4
Algoritmos Primais

4.1
Representação da Solução Primal

Uma solução primal para o PMNC deve informar o valor da função objetivo, as facilidades abertas e a facilidade aberta que atende cada cliente. Para esta solução ser classificada como viável, devem ser abertas p facilidades e cada cliente deve ser atendido por uma única facilidade aberta e esta facilidade deve ser a mais próxima deste cliente dentre todas as facilidades abertas, conforme definido pelas restrições (2-1) a (2-5). Assim, a modelagem de uma solução primal é composta pelos seguintes conjuntos:

- OP - conjunto contendo o identificador das facilidade abertas ($|O| = p$)
- CP - conjunto contendo o identificador da facilidade aberta mais próxima de cada cliente ($|CP| = n$)
- SCP - conjunto contendo o identificador da segunda facilidade aberta mais próxima de cada cliente ($|SCP| = n$)

A justificativa para o armazenamento da segunda facilidade aberta mais próxima de cada cliente é devido ao fato que esta informação facilitará a definição de qual será a facilidade mais próxima de um cliente quando abrimos ou fechamos uma facilidade da solução. Na descrição da heurística de refinamento utilizada nesta dissertação (Seção 4.3) explicaremos melhor esta situação. Os conjuntos definidos nesta modelagem foram implementados utilizando *arrays* de inteiros com o intuito de podermos recuperar e acessar aleatoriamente cada elemento em tempo constante.

A forma usada para representarmos matrizes nos métodos implementados neste trabalho é a forma de representação completa, onde alocamos todas as posições de uma matriz de dimensão nxn devido ao fato de necessitarmos acessar cada uma destas posições em tempo constante.

4.2 Heurísticas Construtivas

Inicialmente, descreveremos a heurística construtiva pseudo-aleatória. Em seguida, discutiremos os métodos de construção gulosa ascendente, gulosa descendente e híbrida. Por fim, apresentaremos o método *Primal-Dual*.

4.2.1 Pseudo-Aleatória

Descrição

Na construção de uma solução pseudo-aleatória, primeiramente selecionamos as p facilidades a serem abertas de forma pseudo-aleatória e, por fim, atribuímos cada cliente a facilidade aberta mais próxima.

Implementação

A seleção das p facilidades a serem abertas possui complexidade de pior caso $O(p^2)$, pois, a cada passo, podemos acertar na escolha da facilidade ainda não aberta somente após ter ocorrida a seleção de k candidatos, onde k é o número de facilidades já abertas. Após a abertura das p facilidades, a atribuição de cada cliente a facilidade aberta mais próxima possui complexidade $O(pn)$ devido ao fato de precisarmos analisar a distância de cada cliente para cada facilidade aberta para podermos determinar a atribuição de custo mínimo. Para melhorar o desempenho de alguns métodos que serão descritos posteriormente, necessitamos armazenar a segunda facilidade aberta mais próxima de cada cliente e esta tarefa também é realizada em $O(pn)$, mas somente após termos definido a facilidade aberta mais dos mesmos. O cálculo da função objetivo da solução é feita durante a atribuição de cada cliente a facilidade aberta mais próxima. Consequentemente, este cálculo não prejudica o desempenho do algoritmo. Portanto, a complexidade final do algoritmo é $O(pn)$.

4.2.2 Gulosa Ascendente

Descrição

Para a construção de uma solução gulosa ascendente, iniciamos com uma solução vazia e adicionamos a mesma, em cada iteração, a facilidade que acarretará a maior queda no valor da função objetivo corrente dentre todas as candidatas a inserção.

Implementação

Inicialmente, escolhemos a primeira facilidade a ser inserida na solução vazia. Esta escolha é feita em $O(mn)$, pois, para cada facilidade, obtemos o somatório das distâncias desta facilidade aos clientes e guardamos as informações a respeito do somatório mínimo. Em seguida, inserimos na solução a facilidade que forneceu o somatório mínimo em tempo $O(n)$ devido ao fato de termos que atribuir cada cliente a esta facilidade. Prosseguindo, atribuímos as demais $p-1$ facilidades a solução corrente em tempo $O(pmn)$, pois teremos $p-1$ iterações onde, em cada iteração, é definida a melhor facilidade a ser inserida na solução corrente. Esta definição é feita calculando, para candidato, o ganho da inserção do mesmo na solução corrente em tempo $O(n)$ e guardando informações a respeito do ganho máximo, totalizando $O(mn)$. Dando continuidade, inserimos a facilidade que forneceu o ganho máximo em tempo $O(n)$, conforme já explicado anteriormente. Estas etapas em conjunto totalizam $O(mn)$ por iteração, finalizando as iterações com tempo $O(pmn)$. Após as $p-1$ iterações, definimos a segunda facilidade aberta mais próxima de cada cliente, tarefa esta realizada em $O(pn)$. Portanto, a complexidade final do algoritmo é $O(pmn)$.

