4
GRASP com Filtro e Reconexão por Caminhos

A metaheurística GRASP *(Greedy Randomized Adaptive Search Procedures)* foi proposta por Feo e Resende [17], onde cada iteração do algoritmo é composta por duas fases: uma fase de construção, na qual uma solução viável é produzida utilizando-se uma heurística construtiva, e uma fase de melhoramento, utilizando-se um algoritmo de busca local, na qual um ótimo local é pesquisado na vizinhança da solução construída. A melhor solução encontrada é mantida como resultado. O pseudo-código da Figura 4.1 ilustra os blocos da metaheurística GRASP para um problema de minimização, na qual *MaxIter* iterações são realizadas, S é uma solução para o problema, $F(.)$ é o valor da função objetivo do problema e *semente* é usada na inicialização do gerador de números aleatórios [46].

```
Procedimento GRASP (MaxIter, semente)
1.  F* ← +∞;
2.  Para it = 1, ..., MaxIter Faça
3.      S ← Fase_Construção(semente);
4.      S' ← Busca_Local(S);
5.      Se (F(S') < F*) Então
6.          F* ← F(S');
7.          S* ← S';
8.      Fim-Se
9.  Fim-Para
10. Retorne S*;
Fim
```

Figura 4.1: Pseudo-código da metaheurística GRASP.

Na fase de construção, responsável pela denominação do método, uma solução viável é iterativamente construída, um elemento a cada vez, utilizando-se as características de um método guloso, a aleatoriedade e a adaptação da função gulosa. Em cada iteração, utiliza-se uma função gulosa característica do problema, que mede o benefício de selecionar cada elemento, para construir uma Lista de Candidatos Restrita (LCR) a partir dos elementos que podem ser adicionados à solução. A heurística é adaptativa porque os benefícios associados com cada elemento são atualizados em cada iteração da fase de construção para refletir as mudanças trazidas pela seleção do elemento anterior.

O componente probabilístico do GRASP é caracterizado pela escolha aleatória de um dos melhores candidatos da LCR, mas não necessariamente o melhor candidato, como acontece em um método puramente guloso. Uma das formas para construir a LCR é definí-la em função de um parâmetro $\alpha \in [0, 1]$, compreendendo todos os elementos que não estão na solução e cujo acréscimo no valor da função objetivo estejam no intervalo $[C_{min}, \alpha(C_{max} - C_{min}) + C_{min}]$, onde C_{min} e C_{max} correspondem, respectivamente, ao menor e ao maior acréscimos no valor da função objetivo dos elementos que ainda não fazem parte da solução. Em um problema de minimização, $\alpha = 0$ implica em uma escolha gulosa e $\alpha = 1$ implica em uma escolha aleatória.

A fase de melhoramento consiste tipicamente de um algoritmo de busca local, já que a solução gerada na fase de construção pode não ser um ótimo local. Assim, partindo-se de cada solução inicial, emprega-se a busca local na tentativa de melhorar a solução.

A metaheurística GRASP possui poucos parâmetros a serem ajustados: um está relacionado ao critério de parada, que é o número máximo de iterações realizadas, e o outro está relacionado a qualidade dos elementos na LCR, influenciada pela escolha do parâmetro α [46].

Os componentes básicos do GRASP para o problema das p-medianas conectadas são: uma heurística construtiva em função do parâmetro α e o algoritmo de busca local concatenada descrito no Capítulo 3 (BL_Conc).

Devido aos altos tempos de processamento apresentados pela busca local nas maiores instâncias, uma estratégia de filtro foi utilizada com o próposito de tentar acelerar as iterações do GRASP. Em geral, a utilização de filtros possui como objetivo evitar a aplicação da busca local a soluções não promissoras, onde provavelmente convergirá para um ótimo local de baixa qualidade.

Uma desvantagem do GRASP em sua versão básica é a independência de cada iteração em relação às demais. Assim, nenhuma informação reunida nas iterações anteriores é utilizada para guiar ou tentar melhorar as soluções encontradas pelas iterações posteriores do algoritmo. Com o objetivo de tentar melhorar a qualidade das soluções, o procedimento de reconexão por caminhos é utilizado como estratégia de intensificação explorando trajetórias entre um ótimo local e um elemento de um conjunto de soluções de alta qualidade, adicionando um mecanismo de memória ao GRASP.

Este capítulo está organizado como se segue: a Seção 4.1 mostra a fase de construção do GRASP; a Seção 4.2 mostra o ambiente de teste, as instâncias e as medidas utilizadas na configuração do GRASP; a Seção 4.3 apresenta o ajuste dos parâmetros da metaheurística; a Seção 4.4 descreve a estratégia de filtro utilizada e resultados computacionais em relação ao GRASP em sua versão básica; a Seção 4.5 apresenta o procedimento de reconexão por caminhos e comparações com o GRASP com filtro que não o utiliza; a Seção 4.6 descreve

a heurística GRASP com filtro e reconexão por caminhos proposta para o problema e, por último, a Seção 4.7 apresenta as conclusões e considerações finais do capítulo.

4.1 Fase de Construção

A Figura 4.2 descreve a fase de construção do GRASP, onde $C(f)$ representa o acréscimo obtido no valor da função objetivo relativo ao custo de atendimento dos usuários ao adicionar a facilidade $f \in V$ à solução S (critério de p-medianas) e $D(f)$ representa o acréscimo obtido no valor da função objetivo relativo ao custo de interconexão das facilidades ao adicionar o caminho mínimo entre a facilidade $f \in V$ e a árvore parcial construída pelo algoritmo executando-se a heurística Prim (critério de Steiner). A função recebe como parâmetros o valor de $\alpha \in [0, 1]$ e a *semente* para inicializar o gerador de números aleatórios.

Função Fase_Construção (α, *semente*)
1. $S \leftarrow \emptyset$;
2. Avalie o acréscimo $C(f), \forall\ f \in V$;
3. $D(f) \leftarrow 0, \forall\ f \in V$;
4. $k \leftarrow 0$;
5. **Enquanto** $(k \neq p)$ **Faça**
6. $\quad C_{min} \leftarrow \min\{C(f) + D(f), f \in V\}$;
7. $\quad C_{max} \leftarrow \max\{C(f) + D(f), f \in V\}$;
8. $\quad LCR \leftarrow \{f \in V \mid C(f) + D(f) \leq C_{min} + \alpha(C_{max} - C_{min})\}$;
9. $\quad$ Selecione f aleatoriamente da LCR;
10. $\quad S \leftarrow S \cup \{f\}$;
11. $\quad V \leftarrow V \setminus \{f\}$;
12. $\quad$ Re-avalie o acréscimo $C(f), \forall\ f \in V$;
13. $\quad k \leftarrow k + 1$;
14. $\quad$ Construa a árvore de Steiner parcial $T_k(S)$;
15. $\quad$ Avalie $D(f)$ em relação à $T_k(S), \forall\ f \in V$;
16. **Fim-Enquanto**
17. $T(S) \leftarrow T_p(S)$;
18. **Retorne** S;

Fim

Figura 4.2: Fase de construção do GRASP.

Na linha 2, avalia-se, para cada facilidade, o acréscimo obtido no valor da função objetivo relativo ao custo de atendimento dos usuários ao adicioná-la à solução (critério de p-medianas). Na linha 3, a distância mínima de cada vértice à arvore parcial é inicializada com zero (critério de Steiner), pois ainda nenhuma facilidade foi escolhida (o conjunto S está vazio). O laço das linhas 5 a 16 é executado iterativamente até que se obtenha p facilidades abertas e a árvore de Steiner conectando as mesmas. A LCR é construída em função

do parâmetro α (linhas 6 a 8), sendo que na primeira iteração, utiliza-se somente o critério de p-medianas para a construção da LCR, pois o conjunto S permanece vazio. Nas $p-1$ iterações restantes, utiliza-se a soma dos critérios de p-medianas e de Steiner para a sua construção. Na linha 9, ocorre a seleção aleatória de uma facilidade da LCR, o que mantém a diversidade das soluções geradas. Na linha 10, a facilidade escolhida é adicionada à solução e, na linha 11, o conjunto de possíveis localizações de facilidades é atualizado excluindo-se a facilidade escolhida anteriormente. Na linha 12, para cada facilidade, reavalia-se o acréscimo obtido no valor da função objetivo referente ao custo de atendimento dos usuários ao adicioná-la à solução. Na linha 14, a árvore de Steiner parcial com k facilidades é construída executando-se a heurística Prim, e na linha 15, avalia-se a menor distância do vértice f à árvore de Steiner parcial. Para as facilidades que são vértices de Steiner na árvore parcial, $D(f) = 0$. Ao final das iterações, o algoritmo conectará os p vértices terminais e possíveis não-terminais através de uma árvore de Steiner (linha 17). Por último, na linha 18, o algoritmo retorna uma solução definida pelo conjunto S formado pelas p facilidades abertas e pelo conjunto $T(S)$ de arestas obtidas quando se conecta estas facilidades por uma árvore.

A linha 1 possui complexidade $O(p)$. As linhas 3, 6, 7 e 8 possuem complexidade $O(n)$, enquanto as linhas 4, 9, 10, 11 e 13 possuem complexidade $O(1)$. Para cada facilidade, avaliar o acréscimo obtido no valor da função objetivo referente ao custo de atendimento ao adicioná-la à solução possui complexidade $O(n)$, correspondente ao número de usuários que serão atendidos pela facilidade. Como existem n possíveis localizações de facilidades candidatas, a complexidade das linhas 2 e 12 é, respectivamente, $O(n^2)$ e $O(pn^2)$. Para a construção da árvore de Steiner parcial (linha 14), no pior caso, p terminais devem ser conectados com complexidade $O(p\,|E|\log n)$, referente à heurística Prim. Para a avaliação do critério de Steiner (linha 15), no máximo p execuções de um algoritmo de caminho mínimo (algoritmo de Dijkstra [15]) devem ser realizadas com complexidade $O(|E|\log n)$, utilizando-se a estrutura de dados *heap* binário [61]. Assim, a linha 15 possui complexidade $O(p\,|E|\log n)$. A complexidade de pior caso do algoritmo é $O(pn^2)$, dominada pela linha 12.

