Redutibilidade

Juliana Kaizer Vizzotto

Universidade Federal de Santa Maria

Disciplina de Teoria da Computação

Roteiro

.

- Introdução
- Problemas Indecidíveis da Teoria da Computação

Introdução

- Estabelecemos a MT como nosso modelo de um computador de propósito geral.
- Apresentamos diversos problemas que são solúveis por máquinas de Turing.
- Vimos um exemplo de um problema, A_{MT}, que é computacionalmente insolúvel.
- Vamos começar a estudar outros problemas insolúveis!
- Vamos aprender o método principal para provar que problemas são computacionalmente insolúveis: **Redutibilidade**

Introdução

- Uma **redução** é uma maneira de converter um problema em outro de forma que uma solução para o segundo problema possa ser usada para resolver o primeiro.
- Por exemplo, suponha que você deseje se orientar em uma nova cidade:
  - Você sabe que seria fácil se tivesse um mapa.
  - Consequentemente, você pode *reduzir* o problema de se orientar na cidade ao problema de se obter um mapa da cidade.
- A redutibilidade sempre envolve dois problemas, que denominamos A e B.
- Se A se reduz a B, podemos usar uma solução para B para resolver A.
- Note que redutibilidade não diz nada sobre resolver A ou B sozinhos, mas somente sobre a solubilidade de A na presença de uma solução para B.

Introdução

- A redutibilidade também ocorre em problemas matemáticos!
- Por exemplo, o problema de se medir a área de um retângulo se *reduz* ao problema de medir o seu comprimento e largura.
- O problema de se resolver um sistema de equações linerares, se *reduz* ao problema de se inverter uma matriz.
- A redutibilidade desempenha um papel importante na classificação de problemas por decidibilidade e também em teoria da complexidade.
- Quando A é *redutível* a B, resolver A não pode ser mais difícil que resolver B, pois uma solução para B dá uma solução para A.
- Em termos da teoria da computação, se A for redutível a B e B for decidível, A também será decidível.

Introdução

- **Chave para provar que vários problemas são indecidíveis:** Equivalentemente, se A for indecidível e redutível a B, B será indecidível.
- **Método para provar que um problema é indecidível:** mostrar que algum outro problema já mostrado como indecidível se reduz a ele.

Problemas Indecidíveis

- Já vimos que A_{MT}, o problema de se determinar se uma máquina de Turing aceita uma dada entrada, é indecidível!
- Vamos considerar um problema relacionado, $PARA_{MT}$, o problema de se determinar se uma MT pára (aceitando ou rejeitando) uma data entrada.
- Usamos o termo *problema da parada* para a linguagem A_{MT}, muito embora $PARA_{MT}$ seja o real problema da parada.
- Daqui em diante, distinguiremos entre os dois chamando A_{MT} de problema de aceitação.
- Vamos usar a indecidibilidade de A_{MT} para provar a indecidibilidade de $PARA_{MT}$ reduzinho A_{MT} a $Para_{MT}$.

$$PARA_{MT} = \{\langle M, w\rangle | \text{M eh uma MT e M para sobre a entrada } w\}.$$

Problemas Indecidíveis

- **Teorema**: $PARA_{MT}$ é indecidível.
- **Ideia da Prova**: Essa prova é por contradição. Suponha que $PARA_{MT}$ seja decidível e usamos essa suposição para mostrar que A_{MT} é decidível, uma contradição!
- A ideia chave é mostrar que A_{MT} é redutível a $PARA_{MT}$.
- Vamos supor que temos uma MT R que decide $PARA_{MT}$.
- Com R, podemos testar se M pára sobre w.
- Se R indica que M não pára sobre w, rejeite, pois $\langle M, w \rangle$ não está em A_{MT}.

Problemas Indecidíveis

- Entretanto, se R indica que M pára sobre w, você pode fazer a simulação sem qualquer perigo de entrar em *loop*.
- Consequentemente, se a MT R existe, podemos decidir A_{MT}; mas sabemos que A_{MT} é indecidível, uma contradição.
- Em virtude dessa contradição, podemos concluir que R não existe.
- Logo $PARA_{MT}$ é indecidível.

