Turing e Complexidade

Fabio Gagliardi Cozman (baseado em material da disciplina PCS2214)

PMR2300
Escola Politécnica da Universidade de São Paulo

Máquina de Turing

- Máquina de Turing: modelo mais poderoso de computador, proposto pelo inglês Alan M. Turing em 1936.
- Similar a um autômato finito, porém com uma memória ilimitada e irrestrita, constituindo um modelo mais exato de um computador de propósito geral.

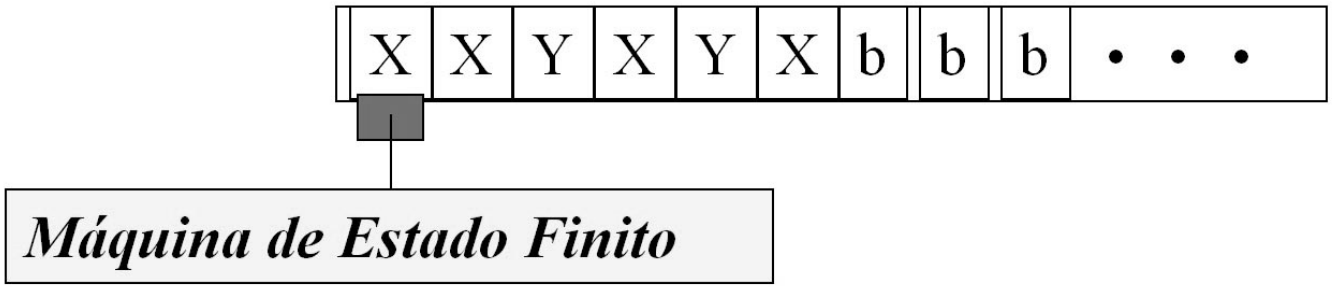
Alan Mathison Turing

Matemático, lógico, criptógrafo e herói de guerra britânico. Considerado o pai da ciência da computação. Fez previsões acerca da Inteligência Artificial e propôs o Teste de Turing, contribuindo para o debate sobre a consciência das máquinas e suas capacidades de pensar. Formalizou o conceito de algoritmo e computação com a Máquina de Turing, gerando sua versão da Tese de Church-Turing. Responsável pela quebra do código alemão Enigma durante a II Guerra Mundial. Depois da guerra, projetou um pioneiro computador digital programável eletronicamente. Foi processado e condenado por ser homossexual (em 1952). Morreu envenenado (provável suicídio). O Turing Award foi criado em sua homenagem.

Máquina de Turing

A Máquina de Turing consiste em:

1. Uma fita semi-infinita dividida em células, cada uma contendo um símbolo de um alfabeto finito (a fita é também a memória externa);
2. Um cursor que efetua leitura e escrita em uma célula e se move para a direita ou esquerda;
3. Uma máquina de estado finito, cujo conjunto de estados é S, e que controla o cursor;



Computação em uma MT

1. Inicialmente a fita contém somente a cadeia de entrada, disposta a partir da primeira célula da fita (fita semi-infinita), com o cursor posicionado no início da cadeia (o resto está em branco, b);
2. Para armazenar algo, a máquina escreve na fita;
3. Se tentar mover o cursor para a esquerda, estando na primeira célula da fita, o cursor não sai do lugar (fita semi-infinita);
4. As saídas aceita e rejeita são obtidas ao entrar nos estados de aceitação e rejeição;
5. Se não entrar em um estado de aceitação ou rejeição, continuará sua computação para sempre (loop infinito).

Definição Formal da Máquina de Turing

Uma $MT = (S, I, \Gamma, f, \sigma_0, \sigma_A, \sigma_R)$ é:

1. Um conjunto finito S de estados;
2. Um conjunto finito I de símbolos de entrada, $b \notin I$;
3. Um conjunto finito Γ de símbolos da fita, com $b \in \Gamma$e $I \subset \Gamma$;
4. Uma função f: $S \times \Gamma \rightarrow \Gamma \times S \times \{E, D\}$, sendo D:direita, E:esquerda;
5. $\sigma_0 \in S$ é o estado inicial;
6. $\sigma_A \in S$ é o estado de aceitação;
7. $\sigma_R \in S$ é o estado de rejeição, com $\sigma_A \neq \sigma_R$.

