Intratabilidade

Andr Lus Mendes Fakhoury
Eduardo Dias Pennone
Gustavo Vincius Vieira Silva Soares
Matheus Steigenberg Populim
Thiago Preischadt Pinheiro

SCC0205 - Teoria da Computao e Linguagens Formais
Prof. Diego Raphael Amancio

1 Introduo

2 Computao eficiente

3 Classes P e NP

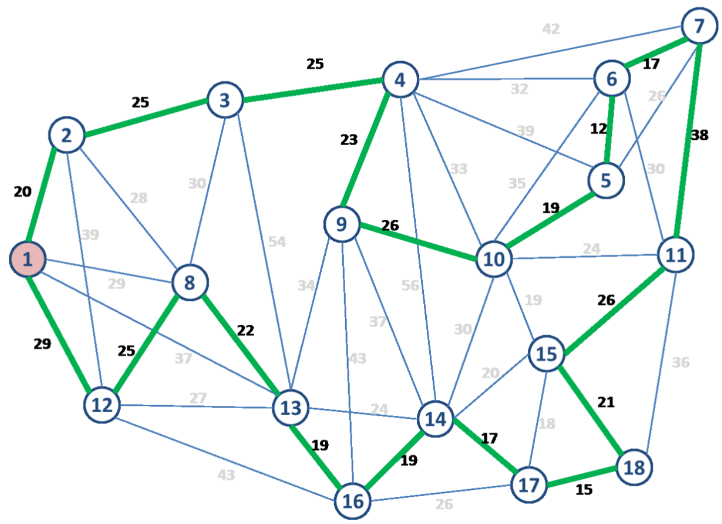
4 Hipteses P $\neq$ NP

5 Redues em tempo polinomial

6 Problemas NP-Difceis

6.1 Exemplos de problemas NP-difceis

Caixeiro viajante (TSP) - verso de otimizao: dado um grafo $G(V, A)$, encontrar um caminho que visite todas os vrtices exatamente uma vez, e que volte ao vrtice de origem, de forma a minimizar a distncia total (soma das arestas).



Exemplo com soluo para o caixeiro viajante [6]

7 Problemas NP-Completos

Durante a dcada de 1970, Stephen Cook e Leonid Levin propuseram, independentemente, um importante avano na questo de P vs. NP. Eles descobriram alguns problemas pertencentes a NP em que a complexidade individual est relacionada com toda a classe - e, caso um algoritmo em tempo polinomial exista para algum destes problemas em especfico, ento todos os problemas em NP tambm podem ser resolvidos em tempo polinomial. Estes *problemas em especfico* so denotados **NP-completos**. Por esta importante caracterstica, os estudos neste campo podem ser focados nos problemas *NP-completos*.

Definio (NP-Completo). Uma linguagem B dita *NP-completa* caso satisfaa estas duas caractersticas:

1. B est em NP;
2. Todo A em NP redutvel em tempo polinomial para B;

Tambm pode-se utilizar a classe *NP-difcil* para definir os problemas *NP-completos*: um problema *NP-completo* *NP-difcil* e est em *NP*. Outro teorema que ser til para definir se uma linguagem C *NP-completa* a seguinte:

Teorema (NP-Completo). Se B *NP-completo* e $B \leq_p C$ (B redutvel em tempo polinomial para C) para C em NP, ento C *NP-completo*.

Assim, ao se conhecer um problema *NP-completo*, pode-se utilizar o conceito de *reduo em tempo polinomial* para mostrar que outros problemas so *NP-completos*. A prova deste teorema a seguinte: j se sabe que C est em NP, ento, pela definio, basta provar que todo A em NP redutvel em tempo polinomial para C. Como B *NP-completo*, todo A redutvel em tempo polinomial para B. Como todo A redutvel para B e B redutvel para C, tem-se que todo A também pode ser redutvel para C.

Ainda assim, necessrio um passo inicial: provar que um problema *NP-completo* sem serem conhecidos outros problemas *NP-completos* (em que se poderia utilizar reduo polinomial para facilitar a prova). O primeiro problema demonstrado NP-completo foi o problema SAT (satisfatibilidade booleana), por Cook e Levin, que ser discutido no prximo tpico.

7.1 Exemplos de problemas NP-completos

Sero aqui discutidos alguns exemplos de problemas definidos como *NP-completos*.

Satisfatibilidade booleana (SAT): verificar se existe uma valorao para as variveis que compe uma frmula booleana, de forma que esta tenha sada verdadeira.

O SAT , historicamente, o primeiro problema provado como *NP-completo*. Outro ponto interessante que este problema *auto-redutvel*, ou seja, o algoritmo que resolve o problema (e verifica se existe tal composio de valores para variveis) tambm consegue encontrar a atribuio de variveis satisfatrias. O teorema que afirma que o SAT NP-completo chamado **teorema de Cook-Levin**, e, devido sua complexidade, sua prova no ser abordada neste trabalho.

