Complexidade Parametrizada - Intratabilidade

Uéverton S. Souza - UFF

ueverton@ic.uff.br

1º Workshop Paulista em Otimização, Combinatória e Algoritmos

Junho 2017

Teoria da NP-completude

Classe P

Solucionáveis por uma máquina de Turing **determinística** em tempo polinomial.

Teoria da NP-completude

Classe NP

Solucionáveis por uma máquina de Turing **não determinística** em tempo polinomial.

Teoria da NP-completude

DTM e NDTM possuem a mesma capacidade de computação.
Entretanto a complexidade de tempo destas computações podem variar.

Teoria da NP-completude

DTM e NDTM possuem a mesma capacidade de computação.
Entretanto a complexidade de tempo destas computações podem variar.

$$P = NP?$$

Teoria da NP-completude

DTM e NDTM possuem a mesma capacidade de computação.
Entretanto a complexidade de tempo destas computações podem variar.

$$P = NP?$$

NP-completude

Teoria da NP-completude

DTM e NDTM possuem a mesma capacidade de computação.
Entretanto a complexidade de tempo destas computações podem variar.

$$P = NP?$$

NP-completude (reduções polinomiais)

Teoria da NP-completude

Classe NP

Solucionáveis por uma máquina de Turing **não determinística** em tempo polinomial.

$$\Updownarrow$$

Verificáveis por uma máquina de Turing **determinística** em tempo polinomial.

$$\Updownarrow$$

Admitem certificados que podem ser verficados por um algoritmo **determinístico** em tempo polinomial.

(Sipser, Introduction to the Theory of Computation)

O problema NP-completo genérico

Aceitação de uma máquina de Turing não determinística (ANDTM)

Instância: Uma máquina de Turing não determinística $\mathbb{M}$ que reconhece uma linguagem $\mathbb{L} \subseteq \Sigma^*$; uma palavra $x \in \Sigma^*$.

Questão: $\mathbb{M}$ aceita x?

- Todo problema Π em *NP* possui uma NDTM que reconhece a linguagem $\mathbb{L}_\Pi$ correspondente.

O problema NP-completo genérico

Aceitação de uma máquina de Turing não determinística (ANDTM)

Instância: Uma máquina de Turing não determinística $\mathbb{M}$ que reconhece uma linguagem $\mathbb{L} \subseteq \Sigma^*$; uma palavra $x \in \Sigma^*$.

Questão: $\mathbb{M}$ aceita x?

- Todo problema Π em *NP* possui uma NDTM que reconhece a linguagem $\mathbb{L}_\Pi$ correspondente. **(por definição)**

O problema NP-completo genérico

Aceitação de uma máquina de Turing não determinística (ANDTM)

Instância: Uma máquina de Turing não determinística $\mathbb{M}$ que reconhece uma linguagem $\mathbb{L} \subseteq \Sigma^*$; uma palavra $x \in \Sigma^*$.

Questão: $\mathbb{M}$ aceita x?

- Todo problema Π em NP possui uma NDTM que reconhece a linguagem $\mathbb{L}_\Pi$ correspondente. **(por definição)**
- Dada uma palavra x (instância) e um algoritmo verificador polinomial para Π, podemos construir uma NDTM $\mathbb{M}_\Pi^x$ em tempo polinomial com relação a $|x|$ tal que $\mathbb{M}_\Pi^x$ aceita x sse $x \in \mathbb{L}_\Pi$.

O problema NP-completo genérico

Aceitação de uma máquina de Turing não determinística (ANDTM)

Instância: Uma máquina de Turing não determinística $\mathbb{M}$ que reconhece uma linguagem $\mathbb{L} \subseteq \Sigma^*$; uma palavra $x \in \Sigma^*$.

Questão: $\mathbb{M}$ aceita x?

