Universidade Estadual de Campinas

Instituto de Computação

MO420 - Programação Linear Inteira

Relatório do Trabalho Prático 1

Luis Henrique Pauleti Mendes
RA 117801

Campinas, 21 de Maio de 2018

Sumário

1 Introdução

Neste trabalho foi estudado o Problema da Árvore Geradora de Custo Mínimos com Conflitos (AGMCC) e implementado algoritmos lagrangianos para resolvê-lo. Seja um grafo $G = (V, E)$. não direcionado, com $|V| = n$ e $|E| = m$. A cada aresta e de E está associado um custo c_e. Além disso, é dado um conjunto S de pares de arestas, chamado de *conjunto de conflito*, com $|S| = p$. Cada elemento de $\{e, f\}$ em S é chamado de um *par (de arestas) conflitante*. Uma árvore geradora T de G é dita ser livre de conflitos se nenhum par de arestas em T é conflitante. No AGMCC o que se deseja é encontrar uma árvore geradora de custo mínimo que seja livre de conflitos.

Seja $\mathcal{F}$ o conjunto de todos os vetores de incidência (ou característicos) que representam árvores geradoras de G. Denotando por x um destes vetores, temos que $x \in \mathbb{B}^m$. Assim, o AGMCC pode ser modelado por:

$$(IP) \quad z = min\{\sum_{e \in E} c_e x_e\} \tag{1.1}$$

sujeito a:

$$x_e + x_f \leq 1, \forall \{e, f\} \in S \tag{1.2}$$

$$x \in \mathcal{F} \tag{1.3}$$

Denote por S_e o conjunto de pares conflitantes envolvendo a aresta e. Seja ainda e^* a aresta de E com menor custo para o qual S_e não é vazio. Finalmente, caso da inequação (1.2), referente ao par $\{e, f\}$ de S seja dualizada (penalizada) em uma relaxação lagrangiana, seja u_{ef} o multiplicador de Lagrange correspondente. A partir daí, vamos considerar duas possíveis relaxações lagrangianas para o AGMCC. São elas:

- **Relaxação 1:**

$$(RL^1) \quad z^1(u) = min\{\sum_{e \in E} (c_e + \sum_{\{e,v\} \in S_e} u_{ef}) x_e - \sum_{\{e,f\} \in S} u_{ef}\} \tag{1.4}$$

  sujeito a:

$$x \in \mathcal{F} \tag{1.5}$$

  onde $u \in \mathbb{R}_+^{|S|}$

- **Relaxação 2:**

$$(RL^2) \quad z^2(u) = min\{\sum_{e \in E} (c_e + \sum_{\{e,v\} \in S_e \setminus S_{e^*}} u_{ef}) x_e - \sum_{\{e,f\} \in S \setminus S_{e^*}} u_{ef}\} \tag{1.6}$$

  sujeito a:

$$x_{e^*} + x_f \leq 1, \forall \{e^*, f\} \in S_{e^*} \tag{1.7}$$

$$x \in \mathcal{F} \tag{1.8}$$

onde $u \in \mathbb{R}_+^{|S \setminus S_{e^*}|}$

Seja z_{LP} o valor da relaxação linear dada por este modelo e w_{LD}^k o valor ótimo do dual lagrangiano para a relaxação RL^k, $k = 1, 2$.

2 Respostas das Questões Teóricas

1. **Formule o problema dual lagrangiano w_{LD}^k associado a cada relaxação RL^k.**

   O problema dual lagrangiano associado à relaxação RL^1 é dado por:

$$(LD^1) \quad w_{LD}^1 = max\{z^1(u) : u \in \mathbb{R}_+^{|S|}\} \tag{2.1}$$

   Pelo Teorema 10.3 de Wolsey (1998), temos:

$$(DL^1) \quad w_{LD}^1 = min\{\sum_{e \in E} c_e x_e\} \tag{2.2}$$

   sujeito a:

$$x_e + x_f \leq 1, \forall \{e, f\} \in S \tag{2.3}$$

$$x \in conv(\mathcal{F}) \tag{2.4}$$

   O problema dual lagrangiano associado à relaxação RL^2 é dado por:

$$(LD^2) \quad w_{LD}^2 = max\{z^2(u) : u \in \mathbb{R}_+^{|S \setminus S_{e^*}|}\} \tag{2.5}$$

   Para cada aresta e de E, define-se $\mathcal{F}_e = \{x \in \mathcal{F} : x_e + x_f \leq 1, \forall \{e, f\} \in S_e\}$.