4.2 Ambiente de Teste, Instâncias e Medidas Utilizadas

Para os testes da estratégia de filtro e configuração do procedimento de reconexão por caminhos, utilizou-se um sub-conjunto das instâncias proporcionais apresentadas na Seção 3.3.4, totalizando 36 problemas: ORM_P1 a ORM_P18 e GRM_P1 a GRM_P18 ($w = 5$). Esses testes foram executados em uma máquina Pentium IV 3.2 GHz com 1 Gbyte de memória RAM sob o sistema operacional Linux RedHat 9.0. Na avaliação da qualidade das soluções e

tempos de processamento, as medidas relativas, descritas no Capítulo 3, serão utilizadas. Para cada par instância-algoritmo, foram realizadas cinco execuções com diferentes sementes aleatórias.

Para o ajuste de α no GRASP, importante parâmetro responsável pela qualidade das soluções construídas, utilizou-se um sub-conjunto maior de instâncias proporcionais apresentadas na Seção 3.3.4, totalizando-se 132 problemas: GRM_P1 a GRM_P20 e ORM_P1 a ORM_P24 ($w = 2$, $w = 5$ e $w = 10$). Esse teste foi executado em uma máquina AMD Sempron 1.8 GHz com 512 Mbytes de memória RAM sob o sistema operacional Windows XP Service Pack 2. Para a avaliação da qualidade das soluções e tempos de processamento, considera-se, para cada instância, a média obtida pelo algoritmo (qualidade/tempo) em um determinado número de execuções. Para cada par instância-algoritmo, foram realizadas três execuções com diferentes sementes aleatórias.

O GRASP foi implementado em C e o gerador de números aleatórios utilizado foi o de Matsumoto e Nishimura [53].

4.3 Configuração dos parâmetros no GRASP

Os dois principais parâmetros do GRASP em sua versão básica são: o número máximo de iterações realizadas, representado pela variável *MaxIter* na Figura 4.1, e o valor de $\alpha \in [0, 1]$.

4.3.1 Número de Iterações

No GRASP, o tempo de execução não varia muito entre uma iteração e outra, sendo o tempo total gasto pelo algoritmo previsível e proporcional ao número de iterações realizadas. Aumentando-se o número de iterações, a tendência é de que melhore a qualidade das soluções encontradas pelo GRASP. Assim, quanto maior o número de iterações, maior o tempo gasto pelo algoritmo e melhor será a solução encontrada [46]. Porém, como apresentado no Capítulo 3, o tempo médio de processamento de cada iteração da busca local é elevado para as maiores instâncias testadas, tais como ORM_P25, ORM_P30, ORM_P34, ORM_P37, ORM_P40, SLM_P1, SLM_P2 e SLM_P3. Por isso, o número máximo de iterações realizadas pelo GRASP foi estipulado em 100 para os testes com o filtro e para a configuração do procedimento de reconexão por caminhos e em 200 para o ajuste do parâmetro α, valores relativamente baixos em comparação com outras aplicações GRASP apresentadas na literatura, como, por exemplo em [44].

4.3.2 Valor de α

Um dos aspectos que melhoram o desempenho da busca local é a geração de soluções iniciais de alta qualidade. Por esse motivo, a fase de construção torna-se muito importante no GRASP e, conseqüentemente o valor de α, pois, em geral, este parâmetro influencia fortemente a qualidade das soluções encontradas e os tempos de processamento obtidos pela busca local.

Para o ajuste do parâmetro α, as seguintes configurações foram analisadas: 0 (escolha gulosa), 0,2, 0,4, 0,6, 0,8 e 1 (escolha aleatória). Nas Tabelas 4.1 a 4.12, os valores em negrito representam o menor valor obtido pelas configurações para cada instância.

Analisando-se a qualidade média das soluções encontradas pelo GRASP (Tabelas 4.1 a 4.6), a configuração gulosa ($\alpha = 0$) obteve os piores resultados. Comparando-se as demais configurações entre si, os resultados apresentados foram muito semelhantes. Somando-se o número de instâncias em que os algoritmos encontraram o menor valor entre todos os métodos, incluindo-se os casos de empate no primeiro lugar para as classes testadas (linha *melhor* nas tabelas), as variantes GRASP com $\alpha = 0,0,2,0,4,0,6,0,8$ e 1 obtiveram os seguintes resultados, respectivamente: 65, 120, 118, 112, 117 e 114.

Analisando-se o tempo médio de execução encontrado pelo GRASP (Tabelas 4.7 a 4.12), em geral, como esperado, quanto mais próximo da escolha gulosa, menor o tempo de execução do GRASP devido à geração de soluções iniciais melhores e, conseqüentemente, executando um menor número de trocas durante a busca local. Somando-se o número de instâncias em que os algoritmos encontraram o menor valor entre todos os métodos, incluindo-se os casos de empate no primeiro lugar para as classes testadas (linha *melhor* nas tabelas), as variantes GRASP com $\alpha = 0,0,2,0,4,0,6,0,8$ e 1 obtiveram os seguintes resultados, respectivamente: 93, 22, 9, 1, 3 e 4.

Em termos de qualidade da solução, os resultados apresentados pelas configurações foram semelhantes, com exceção de $\alpha = 0$. Em termos de tempo de processamento, destacaram-se as configurações próximas da escolha gulosa ($\alpha = 0,0,2$ e $0,4$). Poderia utilizar-se tanto $\alpha = 0,2$ quanto $\alpha = 0,4$. Com o objetivo de obter uma maior diversidade na geração das soluções iniciais, optou-se por utilizar $\alpha = 0,4$.

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
ORM_P1	**6443,00**	**6443,00**	**6443,00**	**6443,00**	**6443,00**	**6443,00**
ORM_P2	**5228,00**	**5228,00**	**5228,00**	**5228,00**	**5228,00**	**5228,00**
ORM_P3	**5368,00**	**5368,00**	**5368,00**	**5368,00**	**5368,00**	**5368,00**
ORM_P4	**5127,00**	**5127,00**	**5127,00**	**5127,00**	**5127,00**	**5127,00**
ORM_P5	**3663,00**	**3663,00**	**3663,00**	**3663,00**	**3663,00**	**3663,00**
ORM_P6	**8185,00**	**8185,00**	**8185,00**	**8185,00**	**8185,00**	**8185,00**
ORM_P7	**6217,00**	**6217,00**	**6217,00**	**6217,00**	**6217,00**	**6217,00**
ORM_P8	**5865,00**	**5865,00**	5865,33	5865,33	5865,33	5865,33
ORM_P9	4703,67	4699,67	**4699,00**	**4699,00**	**4699,00**	**4699,00**
ORM_P10	**3457,00**	**3457,00**	**3457,00**	**3457,00**	**3457,00**	**3457,00**
ORM_P11	7879,00	**7855,00**	**7855,00**	**7855,00**	**7855,00**	**7855,00**
ORM_P12	**7074,00**	**7074,00**	**7074,00**	**7074,00**	**7074,00**	**7074,00**
ORM_P13	5547,00	**5539,00**	**5539,00**	**5539,00**	**5539,00**	**5539,00**
ORM_P14	4981,67	**4981,00**	**4981,00**	**4981,00**	**4981,00**	**4981,00**
ORM_P15	**4477,00**	**4477,00**	**4477,00**	**4477,00**	**4477,00**	**4477,00**
ORM_P16	**8292,00**	**8292,00**	**8292,00**	**8292,00**	**8292,00**	**8292,00**
ORM_P17	**7279,00**	**7279,00**	**7279,00**	**7279,00**	**7279,00**	**7279,00**
ORM_P18	5990,00	**5989,33**	5990,67	5990,33	5991,00	5989,67
ORM_P19	**4787,00**	4788,67	**4787,00**	4788,33	4788,00	4787,67
ORM_P20	4861,00	**4860,00**	**4860,00**	**4860,00**	**4860,00**	**4860,00**
ORM_P21	**9272,00**	**9272,00**	**9272,00**	**9272,00**	**9272,00**	**9272,00**
ORM_P22	**8839,00**	**8839,00**	**8839,00**	**8839,00**	**8839,00**	**8839,00**
ORM_P23	5802,33	5801,33	5801,00	5801,00	**5800,33**	5800,67
ORM_P24	**4985,00**	4985,33	4985,67	4985,33	4985,33	4985,67
melhor	17	20	20	19	20	19

Tabela 4.1: Configuração de α: qualidade média das instâncias ORM_P1 a ORM_P24 ($w = 2$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
ORM_P1	**7184,00**	**7184,00**	**7184,00**	**7184,00**	**7184,00**	**7184,00**
ORM_P2	6580,00	**6572,00**	**6572,00**	**6572,00**	**6572,00**	**6572,00**
ORM_P3	6793,00	**6776,00**	**6776,00**	**6776,00**	**6776,00**	**6776,00**
ORM_P4	6999,00	**6944,00**	**6944,00**	**6944,00**	**6944,00**	**6944,00**
ORM_P5	**5292,00**	**5292,00**	**5292,00**	**5292,00**	**5292,00**	**5292,00**
ORM_P6	8700,00	**8662,00**	**8662,00**	**8662,00**	**8662,00**	**8662,00**
ORM_P7	6990,00	6980,00	**6977,00**	6980,00	**6977,00**	**6977,00**
ORM_P8	7214,00	7206,33	**7203,00**	**7203,00**	**7203,00**	7206,33
ORM_P9	**6436,00**	**6436,00**	**6436,00**	**6436,00**	**6436,00**	**6436,00**
ORM_P10	**5004,00**	**5004,00**	**5004,00**	**5004,00**	**5004,00**	**5004,00**
ORM_P11	**8062,00**	**8062,00**	**8062,00**	**8062,00**	**8062,00**	**8062,00**
ORM_P12	**7666,00**	**7666,00**	**7666,00**	**7666,00**	**7666,00**	**7666,00**
ORM_P13	6696,00	**6693,00**	**6693,00**	**6693,00**	**6693,00**	**6693,00**
ORM_P14	6632,67	**6632,00**	**6632,00**	**6632,00**	**6632,00**	**6632,00**
ORM_P15	6364,00	**6363,00**	**6363,00**	**6363,00**	**6363,00**	**6363,00**
ORM_P16	**8458,00**	**8458,00**	**8458,00**	**8458,00**	**8458,00**	**8458,00**
ORM_P17	**7676,00**	**7676,00**	**7676,00**	**7676,00**	**7676,00**	**7676,00**
ORM_P18	**7259,00**	**7259,00**	**7259,00**	**7259,00**	**7259,00**	**7259,00**
ORM_P19	6540,00	**6537,00**	**6537,00**	**6537,00**	**6537,00**	**6537,00**
ORM_P20	7280,00	**7266,33**	7268,67	7270,67	7270,33	7268,00
ORM_P21	**9473,00**	**9473,00**	**9473,00**	**9473,00**	**9473,00**	**9473,00**
ORM_P22	**9219,00**	**9219,00**	**9219,00**	**9219,00**	**9219,00**	**9219,00**
ORM_P23	**7020,00**	**7020,00**	**7020,00**	**7020,00**	**7020,00**	**7020,00**
ORM_P24	6655,00	**6648,00**	6648,33	6648,67	6648,67	**6648,00**
melhor	12	22	22	21	22	22