- Todo problema Π em *NP* possui uma NDTM que reconhece a linguagem $\mathbb{L}_\Pi$ correspondente. **(por definição)**
- Dada uma palavra x (instância) e um algoritmo verificador polinomial para Π, podemos construir uma NDTM $\mathbb{M}_\Pi^x$ em tempo polinomial com relação a $|x|$ tal que $\mathbb{M}_\Pi^x$ aceita x sse $x \in \mathbb{L}_\Pi$.
- $\Pi \propto ANDTM$, $\forall\ \Pi \in NP$.

Definições alternativas

Um circuito booleano de decisão consiste de variáveis booleanas de entrada, portas lógicas de negação, portas lógica E, portas lógicas Ou, e uma única porta lógica de saída.

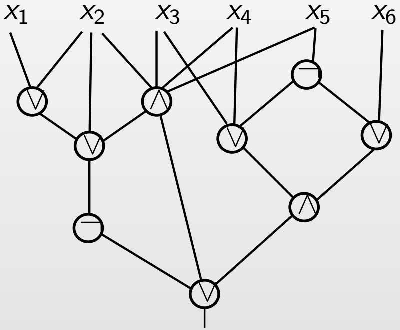


Figure: Exemplo de circuito booleano de decisão, onde ($\wedge$), ($\vee$) e ($\neg$) denotam portas lógicas E, Ou e de negação, respectivamente.

Definição alternativa para a classe P

Circuitos certificadores

Seja $\mathbb{L}_\Pi \subseteq \Sigma^*$.
$\mathbb{L}_\Pi \in P$ se e somente se for possível para cada palavra $x \in \Sigma^*$ construir em tempo polinomial com relação a $|x|$ um ciruito lógico com $|x|$ variáveis de entrada tal que:

- fornecido como entrada para C o valor da string x
  - a saída de C é igual a 1 se $x \in \mathbb{L}_\Pi$
  - a saída de C é igual a 0 se $x \notin \mathbb{L}_\Pi$.

John E. Savage,
Models of Computation - Exploring the power of Computing

Definição alternativa para a classe NP

Circuitos certificadores

Seja $\mathbb{L}_\Pi \subseteq \Sigma^*$.
$\mathbb{L}_\Pi \in NP$ se e somente se for possível para cada palavra $x \in \Sigma^*$ construir em tempo polinomial com relação a $|x|$ um ciruito lógico com $|x| + c$ variáveis de entrada tal que:

- para alguma sequencia s de c bits, ao fornecer como entrada para C o valor da string $|x|+s$
  - a saída de C é igual a 1 se $x \in \mathbb{L}_\Pi$
  - a saída de C é igual a 0 se $x \notin \mathbb{L}_\Pi$.

c é um limite superior para o tamanho de um certificado necessário para uma palavra de tamanho $|x|$.

NP-completo por definição

Satisfabilidade de Circuitos (SC)

Instância: Um circuito lógico C com n variáveis booleanas de entrada.

Questão: Existe uma atribuição de valores para as variáveis de entrada de C que o faça retornar 1?

Complexidade de Circuitos

- Circuitos podem ser construídos para computar funções.

Complexidade de Circuitos

- Circuitos podem ser construídos para computar funções.
- Podemos dizer que uma função f é mais complexa que uma função g, se o circuito necessário para computar f é mais complexo que o circuito necessário para computar g.

Complexidade de Circuitos

- Circuitos podem ser construídos para computar funções.
- Podemos dizer que uma função f é mais complexa que uma função g, se o circuito necessário para computar f é mais complexo que o circuito necessário para computar g.
- A complexidade de um circuito é medida pelo(a):
  - tamanho do circuito;
  - profundidade do circuito.

Complexidade de Circuitos

- Circuitos podem possuir:
  - portas lógicas restritas (apenas duas entradas "**fan-in**")
  - portas lógicas largas (quantidade arbitrária de entradas "**fan-in**")

Complexidade de Circuitos

- Circuitos podem possuir:
  - portas lógicas restritas (apenas duas entradas “**fan-in**”)
  - portas lógicas largas (quantidade arbitrária de entradas “**fan-in**”)
- Se um circuito C possui profundidade constante e apenas portas lógicas restritas, então o número de variáveis de entrada de C é constante.

