TRABALHO PRÁTICO 1:
Um Algoritmo Genético para resolver o problema da p-Mediana com restrições de capacidade

Artur Rodrigues

[1]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais (UFMG)

artur@dcc.ufmg.br

1. INTRODUÇÃO

Problemas de localização de facilidades têm diversas aplicações em telecomunicações, transportes, orquestração de tarefas e distribuição. Uma importante maneira de medir a eficácia de uma localidade de uma facilidade é avaliando o decaimento da média (total) das distâncias em função do aumento da acessibilidade e eficiência da facilidade. O problema da p-Mediana é um dos problemas mais conhecidos no âmbito de localização de facilidades e aplica essa avaliação ao tentar minimizar a distância (total) entre as demandas e um número p de localidades selecionados, chamadas de medianas. O custo total da solução é a soma das distâncias entre os pontos de demanda e as medianas.

No problema da p-Mediana não capacitado, é considerado que cada facilidade candidata a se tornar mediana pode satisfazer um número ilimitado de pontos de demanda. Em contraste, no problema da p-Mediana capacitado, cada facilidade candidata à mediana tem uma capacidade fixa, isto é, um número máximo de unidades de demanda que ela pode satisfazer. Esse problema é NP-Difícil, portanto, mesmo heurísticas especializadas em resolve-lô demandam um esforço computacional considerável.

Nesse trabalho o problema da p-Mediana capacitado é resolvido através de algoritmos genéticos. Adicionalmente, é empregada uma técnica para o assinalamento de pontos de demanda às p medianas candidatas. Para avaliar a qualidade dessa heurística, o método é estudado em comparação a solução ótima para algumas instâncias selecionadas. Finalmente, é feita uma avaliação do impacto de diversos parâmetros nos resultados obtidos.

2. MODELAGEM DO ALGORITMO GENÉTICO

Algoritmo genéticos (GAs) compõem uma classe de heurísticas adaptativas baseadas no conceito de avaliação do "melhor adaptado". O algoritmo genético desenvolvido nesse trabalho é detalhado a seguir.

2.1. Representação do Indivíduo

Cada indivíduo (cromossomo) possui exatamente p genes, onde p é o número de medianas, e cada alelo de cada gene representa as coordenadas de uma facilidade e, por consequência, identifica unicamente uma localidade. No GA em questão, o genoma é interpretado como um conjunto de facilidades, no sentido matemático de conjunto, onde não existem elementos duplicados e não há ordem entre esses elementos.

2.2. Avaliação da *Fitness*

Dado que as medianas são selecionadas, cada facilidade é assinalada à uma mediana (como será apresentado adiante) e assim que todos os vértices são estabelecidos, a *fitness* do indivíduo pode ser computada através da soma das distâncias Euclidianas entre cada vértice e sua mediana. A soma mínima entre um conjunto de indivíduos representa uma solução.

2.3. Assinalamento de Facilidades

O assinalamento de facilidades à medianas candidatas, processo necessário ao cálculo da *fitness*, é feito através da alocação das facilidades com maiores demandas primeiro à mediana mais próxima, em termos de distância Euclidiana que tenha capacidade de atender a demanda. Dessa maneira, como cada mediana tem uma capacidade fixa, algumas facilidades terão de ser assinaladas à segunda (ou terceira, quarta...) mediana mais próxima.

2.4. Seleção

Algoritmos genéticos consistem de três operadores primários: 1) reprodução, 2) *crossover* e 3) mutação.

Reprodução é o processo no qual características das soluções são passadas de uma geração para a outra. A solução escolhida para reprodução é feita através de uma técnica aleatória e ao mesmo tempo enviesada conhecida como torneio. Nela são escolhidos k elementos aleatórios da população, onde esses elementos irão competir no processo de reprodução.

2.5. *Crossover*

O *crossover* usado nesse trabalho inicialmente computa dois vetores denominados vetores de câmbio, um para cada pai, como segue. O vetor de câmbio do pai 1 representa o complemento da interseção entre os cromossomos de ambos os pais em relação ao próprio cromossomo do pai 1, o mesmo ocorre para o vetor 2 em relação ao pai 2. Os conteúdos desses vetores serão cruzados de maneira aleatória e os dois vetores resultantes serão unidos à interseção dos cromossomos dos pais, produzindo dois novos cromossomos de tamanho p, denominados filhos.