4.2.3 Gulosa Descendente

Descrição

A heurística gulosa descendente inicia com todas as m facilidades abertas e, a cada iteração, é fechada a facilidade aberta que proporciona o menor incremento no valor da função objetivo em decorrência deste fechamento. São necessárias $m-p$ iterações para obtermos uma solução com p facilidades abertas.

Implementação

A primeira etapa deste procedimento consiste em abrir as m facilidades e atribuir cada cliente a facilidade aberta mais próxima. Se cada ponto for, ao mesmo tempo, uma possível localização de um cliente e de uma facilidade (os conjuntos de clientes e facilidades são idênticos), então esta etapa é concluída em tempo $O(m)$. Em seguida, definimos a segunda facilidade aberta mais próxima de cada cliente em tempo $O(mn)$. A última etapa consiste no fechamento das $m-p$ piores facilidades abertas da solução ao custo $O(n^2m)$. Esta última etapa consiste de $m-p$ iterações, onde em cada iteração é definida a melhor facilidade ser fechada em tempo $O(mn)$, pois, para cada facilidade aberta, calculamos o ganho obtido com o fechamento da mesma em tempo

$O(n)$. Após definida a melhor facilidade a ser fechada, a mesma é removida da solução em tempo $O(n^2)$, já que esta remoção implica na redefinição da primeira e da segunda facilidades abertas mais próxima de cada cliente afetado pelo fechamento. Portanto, a complexidade final do algoritmo é $O(n^2m)$.

4.2.4 Híbrida

Descrição

A heurística híbrida implementada neste trabalho é aquela proposta em (Captivo, 1991). Ela é similar a heurística construtiva gulosa ascendente descrita na Seção 4.2.2, porém difere devido ao fato de ser utilizado um procedimento para reconfigurar os elementos contidos em partições que são montadas a cada iteração do método construtivo.

Implementação

Após a primeira etapa do método construtivo guloso ascendente, que consiste na escolha e inserção da primeira facilidade para a solução vazia, é criada uma partição contendo a facilidade inserida como elemento central em conjunto com todos os clientes da instância (neste momento existe apenas uma facilidade aberta e todos os clientes estão sendo servidos por esta facilidade definindo, assim, a primeira partição da solução). Na segunda etapa do método construtivo guloso ascendente, são atribuídas as demais $p-1$ facilidades a solução corrente em tempo $O(pmn)$, porém na heurística híbrida esta etapa possui complexidade $O(p^2mn)$ devido ao fato de executarmos o procedimento de reconfiguração de partições no final de cada iteração em tempo $O(kn^2)$, onde k é o total de partições na iteração anterior a iteração corrente. Contudo, antes de executarmos o procedimento de reconfiguração, definimos o elemento central da nova partição que é a facilidade que foi inserida na solução corrente. Neste momento, a nova partição é composta pela facilidade inserida em conjunto com os clientes mais próximos da mesma. Como consequência, as demais partições criadas podem ter sido modificadas em decorrência da alteração da facilidade mais próxima a alguns clientes. Após a criação e ajuste das partições, aplicados o procedimento de reconfiguração sobre as mesmas, exceto para a última partição criada. Como este procedimento analisa cada partição e define a melhor facilidade a ser aberta dentro desta partição, é desnecessário aplicá-la a última partição criada já que está partição possui uma configuração ótima. Contudo, a complexidade final do algoritmo é $O(p^2mn)$.

4.2.5 Primal-Dual

Descrição

A heurística construtiva *Primal-Dual* utiliza a solução dual gerada pelo método *Dual Ascent* para construir uma solução primal viável utilizando as condições de complementaridade de folga descritas na Seção 5.2.1. O método *Dual Ascent* constrói um conjunto de facilidades I^+ tal que $\overline{c}_\pi(j) = 0$ para cada $j \in I^+$ e, para cada $i \in U$, $\lambda_i \geq d_{ij}$ para algum $j \in I^+$. I^+ será o conjunto das facilidades abertas na solução primal, ou seja, $y_i^* = 1$ se $i \in I^+$ e $y_i^* = 0$ caso contrário. De acordo com as restrições (5-1), se $\overline{c}_\pi(j) = 0$ então a variável y_i^* poderá assumir qualquer valor. Porém, se $\overline{c}_\pi(j) \neq 0$ então $y_i^* = 0$. Porém, a facilidade i estará aberta na solução primal ($y_i^* = 1$) se e somente se $\overline{c}_\pi(j) = 0$. Caso contrário, a mesma permanecerá fechada ($y_i^* = 0$). Como consequência, o método construtivo nunca violará as restrições (5-1). No entanto, não temos nenhuma garantia quanto a não violação das restrições (5-2) e (5-3).