3SAT: problema SAT com expresses na forma normal conjuntiva, e cada clusula contm exatamente trs variveis.

Pode-se provar que o 3SAT NP-Completo a partir do problema SAT original: possvel deduzir, em tempo polinomial, qualquer caso de SAT para um caso equivalente em 3SAT. Alm disso, pode-se afirmar que 3SAT NP, pois, dada uma coleo de clusulas e uma valorao para as variveis, possvel verificar em tempo polinomial se estas clusulas so satisfeitas (basta percorrer as clusulas e atribuir os valores das variveis, verificando se a resposta verdadeira).

Para fazer esta deduo de caso SAT para 3SAT, a expresso primeiramente denotada na forma normal conjuntiva (conjuno de disjunes). Com a expresso nesta forma, possvel reduzir clusulas com mais de trs termos em duas ou mais clusulas de 3 termos: por exemplo, a disjuno $(A \vee B \vee C \vee D)$ pode ser reduzida introduzindo uma nova varivel E (sem importncia para a expresso original) para $(A \vee B \vee E) \wedge (\neg E \vee C \vee D)$.

Outro problema de satisfatibilidade o problema 2SAT. Este, caso as frmulas estiverem restringidas forma normal conjuntiva (conjuno de disjunes), pode ser resolvido em tempo polinomial (por exemplo, utilizando-se componentes fortemente conexas).

Clique: dado um grafo $G(V, A)$ e um inteiro k, encontrar se existe um subgrafo de k ou mais vrtices, em que todos ns dois a dois so conectados por uma arestas - ou seja, se G possui um conjunto de k ns mutualmente adjacentes.

Precisa-se provar, inicialmente, que o clique NP: possvel construir um verificador que recebe um grafo $G(V, A)$, um inteiro k e um conjunto S. Ser feita a verificao se S possui tamanho maior ou igual a k, e, ento, verificar se para cada par de vrtices $(u, v) \in S$ existe uma aresta $(u, v) \in A$. Esta verificao pode ser feita em $\mathcal{O}(|S|^2)$.

Agora, prova-se que o clique *NP-difcil*. Para isso, pode-se mostrar que existe uma reduo do problema *3SAT* para o problema *clique*. Ser feita a construo de uma funo que recebe qualquer instncia do 3SAT e retorna uma instncia de Clique, que verdadeira se e somente se a instncia do 3SAT for verdadeira. Dado uma expresso booleana (na forma 3SAT) com clusulas $c_1, \ldots, c_m$ e variveis $x_1, \ldots, x_n$, pode-se construir um grafo $G(V, A)$ com $|V| = 3m$ da seguinte forma:

- Para cada clusula $c_j = (l_1^j \vee l_2^j \vee l_3^j)$ constri-se 3 vrtices l_1^j, l_2^j, l_3^j
- Para cada par l_i^j, l_k^t adicionada uma aresta entre eles se e somente se:
  - As clusulas j e t so diferentes;
  - $l_i^j \neq \neg l_k^t$ (so consistentes)

Como cada um dos trs vrtices de uma mesma clusula no esto ligados entre si, cada clique contm, no mximo, um deles. Ou seja, o tamanho mximo possvel para um clique m. E, um clique de tamanho m existe se e somente se o 3SAT for satisfatvel, pelos seguintes itens [4]:

- Caso exista um clique, os vrtices envolvidos no clique no possuem conflitos com outros (pois sero consistentes). Assim, pode-se designar os valores para cada varivel. Como tem-se apenas um literal por clusula, cada clusula ter um literal satisfeito, tornando isso uma valorao satisfatvel.
- Caso tenha uma valorao satisfatvel para as variveis, cada clusula possui um literal satisfeito, e, como nenhum destes literais tem conflito com nenhum outro, existem arestas entre eles, dando o clique de tamanho m.

Cobertura de vrtices (*Vertex-Cover*) - verso de deciso: dado um grafo $G(V, A)$ e um inteiro k, encontrar se existe um subconjunto de vrtices V' de tamanho k tal que toda aresta em A esteja conectada em no mnimo um vrtice em V'.

Para se provar que este problema NP-Completo, primeiramente, mostra-se que este problema NP. Dado um subconjunto V', pode-se verificar com complexidade polinomial se este um subconjunto correto (basta percorrer por todas as arestas e verificar se algum de seus vrtices pertence ao conjunto V'), logo, NP. Aps isso, pode-se tambm mostrar que o problema do Clique pode ser reduzido em tempo polinomial para um exemplo de problema de cobertura de vrtices - esta prova no ser demonstrada neste trabalho.

