Computação Natural - TP2
Ant Colony Optimization

Guilherme Torres
Departamento de Ciência da Computação - UFMG

1 Introdução

O objetivo deste trabalho foi formular um algoritmo para resolver heuristicamente o problema das p-medianas em um gráfico de entrada com posições dos vértices especificadas. Dados n vértices, o problema insiste em escolher p medianas entre eles e ligá-las a outros vértices como clientes, minimizando o somatório das distâncias totais da mediana ao cliente. Além disso, cada cliente tem uma demanda e cada mediana tem uma capacidade. Em uma solução viável, a demanda de todos os clientes de uma mediana não pode superar a sua capacidade.

A heurística usada foi a otimização por colônia de formigas. Esta técnica foi introduzida nos anos 90 e é baseada na forma de comunicação que as formigas estabelecem entre si, dentro de um formigueiro. Cada um dos membros da colônia procura no objetivo, e, ao achar um caminho mais favorável, a quantidade de feromônio neste caminho se torna mais densa por consequência, fazendo com que as formigas tenham uma convergência para rotas mais curtas dentro do formigueiro. Esse paradigma pode ser modelado de várias formas dependendo do problema, como vamos ver a seguir.

O algoritmo foi implementado em C++11, na pasta src/. Instruções para compilar e executar o teste estão explicitadas no arquivo README.md. O programa também pode ser encontrado no repositório:
`https://github.com/torresguilherme/cn-tp2-aco`.

2 Implementação

O problema das p-medianas é um problema de minimização. Sabendo disso, a escolha foi guiar as formigas para percorrer a menor distância possível para encontrar os clientes, partindo de um ponto imaginário inicial (a distância

desse "ponto" até as medianas é desprezada), escolhendo medianas e, em seguida, os seus clientes, sempre guiadas pelo feromônio. Para ocupar todos os espaços no grafo, as formigas foram fixadas em $n - p$.

Soluções ilegais (com demanda excedente perante a capacidade de algumas medianas) são permitidas no algoritmo, porém sofrem uma penalidade em sua fitness. Essa penalidade raramente as deixam em condições de serem escolhidas dentre as melhores, porém não permitem que sua fitness seja extremamente ruim (isso pode gerar um excesso de feedback negativo em caminhos potencialmente bons no grafo!)

2.1 O algoritmo

O funcionamento de uma otimização de colônia de formigas, modelada especificamente para esse problema, pode ser descrito, no plano geral, pelos seguintes passos:

1. Inicializa as $n - p$ formigas e uma matriz de feromônios n x n, com o valor 0.5 em cada posição, mais um vetor de feromônios de tamanho n para escolher as medianas;
2. Distribui as formigas entre exatamente p medianas, aleatoriamente;
3. Para cada formiga, escolha um cliente para a mediana que ela está, em função da matriz de feromônios;
4. Encontre a fitness bruta da solução, calculando o somatório da distância que todas as formigas andaram;
5. Caso haja um excedente à capacidade de uma mediana, aplique uma penalidade (capacidade excedida $*P$, sendo P um modificador de penalidade) à fitness;
6. Deixe no vetor e matriz de feromônios feedback positivo ou negativo, dependendo se a solução encontrada foi melhor do que a melhor obtida até o momento ou não;
7. Imprima a média das fitness e a melhor encontrada até o momento;
8. Repita desde o passo 2, até que o número N de iterações seja atingido;
9. Termine a execução com a melhor fitness encontrada.

2.2 Parâmetros do algoritmo

Fitness e penalidade

A fitness de uma solução é dada por

$$\sum_{A_i \in A} d(A_i) + (D_e * P)$$

sendo A o conjunto de formigas, d a distância andada por uma formiga de sua mediana ao cliente, D_e a demanda excedida e P o modificador de penalidade. Para as instâncias testadas, que possuem distâncias médias entre nós não muito maiores que 100, esse foi o multiplicador escolhido e usado nos casos de teste, pois é o bastante para deixar a solução inelegível, porém bom o bastante para sua fitness não explodir negativamente, porém, este valor será variado nos experimentos da seção 3 para mais e menos. Observa-se que, dependendo das quantidades que se trata no problema, talvez seja melhor escolher outras multiplicadores.