Se ambos os pais são idênticos, somente um deles é passado para a nova geração no processo de reprodução.

2.6. Mutação

A mutação ocorre como se segue. O gene a ser mutado tem um de seus alelos, escolhido aleatoriamente, substituído por outro alelo aleatório (uma facilidade). Isso ocorre com a restrição de que a nova facilidade não está presente no genótipo atual do indivíduo.

2.7. Preenchimento populacional

O processo de preenchimento da nova população ocorre a partir do processo de seleção dos indivíduos do torneio e da geração de um número aleatório r que determinará quais operações de reprodução serão aplicadas, com base nas probabilidades dessas operações (mutação e *crossover*). Esse procedimento se repete até que a nova população possua o número de indivíduos desejado, como se segue:

Algorithm 1: Preenchimento populacional

```
nova população ← [];
while |nova população| < tamanho da população do
    r ← random(0, 1);
    indivíduos do torneio ← sample(população, tamanho do torneio);
    if r < probabilidade de mutação then
        mutate(melhor indivíduo do torneio);
        insere a mutação na nova população;
    end
    else if r < probabilidade de crossover then
        crossover(dois melhores indivíduos do torneio) insere filhos na nova população;
    end
    insere melhor indivíduo do torneio na nova população;
end
```

2.8. Elitismo

Durante o processo evolucionário, pode ser introduzido um fator de elitismo. Caso ele esteja presente, uma fração, definida por esse fator, dos melhores indivíduos da população atual é introduzida diretamente na nova população, sem alterações em seu material genético.

2.9. Abstrações adicionais

Tendo sido apresentado o modelo, o algoritmo evolucionário foi implementado utilizando-se a linguagem *Python*, com o auxílio da biblioteca *NetworkX*[1] que provê estruturas de dados e funções para o manuseio de Grafos. A estrutura de grafo completo é utilizada para a representação das instâncias, onde as coordenadas de uma facilidades identificam-na como um nó, e a distância entre elas é representada como o peso das arestas do grafo completo.

3. AVALIAÇÃO EXPERIMENTAL E RESULTADOS

3.1. Procedimentos

Com o intuito de se obter testes mais consistentes, os experimentos foram executados em ambiente virtualizado, com capacidade de processamento e memória primário reduzidas, 30% da capacidade da máquina hospedeira e 512MiB, respectivamente. O sistema operacional do ambiente virtualizado era Ubuntu Server 12.04 64 bits e os softwares utilizados foram interpretador Python (2.7.2) PyPy versão 1.9.0, e GCC versão 4.2.1. A máquina hospedeira possuía sistema operacional Mac OS X 10.8.2, processador *quad-core* de 2.3GHz e memória primária com capacidade de 8GiB.

Para cada experimento realizado foram aferidos os valores médios para as *fitness* do melhor indivíduo de cada geração, *fitness* do melhor e pior indivíduo em todo processo evolucionário, geração onde foi encontrada essa melhor solução, *fitness* média e o número de indivíduos repetidos na população final, tempo de execução e porcentagem de filhos melhores que os pais no processo de *crossover* durante todas as gerações. É importante ressaltar que, devido à natureza estocástica da solução, cada problema foi executado 30 vezes e os valores médios, acompanhados do desvio padrão para cada medida foram considerados.

A primeira série de experimentos utilizou a base *SJC1.dat* disponibilizada, e através dessa série foram identificados os melhores valores para os seguintes parâmetros:

[1] http://networkx.lanl.gov/

- $popsize$: tamanho da população;
- $gener$: número de gerações;
- $mutprob$: probabilidade de mutação;
- $coprob$: probabilidade de *crossover*;
- $tsize$: tamanho do torneio;
- $elitism$: fração de elitismo;

O processo de avaliação de parâmetros seguiu a seguinte metodologia:

1. Definição de parâmetros iniciais: $popsize = 100$, $gener = 100$, $mutprob = 0.001$, $coprob = 0.6$, $tsize = 2$, $elitism = 0.01$.
2. Avaliação do incremento de $popsize$ e $gener$. Definição de valores bases para esses parâmetros.
3. Avaliação da sensibilidade da solução, com base em variações nos valores de probabilidade $mutprob$ e $coprob$. Definição de melhores valores para esses parâmetros.
4. Aumento do tamanho do torneio $tsize$. Avaliação do impacto na qualidade da solução.
5. Retirada do fator de elitismo $elitism$. Avaliação do impacto na qualidade da solução.
6. Incremento do fator de elitismo $elitism$. Avaliação do impacto na qualidade da solução.
7. Combinação dos melhores parâmetros com vistas nas outras medidas aferidas, como geração onde a melhor solução foi encontrada e número de indivíduos repetidos.

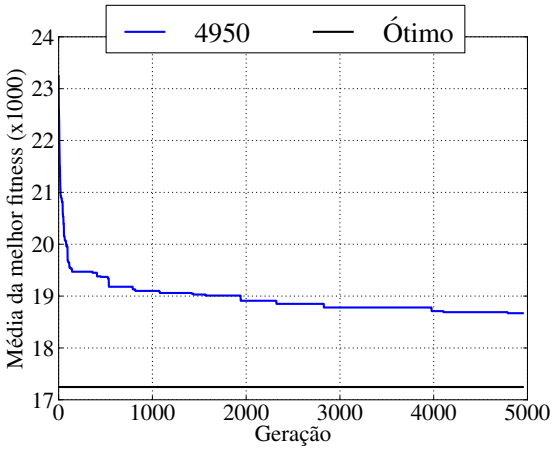
Tendo sido idenficados os melhores parâmetros para a instância base, estes foram aplicados às outras instâncias disponibilizadas: *SJC2.dat, SJC3b.dat* e *SJC4a.dat*. Os resultados podem ser encontrados na próxima seção.

3.2. Resultados

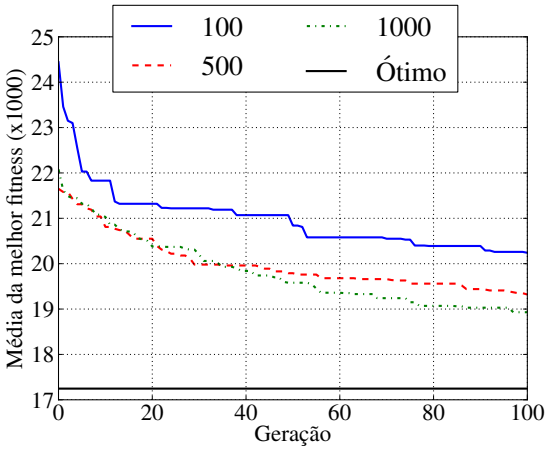
A sequência de avaliação dos parâmetros descrita na seção anterior feita na base *SJC1.dat* pode ser entrada na tabela 1. A seguir, descrevemos detalhadamento cada um desses passos. A primeira alteração realizada sob os parâmetros iniciais, um aumento para 500 gerações, sinalizou para uma melhora na qualidade da solução, o erro caiu de 18.3% para 16.0%, todavia, a solução foi encontrada, em média, na geração 235, indicando que o alto numéro de gerações não contribuiu plenamente para a qualidade da solução. Como consequência do prematuro encontro da melhor solução, a configuração apresentou uma das menores *fitness* médias. Outro detalhe interessante dessa configuração é a baixa porcentagem de filhos melhores que os pais na operação de *crossover*. Uma visão para gerações até 4950, número de vértices no grafo dessa instância, pode ser visto na figura 1a.

Em seguida, tendo sido retornado o valor do número de gerações para o valor inicial, foi elevado o tamanho da população para 500, o que resultou numa melhora significativa, diminuindo o erro da solução encontrada para 11.8%. Realizando novamente o aumento do número de gerações para 500, o erro reduziu ainda mais, chegando ao índice de 9.2%, entretando, ficou evidenciado o grande número de indíviduos repetidos na solução final, e novamente o "desperdício" de gerações (a melhor solução foi encontrada, em média, na geração 257). Uma avaliação mais profunda sobre o impacto do tamanho da população pode ser encontrado na figura 1b, onde fica claro que o aumento nesse valor implica, até certo ponto, em uma menor pressão seletiva. Conclui-se que bons valores para os parâmetro tamanho da população e número de gerações são 200 e 250, respectivamente.

