Metaheurística GRASP

(*Greedy Randomized Adaptive Search Procedures*)

Airton Bordin Junior

[airtonbjunior@gmail.com]

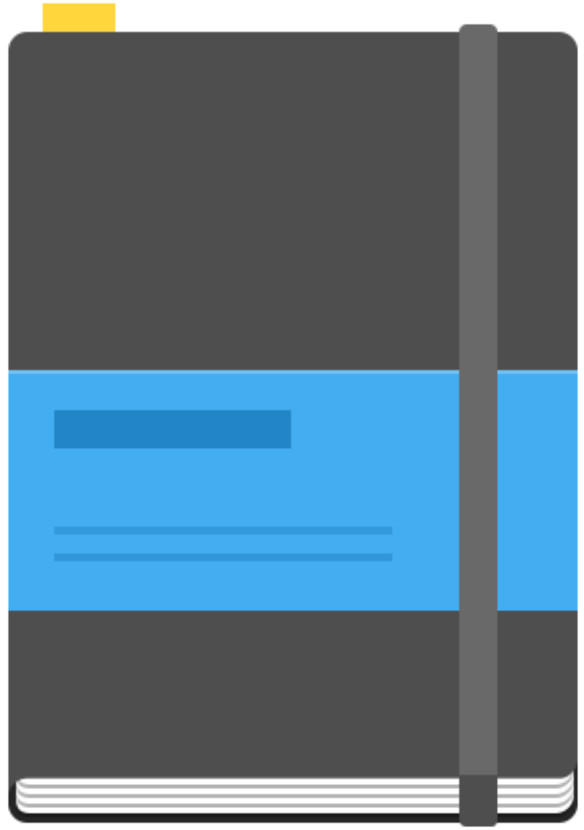
Metaheurísticas - Prof. Dr. Celso Gonçalves Camilo Junior

Mestrado em Ciência da Computação 2017/2

Universidade Federal de Goiás (UFG) - Instituto de Informática – Setembro/2017

Programação

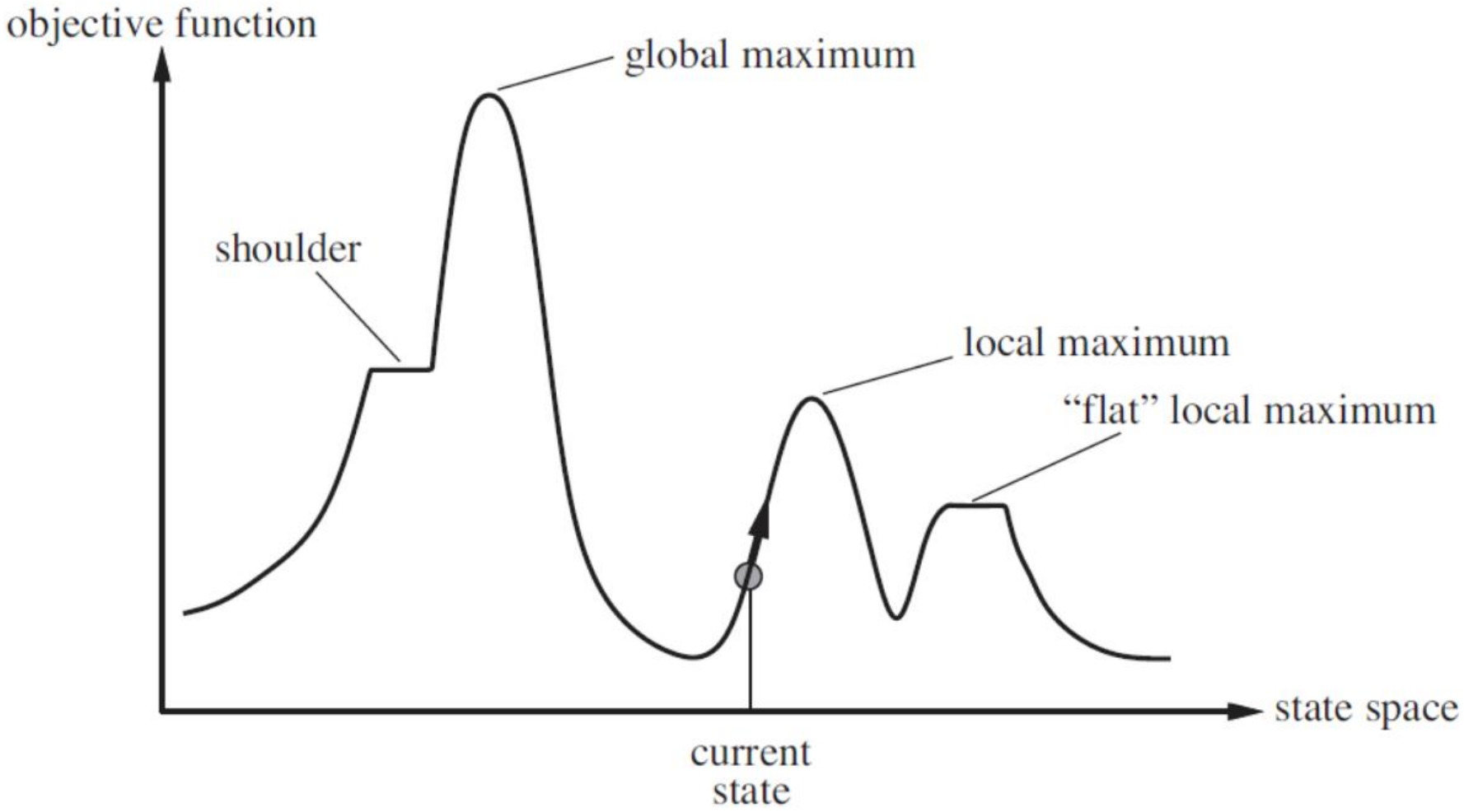
- Introdução
- Busca Local
- GRASP
- Referências



