Universidade Federal da Paraíba

Centro de Informática

Análise e Projeto de Algoritmo

Professor: Bruno Bruck

Aluno: Ednaldo Martins da Silva

Projeto Final da Disciplina: Problema de Otimização Combinatória

Problema de Otimização Combinatória: 1. Caixeiro Viajante

1 Definição do POC

O problema do caixeiro viajante é um problema de otimização combinatória. Para solucionar este problema devemos encontrar a menor rota possível passando por várias cidades, uma única vez em cada, não importando a ordem em que são visitadas, percorrendo o menor percurso possível, e retornar ao ponto de origem.

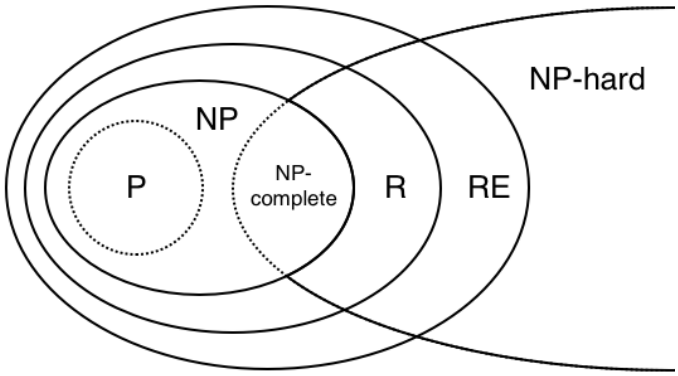
Esse problema é similar aos problemas de percurso do mundo real, onde dado uma quantidade ***n*** de cidades, queremos saber qual o menor caminho possível podemos seguir visitando todas as cidades exatamente uma única vez.

2 O PCV pertence a Classe NP e NP-Difícil

2.1 A Classe NP e NP-Difícil

NP ou **Non-Desterministic Polynomial time** é a classe das linguagens que têm verificadores de tempo polinomial. Enquanto as linguagens da Classe P podem ser decididas rapidamente, as linguagens da Classe NP podem ser verificadas rapidamente.

Para solucionar alguns problemas podemos evitar a força bruta para obter uma solução de tempo polinomial. Todavia isso nem sempre é possível, e isto pode ocorrer porque não exista uma solução de Classe P para o problema, ou porquê ainda não descobrimos uma maneira possível para executar uma solução de Classe P para o problema.



Fonte: https://cs.stackexchange.com/questions/90659.

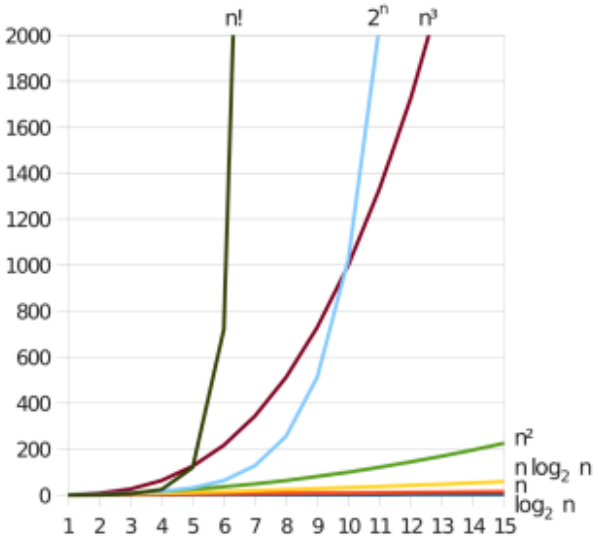
Também existem problemas da Classe NP-Completo (ou NP-Completude), que são problemas mais difíceis da Classe NP, e que provavelmente não fazem parte da Classe P, e caso encontre uma prova para tal, que mostre que faça parte, então estaríamos afirmando que qualquer problema NP-Completo que estão interligados poderiam ser solucionados rapidamente ou em tempo polinomial. Para entendermos esse fato podemos fazer analogia a um problema qualquer que possui um grau de dificuldade X e outro de X^2, portanto se solucionarmos o problema de dificuldade X^2 então encontraríamos a solução para o problema de dificuldade X, pois estão interligados.

2.2 Provando de que PCV pertence a Classe NP-Difícil

Para encontrar o menor caminho possível é preciso tomar decisões. Vamos tomar um caso em que será analisado, um grafo completo, onde todas as cidades estão conectadas entre si, tendo ***n*** como número de cidades nesse grafo. Notemos que na primeira cidade tem-se que decidir entre (***n***-1) cidades, pois a cidade de origem não é uma opção aqui (apenas no fim do percurso deve-se escolher, ou voltar nesse caso, para a cidade inicial), e partindo para a segunda cidade tem-se (***n***-2) opções disponíveis, e analogamente, (***n***-3) opções, (***n***-4), e assim por diante até percorrer todas as cidades e voltar a cidade de origem. Logo:

- ao iniciar o percurso o caixeiro tem ao todo (n-1)! Soluções possíveis;
- [(n-1)!]/2 decisões para tomar ao longo do caminho.

Para este caso temos um problema de complexidade fatorial, e portanto, pertencente a Classe NP-Difícil. Temos aqui um problema de complexidade de Algoritmo que torna a solução impraticável quando precisamos de um N grande, principalmente levando em conta que o tempo para solução desse problema se torna muito pior cada vez que esse N aumenta um pouco.



Fonte: Bruno Bruck, Análise e Projeto de Algoritmo, Aula 1 – Introdução.