Na solução primal derivada da solução dual, cada cliente i será atribuído a facilidade aberta j mais próxima do mesmo, ou seja, $x_{ij}^* = y_i^* = 1$ somente para a facilidade j aberta mais próxima de i. As restrições (5-2) indicam que se $y_i^* - x_{ij}^* = 0$ então $\max(0, \lambda_i^* - d_{ij})$ pode assumir qualquer valor. Porém, se $y_i^* - x_{ij}^* \neq 0$ então $\max(0, \lambda_i^* - d_{ij})$ deverá ser nulo. Assim, $\max(0, \lambda_i^* - d_{ij})$ poderá ser diferente de zero apenas quando $y_i^* - x_{ij}^* = 0$. Consequentemente, para que as restrições (5-2) não sejam violadas, para cada $i \in U$, $\lambda_i \geq d_{ij}$ para um único $j \in I^+$, desde que, na solução primal, cada cliente i será atribuído a facilidade aberta j mais próxima do mesmo. No entanto, o método *Dual Ascent* não garante esta condição, podendo ocorrer violações das restrições (5-2). A respeito das restrições (5-3), γ sempre será diferente de zero, porém o método *Dual Ascent* não fornece nenhuma garantia quanto a abertura de exatamente p facilidades havendo, também, a possibilidade da ocorrência de violações deste último grupo de restrições.

Implementação

Para construir uma solução primal viável α a partir da solução dual π fornecida pelo método *Dual Ascent*, primeiramente definimos as facilidades abertas em α. Para isto, percorremos todo o conjunto I^+, conjunto este construído pelo método *Dual Ascent*, para podermos definir quais facilidades serão abertas. Esta etapa possui complexidade $O(n)$. Na etapa seguinte, atribuímos cada cliente a facilidade aberta mais próxima do mesmo e calculamos o valor da função objetivo primal durante esta atribuição gastando, no total, um tempo

$O(n^2)$. Em seguida, atribuímos cada cliente a segunda facilidade aberta mais próxima do mesmo, também em tempo $O(n^2)$. Neste momento, possuímos uma solução primal que pode não ser viável devido ao fato desta solução possuir mais ou menos do que p facilidades abertas. Seja $p^{'}$ o total de facilidades abertas em α. Se $p^{'} > p$ então necessitamos fechar $p^{'} - p$ facilidades. Se $p^{'} < p$ então necessitamos abrir $p - p^{'}$ facilidades. Ambas as possibilidades (fechamento ou abertura) gastam um tempo $O(pn^2)$ para serem efetivadas.

No fechamento de facilidades, temos $p^{'} - p$ facilidades a serem fechadas. São $p^{'} - p$ iterações onde, em cada iteração, é definida a melhor facilidade a ser fechada em tempo $O(mn)$, pois, para cada facilidade aberta, calculamos o ganho obtido com o fechamento da mesma em tempo $O(n)$. Após definida a melhor facilidade a ser fechada, a mesma é removida da solução em tempo $O(n^2)$, já que esta remoção implica na redefinição da primeira e da segunda facilidades abertas mais próxima de cada cliente afetado pelo fechamento. Portanto, a complexidade final do algoritmo é $O(n^2m)$.

Na abertura de facilidades, atribuímos as demais $p - p^{'}$ facilidades a solução corrente. São $p - p^{'}$ iterações onde, em cada iteração, é inserida a facilidade que fornece o maior ganho para a solução. Esta inserção é realizada calculando, para cada um dos $m - p^{'}$ candidatos, o ganho da inserção do mesmo na solução. Este cálculo gasta um tempo $O(n)$, totalizando $O((m-p^{'})n)$ para a definição da melhor facilidade a ser inserida. Em seguida, inserimos na solução esta facilidade em tempo $O(n)$. Contudo, a etapa de abertura de facilidades possui complexidade $O(pmn)$, pois são $p-p^{'}$ iterações ao custo de $O((m-p^{'})n)$ por iteração.

4.3 Heurística de Refinamento

4.3.1 Descrição

Nesta seção descreveremos o método de busca local implementando nesta dissertação. Esta implementação é aquela proposta em (Resende e Werneck, 2004). Conforme discutido na Seção 4.3, este método consiste em analisar todas as possíveis trocas de papéis entre um cliente e uma facilidade em uma dada solução e efetuar a troca mais vantajosa. Quando obtemos a nova solução decorrente desta troca, repetimos esta análise. São feitas análises até que não seja mais possível melhorar a solução corrente.

4.3.2 Implementação

O pseudocódigo da heurística de refinamento está descrito em (Resende e Werneck, 2004).

4.4 Análise Comparativa

4.4.1 Soluções Obtidas

As tabelas 4.1, 4.2, 4.3, 4.4 e 4.5 apresentam, para cada instância das classes tratadas e para cada heurística construtiva implementada (*A* - Pseudo-Aleatória, *GA* - Gulosa Ascendente, *GD* - Gulosa Descendente, *H* - Híbrida, *PD* - Primal-Dual), duas medidas de qualidade. São elas:

- dP_1: desvio primal sem a aplicação da busca local sobre a solução primal encontrada;
- dP_2: desvio primal com a aplicação da busca local sobre solução primal encontrada.

