* PROBLEMAS DE SATISFATIBILIDADE * (Cormen, 2ª ed., §34.3)

1. DEFINIÇÃO: o problema Satisfatibilidade de Circuitos ("SAT-CIRCUITOS") é o de, dado um certo circuito (não circular) com portas lógicas E, OU e NÃO, determinar se existe uma ATRIBUIÇÃO de entrada que faça o circuito produzir o valor de saída 1.

2. EXEMPLOS: uma maneira de codificar entradas para o problema Satisfatibilidade de Circuitos por meio do alfabeto

   0 1 { } , E O N

   é a seguinte:

   1. Vírgulas separam as partes, subpartes, etc, da entrada.

   2. A 1a parte da entrada fica entre chaves e é a lista, em ordem crescente, dos índices dos fios de entrada, começando de 1.

   3. As demais partes da entrada correspondem às portas lógicas. Cada porta lógica é especificada entre chaves. Dentro das chaves, temos:

      * A 1a parte é o número da porta lógica enquanto elemento do circuito (assim como os fios de entrada, as portas lógicas também são numeradas).

      * A 2a parte é uma letra que especifica o tipo da porta lógica: "E" (e), "O" (ou) e "N" (não).

      * A 3a parte é o número do elemento usado como 1a entrada da porta lógica.

      * A 4a parte -- que existe apenas para portas E e OU -- é o número do segundo elemento usado como 2a entrada da porta.

   4. Os elementos do circuito (fios e portas lógicas) devem ser fornecidos em ordem crescente e sucessiva de índice, e portas lógicas só podem receber como entrada elementos de índice estritamente menor.

   5. A saída do circuito é dada pelo elemento de maior número.

   6. Todo elemento (fio ou porta lógica) que não for o elemento de saída do circuito deve ser entrada de alguma porta lógica.

   Seguem abaixo exemplos de entrada no formato discutido acima. Por simplicidade, os números foram apresentados em base decimal, mas é imediato convertê-los para a base binária, conforme exigido pelo alfabeto de 8 símbolos acima. Naturalmente, na versão concreta do problema, todos os 8 símbolos do alfabeto acima têm que ser mapeados em sequências binárias, o que, no caso, se faz facilmente com 3 bits por símbolo. A fim de melhor ilustrar a função booleana computada por cada circuito, segue abaixo de cada um uma fórmula lógica equivalente, a qual, entretanto, não está envolvida na definição formal do problema, nem como entrada nem como saída. Nessas fórmulas ilustrativas, "Fx" -- "x" sendo um índice -- denota a variável booleana correspondente ao x-ésimo fio de entrada.

   a) "{1}"
      |
      '--> * 1: entrada e saída.
           |
           '-> equivalente à fórmula "F1".

b) "{1,2,3},{4,E,1,2},{5,N,3},{6,O,4,5}"
   |
   '--> * 1,2,3: entradas.
        * 4: 1 e 2.
        * 5: não 3.
        * 6: 4 ou 5 (saída).
        |
        '-> equivalente à fórmula "(F1 e F2) ou (não F3)".

c) "{1,2,3,4},{5,E,2,3},{6,O,1,5},{7,O,5,4},{8,E,6,7}"
   |
   '--> * 1,2,3,4: entradas.
        * 5: 2 e 3.
        * 6: 1 ou 5.
        * 7: 5 ou 4.
        * 8: 6 e 7 (saída).
        |
        '-> equivalente à fórmula "(F1 ou (F2 e F3)) e ((F2 e F3) ou F4)".

d) "{1},{2,N,1},{3,E,1,2}"
   |
   '--> * 1: entrada.
        * 2: não 1.
        * 3: 1 e 2 (saída).
        |
        '-> equivalente à fórmula "F1 e (não F1)".

3. EXERCÍCIO: Escreva, como exemplificado acima, o circuito representado pela
              entrada

   "{1,2,3,4},{5,E,1,2},{6,E,5,3},{7,E,5,4},{8,E,6,7}" ,

   assim como a fórmula lógica correspondente.

