DESENVOLVIMENTO DE ALGORITMOS GENÉTICOS PARA O PROBLEMA DAS *P*-MEDIANAS UTILIZANDO OPERADORES DE CRUZAMENTO CONVENCIONAIS E NÃO-CONVENCIONAIS

Maycon Guedes Cordeiro
Centro de Pesquisa Candido Mendes - UCAM-Campos
Av. Anita Peçanha, 100, Campos dos Goytacazes - RJ, CEP 28030-335
mayconguedes@yahoo.com.br

Dalessandro Soares Vianna
Departamento de Computação – UFF/PURO
Rua Recife s/n, Rio das Ostras – RJ, CEP 28890-000
dalessandrosoares@yahoo.com.br

Marcilene de Fátima Dianin Vianna
Departamento de Economia – UFF/PUCG
Rua José do Patrocínio 71, Campos dos Goytacazes – RJ, CEP 28010-385
marcilenedianin@vm.uff.br

RESUMO

Neste trabalho são propostos algoritmos genéticos (AGs) para o problema das *p*-medianas utilizando diferentes operadores genéticos de cruzamento. Foram desenvolvidos cinco operadores de cruzamento, sendo dois deles variações de operadores clássicos da literatura – operadores convencionais – e outros três não-convencionais. Dois destes operadores não-conencionais são baseados na técnica de reconexão por caminhos. O último é uma versão "reativa", a qual visa explorar de forma eficiente os diferentes operadores de cruzamento desenvolvidos. Estes AGs foram avaliados utilizando problemas testes disponíveis na literatura, para os quais já foram definidos limites inferiores. Os resultados obtidos demonstram a eficiência de se utilizar a técnica de reconexão por caminhos como operador de cruzamento.

PALAVARAS CHAVE. Problema das *p*-medianas, algoritmos genéticos, algoritmos reativos.

ABSTRACT

The aim of this study is propose genetic algorithms (GA) for the *p*-median problem using different genetic operators of crossover. Five crossover operators were developed, in which two of them are variations of classic operators on the literature – conventional operators – and three are non-conventional operators. Two of these non-conventional operators are based on the path-relinking technique. The latter is a "reactive" approach, which seeks to explore efficiently the different crossover operators developed. These GA's were evaluated using test problems available in literature, for which have already been set lower limits. The results demonstrate the efficiency of using the technique of path-relinking as a crossover operator.

KEYWORDS. *p*-median problem, genetic algorithms, reactive algorithms.

1. Introdução

No clássico problema das *p*-medianas pretende-se localizar facilidades para melhor servir a clientes de forma a otimizar um certo critério (DREZNER, 1995). "Facilidades" é um termo genérico que pode ser substituído por postos de combustível, escolas, hospitais, indústrias, antenas de telecomunicações, etc. Já o termo "clientes" refere-se a motoristas, estudantes, pacientes, revendedores, receptores de ondas ou qualquer termo a ser servido pelas facilidades.

O problema das *p*-medianas é NP-difícil (GAREY; JOHNSON, 1979) e, desta forma, o tempo para se obter a solução ótima cresce exponencialmente à medida que se aumenta os dados de entrada (número *p* de facilidades e o número *n* de clientes). Na literatura científica são encontrados diferentes artigos que abordam o problema das *p*-medianas, e suas variações, utilizando heurísticas/metaheurísticas. Dentre eles estão: Teitz e Bart (1968) e Resende e Werneck (2003) que propuseram heurísticas baseadas em busca local; heurísticas busca tabu foram propostas por Goncharov e Kochetov (2002) e Mladenović *et al*. (1996); Lorena *et al*., (1999), Fleszar e Hindi (2008) e Crainic *et al*. (2004) desenvolveram métodos VNS (*Variable Neighborhood Search*); algoritmos genéticos foram desenvolvidos por Alp *et al*. (2003) e Chaudhry *et al*. (2003); estratégias neurais foram propostas por Dominguez e Munoz (2008) e Domínguez Merino *et al*. (2003); Levanova e Loresh (2004) e García-López *et al*. (2003) propuseram, respectivamente, métodos baseados em colônia de formigas e *scatter search*.

A ligação cliente/facilidade requer grandes investimentos e possui despesas operacionais muito altas, principalmente no que se refere a distâncias percorridas em autovias e qualidade do sinal em telecomunicações. Por este motivo, uma redução no custo cliente/facilidade pode representar uma diferença considerável na despesa final às empresas que a custeiam. Visando resolver esse problema, há alguns estudos na literatura para tentar diminuir ao máximo este custo, reduzindo assim as despesas e aumentando a qualidade desses serviços.