Complexidade de Circuitos

- Circuitos podem possuir:
  - portas lógicas restritas (apenas duas entradas "**fan-in**")
  - portas lógicas largas (quantidade arbitrária de entradas "**fan-in**")
- Se um circuito C possui profundidade constante e apenas portas lógicas restritas, então o número de variáveis de entrada de C é constante.
- Se um circuito C com n variáveis de entrada possui profundade constante, então C possui portas lógicas largas.

Complexidade de Circuitos

- Circuitos podem possuir:
  - portas lógicas restritas (apenas duas entradas "**fan-in**")
  - portas lógicas largas (quantidade arbitrária de entradas "**fan-in**")
- Se um circuito C possui profundidade constante e apenas portas lógicas restritas, então o número de variáveis de entrada de C é constante.
- Se um circuito C com n variáveis de entrada possui profundade constante, então C possui portas lógicas largas.
- Dados dois circuitos C' e C'' de tamanho polinomial e profundidade constante que computam a mesma função:

Complexidade de Circuitos

- Circuitos podem possuir:
  - portas lógicas restritas (apenas duas entradas "**fan-in**")
  - portas lógicas largas (quantidade arbitrária de entradas "**fan-in**")
- Se um circuito C possui profundidade constante e apenas portas lógicas restritas, então o número de variáveis de entrada de C é constante.
- Se um circuito C com n variáveis de entrada possui profundade constante, então C possui portas lógicas largas.
- Dados dois circuitos C' e C'' de tamanho polinomial e profundidade constante que computam a mesma função:
  - Se a quantidade de portas lógicas largas de C' é menor que a quantidade de portas lógicas largas de C'', dizemos que C' é menos complexo que C''.

Teoria da Intratabilidade Parametrizada

Baseado em que podemos formular uma teoria de intratabilidade parametrizada?

Teoria da Intratabilidade Parametrizada

Baseado em que podemos formular uma teoria de intratabilidade parametrizada?

- Qual será a ferramenta a ser utilizada para demonstrar pertinência e dificuldade?

Teoria da Intratabilidade Parametrizada

Baseado em que podemos formular uma teoria de intratabilidade parametrizada?

- Qual será a ferramenta a ser utilizada para demonstrar pertinência e dificuldade?
  R. Reduções, FPT-reduções (reduções em tempo FPT).
- Qual é o problema parametrizado de partida?

Candidatos a problemas de partida

Candidatos a problemas de partida

Opção 1:

Aceitação de uma máquina de Turing não determinística(k)

Instância: Uma máquina de Turing não determinística $\mathbb{M}$ que reconhece uma linguagem $\mathbb{L} \subseteq \Sigma^*$; uma palavra $x \in \Sigma^*$.

Parâmetro: Um inteiro positivo k.

Questão: $\mathbb{M}$ aceita x com no máximo k passos?

Candidatos a problemas de partida

Opção 1:

Aceitação de uma máquina de Turing não determinística(k)

Instância: Uma máquina de Turing não determinística $\mathbb{M}$ que reconhece uma linguagem $\mathbb{L} \subseteq \Sigma^*$; uma palavra $x \in \Sigma^*$.
Parâmetro: Um inteiro positivo k.
Questão: $\mathbb{M}$ aceita x com no máximo k passos?

Classe U

Candidatos a problemas de partida

Opção 2:

Candidatos a problemas de partida

Opção 2:

Satisfabilidade Ponderada de Circuitos (WCS)

Instância: Um circuito lógico C com n variáveis booleanas de entrada.
Parâmetro: Um inteiro k.
Questão: Existe uma atribuição de peso k para as variáveis de entrada de C que o faça retornar 1?