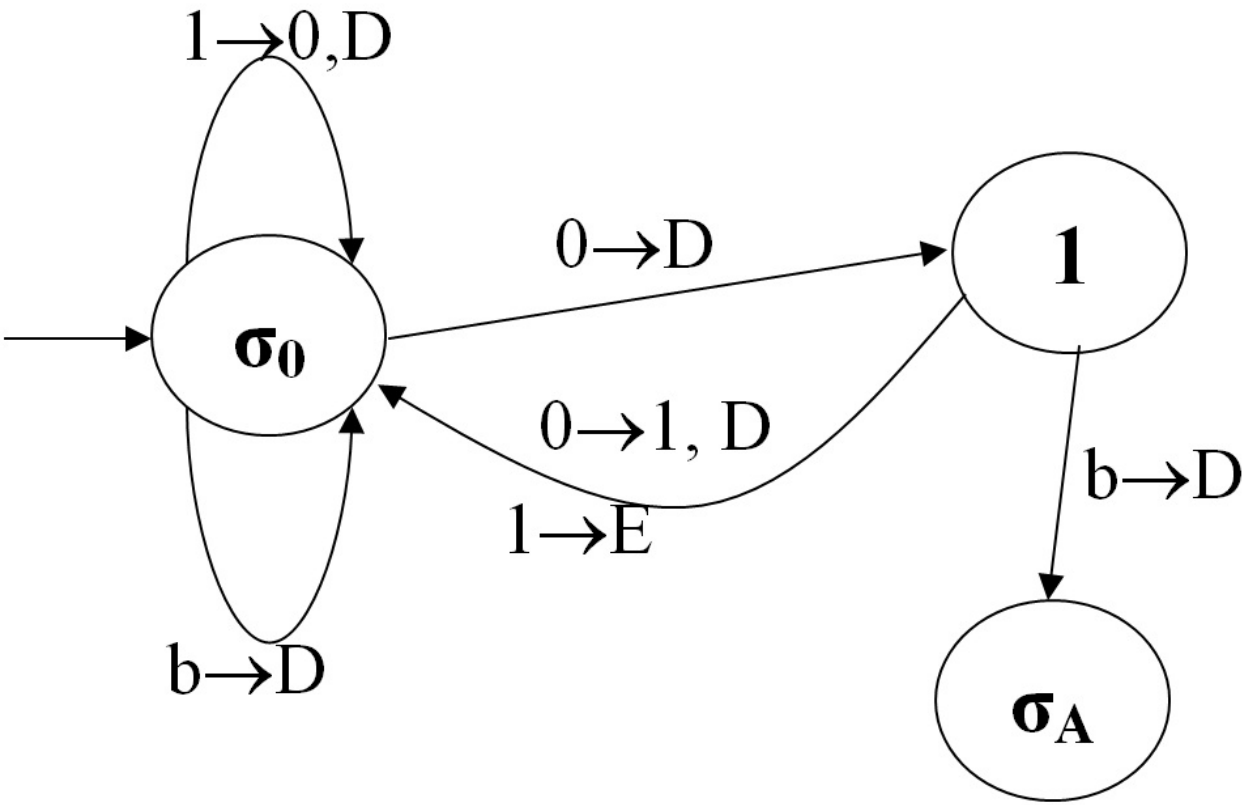
OBS: b: célula em branco da fita

Diagrama de Transições de uma MT

Seja $M = (S, I, \Gamma, f, \sigma_0, \sigma_A, \sigma_R)$ uma máquina de Turing. O diagrama de transições de M é um grafo orientado G com:

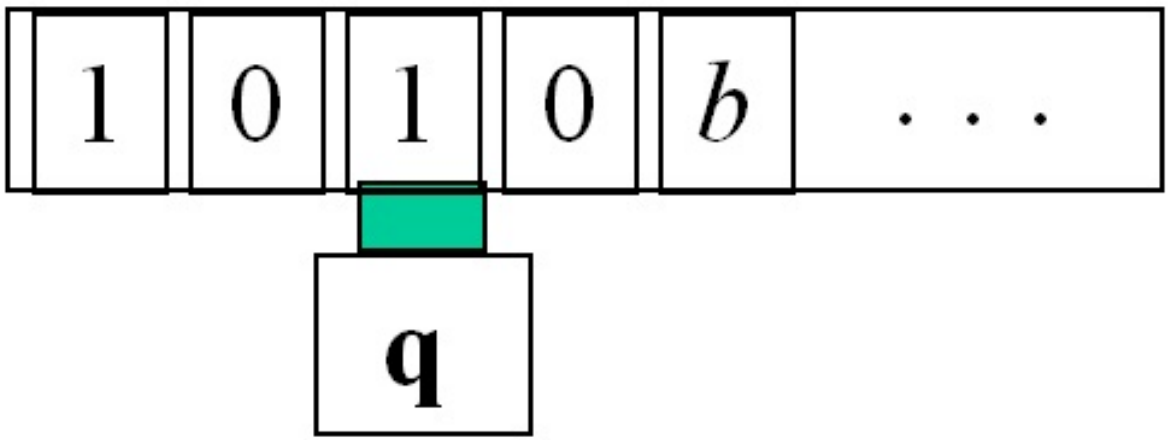
1. nós membros de S.
2. uma seta indica o estado inicial σ_0.
3. uma aresta orientada (σ_1,σ_2) existe em G se existir uma entrada de fita i $\in \Gamma$ com f($\sigma_1$1,i)=(σ_2, j, m), sendo σ_1,$\sigma_2 \in$ S, j $\in \Gamma$ e m indica o movimento do cursor, m $\in$ {E, D}. Neste caso, a aresta (σ_1,σ_2) é rotulada com i $\rightarrow$ j, m, para i $\neq$ j, e i $\rightarrow$ m se i = j.

Exemplo



Configuração de uma MT

1. Conforme a MT efetua sua computação, mudanças ocorrem no estado atual, no conteúdo da fita e na posição do cursor.
2. Uma situação na computação pode ser representada por uma configuração.
3. A configuração 10q10 representa:

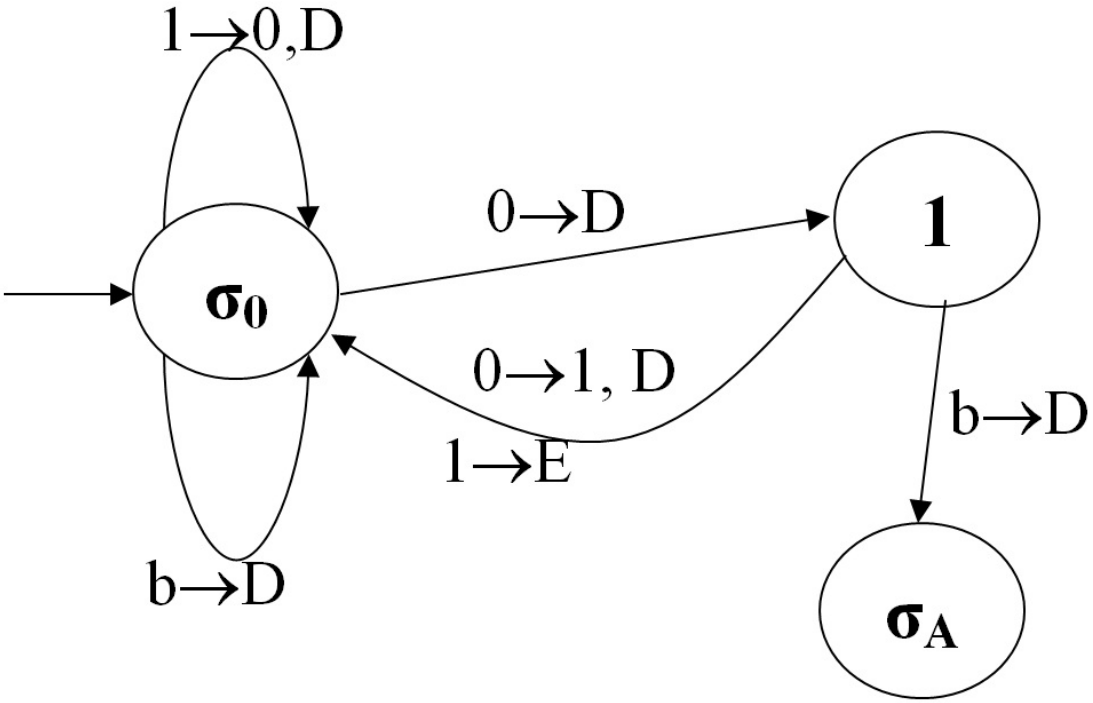


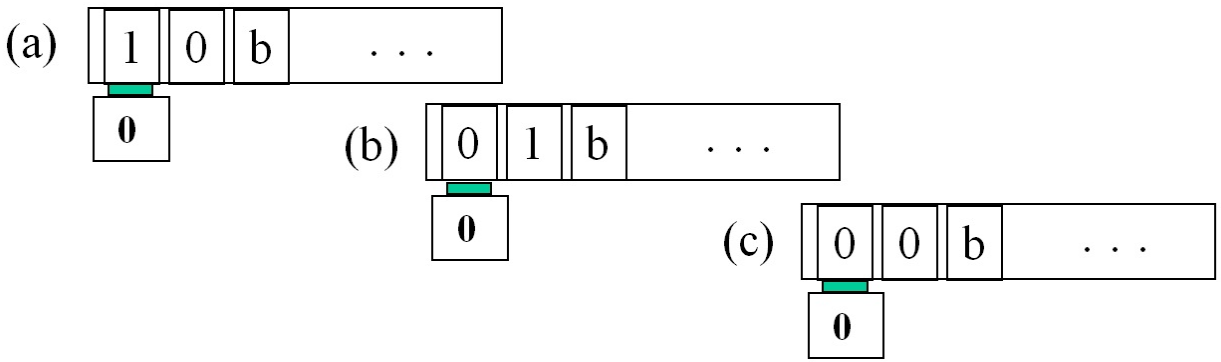
Aceitação de Cadeias na MT

1. Ao iniciar uma MT com uma entrada, pode-se: aceitar a entrada ou não aceitar a entrada.
2. Uma MT pode não aceitar uma entrada:
   1. ao entrar em σ_R e rejeitar a cadeia ou
   2. ao entrar em loop infinito e não parar.

OBS: Pode ser muito difícil distinguir uma MT que entrou em loop infinito de outra que está demorando para computar.

Exemplo: verfique funcionamento para cadeias:





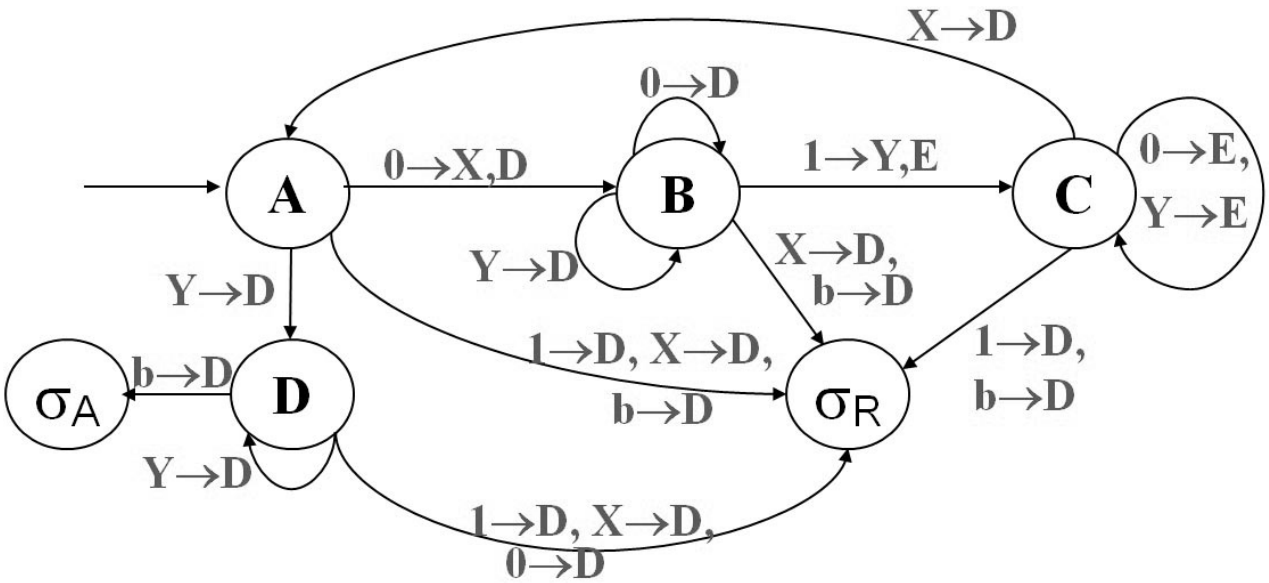
Exemplo de MT

Projetar uma Máquina de Turing M que aceita apenas cadeias $\{0^n1^n|n > 0\}$.
Algoritmo:

1. Se w está vazia, rejeite.
2. Leia a célula corrente. Se 0, troque por X e vá para o passo 3. Se Y, procure o final da cadeia de Ys e aceite. Senão, rejeite.
3. Vá para a direita da fita, procurando o primeiro 1. Se encontrar, troque-o por Y e vá para o passo 4. Senão, rejeite.
4. Vá para a esquerda na fita, procurando por um X. Se encontrar, volte a avançar na fita e vá para o passo 2. Senão, rejeite. ”

Exemplo de MT

Descrição formal de $M = (S, I, \Gamma, f, \sigma_0, \sigma_A, \sigma_R)$:
$S = \{A, B, C, D, \sigma_A, \sigma_R\}$, $\sigma_0 = A$, $I = \{0, 1\}$, $\Gamma = \{0, 1, X, Y, b\}$,
f = diagrama de transições:



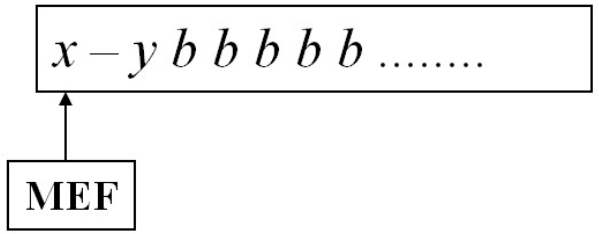
Funções

- Dada uma MT T e uma cadeia α, começamos com T na configuração inicial padrão em uma fita contendo α.
- Se T em algum momento pára deixando uma cadeia β na fita, podemos considerar β o valor de uma função avaliada em α.
- Assim, T(α) = β. O domínio da função T consiste de todas as cadeias α para as quais T, em algum momento, pára.

MT para Computar Funções

Exemplo: Projetar uma MT que calcule a subtração x - y, com x > y; x,y $\in$ N. Considere x e y em representação unária, fornecidos na fita da seguinte forma:

unária	**decimal**
1	0
11	1
111	2
1111	3
...	...



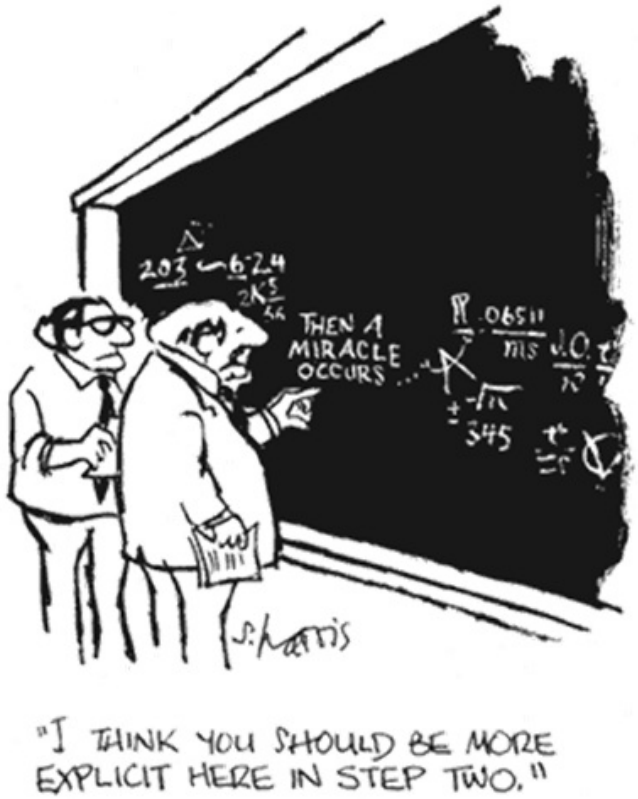
Ex: x=5, y=2 fita: 111111–111bbbb Resposta na fita: 1111bbbbb

Modifique sua máquina para aceitar x $\geq$ y.