O estudo continuou, dessa vez avaliando o aumento da probabilidade de *crossover* para 0.6 para 0.8, o que não representou nenhuma melhora significativa, levando à manutenção do



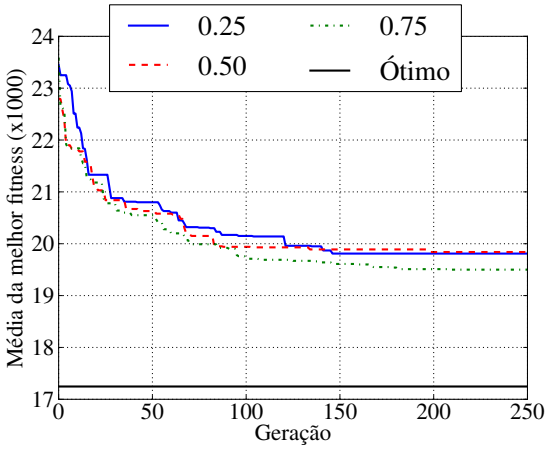
(a) Geração e a *fitness*



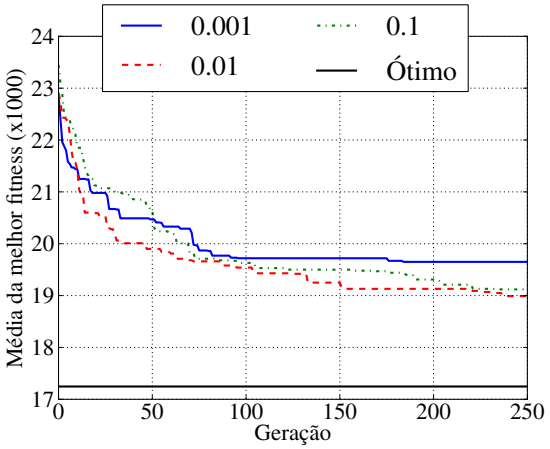
(b) Impacto do tamanho da população na *fitness*

Figura 1: Número de Gerações e Tamanho da População

valor desse parâmetro em 0.6. Essa estabilidade pode ser observada na figura 2a. Em seguida, foi alterado o valor da probabilidade de mutação, o que causa uma diminuição na pressão seletiva. A primeira alteração de 0.001 para 0.01 não trouxe melhoras visíveis a não ser o encontro da solução em uma geração mais avançada, mas alteração seguinte, para 0.1, implicou diretamente na melhora da solução, fazendo com que o erro caísse de 11.7% para 9.8%. Esse valor foi considerado ideal, pois, ao ser evidenciado que um aumento para 0.3 nesse parâmetro implica num retorno ao valor do erro anterior: 11.4%, não obstante o número de indivíduos repetidos tenha caído de 27, em média, para 19. O gráfico da figura 2b evidencia a diminuição da pressão seletiva com o aumento da probabilidade de mutação.



