NHH

INSTITUTT FOR FORETAKSØKONOMI

DEPARTMENT OF BUSINESS AND MANAGEMENT SCIENCE

FOR 1 2014

ISSN: 1500-4066

January 2014

Discussion paper

# A New Semi-Lagrangean Relaxation for the K-Cardinality Assignment Problem

BY

**Ivan Belik** AND **Kurt Jörnsten**

Norges Handelshøyskole

NORWEGIAN SCHOOL OF ECONOMICS

# A NEW SEMI-LAGRANGEAN RELAXATION FOR THE *K*-CARDINALITY ASSIGNMENT PROBLEM

**Ivan Belik[a], Kurt Jörnsten[b]**

[a, b] *Norwegian School of Economics, Helleveien 30, 5045 Bergen, Norway*
*Email addresses: [a] Ivan.Belik@nhh.no, [b] Kurt.Jornsten@nhh.no*

**Abstract**

*Recently Beltrán-Royo, Vial & Alonso-Ayuso (2012) presented a semi-Lagrangean relaxation for the classical p-median location problem and for the incapacitated facility location problem. The results, obtained using the semi-Lagrangean relaxation approach, were quite impressive. In this paper we use a semi-Lagrangean relaxation to obtain an efficient solution method for the k-cardinality assignment problem. The method has only one semi-Lagrangean multiplier that can only take on a limited number of values, making the search for the optimal multiplier easy. Since the semi-Lagrangean relaxation closes the duality gap, this leads to an extremely reliable and easily implementable method for finding k-cardinality assignments in large-scale cases. The method is computationally tested on the examples commonly used in the literature.*

**Keywords: *k*-cardinality assignment, Lagrangean Relaxation, Mathematical Programming**

## 1. Introduction

The assignment problem is one of the basic combinatorial optimization problems in mathematical optimization and operations research. Many practical real-world problems are based on solving the *k*-cardinality assignment problem. This is due to the fact that in many cases it is required to do *k* specific assignments (Volgenant, 2004; Bai, 2009). One of the trivial examples is when it is necessary to assign workers and machines. Many alternatives exist when we have to assign only a subset of workers to the specific number of machines (Dell'Amico & Martello, 1997; Dell'Amico, Lodi, & Martello, 2001).

The *k*-cardinality assignment problem is useful for solving problems that are even more complicated (Bruglieri, Ehrgott, Hamacher, & Maffioli, 2006). Dell'Amico & Martello (1997) describe the Satellite-Switched Time-Division Multiple Access (SS/TDMA) time slot assignment problem. According to the problem, there exist *m* earth stations that has to transmit the information to another *n* earth stations. All interconnections are operated by the $k \times k$ switch that is located on the satellite. The non-negative $m \times n$ matrix contains the weights that corrspond to the information slot to be sent from station *i* to station *j* through the given $k \times k$ switch. The

general solution to the *k*-cardinality assignment problem can be adapted to the given time assignment problem very effectively.

In this paper we apply the Semi-Lagrangean relaxation (Beltran, Tadonki & Vial, 2004) to solve the *k*-cardinality assignment problem that has many applications in the real-world situations.

In Sections 2 and 3 we give the theoretical base for our approach. Specifically, we show the integer programming formulation and explain the semi-Lagrangean relaxation procedure for the *k*-cardinality assignment problem. Section 4 explains our approach based on the small examples. Section 5 is devoted to the description of two selection procedures of the initial multiplier value applied in the semi-Lagrangean relaxation procedure. The computational results over the large-scale *k*-cardinality assignment problems are presented in Section 6 accompanied with the detailed information about each test.

## 2. The integer programming formulation of the *k*-cardinality assignment problem

The integer programming formulation of the *k*-cardinality tree problem is as follows:

$$Min \sum_i \sum_j c_{ij}\, x_{ij} \qquad (1)$$

*Subject to:*

$$\sum_{ij} x_{ij} = k \qquad (2)$$

$$\sum_i x_{ij} \leq 1 \qquad \forall\, j \in J \qquad (3)$$

$$\sum_j x_{ij} \leq 1 \qquad \forall\, i \in I \qquad (4)$$

$$x_{ij} \in \{0,1\} \qquad \forall\, i \in I, j \in J \qquad (5)$$

## 3. Semi-Lagrangean Relaxation

The semi-Lagrangean approach builds upon the well-known Lagrangean relaxation, but with the difference that when having equality constraint, the constraint is divided into two inequalities, namely a "greater than or equal to" inequality and a "less than or equal to" inequality. The former is relaxed and added to the objective function, while the latter is left as an inequality constraint in the sub-problem (Beltran, Tadonki, & Vial, 2006). Mathematically, if we have a minimisation problem of the following type:

$$z^* = \min_x \{c^T x | Ax = b; x \in S \coloneqq X \cap \mathbb{N}^n\}$$

then the semi-Lagrangean function is written as

$$z_{SLR} = \max_\lambda \mathcal{L}_{SLR}(\lambda) = \max_\lambda \{b^T \lambda + \min_x \{(c - A^T \lambda)^T x | Ax \leq b; x \in S\}\}$$

It is proved by Beltrán-Royo et al. (2012) that the semi-Lagrangean relaxation closes the duality gap. The easiest way to see this is that the relaxation is the result of the intersection between the following two polytopes:

$$\mathrm{Conv}\{\min_{x}(c^{T}x|Ax \leq b, x \in S)\} \cap \mathrm{Conv}\{Ax \geq b\},$$

which is obviously equal to:

$$\mathrm{Conv}\Big\{\{\min(c^{T}x|Ax = b, x \in S)\}\Big\}$$

The following theorem from Beltrán-Royo et al. (2012) gives the properties of the semi-Lagrangean dual theorem:

1. The semi-Lagrangean dual $\mathcal{L}(u)$ is concave and $b-Ax(u)$ is a subgradient at $u$
2. $\mathcal{L}(u)$ is monotone and $\mathcal{L}(u') \geq \mathcal{L}(u)$ if $u' \geq u$ with strict inequality if $u' > u$ and $u \notin U^{*}$
3. $U^{*} = U^{*} \cup \mathbb{R}_{+}^{m}$; thus $U^{*}$ is unbounded
4. If $x(u)$ is such that $Ax(u)=b$, then $u \in U^{*}$ and $x(u)$ solves the original problem
5. Conversely, if $u \in int(U^{*})$, then any minimizer $x(u)$ is optimal in the original problem
6. The semi-Lagrangean relaxation closes the duality gap

However there are some difficulties involved in calculating the optimal semi-Lagrangean multipliers especially in the multi-dimensional case. The main problem is that the optimal semi-Lagrangean prices are non-unique (in the multi-dimensional case). Moreover, for large enough multipliers $u$ $x(u)$ will be a solution to the original problem and the relaxed problem is basically identical to the original problem. Also we are not looking for a maximum of the concave function $\mathcal{L}(u)$ rather we are looking for the "minimal" multiplier values, for which $\mathcal{L}(u)$ reaches its maximal value.

Applying the semi-Lagrangean relaxation to the $k$-cardinality problem, relaxing the single equality constraint yields the following dual problem:

*Max SL(u) subject to u≥0,*

*where $\mathcal{L}(u)$ is defined by the following optimization problem:*

$$Min \sum_i \sum_j c_{ij} x_{ij} - u\left(\sum_{ij} x_{ij} - k\right) \quad (6)$$

*Subject to:*

$$\sum_{ij} x_{ij} \leq k \quad (7)$$

$$\sum_i x_{ij} \leq 1 \qquad \forall j \in J \quad (8)$$

$$\sum_j x_{ij} \leq 1 \qquad \forall i \in I \quad (9)$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J \quad (10)$$

We have only one semi-Lagrangean multiplier. This means that any one-dimensional search procedure can be used. In addition, the optimal semi-Lagrangean price has a meaningful economic interpretation. The optimal semi-Lagrangean price $u^*$ is the price for which an assignment of cardinality $k$ is obtained. Hence, the optimal semi-Lagrangean price can be regarded as a market price in order to get $k$ objects assigned. It should also be noted that it is easy to get an initial estimate on $u$ since we know that at least $k$ variables has to be in the $k$-cardinality assignment. Hence, the minimal price is the price that guarantees that at least $k$ rows and $k$ columns has a non-positive edge cost. Also the number of possible prices is the number of different weights in the bipartite graph since no other price will give a result that can be better than when restricting the multiplier values to weights that exists in the bipartite graphs cost matrix. Also, note that in the subproblems the only edges that can be selected are the edges with the negative weights.