   Pelo Teorema 10.3 de Wolsey (1998), temos:

$$(DL^2) \quad w_{LD}^2 = min\{\sum_{e \in E} c_e x_e\} \tag{2.6}$$

   sujeito a:

$$x_e + x_f \leq 1, \forall \{e, f\} \in S \setminus S_{e^*} \tag{2.7}$$

$$x \in conv(\mathcal{F}_{e^*}) \tag{2.8}$$

2. **Algoritmos para resolver o *problema da árvore geradora mínima* (PAGM) e, por consequência, o problema primal lagrangiano ($z^1(u)$) da primeira relaxação, podem ser encontrados em bons livros-texto de projeto e análise de algoritmos. Diga qual o algoritmo você usou para resolver o PAGM na sua implementação, destacando a complexidade de pior caso**

do mesmo e citando a fonte de suas informações (referências bibliográficas). Justifique a escolha do algoritmo utilizado. levando em consideração as características dos grafos usados nos testes.

O algoritmo utilizado para resolver o PAGM foi o algoritmo de Kruskal, cuja complexidade de pior caso é dada por $O(m \log n)$. Outro algoritmo conhecido para resolver o PAGM é o algoritmo de Prim, cuja complexidade de pior caso é dada por $O(m + n \log n)$ (CORMEN et al., 2009).

O algoritmo de Kruskal foi escolhido pois o grafo mais denso nas instâncias de teste tem $m = 5n$, ou seja, os grafos nas instâncias de teste são tais que $m \in O(n)$. Deste modo, nas instâncias de teste, temos que a complexidade pior caso do algoritmo de Kruskal, dada por $O(n \log n)$, é melhor que a complexidade de pior caso do algoritmo de Prim, dada por $O(n + n \log n)$.

3. **Usando resultados teóricos vistos em aula, diga se a afirmação seguinte é falsa ou verdadeira: *"A relaxação* RL^1 *tem a propriedade de integralidade, ou seja,* $w^1_{LD} = z_{LP}$*."* Em caso afirmativo, qual seria então a vantagem de usar a relaxação RL^1?**

   Como $\mathcal{F}$ é o conjunto de todos os vetores de incidência que representam árvores geradoras mínimas de G, temos, por Darmann, Pferschy e Schauer (2009), que: $\mathcal{F} = \{x \in \mathbb{Z}^m_+ : \sum_{e\ inE} x_e = n - 1, \sum_{e \in E(V')} x_e \leq |V'| - 1 \forall \emptyset \neq V' \subseteq V, x_e \leq 1 \forall e \in E\}$

   Vamos considerar o conjunto $\mathcal{F}' = \{x \in \mathbb{R}^m_+ : \sum_{e\ inE} x_e = n - 1, \sum_{e \in E(V')} x_e \leq |V'| - 1 \forall \emptyset \neq V' \subseteq V, x_e \leq 1 \forall e \in E\}$. A matriz de restrições A que define $\mathcal{F}'$ é da forma $\binom{B}{I}$, onde B vem das restrições de conexidade e de eliminação de subciclos, e I vem das restrições de limitação superior. Porém, temos que B tem a propriedade dos 1's consecutivos, logo, pelo resultado teórico visto em aula (exercício 3.3 de Wolsey (1998)), temos que B é totalmente unimodular. Logo, pela Proposição 3.1 de Wolsey (1998), temos que A é totalmente unimodular. Então, pela Proposição 2.2 de Nemhauser e Wolsey (1988), temos que $\mathcal{F}'$ é inteiro. Portanto, pelo Corolário 6.6 de Nemhauser e Wolsey (1988), temos que a afirmação é verdadeira.

   A vantagem de usar a relaxação RL^1 consiste no fato de existirem algoritmos combinatórios polinimiais para resolvê-la, enquanto que resolver a relaxação linear LP do problema é impraticável, pois é necessário carregar na memória uma quantidade de restrições exponencial no tamanho da entrada.