Busca Local

- Algoritmos de busca local operam sobre um único estado corrente, ao invés de vários caminhos;
- Em geral se movem apenas para os vizinhos desse estado;
- Geralmente, cada solução candidata tem mais de uma solução vizinha
  - A escolha de qual será a próxima deve ser feita tomando em consideração apenas a vizinhança da solução atual.
- O caminho seguido pelo algoritmo não é guardado*.

Busca Local


objective function
global maximum
shoulder
local maximum
"flat" local maximum
state space
current
state

GRASP

- Feo e Resende, 1989
  - *A probabilistic heuristic for a computationally difficult set covering problem*
  - Problemas difíceis de cobertura de conjuntos

A probabilistic heuristic for a computationally difficult set covering problem

Authors: Thomas A Feo — The University of Texas, Austin, TX 78712, USA
Mauricio G. C Resende — University of California, Berkeley, CA 94720, USA

1989 Article

Published in:
· Journal
Operations Research Letters archive
Volume 8 Issue 2, April, 1989
Pages 67-71
Elsevier Science Publishers B. V. Amsterdam, The Netherlands, The Netherlands
table of contents doi>10.1016/0167-6377(89)90002-3

Bibliometrics
· Citation Count: 106
· Downloads (cumulative): n/a
· Downloads (12 Months): n/a
· Downloads (6 Weeks): n/a

GRASP

- Processo iterativo em que cada iteração consiste em duas fases: construção e busca local;
- A melhor solução, de maneira geral, é mantida como resultado;
- Apresenta um componente probabilístico que faz a escolha aleatória de um entre os melhores candidatos na fase de construção
  - Permite que soluções diferentes sejam obtidas a cada iteração, mas não compromete o potencial adaptativo.

Fase de construção

- Solução é construída elemento a elemento;
- Inicialmente o elemento está em uma lista de candidatos – **LC;**
- Usando um fator **α** é criada uma lista restrita de candidatos – **LCR** - que possui os melhores elementos de **LC.**

$$CardinalidadeLCR = \alpha * CardinalidadeLC$$

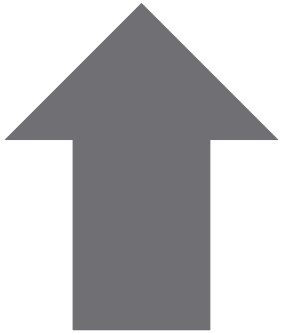
Fase de construção

- Definida a **LCR**, seleciona-se um elemento da mesma para compor a solução
  - Aleatoriamente ou por um critério guloso;
  - Duas variações do GRASP.
- Após a adição do elemento, o processo continua com a atualização de **LC** e **LCR;**
- Processo de construção é finalizado quando a ***CardinalidadeLC*** for zero.

Fase de construção

- Valor de **α** influencia na qualidade e diversidade da solução;

α alto



Soluções gulosas de boa qualidade;
Pouca diversidade

α baixo

Grande diversidade
Qualidade inferior

Fase de construção

- Valor de **α** também influencia o processo de busca local
  - Soluções de qualidade inferior tornam o processo de busca local mais lento;
    - Com **α=1**, o melhor elemento da LCR seria adicionado, assim a fase de construção é uma heurística gulosa.
    - **α>1** a construção é randômica.
- **α** pode ser constante ou não.