Tabela 4.2: Configuração de α: qualidade média das instâncias ORM_P1 a ORM_P24 ($w = 5$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
ORM_P1	**8146,00**	**8146,00**	**8146,00**	**8146,00**	**8146,00**	**8146,00**
ORM_P2	**7706,00**	**7706,00**	**7706,00**	**7706,00**	**7706,00**	**7706,00**
ORM_P3	8296,00	**8265,00**	**8265,00**	**8265,00**	**8265,00**	**8265,00**
ORM_P4	9183,00	**9135,00**	**9135,00**	**9135,00**	**9135,00**	**9135,00**
ORM_P5	**7485,00**	**7485,00**	**7485,00**	**7485,00**	**7485,00**	**7485,00**
ORM_P6	**9387,00**	**9387,00**	**9387,00**	**9387,00**	**9387,00**	**9387,00**
ORM_P7	7882,00	**7878,00**	**7878,00**	7879,00	**7878,00**	7879,00
ORM_P8	8667,00	**8632,00**	**8632,00**	**8632,00**	**8632,00**	**8632,00**
ORM_P9	8214,00	**8174,00**	**8174,00**	**8174,00**	**8174,00**	**8174,00**
ORM_P10	6918,00	**6894,00**	**6894,00**	6894,33	**6894,00**	6894,67
ORM_P11	8405,00	**8383,00**	**8383,00**	**8383,00**	**8383,00**	**8383,00**
ORM_P12	**8426,00**	**8426,00**	**8426,00**	**8426,00**	**8426,00**	**8426,00**
ORM_P13	**7679,00**	**7679,00**	**7679,00**	**7679,00**	**7679,00**	**7679,00**
ORM_P14	8542,00	8505,67	8505,33	8505,67	**8505,00**	8505,33
ORM_P15	8981,00	**8865,67**	8871,00	8869,00	8869,00	8867,33
ORM_P16	8738,00	**8728,00**	**8728,00**	**8728,00**	**8728,00**	**8728,00**
ORM_P17	**8317,00**	**8317,00**	**8317,00**	**8317,00**	**8317,00**	**8317,00**
ORM_P18	8610,00	**8547,00**	**8547,00**	**8547,00**	**8547,00**	**8547,00**
ORM_P19	8371,00	**8320,33**	8329,33	8331,00	8330,67	8331,00
ORM_P20	10492,00	**10407,67**	10452,00	10456,00	10450,00	10468,33
ORM_P21	**9808,00**	**9808,00**	**9808,00**	**9808,00**	**9808,00**	**9808,00**
ORM_P22	**9776,00**	**9776,00**	**9776,00**	**9776,00**	**9776,00**	**9776,00**
ORM_P23	8361,00	**8359,00**	**8359,00**	**8359,00**	**8359,00**	**8359,00**
ORM_P24	8573,00	**8552,00**	8552,67	8555,33	8552,67	8556,33
melhor	9	23	19	17	20	17

Tabela 4.3: Configuração de α: qualidade média das instâncias ORM_P1 a ORM_P24 ($w = 10$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
GRM_P1	**6403,00**	**6403,00**	**6403,00**	**6403,00**	**6403,00**	**6403,00**
GRM_P2	5493,00	**5489,00**	**5489,00**	**5489,00**	**5489,00**	**5489,00**
GRM_P3	5207,00	**5157,00**	**5157,00**	**5157,00**	**5157,00**	**5157,00**
GRM_P4	5252,00	**5246,00**	**5246,00**	**5246,00**	**5246,00**	**5246,00**
GRM_P5	**5403,00**	**5403,00**	**5403,00**	**5403,00**	**5403,00**	**5403,00**
GRM_P6	**5580,00**	**5580,00**	**5580,00**	**5580,00**	**5580,00**	**5580,00**
GRM_P7	**5766,00**	**5766,00**	**5766,00**	**5766,00**	**5766,00**	**5766,00**
GRM_P8	**5961,00**	**5961,00**	**5961,00**	**5961,00**	**5961,00**	**5961,00**
GRM_P9	**6364,00**	**6364,00**	**6364,00**	**6364,00**	**6364,00**	**6364,00**
GRM_P10	**11475,00**	**11475,00**	**11475,00**	**11475,00**	**11475,00**	**11475,00**
GRM_P11	**9862,00**	**9862,00**	**9862,00**	**9862,00**	**9862,00**	**9862,00**
GRM_P12	8921,00	**8910,00**	**8910,00**	**8910,00**	**8910,00**	**8910,00**
GRM_P13	8455,00	**8352,00**	**8352,00**	**8352,00**	**8352,00**	**8352,00**
GRM_P14	8199,00	8180,00	**8174,00**	8180,67	8176,67	8180,00
GRM_P15	8174,00	**8133,00**	**8133,00**	**8133,00**	**8133,00**	**8133,00**
GRM_P16	8261,00	8244,00	8243,33	**8243,00**	8243,33	8243,67
GRM_P17	8388,00	**8372,00**	**8372,00**	**8372,00**	**8372,00**	**8372,00**
GRM_P18	8531,00	**8530,00**	**8530,00**	**8530,00**	**8530,00**	**8530,00**
GRM_P19	**8701,00**	**8701,00**	**8701,00**	**8701,00**	**8701,00**	**8701,00**
GRM_P20	**9080,00**	**9080,00**	**9080,00**	**9080,00**	**9080,00**	**9080,00**
melhor	10	18	19	19	18	18

Tabela 4.4: Configuração de α: qualidade média das instâncias GRM_P ($w = 2$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
GRM_P1	7157,00	**7036,00**	**7036,00**	**7036,00**	**7036,00**	**7036,00**
GRM_P2	6763,00	**6662,00**	**6662,00**	**6662,00**	**6662,00**	**6662,00**
GRM_P3	6973,00	**6923,00**	**6923,00**	**6923,00**	**6923,00**	**6923,00**
GRM_P4	7608,00	**7602,00**	**7602,00**	**7602,00**	**7602,00**	**7602,00**
GRM_P5	**8379,00**	**8379,00**	**8379,00**	**8379,00**	**8379,00**	**8379,00**
GRM_P6	**9178,00**	**9178,00**	**9178,00**	**9178,00**	**9178,00**	**9178,00**
GRM_P7	**9984,00**	**9984,00**	**9984,00**	**9984,00**	**9984,00**	**9984,00**
GRM_P8	**10802,00**	**10802,00**	**10802,00**	**10802,00**	**10802,00**	**10802,00**
GRM_P9	12448,00	**12447,00**	**12447,00**	**12447,00**	**12447,00**	**12447,00**
GRM_P10	12254,00	**12202,00**	**12202,00**	**12202,00**	**12202,00**	**12202,00**
GRM_P11	**11086,00**	**11086,00**	**11086,00**	**11086,00**	**11086,00**	**11086,00**
GRM_P12	10779,00	**10767,00**	**10767,00**	**10767,00**	**10767,00**	**10767,00**
GRM_P13	10906,00	**10902,00**	**10902,00**	**10902,00**	**10902,00**	**10902,00**
GRM_P14	11376,00	**11293,00**	**11293,00**	11294,33	11295,67	11295,67
GRM_P15	11890,00	11875,33	**11874,00**	11877,33	**11874,00**	**11874,00**
GRM_P16	12554,00	**12539,00**	**12539,00**	12540,33	12540,33	12541,00
GRM_P17	13284,00	**13276,00**	**13276,00**	13278,33	**13276,00**	**13276,00**
GRM_P18	**14036,00**	**14036,00**	**14036,00**	**14036,00**	**14036,00**	**14036,00**
GRM_P19	**14814,00**	**14814,00**	**14814,00**	**14814,00**	**14814,00**	**14814,00**
GRM_P20	**16422,00**	**16422,00**	**16422,00**	**16422,00**	**16422,00**	**16422,00**
melhor	8	19	20	16	18	18