It should be noted that if we are not only looking for the optimal solution but also the optimal semi-Lagrangean "market" price we have to extend the search for multiplier values to values that do not necessarily appear in the initial cost matrix.

## 4. Illustrative Examples

First, we illustrate the approach on a small example.

*Example 1*. Let m = 3, n = 3, $k$ = 2, and

$$C=\begin{pmatrix}1 & 4 & 6\\ 4 & 9 & 9\\ 6 & 9 & 8\end{pmatrix}$$

Here it is obvious that the optimal solution consists of two assignments 1-2 and 2-1 giving the optimal value "8".

If we use a conservative approach to select the initial multiplier value we get the value $u$=4. The resulting cost matrix for the semi-Lagrangean subproblem is then:

$$\begin{pmatrix}-3 & 0 & X\\ 0 & X & X\\ X & X & X\end{pmatrix},$$

where all positive (i.e. not allowable assignments) are marked by X.

This will give us as solution to the semi-Lagrangean subproblem the assignment 1-1 and a lower bound of -3+8=5.

Increasing the semi-Lagrangean multiplier value to $7+\varepsilon$ we get the following cost matrix for the subproblem:

$$\begin{pmatrix} -6-\varepsilon & -3-\varepsilon & -1-\varepsilon \\ -3-\varepsilon & X & X \\ -1-\varepsilon & X & X \end{pmatrix}$$

The optimal assignment is 1-2 and 2-1 and it gives the lower bound: $-6-2\varepsilon+14+2\varepsilon=8$. Hence, optimality is proved. Note that the optimal semi-Lagrangean multiplier value is not a cost that appears in the original cost matrix.

An alternative to generate the initial multiplier value is to use a greedy heuristic and to choose the initial multiplier value as the most costly assignment in the feasible solution. For this small example it gives us the initial multiplier value *u*=8 and the subproblem cost matrix:

$$\begin{pmatrix} -7 & -4 & -1 \\ -4 & X & X \\ -1 & X & X \end{pmatrix}$$

with the optimal assignment 1-2 and 2-1 and the lower bound -8+16=8. The optimality is proved. However, as can be seen, the optimal semi-Lagrangean multiplier value is less than "8".

*Example 2.*

Here we illustrate the procedure for a 15 by 15 example with *k* ranging from 5 to 15.

Table 1. Illustrative example: cost matrix

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 17 | 21 | 22 | 18 | 24 | 15 | 20 | 18 | 19 | 18 | 16 | 22 | 24 | 24 | 16 |
| 23 | 16 | 21 | 16 | 17 | 16 | 19 | 25 | 18 | 21 | 17 | 15 | 25 | 17 | 24 |
| 16 | 20 | 16 | 25 | 24 | 16 | 17 | 19 | 19 | 18 | 20 | 16 | 17 | 21 | 24 |
| 19 | 19 | 22 | 22 | 20 | 16 | 19 | 17 | 21 | 19 | 25 | 23 | 25 | 25 | 25 |
| 18 | 19 | 15 | 15 | 21 | 25 | 16 | 16 | 23 | 15 | 22 | 17 | 19 | 22 | 24 |
| 8 | 15 | 14 | 23 | 8 | 16 | 8 | 25 | 9 | 17 | 25 | 15 | 10 | 8 | 24 |
| 15 | 7 | 23 | 22 | 11 | 11 | 12 | 10 | 17 | 16 | 7 | 16 | 10 | 18 | 22 |
| 21 | 20 | 6 | 22 | 24 | 10 | 24 | 9 | 21 | 14 | 11 | 14 | 11 | 19 | 16 |
| 20 | 11 | 8 | 14 | 9 | 5 | 6 | 19 | 19 | 7 | 6 | 6 | 13 | 9 | 18 |
| 8 | 13 | 13 | 13 | 10 | 20 | 25 | 16 | 16 | 17 | 10 | 10 | 5 | 12 | 23 |
| 19 | 23 | 24 | 20 | 20 | 25 | 16 | 21 | 24 | 15 | 17 | 17 | 20 | 20 | 20 |
| 25 | 24 | 16 | 21 | 19 | 17 | 17 | 19 | 23 | 21 | 21 | 23 | 20 | 15 | 16 |
| 16 | 21 | 25 | 22 | 24 | 24 | 16 | 17 | 15 | 18 | 15 | 17 | 18 | 24 | 18 |
| 25 | 24 | 18 | 19 | 15 | 18 | 20 | 22 | 23 | 18 | 16 | 19 | 17 | 15 | 22 |
| 25 | 19 | 21 | 22 | 20 | 15 | 20 | 19 | 18 | 18 | 17 | 23 | 17 | 25 | 25 |

The results for the current illustrative example are presented in Table 2 and the detailed results including the number of iterations and lower bounds (LBDs) for each subproblem are presented in Appendix A.

Table 2. Illustrative example: computational results

| Test # | *k* | *u** | Solution | R,% |
|---|---|---|---|---|
| 1 | 5 | 8 | 31 | 93.33 |
| 2 | 6 | 15 | 46 | 75.11 |
| 3 | 7 | 15 | 61 | 75.11 |
| 4 | 8 | 15 | 76 | 75.11 |
| 5 | 9 | 15 | 91 | 75.11 |
| 6 | 10 | 15 | 106 | 75.11 |
| 7 | 11 | 15 | 121 | 75.11 |
| 8 | 12 | 16 | 137 | 64.44 |
| 9 | 13 | 16 | 153 | 64.44 |
| 10 | 14 | 16 | 169 | 64.44 |
| 11 | 15 | 17 | 186 | 55.55 |

As it is shown in Table 2 and in Appendix A, we are running 11 experiments. Column "*k*" shows the *k*-cardinality assignment for the corresponding problem. Column "$u^*$" corresponds to the optimal semi-Lagrangean multiplier. "Iteration #" is the number of subproblems we had to solve before we find the optimal solution. "LBD" is a lower bound value for the corresponding problem. "Solution" shows the optimal result for the given *k*-cardinality assignment problem, and "R, %" corresponds to the problem reduction obtained, i.e. the number of variables in the subproblem for the optimal multiplier value versus the total number of variables in original problem. According to the results represented in Table 2, we have obtained the problem reduction in the range between 55.55% and 93.33%.

## 5. The Selection of the Initial Multiplier Value

When selecting the initial multiplier value we have used two different approaches:

I. A conservative approach in which the initial multiplier value *u* is selected as the maximal of the *k* smallest cost coefficient values in either the columns or the rows. Choosing this conservative value as the initial multiplier value will normally result in a large problem reduction. However it is likely that when the semi-Lagrangean subproblem is solved then it will result in a solution with less than *k* assignments. This will lead to a necessary increase in the multiplier value and the need for more iterations.

II. Use a greedy algorithm to find a feasible solution to the *k*-cardinality assignment problem. Select the initial multiplier value to be the assignment in the greedy solution with the highest cost. Using this initialization procedure leads to a smaller problem reduction in the first iteration than procedure I.

***Approach I: illustrative example***

We consider the cost matrix presented in Table 1 as an illustrative example with m=15, n=15 and *k*=12. Following the algorithm, the maximal of the *k* smallest cost coefficient values in either the columns or the rows is equal to "15".