Dos trabalhos pesquisados na literatura científica, pode-se destacar: Teitz e Bart (1968) e Lorena *et al*. (1999). O primeiro apresenta uma heurística eficiente para o problema, a qual é usada, neste trabalho, na construção de soluções iniciais. No trabalho de Lorena *et al*. (1999), além de uma heurística para o problema, limites inferiores para um conjunto de problemas testes são apresentados. O objetivo deste trabalho foi desenvolver heurísticas eficientes que alcançassem resultados iguais ou melhores aos obtidos pelas heurísticas propostas por Teitz e Bart (1968) e Lorena *et al*. (1999). Para alcançar tais resultados, dentre diversas metodologias presentes na literatura, foi escolhida a metaheurística de algoritmos genéticos (AG). Utilizando-se esta metaheurística, foram realizadas modificações específicas para o tratamento do problema das *p*-medianas em sua estrutura, desenvolvendo assim uma heurística específica para o tratamento deste problema.

2. Definição do problema das *p*-medianas

No problema das *p*-medianas abordado neste trabalho pretende-se localizar *p* facilidades (medianas) para melhor servir a *n* clientes, de forma a minimizar o custo total (a soma das distâncias de cada cliente à sua mediana mais próxima).

Os dados relevantes para um problema das *p*-medianas são:

- um número finito de pontos (vértices), com valores conhecidos de demanda, denominados clientes ou pontos de demanda;
- um número finito de locais candidatos para a instalação de facilidades (medianas). Neste trabalho considera-se que os pontos de demanda são candidatos para a instalação de facilidades;
- a distância entre cada ponto de demanda e os locais candidatos. Essas distâncias podem ser calculadas sobre a rede de caminhos que conectam os pontos (ex.: ruas, dutos e cabos) ou como distâncias euclidianas, ou seja, uma simples reta que liga dois pontos; e
- número *p* de facilidades a serem instaladas.

Como base de dados para este trabalho foram usadas as mesmas coordenadas (x,y) no plano cartesiano apresentadas em (LORENA *et al*., 1999) e em seguida foi gerada uma matriz de distâncias utilizando distância euclidiana.

3. Algoritmos genéticos desenvolvidos

Foram desenvolvidos neste trabalho cinco algoritmos genéticos (AGs), utilizando diferentes operadores genéticos de cruzamento. Todos os AGs desenvolvidos seguem o modelo descrito na Subseção 3.2. Na geração da população inicial dos AGs desenvolvidos foi usado um algoritmo construtivo ao invés de simples soluções aleatórias. Entre os algoritmos construtivos analisados o que teve melhores resultados foi o algoritmo **Teitz and Bart** (Teitz & Bart, 1968), que é apresentado em detalhes na Subseção 3.1. Nas subseções seguintes são descritas as demais etapas dos AGs desenvolvidos.

3.1. Algoritmo Teitz and Bart

Proposto por Teitz e Bart (1968), é um método aproximado para a determinação de p medianas entre n vértices candidatos à mediana do problema, baseado na substituição de vértices.

Escolhe-se, inicialmente, p vértices aleatórios para formar um conjunto S inicial. S é considerado uma aproximação inicial de V_p (conjunto ideal de facilidades). Sendo V o conjunto de todos os vértices do grafo, verifica-se se pode haver um vértice v_i pertencente ao conjunto V - S que possa substituir um vértice v_j pertencente a S, produzindo um novo conjunto S' ($S' = S + \{v_i\} - \{v_j\}$), tal que o somatório das distâncias entre cada cliente (vértices de V) e a mediana mais próxima deste seja menor em S' do que em S. Se isto for possível, é feita a substituição de v_j por v_i e S' passa a ser a melhor aproximação, até o momento, para o conjunto V_p. O processo continua até obter-se um conjunto S^*, onde nenhuma substituição de vértice de S^* por outro vértice em $V - S^*$ produza melhora no somatório das distâncias entre os clientes e as medianas.

3.2. Algoritmo genético básico

O pseudocódigo de um algoritmo genético básico é mostrado na Figura 1. Nele pode-se notar que os algoritmos genéticos começam com uma população P de *tampop* cromossomos, onde cada estrutura codifica uma solução do problema. O desempenho de cada cromossomo é avaliado com base em uma função de avaliação ou aptidão. Os melhores cromossomos tenderão a ser os progenitores da geração seguinte, possibilitando que as suas características sejam transmitidas para as próximas gerações (PALAZZO, 1997).

```
Algoritmo AG {
      t ← 0;                          //contador
      Inicia_população(P,t);          //iniciar uma população de tampop cromossomos
      Avaliação(P,t);                 //avaliar aptidão dos cromossomos da população
      Repita até (critério de parada satisfeito){
            t ← t + 1;                //incrementar o contador de gerações
            Seleção_dos_pais(P,t);    //selecionar os pares para cruzamento
            Recombinação(P,t);        //realizar cruzamento dos pares selecionados
            Mutação(P,t);             //perturbar o grupo gerado pelo cruzamento
            Avaliação(P,t);           //avaliar as novas aptidões
            Seleção_mais_aptos(P,t);  //selecionar os mais aptos a sobreviver

      }
}
```

Figura 1 – Pseudocódigo básico de um Algoritmo Genético. Fonte: Sobrinho e Girardi (2003).