Lembrando que estamos lidando com um problema de minimização, fitness menores são fitness melhores.

Atualização de feromônios

A função que atualiza feromônios foi baseada no trabalho de França et al. [1], porém com algumas alterações, já que no caso cada iteração produz apenas uma solução local. O feromônio também produz feedback negativo, caso a solução não seja boa o bastante. A fórmula

$$\tau_{ij} = \tau_{ij} * (Flocal/Fref)$$

mostra como se dá essa atualização, para uma fitness $Flocal$, sendo τ_{ij} o feromônio na matriz entre a mediana i e o cliente j. $Fref$ é uma fitness já obtida que pode servir de referência para gerar feedback positivo ou negativo. No caso, consideramos a melhor fitness já encontrada ou a média das fitness, e vemos que a melhor leva a resultados melhores na próxima seção.

Como as soluções piores já causam o feedback negativo automaticamente, não é necessário haver evaporação no feromônio. Os feromônios dos caminhos não levados em conta na solução local permanecem invariantes.

Probabilidade de escolha do cliente

Assim como demonstrado na citação anterior, a probabilidade de escolha de uma mediana para determinado vértice seria de

$$\frac{\tau_{ij}}{\sum_{Vk \in V-M} \tau_{iVk}}$$

sendo agora V o conjunto de vértices e M as medianas do grafo. Adicionando uma heurística proporcinal ao inverso da distância, a nova fórmula que apresentou melhor desempenho nos testes foi

$$\frac{\tau_{ij} * \frac{1}{d_{ij}}}{\sum_{Vk \in V-M} (\tau_{iVk} * \frac{1}{d_{iVk}})}$$

sendo d_{ij} a distância entre os vértices i e j, o que ajuda o algoritmo a convergir para opções boas desde o início.

Número de iterações

Observou-se que, a partir de um certo ponto, as soluções param de convergir dependendo do tamanho do problema de que se trata. O número de iterações igual a $|A| * 5$, sendo A o conjunto de formigas, mostrou-se razoável para chegar a esse ponto. Isso é demonstrado na convergência das médias aritméticas das soluções melhores e médias encontradas a cada experimento, como será mostrado na próxima seção.

3 Experimentos

Os experimentos abaixo foram realizados em um desktop com sistema Debian 9 unstable e processador Intel Pentium G4400 3.30GHz x 2. Cada gráfico foi montado a partir da média de 30 experimentos. A variância dos valores entre os experimentos foi baixa, portanto, são resultados estáveis. Também serão dadas as melhores soluções para cada uma das instâncias, no melhor desempenho do algoritmo.

3.1 Cálculo do feromônio: em função da melhor fitness vs. da média

Os gráficos das figuras 1 a 3 mostram os resultados das melhores soluções encontradas (em vermelho) e das médias (em azul) por iteração usando a