*Iteration 1*

For *u*=15 the subproblem has the following cost matrix represented in Table 3.

Table 3. Iteration 1: resulting cost matrix

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| X | X | X | X | X | 0 | X | X | X | X | X | X | X | X | X |
| X | X | X | X | X | X | X | X | X | X | X | 0 | X | X | X |
| X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |
| X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |
| X | X | 0 | 0 | X | X | X | X | X | 0 | X | X | X | X | X |
| -7 | 0 | -1 | X | -7 | X | -7 | X | -6 | X | X | 0 | -5 | -7 | X |
| 0 | -8 | X | X | -4 | -4 | -3 | -5 | X | X | -8 | X | -5 | X | X |
| X | X | -9 | X | X | -5 | X | -6 | X | -1 | -4 | -1 | -4 | X | X |
| X | -4 | -7 | -1 | -6 | -10 | -9 | X | X | -8 | -9 | -9 | -2 | -6 | X |
| -7 | -2 | -7 | -2 | -5 | X | X | X | X | X | -5 | -5 | -10 | -3 | X |
| X | X | X | X | X | X | X | X | X | 0 | X | X | X | X | X |
| X | X | X | X | X | X | X | X | X | X | X | X | X | 0 | X |
| X | X | X | X | X | X | X | X | 0 | X | 0 | X | X | X | X |
| X | X | X | X | 0 | X | X | X | X | X | X | X | X | 0 | X |
| X | X | X | X | X | 0 | X | X | X | X | X | X | X | X | X |

On this iteration we get the solution with the following "row-column" assignments: 2-12, 5-4, 6-7, 7-2, 8-3, 9-6, 10-13, 11-10, 12-14, 13-9 and 14-5 of cardinality "11" with the objective function value equal to "-44". Hence, the lower bound is 180-44=136. Since we have "k-1" assignments in the solution for the current subproblem, we have to update to the next *u*.

*Iteration 2*

Assigning *u*=16 we get the resulting cost matrix presented in Table 4.

The optimal solution to the subproblem is twelve "row-column" assignments 1-15, 2-12, 5-4, 6-7, 7-2, 8-3, 9-6, 10-13, 11-10, 12-14, 13-9 and 14-15 with the objective function value "-55". Since we have LBD to be equal to "137", and this is the value of the feasible solution (i.e., the upper bound is equal to "137"), we have found the optimal solution corresponding to the semi-Lagrangean multiplier *u*=16.

Thus, the optimal solution is obtained by the conservative approach in two iterations.

Table 4. Iteration 2: resulting cost matrix

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| X | X | X | X | X | -1 | X | X | X | X | 0 | X | X | X | 0 |
| X | 0 | X | 0 | X | 0 | X | X | X | X | X | -1 | X | X | X |
| 0 | X | 0 | X | X | 0 | X | X | X | X | X | 0 | X | X | X |
| X | X | X | X | X | 0 | X | X | X | X | X | X | X | X | X |
| X | X | -1 | -1 | X | X | 0 | 0 | X | -1 | X | X | X | X | X |
| -8 | -1 | -2 | X | -8 | 0 | -8 | X | -7 | X | X | -1 | -6 | -8 | X |
| -1 | -9 | X | X | -5 | -5 | -4 | -6 | X | 0 | -9 | 0 | -6 | X | X |
| X | X | -10 | X | X | -6 | X | -7 | X | -2 | -5 | -2 | -5 | X | 0 |
| X | -5 | -8 | -2 | -7 | -11 | -10 | X | X | -9 | -10 | -10 | -3 | -7 | X |
| -8 | -3 | -8 | -3 | -6 | X | X | 0 | 0 | X | -6 | -6 | -11 | -4 | X |
| X | X | X | X | X | X | 0 | X | X | -1 | X | X | X | X | X |
| X | X | 0 | X | X | X | X | X | X | X | X | X | X | -1 | 0 |
| 0 | X | X | X | X | X | 0 | X | -1 | X | -1 | X | X | X | X |
| X | X | X | X | -1 | X | X | X | X | X | 0 | X | X | -1 | X |
| X | X | X | X | X | -1 | X | X | X | X | X | X | X | X | X |

***Approach II: Illustrative example***

We consider the illustrative example once more.

Using a simple greedy heuristic we get the following “row-column” assignments:

| | | | |
|---|---|---|---|
| 2-12 | with value 15; | 10-13 | with value 5; |
| 3-1 | with value 16; | 11-10 | with value 15; |
| 5-4 | with value 15; | 12-14 | with value 15; |
| 6-7 | with value 8; | 13-11 | with value 15; |
| 8-3 | with value 6; | 14-5 | with value 15. |
| 9-6 | with value 5; | | |

The greedy heuristic assignments are reflected in Table 5.

Table 5. Greedy heuristic assignments

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Since the assignment 3-1 with value "16" is the worst in the solution, we choose it as the semi-Lagrangean multiplier $u$=16. Solving the semi-Lagrangean subproblem, we get the objective function value "-55" and the lower bound of 192-55=137. Since this is the value of the greedy solution, optimality has been verified with value "137" in one iteration. For the 15×15 illustrative example (Table 1) the greedy heuristic yields the optimal solution for all $k$.

Characterizing the given approaches we should say that the conservative approach always gives us the smallest possible $u$. When using a greedy approach, the $u$ that is calculated, will be larger than or equal to this smallest possible $u$ value obtained by the conservative approach. It might not be the optimal solution, but selecting $u$ to be equal to the largest cost assignment in the greedy solution gives us a good initial $u$, for which we know that there exist a feasible solution. Hence, this $u$ gives us a semi-Lagrangean subproblem that (when it is solved) leads to one of the following cases. First, it provides us with the optimal solution to the original problem and with a proof for the optimality based on the bounds. Second, it gives us the solution with less than $k$ assignment, and in this case we have to increase $u$ until we get the optimal solution.

In Appendices A-I we have reported the number of iterations for both approaches (i.e., conservative and greedy). In many tests the number of iterations to obtain the optimal solution by the greedy approach is less than or equal to the number of iterations produced by the conservative one. It makes the selection of the initial multiplier value using the greedy approach more preferable in terms of time required to find the optimal solution.

## 6. Computational Results

We have tested the approach for various values of $k$ on the assignment problems in the OR-library, (*OR-library: Assignment problem,* Beasley, J. E. (2013)), where the number of decision variables is in the range between 10,000 and 650,000 variables (see Table 6).

Table 6. Assignment problem tests

| Test title | Number of decision variables | | Test title | Number of decision variables |
|---|---|---|---|---|
| assign100 | 10,000 | | assign500 | 250,000 |
| assign200 | 40,000 | | assign600 | 360,000 |
| assign300 | 90,000 | | assign700 | 490,000 |
| assign400 | 160,000 | | assign800 | 640,000 |

The results are presented in Table 7 and the detailed results, including all iterations with corresponding LBDs, are presented in Appendix B-I, respectively.