(O peso de uma atribuição é o número de 1's atribuídos às variáveis de C)

Candidatos a problemas de partida

Opção 2:

Satisfabilidade Ponderada de Circuitos (WCS)

Instância: Um circuito lógico C com n variáveis booleanas de entrada.
Parâmetro: Um inteiro k.
Questão: Existe uma atribuição de peso k para as variáveis de entrada de C que o faça retornar 1?

(O peso de uma atribuição é o número de 1's atribuídos às variáveis de C)

Classe W

FPT-redução

FPT-redução

Seja $\Pi(k)$ e $\Pi'(k')$ dois problemas parametrizados, onde $k' \leq g(k)$.

Uma FPT-redução (ou transformação paramétrica) de $\Pi(k)$ para $\Pi(k')$ é uma transformação R tal que:

1. Para todo x, temos que $x \in \Pi(k)$ se e somente se $R(x) \in \Pi'(k')$;
2. R é computável por um *FPT*-algoritmo (com relação a k);

Se uma *FPT*-redução existe entre Π e Π' então Π é transformado (ou se reduz) parametricamente a Π'.

FPT-redução

Lema

(transitividade) *Dado dois problemas parametrizados Π, Π' and Π'', se Π se reduz parametricamente a Π' e Π' se reduz parametricamente a Π'' então Π se reduz parametricamente a Π''.*

FPT-redução

Lema

(preservação da tratabilidade por parâmetro fixo) *Dado dois problemas Π e Π', se Π se reduz parametricamente a Π' e Π' é tratável por parâmetro fixo então Π é tratável por parâmetro fixo.*

Classe U $\times$ classe W

Qual a relação entre essas classes?

Classe U $\times$ classe W

Qual a relação entre essas classes?

Difícil analisar profundamente, talvez seja melhor refinar a classe W.

Conceitos de Circuito

Definição

Seja C um circuito booleano de decisão com variáveis de entrada $x_1, \ldots, x_n$.

- *O* ***entrelaçamento*** *de C é definido como o número máximo de portas lógicas largas em qualquer caminho da variável de entrada até a linha de saída (Uma porta é denominada* ***larga*** *se suas entradas excedem algum limite constante, em geral dois).*
- *A* ***profundidade*** *de C é definida como o comprimento do maior caminho de uma variável de entrada até a linha de saída em C.*
- *O* ***peso*** *de uma atribuição às variáveis de um circuito booleano C (uma atribuição para C) é o número de $1's$ nesta atribuição.*

W-hierarquia

Satisfabilidade Ponderada em Circuitos de Entrelaçamento t e Profundidade h WCS(t,h)

Instância: Um circuito de decisão C com entrelaçamento t e profundidade h.
Parâmetro: Um inteiro positivo k.
Questão: C possui uma atribuição satisfatível com peso k?

W-hierarquia

Satisfabilidade Ponderada em Circuitos de Entrelaçamento t e Profundidade h WCS(t,h)

Instância: Um circuito de decisão C com entrelaçamento t e profundidade h.
Parâmetro: Um inteiro positivo k.
Questão: C possui uma atribuição satisfatível com peso k?

Definição

Um problema parametrizado Π pertence à classe W[t] se e somente se Π se FPT-reduz a WCS(t,h), para alguma constante h.

A classe W[P]

Definição

Um problema parametrizado Π *pertence à classe W[P] se e somente se* Π *se FPT-reduz ao problema WCS(0,h), para algum valor árbitrário h.*

A classe W[P]

Definição

Um problema parametrizado Π *pertence à classe W[P] se e somente se* Π *se FPT-reduz ao problema WCS(0,h), para algum valor árbitrário h.*

A classe que definimos anteriormente como W, agora será chamada de W[P].

Intratabilidade

Agora apresentaremos o análogo parametrizado da classe NP na teoria de NP-completude.