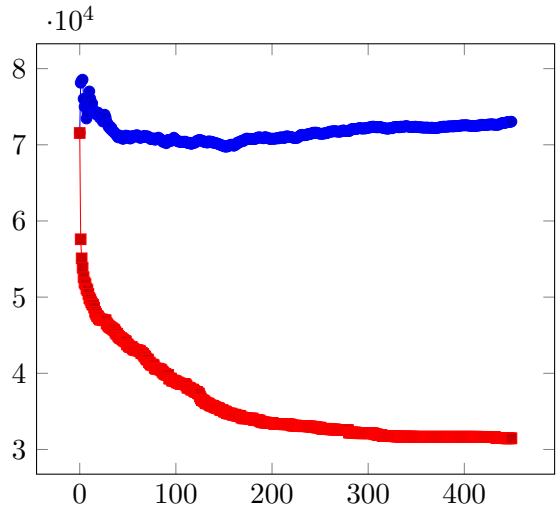

Figure 1: Atualização de feromônio com melhor fitness, SJC1.dat

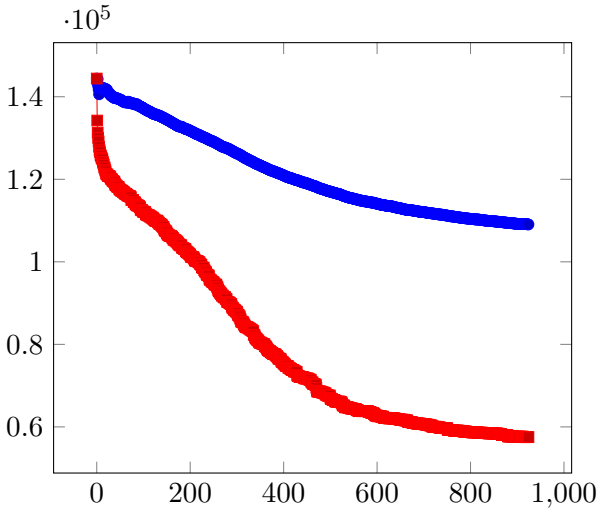

Figure 2: Atualização de feromônio com melhor fitness, SJC2.dat

melhor fitness como $Fref$ na fórmula de atualização do feromônio, enquanto 4 a 6 demonstram o desempenho da fitness média como este parâmentro.

Observa-se que, nos dois casos, há certa convergência. Porém, é notável que, em todos os casos de teste, a busca fica estagnada muito mais cedo quando se uma a média das fitness em vez da melhor como $Fref$ na fórmula

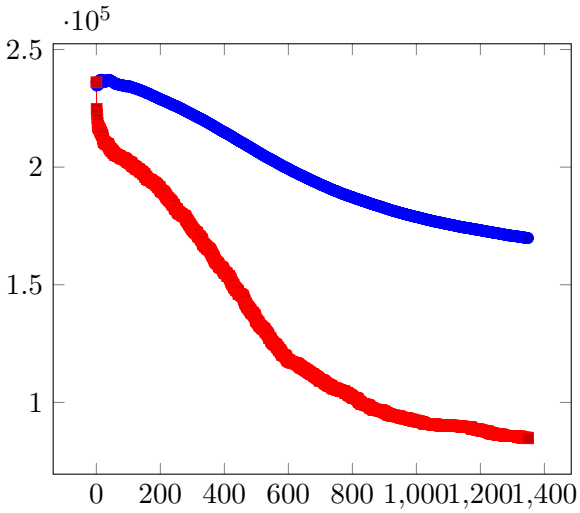

Figure 3: Atualização de feromônio com melhor fitness, SJC3b.dat

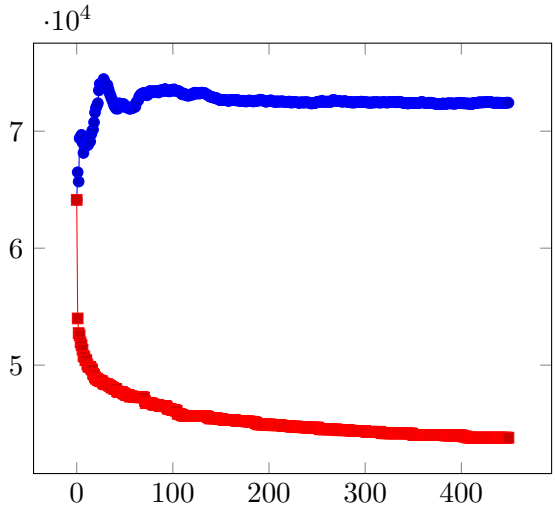

Figure 4: Atualização de feromônio com média, SJC1.dat

de atualização do feromônio.