Table 7. Assignment problems: computational results

| No. | $k$ | $u^*$ | Sol. | No. | $k$ | $u^*$ | Sol. | No. | $k$ | $u^*$ | Sol. | No. | $k$ | $u^*$ | Sol. | No. | $k$ | $u^*$ | Sol. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **assign100** | | | | **assign200** | | | | 21 | 200 | 2 | 390 | **assign500** | | | | 9 | 460 | 2 | 881 |
| 1 | 20 | 2 | 39 | 1 | 50 | 2 | 98 | 22 | 205 | 2 | 400 | 1 | 100 | 2 | 179 | 10 | 480 | 2 | 921 |
| 2 | 50 | 2 | 99 | 2 | 55 | 2 | 108 | 23 | 210 | 2 | 410 | 2 | 120 | 2 | 219 | 11 | 500 | 2 | 961 |
| 3 | 51 | 2 | 101 | 3 | 60 | 2 | 118 | 24 | 215 | 2 | 420 | 3 | 140 | 2 | 259 | 12 | 520 | 2 | 1001 |
| 4 | 52 | 2 | 103 | 4 | 65 | 2 | 128 | 25 | 220 | 2 | 430 | 4 | 160 | 2 | 299 | 13 | 540 | 2 | 1041 |
| 5 | 53 | 2 | 105 | 5 | 70 | 2 | 138 | 26 | 225 | 2 | 440 | 5 | 180 | 2 | 339 | 14 | 560 | 2 | 1081 |
| 6 | 54 | 2 | 107 | 6 | 75 | 2 | 148 | 27 | 230 | 2 | 450 | 6 | 200 | 2 | 379 | 15 | 580 | 2 | 1121 |
| 7 | 55 | 3 | 110 | 7 | 80 | 2 | 158 | 28 | 235 | 2 | 460 | 7 | 220 | 2 | 419 | 16 | 600 | 2 | 1161 |
| 8 | 56 | 3 | 113 | 8 | 85 | 2 | 168 | 29 | 240 | 2 | 470 | 8 | 240 | 2 | 459 | 17 | 620 | 2 | 1201 |
| 9 | 57 | 3 | 116 | 9 | 90 | 2 | 178 | 30 | 245 | 2 | 480 | 9 | 260 | 2 | 499 | 18 | 640 | 2 | 1241 |
| 10 | 58 | 3 | 119 | 10 | 95 | 2 | 188 | 31 | 250 | 2 | 490 | 10 | 280 | 2 | 539 | 19 | 660 | 2 | 1281 |
| 11 | 59 | 3 | 122 | 11 | 100 | 2 | 198 | 32 | 255 | 2 | 500 | 11 | 300 | 2 | 579 | 20 | 680 | 2 | 1321 |
| 12 | 60 | 3 | 125 | 12 | 105 | 2 | 208 | 33 | 260 | 2 | 510 | 12 | 320 | 2 | 619 | 21 | 700 | 3 | 1362 |
| 13 | 61 | 3 | 128 | 13 | 110 | 2 | 218 | 34 | 265 | 2 | 520 | 13 | 340 | 2 | 659 | **assign800** | | | |
| 14 | 62 | 3 | 131 | 14 | 115 | 2 | 228 | 35 | 270 | 2 | 530 | 14 | 360 | 2 | 699 | 1 | 100 | 2 | 151 |
| 15 | 63 | 3 | 134 | 15 | 120 | 2 | 238 | 36 | 275 | 2 | 540 | 15 | 380 | 2 | 739 | 2 | 125 | 2 | 201 |
| 16 | 64 | 3 | 137 | 16 | 125 | 2 | 248 | 37 | 280 | 3 | 555 | 16 | 400 | 2 | 779 | 3 | 150 | 2 | 251 |
| 17 | 65 | 3 | 140 | 17 | 130 | 2 | 258 | 38 | 285 | 3 | 570 | 17 | 420 | 2 | 819 | 4 | 175 | 2 | 301 |
| 18 | 66 | 3 | 143 | 18 | 135 | 2 | 268 | 39 | 290 | 3 | 585 | 18 | 440 | 2 | 859 | 5 | 200 | 2 | 351 |
| 19 | 67 | 3 | 146 | 19 | 140 | 2 | 278 | 40 | 295 | 4 | 604 | 19 | 460 | 2 | 899 | 6 | 225 | 2 | 401 |
| 20 | 68 | 3 | 149 | 20 | 145 | 2 | 288 | 41 | 300 | 5 | 626 | 20 | 480 | 2 | 939 | 7 | 250 | 2 | 451 |
| 21 | 69 | 3 | 152 | 21 | 150 | 2 | 298 | **assign400** | | | | 21 | 500 | 5 | 991 | 8 | 275 | 2 | 501 |
| 22 | 70 | 3 | 155 | 22 | 155 | 3 | 309 | 1 | 100 | 2 | 184 | **assign600** | | | | 9 | 300 | 2 | 551 |
| 23 | 71 | 3 | 158 | 23 | 160 | 3 | 324 | 2 | 110 | 2 | 204 | 1 | 200 | 2 | 375 | 10 | 325 | 2 | 601 |
| 24 | 72 | 3 | 161 | 24 | 165 | 3 | 339 | 3 | 120 | 2 | 224 | 2 | 220 | 2 | 415 | 11 | 350 | 2 | 651 |
| 25 | 73 | 3 | 164 | 25 | 170 | 3 | 354 | 4 | 130 | 2 | 244 | 3 | 240 | 2 | 455 | 12 | 375 | 2 | 701 |
| 26 | 74 | 4 | 168 | 26 | 175 | 3 | 369 | 5 | 140 | 2 | 264 | 4 | 260 | 2 | 495 | 13 | 400 | 2 | 751 |
| 27 | 75 | 4 | 172 | 27 | 180 | 3 | 384 | 6 | 150 | 2 | 284 | 5 | 280 | 2 | 535 | 14 | 425 | 2 | 801 |
| 28 | 76 | 4 | 176 | 28 | 185 | 4 | 401 | 7 | 160 | 2 | 304 | 6 | 300 | 2 | 575 | 15 | 450 | 2 | 851 |
| 29 | 77 | 4 | 180 | 29 | 190 | 4 | 421 | 8 | 170 | 2 | 324 | 7 | 320 | 2 | 615 | 16 | 475 | 2 | 901 |
| 30 | 78 | 4 | 184 | 30 | 195 | 5 | 444 | 9 | 180 | 2 | 344 | 8 | 340 | 2 | 655 | 17 | 500 | 2 | 951 |
| 31 | 79 | 4 | 188 | 31 | 200 | 9 | 475 | 10 | 190 | 2 | 364 | 9 | 360 | 2 | 695 | 18 | 525 | 2 | 1001 |
| 32 | 80 | 4 | 192 | **assign300** | | | | 11 | 200 | 2 | 384 | 10 | 380 | 2 | 735 | 19 | 550 | 2 | 1051 |
| 33 | 81 | 4 | 196 | 1 | 100 | 2 | 190 | 12 | 210 | 2 | 404 | 11 | 400 | 2 | 775 | 20 | 575 | 2 | 1101 |
| 34 | 82 | 4 | 200 | 2 | 105 | 2 | 200 | 13 | 220 | 2 | 424 | 12 | 420 | 2 | 815 | 21 | 600 | 2 | 1151 |
| 35 | 83 | 4 | 204 | 3 | 110 | 2 | 210 | 14 | 230 | 2 | 444 | 13 | 440 | 2 | 855 | 22 | 625 | 2 | 1201 |
| 36 | 84 | 4 | 208 | 4 | 115 | 2 | 220 | 15 | 240 | 2 | 464 | 14 | 460 | 2 | 895 | 23 | 650 | 2 | 1251 |
| 37 | 85 | 5 | 213 | 5 | 120 | 2 | 230 | 16 | 250 | 2 | 484 | 15 | 480 | 2 | 935 | 24 | 675 | 2 | 1301 |
| 38 | 86 | 5 | 218 | 6 | 125 | 2 | 240 | 17 | 260 | 2 | 504 | 16 | 500 | 2 | 975 | 25 | 700 | 2 | 1351 |
| 39 | 87 | 5 | 223 | 7 | 130 | 2 | 250 | 18 | 270 | 2 | 524 | 17 | 520 | 2 | 1015 | 26 | 725 | 2 | 1401 |
| 40 | 88 | 5 | 228 | 8 | 135 | 2 | 260 | 19 | 280 | 2 | 544 | 18 | 540 | 2 | 1055 | 27 | 750 | 2 | 1451 |
| 41 | 89 | 5 | 233 | 9 | 140 | 2 | 270 | 20 | 290 | 2 | 564 | 19 | 560 | 2 | 1095 | 28 | 775 | 2 | 1501 |
| 42 | 90 | 5 | 238 | 10 | 145 | 2 | 280 | 21 | 300 | 2 | 584 | 20 | 580 | 2 | 1135 | 29 | 800 | 3 | 1552 |
| 43 | 91 | 6 | 244 | 11 | 150 | 2 | 290 | 22 | 310 | 2 | 604 | 21 | 600 | 3 | 1176 | | | | |
| 44 | 92 | 6 | 250 | 12 | 155 | 2 | 300 | 23 | 320 | 2 | 624 | **assign700** | | | | | | | |
| 45 | 93 | 6 | 256 | 13 | 160 | 2 | 310 | 24 | 330 | 2 | 644 | 1 | 300 | 2 | 561 | | | | |
| 46 | 94 | 6 | 262 | 14 | 165 | 2 | 320 | 25 | 340 | 2 | 664 | 2 | 320 | 2 | 601 | | | | |
| 47 | 95 | 6 | 268 | 15 | 170 | 2 | 330 | 26 | 350 | 2 | 684 | 3 | 340 | 2 | 641 | | | | |
| 48 | 96 | 6 | 274 | 16 | 175 | 2 | 340 | 27 | 360 | 2 | 704 | 4 | 360 | 2 | 681 | | | | |
| 49 | 97 | 6 | 280 | 17 | 180 | 2 | 350 | 28 | 370 | 2 | 724 | 5 | 380 | 2 | 721 | | | | |
| 50 | 98 | 7 | 287 | 18 | 185 | 2 | 360 | 29 | 380 | 2 | 744 | 6 | 400 | 2 | 761 | | | | |
| 51 | 99 | 8 | 295 | 19 | 190 | 2 | 370 | 30 | 390 | 3 | 766 | 7 | 420 | 2 | 801 | | | | |
| 52 | 100 | 10 | 305 | 20 | 195 | 2 | 380 | 31 | 400 | 5 | 804 | 8 | 440 | 2 | 841 | | | | |