Para as instâncias da classe *OR-Library* a coluna *(n,p)* informa a dimensão da instância, sendo n o número de pontos representando clientes/facilidades e p o número de pontos que devem representar facilidades abertas. Para as instâncias das classes *TSP-Library* apresentamos apenas o valor do parâmetro p, pois o nome de cada instância já fornece o valor do parâmetro n.

Para que fosse possível obter uma análise consistente das soluções fornecidas pela heurística construtiva pseudo-aleatório (A), para cada instância tratada foram realizadas 10 execuções, cada qual partindo de uma semente diferente de números pseudo-aleatórios. Os tempos de execuções de todas as heurísticas construtivas aplicadas às instâncias apresentadas a seguir estão descritas no apêndice B.

A heurística construtiva A apresentou os piores resultados para todas as instâncias testadas, conforme já esperado. Porém, aplicando-se a busca local a partir das soluções obtidos por A, foram encontradas soluções de qualidade bem próximas a aquelas fornecidas com a aplicação da busca local a partir das soluções encontradas pelos demais métodos construtivos. Isto comprova a eficiência deste tipo de busca local quando aplicado ao PMNC, pois partindo de uma solução inicial de péssima qualidade, a busca local consegue refinar estas soluções deixando-as bem próximas do ótimo das instâncias onde o ótimo é conhecido.

Considerando o número de melhores soluções encontradas por cada método, analisando os resultados obtidos para as instâncias da classe *OR-Library* (tabelas 4.1 e 4.2), podemos observar que o método que apresentou melhores resultados desconsiderando a aplicação da busca local (Tabela 4.1) foi o *PD*, seguido do *H*. Os piores desempenhos foram obtidos pelos métodos *GD* e *A*, nesta ordem. No entanto, com a aplicação da busca local (Tabela 4.2) após a construção da solução inicial, os resultados obtidos foram bastante equilibrados.

Instância	**(n,p)**	**A**	**GA**	**GD**	**H**	**PD**
pmed1	(100,5)	41,49	1,24	**0,14**	1,24	1,27
pmed2	(100,10)	46,53	0,61	1,20	1,32	**0,00**
pmed3	(100,10)	54,48	3,51	3,84	**0,00**	0,07
pmed4	(100,20)	52,92	1,78	4,58	0,40	**0,13**
pmed5	(100,33)	72,59	1,70	2,88	**0,15**	0,22
pmed6	(200,5)	30,04	2,59	2,24	2,59	**0,12**
pmed7	(200,10)	45,77	0,27	2,61	0,25	**0,14**
pmed8	(200,20)	46,83	0,61	3,53	**0,27**	2,05
pmed9	(200,40)	55,40	3,91	6,88	0,95	**0,00**
pmed10	(200,67)	85,56	3,19	5,10	1,75	**0,72**
pmed11	(300,5)	28,08	0,32	3,44	**0,23**	0,35
pmed12	(300,10)	43,85	0,26	5,44	0,26	**0,21**
pmed13	(300,30)	41,98	2,13	5,44	1,69	**0,21**
pmed14	(300,60)	51,96	1,52	4,45	1,68	**0,47**
pmed15	(300,100)	64,02	1,85	5,44	0,58	**0,23**
pmed16	(400,5)	35,40	**0,86**	1,52	**0,86**	1,19
pmed17	(400,10)	44,16	0,29	4,31	**0,16**	0,69
pmed18	(400,40)	41,25	1,33	4,20	0,42	**0,17**
pmed19	(400,80)	52,68	1,90	6,36	0,60	**0,35**
pmed20	(400,133)	76,14	4,30	6,15	1,06	**0,28**
pmed21	(500,5)	40,83	**0,00**	3,76	**0,00**	**0,00**
pmed22	(500,10)	39,20	1,06	1,91	1,06	**0,68**
pmed23	(500,50)	43,32	1,62	4,16	**0,32**	1,15
pmed24	(500,100)	51,52	1,62	6,69	0,74	**0,44**
pmed25	(600,167)	71,35	3,72	6,78	1,31	**0,88**
pmed26	(600,5)	37,33	1,77	2,32	0,44	**0,07**
pmed27	(600,10)	37,38	0,69	2,35	**0,64**	1,56
pmed28	(600,60)	41,90	1,80	5,54	0,69	**0,27**
pmed29	(600,120)	48,65	2,34	7,39	**0,59**	0,69
pmed30	(600,200)	63,45	2,41	6,64	**1,11**	**1,11**
pmed31	(700,5)	36,15	**0,00**	2,65	**0,00**	0,70
pmed32	(700,10)	39,61	0,37	2,67	**0,23**	1,31
pmed33	(700,70)	42,41	2,09	4,55	0,74	**0,38**
pmed34	(700,140)	51,62	2,79	5,64	1,66	**0,33**
pmed35	(800,5)	32,45	0,06	3,45	**0,00**	0,05
pmed36	(800,10)	37,50	0,20	2,12	**0,13**	0,45
pmed37	(800,80)	44,72	1,21	5,12	0,53	**0,42**
pmed38	(900,5)	39,14	**0,84**	1,86	**0,84**	1,10
pmed39	(900,10)	37,57	0,30	2,00	0,30	**0,17**
pmed40	(900,90)	40,93	1,21	4,33	0,76	**0,62**