(a) Impacto da probabilidade de *crossover* na *fitness*

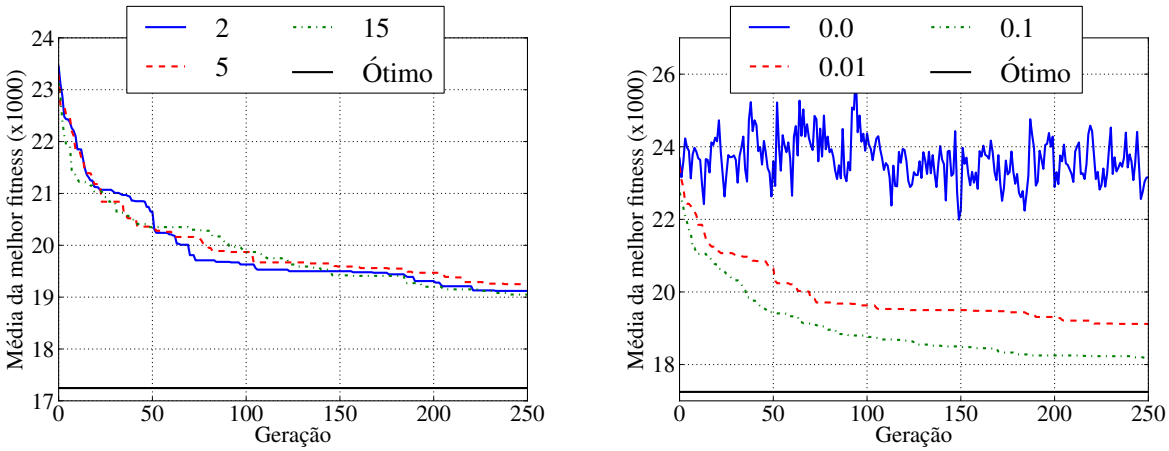


(b) Impacto da probabilidade de mutação na *fitness*

Figura 2: *Crossover* e Mutação

Adiante, avaliou-se o impacto do tamanho do torneio. O valor inicial de 2 foi elevado para 5, o que ocasionou numa solução pior: o erro subiu de 9.8% para 11.2%. Assim, o valor de 2 para esse parâmetro foi mantido no restante da avaliação. Uma análise mais completo do tamanho do torneio na qualidade da solução pode ser feita a partir do gráfico da figura 3a, onde percebe-se a elevação nesse valor não traz benefícios e a pressão seletiva permanece a mesma.

Por fim, a retirada do elitismo ocasionou uma piora signficantiva no melhor valor da *fitness*, quando o erro subiu de 9.8% para 20.3%. Além disso, essa melhor solução foi encontrada demasiadamente cedo, em média, na geração 68. Em contrapartida, um aumento no valor, passando de 0.01 para 0.1, resultou na melhor configuração encontrada para a avaliação experimental: um erro médio de 5.65%, sendo, além disso, o terceiro conjunto de parâmetros mais rápido entre os testados, definitivamente uma ótima configuração, que foi inclusive usada nas instâncias restantes. Esses resultados podem ser constatados no gráfico da tabela 3b, onde fica claro que o aumento do fator de elitismo ocasiounou um aumento da pressão seletiva e também uma melhora na qualidade da solução. A título de experimentação, ainda foi experimentado uma última configuração com probabilidade de mutação de 0.3 e fator de elitismo de 0.25; essa configuração se mostrou também muito boa, com um erro médio de 5.9%, a terceira melhor *fitness* média e a segunda execução mais rápida.



(a) Impacto do tamanho do torneio na *fitness*

(b) Impacto do fator de elitismo na *fitness*

Figura 3: Tamanho do Torneio e Elitismo

Dessa maneira, chegou-se a melhor configuração, que foi utilizada nas demais instâncias:

- tamanho da população: 200
- número de gerações: 250
- probabilidade de mutação: 0.1
- probabilidade de *crossover*: 0.6
- tamanho do torneio: 2
- fração de elitismo: 0.1

Tendo essa configuração em vista, a figura 4 mostra como a quantidade de indivíduos repetidos cresce com o processo evolucionário, ao passo que a fitness média tende a diminuir.

Finalmente, essa configuração se mostrou também boa nas demais instâncias, *SJC2.dat, SJC3b.dat* e *SJC4a.dat*, obtendo erros de 4.12%, 6.04% e 9.96% respectivamente. Possiíveis fontes de melhorias estão no aumento do tamanho da população e do número de gerações proporcionalmente ao número de arestas dos grafos dessas instâncias. Detalhes adicionais para essas instâncias podem ser encontrados na tabela 2.