In the tests, presented in Table 7, we achieved a problem reduction in the range between 90.99% and 99.05% (see Table 8).

Table 8. Assignment problems reduction

| Test | Reduction range, % | | Test | Reduction range, % |
|---|---|---|---|---|
| assign100 | 90.99-99.03 | | assign500 | 95.98-98.99 |
| assign200 | 92.1-99.05 | | assign600 | 97.95-98.96 |
| assign300 | 96.01-99.02 | | assign700 | 97.97-98.97 |
| assign400 | 95.96-98.98 | | assign800 | 97.95-98.97 |

## Conclusion

In this paper we have presented a semi-Lagrangean relaxation method for the *k*-cardinality assignment problem. The method is proven to be efficient since the number of potential semi-Lagrangean multipliers are bounded and hence no subgradient search method needs to be used. The optimal semi-Lagrangean price *u* has also a clear economic interpretation as the market price needed to get a solution where *k* assignments are done. The problem reduction obtained over the large scale test problems shows a problem reduction between 90.99% and 99.05%. In addition, it shall be noted that the subproblems to be solved are the simple assignment problems. Hence, the semi-Lagrangean algorithm presented here is polynomial.

## References


Bai, G. Z. (2009, December). A new algorithm for k-cardinality assignment problem. In *Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on* (pp. 1-4). IEEE.

Beasley, J. E. (2013, October 20). *OR-library: Assignment problem.* Retrieved October 20, 2013, from http://people.brunel.ac.uk/~mastjjb/jeb/orlib/assigninfo.html

Beltran, C., Tadonki, C., & Vial, J. (2004). *Semi-Lagrangian relaxation.* (332/658). Retrieved December 3, 2013, from http://archive-ouverte.unige.ch/unige:5768

Beltran, C., Tadonki, C., & Vial, J. P. (2006). Solving the p-median problem with a semi-Lagrangian relaxation. *Computational Optimization and Applications*, *35*(2), 239-260.

Beltrán-Royo, C., Vial, J. P., & Alonso-Ayuso, A. (2012). Semi-Lagrangian relaxation applied to the uncapacitated facility location problem. *Computational Optimization and Applications*, *51*(1), 387-409.

Bruglieri, M., Ehrgott, M., Hamacher, H. W., & Maffioli, F. (2006). An annotated bibliography of combinatorial optimization problems with fixed cardinality constraints. *Discrete Applied Mathematics*, *154*(9), 1344-1357.

Dell'Amico, M., & Martello, S. (1997). The *k*-cardinality assignment problem. *Discrete Applied Mathematics*, *76*(1), 103-121

Dell'Amico, M., Lodi, A., & Martello, S. (2001). Efficient algorithms and codes for *k*-cardinality assignment problems. *Discrete applied mathematics*, *110*(1), 25-40.

Volgenant, A. (2004). Solving the *k*-cardinality assignment problem by transformation. *European Journal of Operational Research*, *157*(2), 322-331.

## APPENDIX A

Table A.1 Illustrative example: computational results

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution | R, % |
|---|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | | |
| 1 | 5 | 8 | 1 | 1 | 8 | 31 | 31 | 93.33 |
| 2 | 6 | 15 | 1 | 1 | 15 | 46 | 46 | 75.11 |
| 3 | 7 | 15 | 1 | 1 | 15 | 61 | 61 | 75.11 |
| 4 | 8 | 15 | 1 | 1 | 15 | 76 | 76 | 75.11 |
| 5 | 9 | 15 | 1 | 1 | 15 | 91 | 91 | 75.11 |
| 6 | 10 | 15 | 1 | 1 | 15 | 106 | 106 | 75.11 |
| 7 | 11 | 15 | 1 | 1 | 15 | 121 | 121 | 75.11 |
| 8 | 12 | 16 | 1 | | 15 | 136 | 137 | 75.11 |
| | | | 2 | 1 | 16 | 137 | | |
| 9 | 13 | 16 | 1 | | 15 | 151 | 153 | 64.44 |
| | | | 2 | 1 | 16 | 153 | | |
| 10 | 14 | 16 | 1 | 1 | 16 | 169 | 169 | 64.44 |
| 11 | 15 | 17 | 1 | | 16 | 185 | 186 | 55.55 |
| | | | 2 | 1 | 17 | 186 | | |

## APPENDIX B

Table B.1 Assign100: computational results

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | |
| 1 | 20 | 2 | 1 | 1 | 2 | 39 | 39 |
| 2 | 50 | 2 | 1 | 1 | 2 | 99 | 99 |
| 3 | 51 | 2 | 1 | 1 | 2 | 101 | 101 |
| 4 | 52 | 2 | 1 | 1 | 2 | 103 | 103 |
| 5 | 53 | 2 | 1 | 1 | 2 | 105 | 105 |
| 6 | 54 | 2 | 1 | 1 | 2 | 107 | 107 |
| 7 | 55 | 3 | 1 | | 2 | 109 | 110 |
| | | | 2 | 1 | 3 | 110 | |
| 8 | 56 | 3 | 1 | | 2 | 111 | 113 |
| | | | 2 | 1 | 3 | 113 | |
| 9 | 57 | 3 | 1 | | 2 | 113 | 116 |
| | | | 2 | 1 | 3 | 116 | |
| 10 | 58 | 3 | 1 | | 2 | 115 | 119 |
| | | | 2 | 1 | 3 | 119 | |
| 11 | 59 | 3 | 1 | | 2 | 117 | 122 |
| | | | 2 | 1 | 3 | 122 | |
| 12 | 60 | 3 | 1 | | 2 | 119 | 125 |
| | | | 2 | 1 | 3 | 125 | |
| 13 | 61 | 3 | 1 | | 2 | 121 | 128 |
| | | | 2 | 1 | 3 | 128 | |
| 14 | 62 | 3 | 1 | 1 | 3 | 131 | 131 |
| 15 | 63 | 3 | 1 | 1 | 3 | 134 | 134 |
| 16 | 64 | 3 | 1 | 1 | 3 | 137 | 137 |
| 17 | 65 | 3 | 1 | 1 | 3 | 140 | 140 |
| 18 | 66 | 3 | 1 | 1 | 3 | 143 | 143 |
| 19 | 67 | 3 | 1 | 1 | 3 | 146 | 146 |
| 20 | 68 | 3 | 1 | 1 | 3 | 149 | 149 |
| 21 | 69 | 3 | 1 | 1 | 3 | 152 | 152 |
| 22 | 70 | 3 | 1 | 1 | 3 | 155 | 155 |
| 23 | 71 | 3 | 1 | 1 | 3 | 158 | 158 |
| 24 | 72 | 3 | 1 | 1 | 3 | 161 | 161 |
| 25 | 73 | 3 | 1 | 1 | 3 | 164 | 164 |
| 26 | 74 | 3 | 1 | 1 | 3 | 168 | 168 |
| 27 | 75 | 3 | 1 | | 3 | 170 | 172 |
| | | | 2 | 1 | 4 | 172 | |
| 28 | 76 | 4 | 1 | | 3 | 173 | 176 |
| | | | 2 | 1 | 4 | 176 | |
| 29 | 77 | 4 | 1 | | 3 | 176 | 180 |
| | | | 2 | 1 | 4 | 180 | |