3.3. Construção da população inicial

A população inicial é composta de *tampop* cromossomos, onde cada cromossomo é codificado como um vetor de p posições, nas quais são armazenadas as medianas escolhidas. A escolha dessas medianas, ou seja, a construção de cada cromossomo é realizada utilizando o algoritmo construtivo **Teitz and Bart** descrito na Subseção 3.1.

3.4. Avaliação

A avaliação ou aptidão do cromossomo é calculada através do somatório das distâncias euclidianas entre cada cliente e a mediana pertencente à solução que está mais próxima dele.

3.5. Critério de parada

O critério de parada por tempo é utilizado neste trabalho. Este tempo foi definido empiricamente de acordo com as dimensões de cada problema teste avaliado. A Equação 1 foi utilizada para definição dos tempos, onde p e n são, respectivamente, o número de medianas e de clientes do problema.

$$Tempo = (p \times n)/100 \tag{1}$$

3.6. Seleção dos pais

Os pais são selecionados por um método conhecido como "roleta" (GOLDBERG, 1989), no qual quanto melhor a avaliação do cromossomo, maior será a chance deste ser escolhido para o cruzamento. O cálculo da probabilidade $prob_i$ do cromossomo i da população P foi definido empiricamente. A Equação 2 mostra como é realizado o cálculo de $prob_i$, onde f_i representa a avaliação ou aptidão do cromossomo i e $tampop$ representa o número de cromossomos da população P.

$$prob_i = \frac{f_i^{-4} \times 100}{\sum_{j=1}^{tampop} f_j^{-4}} \tag{2}$$

É importante citar algumas observações:

- o valor de aptidão deve ser sempre elevado a um número negativo para priorizar as menores avaliações, já que o que se quer é minimizar o custo;
- elevando a menos quatro aumenta-se consideravelmente a chance dos melhores cromossomos serem escolhidos.

3.7. Cruzamento ou recombinação

O operador de cruzamento ou recombinação cria novos cromossomos através da combinação de dois ou mais cromossomos. A idéia intuitiva por trás do operador de cruzamento é a troca de informação entre diferentes soluções candidatas (VIANNA & RIBEIRO, 2004).

Com o intuito de avaliar diferentes operadores de cruzamento para o problema das p-medianas, foram desenvolvidos cinco operadores:

- **Ponto de corte**: baseado no clássico operador de ponto de corte (GOLDBERG, 1989), neste operador cada cromossomo pai é dividido em um ponto (chamado ponto de corte) definido aleatoriamente. Dois novos cromossomos são gerados permutando-se a metade inicial de um cromossomo pai com a metade final do outro. Um exemplo é ilustrado na Figura 2.

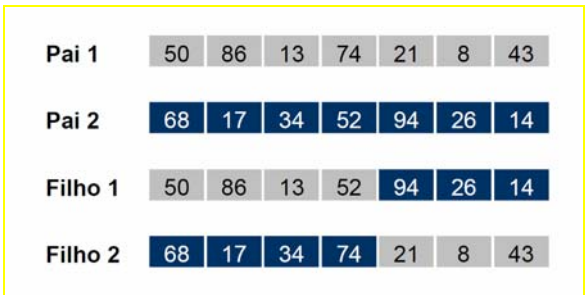

Figura 2 – Ponto de corte.

- **Ponto mais próximo:** este algoritmo baseia-se na idéia de definir cada gene do

cromossomo filho escolhendo-se aleatoriamente o pai de onde tal gene será herdado (copiado). No entanto é realizada neste trabalho uma etapa de pré-processamento com o intuito de melhorar a qualidade dos filhos gerados. Isto é feito privilegiando o cruzamento apenas entre genes que se encontram próximo no plano cartesiano. Quando os cromossomos pais são selecionados, um deles é reorganizado (posição de seus genes são alteradas) posicionando os genes mais próximos dos seus respectivos no outro pai. Em seguida uma máscara de cruzamento é gerada aleatoriamente. Onde houver 1 na máscara de cruzamento, o gene correspondente será copiado do primeiro pai e onde houver 0 será copiado do segundo. O segundo filho é gerado utilizando a mesma máscara, mas alterando a maneira que esta é interpretada: onde houver 1 na máscara, o gene correspondente será copiado do segundo pai e onde houver 0 será copiado do primeiro. A Figura 3 ilustra um exemplo de cruzamento utilizando este operador. Note que o cromossomo "pai 2" foi reorganizado pois a mediana pertencente a este mais próxima da mediana 50 (gene 1 de "pai 1") é a 34; a mediana mais próxima da mediana 86 (gene 2 de "pai 1") é a 52; e assim por diante.