Se levarmos em consideração situações reais temos muitos problemas ao calcular a melhor rota quando trabalhamos com muitas cidades. Suponhamos que temos 8 cidades, então temos R=(8-1)! => R= 5040 rotas possíveis, todavia se tivermos uma situação com 21 cidades, então teríamos R=(21-1)! => R=2432902008176640000 de rotas possíveis. Isso é inviável. Isso acontece porque esse é um método de força bruta. Até o momento ainda não foi possível encontrar um método ou técnica que realize esse trabalho com complexidade em tempo polinomial.

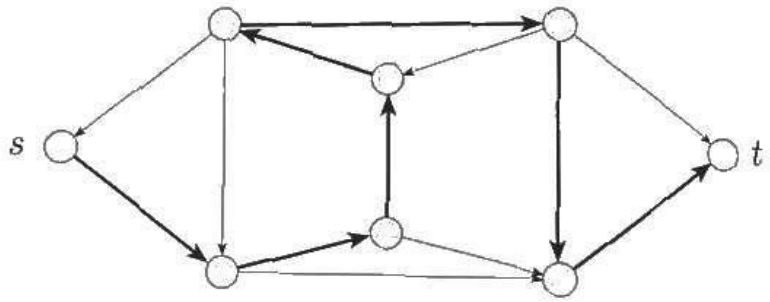
3 Problema reduzido para o POC

3.1 Redutibilidade em Tempo Polinomial

Para solucionar esses problemas é utilizado um método chamado de **Redutibilidade em Tempo Polinomial**. De acordo com Sipser (2007, p. 288) quando o problema A se reduz ao problema B, uma solução para B pode ser usada para resolver A. Agora definimos uma versão da redutibilidade que leva em consideração a eficiência da computação. Quando o problema A é eficientemente redutível ao problema B, uma solução eficiente para B pode ser utilizada para resolver A eficientemente.

3.2 Caminho Hamiltoniano

Um **caminho hamiltoniano** em um grafo direcionado G é um caminho direcionado que passa por cada nó exatamente uma vez. Podemos dizer que se um grafo direcionado contém um caminho hamiltoniano, ou seja, um possível caminho que passe por todos os nós ou vértices até chegar ao destino, então este grafo possui um caminho hamiltoniano.



Fonte: Sipser, página 280.

Vemos na imagem acima um grafo direcionado onde o caminho hamiltoniano vai de ‘s’ para ‘t’.

Podemos facilmente encontrar similaridades no problema do caminho Hamiltoniano com o PCV, e podemos reduzir o PCV para este, apenas fazendo uma simples mudança. Basta pegar o grafo G do PCV, e tornar o custo do percurso de cada aresta existente no grafo, igual a 1. Partindo disso teremos o mesmo grafo, desprezando apenas o custo do caminho percorrido, e que através da solução do Caminho Hamiltoniano possamos saber se existe pelo menos um caminho hamiltoniano existente no grafo, e caso exista, esse caminho tem custo total igual ao custo da rota original existente também no grafo G do PCV. O problema do caminho hamiltoniano é NP-Completo, portanto, é improvável encontrar uma solução que resolva esse problema em tempo polinomial.

4 Instâncias na literatura

4.1 Exemplo prático

Vamos tomar como exemplo um grafo com 4 cidades, daí temos o grafo G={1,2,3,4}, onde o conjunto de valores contido no grafo representam os vértices, ou seja, as cidades para o nosso exemplo. Digamos que esse é um grafo completo, onde todas as cidades estão conectadas entre si.

Instância para leitura em arquivo:

4

0 1 3 4

1 0 2 7

3 2 0 2

4 7 2 0

Vejamos então os possíveis caminhos partindo da cidade 1:

- R = (n-1)! = 6 Rotas possíveis no início do percurso:

  12341, 12431, 13421, 13241, 14321, 14231.

- D = [(n-1)!]/2 = 3 decisões a serem tomadas durante o percurso:

Número de Decisões:	(n-1) = 3	(n-2) = 2	(n-3) = 1	0	Voltar!
Cidades visitada:	1	12	123	1234	12341
Próximas cidades:	234	34	4	1	encerrado
Custo Total:	0	1	3	5	9

Note que encontramos 6 possíveis caminhos. Partindo da cidade 1 temos que tomar a decisão entre 3 cidades possíveis a seguir. Ao chegar na segunda cidade temos que decidir entre 2 cidades seguintes, pois uma delas já se foi e não devemos voltar nela e também não devemos voltar na origem enquanto não percorremos todas as cidades restantes. Após tomar a segunda decisão restará apenas uma cidade antes de voltar para origem. Veja, decidimos entre 3 cidades, depois 2 (duas), e então por 1 (uma).

4.2 Instâncias

Instâncias aleatórias:

https://eden.dei.uc.pt/~paquete/tsp/#Exp3

Instâncias misturadas ou mistas de outros tipos de instâncias:

https://eden.dei.uc.pt/~paquete/tsp/#Exp4

REFERÊNCIAS

SIPSER, Michael. Introdução a Teoria da Computação. **Cengage Learning**, p. 271-311, 2007.

http://www.decom.ufop.br/marcone/Disciplinas/InteligenciaComputacional/InteligenciaComputacional.pdf

https://www.ime.usp.br/~pf/analise_de_algoritmos/aulas/NPcompleto.html

http://people.brunel.ac.uk/~mastjjb/jeb/info.html

http://www.mat.ufrgs.br/~portosil/caixeiro.html

https://cs.stackexchange.com/questions/90659

https://homepages.dcc.ufmg.br/~loureiro/alg/071/paa_ProbNP_1pp.pdf

https://pt.wikipedia.org/wiki/Caminho_hamiltoniano

https://pt.wikipedia.org/wiki/NP_(complexidade)