3.2 Aumentando o multiplicador de penalidade

Quando se usa um modificador de penalidade menor ou maior, o que acontece com as soluções? Determinamos heuristicamente que um bom valor seria 100,

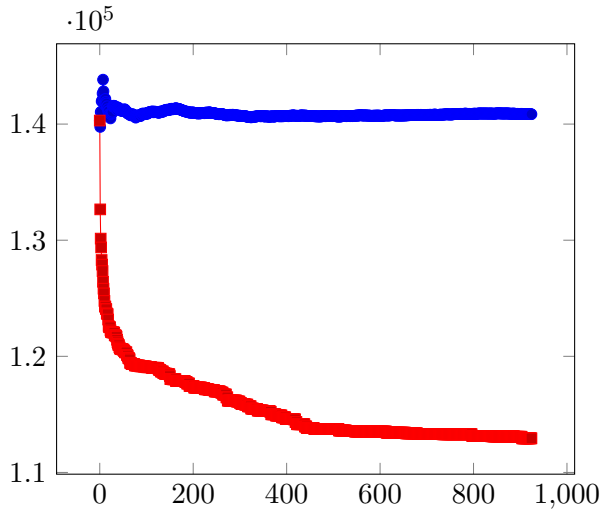

Figure 5: Atualização de feromônio com média, SJC2.dat

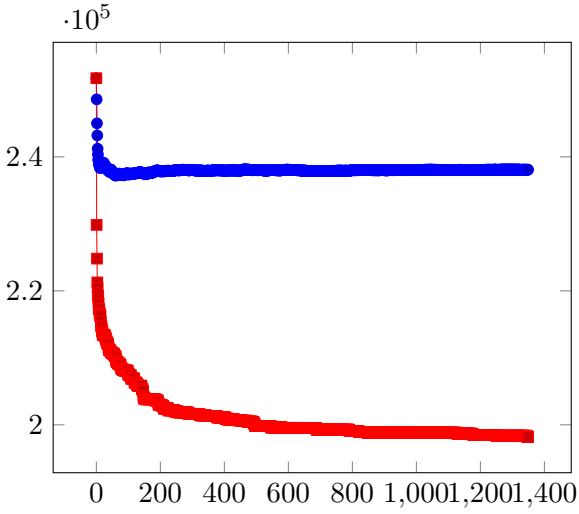

Figure 6: Atualização de feromônio com média, SJC3b.dat

porém vamos experimentar com 10 e 1000. Os resulados estão nas figuras de 7 a 11 (o gráfico da instância SJC3b.dat com o modificador 100 não pode ser plotado por limite de recursos computacionais, porém é possível deduzir que ele se assemelha às outras duas instâncias, pois elas são muito similares).