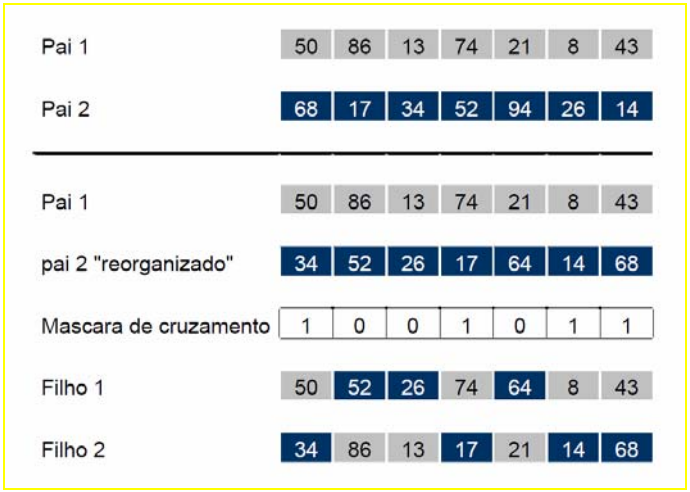

Figura 3 – Cruzamento mediana mais próxima.

- ***Path relinking* aleatório:** o uso da técnica de reconexão por caminhos (*path relinking*) como operador de cruzamento foi proposta inicialmente por Ribeiro e Vianna (2003). O intuito é explorar a trajetória entre dois cromossomos pais, retornando como filho o melhor indivíduo neste trajeto. Neste operador, dois cromossomos são selecionados como pais, sendo um deles considerado como "guia" e o outro como "partida" (que é o cromossomo corrente inicial). O objetivo é partir do cromossomo partida e chegar ao cromossomo guia por meio da adição no cromossomo corrente de genes do cromossomo guia. A cada passo, uma mediana (gene) g_i da solução corrente que não existe na solução guia é escolhida aleatoriamente para ser substituída por alguma mediana do cromossomo guia. Será escolhida para substituir g_i aquela mediana presente no cromossomo guia que acarretar o menor custo no cromossomo corrente. Este processo é repetido até que o cromossomo corrente se iguale ao cromossomo guia. É definido como filho do cruzamento o cromossomo intermediário de melhor avaliação no trajeto entre o cromossomo de partida e o guia. A Figura 4 ilustra um exemplo deste tipo de cruzamento. Neste exemplo, o cromossomo definido como filho seria o cromossomo intermediário 4.

Cromossomos	Genes							Avaliação
Cromossomo partida	50	86	13	74	21	8	43	3268
Cromossomo intermediário 1	50	86	94	74	21	8	43	3602
Cromossomo intermediário 2	50	86	94	74	21	8	26	3872
Cromossomo intermediário 3	50	68	94	74	21	8	26	3411
Cromossomo intermediário 4	50	68	94	74	52	8	26	3191
Cromossomo intermediário 5	50	68	94	34	52	8	26	3294
Cromossomo intermediário 6	50	68	94	34	52	17	26	3486
Cromossomo guia	14	68	94	34	52	17	26	3347

Figura 4 – Cruzamento *path relinking* aleatório.

- ***Path relinking* melhor:** funciona praticamente igual ao anterior. A única diferença é que ao invés de escolher aleatoriamente o gene no cromossomo corrente a ser substituído, este operador busca entre todos os genes do cromossomo corrente e do cromossomo guia aquele par que produzirá um cromossomo de menor custo.

Reativo: o tipo de cruzamento reativo, adaptado do trabalho de Prais e Ribeiro(2000) para a metaheurística GRASP, abrange todos os cruzamentos anteriormente citados, no qual, a cada iteração do AG, um método de cruzamento diferente pode ser escolhido. Cada tipo de cruzamento tem uma determinada probabilidade de ser escolhido, a qual se dá conforme um valor de desempenho ditado pela Equação 3, onde p_i representa a probabilidade do método de cruzamento i ser escolhido e q_j é o coeficiente do método de cruzamento j definido pela Equação 4.

$$p_i = \frac{q_i \times 100}{\sum_{j=1}^{4} q_j} \tag{3}$$

Na Equação 4, a variável *me* representa o melhor *fitness* alcançado durante todas iterações anteriores; α é uma constante definida para aumentar a probabilidade de escolha dos métodos que tem melhor desempenho no AG; e a_i é a média dos *fitness* gerados pelo método de cruzamento i.

$$q_i = \left(\frac{me}{a_i} \right)^{\alpha} \tag{4}$$

O valor de a_i é dado pela Equação 5, onde w representa o número de vezes que o

método de cruzamento *i* foi escolhido para o cruzamento; e f_j é o valor do *fitness* referente ao *j*-ésimo cromossomo encontrado pelo método de cruzamento *i*.

$$a_i = \frac{\sum_{j=1}^{w} f_j}{w} \quad (5)$$

Após *tampop*/2 iterações, estes coeficientes são recalculados, alterando as probabilidades de escolha de cada método de cruzamento. Vale lembrar que *tampop* representa o número de cromossomos existentes na população.

