com um truque técnico. A ideia é construir o requirimento de começar com o primeiro dominó diretamente no problema, de forma que declarando o requerimento explícito se torne desnecessário. Nós precisamos introduzirum pouco de notação para este propósito.

Parte 7

Prova

Seja $u = u_1 u_2 \cdots u_n$ qualquer palavra de tamanho n. Defina $\star u$, $u\star$, e $\star u\star$ como sendo as três palavras

$$\star u = *u_1 * u_2 * u_3 * \cdots * u_n$$

$$u\star = u_1 * u_2 * u_3 * \cdots * u_n*$$

$$\star u\star = *u_1 * u_2 * u_3 * \cdots * u_n*$$

Aqui, $\star u$ adiciona o símbolo $*$ antes de cada caractere de u, $u\star$ adiciona um símbolo $*$ depois de cada caractere de u, e $\star u\star$ adiciona um antes e depois de cada caractere de u.

Parte 7

Prova

Para converter P' para P, uma instância de PCP, nós devemos fazer o seguinte. Se P' fosse a coleção

$$\left\{ \left[\frac{t_1}{b_1}\right], \left[\frac{t_2}{b_2}\right], \left[\frac{t_3}{b_3}\right], \cdots, \left[\frac{t_k}{b_k}\right] \right\},$$

nós deixamos P ser a coleção

$$\left\{ \left[\frac{\star t_1}{\star b_1 \star}\right], \left[\frac{\star t_1}{b_1 \star}\right], \left[\frac{\star t_2}{b_2 \star}\right], \left[\frac{\star t_3}{b_3 \star}\right], \cdots, \left[\frac{\star t_k}{b_k \star}\right], \left[\frac{\star \diamond}{\diamond}\right], \right\}.$$

Parte 7

Prova

Considerando P como uma instância de PCP, nós vemos que o único dominó que poderia começar uam combinação seria o primeiro,

$$\left[\frac{\star t_1}{\star b_1 \star}\right],$$

porque ele é o único onde ambos a parte superior quanto a inferior iniciam com o mesmo símbolo - a saber, $*$. Além disso, ao forçar que a combinação começe com o primeiro dominó, a presença dos $*$ não afeta possíveis combinações, pois ele simplesmente intercala os símbolos originais.

Parte 7

Prova

Os símbolos originais agora ocorrem nas posições pares da combinação. O dominó

$$\left[\frac{\star\diamond}{\diamond}\right]$$

existe para adicionar um $*$ extra no final da combinação. $\square$

Referências


Michael Sipser.
Introduction to the Theory of Computation.
Thomson Course Technology, Boston, 2 nd edition, 2006.