Tabela 4.5: Configuração de α: qualidade média das instâncias GRM_P ($w = 5$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
GRM_P1	7939,00	**7884,00**	**7884,00**	**7884,00**	**7884,00**	**7884,00**
GRM_P2	8657,00	**8580,00**	**8580,00**	**8580,00**	**8580,00**	**8580,00**
GRM_P3	9897,00	**9793,00**	**9793,00**	**9793,00**	**9793,00**	**9793,00**
GRM_P4	11508,00	**11487,00**	**11487,00**	**11487,00**	**11487,00**	**11487,00**
GRM_P5	**13273,00**	**13273,00**	**13273,00**	**13273,00**	**13273,00**	**13273,00**
GRM_P6	**15079,00**	**15079,00**	**15079,00**	**15079,00**	**15079,00**	**15079,00**
GRM_P7	16917,00	**16909,00**	**16909,00**	**16909,00**	**16909,00**	**16909,00**
GRM_P8	**18754,00**	**18754,00**	**18754,00**	**18754,00**	**18754,00**	**18754,00**
GRM_P9	**22507,00**	**22507,00**	**22507,00**	**22507,00**	**22507,00**	**22507,00**
GRM_P10	**13083,00**	**13083,00**	**13083,00**	**13083,00**	**13083,00**	**13083,00**
GRM_P11	**13040,00**	**13040,00**	**13040,00**	**13040,00**	**13040,00**	**13040,00**
GRM_P12	13846,00	**13845,00**	**13845,00**	**13845,00**	**13845,00**	**13845,00**
GRM_P13	15044,00	14986,33	**14985,00**	**14985,00**	**14985,00**	**14985,00**
GRM_P14	16375,00	**16355,00**	**16355,00**	**16355,00**	**16355,00**	**16355,00**
GRM_P15	17925,00	**17919,00**	**17919,00**	**17919,00**	**17919,00**	**17919,00**
GRM_P16	19592,00	19579,67	19579,33	**19579,00**	19579,33	**19579,00**
GRM_P17	21346,00	**21317,00**	21318,00	**21317,00**	**21317,00**	**21317,00**
GRM_P18	**23095,00**	**23095,00**	**23095,00**	**23095,00**	**23095,00**	**23095,00**
GRM_P19	**24895,00**	**24895,00**	**24895,00**	**24895,00**	**24895,00**	**24895,00**
GRM_P20	**28566,00**	**28566,00**	**28566,00**	**28566,00**	**28566,00**	**28566,00**
melhor	9	18	18	20	19	20

Tabela 4.6: Configuração de α: qualidade média das instâncias GRM_P ($w = 10$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
ORM_P1	**0,47**	0,85	1,02	1,15	1,22	1,28
ORM_P2	**0,99**	1,54	1,81	1,97	2,13	2,27
ORM_P3	**1,44**	1,73	1,99	2,11	2,22	2,28
ORM_P4	11,17	**7,40**	7,43	7,57	7,73	7,87
ORM_P5	32,35	29,85	**27,26**	27,53	29,54	29,36
ORM_P6	**1,59**	2,72	3,33	3,76	4,14	4,36
ORM_P7	**2,45**	5,00	5,76	6,33	6,85	7,40
ORM_P8	**15,74**	19,45	20,20	21,38	22,51	22,53
ORM_P9	**77,72**	86,62	89,97	90,65	93,59	93,31
ORM_P10	300,30	258,98	**252,49**	264,38	262,20	264,47
ORM_P11	**2,08**	3,81	4,66	5,48	6,25	7,13
ORM_P12	**4,18**	8,04	9,32	10,46	11,43	12,05
ORM_P13	**30,93**	75,68	85,31	91,72	94,68	99,85
ORM_P14	**260,12**	366,94	370,41	375,94	380,09	379,68
ORM_P15	1.187,60	**1.154,76**	1.229,68	1.272,51	1.275,23	1.266,15
ORM_P16	**4,65**	7,19	8,86	10,44	12,04	13,23
ORM_P17	**7,17**	13,76	16,69	18,42	20,47	22,52
ORM_P18	**151,61**	197,63	210,90	224,09	227,61	239,71
ORM_P19	**1.066,82**	1.202,28	1.230,71	1.291,34	1.280,10	1.327,74
ORM_P20	4.771,15	4.717,76	**4.716,19**	4.725,71	4.739,15	4.813,62
ORM_P21	**5,44**	10,79	13,62	16,23	19,11	21,05
ORM_P22	**12,74**	24,02	27,87	31,09	34,46	37,48
ORM_P23	**308,70**	549,93	614,10	638,76	654,95	666,53
ORM_P24	**1.959,88**	3.763,09	4.086,61	4.143,00	4.170,58	4.319,72
melhor	19	2	3	0	0	0

Tabela 4.7: Configuração de α: tempo médio de execução das instâncias ORM_P1 a ORM_P24 ($w = 2$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
ORM_P1	**0,69**	1,38	1,55	1,64	1,70	1,73
ORM_P2	11,68	**5,67**	6,06	6,30	6,78	6,71
ORM_P3	7,03	**6,07**	6,30	6,35	6,53	6,33
ORM_P4	**8,61**	11,14	10,96	11,21	11,11	11,39
ORM_P5	**15,69**	16,90	18,80	18,90	19,85	19,77
ORM_P6	**2,43**	3,77	4,38	4,72	5,04	5,25
ORM_P7	**12,41**	13,92	14,49	15,64	16,03	16,14
ORM_P8	**29,25**	51,66	51,46	52,85	53,67	53,30
ORM_P9	310,39	210,42	**207,84**	219,20	222,88	211,33
ORM_P10	303,96	**227,21**	231,58	244,23	247,27	233,84
ORM_P11	**2,43**	4,73	5,81	6,59	7,48	8,03
ORM_P12	**8,87**	14,91	16,88	17,84	18,72	18,82
ORM_P13	**141,85**	238,09	250,31	247,06	266,17	254,06
ORM_P14	**587,91**	601,83	616,21	619,47	596,50	594,29
ORM_P15	1.931,14	1.185,02	1.161,25	1.174,35	**1.128,74**	1.151,38
ORM_P16	**5,82**	8,34	9,84	11,34	12,89	13,91
ORM_P17	**12,95**	19,15	21,97	24,31	26,31	27,86
ORM_P18	**583,69**	1.020,20	1.019,43	1.027,97	1.035,02	1.026,50
ORM_P19	3.527,33	2.916,49	**2.851,93**	2.855,31	2.948,18	2.883,65
ORM_P20	6.894,01	5.219,73	5.427,66	5.402,04	**5.194,20**	5.328,25
ORM_P21	**6,10**	14,44	17,48	20,12	22,63	24,85
ORM_P22	**20,75**	39,43	44,49	47,50	49,87	52,05
ORM_P23	**3.046,26**	3.904,04	3.913,11	3.916,11	3.934,68	3.830,63
ORM_P24	8.964,07	**7.369,77**	8.006,63	7.639,83	7.596,67	7.945,12
melhor	16	4	2	0	2	0

Tabela 4.8: Configuração de α: tempo médio de execução das instâncias ORM_P1 a ORM_P24 ($w = 5$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
ORM_P1	**1,43**	2,23	2,29	2,35	2,37	2,32
ORM_P2	**3,63**	3,95	4,26	4,25	4,54	4,48
ORM_P3	9,88	**3,85**	4,18	4,28	4,32	4,35
ORM_P4	**8,72**	9,88	10,68	10,51	11,57	11,32
ORM_P5	53,64	23,95	**17,94**	19,20	19,20	18,67
ORM_P6	**3,82**	7,86	8,65	8,73	8,81	9,07
ORM_P7	**12,15**	19,04	20,06	19,71	20,71	21,34
ORM_P8	**40,19**	48,51	49,63	52,17	54,89	54,01
ORM_P9	**93,83**	117,68	125,71	123,41	124,82	126,82
ORM_P10	301,59	214,69	216,49	226,75	216,86	**209,65**
ORM_P11	**3,61**	7,29	8,30	9,30	10,10	10,59
ORM_P12	**23,00**	29,22	30,87	32,85	32,67	34,02
ORM_P13	301,08	**190,95**	200,51	201,85	204,33	209,56
ORM_P14	**678,01**	926,16	899,62	902,13	887,14	862,08
ORM_P15	1.225,15	1.243,00	1.192,30	1.233,70	1.177,55	**1.152,95**
ORM_P16	**4,51**	9,56	11,14	13,13	14,34	15,25
ORM_P17	**27,21**	40,52	44,48	46,88	47,93	49,97
ORM_P18	1.197,37	**931,95**	965,06	1.004,01	988,67	973,76
ORM_P19	**1.917,93**	3.234,97	3.470,85	2.960,18	3.007,70	2.979,12
ORM_P20	**4.909,04**	5.944,87	5.733,55	5.940,10	5.795,23	5.877,47
ORM_P21	**8,56**	20,41	23,75	26,58	28,82	30,07
ORM_P22	**48,05**	93,60	98,03	102,05	102,81	103,92
ORM_P23	3.844,91	**3.429,50**	3.738,92	3.646,77	3.684,44	3.666,73
ORM_P24	8.948,75	8.835,02	8.914,13	8.685,05	9.153,27	**8.526,51**
melhor	16	4	1	0	0	3

Tabela 4.9: Configuração de α: tempo médio de execução das instâncias ORM_P1 a ORM_P24 ($w = 10$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
GRM_P1	**4,78**	5,74	7,07	7,86	8,27	8,41
GRM_P2	20,59	**17,01**	20,07	21,14	22,50	21,61
GRM_P3	**40,97**	62,70	61,70	64,83	65,02	66,32
GRM_P4	**181,18**	191,69	192,49	203,10	204,20	212,04
GRM_P5	**217,59**	384,33	366,42	380,65	385,04	365,03
GRM_P6	**338,92**	433,34	419,74	447,31	462,14	451,88
GRM_P7	725,05	573,93	**538,45**	559,63	591,74	580,43
GRM_P8	545,38	542,01	**474,57**	524,64	508,27	499,67
GRM_P9	**293,67**	372,09	348,92	359,53	363,49	363,68
GRM_P10	20,59	**17,82**	21,75	21,82	21,80	22,08
GRM_P11	109,00	**62,84**	68,89	73,76	79,47	81,77
GRM_P12	**50,79**	97,66	109,15	112,65	121,40	119,68
GRM_P13	**101,46**	188,40	206,51	216,36	213,86	233,37
GRM_P14	**353,74**	382,72	431,03	420,51	417,32	428,22
GRM_P15	**632,82**	845,09	849,34	874,79	870,21	870,24
GRM_P16	1.915,71	**1.403,08**	1.535,41	1.565,98	1.569,12	1.574,71
GRM_P17	**1.503,46**	1.814,67	1.929,64	2.018,26	2.020,01	1.970,98
GRM_P18	3.648,27	**2.109,44**	2.119,83	2.217,90	2.263,40	2.221,14
GRM_P19	2.411,73	**2.398,63**	2.404,29	2.529,05	2.577,90	2.561,27
GRM_P20	**2.343,01**	2.504,94	2.565,46	2.690,63	2.745,63	2.693,28
melhor	12	6	2	0	0	0

