NP-complete: Conjunctive Boolean Query SR-31

A conceituação de Np-complete foi criada no início de 1970 simultaneamente por pesquisadores Norte Americanos e da extinta URSS.

Em 1971 Stephen Arthur Cook publicou nos Estados Unidos seu artigo "The Complexity of Theorem Proving Procedures", Cook explica a redução de tempo polinomial (conhecida como teorema de cook) e o Np-complete.

Independentemente Leonid Levin publicou o teorema na URSS, com isso o teorema ficou conhecido como Teorema de Cook-Levin.

Para a teoria da complexidade computacional a Consulta Booleana Conjuntiva (do inglês Conjunctive Boolean Query) foi a primeira a pertencer a classe.

A Conjunctive Boolean Query é a utilização da lógica de Boolean, criada por George Boole, matemático e lógico Britânico.

Boole criou a lógica binária, teoria na qual existem apenas dois valores 0,1 verdadeiro ou falso.

A Conjunctive Boolean Query aplica a lógica Boole em sistemas de recuperação de dados e mecanismos de busca, fazendo a combinação de dois ou mais termos, os correlacionado por executores lógicos com isso conseguindo buscas mais detalhadas e específicas.

A fórmula está na forma conjuntiva (CNF) se for uma conjunção de cláusulas, que utiliza apenas os operadores lógicos de conjunção ($\wedge$), disjunção ($\vee$) e negação ($\neg$).

Quais as aplicações práticas do Conjunctive Boolean Query ?

As aplicações da Conjunctive Boolean Query (comumente abreviada como SATISFIABILITY ou apenas SAT) em geral estão presentes em mecanismos de busca, recuperação de dados e também em bancos de dados que utilizam a linguagem SQL.

O seu uso permite que a busca seja mais precisa, permitindo o foco em palavras contidas apenas no corpo de textos ou apenas em título.

A busca pode ser composta ou simples com possibilidade de troca de palavras e substituição de caracteres no meio das palavras, possibilitando grande melhoria nas pesquisas dos usuários.

A Consulta Booleana Conjuntiva permite efetuar operações lógico-matemático usando operadores: AND, OR e NOT, com ele é possível combinar palavras-chave na busca em bases de dados eletrônicos.

Freqüentemente, os conceitos booleanos são explicados por meio dos diagramas de Venn. No presente caso, os diagramas de Venn podem representar o espaço de informação da Rede. Um círculo contendo uma palavra mostra o subconjunto de documentos da Rede que contêm esta palavra. Quando há dois conceitos em um único diagrama, a interseção entre os círculos representa aqueles documentos que contêm ambos os conceitos. (WEB ARCHITECTS, 2004).

Todos os problemas da classe (NP) são amplamente estudados, a habilidade de resolver(NP) está relacionada a resolução de problemas em (P), o problema P vs NP é o maior desafio da ciência e engenharia da computação ainda em aberto.

Exemplos de problemas NP

Para entendermos melhor a complexidade dos problemas SAT vou listar alguns outros problemas computacionais.

Problemas de otimização.

Problemas de otimização. (= optimization), ou seja, problemas que pedem o mínimo ou o máximo de alguma coisa.

- Máximo divisor comum: Encontrar o maior divisor comum de dois números inteiros positivos m e n.
- Subsequência crescente máxima: Encontrar uma subsequência crescente máxima de uma sequência s1, … , sn de números naturais.
- Mochila ótima: Dados números naturais p1, … , pn, v1, … , vn e c, encontrar um subconjunto K de {1, … , n} tal que a soma $\sum$ (pk : k $\in$ K) não passe de c e a soma $\sum$ (vk : k $\in$ K) seja máxima.

Problemas de busca.

Cada problema pede um objeto que tenha certas propriedades.

- Quadrado perfeito: Dado um número natural n, encontrar um número natural x tal que $x^2 = n$ (ou constatar que tal x não existe).
- Equação inteira do segundo grau: Dados números inteiros a, b e c, encontrar um número inteiro x tal que $ax^2 + bx + c = 0$ (ou constatar que tal x não existe).
- Fatoração: Dado um número natural n, encontrar um número natural p maior que 1 e menor que n que divida n (ou constatar que tal p não existe).

Problemas de decisão.

problemas cujas instâncias admitem apenas dois tipos de solução: sim e não. Diz-se que esses problemas são de decisão (= decision problem).

- Quadrado-Perfeito: Dado um número natural n, decidir se o número $\sqrt{n}$ é natural, ou seja, decidir se existe um número natural x tal que $x^2 = n$
- Eq-Grau-2: Dados números inteiros a, b e c, decidir se existe um número inteiro x tal que $ax^2 + bx + c = 0$
- Eq-Diofantina: Dada uma equação polinomial com coeficientes inteiros e número arbitrário de variáveis (como x 3yz + 2y 4z 2 − 7xy 4z = 6, por exemplo) decidir se existem valores inteiros das variáveis que satisfazem a equação.

Problemas de estratégias.

Se provarmos que um dado problema de decisão não tem um algoritmo razoavelmente rápido, ficará demonstrado também que os correspondentes problemas de busca e de otimização não têm algoritmos razoavelmente rápidos.

(Fonte:USP-SP)

Resolvendo os Problemas SAT

Ainda não possuímos algoritmo eficiente para resolver o problema da satisfatibilidade booleana conjuntiva, mas possuímos algoritmos para encontrar soluções aproximadas em problemas NP-complete

- Aproximação: algoritmo que encontra rapidamente uma solução não perfeita mas dentro de uma média de erro.
- Probabilístico: algoritmo que consegue em média boas soluções.
- Restrição: restringe a estrutura de entrada, existem algoritmos mais rápido
- Parametrização: geralmente são algoritmos rápidos se as entradas certas forem feitas
- Heurísticas. Algoritmo trabalha relativamente bem, mas não possui provas da sua velocidade e da sua eficácia.

Viu? Bem simples. Repare que não há solução definitiva, mas os avanços continuam.