Tabela 4.1: Resultados - Métodos construtivos aplicados as instâncias da classe *OR-Library* - Medida de qualidade dP_1

Considerando, também, o número de melhores soluções encontradas por cada método, analisando as tabelas 4.3, 4.4 e 4.5, que apresentam os resultados obtidos sobre as instâncias *fl1400*, *pcb3038* e *rl5934*, podemos concluir o seguinte: para ambas as instâncias, os melhores resultados foram obtidos com os métodos construtivos *H* e *PD*, desconsiderando o refinamento

Instância	(n,p)	A	GA	GD	H	PD
pmed1	(100,5)	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
pmed2	(100,10)	0,31	0,29	**0,00**	**0,00**	**0,00**
pmed3	(100,10)	0,22	**0,00**	**0,00**	**0,00**	**0,00**
pmed4	(100,20)	0,36	0,40	**0,13**	0,40	**0,13**
pmed5	(100,33)	0,44	**0,00**	1,55	0,15	0,15
pmed6	(200,5)	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
pmed7	(200,10)	0,10	0,25	**0,00**	0,25	0,14
pmed8	(200,20)	0,34	0,27	**0,00**	0,20	0,20
pmed9	(200,40)	0,56	0,69	0,48	0,48	**0,00**
pmed10	(200,67)	**0,62**	0,64	0,64	0,64	0,64
pmed11	(300,5)	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
pmed12	(300,10)	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
pmed13	(300,30)	0,03	**0,00**	0,16	**0,00**	**0,00**
pmed14	(300,60)	0,30	0,10	**0,07**	0,44	0,10
pmed15	(300,100)	0,73	0,52	0,98	0,52	**0,06**
pmed16	(400,5)	0,08	**0,00**	0,26	**0,00**	**0,00**
pmed17	(400,10)	0,17	**0,00**	**0,00**	**0,00**	**0,00**
pmed18	(400,40)	0,16	**0,04**	0,12	**0,04**	**0,04**
pmed19	(400,80)	0,44	0,49	**0,18**	0,21	0,21
pmed20	(400,133)	0,41	0,89	0,17	0,67	**0,00**
pmed21	(500,5)	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
pmed22	(500,10)	0,30	1,05	**0,00**	1,05	**0,00**
pmed23	(500,50)	0,14	**0,00**	0,28	**0,00**	**0,00**
pmed24	(500,100)	0,59	**0,20**	0,34	0,24	0,24
pmed25	(600,167)	1,12	0,82	0,66	0,93	**0,33**
pmed26	(600,5)	0,01	**0,00**	0,07	**0,00**	**0,00**
pmed27	(600,10)	0,01	**0,00**	0,04	**0,00**	**0,00**
pmed28	(600,60)	0,18	0,33	0,20	**0,02**	**0,02**
pmed29	(600,120)	0,50	0,20	0,56	**0,03**	**0,03**
pmed30	(600,200)	1,12	1,01	0,85	**0,60**	**0,60**
pmed31	(700,5)	**0,00**	**0,00**	0,01	**0,00**	**0,00**
pmed32	(700,10)	0,03	0,04	**0,00**	0,04	0,04
pmed33	(700,70)	0,28	0,30	**0,09**	**0,09**	**0,09**
pmed34	(700,140)	0,39	0,56	0,60	0,63	**0,27**
pmed35	(800,5)	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
pmed36	(800,10)	0,27	**0,00**	0,40	**0,00**	**0,00**
pmed37	(800,80)	0,23	0,12	0,32	**0,02**	**0,02**
pmed38	(900,5)	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
pmed39	(900,10)	**0,00**	**0,00**	**0,00**	**0,00**	**0,00**
pmed40	(900,90)	0,22	0,25	**0,12**	0,25	0,25

Tabela 4.2: Resultados - Métodos construtivos aplicados as instâncias da classe *OR-Library* - Medida de qualidade dP_2

da solução com a busca local. Para as instâncias *fl1400* e *rl5934*, o método *H* apresentou melhores resultados e para a instância *pcb3038* o método *PD* apresentou melhores resultados. No entanto, aplicando a busca local sobre as soluções iniciais, o resultado tende a ser mais homogêneo, com a obtenção das melhores soluções distribuídas entre os métodos, exceto, é claro, para o método *A*, que apresenta péssimos resultados em ambas as instâncias. Considerando ainda as soluções obtidas após o refinamento, os métodos *GD*, *H* e *PD* se destacaram para ambas as instâncias e os métodos *GA* e *A* apresentaram os piores resultados, nesta ordem.