Tabela 4.10: Configuração de α: tempo médio de execução das instâncias GRM_P ($w = 2$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
GRM_P1	**10,10**	13,71	14,66	15,11	15,08	15,10
GRM_P2	**25,65**	43,65	51,49	55,63	55,58	54,26
GRM_P3	**58,48**	116,48	139,23	142,33	146,04	141,49
GRM_P4	399,75	210,04	221,94	218,89	215,75	**209,72**
GRM_P5	**148,79**	165,66	177,90	184,47	176,02	176,74
GRM_P6	251,47	**176,85**	187,62	186,77	189,25	189,23
GRM_P7	**183,13**	186,71	195,32	198,94	196,85	200,40
GRM_P8	**193,70**	210,20	220,34	222,30	221,75	221,10
GRM_P9	**231,21**	269,20	275,40	279,50	281,94	284,22
GRM_P10	37,68	**32,90**	35,06	34,47	34,40	34,71
GRM_P11	**71,21**	117,29	135,64	140,20	151,00	149,20
GRM_P12	397,29	**300,00**	350,85	348,25	348,83	371,38
GRM_P13	810,00	**591,38**	684,21	652,92	703,30	698,08
GRM_P14	**777,01**	871,67	970,51	933,88	980,46	953,04
GRM_P15	**1.275,39**	1.324,43	1.391,48	1.433,40	1.473,57	1.435,10
GRM_P16	2.317,94	1.643,05	1.711,12	**1.634,44**	1.699,72	1.739,01
GRM_P17	**1.695,34**	1.696,43	1.800,04	1.850,41	1.796,16	1.863,34
GRM_P18	**1.553,37**	1.850,06	2.021,51	2.041,42	2.066,32	2.083,91
GRM_P19	3.665,57	3.005,65	**2.972,64**	3.077,71	3.155,80	3.127,15
GRM_P20	**2.371,96**	2.485,78	2561,56	2.695,46	2.649,14	2.583,37
melhor	13	4	1	1	0	1

Tabela 4.11: Configuração de α: tempo médio de execução das instâncias GRM_P ($w = 5$).

	$\alpha = 0$	$\alpha = 0,2$	$\alpha = 0,4$	$\alpha = 0,6$	$\alpha = 0,8$	$\alpha = 1$
Instância	Média	Média	Média	Média	Média	Média
GRM_P1	**14,32**	18,31	20,65	20,82	21,12	21,23
GRM_P2	**29,42**	51,70	66,65	63,40	66,12	67,00
GRM_P3	137,07	121,14	121,83	122,01	**120,97**	122,22
GRM_P4	**95,30**	119,17	123,37	126,02	120,97	122,28
GRM_P5	193,72	**132,15**	137,44	138,03	137,63	140,06
GRM_P6	**145,84**	160,71	164,69	167,00	166,85	169,30
GRM_P7	**175,04**	183,36	192,06	192,13	193,05	194,99
GRM_P8	**202,98**	212,55	227,24	228,47	230,63	228,33
GRM_P9	**232,90**	244,98	280,91	282,78	286,53	286,02
GRM_P10	**35,39**	56,84	54,25	51,63	49,61	49,71
GRM_P11	**141,62**	226,43	249,51	267,15	259,66	255,71
GRM_P12	**236,38**	405,95	473,33	486,91	473,89	461,54
GRM_P13	**324,54**	612,21	652,51	668,74	669,89	701,32
GRM_P14	**536,78**	927,72	994,16	1.043,23	1.001,46	963,52
GRM_P15	1.064,86	**1.059,59**	1.130,43	1.159,62	1.151,00	1.130,40
GRM_P16	**946,27**	1.298,26	1.457,37	1.336,11	1.371,48	1.437,10
GRM_P17	**1.253,96**	1.533,02	1.592,81	1.651,05	1.607,05	1.570,54
GRM_P18	**1.461,66**	1.898,23	1.926,82	1.987,33	1.910,28	1.902,66
GRM_P19	**1.738,16**	1.865,76	1.883,49	1.959,56	1.813,52	1.830,02
GRM_P20	**1.373,44**	1.926,96	1.928,98	1.922,69	1.915,16	1.926,77
melhor	17	2	0	0	1	0

Tabela 4.12: Configuração de α: tempo médio de execução das instâncias GRM_P ($w = 10$).

4.4 Estratégia de Filtro

Em virtude dos altos tempos de processamento apresentados pela busca local para as maiores instâncias, uma estratégia de filtro foi utilizada na tentativa de acelerar as iterações do GRASP básico. O filtro é aplicado entre a busca local mais rápida (busca local pelas bordas com teste da menor aresta) e a busca local mais lenta (busca local com teste). Assim, dependendo da solução e da qualidade do ótimo local da primeira busca e da melhor solução encontrada até o momento, executa-se ou não a busca local mais lenta, limitando-se a execução da busca local mais cara somente a ótimos locais da primeira busca suficientemente bons [35].

Seja a solução formada pelo conjunto $S_1^{'}$ de p facilidades abertas e por sua respectiva árvore de Steiner obtida após a execução da busca local mais rápida e $F(S_1^{'})$ o valor de sua função objetivo. Seja também $\lambda > 0$ um parâmetro de corte. Aplica-se a segunda busca local (mais lenta) caso a solução $S_1^{'}$ não tenha sido visitada previamente e $F(S_1^{'}) < (1 + \lambda)\ F^*$, onde F^* é o valor da função objetivo da melhor solução encontrada até o momento. Assim, executa-se a busca local mais lenta quando o ótimo local da busca local mais rápida não foi visitado anteriormente e está suficientemente próximo do valor da função objetivo da melhor solução encontrada até o momento. Como em [35], utilizou-se $\lambda = 1\%$.

Em geral, a utilização do filtro reduz o tempo de processamento às custas de perdas na qualidade média das soluções encontradas. Para verificar quanto o filtro afeta o tempo de processamento e a qualidade da solução, testes foram realizados comparando-se o GRASP sem filtro (GRASPb) com o GRASP com filtro (GRASPf). A Tabela 4.13 apresenta as medidas relativas *drpm* e *cm* utilizadas na análise da qualidade das soluções encontradas pelo GRASPb e GRASPf.

Analisando-se essa tabela, como esperado, o GRASPf perde em qualidade das soluções comparando-se com o GRASPb. Em termos absolutos, dos 180 testes realizados (36 instâncias com cinco execuções cada), o GRASPf manteve o mesmo valor em 170 testes.

	GRASPb	GRASPf
drpm	**-0,0035**	0,0035
cm	**1,43**	1,57

Tabela 4.13: Qualidade relativa na configuração do filtro no GRASP.

A Tabela 4.14 apresenta as medidas relativas *drpm*, *cm* e *trm* utilizadas na análise dos tempos de processamento obtidos pelo GRASPb e GRASPf. Nessa última medida, o GRASPb foi utilizado como algoritmo de referência.

	GRASPb	GRASPf
drpm	37,25	**-37,25**
cm	1,00	**2,00**
trm	1,00	**0,50**

Tabela 4.14: Tempo relativo na configuração do filtro no GRASP.

Na medida *cm*, observa-se que, em todos os testes ocorreram uma diminuição nos tempos de processamento. Na medida *trm*, o GRASPf gastou, em média, 50% do tempo de execução do GRASPb nas instâncias testadas. Em geral, as três medidas relativas mostraram que houve ganhos em tempos de processamento do GRASPf em relação ao GRASPb, com perdas na qualidade média das soluções encontradas.

4.5 Reconexão por Caminhos

O procedimento de reconexão por caminhos, proposto por Glover [19], foi primeiramente utilizado em busca tabu e *scatter search* como estratégia de intensificação explorando trajetórias entre uma solução corrente e soluções de elite. Iniciando-se de uma ou mais soluções, caminhos ou trajetórias que conduzem a soluções de elite são gerados e explorados em busca de melhorias. Um caminho é construído selecionando movimentos que introduzem atributos na solução atual que estão presentes nas soluções de elite. Reconexão por caminhos pode ser visto como uma estratégia que busca atributos presentes em soluções de alta qualidade para serem incorporados a uma outra solução.

O procedimento de reconexão por caminhos foi primeiramente utilizado no contexto do GRASP por Laguna e Martí [33] como estratégia de intensificação aplicada a cada ótimo local. Em geral, duas estratégias básicas têm sido utilizadas [46, 49]:

- aplicá-lo em todos os pares de soluções de elite periodicamente durante as iterações do GRASP ou como um passo de pós-otimização após todas as iterações terem sido realizadas;
- aplicá-lo como estratégia de intensificação após cada ótimo local produzido pela fase de busca local.

A utilização de reconexão por caminhos como estratégia de intensificação tem sido mais efetiva do que simplesmente utilizá-lo como passo de pós-otimização. Em geral, o melhor é combinar intensificação com pós-otimização [46, 49], porém, optou-se por utilizá-lo apenas como estratégia de intensificação.

Sejam duas soluções quaisquer formadas pelos conjuntos S_1 e S_2 compostos por p facilidades abertas e por suas respectivas árvores de Steiner. O

procedimento como estratégia de intensificação atua em um par de soluções (S_1, S_2), onde S_1 é um ótimo local produzido pela busca local em cada iteração do GRASP e S_2 é uma solução de elite escolhida de um conjunto P composto por *MaxElite* soluções de alta qualidade encontradas durante as iterações do GRASP. Inicia-se a trajetória da melhor solução entre S_1 e S_2, pois permite investigar com maior detalhe a vizinhança da melhor solução. Quando investigando em um único sentido, essa estratégia tem apresentado bons resultados na literatura [46, 49].

A Figura 4.3 ilustra o pseudo-código do algoritmo que recebe como parâmetros o par de soluções S_1 (solução inicial) e S_2 (solução alvo ou guia), além da primeira e segunda facilidade mais próxima de cada usuário u, ϕ_1^u e ϕ_2^u, respectivamente.