Circuito hamiltoniano (HAMPATH): verifica se existe um circuito hamiltoniano entre s, t em um grafo G.

O problema do circuito hamiltoniano consiste em, dado um grafo $G(V, A)$ e dois vrtices $s, t \in V$, existe um caminho de s at t que passa por todos os vrtices $\in V$ exatamente uma vez. Para provar este teorema, primeiramente precisa-se provar que ele NP. Isto pode ser verificado pois, dado um circuito C, pode-se facilmente verificar se este um circuito vlido com complexidade polinomial $\mathcal{O}(|V|+|A|)$. Agora, resta provar que existe um problema NP-completo que possua uma reduo em tempo polinomial para o *HAMPATH*. Pode-se escolher, por exemplo, o 3SAT para realizar esta prova - que tambm no ser demonstrada neste trabalho.

Caixeiro viajante (TSP) - verso de deciso: dados um grafo $G(V, A)$ e uma distncia L, retornar se existe um ciclo (que visite todos os vrtices exatamente uma vez) com distncia total no mximo L.

Embora a variao de otimizao do problema seja “apenas” considerada *NP-difcil* (como visto anteriormente), pode-se definir a verso de deciso do TSP como *NP-completa.* Para provar que esta variao NP, dado um grafo $G(V, A)$, um inteiro L e um ciclo c, pode-se verificar se o ciclo c de fato um ciclo vlido neste grafo, e se a soma de suas arestas no ultrapassa o limite L - esta verificao feita polinomialmente, podendo ser feita com complexidade $\mathcal{O}(|c|)$.

Para provar que esta variao NP-difcil, pode-se construir uma reduo em tempo polinomial de outro problema NP-completo, o *problema do ciclo hamiltoniano* - ou seja, mostrar que HamCycle $\leq_p$ TSP. Para isso, seleciona-se que qualquer instncia $G(V, A)$ do ciclo hamiltoniano. Objetiva-se montar uma instncia do TSP, em que o grafo $G'(V', A')$ do TSP montado da seguinte forma:

- $V' = V$ (utiliza-se os mesmos vrtices)
- $E' = \{(u, v) : u, v \in V \ \ \& \ \ u \neq v\}$

E o custo de cada aresta e em E' calculado da seguinte forma:

$$c(e) = \begin{cases} 0, \text{ se } e \in E \\ 1, \text{ se } e \notin E \end{cases}$$

Agora, suponha que um ciclo hamiltoniano h exista em G. O custo de cada aresta de h em G' 0, visto que todas as arestas esto no conjunto E. Desta forma, h tem custo 0 em G'. De maneira anloga, assume-se que G' tem um circuito h' de custo no mximo 0 - desta forma, as arestas em E' escolhidas tem todas custo 0. Isto significa que todas as arestas escolhidas em h' tambm pertencem ao conjunto E. Assim, pde-se concluir uma reduo (em tempo polinomial) do problema do ciclo hamiltoniano para esta verso do TSP, provando-se que esta variao do TSP NP-difcil.

Outros exemplos

Tambm existem vrios outros problemas clssicos *NP-completos* (ou com variaes que so), como, por exemplo, colorao completa em grafos (nmero acromtico), conjunto independente mximo, *Steiner tree* (rvore geradora mnima para um subconjunto de vrtices de um grafo), *bin packing*, soma de subconjuntos e batalha naval. Existem diversos livros e artigos listando mais diversos problemas assim classificados - um dos mais importantes artigos deste tema foi feito por Richard Karp [3].

8 Concluso

A teoria de NP-completude garante diversas formas de provar que determinados problemas so "to difceis"quanto vrios outros estudados. Por exemplo, se certa pessoa est tentando resolver um problema, e este problema pode ser provado como *NP-completo*, ela pode ficar mais tranquila caso no consiga achar uma soluo (pois, at hoje, esta soluo ainda no foi encontrada). Por isso, o estudo de complexidade e intratabilidade de suma importncia no estudo de computao, e a separao dos problemas em classes de acordo com suas possibilidades de serem resolvidos em tempo polinomial ou no (dilemas P vs. NP) muito interessante.

Referências

[1] Michael R. Garey and David S. Johnson. *Computers and Intractability - A Guide to the Theory of NP-Completeness*. 1979.

[2] David Harel and Yishai Feldman. *Algorithmics - Spirit of Computing*. Addison Wesley, 2004.

[3] Richard M. Karp. Reducibility among combinatorial problems. 1972.

[4] Richard Peng. Cs 3510 design and analysis of algorithms - np-complete problems, 2016.

[5] Michael Sipster. *Introduction to the Theory of Computation*. 3rd edition, 2013.

[6] Tlio Ferreira Valeri. Aplicabilidade do problema do caixeiro viajante na roteirizao de visitas de representantes de empresas aos clientes., 2019.