Variantes da MT

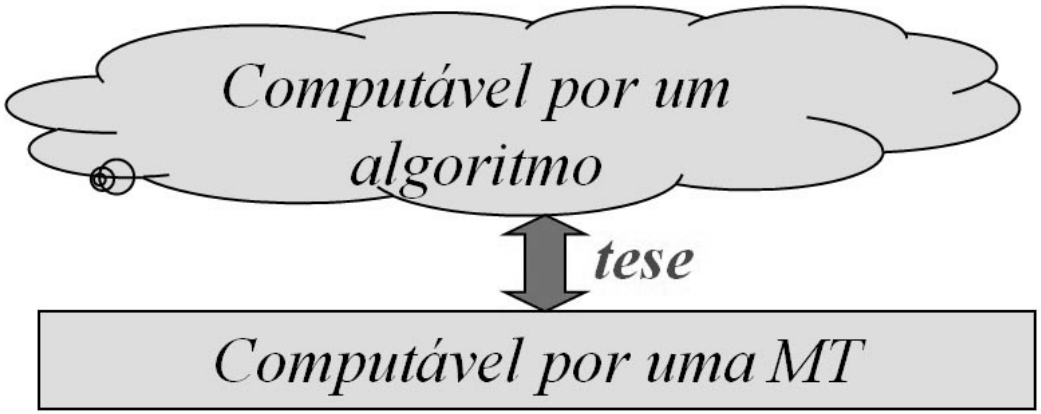
Existem inúmeros modelos variantes de MT, porém todos se mostraram equivalentes em termos de poder de computação (a classe de funções que todos descrevem é única).

Noção intuitiva de algoritmo: é um conjunto finito de instruções que podem ser executadas mecanicamente em tempo finito para resolver algum problema. Com dados de entrada apropriados ao problema, o algoritmo tem que parar e produzir a resposta correta.



Tese de Church-Turing

Tese de Church-Turing (1936):
”Qualquer função de teoria dos números é computável por um algoritmo se, e somente se, for computável por uma Máquina de Turing.”



Alonzo Church

Matemático americano, responsável por alguns dos fundamentos da CC. Nasceu em Washington, DC, recebeu BA e PhD na Universidade de Princeton, USA, em 1924 e 1927, respectivamente. Tornou-se professor em Princeton em 1929. Foi orientador de Stephen C. Kleene (1909-1994), entre outros. Também orientou Alan Turing de 1936 a1938. Seu trabalho mais conhecido é o desenvolvimento do cálculo-λ no seu famoso artigo de 1936, mostrando a existência de problemas indecidíveis. Ele e Turing mostraram que o cálculo-λ e a MT são equivalentes em capacidades, resultando na tese de Church-Turing. Como há disputa sobre "quem foi o primeiro", esta tese também é conhecida como tese de Church e tese de Turing.

Computabilidade

- Se encontrarmos problemas que as MTs não podem computar, “saberemos” que os computadores também não o poderão.
- Um problema é incomputável (ou insolúvel) quando não existe um algoritmo que possa executá-lo, isto é, não existe uma MT que possa computá-lo.

Computabilidade e Decisão

Um problema de decisão é um problema cuja solução é SIM ou NÃO.

Um problema indecidível é um problema de decisão incomputável.

Exemplo de Problema Indecidível: Problema da Parada

Uma MT T que começa com uma cadeia α, ou pára ou nunca pára.

Existe um algoritmo que decide, dada qualquer MT e qualquer cadeia α, se a MT começando com α vai parar?

O problema da parada de uma MT é indecidível.

Complexidade Computacional

1. Como modelo de computação, a MT nos forneceu um modo de provar a existência de problemas indecidíveis (incomputáveis).
2. A MT não só nos ajuda a encontrar os limites da computabilidade, como também nos ajuda a classificar problemas computáveis – os que têm um algoritmo que os resolvem – pela quantidade de trabalho necessário para se executar o algoritmo.

Complexidade Computacional

A complexidade de um problema é o consumo de tempo de um algoritmo ótimo ideal para solucionar o problema.

1. O consumo é medido em função do tamanho n da instância (cadeia de entrada).
2. Um algoritmo resolve um problema de decisão em tempo O(T(n)) se, dada uma entrada de tamanho n, o algoritmo produz a solução em, no máximo, T(n).

Considera-se, sem perda de generalidade, somente problemas de decisão.

Classe P

Um problema de decisão é dito solúvel em tempo polinomial, se existir um algoritmo para solucioná-lo em O(nk), para algum valor de k.

A classe de complexidade P é o conjunto dos problemas de decisão que são solúveis em tempo polinomial.