O algoritmo inicia na linha 2 calculando a diferença simétrica $\Delta(S_1, S_2)$ entre as soluções, o que corresponde ao conjunto de facilidades diferentes encontradas em S_1 e S_2. O valor $|\Delta(S_1, S_2)|$ corresponde a quantidade de trocas de facilidades necessárias para alcançar a solução alvo S_2 partindo-se da solução inicial S_1. A linha 5 do algoritmo consiste em construir, para cada usuário u, uma lista com todas as facilidades ordenadas pela distância ao usuário. Na linha 6, o conjunto de usuários afetados V_a é inicializado com todos os usuários. Na linha 7, as estruturas de dados auxiliares *ganho*, *perda* e *extra* são inicializadas. A idéia do procedimento é gerar um caminho de soluções entre S_1 e S_2 e caso o algoritmo obtenha sucesso, retorna-se a melhor solução encontrada no caminho; caso contrário, retorna-se a melhor solução entre S_1 e S_2 (linha 31). O procedimento termina quando a solução alvo é alcançada, isto é, quando $\Delta(S, S_2) = \emptyset$. Enquanto essa condição não for satisfeita (laço das linhas 8 a 30), em cada iteração, ocorre a atualização eficiente das estruturas de dados auxiliares (linhas 9 a 11). Em seguida, examina-se todas as trocas de facilidades possíveis de serem realizadas na solução corrente S selecionando aquela que maximiza a soma dos lucros obtidos no custo de atendimento dos usuários e no custo de interconexão das facilidades (linha 12). Essa função será explicada com maior detalhe posteriormente. A melhor troca é realizada atualizando-se a solução S (linhas 13 e 14) e as facilidades ainda disponíveis no conjunto da diferença simétrica (linha 15). Quando necessário, a melhor solução S^* é também atualizada (linhas 16 a 19). Nas linhas 20 a 25 ocorre a atualização do conjunto de usuários afetados. As informações presentes nas estruturas auxiliares encontram-se desatualizadas em relação à troca ocorrida e são corrigidas executando-se a função *Desfaz_Atualiza_Estruturas* somente para os usuários afetados (linhas 26 a 28). Por último, na linha 29, ocorre a atualização da primeira e da segunda facilidade mais próxima de cada usuário afetado, levando-se em consideração a troca efetuada.

A Figura 4.4 apresenta a função responsável pela escolha da troca mais

Procedimento `Reconexão_Caminhos`$(S_1, S_2, \phi_1, \phi_2)$
1. $S \leftarrow S_1$;
2. Calcular a diferença simétrica $\Delta(S, S_2)$;
3. $F^* \leftarrow \min\{F(S), F(S_2)\}$;
4. $S^* \leftarrow \text{argmin}\{F(S), F(S_2)\}$;
5. Para cada $u \in V$, ordenar as facilidades de acordo com a distância ao usuário;
6. $V_a \leftarrow V$;
7. $ganho(i) \leftarrow 0, perda(r) \leftarrow 0, extra(i, r) \leftarrow 0, \forall\ i \notin S, \forall\ r \in S$;
8. **Enquanto** $(\Delta(S, S_2) \neq \emptyset)$ **Faça**
9. **Para Todo** $(u \in V_a)$ **Faça**
10. *Atualiza_Estruturas(ganho, perda, extra, u, ϕ_1^u, ϕ_2^u)*;
11. **Fim-Para-Todo**
12. $(i, r, melhor) \leftarrow$ *Melhor_Troca(S, S_2, ganho, perda, extra)*;
13. $S \leftarrow S \setminus \{r\}$;
14. $S \leftarrow S \cup \{i\}$;
15. $\Delta(S, S_2) \leftarrow \Delta(S, S_2) \setminus \{r\}$;
16. **Se** $(F(S) < F^*)$ **Então**
17. $F^* \leftarrow F(S)$;
18. $S^* \leftarrow S$;
19. **Fim-Se**
20. $V_a \leftarrow \emptyset$;
21. **Para Todo** $(u \in V)$ **Faça**
22. **Se** $(\phi_1^u = r$ **ou** $\phi_2^u = r$ **ou** $d_{ui} < d_2^u)$ **Então**
23. $V_a \leftarrow V_a \cup \{u\}$;
24. **Fim-Se**
25. **Fim-Para-Todo**
26. **Para Todo** $(u \in V_a)$ **Faça**
27. *Desfaz_Atualiza_Estruturas(ganho, perda, extra, u, ϕ_1^u, ϕ_2^u)*;
28. **Fim-Para-Todo**
29. Atualizar a primeira e segunda facilidade mais próxima de cada usuário $u \in V_a$;
30. **Fim-Enquanto**
31. **Retorne** S^*;

Fim

Figura 4.3: Pseudo-código do procedimento de reconexão por caminhos.

lucrativa (gulosa) realizada em cada passo do procedimento. A função recebe como parâmetros a solução corrente composta pelo conjunto S de p facilidades abertas e pela árvore de Steiner $T(S)$ correspondente, a solução alvo composta pelo conjunto S_2 de p facilidades abertas, além das estruturas *ganho*, *perda* e *extra*.

```
Função Melhor_Troca(S, S2, ganho, perda, extra)
1.  melhor ← −∞;
2.  Para Todo (i1 ∉ S2 \ S) Faça
3.      Para Todo (r1 ∈ S \ S2) Faça
4.          lucroP(i1, r1) ← ganho(i1) − perda(r1) + extra(i1, r1);
5.          lucroS(i1, r1) ← FT(S) − FT(S');
6.          lucro(i1, r1) ← lucroP(i1, r1) + lucroS(i1, r1);
7.          Se (lucro(i1, r1) > melhor) Então
8.              melhor ← lucro(i1, r1);
9.              i ← i1; r ← r1;
10.         Fim-Se
11.     Fim-Para-Todo
12. Fim-Para-Todo
13. Retorne i, r, melhor;
Fim
```

Figura 4.4: Função que encontra a melhor troca no algoritmo de reconexão por caminhos.

A função é composta por dois laços aninhados. O externo (linhas 2 a 12) percorre as facilidades pertencentes ao conjunto $S_2 \setminus S$ e o interno (linhas 3 a 11) percorre as facilidades pertencentes ao conjunto $S \setminus S_2$. Ambos são executados até que se obtenha a melhor troca possível. Os lucros obtidos no custo de atendimento dos usuários e no custo de inteconexão das facilidades ao se trocar r_i por i_i são calculados nas linhas 4 e 5, respectivamente. O lucro total é calculado na linha 6 e a melhor troca é armazenada nas linhas 7 a 10. Na linha 13, retorna-se as facilidades i e r da melhor troca junto com seu lucro.

Dois aspectos importantes do procedimento são: gerenciamento do conjunto P e seleção de uma solução de elite presente em P em cada iteração.

Observações empíricas mostraram que, quanto maior o caminho percorrido entre as soluções, maior a probabilidade que uma solução melhor seja encontrada [48]. Assim, deve-se evitar aplicar reconexão por caminhos em soluções similares, sendo importante priorizar tanto a qualidade quanto a diversidade das soluções presentes em P. No início do algoritmo, P está vazio. Cada ótimo local é considerado candidato a ser inserido em P se é diferente de cada solução atualmente presente no conjunto. Quando *MaxElite* soluções estão presentes (P está cheio), o ótimo local candidato substitui o pior elemento de P se um dos critérios for satisfeito:

– o custo do ótimo local candidato é melhor do que o custo da melhor solução de elite presente em P, priorizando, assim, a qualidade;

– o custo do ótimo local candidato é melhor do que o custo da pior solução de elite presente em P e é diferente de todas as soluções do conjunto, priorizando, assim, tanto a qualidade quanto a diversidade.

Se o algoritmo encontra uma solução melhor do que a melhor solução encontrada até o momento, esta torna-se também um candidato a inserção em P de acordo com as condições mencionadas acima.

A partir da segunda iteração do GRASP, ocorre a seleção de uma solução de elite presente em P. Uma maneira simples seria selecioná-la aleatoriamente. Porém, pode-se selecionar uma solução semelhante ao ótimo local candidato, diminuindo as chances de encontrar uma boa solução no caminho entre S_1 e S_2. A diferença simétrica representa o tamanho da trajetória explorada entre as soluções. Assim, a seleção de um elemento de P no algoritmo favorece soluções de elite com uma maior diferença simétrica em relação ao ótimo local candidato. Isto é, quanto maior a diferença simétrica entre o ótimo local e a solução de elite, maior a chance de seleção do elemento de P. Esses dois métodos foram comparados em [48] e a seleção com probabilidade proporcional à diferença simétrica apresentou resultados melhores do que a seleção aleatória.

Reconexão por caminhos pode ser visto como uma busca local aplicada à solução S_1, com três principais diferenças:

1. na reconexão por caminhos, o número de movimentos permitidos é muito menor do que na busca local; somente um sub-conjunto das facilidades são candidatos a inserção $(S_2 \setminus S_1)$ e a remoção $(S_1 \setminus S_2)$. À medida que o algoritmo progride, esses subconjuntos tornam-se menores;

2. a busca local termina quando encontra um mínimo local, não aceitando movimentos que piorem a solução corrente. A reconexão por caminhos termina quando encontra a solução alvo, podendo aceitar movimentos que piorem a solução corrente;

3. a busca local utiliza melhoria iterativa selecionando a primeira solução aprimorante na vizinhança da solução corrente e a reconexão por caminhos utiliza descida mais rápida selecionando a melhor solução na vizinhança restrita da solução corrente.

4.5.1 Configuração do Tamanho do Conjunto de Soluções de Elite

Um dos parâmetros a serem ajustados no procedimento de reconexão por caminhos é o tamanho máximo do conjunto de soluções de elite (*MaxElite*). As seguintes configurações foram avaliadas: GRASP com filtro sem reconexão por caminhos (GRASPf) e GRASP com filtro e reconexão por caminhos com $MaxElite = 5, 10, 15$ e 20 (GRASPf_RC5, GRASPf_RC10, GRASPf_RC15 e GRASPf_RC20, respectivamente).