4. **Em teoria, os limitantes duais gerados por uma das relaxações domina aqueles obtidos pela outra? Justifique.**

   Conforme visto em aula, temos que $z \geq w^k_{LD} \geq z_{LP}$, para $k = 1, 2$. Porém, vimos que $w^1_{LD} = z_{LP}$. Logo, temos que $w^2_{LD} \geq w^1_{LD}$. Portanto, os limitante duais gerados pela relaxação RL^2 dominam aqueles gerados pela relaxação RL^1.

5. **Descreva um algoritmo polinomial para resolver o problema lagrangiano ($z^2(u)$) da relaxação RL^2. Dê a complexidade de pior caso do seu**

algoritmo e argumente porque ele está correto.

Para cada aresta e de E, definimos $E_e = \{f \in E : \{e, f\} \in S_e, f \neq e\}$ como o subconjunto de arestas de E que conflitam com a aresta e. Um algoritmo que resolve o problema primal lagrangiano da relaxação RL^2 consiste em resolver o PAGM com custos lagrangianos no subgrafo $G[E \setminus E_{e^*}]$ induzido pelo conjunto de arestas $E \setminus E_{e^*}$ que não conflitam com a aresta e^* e subtrair o termo $\sum_{\{e,f\} \in S \setminus S_{e^*}} u_{ef}$ do valor ótimo obtido. Como $E \setminus E_{e^*} \subset E$ e $V(E \setminus E_{e^*}) \subseteq V$, temos que $|E \setminus E_{e^*}| < m$ e $|V(E \setminus E_{e^*})| \leq n$. Então, usando o algoritmo de Kruskal, obtemos a complexidade de pior caso de $O(m \log n)$.

Seja x a solução obtida pelo algoritmo descrito acima, temos que $x_f = 0, \forall f \in E_{e^*}$, pois o subgrafo $G[E \setminus E_{e^*}]$ não possui nenhuma aresta de E_{e^*}. Logo, temos que $x_{e^*} + x_f \leq 1, \forall \{e^*, f\} \in S_{e^*}$. Portanto, temos que x é uma solução viável para o problema primal lagangiano da relaxação RL^2.

Vamos mostrar que a solução x, que define a árvore T, é uma solução ótima por indução. Seja x^* uma solução ótima para o problema primal lagangiano da relaxação RL^2 e T^* a árvore definida por x^*. Se $x = x^*$, então x é uma solução ótima. Se $x \neq x^*$, então existe uma aresta e em $T^* \setminus T$ de custo mínimo. Temos que $T \cup \{e\}$ contém um ciclo C tal que toda aresta em C tem custo menor ou igual à c_e e existem alguma aresta f em C que não está em T^*. Seja $T' = T \setminus \{e\} \cup \{f\}$, temos que T' é uma árvore geradora de $G[E \setminus E_{e^*}]$ que tem mais arestas em comum com T^* do que T tinha e $\sum_{e \in T'} c_e x_e \geq \sum_{e \in T} c_e x_e$. Podemos refazer este processo com T' para encontrar uma árvore geradora T'' com mais arestas em comum com T^*. Por indução, podemos repetir este processo até obter T^*, daonde obtemos que:

$$\sum_{e \in T} c_e x_e \leq \sum_{e \in T'} c_e x_e \leq \sum_{e \in T''} c_e x_e \leq \ldots \leq \sum_{e \in T^*} c_e x_e.$$

Como T^* é uma árvore geradora mínima, então estas inequações devem ser atendidas na igualdade e concluímos que T é uma árvore geradora mínima de $G[E \setminus E_{e^*}]$ e, portanto, a solução x é uma solução ótima.

3 Pré-processamento

Com o intuito de melhorar o processo de solução, foi utilizado o algoritmo de pré-processamento descrito por Samer e Urrutia (2015). O algoritmo consiste em um processo iterativo de três fases, onde cada fase é executada enquanto a instância do problema for atualizada.

A primeira fase busca por arestas de corte em G, usando uma busca em profundidade. Enquanto o G for conexo, qualquer aresta de corte e é fixada na solução e as arestas que conflitam com e são removidas de G e seus respectivos pares conflitantes são removidos de S.

Na segunda fase, checamos se fixar uma aresta e na solução e remover as arestas que conflitam com e de G e, então, fixar as possíveis novas arestas de corte, nos leva a um grafo disconexo, então podemos remover e de G e seus conflitos correspondentes de S. Neste caso, retornamos à fase 1, pois G pode conter novas arestas de corte.

