Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

CIC 117536 - Projeto e Análise de Algoritmos

Terceira Prova

Turma: B

NP-completude

Prof. Flávio L. C. de Moura
Claudio Segala R. Silva Filho - 15/0032552
Maria Fernanda do Carmo - 14/0153641

7 de dezembro de 2018

1. **(2.5 pontos)** O problema 2-SAT tem como instâncias as fórmulas lógicas formadas por conjunções de disjunções de até dois literais, onde um literal é uma variável booleana ou a negação de uma variável booleana. Por exemplo, a expressão a seguir é uma instância de 2-SAT:

$$(x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_3) \wedge (x_1 \vee x_2) \wedge x_3$$

Prove que 2-SAT $\in$ P.

Solução.

Sabemos que:

$$(x_i \vee x_j) \equiv (\neg x_i \rightarrow x_j) \wedge (\neg x_j \rightarrow x_i)$$

É possível também notar que um caso do 2-SAT só será insatisfatível se e somente se existe algum literal que não importa o que seja feito sempre gere uma contradição.

Para provar isso vamos usar uma prova por contradição. Suponhamos que temos um caso P' de 2-SAT satisfatível e que possua algum literal l que se assumir valor falso, depois vai ser exigido que possua um valor verdadeiro e vice-versa e que temos a solução para esse caso. Tendo a solução é possível verificá-la em P. Porém, se l for falso nessa solução, alguma cláusula que exige que ele seja verdadeira não será e a cláusula se tornará falsa. Contradição, pois P' era para ser satisfatível.

Dado uma instância P de 2-SAT, transforme cada cláusula para a forma de implicações como mostrado no início dessa questão. Após fazer isso, crie um grafo G que represente essa rede de implicações, onde uma implicação é uma aresta e cada literal e sua versão negada são vértices. Agora é preciso saber se sempre existe algum literal que vai exigir o oposto de seu valor não importando qual valor for escolhido, em outras palavras, se existe isso:

$$(x_i \rightarrow \neg x_i) \wedge (\neg x_i \rightarrow x_i)$$

Isso pode ser feito em tempo polinomial utilizando o algoritmo de Kosajaru que encontra componentes conexas, isto é, vai encontrar se é possível chegar diretamente ou indiretamente de x_i para $\neg x_i$ e o inverso.

2. **(2.5 pontos)** Em aula, assumimos que SAT é um problema NP-completo (Teorema de Cook-Levin), e a partir deste fato mostramos que 3-SAT e CLIQUE também são problemas NP-completos. As reduções foram feitas de acordo com o seguinte diagrama:

$$\begin{tikzcd} SAT \arrow[d] \\ 3\text{-}SAT \arrow[d] \\ CLIQUE \end{tikzcd}$$

Um ciclo Hamiltoniano é um ciclo simples que visita cada vértice de um grafo exatamente uma vez. Considere o problema de decisão HAM-CYCLE que pergunta se um dado grafo (não-dirigido) G possui um ciclo Hamiltoniano. Mostre que HAM-CYCLE é um problema NP-completo. Sua solução deve ser construída a partir de SAT, 3-SAT ou CLIQUE. Caso, você não veja como reduzir diretamente HAM-CYCLE a partir destes, mas sabe como fazê-lo a partir de um certo problema Q então inicialmente mostre que Q é NP-completo a partir de SAT, 3-SAT ou CLIQUE, e assim por diante. Digamos que você não saiba como mostrar que Q é NP-completo diretamente a partir de SAT, 3-SAT ou CLIQUE, mas você sabe como fazê-lo a partir de outro problema Q', e também sabe como mostrar que Q' é NP-completo a partir de 3-SAT, por exemplo. Então o diagrama correspondente à sua solução seria:

$$\begin{tikzcd} SAT \arrow[d] & & & \\ 3\text{-}SAT \arrow[r] \arrow[d] & Q' \arrow[r] & Q \arrow[r] & \text{HAM-CYCLE} \\ CLIQUE & & & \end{tikzcd}$$

E todas as reduções (de 3-SAT para Q', de Q' para Q e de Q para HAM-CYCLE) devem ser detalhadas na sua solução.

Solução.
O caminho escolhido foi: 3-SAT para Hamiltonian Cycle.

Começamos com uma expressão 3-SAT, ela contem n variáveis, logo são 2^n possibilidades. Queremos que cada uma dessas possibilidades seja modelada com um Hamiltonian Cycle. Cada variável tem seu caminho correspondente c_1, $c_2 ... c_n$. Cada um desses caminhos consiste de 2k nós. Onde k é o número de cláusulas na expressão.
Depois vamos atribuir os nós aos caminhos, e ligar cada nó ao nó do lado direito. Depois cada nó será ligado ao nó do lado esquerdo. Em seguida todos os nós das pontas serão conectados também, tanto ao nó diretamente abaixo, como também ao diretamente abaixo da outra extremidade.
Dois novos nós são adicionados, um acima e outro abaixo do caminho, o de cima é ligado às duas extremidades superiores e o de baixo às duas extremidades inferiores. O nó inferior, é também conectado ao superior. E novos nós são adicionados ao grafo, seu número é igual ao número de cláusulas usadas anteriormente. As cláusulas são conectadas às variáveis que as contêm. O sentido sempre é nó d cláusula, que vai ser conectada a primeira variável do caminho e depois à segunda variável do caminho.
Após todas as ligações, qualquer ciclo Hamiltoniano construído no Grafo independe de direções. Como cada caminho pode ser percorrido em 2 direções e são n caminhos, que mapeiam n variáveis. Então existem 2^n ciclos Hamiltonianos correspondentes a um conjunto de variáveis. Logo, esse grafo pode ser construído em tempo polinomial. a