Claramente usar uma penalidade alta prejudicou a exploração do algo-

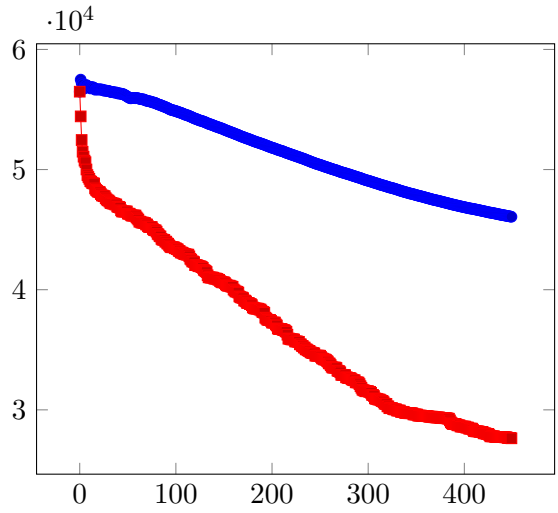

Figure 7: Modificador de penalidade = 10, SJC1.dat

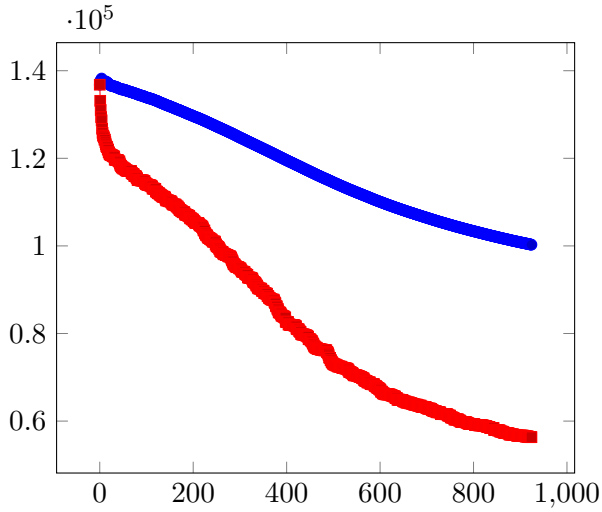

Figure 8: Modificador de penalidade = 10, SJC2.dat

ritmo pelo excesso de feedback negativo, como previsto. As soluções que estavam sendo encontradas, ao final de cada experimento, pioravam muito devido à penalidade, o que explica a média das fitness aumentando.

Por outro lado, usar uma penalidade mais branda favoreceu o funcionamento do algoritmo e gerou soluções melhores, presume-se, pela abertura

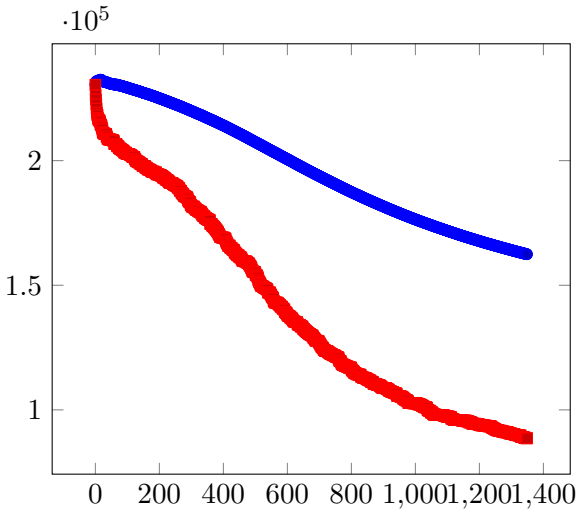

Figure 9: Modificador de penalidade = 10, SJC3b.dat

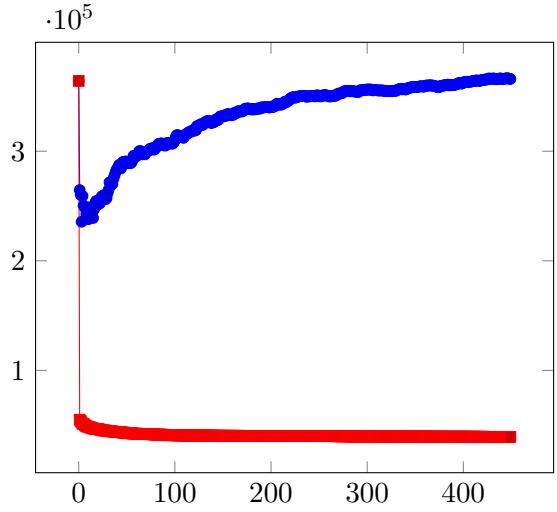

Figure 10: Modificador de penalidade = 1000, SJC1.dat

maior do espaço de exploração das formigas. Observa-se que, até o seus últimos momentos, a média das soluções ainda caia, o que sequer era sempre verdade para o multiplicador 100 (os valores ficavam estagnados mais cedo).