Table B.1 *Continued*

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | |
| 30 | 78 | 4 | 1 | | 3 | 179 | 184 |
| | | | 2 | 1 | 4 | 184 | |
| 31 | 79 | 4 | 1 | | 3 | 182 | 188 |
| | | | 2 | 1 | 4 | 188 | |
| 32 | 80 | 4 | 1 | | 3 | 185 | 192 |
| | | | 2 | 1 | 4 | 192 | |
| 33 | 81 | 4 | 1 | | 3 | 188 | 196 |
| | | | 2 | 1 | 4 | 196 | |
| 34 | 82 | 4 | 1 | | 3 | 191 | 200 |
| | | | 2 | 1 | 4 | 200 | |
| 35 | 83 | 4 | 1 | | 3 | 194 | 204 |
| | | | 2 | 1 | 4 | 204 | |
| 36 | 84 | 4 | 1 | | 3 | 197 | 208 |
| | | | 2 | 1 | 4 | 208 | |
| 37 | 85 | 5 | 1 | | 3 | 200 | 213 |
| | | | 2 | | 4 | 212 | |
| | | | 3 | 1 | 5 | 213 | |
| 38 | 86 | 5 | 1 | | 3 | 203 | 218 |
| | | | 2 | | 4 | 216 | |
| | | | 3 | 1 | 5 | 218 | |
| 39 | 87 | 5 | 1 | | 4 | 220 | 223 |
| | | | 2 | 1 | 5 | 223 | |
| 40 | 88 | 5 | 1 | | 4 | 224 | 228 |
| | | | 2 | 1 | 5 | 228 | |
| 41 | 89 | 5 | 1 | | 4 | 228 | 233 |
| | | | 2 | 1 | 5 | 233 | |
| 42 | 90 | 5 | 1 | | 4 | 232 | 238 |
| | | | 2 | 1 | 5 | 238 | |
| 43 | 91 | 6 | 1 | | 4 | 236 | 244 |
| | | | 2 | 1 | 5 | 243 | |
| | | | 3 | 2 | 6 | 244 | |
| 44 | 92 | 6 | 1 | | 4 | 240 | 250 |
| | | | 2 | 1 | 5 | 248 | |
| | | | 3 | 2 | 6 | 250 | |
| 45 | 93 | 6 | 1 | | 4 | 244 | 256 |
| | | | 2 | | 5 | 253 | |
| | | | 3 | 1 | 6 | 256 | |
| 46 | 94 | 6 | 1 | | 4 | 248 | 262 |
| | | | 2 | | 5 | 258 | |
| | | | 3 | 1 | 6 | 262 | |

Table B.1 *Continued*

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | |
| 47 | 95 | 6 | 1 | | 4 | 252 | 268 |
| | | | 2 | | 5 | 263 | |
| | | | 3 | 1 | 6 | 268 | |
| 48 | 96 | 6 | 1 | | 4 | 256 | 274 |
| | | | 2 | | 5 | 268 | |
| | | | 3 | 1 | 6 | 274 | |
| 49 | 97 | 6 | 1 | | 4 | 260 | 280 |
| | | | 2 | | 5 | 273 | |
| | | | 3 | 1 | 6 | 280 | |
| 50 | 98 | 7 | 1 | | 4 | 264 | 287 |
| | | | 2 | | 5 | 278 | |
| | | | 3 | 1 | 6 | 286 | |
| | | | 4 | 2 | 7 | 287 | |
| 51 | 99 | 8 | 1 | | 5 | 283 | 295 |
| | | | 2 | 1 | 6 | 292 | |
| | | | 3 | 2 | 7 | 294 | |
| | | | 4 | 3 | 8 | 295 | |
| 52 | 100 | 10 | 1 | | 5 | 293 | 305 |
| | | | 2 | | 6 | 298 | |
| | | | 3 | 1 | 7 | 301 | |
| | | | 4 | 2 | 8 | 303 | |
| | | | 5 | 3 | 9 | 304 | |
| | | | 6 | 4 | 10 | 305 | |

## APPENDIX C

Table C.1 Assign200: computational results

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | |
| 1 | 50 | 2 | 1 | 1 | 2 | 98 | 98 |
| 2 | 55 | 2 | 1 | 1 | 2 | 108 | 108 |
| 3 | 60 | 2 | 1 | 1 | 2 | 118 | 118 |
| 4 | 65 | 2 | 1 | 1 | 2 | 128 | 128 |
| 5 | 70 | 2 | 1 | 1 | 2 | 138 | 138 |
| 6 | 75 | 2 | 1 | 1 | 2 | 148 | 148 |
| 7 | 80 | 2 | 1 | 1 | 2 | 128 | 158 |
| 8 | 85 | 2 | 1 | 1 | 2 | 168 | 168 |
| 9 | 90 | 2 | 1 | 1 | 2 | 178 | 178 |
| 10 | 95 | 2 | 1 | 1 | 2 | 188 | 188 |
| 11 | 100 | 2 | 1 | 1 | 2 | 198 | 198 |
| 12 | 105 | 2 | 1 | 1 | 2 | 208 | 208 |
| 13 | 110 | 2 | 1 | 1 | 2 | 218 | 218 |
| 14 | 115 | 2 | 1 | 1 | 2 | 228 | 228 |
| 15 | 120 | 2 | 1 | 1 | 2 | 238 | 238 |
| 16 | 125 | 2 | 1 | 1 | 2 | 248 | 248 |
| 17 | 130 | 2 | 1 | 1 | 2 | 258 | 258 |
| 18 | 135 | 2 | 1 | 1 | 2 | 268 | 268 |
| 19 | 140 | 2 | 1 | 1 | 2 | 278 | 278 |
| 20 | 145 | 2 | 1 | 1 | 2 | 288 | 288 |
| 21 | 150 | 2 | 1 | 1 | 2 | 298 | 298 |
| 22 | 155 | 3 | 1 | | 2 | 308 | 309 |
| | | | 2 | 1 | 3 | 309 | |
| 23 | 160 | 3 | 1 | | 2 | 318 | 324 |
| | | | 2 | 1 | 3 | 324 | |
| 24 | 165 | 3 | 1 | | 2 | 328 | 339 |
| | | | 2 | 1 | 3 | 339 | |
| 25 | 170 | 3 | 1 | 1 | 3 | 354 | 354 |
| 26 | 175 | 3 | 1 | 1 | 3 | 369 | 369 |
| 27 | 180 | 3 | 1 | 1 | 3 | 384 | 384 |
| 28 | 185 | 4 | 1 | 1 | 3 | 399 | 401 |
| | | | 2 | 2 | 4 | 401 | |
| 29 | 190 | 4 | 1 | 1 | 3 | 414 | 421 |
| | | | 2 | 2 | 4 | 421 | |
| 30 | 195 | 5 | 1 | | 3 | 429 | 444 |
| | | | 2 | 1 | 4 | 441 | |
| | | | 3 | 2 | 5 | 444 | |
| 31 | 200 | 9 | 1 | 1 | 5 | 469 | 475 |
| | | | 2 | 2 | 6 | 471 | |
| | | | 3 | 3 | 7 | 473 | |
| | | | 4 | 4 | 8 | 474 | |
| | | | 5 | 5 | 9 | 475 | |