As medidas *drpm* e *cm* serão utilizadas para analisar a qualidade das soluções encontradas, como mostra a Tabela 4.15.

	drpm	*cm*
GRASPf	0,002038	3,18
GRASPf_RC5	-0,000312	3,01
GRASPf_RC10	**-0,001341**	**2,83**
GRASPf_RC15	-0,000578	2,96
GRASPf_RC20	0,000193	3,01

Tabela 4.15: Qualidade relativa na configuração de *MaxElite* no GRASP.

Analisando-se a Tabela 4.15, em ambas as medidas, o melhor resultado foi obtido pelo GRASPf_RC10. A adição do procedimento de reconexão por caminhos não pode piorar a qualidade da solução encontrada pelo GRASPf. Assim, dos 180 testes realizados (36 instâncias com cinco execuções cada), GRASPf_RC5, GRASPf_RC10, GRASPf_RC15 e GRASPf_RC20 melhoraram a qualidade da solução encontrada pelo GRASPf em 5, 7, 6 e 5 testes, respectivamente.

Para a análise dos tempos de processamento, os valores obtidos pelos algoritmos nas medidas *drpm*, *cm* e *trm* são mostrados na Tabela 4.16. Nessa última medida, utilizou-se o algoritmo GRASPf como referência.

	drpm	*cm*	*trm*
GRASPf	**-8,75**	**1,51**	**1,00**
GRASPf_RC5	2,04	3,33	1,13
GRASPf_RC10	1,93	3,00	1,13
GRASPf_RC15	2,19	3,46	1,13
GRASPf_RC20	2,59	3,69	1,14

Tabela 4.16: Tempo relativo na configuração de *MaxElite* no GRASP.

Analisando-se a Tabela 4.16, como esperado, os menores tempos de processamento foram obtidos pelo GRASPf, pois não existe a necessidade de controle e gerenciamento do conjunto de soluções de elite. Os algoritmos com reconexão por caminhos não apresentaram uma diferença significativa entre eles, porém, em geral, quanto maior o número de soluções a serem gerenciadas no conjunto P, maior o tempo de processamento do algoritmo. Na medida

trm, o tempo de processamento mais elevado foi obtido pelo GRASPf_RC20, que excedeu 14%, em média, o tempo de execução do GRASPf nas instâncias testadas.

Levando-se em consideração a qualidade das soluções e os tempos de processamento em relação ao GRASPf, os melhores resultados foram obtidos pelo GRASPf_RC10. Assim, o tamanho do conjunto de soluções de elite foi configurado em dez.

4.5.2 Comparação entre o GRASP com Filtro e o GRASP com Filtro e Reconexão por Caminhos

Uma hipótese a ser analisada é descobrir se o tempo extra gasto na execução do algoritmo de reconexão por caminhos, se utilizado pelo GRASP com filtro para iterações adicionais produziria resultados melhores. Com o objetivo de testá-la, comparações serão realizadas entre o GRASP com filtro sem reconexão por caminhos (GRASPf) e o GRASP com filtro com reconexão por caminhos e $MaxElite = 10$ (GRASPf_RC). Para isso, serão utilizados gráficos que comparam experimentalmente diferentes algoritmos aleatórios ou diferentes versões do mesmo algoritmo aleatório [1]. Os gráficos mostram a distribuição de probabilidade empírica da variável aleatória tempo gasto para encontrar um valor alvo. Para cada algoritmo, fixa-se o valor alvo e faz-se 200 execuções independentes. O algoritmo termina quando uma solução de valor menor ou igual ao valor alvo é encontrada. Associa-se ao i-ésimo menor tempo de execução t_i à probabilidade $prob_i = (i - 1/2)/200$, para $i = 1, 2, \ldots, 200$. As seguintes instâncias (com seus respectivos valores alvo entre parênteses) foram utilizadas para comparação: GRM_P11 (11086), GRM_P17 (13276), ORM_P9 (6436) e ORM_P15 (6363) para $w = 5$.

As Figuras 4.5 e 4.6 mostram que as curvas do GRASPf e do GRASPf_RC são próximas para as instâncias GRM_P11 e ORM_P9, respectivamente. Os ganhos obtidos nas instâncias menores com a adição do procedimento são pequenos ou quase nulos nas instâncias consideradas.

Quando aumenta-se o tamanho da instância, os ganhos com o procedimento de reconexão por caminhos tornam-se significativos, como mostram as Figuras 4.7 e 4.8.

Os gráficos ilustram que a probabilidade de encontrar uma solução com valor pelo menos tão bom quanto o valor alvo aumenta do GRASPf para o GRASPf_RC em ambas as instâncias. Observando-se a Figura 4.7, a probabilidade de encontrar o valor alvo em menos do que 140 segundos é de 100% para o GRASPf_RC e aproximadamente 73% para o GRASPf. A figura mostra claramente que o GRASPf_RC é mais rápido do que o GRASPf para encontrar o valor alvo 13276. Já na Figura 4.8, o ganho não é tão significativo

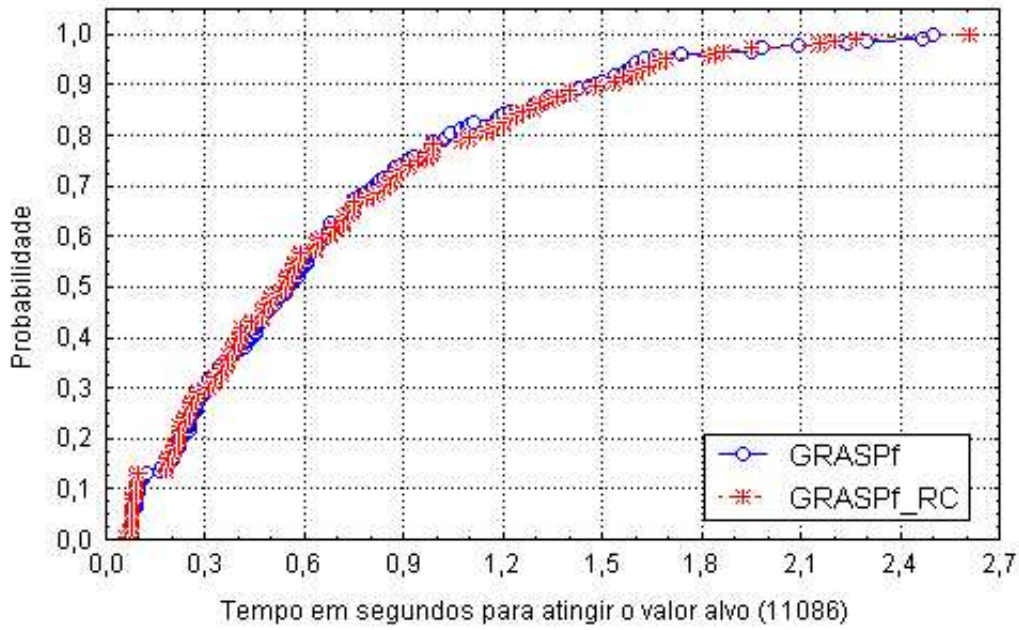

Figura 4.5: Distribuição de probabilidade empírica do tempo gasto para encontrar o valor alvo 11086 para a instância GRM_P11 ($w = 5$).

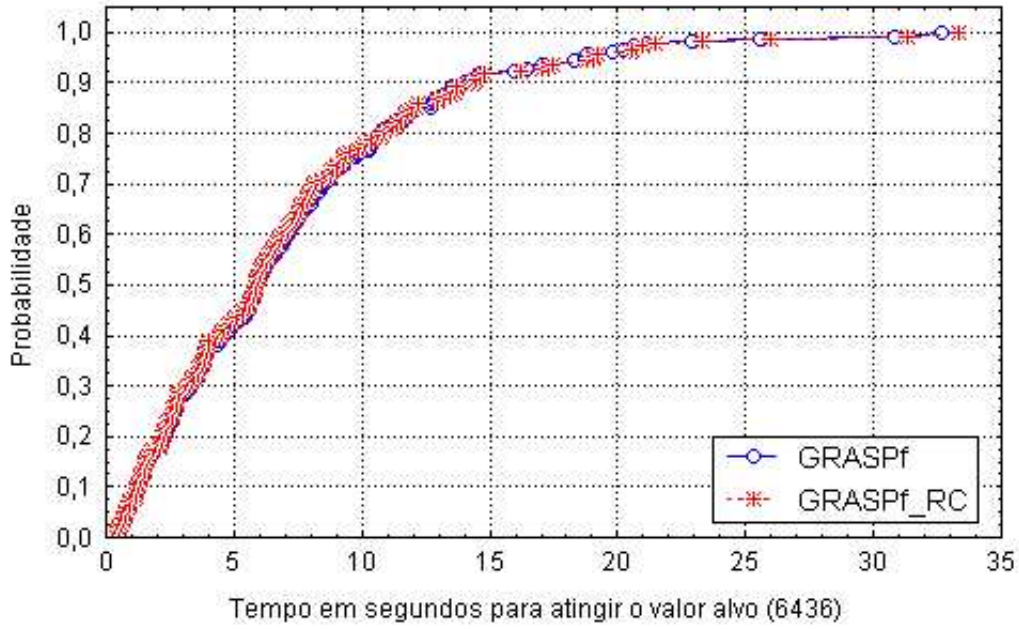

Figura 4.6: Distribuição de probabilidade empírica do tempo gasto para encontrar o valor alvo 6436 para a instância ORM_P9 ($w = 5$).