O análogo ao teorema de Cook

O análogo ao teorema de Cook

Aceitação de uma máquina de Turing não determinística(k)

Instância: Uma máquina de Turing não determinística $\mathbb{M}$ que reconhece uma linguagem $\mathbb{L} \subseteq \Sigma^*$; uma palavra $x \in \Sigma^*$.
Parâmetro: Um inteiro positivo k.
Questão: $\mathbb{M}$ aceita x com no máximo k passos?

O análogo ao teorema de Cook

Aceitação de uma máquina de Turing não determinística(k) pode ser trivialmente resolvido em tempo $O(n^{k+1})$, onde n denota o tamanho total da entrada.

Isso é feito explorando-se exaustivamente todos os caminhos computacionais de k passos. Acredita-se que este resultado não possa ser significativamente melhorado.

O análogo ao teorema de Cook

Teorema

(Análogo ao Teorema de Cook)

1 **Aceitação da máquina de Turing não determinística**(k) *é W[1]-completo.*

W-hierarquia

Definição

(W-hierarquia) *A união das classes $W[t]$ juntamente com a classe $W[P]$, denota-se W-hierarquia.*

W-hierarquia

Definição

(W-hierarquia) *A união das classes $W[t]$ juntamente com a classe $W[P]$, denota-se W-hierarquia.*

$W[P]$ ***(antiga classe W)*** *denota a classe obtida por considerar nenhuma restrição sobre profundidade.*

W-hierarquia

Definição

(W-hierarquia) A união das classes $W[t]$ juntamente com a classe $W[P]$, denota-se W-hierarquia.

$W[P]$ (antiga classe W) denota a classe obtida por considerar nenhuma restrição sobre profundidade.

A classe U definida inicialmente é igual a classe $W[1]$.
Portanto, a W-hierarquia é:

$$FPT \subseteq W[1] \subseteq W[2] \subseteq \ldots \subseteq W[P].$$

W-hierarquia

Definição

(W-hierarquia) *A união das classes $W[t]$ juntamente com a classe $W[P]$, denota-se W-hierarquia.*

$W[P]$ ***(antiga classe W)*** *denota a classe obtida por considerar nenhuma restrição sobre profundidade.*

A classe U *definida inicialmente é igual a classe $W[1]$.*
Portanto, a W-hierarquia é:

$$FPT \subseteq W[1] \subseteq W[2] \subseteq \ldots \subseteq W[P].$$

Downey e Fellows conjecturaram que cada uma dessas relações de inclusão na W-hierarquia é própria.

W-hierarquia

- Se $P = NP$ então $FPT = W[P]$.

W-hierarquia

- Se $P = NP$ então $FPT = W[P]$.
- Se $FPT \neq W[1]$ então $P \neq NP$

W-hierarquia

- Se $P = NP$ então $FPT = W[P]$.
- Se $FPT \neq W[1]$ então $P \neq NP$
- E se $FPT = W[P]$? O que acontece?

W-hierarquia

- Se $P = NP$ então $FPT = W[P]$.
- Se $FPT \neq W[1]$ então $P \neq NP$
- E se $FPT = W[P]$? O que acontece?
- E se $FPT = W[1]$? O que acontece?

Hipótese de Tempo Exponencial

A Hipótese de Tempo Exponencial (ETH) afirma que 3-SAT não pode ser solucionado em tempo subexponencial.

Se verdadeira ETH implicaria que $P \neq NP$. No entanto, essa hipótese é uma afirmação mais forte.

ETH pode ser usada para mostrar que problemas computacionais são equivalentes em complexidade, no sentido de que se um deles admite um algoritmo subexponencial então os demais também admitiriam.