4. EXERCÍCIO: para cada natural positivo par "n", seja C_n o circuito de "n"
              entradas definido indutivamente a seguir:

   - C_2: * 1,2: entradas.
          * 3: 1 e 2 (saída).

   - C_{n+2}: como C_n, mas com as seguintes adições/modificações:

          * n+1,n+2: entradas adicionais. Logo, os números das portas lógicas de
                     C_n devem ser aumentados em 2 unidades. Seja "m" o número
                     da última porta lógica de C_n após essa alteração.
          * m+1: m e n+1.
          * m+2: m e n+2.
          * m+3: m+1 e m+2 (saída).

   Assim sendo:

   a) Verifique que C_4 é exatamente o circuito do exercício 3 acima.

   b) Escreva a fórmula lógica correspondente a C_4, à maneira exemplificada
      acima.

   c) Escreva o circuito C_6.

   d) Prove que o número de portas lógicas de C_n é polinomial sobre "n".

   e) Prove que a fórmula lógica obtida a partir de C_n à maneira exemplificada
      anteriormente não tem tamanho polinomial sobre "n" (você pode, por
      exemplo, estimar o número de conectivos da fórmula, assim como fizemos em
      sala).

5. LEMA: SAT-CIRCUITOS ∈ NP.

==========
PROVA:

O seguinte algoritmo verifica a linguagem do problema, utilizando como certificado uma atribuição de entrada que faça o circuito produzir a saída "1". O algoritmo supõe que a entrada está representada como estipulado no EXEMPLO 2 acima, devidamente transformada em sequência binária.

==========
Algoritmo: VSATC
Entrada: x,y ∈ {0,1}*
Saída: 0 ou 1.
----
1. Compute o circuito representado por "x",
   ou retorne "0" se "x" não representa um circuito.
2. Sejam "n" o número de entradas e "m" o número total de elementos do circuito.
3. Crie um vetor V[1..m] de bits.
4. Para computar a atribuição de entrada representada por "y",
   faça V[i] := y[i] para todo "i" de 1 a "n",
   ou retorne "0" se |y| != n.
5. Para cada "i" de n+1 a "m", armazene em V[i] o valor produzido pela porta
   lógica de índice "i", computando esse valor com base nos valores já
   armazenados em V[1..i-1].
6. Retorne V[m].
==========

Para verificar que VSATC verifica a linguagem SAT-CIRCUITOS em tempo polinomial, observe que:

  a) Para toda instância "x" "sim" do problema, existe y ∈ {0,1}* tal que VSATC(x,y) = 1 e |y| <= |x|, e, para toda instância "x" "não" do problema, VSATC(x,y) = 0 para todo "y". Em particular, VSATC somente retorna "1" se "y" é uma atribuição de entrada válida para o circuito representado por "x" e se tal atribuição faz o circuito produzir a saída "1". Em todos os outros casos, VSATC retorna "0".

  b) VSATC sempre executa em tempo polinomial sobre |x|+|y|, pois:

     * Por suposição sobre a codificação do problema (OBSERVAÇÃO 5 da nota de aula sobre a classe NP), a linha 1 executa em tempo polinomial sobre |x|.

     * As linhas 2 a 6 executam em tempo O(m), e m <= |x|. Em particular, observe que a linha 5 pode ser executada em tempo O(m), já que a entrada fornece as portas lógicas ordenadas de forma que a computação dos valores por elas produzidos pode ser feita em um único percurso pelo circuito.

Como VSATC verifica SAT-CIRCUITOS em tempo polinomial, então SAT-CIRCUITOS ∈ NP, CQD.
==========

6. FATO: SAT-CIRCUITOS é NP-difícil. (Cormen, 2ª ed, lema 34.6.)

7. COROLÁRIO: SAT-CIRCUITOS ∈ NPC.