Na terceira fase, checamos se fixar na solução um par de arestas e e f, tal que e e f não conflitam entre si, e remover as arestas que conflitam com e e com f e fixar todas as novas arestas de corte nos leva a um grafo disconexo. Neste caso, o novo par conflitante $\{e, v\}$ é adicionado à S, e retornamos à segunda fase, para checar se é possível remover alguma aresta.

Como este processo pode demorar muito tempo, foi definido o parametro *preProcessingTimeLimit* que limita quantos segundos são gastos no pré-processamento.

4 Heurística Lagrangiana

Com o intuito de gerar soluções primais a partir das soluções duais obtidas pelas relaxações lagrangianas RL^k, com $k = 1, 2$, foi implementada a heurística lagrangiana descrita a seguir.

Dada uma solução dual T^D do problema e um subconjunto de arestas fixadas $F \subseteq T^D$ obtido pelo algoritmo de pré-processamento descrito na Seção 3, computamos o conjunto de arestas não fixadas conflitantes em T^D, dado por $E^R = \{e \in T^D : e \notin F, \exists \{e, f\} \in S$ para alguma $f \in T^D\}$. A seguir, removemos as arestas em E^R de T^D, obtendo a floresta $T' = T^D \setminus E^R$. Então, computamos o conjunto de arestas que não estão em T' e nem conflitam com nenhuma aresta de T', dado por $E^I = \{e \in E : e \notin T'$ e $\nexists \{e, f\} \in S \forall f \in T'\}$. Finalmente, obtemos um subconjunto de arestas T^P a partir de T' adicionando as arestas e em E^I que não conflitam com nenhuma aresta que já esteja em T^P e que conecte duas componentes conexas distintas de T^P. Neste último passo, as arestas em E^I são analisadas em ordem crescente de custo, de maneira análoga ao algoritmo de Kruskal. Se, ao final do processo, T^P for conexo, então obtemos uma solução primal para o problema.

A complexidade da minha implementação da heurística lagrangiana é dada por $O(p \log n + mn \log p)$. Assim como no caso do algoritmo de pré-processamento, foi definido o parâmetro *fixSolutionTimeLimit* para limitar quantos segundos são gastos na heurística lagrangiana.

5 Heurística Construtiva

Com o intuito de obter um limitante primal inicial para o MS, foi implementada a heurística construtiva apresentada por Zhang, Kabadi e Punnen (2011).

Esta heurística consiste em, enquanto $S \neq \emptyset$, escolher uma aresta e de E que aparece no maior número de pares conflitantes, removê-la de G e remover de S todos os pares conflitantes envolvendo e. Se, ao final fo procedimento, G ainda for conexo, então o custo de uma árvore geradora mínima de G é um limitante primal para o AGMCC.

Além desta heurítica construtiva, também é utilizada a heurística lagrangiana, descrita na Seção 4, a partir da solução dual obtida ao se resolver o PAGM em G (desconsiderando os conflitos) para obter um limitante primal inicial.

Assim como no caso do algoritmo de pré-processamento e da heurística lagrangiana, foi definido o parâmetro *constructiveHeuristicTimeLimit* para limitar quantos segundos são gastos na heurística construtiva.

6 Resultados

Nesta Seção são apresentados e analisados os resultados obtidos pelos experimentos computacionais. Os experimentos foram realizados em uma máquina com processador Intel® Pentium(R) CPU G4560 @ 3.50GHz x 4, com 8GB de memória RAM. Os valores dos parâmetros usados são apresentados na Tabela 6.1. Os parâmetros *totalTimeLimit*, *preProcessingTimeLimit*, *constructiveHeuristicTimeLimit* e *fixSolutionTimeLimit* limitam o tempo de execução, já os parâmetros π e N são usados para calcular o passo do MS, enquano que o parâmetro $min\pi$ é utilizado como um critério de parada (BEASLEY, 1993).