Fase de construção

```
s ← ϕ  %  s que representa uma solução parcial nesse caso
α ← determinaTamanhoListaCandidatos
enquanto SoluçãoNãoCompleta faça
        RCLα ← gerarListaCandidatosRestritos(s)
        x ← selecionarElementoAleatório(RCLα)
        s ← s ∪ {x}
        atualizaFunçãoGulosa(s) % atualiza valores da heurística
fim enquanto
```

- Solução inicial é um conjunto vazio;
- **α** determina o tamanho da lista de candidatos;
- Novos elementos são agregados a solução inicial;
- Função gulosa avalia os elementos pelo benefício imediato;
- Melhores elementos formam uma lista de candidatos restritos em quantidade fixa ou de acordo com algum parâmetro **α** (aleatório ou guloso);
- Elemento da lista é escolhido aleatoriamente;
- Função gulosa é adaptada de acordo com a solução parcial;
- Termina com uma solução aceitável ao problema.

Fase de busca local

- Refinar a solução encontrada na fase de construção aplicando um método de busca local;
- Intensificação na solução encontrada explorando regiões vizinhas para encontrar um ótimo local;
- Quanto melhor for a solução gerada na fase de construção, maior será a velocidade para encontrar um ótimo local pela fase de busca local.

Fase de busca local

- Podem ser utilizados nesta segunda fase algoritmos básicos como o *Hill-Climbing* ou metaheurísticas mais avançadas como Busca Tabu, *Simulated Annealing*, etc;
- GRASP não faz uso de históricos no processo de busca
  - Possível armazenar as melhores soluções até o momento;
  - GRASP é simples, rápido e facilmente integrável com outras técnicas de busca.

Pseudocódigo GRASP

```
procedure ConstructGreedyRandomizedSolution(Solution)
1     Solution = {};
2     for Solution construction not done →
3           MakeRCL(RCL);
4           s = SelectElementAtRandom(RCL);
5           Solution = Solution ∪ {s};
6           AdaptGreedyFunction(s);
7     rof;
end ConstructGreedyRandomizedSolution;

procedure grasp()
1     InputInstance();
2     for GRASP stopping criterion not satisfi ed →
3           ConstructGreedyRandomizedSolution(Solution);
4           LocalSearch(Solution);
5           UpdateSolution(Solution,BestSolutionFound);
6     rof;
7     return(BestSolutionFound)
end grasp;
```

Vantagens GRASP

- Facilmente implementável, com um ajuste ou outro de parâmetros do método;
- Implementação paralela é trivial
  - Cada CPU pode ser iniciado com sua própria cópia do procedimento e dados da instância;
  - Iterações são executadas em paralelo e apenas se utiliza uma variável global para armazenar a melhor solução encontrada entre as CPUs.

Aplicações

- **Feo e Bard, 1989** - Localização de estações de manutenção e escalonamentos de voos de acordo com a demanda cíclica por manutenção;
- **Xu e Chiu , 1996** – Escalonamento de serviços em diferentes locais e com janelas de tempo para técnicos com diferentes habilidades de trabalho,.
- **Lourenço, Paixão e Portugal, 1998** - Escalonamento de tripulação;
- **Rivera, 1998** - Escalonamento de cursos;
- **Binato, Hery, Loewenstern e Resende, 2001** - Escalonamento do Job-Shop.

Referências

- RUSSELL, S., NORVIG, P. **Artificial Intelligence: A modern approach**. Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs, v. 25, p. 27, 1995
- FEO, T. A., RESENDE M. G. C. **Greedy Randomized Adaptive Search Procedures.** Journal of Global Optimization
- FREDO, A. R., BRITO, R. C. **Implementação da Metaheurística GRASP para o Problema do Caixeiro Viajante Simétrico.** Universidade Federal do Paraná – Tópicos em Inteligência Artificial – Profª Aurora Pozo
- RODRIGUES, M. C., LUZIA, L. F. **Introdução ao Escalonamento e Aplicações – Estudo sobre as Metaheurísticas.** Universidade de São Paulo
- FERNANDES, C. R., CARNIERI, C., BARBOSA, S. G. **Aplicação da metaheurística GRASP na programação de caminhões para o transporte de aves domésticas.** Semina: Ciências Exatas e Tecnológicas, Londrina