## APPENDIX D

Table D.1 Assign300: computational results

| Test # | *k* | *u** | Iteration # | | *u* | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | |
| 1 | 100 | 2 | 1 | 1 | 2 | 190 | 190 |
| 2 | 105 | 2 | 1 | 1 | 2 | 200 | 200 |
| 3 | 110 | 2 | 1 | 1 | 2 | 210 | 210 |
| 4 | 115 | 2 | 1 | 1 | 2 | 220 | 220 |
| 5 | 120 | 2 | 1 | 1 | 2 | 230 | 230 |
| 6 | 125 | 2 | 1 | 1 | 2 | 240 | 240 |
| 7 | 130 | 2 | 1 | 1 | 2 | 250 | 250 |
| 8 | 135 | 2 | 1 | 1 | 2 | 260 | 260 |
| 9 | 140 | 2 | 1 | 1 | 2 | 270 | 270 |
| 10 | 145 | 2 | 1 | 1 | 2 | 280 | 280 |
| 11 | 150 | 2 | 1 | 1 | 2 | 290 | 290 |
| 12 | 155 | 2 | 1 | 1 | 2 | 300 | 300 |
| 13 | 160 | 2 | 1 | 1 | 2 | 310 | 310 |
| 14 | 165 | 2 | 1 | 1 | 2 | 320 | 320 |
| 15 | 170 | 2 | 1 | 1 | 2 | 330 | 330 |
| 16 | 175 | 2 | 1 | 1 | 2 | 340 | 340 |
| 17 | 180 | 2 | 1 | 1 | 2 | 350 | 350 |
| 18 | 185 | 2 | 1 | 1 | 2 | 360 | 360 |
| 19 | 190 | 2 | 1 | 1 | 2 | 370 | 370 |
| 20 | 195 | 2 | 1 | 1 | 2 | 380 | 380 |
| 21 | 200 | 2 | 1 | 1 | 2 | 390 | 390 |
| 22 | 205 | 2 | 1 | 1 | 2 | 400 | 400 |
| 23 | 210 | 2 | 1 | 1 | 2 | 410 | 410 |
| 24 | 215 | 2 | 1 | 1 | 2 | 420 | 420 |
| 25 | 220 | 2 | 1 | 1 | 2 | 430 | 430 |
| 26 | 225 | 2 | 1 | 1 | 2 | 440 | 440 |
| 27 | 230 | 2 | 1 | 1 | 2 | 450 | 450 |
| 28 | 235 | 2 | 1 | 1 | 2 | 460 | 460 |
| 29 | 240 | 2 | 1 | 1 | 2 | 470 | 470 |
| 30 | 245 | 2 | 1 | 1 | 2 | 480 | 480 |
| 31 | 250 | 2 | 1 | 1 | 2 | 490 | 490 |
| 32 | 255 | 2 | 1 | 1 | 2 | 500 | 500 |
| 33 | 260 | 2 | 1 | 1 | 2 | 510 | 510 |
| 34 | 265 | 2 | 1 | 1 | 2 | 520 | 520 |
| 35 | 270 | 2 | 1 | 1 | 2 | 530 | 530 |
| 36 | 275 | 2 | 1 | 1 | 2 | 540 | 540 |
| 37 | 280 | 3 | 1 | 1 | 3 | 555 | 555 |
| 38 | 285 | 3 | 1 | 1 | 3 | 570 | 570 |
| 39 | 290 | 3 | 1 | 1 | 3 | 585 | 585 |

Table D.1 *Continued*

| Test # | *k* | *u** | Iteration # | | *u* | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | |
| 40 | 295 | 4 | 1 | 1 | 3 | 600 | 604 |
| | | | 2 | 2 | 4 | 604 | |
| 41 | 300 | 5 | 1 | 1 | 4 | 624 | 626 |
| | | | 2 | 2 | 5 | 626 | |

## APPENDIX E

Table E.1 Assign400: computational results

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | |
| 1 | 100 | 2 | 1 | 1 | 2 | 184 | 184 |
| 2 | 110 | 2 | 1 | 1 | 2 | 204 | 204 |
| 3 | 120 | 2 | 1 | 1 | 2 | 224 | 224 |
| 4 | 130 | 2 | 1 | 1 | 2 | 244 | 244 |
| 5 | 140 | 2 | 1 | 1 | 2 | 264 | 264 |
| 6 | 150 | 2 | 1 | 1 | 2 | 284 | 284 |
| 7 | 160 | 2 | 1 | 1 | 2 | 304 | 304 |
| 8 | 170 | 2 | 1 | 1 | 2 | 324 | 324 |
| 9 | 180 | 2 | 1 | 1 | 2 | 344 | 344 |
| 10 | 190 | 2 | 1 | 1 | 2 | 364 | 364 |
| 11 | 200 | 2 | 1 | 1 | 2 | 384 | 384 |
| 12 | 210 | 2 | 1 | 1 | 2 | 404 | 404 |
| 13 | 220 | 2 | 1 | 1 | 2 | 424 | 424 |
| 14 | 230 | 2 | 1 | 1 | 2 | 444 | 444 |
| 15 | 240 | 2 | 1 | 1 | 2 | 464 | 464 |
| 16 | 250 | 2 | 1 | 1 | 2 | 484 | 484 |
| 17 | 260 | 2 | 1 | 1 | 2 | 504 | 504 |
| 18 | 270 | 2 | 1 | 1 | 2 | 524 | 524 |
| 19 | 280 | 2 | 1 | 1 | 2 | 544 | 544 |
| 20 | 290 | 2 | 1 | 1 | 2 | 564 | 564 |
| 21 | 300 | 2 | 1 | 1 | 2 | 584 | 584 |
| 22 | 310 | 2 | 1 | 1 | 2 | 604 | 604 |
| 23 | 320 | 2 | 1 | 1 | 2 | 624 | 624 |
| 24 | 330 | 2 | 1 | 1 | 2 | 644 | 644 |
| 25 | 340 | 2 | 1 | 1 | 2 | 664 | 664 |
| 26 | 350 | 2 | 1 | 1 | 2 | 684 | 684 |
| 27 | 360 | 2 | 1 | 1 | 2 | 704 | 704 |
| 28 | 370 | 2 | 1 | 1 | 2 | 724 | 724 |
| 29 | 380 | 2 | 1 | 1 | 2 | 744 | 744 |
| 30 | 390 | 3 | 1 | | 2 | 764 | 766 |
| | | | 2 | 1 | 3 | 766 | |
| 31 | 400 | 5 | 1 | 1 | 3 | 796 | 804 |
| | | | 2 | 2 | 4 | 803 | |
| | | | 3 | 3 | 5 | 804 | |