3.8. Mutação

Ao invés da mutação acontecer determinada por uma probabilidade como na maioria dos algoritmos genéticos, neste trabalho a mutação é determinada por um operador chamado **Detecção de Clones**, o qual faz a detecção de cromossomos com a mesma aptidão (foi detectado através de experimentos computacionais que na maioria das vezes que dois cromossomos possuíam a mesma aptidão, eles eram idênticos). É comum em algoritmos genéticos a população convergir, ao longo das gerações, para cromossomos idênticos ou semelhantes. A mutação proposta neste trabalho ocorre sempre que clones são detectados na geração corrente, preservando assim a diversidade da população.

Quando um clone é detectado, a mutação ocorre com probabilidade de 50%. Caso ela não ocorra, um dos cromossomos *clones* é removido da geração dando lugar a um outro cromossomo criado a partir do algoritmo de construção **Teitz and Bart** descrito na Subseção 3.1.

A estratégia de mutação proposta funciona de modo semelhante ao cruzamento por mediana mais próxima: é gerada uma máscara e onde houver 1 na máscara, o gene correspondente será substituído pelo ponto "cliente" mais próximo, onde houver 0 não há modificação, conforme ilustra a Figura 5. O percentual de valores 1 presentes na máscara é definido por um parâmetro de entrada.

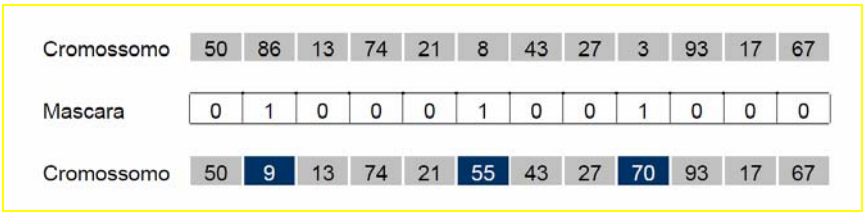

Figura 5 – Mutação.

3.9. Seleção dos mais aptos

A estrutura dos algoritmos genéticos desenvolvidos é a seguinte:

- A cada geração do algoritmo é realizado um único cruzamento gerando um ou dois filhos dependendo do tipo de cruzamento utilizado.
- Apenas o filho de melhor aptidão poderá ser inserido na população da geração posterior.
- Caso o melhor filho seja mais apto que um dos pais, será removido da população o pai com pior aptidão, dando lugar ao melhor filho.

4. Resultados computacionais

Todos os experimentos computacionais deste trabalho foram realizados em um microcomputador equipado com processador Intel Core 2 Duo CPU E4500 com *clock* de 2.20GHz e 2.0 Gb de memória RAM sob a plataforma Windows ™ XP.

Todos os algoritmos genéticos foram desenvolvidos utilizando a linguagem de

programação C, compilados no ambiente Dev-C++ versão 4.9.9.2 "IDE para programação de executáveis Win32, console ou GUI na linguagem C/C++".

As instâncias usadas durante os testes foram as mesmas utilizadas por Lorena *et al*.(1999), para as quais já foram definidos os limites inferiores. As instâncias testes podem ser obtidas através do link http://www.lac.inpe.br/~lorena/instancias.html

A Tabela 1 apresenta as instâncias com o número *n* de clientes e *p* medianas.

Tabela 1 - instâncias e medianas.

(n)	(p)	(n)	(p)
324	5	818	5
324	10	818	10
324	20	818	20
324	50	818	50
324	108	818	100
		818	150
		818	272

4.1. Experimentos realizados – Desempenho por execução

Foram desenvolvidos neste trabalho cinco algoritmos genéticos (AGs), os quais se diferenciam pelo operador genético de cruzamento utilizado. A seguir são apresentadas as nomenclaturas de cada AG, assim como o operador genético que este utiliza. É importante lembrar que todos os AGs desenvolvidos possuem a estrutura descrita na Seção 3.2.

- AGPC - Algoritmo genético com cruzamento "ponto de corte".
- AGMMP - Algoritmo genético com cruzamento "mediana mais próxima".
- AGPRa - Algoritmo genético com cruzamento "*path relinking* aleatório".
- AGPRm - Algoritmo genético com cruzamento "*path relinking* melhor".
- AGR - Algoritmo genético com cruzamento "Reativo".

No experimento realizado, cada algoritmo genético (AG) realizou 10 (dez) execuções para cada uma das 12 (doze) instâncias descritas na Tabela 1. As Tabelas 2 e 3 apresentam os resultados obtidos para $n = 324$ e $n = 818$, respectivamente. Na coluna 1 é apresentado o número *p* de medianas. O limite inferior definido em (LORENA et al., 1999) para cada instância é descrito na coluna 2. Nas colunas 3 e 4 são apresentados, respectivamente, os melhores resultados obtidos por Lorena *et al*. (1999) e o *gap* (diferença percentual entre o resultado obtido e o limite inferior). Nas colunas seguintes são apresentados o custo médio e o *gap* obtidos por cada AG desenvolvido. Foram destacados em negrito os melhores resultados (custos) obtidos para cada instância.