Tabela 6.1: Valores dos parâmetros usados em cada relaxação RL^k

Parâmetro	RL^1	RL^2
totalTimeLimit	90	90
preProcessingTimeLimit	15	15
constructiveHeuristicTimeLimit	15	15
fixSolutionTimeLimit	5	5
π	2.0	3.0
N	15	30
$min\pi$	0.05	0.05

Os melhores limitantes primais e duais encontrados pelas relaxações lagrangianas e o GAP de otimalidade são apresentados na Tabela 6.2. O GAP de otimalidade é definido como: $GAP = \frac{\overline{z} - \underline{z}}{\overline{z}}$. Como os custos das arestas dos grafos das instâncias de testes são todos inteiros, podemos arredondar para cima o limitante dual e calcular o GAP como: $GAP = \frac{\overline{z} - \lceil \underline{z} \rceil}{\overline{z}}$. Quando o GAP for igual a zero, isto é, quando $\overline{z} = \lceil \underline{z} \rceil$, temos que a solução encontrada é ótima. Logo, podemos parar o MS quando $\overline{z} - \underline{z} < 1$.

Pela Tabela 6.2 podemos ver que que a relaxação RL^1 conseguiu encontrar soluções primais viáveis para 21 instâncias, sendo que para 10 instâncias foi encontrada uma solução ótima, enquanto que a relaxação RL^2 conseguiu encontrar soluções primais viáveis para 20 instâncias, sendo que para 10 instâncias foi encontrada uma solução ótima. Porém, também podemos ver que a relaxação RL^2 obteve GAP's negativos para 3 instâncias, ou seja, obteve limitantes duais superiores aos limitantes primais para 3 instâncias. Disso podemos concluir que há um erro na minha implementação do algoritmo para resolver

o problema lagrangiano da relaxação RL^2. Contudo, mesmo com este erro, a relaxação lagrangiana RL^2 conseguiu obter bons resultados.

A Tabela 6.3 apresenta os tempos computacionais totais e os de obtenção de cada um dos melhores limitantes duais e primais para todas as instâncias de teste e relaxações. Analisando a Tabela 6.3 podemos ver que, em geral, os tempos de obtenção do melhor limitante primal na relaxação RL^1 são inferiores aos da relaxação RL^2 enquanto que os tempos de obtenção do melhor limitante dual na relaxação RL^2 são inferiores aos das relaxação RL^1.

A Figura 6.1 apresenta um gráfico mostrando a variação dos limitantes inferior e superior em função do número de iterações do MS para a instância *type1/z50-200-398.gcc* obtidos pela relaxação RL^1. Podemos ver que tanto o limitante dual quando o primal oscilam, mas tendem ao valor ótimo conforme o número de iterações cresce. Além disso, podemos ver uma melhora significativa do limitante dual por volta da iteração 20. Isto provavelmente deve-se à atualização do valor de π após N iterações sem melhoras, conforme descrito por BEASLEY.

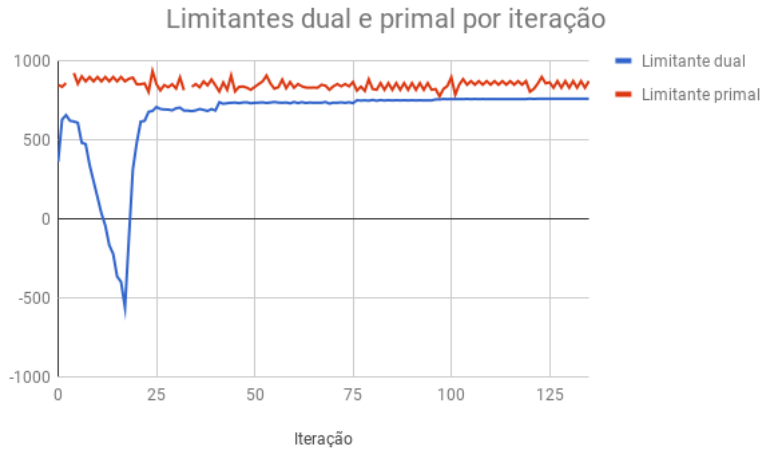

Figura 6.1: Variação dos limitantes dual e primal em função do número de iterações para a instância *type1/z50-200-398.gcc* obtidos pela relaxação RL^1.

Referências Bibliográficas

WOLSEY, L. *Integer Programming*. [S.l.]: Wiley-Interscience, 1998. (Wiley-Interscience Series in Discrete Mathematics and Optimization). ISBN 9780471283669.