As figuras 4.1 e 4.2 mostram, respectivamente, a influência do parâmetro *p* no valor das soluções primal e dual para a instância *fl1400* da classe *TSP-Library*. Para o valor da solução primal, a medida que *p* aumenta, o valor da solução diminui. Quando aumentamos a quantidade de facilidades abertas,

p	A		GA		GD		H		PD	
	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2
10	176,55	0,07	5,22	**0,00**	9,70	0,68	**0,00**	**0,00**	0,30	**0,00**
20	102,54	0,51	9,38	0,08	9,44	**0,00**	0,78	0,49	0,35	0,49
30	132,44	0,86	4,27	2,01	8,47	1,47	0,75	**0,62**	2,06	**0,62**
40	121,15	0,19	5,79	**0**,06	9,00	0,57	0,93	**0,05**	1,56	**0,05**
50	119,51	0,90	6,88	1,91	9,71	1,94	2,07	**0,00**	1,96	**0,00**
60	143,83	0,86	5,83	1,36	9,08	0,28	1,97	**0,23**	1,71	**0,23**
70	153,88	0,97	5,78	1,39	9,30	0,90	1,43	**0,48**	1,28	**0,48**
80	162,38	0,71	5,20	0,61	9,19	0,72	1,28	**0,60**	1,38	**0,60**
90	143,80	1,25	4,85	1,17	8,71	**0,43**	1,55	0,87	2,38	0,87
100	156,94	0,55	5,05	**0,20**	8,26	0,48	0,71	0,41	1,55	0,41
150	172,72	0,80	4,57	0,73	6,47	**0,31**	1,82	1,02	2,55	1,02
200	176,08	0,88	6,02	1,12	7,62	0,95	1,48	**0,51**	2,07	**0,51**
250	177,69	0,55	5,11	0,72	6,97	0,41	1,39	**0,30**	1,81	**0,30**
300	196,62	0,78	4,00	**0,74**	7,59	0,82	1,78	1,19	1,88	1,19
350	208,57	1,39	3,80	1,22	8,36	**1,01**	2,72	1,82	2,36	1,82
400	196,41	1,76	4,44	1,87	8,21	**1,35**	4,08	2,34	3,95	2,34
450	191,73	1,12	3,29	1,41	7,48	0,67	2,95	**0,93**	3,19	**0,93**
500	200,08	**0,97**	2,48	1,21	6,85	1,01	2,24	1,13	2,57	1,13

Tabela 4.3: Resultados - Métodos construtivos aplicados a instância *fl1400* da classe *TSP-Library*

p	A		GA		GD		H		PD	
	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2
10	39,51	0,39	4,64	**0,00**	10,90	**0,00**	1,71	1,36	3,44	1,36
20	43,61	0,68	5,06	0,68	7,58	0,93	0,75	0,32	3,85	**0,32**
30	46,73	0,68	6,72	0,58	5,92	1,09	1,74	0,32	2,17	**0,32**
40	42,13	0,83	7,47	1,35	7,05	0,92	1,56	0,45	2,52	**0,45**
50	47,10	0,69	6,55	**0,31**	6,71	0,76	0,95	0,35	1,84	0,35
60	46,64	1,08	6,03	0,68	6,45	**0,45**	1,98	1,35	1,43	1,35
70	45,96	0,87	4,97	0,46	6,11	**0,45**	1,94	0,90	1,92	0,90
80	42,66	0,98	4,72	1,04	6,02	0,70	1,40	**0,66**	2,07	**0,66**
90	44,45	1,13	4,76	1,71	6,16	1,05	1,87	**0,79**	2,08	**0,79**
100	43,33	1,08	4,94	1,50	6,29	0,95	1,85	**0,59**	1,90	**0,59**
150	41,88	0,90	5,63	0,84	6,38	**0,71**	2,16	0,86	1,77	0,86
200	40,92	0,86	5,90	1,03	6,10	**0,51**	1,92	0,83	1,36	0,83
250	46,50	0,83	6,15	0,68	6,08	**0,62**	2,01	0,82	1,34	0,82
300	43,59	0,85	5,98	0,82	6,40	**0,77**	2,31	0,99	1,02	0,99
350	43,89	0,94	5,73	1,09	6,31	**0,78**	2,10	0,68	1,14	0,68
400	42,99	0,95	5,73	1,15	6,25	**0,82**	2,57	1,04	0,93	1,04
450	44,02	**0,93**	5,65	**0,93**	6,11	0,88	2,41	1,00	1,04	1,00
500	44,63	1,01	5,48	0,93	5,89	**0,87**	2,42	1,04	0,88	1,04
550	44,58	1,06	5,31	0,98	5,76	**0,87**	2,50	1,04	0,99	1,04
600	46,68	1,11	5,32	1,24	5,93	0,98	2,60	0,92	1,11	**0,92**
650	46,47	1,31	5,46	1,31	6,31	**1,24**	2,81	1,30	1,61	1,30
700	46,38	1,40	5,64	1,60	6,55	1,46	2,98	**1,24**	1,78	**1,24**
750	45,32	**1,38**	5,69	1,53	6,54	1,47	2,96	1,47	1,54	1,47
800	44,10	**1,39**	5,69	1,45	6,41	1,55	3,09	1,44	1,65	1,44
850	44,34	1,44	5,63	1,41	6,27	1,72	3,07	**1,24**	1,75	**1,24**
900	44,11	**1,28**	5,61	1,56	6,20	1,62	2,98	**1,28**	1,71	**1,28**
950	42,89	**1,31**	5,52	**1,31**	6,10	1,38	2,89	1,32	1,41	1,32
1000	43,25	1,19	5,27	**1,10**	5,83	1,40	2,61	1,11	1,17	1,11