Problemas Indecidíveis

- **Prova Formal**: Vamos supor para os propósitos de obter uma contradição, que a MT R decida $PARA_{MT}$. Construímos a MT S para decidir A_{MT}, com S operando da seguinte forma:
- "Sobre a entrada $\langle M, w \rangle$, uma codificação de uma MT M e uma cadeia W:
  1. Rode a MT R sobre a entrada $\langle M, w \rangle$.
  2. Se R rejeita, *rejeite*.
  3. Se R aceita, simule M sobre w até que ela pare.
  4. Se M aceitou, *aceite*; se M rejeitou, *rejeite*."
- Claramente, se R decide $PARA_{MT}$, então S decide A_{MT}. Como A_{MT} é indecidível, $PARA_{MT}$ também deve ser indecidível.

Problemas Indecidíveis

- Esse teorema ilustra mostra nossa estratégia para provar que um problema é indecidível.

- Essa estratégia é comum à maioria das provas de indecidibilidade, exceto no caso da indecidibilidade da própria A_{MT}, que é provada diretamente através do método da diagonalização.

- Vamos ver outros teoremas e suas provas como exemplos adicionais do método da redutibilidade para provar indecidibilidade.

Problemas Indecidíveis

- Seja
$$V_{MT} = \{\langle M\rangle | M \text{ eh uma MT e } L(M) = \emptyset\}$$
- **Teorema**: V_{MT} é indecidível.
- **Idéia da Prova**: Seguimos o padrão adotado no teorema anterior.
- Supomos para o propósito de obter uma contradição que V_{MT} é decidível e então, mostramos que A_{MT} é decidível, uma contradição.
- Seja R uma MT que decide V_{MT}. Usamos R para construir uma MT S que decide A_{MT}.
- Como S funcionará quando ela receber a entrada $\langle M, w\rangle$?

Problemas Indecidíveis

- Uma ideia é S rodar R sobre a entrada $\langle M \rangle$ é ver se ela aceita.
- Se aceita, sabemos que $L(M)$ é vazia e, por conseguinte, que M não aceita w.
- Mas se R rejeita $\langle M \rangle$, tudo o que sabemos é que $L(M)$ não é vazia, e consequentemente que M aceita alguma cadeia, porém ainda não sabemos se M aceita a cadeia específica w.
- Dessa forma precisamos de uma ideia diferente!

Problemas Indecidíveis

- Em vez de rodar R sobre $\langle M \rangle$, rodamos R sobre uma modificação de M.
- Modificamos M para garantir que M rejeite todas as cadeias, exceto w, mas que sobre a entrada w ela funcione normalmente.
- Então, usamos R para determinar se a máquina modificada reconhece a linguagem vazia.
- A única cadeia que a linguagem agora aceita é w, e, portanto, sua linguagem não será vazia se e somente se ela aceita w.

Problemas Indecidíveis

- **Prova**: Vamos escrever a máquina modificada descrita na idéia da prova utilizando nossa notação padrão.
- M_1 = "Sobre a entrada x:
    1. Se $x \neq w$, *rejeite*.
    2. Se $x = w$, rode M sobre a entrada w e aceite se M aceita."
- Essa máquina tem a cadeia w como parte de sua descrição.
- Ela conduz o teste $x = w$ de uma maneira óbvia, fazendo uma varredura na entrada e comparando-a caractere por caractere com w para determinar se elas são iguais.

Problemas Indecidíveis

- Juntando tudo, supomos que a MT R decide V_{MT} e construímos a MT S que decide A_{MT} da seguinte forma:
- "Sobre a entrada $\langle M, w \rangle$, uma codificação de uma MT M e uma cadeia w:
  1. Use a descrição de M e w para construir a MT M_1 descrita anteriormente.
  2. Rode R sobre a entrada $\langle M_1 \rangle$.
  3. Se R aceita, *rejeite*, se R rejeita, *aceite*.
- Note que S, na realidade, tem de ser capaz de computar uma descrição de M_1 a partir de uma descrição de M e de w.
- Ela é capaz de fazer isso facilmente, adicionando novos estados a M que realizem o teste $x = w$.