## APPENDIX F

Table F.1 Assign500: computational results

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | Conservative | Greedy | | | |
| 1 | 100 | 2 | 1 | 1 | 2 | 179 | 179 |
| 2 | 120 | 2 | 1 | 1 | 2 | 219 | 219 |
| 3 | 140 | 2 | 1 | 1 | 2 | 259 | 259 |
| 4 | 160 | 2 | 1 | 1 | 2 | 299 | 299 |
| 5 | 180 | 2 | 1 | 1 | 2 | 339 | 339 |
| 6 | 200 | 2 | 1 | 1 | 2 | 379 | 379 |
| 7 | 220 | 2 | 1 | 1 | 2 | 419 | 419 |
| 8 | 240 | 2 | 1 | 1 | 2 | 459 | 459 |
| 9 | 260 | 2 | 1 | 1 | 2 | 499 | 499 |
| 10 | 280 | 2 | 1 | 1 | 2 | 539 | 539 |
| 11 | 300 | 2 | 1 | 1 | 2 | 579 | 579 |
| 12 | 320 | 2 | 1 | 1 | 2 | 619 | 619 |
| 13 | 340 | 2 | 1 | 1 | 2 | 659 | 659 |
| 14 | 360 | 2 | 1 | 1 | 2 | 699 | 699 |
| 15 | 380 | 2 | 1 | 1 | 2 | 739 | 739 |
| 16 | 400 | 2 | 1 | 1 | 2 | 779 | 779 |
| 17 | 420 | 2 | 1 | 1 | 2 | 819 | 819 |
| 18 | 440 | 2 | 1 | 1 | 2 | 859 | 859 |
| 19 | 460 | 2 | 1 | 1 | 2 | 899 | 899 |
| 20 | 480 | 2 | 1 | 1 | 2 | 939 | 939 |
| 21 | 500 | 5 | 1 | | 3 | 987 | 991 |
| | | | 2 | 1 | 4 | 990 | |
| | | | 3 | 2 | 5 | 991 | |

## APPENDIX G

Table G.1 Assign600: computational results

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | *Greedy* | | | |
| 1 | 200 | 2 | 1 | 1 | 2 | 375 | 375 |
| 2 | 220 | 2 | 1 | 1 | 2 | 415 | 415 |
| 3 | 240 | 2 | 1 | 1 | 2 | 455 | 455 |
| 4 | 260 | 2 | 1 | 1 | 2 | 495 | 495 |
| 5 | 280 | 2 | 1 | 1 | 2 | 535 | 535 |
| 6 | 300 | 2 | 1 | 1 | 2 | 575 | 575 |
| 7 | 320 | 2 | 1 | 1 | 2 | 615 | 615 |
| 8 | 340 | 2 | 1 | 1 | 2 | 655 | 655 |
| 9 | 360 | 2 | 1 | 1 | 2 | 695 | 695 |
| 10 | 380 | 2 | 1 | 1 | 2 | 735 | 735 |
| 11 | 400 | 2 | 1 | 1 | 2 | 775 | 775 |
| 12 | 420 | 2 | 1 | 1 | 2 | 815 | 815 |
| 13 | 440 | 2 | 1 | 1 | 2 | 855 | 855 |
| 14 | 460 | 2 | 1 | 1 | 2 | 895 | 895 |
| 15 | 480 | 2 | 1 | 1 | 2 | 935 | 935 |
| 16 | 500 | 2 | 1 | 1 | 2 | 975 | 975 |
| 17 | 520 | 2 | 1 | 1 | 2 | 1015 | 1015 |
| 18 | 540 | 2 | 1 | 1 | 2 | 1055 | 1055 |
| 19 | 560 | 2 | 1 | 1 | 2 | 1095 | 1095 |
| 20 | 580 | 2 | 1 | 1 | 2 | 1135 | 1135 |
| 21 | 600 | 3 | 1 | 1 | 2 | 1175 | 1176 |
| | | | 2 | 2 | 3 | 1176 | |

## APPENDIX H

Table H.1 Assign700: computational results

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | *Conservative* | Greedy | | | |
| 1 | 300 | 2 | 1 | 1 | 2 | 561 | 561 |
| 2 | 320 | 2 | 1 | 1 | 2 | 601 | 601 |
| 3 | 340 | 2 | 1 | 1 | 2 | 641 | 641 |
| 4 | 360 | 2 | 1 | 1 | 2 | 681 | 681 |
| 5 | 380 | 2 | 1 | 1 | 2 | 721 | 721 |
| 6 | 400 | 2 | 1 | 1 | 2 | 761 | 761 |
| 7 | 420 | 2 | 1 | 1 | 2 | 801 | 801 |
| 8 | 440 | 2 | 1 | 1 | 2 | 841 | 841 |
| 9 | 460 | 2 | 1 | 1 | 2 | 881 | 881 |
| 10 | 480 | 2 | 1 | 1 | 2 | 921 | 921 |
| 11 | 500 | 2 | 1 | 1 | 2 | 961 | 961 |
| 12 | 520 | 2 | 1 | 1 | 2 | 1001 | 1001 |
| 13 | 540 | 2 | 1 | 1 | 2 | 1041 | 1041 |
| 14 | 560 | 2 | 1 | 1 | 2 | 1081 | 1081 |
| 15 | 580 | 2 | 1 | 1 | 2 | 1121 | 1121 |
| 16 | 600 | 2 | 1 | 1 | 2 | 1161 | 1161 |
| 17 | 620 | 2 | 1 | 1 | 2 | 1201 | 1201 |
| 18 | 640 | 2 | 1 | 1 | 2 | 1241 | 1241 |
| 19 | 660 | 2 | 1 | 1 | 2 | 1281 | 1281 |
| 20 | 680 | 2 | 1 | 1 | 2 | 1321 | 1321 |
| 21 | 700 | 3 | 1 | | 2 | 1361 | 1362 |
| | | | 2 | 1 | 3 | 1362 | |

## APPENDIX I

Table I.1 Assign800: computational results

| Test # | *k* | *u** | Iteration # | | u | LBD | Solution |
|---|---|---|---|---|---|---|---|
| | | | Conservative | Greedy | | | |
| 1 | 100 | 2 | 1 | 1 | 2 | 151 | 151 |
| 2 | 125 | 2 | 1 | 1 | 2 | 201 | 201 |
| 3 | 150 | 2 | 1 | 1 | 2 | 251 | 251 |
| 4 | 175 | 2 | 1 | 1 | 2 | 301 | 301 |
| 5 | 200 | 2 | 1 | 1 | 2 | 351 | 351 |
| 6 | 225 | 2 | 1 | 1 | 2 | 401 | 401 |
| 7 | 250 | 2 | 1 | 1 | 2 | 451 | 451 |
| 8 | 275 | 2 | 1 | 1 | 2 | 501 | 501 |
| 9 | 300 | 2 | 1 | 1 | 2 | 551 | 551 |
| 10 | 325 | 2 | 1 | 1 | 2 | 601 | 601 |
| 11 | 350 | 2 | 1 | 1 | 2 | 651 | 651 |
| 12 | 375 | 2 | 1 | 1 | 2 | 701 | 701 |
| 13 | 400 | 2 | 1 | 1 | 2 | 751 | 751 |
| 14 | 425 | 2 | 1 | 1 | 2 | 801 | 801 |
| 15 | 450 | 2 | 1 | 1 | 2 | 851 | 851 |
| 16 | 475 | 2 | 1 | 1 | 2 | 901 | 901 |
| 17 | 500 | 2 | 1 | 1 | 2 | 951 | 951 |
| 18 | 525 | 2 | 1 | 1 | 2 | 1001 | 1001 |
| 19 | 550 | 2 | 1 | 1 | 2 | 1051 | 1051 |
| 20 | 575 | 2 | 1 | 1 | 2 | 1101 | 1101 |
| 21 | 600 | 2 | 1 | 1 | 2 | 1151 | 1151 |
| 22 | 625 | 2 | 1 | 1 | 2 | 1201 | 1201 |
| 23 | 650 | 2 | 1 | 1 | 2 | 1251 | 1251 |
| 24 | 675 | 2 | 1 | 1 | 2 | 1301 | 1301 |
| 25 | 700 | 2 | 1 | 1 | 2 | 1351 | 1351 |
| 26 | 725 | 2 | 1 | 1 | 2 | 1401 | 1401 |
| 27 | 750 | 2 | 1 | 1 | 2 | 1451 | 1451 |
| 28 | 775 | 2 | 1 | 1 | 2 | 1501 | 1501 |
| 29 | 800 | 3 | 1 | 1 | 2 | 1551 | 1552 |
| | | | 2 | 2 | 3 | 1552 | |