Tabela 4.4: Resultados - Métodos construtivos aplicados a instância *pcb3038* da classe *TSP-Library*

p	A		GA		GD		H		PD	
	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2	dP_1	dP_2
10	37,73	0,46	5,31	0,32	6,39	0,27	0,61	**0,03**	0,61	**0,03**
20	33,23	0,39	5,41	**0,07**	7,29	1,19	2,52	0,28	2,52	0,28
30	41,86	0,73	6,89	0,55	6,74	0,40	0,85	**0,32**	0,85	**0,32**
40	46,69	0,78	6,91	1,77	6,34	0,46	1,13	**0,27**	1,13	**0,27**
50	47,27	0,66	6,40	1,12	5,49	0,79	1,40	**0,36**	1,40	**0,36**
60	45,23	0,77	5,74	**0,33**	5,82	0,76	1,78	0,96	1,78	0,96
70	46,07	0,73	5,21	**0,54**	6,46	1,06	1,15	0,71	1,15	0,71
80	47,31	0,77	5,27	0,53	6,87	1,32	0,94	**0,55**	0,94	**0,55**
90	50,17	0,78	5,32	0,74	6,66	0,81	1,23	**0,59**	1,23	**0,59**
100	49,53	0,81	5,39	0,70	6,35	0,75	1,10	**0,62**	1,10	**0,62**
150	51,35	0,86	6,12	0,93	6,66	0,96	1,43	**0,74**	1,43	**0,74**
200	50,82	**0,74**	6,17	0,82	6,16	0,75	1,50	0,81	1,50	0,81
250	52,89	0,80	6,99	0,88	6,07	0,75	1,41	0,64	1,41	**0,64**
300	53,81	0,91	7,30	1,04	5,78	**0,56**	1,33	0,66	1,33	0,66
350	53,73	0,85	7,37	0,86	5,71	0,74	1,36	**0,58**	1,36	**0,58**
400	55,26	0,79	7,39	0,74	5,81	0,78	1,36	**0,61**	1,36	**0,61**
450	56,72	0,70	7,47	0,82	5,69	0,64	1,47	**0,56**	1,47	**0,56**
500	58,26	0,76	7,67	0,89	5,55	0,69	1,56	**0,65**	1,56	**0,65**
600	60,19	0,61	7,62	0,77	5,13	0,40	1,49	**0,60**	1,49	**0,60**
700	61,42	0,58	7,25	0,55	5,05	**0,40**	1,66	0,64	1,66	0,64
800	63,62	0,61	7,03	0,79	5,05	**0,37**	1,63	0,64	1,63	0,64
900	65,01	0,68	7,04	0,78	5,26	**0,41**	1,74	0,87	1,74	0,87
1000	66,49	0,74	6,99	0,86	5,47	**0,55**	1,80	0,72	1,80	0,72
1100	66,80	0,73	6,92	0,83	5,65	**0,61**	1,65	0,71	1,65	0,71
1200	69,14	0,88	6,94	0,95	5,95	0,96	1,78	**0,73**	1,78	**0,73**
1300	72,54	0,97	7,17	0,93	6,37	0,97	1,79	**0,77**	1,79	**0,77**
1400	74,40	1,05	7,32	1,10	6,59	1,06	1,77	**0,85**	1,77	**0,85**
1500	72,74	1,17	7,38	1,15	6,76	1,28	1,86	**0,88**	1,86	**0,88**

Tabela 4.5: Resultados - Métodos construtivos aplicados a instância *rl5934* da classe *TSP-Library*

a distância mínima de cada cliente a facilidade aberta mais próxima diminui. Como o valor da solução é o somatório destas distâncias, este valor decrementa. Podemos observar um comportamento similar para o valor da solução dual.