Tabela 2 - Instância de $n = 324$ vértices.

Tabela 3 - Instância de $n = 818$ vértices.

Analisando os resultados apresentados nas Tabelas 2 e 3, percebe-se que o algoritmo AGPRa, que utiliza o operador de cruzamento "*path relinking* aleatório", foi o que alcançou os melhores resultados, obtendo o melhor custo médio para todas as instâncias analisadas. O algoritmo AGMMP, que utiliza o operador de cruzamento "mediana mais próxima", também atingiu bons resultados, obtendo o melhor custo médio para 7 das 12 instâncias. O algoritmo AGPRm, que utiliza o operador de cruzamento "*path relinking* melhor", não obteve os resultados esperados. Isso ocorreu pois o operador de cruzamento "*path relinking* melhor" exige maior esforço computacional, o que acarreta em um menor número de gerações executadas pelo algoritmo AGPRm dentro do tempo estabelecido (critério de parada). O método AGR, que utiliza o operador de cruzamento "reativo", assim como o AGPRm, não obteve os resultados esperados. Isso ocorreu devido a grande alternância de métodos de cruzamento, onde percebeu-se que no AGR é necessário um certo número de iterações em um único método de cruzamento para haver evolução da população. O algoritmo AGPC, que utiliza o operador de cruzamento "ponto de corte", foi o que obteve os piores resultados entre os AGs desenvolvidos, o que já era esperado pela simplicidade do operador.

As Tabelas 4 e 5 apresentam os tempos médios obtidos por cada AG desenvolvido para $n = 324$ e $n = 818$, respectivamente. O tempo de uma execução de um AG é o tempo gasto até se encontrar a melhor solução daquela execução. Percebe-se que o algoritmo AGPRa, além de obter os melhores custos médios, obteve também, em geral, os melhores tempos médios.

Tabela 4 - Instância de n = 324 vértices.

Tabela 5 - Instância de n = 818 vértices.

(p)	Lorena	AGPC	AGMMP	AGPRa	AGPRm	AGR
5	102,66	42	2	**0**	37	27
10	97,48	157	28	**19**	168	204
20	**60,39**	342	267	210	295	319
50	**43,73**	859	671	516	738	790
100	**57,93**	1699	1325	1002	1425	1585
150	**66,19**	2541	1922	1486	2137	2348
272	**85,58**	4918	3547	2910	4022	4488

4.2. Experimentos realizados – Aptidão alvo

Neste teste pode-se observar o desempenho de cada método de cruzamento proposto para este trabalho em decorrência do tempo de execução dos mesmo. Foi definida a instância com 818 vértices e 50 medianas, por ser uma instância com alto grau de dificuldade, porém com

um tempo computacional viável. Foram realizadas 100 independentes execuções de cada AG para esta instância. Cada execução terminava quando uma solução de valor menor ou igual a um certo valor alvo era encontrado. Este valor "*fitness*" foi escolhido de tal forma que o AG pudesse terminar depois de um tempo computacional considerável. São definidos pelo usuário 3 parâmetros de entrada: *fitness*, tempo e número de execuções.

- ***Fitness*:** é o valor alvo "aptidão" a ser alcançado; caso o programa alcance ou ultrapasse o *fitness* alvo, o programa é finalizado, registrando em um arquivo o tempo gasto para alcançar o alvo. Para este teste foi definido o *fitness* alvo de 147.500.
- **Tempo:** é o tempo limite que o programa ficará em execução para que atinja o *fitness* alvo. Caso o tempo limite se esgote, o programa é finalizado e nada é registrado já que não foi alcançado o *fitness* alvo. Para este teste foi definido o tempo de 1000 segundos.
- **Número de execuções:** é a quantidade de vezes que o programa será executado para a amostragem deste teste. Para este teste foi definido que o programa terá 100 amostragens da sua execução.

A Figura 6 mostra o desempenho da cada um dos métodos de cruzamentos proposto neste trabalho em para este teste. O eixo x representa o tempo de execução em segundos para alcançar o alvo e o eixo y mostra a probabilidade do *fitness* alvo ser alcançado em decorrência ao tempo.