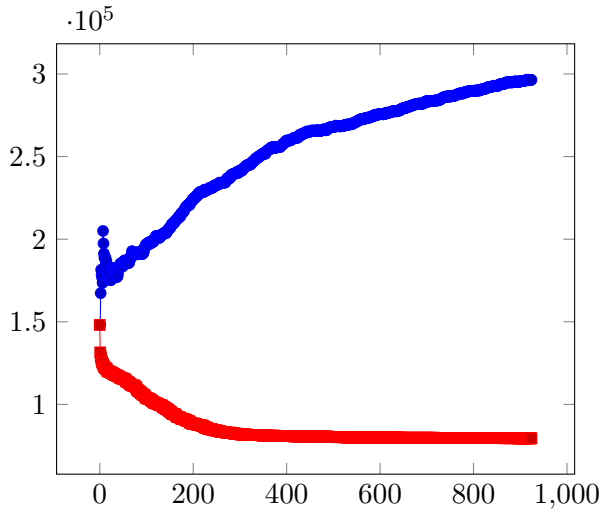

Figure 11: Modificador de penalidade = 1000, SJC2.dat

3.3 Resultados e conclusão

Contra-intuitivamente, vemos que, apesar de usar a melhor solução como referência para atualizar o feromônio gerar resultados melhores, o que sugere que o algoritmo exige uma maior convergência, aplicar uma penalidade menor pode levar a melhores resultados, apesar de a diferença entre os resultados com $P = 10$ e $P = 100$ não serem tão diferentes.

A diferença no gráfico não é tão notável, porém, dentre todos os experimentos, as melhores soluções encontradas para o estado inicial do algoritmo (testes das figuras 1 a 3) nas instâncias de teste foram [29373.9, 50511.2, 79112]. O mesmo algoritmo, usando multiplicador de penalidade 10, encontrou [24294.5, 49008.2, 80614.1] e, com multiplicador 1000, [36109.1, 72574.7, 105176]. Para referência, as soluções ótimas são [17246.53, 33225.88, 40635.80]. Uma busca aleatória encontrou resultados a cerca de 60000 na primeira instância.

Tudo isso sugere um equilíbrio singelo de parâmetros para esse algoritmo para que ele funcione de forma correta. Como sugerem os resultados, não é preciso se importar apenas com a convergência do algoritmo ou a sua capacidade de exploração, mas ambos, e um tuning de parâmetros não é o melhor para todas as instâncias.

4 Conclusões

Algumas conclusões podem ser tiradas a partir desse trabalho sobre a otimização de colônias de formigas. Primeiramente, quando se fala de ACO, devemos pensar além to tuning de parâmetros.

Diferente de um algoritmo genético, a flexibilidade dessa metaheurística não é tão abrangente para que possamos modelá-la tão confortavelmente para quaisquer tipos de problemas. A modelagem do problema é muito importante. Em França et al. [1], uma abordagem diferente, com cada formiga representando uma solução inteira e com meios de evitar que soluções ilegais apareçam foi formulada para este problema.

Outra possibilidade no caso desse algoritmo seria produzir apenas feedback positivo com os feromônios e usar uma função evaporação, como funcionam tradicionalmente as modelagens de ACO.

Pode-se argumentar que existem casos em que nem sequer vale a pena usar o ACO e deve-se procurar uma outra estratégia. Problemas muito distantes da abstração da colônia de formigas precisam de uma modelagem complexa para serem resolvidos assim, porém outros modelos talvez sejam mais fáceis de lidar. No caso do problema das p-medianas, por exemplo, o PSO (Particle Swarm Optimization) se apresenta como uma alternativa mais viável, se considerarmos essa perspectiva.

5 Referências

1. *Max Min Ant System and Capacitated p-Medians: Extensions and Improved Solutions.* Fabrício Olivetti de França, Fernando J. Von Zuben, Leandro Nunes de Castro. Informatica 29 (2005) p. 163–171