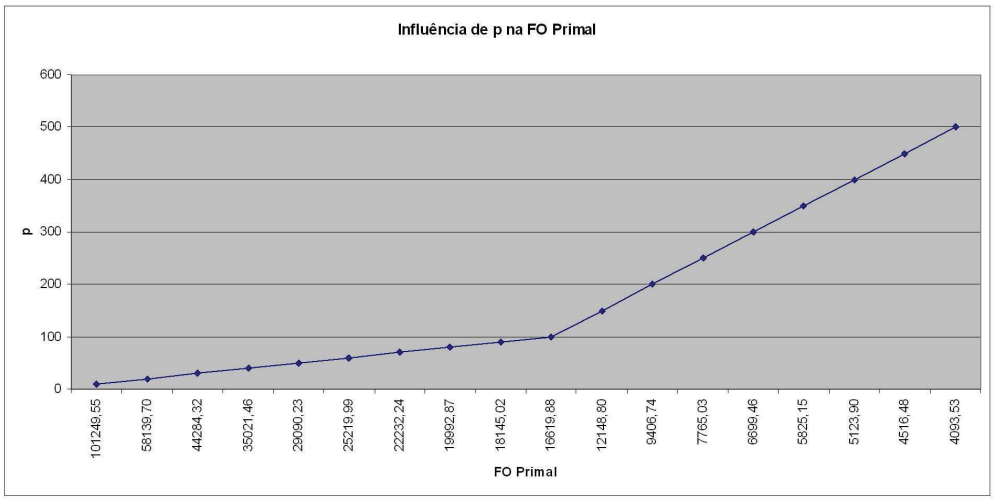

Figura 4.1: Influência de p no valor da solução primal para a instância *fl1400* da classe *TSP-Library*

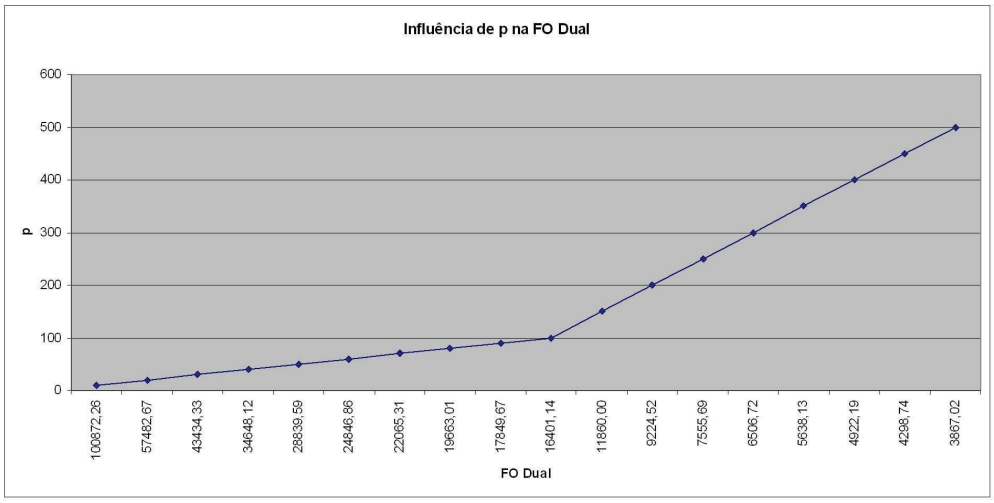

Figura 4.2: Influência de p no valor da solução dual para a instância *fl1400* da classe *TSP-Library*

4.4.2
Complexidade e Tempo de Execução

A Tabela 4.6 apresenta a complexidade de pior caso dos algoritmos primais implementados para este trabalho.

Algoritmo	**Complexidade de Pior Caso**
H. C. Pseudo-Aleatória	$O(pn)$
H. C. Gulosa Ascentende	$O(pmn)$
H. C. Gulosa Descendente	$O(n^2m)$
H. C. Híbrida	$O(p^2mn)$
H. C. Primal-Dual	$O(pmn)$
H. de Refinamento	$O(mn)$

Tabela 4.6: Complexidade de pior caso dos algoritmos primais

Em B são apresentados os tempos de execuções das heurísticas construtivas.

4.5
Conclusão

Diante dos resultados expostos neste capítulo, podemos concluir que a escolha da melhor heurística construtiva é fortemente dependente da configuração da instância tratada. Portanto, a identificação de qual destes métodos construtivos é o melhor não é adequada devido ao estreito relacionamento entre a qualidade fornecida pelos métodos e a configuração das instâncias tratadas, exceto para o método A cujo desempenho foi insatisfatório para todas as instâncias quando não aplicamos a busca local sobre as soluções construídas.