Verificação de Problemas

1. Um verificador testa se uma dada cadeia w é solução para um determinado problema.
2. Um problema é verificável em tempo polinomial se suas soluções puderem ser verificadas em tempo polinomial no tamanho de w.

MT não Determinística

1. Uma MT não determinística pode ter mais de uma ação que inicia com os mesmos (s,i): s: estado atual da MEF; i: símbolo lido na fita.
2. Seja N uma MT não determinística que é um decisor. O tempo de execução f(n) de N é o número máximo de passos que N usa, para qualquer ramificação de sua computação e para qualquer entrada de comprimento n.

Classe NP: definições equivalentes

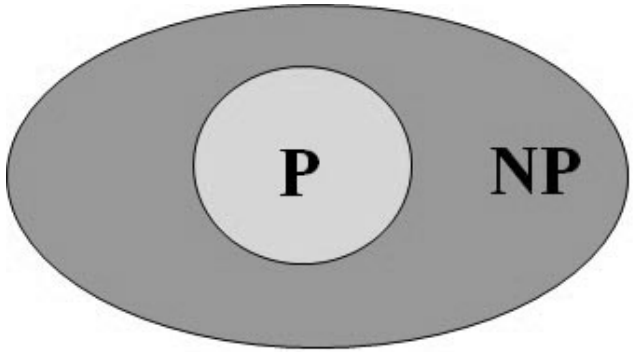
Um problema pertence à classe NP se e somente se ele puder ser decidido em tempo polinomial por alguma Máquina de Turing não determinística

Um problema pertence à classe NP se ele puder ser verificado em tempo polinomial (por uma MT determinística).

NP vem de Não determinística em tempo Polinomial

Classe P e Classe NP

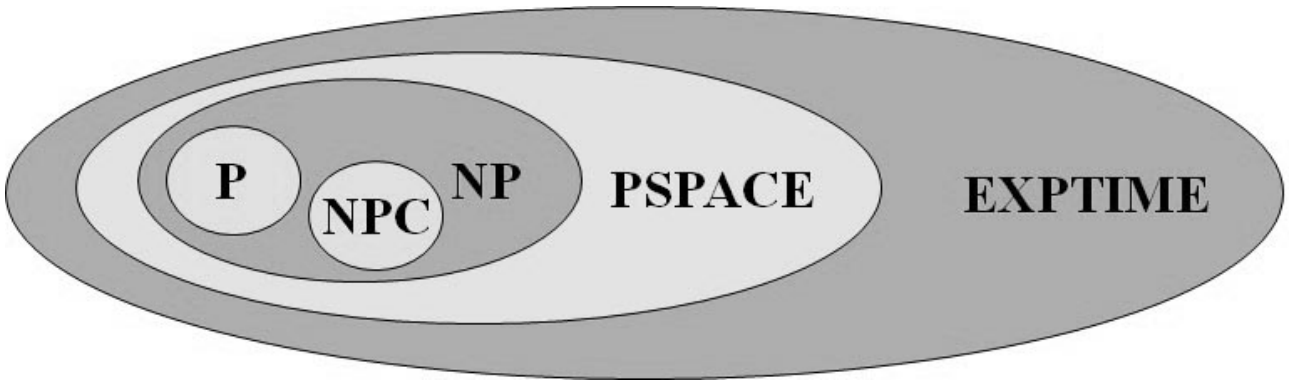
1. Todos os problemas de complexidade P são também de complexidade NP.



2. O inverso “parece” não ser verdade. Porém não se pode afirmar isso. Alguns problemas “parecem” não ter soluções passíveis de serem obtidas em tempo polinomial. Entretanto, os algoritmos podem simplesmente ainda não serem conhecidos!

Outras Classes de Problemas

PSPACE – problemas que podem ser decididos em espaço polinomial em uma MT determinística.
EXPTIME – problemas que podem ser resolvidos em tempo exponencial.
Conjecturas atuais sobre as relações entre as classes:



NPC: NP-completo

Bibliografia

Adaptado do Material da disciplina:
PCS 2214 - Fundamentos de Engenharia de Computação I
Escola Politécnica da USP - 2011
Professores:
Anarosa Alves Franco Brandão
Anna Helena Reali Costa
Ricardo Luis de Azevedo da Rocha
Ricardo Nakamura