item **(2.5 pontos)** Considere o seguinte jogo em um grafo (não-dirigido) G, que inicialmente contém 0 ou mais bolas de gude em seus vértices: um movimento deste jogo consiste em remover duas bolas de gude de um vértice $v \in G$, e adicionar uma bola a algum vértice adjacente de v. Agora, considere o seguinte problema: Dado um grafo G, e uma função $p(v)$ que retorna o número de bolas de gude no vértice v, existe uma sequência de movimentos que remove todas as bolas de G, exceto uma? Mostre que este problema é NP-completo. A mesma observação feita no exercício anterior vale aqui: a prova deve ser feita a partir de problemas que provamos serem NP-completos, e reduções intermediárias, caso existam, devem ser incluídas na solução.

Solução.

Considere uma versão mais simples desse problema onde todos os vértices tem uma bola e apenas um deles tem duas bolas. Além disso, considere que Hamiltonian Path é NP-Completo.

O objetivo é prova que essa versão mais simples do problema é NP-Completo através do Hamiltonian Path e usar a versão mais simples para mostrar que o problema dado é NP-Completo.

Primeiro vamos mostrar que a versão mais simples é verificável em P. Para isso será mostrado um algoritmo polinomial que o resolve. Dado uma solução (uma sequência de movimentos), basta executar em sequência as operações descritas e depois verificar se no final temos apenas um vértice com uma bola.

Segundo, vamos mostrar que Hamiltonian Path tem solução se e somente se a versão mais simples também tem solução.

Considere um grafo qualquer Q e para cada vértice adicione um a bola e para um vértice qualquer coloque duas bolas. Ao achar um caminho hamiltoniano em Q, podemos começar do vértice que tem duas bolas e executar a operação de transferência de bolas para o próximo vértice do caminho, quando terminar de passar por todos os vértices, vamos ter somente um vértice com duas bolas, para tornar terminar seria preciso somente um movimento de realizar a operação de transferência de bolas para qualquer vértice adjacente.

Considere agora um grafo qualquer Q' com todos os vértices com uma bola e um com duas bolas. Sabemos que:

(a) Só podemos realizar a operação de transferência a partir de um vértice com duas bolas e vértice que recebeu agora tem duas bolas.

(b) Quando realizado a operação, o vértice que cedeu as bolas fica sem bolas

(c) Ao ir para um vértice já visitado (sem bolas), não há mais movimentos para fazer (não há mais vértices com dois).

Ao encontrar um jeito de resolver o problema, temos uma sequencia de ações de um vértice u para um ouro v. Sendo que somente no último houve uma visitação de um vértice já visitado para terminar. Assim, ao desconsiderar a última ação, temos um fluxo das bolas que passa por todos os vértices sem nunca repetir um vértice, um caminho hamiltoniano.

Logo, a versão mais simples do problema é NP-Completo.

Terceiro, vamos mostrar que a versão simplificada só tem solução se e somente se a versão original tem solução.

Considere um grafo G...

3. **(2.5 pontos)** Uma fórmula booleana em *forma normal conjuntiva com disjunção exclusiva (FNCX)* é uma conjunção de diversas cláusulas, e cada cláusula é uma disjunção exclusiva (XOR) de diversos literais. Lembre-se que a disjunção exclusiva é dada por:

a	b	$a \oplus b$
V	V	F
V	F	V
F	V	V
F	F	F

O problema FNCX-SAT pergunta se uma dada fórmula em FNCX é satisfatível. Mostre que o problema FNCX-SAT está em P, ou então que FNCX-SAT é NP-completo. No último caso, a mesma observação feita nos dois exercícios anteriores vale aqui: a prova deve ser feita a partir de problemas que provamos serem NP-completos, e reduções intermediárias, caso existam, devem ser incluídas na solução.

Solução.

Dada uma instância P qualquer de FNCX. Toda cláusula de P estará na forma:

$$x_1 \oplus ... \oplus x_n$$

Com n maior que 0. Como esse tipo de cláusula só pode ser verdadeira se a quantidade de literais verdadeiros for ímpar, podemos por qualquer cláusula na forma de uma expressão aritmética modular em 2. Sendo que o valor verdade vira 1 e o valor falso vira 0. Logo a cláusula só é verdade se é possível resolver a seguinte equação:

$$x_1 + ... + x_n \equiv 1 \mod 2$$

Em caso de um literal negado, podemos inverter o valor que contribuiria para a equação adicionando 1, visto que:

$$1 + 0 \equiv 1 \mod 2$$

e

$$1 + 1 \equiv 0 \mod 2$$

Exemplo:

$$(x_1 \oplus \neg x_2 \oplus \neg x_3) \implies (x_1 + (1 + x_2) + (1 + x_3)) \equiv 1 \mod 2)$$

Como todas as cláusulas tem que ser verdadeiras, podemos montar um sistema composto das transformações das cláusulas em equação.
Como um sistema, podemos classificar "consistente"se possui pelo menos uma solução e "inconsistente"se não possui solução. Logo, se o sistema for consistente, teremos que a expressão será satisfatível, já que existe uma solução que dê $1 \mod 2$ para todas as equações. Se for inconsistente, não terá solução e será insatisfatível.
Existem vários métodos que podem classificar aquela expressão, sendo um deles Gauss Row Reduction.