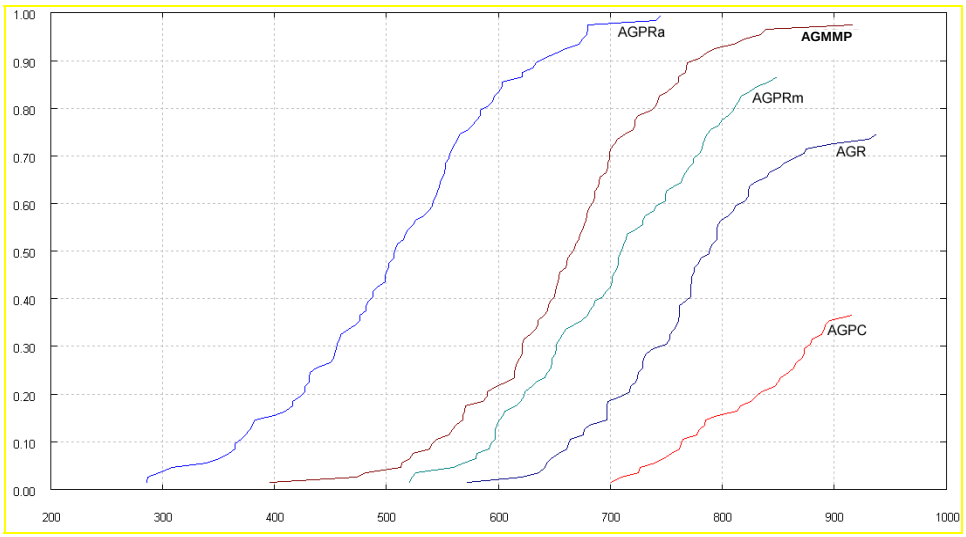

Figura 6 – Aptidão alvo.

Dentre os 5 AG's testados 2 se destacaram:

- **AGMMP:** método de cruzamento que apresenta o segundo melhor desempenho. Consegue atingir o alvo em quase 100% das execuções.
- **AGPRa**: é o método que mostra o melhor desempenho dentre os cinco métodos propostos, conseguindo atingir o alvo em 100% das execuções. Observa-se que em de 50% das execuções o AGPRa consegue atingir o alvo em pouco mais de 500 segundos, antes mesmo que a maioria dos outros métodos propostos consiga atingir o alvo pela primeira vez.

5. Conclusão

Neste trabalho foram propostos cinco algoritmos genéticos para o problema das *p*-medianas,(AGPC, AGMMP, AGPRa, AGPRm, AGR), os quais se diferem pelo operador

genético de cruzamento utilizado. Dois desses operadores, denominados "*Path relinking aleatório*" e "*Path relinking melhor*", foram aplicados pela primeira vez ao problema das *p*-medianas neste trabalho, onde se obteve bons resultados, utilizando a técnica de reconexão por caminhos, originalmente proposta como uma estratégia de intensificação para explorar trajetórias conectando soluções elites obtidas por heurísticas busca tabu e busca dispersa (*scatter search*) (GLOVER, 1996, 2000; GLOVER & LAGUNA, 1997; GLOVER *et al*., 2000). Outro operador de destaque é o operador denominado neste trabalho como "Mediana mais próxima" desenvolvido pela primeira vez neste trabalho que se baseia em uma ideia já conhecida de escolha aleatória de informações dos pais para preenchimento dos filhos, mas utiliza uma eficiente etapa de recombinação de genes durante o cruzamento.

Destaca-se também neste trabalho o desenvolvimento de um novo operador em AG, denominado "Detecção de Clone" responsável por manter a heterogeneidade na população durante as gerações.

Os experimentos realizados mostraram que o algoritmo AGPRa, que utiliza o operador de cruzamento "*Path relinking* aleatório", foi o que obteve os melhores resultados quando os custos médios foram comparados e também quando os tempos médios foram comparados. O algoritmo AGMMP, que utiliza o operador de cruzamento "Mediana mais próxima", também obteve bons resultados. Já o algoritmo AGPRm, que utiliza o operador de cruzamento "Path *relinking melhor*", e o operador AGR, que utiliza o operador de cruzamento "reativo" não obtiveram os resultados esperados.

Referências

Alp, O., Erkut, E. e Drezner, D. (2003), An efficient genetic algorithm for the *p*-median problem, *Annals of Operations Research,* 122, 21-42.

Chaudhry, S. S., He, S. e Chaudhry. (2003), P.E. Solving a class of facility location problems using genetic algorithm, *Expert Systems,* 20, 86-91.

Cotta, C. (2006), Scatter search with path relinking for phylogenetic inference*, European Journal of Operational Research*, 169, 520-532.

Crainic, T., Gendreau, M., Hansen, P. e Mladenovic, N. (2004), Cooperative parallel variable neighborhood search for the *p*-median, *Journal of Heuristics,* 10, 293-314.

Dominguez, M. E. e Munoz, P. J. (2008), A neural model for the p-median problem, *Computers & Operations Research,* 35, 404-416.

Dominguez, M. E., Munoz , P. J. e Jerez, A. J. (2003), Neural Network Algorithms for the *p*-median problem, *ESANN'2003 Proceedings – European Symposium on Artificial Neural Networks,*Bruges, Belgium, 385-391.

Drezner, D. (1995), A Survey of Applications and Methods*, NY: Springer-Verlag*.