CORMEN, T. H. et al. *Introduction to algorithms*. [S.l.]: MIT press, 2009.

DARMANN, A.; PFERSCHY, U.; SCHAUER, J. Determining a minimum spanning tree with disjunctive constraints. In: ROSSI, F.; TSOUKIAS, A. (Ed.). *Algorithmic*

Decision Theory. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 414–423. ISBN 978-3-642-04428-1.

NEMHAUSER, G. L.; WOLSEY, L. A. Integer and combinatorial optimization. Wiley-Interscience, 1988.

SAMER, P.; URRUTIA, S. A branch and cut algorithm for minimum spanning trees under conflict constraints. *Optimization Letters*, v. 9, n. 1, p. 41–55, Jan 2015. ISSN 1862-4480. Disponível em: <https://doi.org/10.1007/s11590-014-0750-x>.

ZHANG, R.; KABADI, S. N.; PUNNEN, A. P. The minimum spanning tree problem with conflict constraints and its variations. *Discrete Optimization*, v. 8, n. 2, p. 191–205, 2011. ISSN 1572-5286. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1572528610000551>.

BEASLEY, J. E. Modern heuristic techniques for combinatorial problems. In: REEVES, C. R. (Ed.). New York, NY, USA: John Wiley & Sons, Inc., 1993. cap. Lagrangian Relaxation, p. 243–303. ISBN 0-470-22079-1. Disponível em: <http://dl.acm.org/citation.cfm?id=166648.166660>.

Tabela 6.2: Limitantes primal ($\overline{z}$) e dual ($\underline{z}$) e GAP obtidos pelas relaxações lagrangianas RL^k.

Instância	RL^1			RL^2		
	$\overline{z}$	$\underline{z}$	GAP	$\overline{z}$	$\underline{z}$	GAP
type1/z50-200-199.gcc	708	704.402992	0.004237	708	704.056149	0.004237
type1/z50-200-398.gcc	775	759.812786	0.019355	794	779.524927	0.017632
type1/z50-200-597.gcc	1358	584.000000	0.569956	1072	584.000000	0.455224
type1/z50-200-995.gcc	∞	584.000000	NaN	∞	584.000000	NaN
type1/z100-300-448.gcc	4784	3125.000000	0.346781	4498	3125.000000	0.305247
type1/z100-300-897.gcc	∞	3132.000000	NaN	∞	3132.000000	NaN
type1/z100-500-1247.gcc	4282	4266.143677	0.003503	4304	4270.583522	0.007667
type1/z100-500-2495.gcc	∞	3241.000000	NaN	∞	3241.000000	NaN
type1/z100-500-3741.gcc	∞	3241.000000	NaN	∞	3241.000000	NaN
type1/z200-600-1797.gcc	∞	7386.000000	NaN	∞	7386.000000	NaN
type1/z200-800-3196.gcc	∞	11939.000000	NaN	∞	11939.000000	NaN
type2/z50-200-3903.gcc	1177	1088.917737	0.074766	1177	1431.376241	-0.216653
type2/z50-200-4877.gcc	∞	-∞	NaN	∞	-∞	NaN
type2/z50-200-5864.gcc	2350	2350.000000	0.000000	2350	2350.000000	0.000000
type2/z100-300-8609.gcc	7460	7460.000000	0.000000	7460	7460.000000	0.000000
type2/z100-300-10686.gcc	∞	-∞	NaN	∞	-∞	NaN
type2/z100-300-12761.gcc	8166	8166.000000	0.000000	8166	8166.000000	0.000000
type2/z100-500-24740.gcc	∞	3333.000000	NaN	∞	3333.000000	NaN
type2/z100-500-30886.gcc	∞	4243.000000	NaN	∞	4206.000000	NaN
type2/z100-500-36827.gcc	11637	11637.000000	0.000000	11637	11637.000000	0.000000
type2/z200-400-13660.gcc	17728	17728.000000	0.000000	17728	17728.000000	0.000000
type2/z200-400-17089.gcc	18617	18617.000000	0.000000	18617	18617.000000	0.000000
type2/z200-400-20469.gcc	19140	19140.000000	0.000000	19140	19140.000000	0.000000
type2/z200-600-34504.gcc	20716	10176.000000	0.508785	∞	118177514148	NaN
type2/z200-600-42860.gcc	18025	9881.000000	0.451817	18025	10444.91773	0.420527
type2/z200-600-50984.gcc	20864	10261.000000	0.508196	20864	20864.00008	-0.000048
type2/z200-800-62625.gcc	∞	13263.000000	NaN	∞	13263.000000	NaN
type2/z200-800-78387.gcc	∞	14727.000000	NaN	∞	5219674241	NaN
type2/z200-800-93978.gcc	∞	14649.000000	NaN	∞	14649.000000	NaN
type2/z300-600-31000.gcc	43721	43721.000000	0.000000	43721	43721.000000	0.000000
type2/z300-600-38216.gcc	44267	44267.000000	0.000000	44267	44267.000000	0.000000
type2/z300-600-45310.gcc	43071	43071.000000	0.000000	43071	43071.000000	0.000000
type2/z300-800-59600.gcc	43125	25258.000000	0.414307	43125	43125.00007	-0.000023
type2/z300-800-74500.gcc	42292	27388.000000	0.352407	∞	40224582637	NaN
type2/z300-800-89300.gcc	∞	26956.000000	NaN	44114	26956.000000	0.388947
type2/z300-1000-96590.gcc	∞	27733.000000	NaN	∞	27733.000000	NaN
type2/z300-1000-120500.gcc	∞	29134.000000	NaN	∞	756952651.7	NaN
type2/z300-1000-144090.gcc	∞	28138.000000	NaN	∞	28138.000000	NaN