Tabela 1: Avaliação de parâmetros para a instância *SJC1.dat*

Operação	Melhor Fitness		Erro	na Geração		Pior Fitness		Fitness Média		Indiv. Rep.		Tempo de Exec.(s)		Filhos Melhores(%)
	Média	Desvio P.	Média	Média	Desvio P.	Média	Desvio P.	Média	Desvio P.	Média	Desvio P.	Média	Desvio P.	Média
0) população inicial	20402.10	665.91	18.30%	70.0	22.00	83982.73	3031.61	34273.16	7344.27	29.2	6.97	17.05	0.13	49.34%
1) gener=500	20010.37	403.13	16.03%	234.9	152.99	85913.78	4588.55	20326.80	480.76	97.9	1.14	85.04	2.02	30.53%
2) gener=100, popsize=500	19284.07	511.12	11.81%	83.0	10.86	93121.14	3363.94	34607.48	2206.09	66.0	11.18	38.49	1.08	49.31%
3) gener=500, popsize=500	18842.85	306.43	9.26%	257.0	129.07	93292.44	2990.64	20236.16	912.97	491.0	6.63	182.30	4.98	40.28%
4) gener=250, popsize=250	19329.45	392.82	12.08%	142	37.14	88663.55	5077.01	27534.42	2696.14	149.0	24.15	37.43	1.15	49.11%
5) gener=200, popsize=250, cprob=0.8	19310.84	467.83	11.97%	134.0	50.77	89492.99	3135.85	27399.69	3350.50	147.0	18.33	43.93	1.08	48.95%
6) cprob=0.6, mutprob=0.01	19274.54	319.64	11.76%	201.0	31.40	88850.68	3219.30	28512.04	3587.08	121.0	16.16	39.27	1.17	49.27%
7) mutprob=0.1	18944.68	377.56	9.85%	204.0	42.56	92126.15	3257.53	35351.81	2392.47	27.0	7.21	39.73	1.79	49.29%
8) mutprob=0.3	19219.22	354.24	11.44%	193.0	46.66	94694.45	1977.13	38304.52	1926.10	19.0	3.87	40.02	1.95	49.51%
9) mutprob=0.1, tsize=5	19180.68	354.23	11.21%	204.0	26.59	92282.33	4585.54	34561.99	1397.58	29.0	5.48	37.78	2.10	49.29%
10) tsize=2, elitism=0	20751.42	522.79	20.32%	68.0	57.67	93790.43	3412.25	38953.09	4866.43	23.0	8.72	40.08	1.30	49.04%
11) elitism=0.1	18221.37	183.96	5.65%	200.0	44.69	87181.98	4258.95	24458.78	1656.04	118.0	6.56	33.06	1.16	48.76%
12) elitism=0.25, mutprob=0.3	18276.63	352.72	5.97%	175.0	30.50	85356.41	4601.98	23430.54	1195.40	117.0	8.12	27.23	1.39	47.89%

Tabela 2: Melhor configuração de parâmetros aplicada às demais instâncias

Instância	Melhor Fitness		Erro	na Geração		Pior Fitness		Fitness Média		Indiv. Rep.		Tempo de Exec.(s)		Filhos Melhores(%)
	Média	Desvio P.	Média	Média	Desvio P.	Média	Desvio P.	Média	Desvio P.	Média	Desvio P.	Média	Desvio P.	Média
SJC2.dat	34595.741	288.97	4.12%	220.0	22.83	173646.60	4626.67	39646.87	858.49	111	8.72	105.34	4.95	48.77%
SJC3b.dat	43088.299	593.01	6.04%	236.0	17.35	139948.18	16204.68	45019.74	801.56	105	12.33	333.82	7.25	49.72%
SJC4a.dat	68001.947	1107.33	9.96%	210.0	16.69	275545.13	21618.21	72824.61	903.47	99	13.76	482.35	6.32	48.70%