W-hierarquia

- Se $P = NP$ então $FPT = W[P]$.
- Se $FPT \neq W[1]$ então $P \neq NP$
- E se $FPT = W[P]$? O que acontece?
- E se $FPT = W[1]$? O que acontece? $\Rightarrow$ **ETH falha**

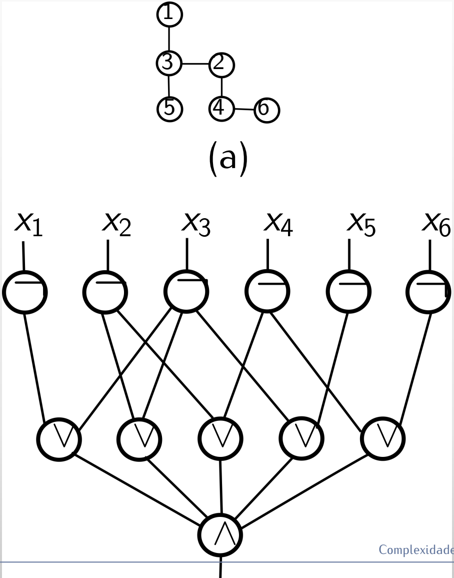
W-hierarquia

- Se $P = NP$ então $FPT = W[P]$.
- Se $FPT \neq W[1]$ então $P \neq NP$
- E se $FPT = W[P]$? O que acontece? $\Rightarrow$ **ETH falha**
- E se $FPT = W[1]$? O que acontece? $\Rightarrow$ **ETH falha**

Exemplos

Conjunto Independente(c)
Instância: Um grafo $G = (V, E)$.
Parâmetro: Um inteiro positivo c.
Questão: G possui um conjunto de vértices I, tal que $|I| \geq c$ e I não contém nenhum par de vértices adjacentes?

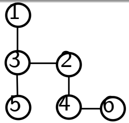
Exemplos



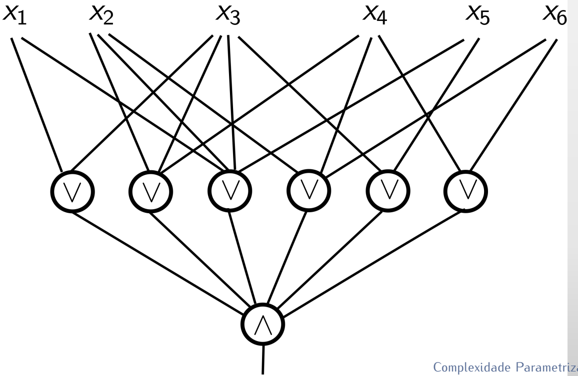
Exemplos

Conjunto Dominante(k)
Instância: Um grafo $G = (V, E)$.
Parâmetro: Um inteiro positivo k.
Questão: G possui um conjunto de vértices D, tal que $|D| \leq k$ e todo vértice $v \in V \setminus D$ é adjacente a pelo menos um vértice em D?

Exemplos



(a)



W-hierarquia

Observação

Uma alternativa para mostrar que um problema parametrizado Π pertence à classe $W[t]$, $t \geq 1$, é apresentar uma FPT-redução para algum problema parametrizado pertencente a $W[t]$.

W-hierarquia

Observação

Na complexidade parametrizada, definimos $W[t]$***-dificuldade*** *e* $W[t]$***-completude*** *de um problema parametrizado* $\Pi(k)$ *com relação à classe de complexidade* $W[t]$ *(*$t \geq 1$*), como na teoria da complexidade clássica:*

$\Pi(k)$ *é* $W[t]$*-difícil sob FPT-reduções se todo problema em* $W[t]$ *se FPT-reduz a* $\Pi(k)$*;*

$\Pi(k)$ *é* $W[t]$*-completo sob FTP-reduções se* $\Pi(k) \in W[t]$ *e* $\Pi(k)$ *é* $W[t]$*-difícil.*

FPT-reduções × Reduções de Karp

COBERTURA POR VÉRTICES $\in$ FPT

CONJUNTO INDEPENDENTE é W[1]-completo

CLIQUE é W[1]-completo

CONJUNTO DOMINANTE é W[2]-completo

Obrigado!

Perguntas?