Tabela 6.3: Tempo computacional total (t) e tempo de obtenção do melhor limitante primal ($t(\overline{z})$) e do melhor limitante dual ($t(\underline{z})$) em segundos, obtidos pelas relaxações lagrangianas RL^k.

Instância	RL^1			RL^2		
	t	$t(\overline{z})$	$t(\underline{z})$	t	$t(\overline{z})$	$t(\underline{z})$
type1/z50-200-199.out	15	15	15	15	15	15
type1/z50-200-398.out	15	15	15	16	15	16
type1/z50-200-597.out	15	15	15	16	15	15
type1/z50-200-995.out	16	-	15	17	-	15
type1/z100-300-448.out	16	16	15	17	16	15
type1/z100-300-897.out	17	-	15	18	-	15
type1/z100-500-1247.out	28	28	27	33	25	32
type1/z100-500-2495.out	31	-	15	35	-	15
type1/z100-500-3741.out	38	-	15	44	-	15
type1/z200-600-1797.out	29	-	15	33	-	15
type1/z200-800-3196.out	42	-	15	49	-	15
type2/z50-200-3903.out	21	17	20	22	17	18
type2/z50-200-4877.out	90	-	-	90	-	-
type2/z50-200-5864.out	90	-	-	90	-	-
type2/z100-300-8609.out	2	2	2	2	2	2
type2/z100-300-10686.out	90	-	-	90	-	-
type2/z100-300-12761.out	2	2	2	2	2	2
type2/z100-500-24740.out	85	-	15	90	-	15
type2/z100-500-30886.out	44	-	15	50	-	15
type2/z100-500-36827.out	16	15	16	15	15	15
type2/z200-400-13660.out	30	15	30	30	15	30
type2/z200-400-17089.out	30	15	30	31	15	31
type2/z200-400-20469.out	30	30	30	30	30	30
type2/z200-600-34504.out	59	39	15	90	-	89
type2/z200-600-42860.out	73	44	15	90	42	90
type2/z200-600-50984.out	84	49	15	60	42	60
type2/z200-800-62625.out	90	-	15	90	-	15
type2/z200-800-78387.out	91	-	15	90	-	83
type2/z200-800-93978.out	90	-	15	91	-	15
type2/z300-600-31000.out	32	15	32	32	15	32
type2/z300-600-38216.out	35	15	35	36	15	36
type2/z300-600-45310.out	31	16	16	31	16	16
type2/z300-800-59600.out	90	74	15	61	51	61
type2/z300-800-74500.out	90	63	16	90	-	90
type2/z300-800-89300.out	91	-	15	91	67	15
type2/z300-1000-96590.out	93	-	15	91	-	15
type2/z300-1000-120500.out	94	-	15	90	-	87
type2/z300-1000-144090.out	94	-	15	91	-	15