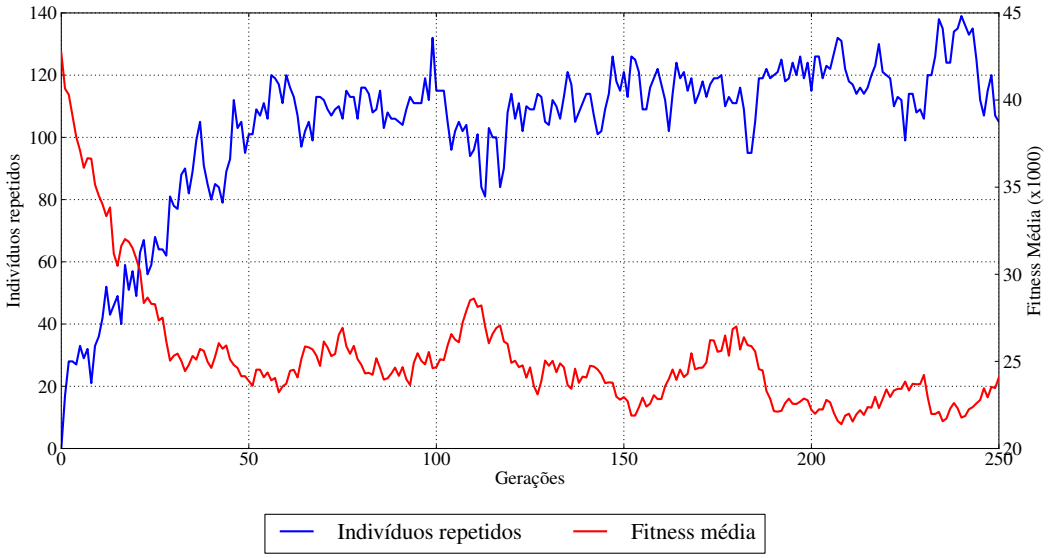


Figura 4: Número de indivíduos repetidos e *fitness* média

4. CONCLUSÃO

O problema da p-Mediana é uma importante variação do problema de localização de facilidades onde p medianas são economicamente selecionadas para servir um conjunto de facilidades demandantes de maneira que as capacidades das medianas seja respeitada. Nesse trabalho foi apresentada uma abordagem genética para a solução desse problema e o comportamento e eficiência dessa abordagem foram avaliados sob diferentes perspectivas.

Tendo em vista a adoção de uma outra heurística para o assinalamento de facilidades a medianas, o grande aprendizado desse trabalho vem, em primeira instância, da oportunidade de por em prática os conceitos que movem técnicas evolucionarias e, posteriormente, avaliar o impacto dos parâmetros na qualidade da solução. Ficou claro, por exemplo, que a pressão seletiva desempenha papel fundamental no resultado das soluções encontradas, sendo, portanto, essencial a inserção de fatores que contrabalancem esse efeito (aumento da probabilidade de mutação), e que é benéfica a presença de um fator de elitismo no processo evolucionário.

No caso do trabalho em questão, isso ficou mais evidente, ao ser observado que uma heurística orientada pela busca aleatória no espaço de soluções (através de mutações mais frequentes) produziu melhores soluções que a busca puramente evolucionária através de cruzamento.

Finalmente, fica sinalizado para trabalhos futuros, a avaliação da utilização de outros conceitos como *niching* e otimização multi-objetiva que podem aperfeiçoar ainda mais a heurística. As restrições de desempenho do algoritmo também sugerem o estudo de uma abordagem evolucionária paralela para a solução desse problema.

Referências

Alp, O., Erkut, E., and Drezner, Z. (2003). An efficient genetic algorithm for the p-median problem. *Annals of Operations Research*, 122:21–42. 10.1023/A:1026130003508.

Chiyoshi, F. and GalvÃ£o, R. (2000). A statistical analysis of simulated annealing applied to the p-median problem. *Annals of Operations Research*, 96:61–74. 10.1023/A:1018982914742.

Christofides, N. (1975). *Graph theory: An algorithmic approach (Computer science and applied mathematics)*. Academic Press, Inc., Orlando, FL, USA.

Correa, E. S., Steiner, M. T. A., Freitas, A. A., and Carnieri, C. (2004). A genetic algorithm for solving a capacitated p-median problem. *Numerical Algorithms*, 35:373–388. 10.1023/B:NUMA.0000021767.42899.31.

Ghoseiri, K. and Ghannadpour, S. (2007). Solving capacitated p-median problem using genetic algorithm. In *Industrial Engineering and Engineering Management, 2007 IEEE International Conference on*, pages 885 –889.

Hosage, C. and Goodchild, M. (1986). Discrete space location-allocation solutions from genetic algorithms. *Annals of Operations Research*, 6:35–46.

Mulvey, J. M. and Beck, M. P. (1984). Solving capacitated clustering problems. *European Journal of Operational Research*, 18(3):339 – 348.