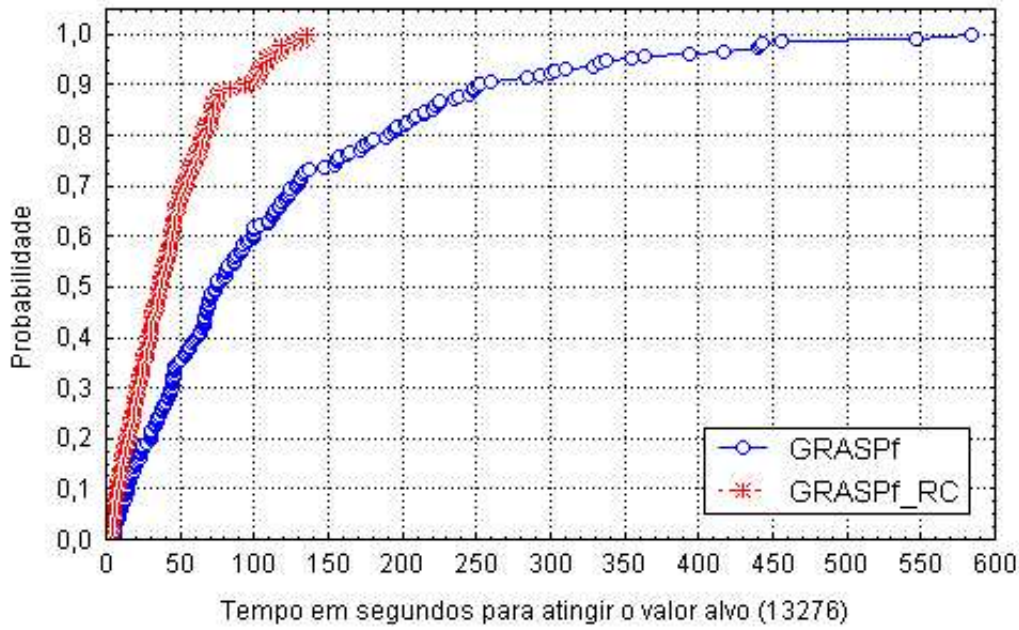

Figura 4.7: Distribuição de probabilidade empírica do tempo gasto para encontrar o valor alvo 13276 para a instância GRM_P17 ($w = 5$).

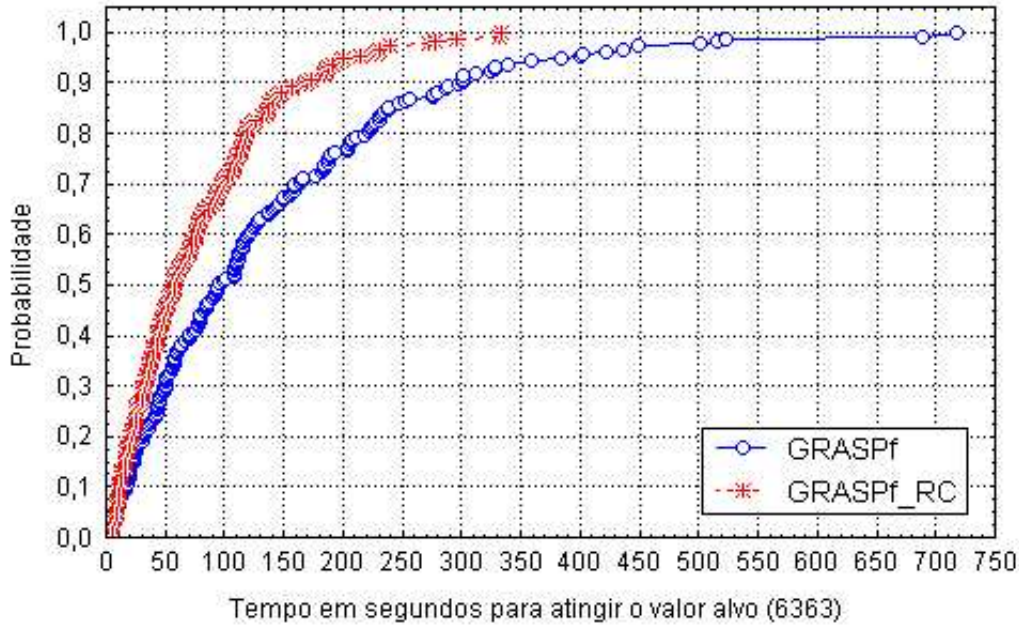

Figura 4.8: Distribuição de probabilidade empírica do tempo gasto para encontrar o valor alvo 6363 para a instância ORM_P15 ($w = 5$).

quanto ao anteriormente apresentado, mas o GRASPf_RC se mostra mais rápido do que o GRASPf para encontrar o valor alvo 6363. Para o primeiro algoritmo, a probabilidade de encontrar o valor alvo em menos do que 340 segundos é de 100% enquanto que para o segundo é aproximadamente 94%.

4.6 Algoritmo GRASP com Filtro e Reconexão por Caminhos

A heurística GRASP com filtro e reconexão por caminhos é mostrada na Figura 4.9.

Procedimento GRASP_Filtro_Reconexão_Caminhos (α, *semente*)
1. $P \leftarrow \emptyset$;
2. $F^* \leftarrow +\infty$;
3. **Para** $it = 1, \ldots, MaxIter$ **Faça**
4. $S \leftarrow$ Fase_Construção(α, *semente*);
5. $S_1^{'} \leftarrow$ BL_BTMA(S);
6. **Se** ($S_1^{'}$ não foi visitado previamente) **e** ($F(S_1^{'}) < (1+\lambda) \times F^*$) **Então**
7. $S^{'} \leftarrow$ BL_Teste($S_1^{'}$);
8. **Fim-Se**
9. **Senão**
10. $S^{'} \leftarrow S_1^{'}$;
11. **Fim-Senão**
12. $S_1 \leftarrow S^{'}$;
13. Atualizar o conjunto de soluções de elite P com S_1;
14. **Se** ($it \geq 2$) **Então**
15. Escolher uma solução $S_2 \in P$;
16. **Se** ($F(S_1) \leq F(S_2)$) **Então**
17. $S^{''} \leftarrow$ Reconexão_por_Caminhos(S_1, S_2);
18. **Fim-Se**
19. **Senão**
20. $S^{''} \leftarrow$ Reconexão_por_Caminhos(S_2, S_1);
21. **Fim-Senão**
22. **Se** ($F(S^{''}) < F^*$) **Então**
23. $F^* \leftarrow F(S^{''})$;
24. $S^* \leftarrow S^{''}$;
25. **Fim-Se**
26. **Fim-Se**
27. **Senão**
28. $F^* \leftarrow F(S^{'})$;
29. $S^* \leftarrow S^{'}$;
30. **Fim-Senão**
31. **Fim-Para**
32. **Retorne** S^***;**

Fim

Figura 4.9: Algoritmo GRASP com filtro e reconexão por caminhos.

O conjunto de soluções de elite P com tamanho máximo igual a dez está inicialmente vazio (linha 1). Executa-se o procedimento um número máximo de iterações (parâmetro *MaxIter*). Em cada iteração (laço das linhas 3 a 31), uma solução inicial é gerada pela fase de construção na linha 4. Em seguida, executa-se a busca local mais rápida (busca local pelas bordas com teste da menor aresta) na solução gerada pela fase de construção (linha 5). A linha 6 é responsável pelo teste da estratégia de filtro e caso ambas as condições sejam satisfeitas, executa-se a busca local mais lenta (busca local com teste) no ótimo local encontrado pela primeira busca (linha 7); caso contrário, se pelo menos uma das condições não for satisfeita, a busca local com teste não é executada. O ótimo local encontrado pela busca local mais rápida ou pela busca local mais lenta é um candidato a inserção no conjunto de soluções de elite. Caso satisfaça as condições de diversidade e qualidade apresentadas na Seção 4.5, torna-se um novo elemento do conjunto P (linha 13). Na linha 14, o procedimento de reconexão por caminhos começa a atuar a partir da segunda iteração. Primeiramente, uma solução é escolhida do conjunto de soluções de elite privilegiando-se elementos com uma maior diferença simétrica em relação ao ótimo local candidato (linha 15). Na linha 16, determina-se qual será a solução inicial e a solução alvo entre o ótimo local candidato e o elemento de P. A melhor solução será a solução inicial e a pior solução será a solução alvo (guia). Assim, de acordo com esse teste, executa-se o procedimento de reconexão por caminhos entre a solução inicial e a solução alvo nas linhas 17 ou 20. Caso o procedimento obtenha sucesso, retorna-se a melhor solução encontrada no caminho; caso contrário, retorna-se a melhor solução entre S_1 e S_2. Durante cada passo, se o procedimento encontra uma solução melhor do que a melhor solução encontrada até o momento, esta também torna-se uma candidata a inserção em P. Quando necessário, a melhor solução S^* é atualizada (linhas 22 a 25). Na primeira iteração, atualiza-se a melhor solução encontrada nas linhas 27 a 30. A melhor solução encontrada pelo GRASP com filtro e reconexão por caminhos é retornada na linha 32.

4.7 Considerações Finais

Este capítulo apresentou a heurística GRASP com filtro e reconexão por caminhos para o problema das p-medianas conectadas.

O algoritmo é composto por uma fase de construção cuja RCL é construída em função de dois critérios: critério de p-medianas e de Steiner. Comparando-se diversos valores de α, escolheu-se a configuração $\alpha = 0,4$, próxima da escolha gulosa para garantir qualidade média, mas suficientemente grande para garantir diversidade. A fase de busca local utilizada é aquela proposta no Capítulo 3.

Com o objetivo de tentar acelerar as iterações do GRASP, uma estratégia de filtro apresentada na literatura foi testada, atuando entre a busca local pelas bordas com teste da menor aresta e a busca local com teste. Em geral, a utilização do filtro reduz o tempo de processamento do algoritmo às custas de uma pequena perda na qualidade das soluções.

Com a finalidade de melhorar a qualidade das soluções encontradas pelo GRASP, o procedimento de reconexão por caminhos foi utilizado como estratégia de intensificação. Gráficos de distribuição de probabilidade empírica do tempo gasto para atingir o valor alvo foram mostrados com o objetivo de comparar o GRASP com filtro com e sem reconexão por caminhos. Os resultados mostraram que os ganhos são significativos em instâncias maiores. Isto pode ser explicado pelo fato de que, nessas instâncias, o caminho percorrido entre o ótimo local e o elemento do conjunto de soluções de elite tende a ser maior (maior diferença simétrica), aumentando-se a probabilidade de encontrar uma solução melhor do que a solução inicial e a solução alvo.