Festa, P. e Resende, M. G. C. GRASP An annotated bibliography. Em: Ribeiro, C. C., Hansen, P. (editores). *Essays and Surveys in Metaheuristics*. Kluwer, Dordrecht, 325-367, 2002.

Fleszar, K. e Hindi, K. S. (2008), An effective VNS for the capacitated p-median problem. *European Journal of Operational Research*, 191(3), 612-622.

García-López, F., Melián Batista, B. e Moreno-Pérez, Moreno-Veja, J. A. (2002) The parallel variable neighborhood search for the *p*-median problem, *Journal of Heuristics,* 8, 375-388.

Garey, M. R. e Johnson, D. S., *Computers and intractability: a guide to the theory of NPcompleteness*, San Francisco: W. H. Freeman and Co, 1979.

Glover, F., Multi-start and strategic oscillation methods – principles to exploit adaptive memory. Em: Laguna, M., Gonzáles-Velarde, J.L. (Eds). *Computing Tools for Modeling, Optimization and*

Simulation: Interfaces in Computer Science and Operations Research, Kluwer, Dordrecht, 1-24, 2000.

Glover, F., Tabu search and adaptive memory programing – advances, applications and challenges. Em: Barr, R.S., Helgason, R.V., Kennington, J.L. (Eds). *Interfaces in Computer Science and Operations Research*, Kluwer, Dordrecht, 1-75, 1996.

Glover, F. e Kochenberger, G, *Handbook of Metaheuristics*. Dordrecht, Kluwer, 2003.

Glover, F. e Laguna, M, *Tabu Search*. Dordrecht, Kluwer, 1997.

Gordberg, D, E, *Genetic Algorithms in Search, Optimization and Machine Learning.* Massachusetts, Addison Wesley. 1989.

Levanova, T. e Loresh, M. A. (2004), Algorithms of ant system and simulated annealing for the *p*-median problem, *Automation and Remote Control,* 65, 431-438.

Lorena, L. A. N., Senne, E. L. F., Paiva, J. A. M. e Mmarcondes, S. S. P. B. (1999), Integração de um modelo de *p*-medianas a sistemas de informações geográficas, *Anais do XXXI Simpósio Brasileiro de Pesquisa Operacional*, 635-647.

Mladenovic, N., Moreno-Pérez, J. A. e Moreno-Veja, J. M. (1996), A chain-interchange heuristic method, *Yugoslav Journal of Operations Research,* 6, 41-54.

Palazzo, L. A. M., Algoritmos para Computação Evolutiva, *Relatório Técnico, Grupo de Pesquisa em Inteligência Artificial,* Universidade Católica de Pelotas, 1997.

Prais, M. e Ribeiro, C. C. (2000), Reactive GRASP: An Application to a Matrix Decomposition Problem in TDMA Traffic Assignment, *INFORMS Journal on Computing*, 12, 164-176.

Resende, M. G. C. e Ribeiro, C. C., GRASP with path-relinking: Recent advances and applications. Ibaraki, T., Nonobe, K., Yagiura, M. (eds). *Metaheuristics: Progress as Real Problem Solvers.* Springer, Berlin, 29-63, 2005.

Resende, M. G. C. e Ribeiro, C. C., Greedy randomized adaptive search procedures. Glover, F., Kochenberger, G. (eds). *Handbook of Metaheuristics*. Kluwer, Dordrecht, 219-249. 2003.

Resende, M. e Werneck, R. F., On the implementation of a swap-based local search procedure for the *p*-median problem. Ladner, R. E. (eds), *Proceedings of the 5th Workshop on Algorithm Engineering and Experiments*, SIAM, Philadelphia, 119-127, 2003.

Ribeiro, C. C. e Vianna, D. S., (2003), A genetic algorithm for the phylogeny problem using an optimized crossover strategy based on path-relinking, *Anais do II Workshop Brasileiro de Bioinformática. Editora Universo*, 97-102.

Ribeiro, C. C. e Vianna, D. S., (2009), A hybrid genetic algorithm for the phylogeny problem using path-relinking as a progressive crossover strategy, *International Transactions in Operational Research*, 16, 641-657.

Sobrinho, A. C. *Uma análise dos algoritmos genéticos e suas aplicações em sistemas de acesso à informação*, Universidade Federal do Maranhão, 2003.

Sobrinho, Antonio Carlos e Girardi, Rosario., (2003), Uma Análise das Aplicações dos Algoritmos Genéticos em Sistemas de Acesso à Informação Personalizada. *REIC. Revista Eletrônica de Iniciação Científica*, 3(4), 1.

Teitz, M. B. e Bart, P., (1968), Heuristic methods for estimating the generalized vertex median of a weighted graph. *Operations Research,* 16, 955-961.

Vianna, D. S. e Ribeiro, C. C., *Heurísticas híbridas para o problema da filogenia*, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 101, 2004.