Combination of nearest neighbor and heuristics algorithms for sequential two dimensional loading capacitated vehicle routing problem

A.A. Sarwono, T.J. Ai and S.S. Wigati
Department of Industrial Engineering, Universitas Atma Jaya Yogyakarta, Jl. Babarsari No. 44 Yogyakarta 55281, Indonesia
E-mails: asa.adiaji@yahoo.com, jinai@mail.uajy.ac.id, yayan@mail.uajy.ac.id

Abstract. Vehicle Routing Problem (VRP) is a method for determining the optimal route of vehicles in order to serve customers starting from depot. Combination of the two most important problems in distribution logistics, which is called the two dimensional loading vehicle routing problem, is considered in this paper. This problem combines the loading of the freight into the vehicles and the successive routing of the vehicles along the route. Moreover, an additional feature of last-in-first-out loading sequences also considered. In the sequential two dimensional loading capacitated vehicle routing problem (sequential 2L-CVRP), the loading must be compatible with the trip sequence: when the vehicle arrives at a customer \(i\), there must be no obstacle (items for other customers) between the item of \(i\) and the loading door (rear part) of the vehicle. In other words, it is not necessary to move non-\(i\)'s items whenever the unloading process of the items of \(i\). According with aforementioned conditions, a program to solve sequential 2L-CVRP is required. A nearest neighbor algorithm for solving the routing problem is presented, in which the loading component of the problem is solved through a collection of 5 packing heuristics.

1. Introduction
Distribution is one of the logistic system components which has the responsibility to handle material movement among facilities. Even though, it is believe that the distribution process has a broader scope than just transportation of goods. Distribution is the factor that determine total profit of company because it affects the supply chain cost and direct customers satisfaction, simultaneously [1]. Survey said that distribution cost of American company is equal to 20% of cost of goods sold of a product [2]. So, distribution planning is one of the most important activities to make supply chain process more efficient. Scope of distribution problem consists of several aspects, such as route determination, facility location, and delivery capacity [3].

The vehicle routing problem (VRP) is one of the most frequently studied problems in distribution. It consists of finding an optimal set of trips for a fleet of vehicles which must serve a given set of customers. Several versions of the vehicle routing problem exist in the literature, in particular the capacitated VRP (CVRP). In this variant, customers with known demands are spread over an undirected network and must be visited using a fleet of identical vehicles with limited capacity [4]. CVRP has been solved with many methods, heuristics or metaheuristics in [5], [6], [7], [8], and [9]. Gendreau et al. [10] has developed Tabu search algorithm and Fuellerer et al. [11] used Ant Colony System (ACO) to solve CVRP problem. Exact method also used in [12], [13], and [14] which is able...
to solve this CVRP properly. Beside the problem settings defined in the CVRP, we should know that loading and unloading problem to-from container is a common setting in real distribution cases.

Two-dimensional bin packing problem (2BPP) is the term for defining a loading problem which seen the problem in two dimensions only. Exact algorithms and lower bounds was used by [15], [16], [17], [18], [19], [20], and [21] to solve the 2BPP. A lot of heuristics methods have also been used to solve this problem [22]. Beside that, metaheuristics method such as GRASP algorithm and tabusearch [23, 24], and also iterated local search method [25] have been used to solve the 2BPP.

This paper addresses the two-dimensional loading capacitated vehicle routing problem, which is denoted as 2L-CVRP [26, 27]. The 2L-CVRP is a variant of one of the most frequently studied combinatorial optimization problems, the capacitated vehicle routing problem (CVRP). If it is viewed from the loading type, there are two version of 2L-CVRP, unrestricted and sequential loading.

**Figure 1.** Example of a 2L-CVRP route with unrestricted and sequential loadings

Figure 1 depicts that unrestricted version does not pay attention between items location in container and the order of customers that should be visited. This version is used for the vehicle which the container can be unloaded vertically using crane. Beside, sequential version have additional feature named last-in-first-out, in other words there is no need to move non-i’s items when unload the items of customer i. In this version, we need to pay attention in items arrangement inside the container, because it should be compatible with the order of customers that should be visited. Specifically, this paper addresses the 2L-CVRP with sequential version of loading, named sequential 2L-CVRP.

The 2L-CVRP was first solved by an exact algorithm which used the branch-and-cut technique [28]. In the test dataset, their approach can deal with the instances with no more than 30 customers and 91 items. As for the larger scale problems, a metaheuristic approach was proposed by Gendreau et al. [29]. Precisely, the Tabu search was employed for routing aspects of the problem. Usually, the means of lower bounds, heuristics, local search and a truncated branch-and-bound were used to check the loading feasibility. 180 problem instances were tested in their work. The number of customers went up to 255 and the items up to 786. Recently, a new method, the guided Tabu search [30], which combines
the Tabu search with guided local search, was presented. For checking the feasibility of loading, a collection of packing heuristics was used. To accelerate the algorithm, two strategies that reduced the neighborhoods explored, and record of the loading feasibility information, were employed. A nature inspired metaheuristic algorithm, an effective heuristic based on ant colony optimization, has been proposed by Fuellerer et al. [11]. The costs of four different loading configurations were compared in this work.

The importance of the 2L-CVRP is mainly reflected in two aspects. Theoretically, composed of two NP-hard optimization problems (CVRP and 2BPP), it is also a high complexity NP-hard problem. For practical applications, this problem may exist at many companies. An efficient method to solve this problem can significantly reduce costs for the companies.

2. Problem description
This paper considers the sequential version of the 2L-CVRP defined as follows:

- Like the VRP, the problem is based on a complete undirected graph with a set of n + 1 nodes. Node 0 is a depot while node 1 to n are customers. The edge \([i,j]\) between any two distinct nodes \(i\) and \(j\) models a shortest path of length \(d_{ij} = d_{ji}\).
- The depot contains a virtually unlimited fleet of homogeneous vehicles with a capacity \(Q\) (maximum weight) and a rectangular loading surface of length \(L\) and width \(W\). The loading surface is denoted as \(A = WL\).
- A set \(D\) of \(m\) items must be delivered to customers. Item \(t (t = 1, 2, ..., m)\) has a length \(l_t \leq L\) and a width \(w_t \leq W\). When loaded in a vehicle, its \(l\)-edge and \(w\)-edge must be respectively parallel to the vehicle \(L\) and \(W\)-edges, i.e., rotations are not allowed.
- \(D\) is partitioned into \(n\) subsets \(D_1 \cup D_2 \cup ... \cup D_n\), where \(D\) is the subset of items requested by customer \(i\), with a known total weight \(q_i\).
- Each set \(D_i\) must be loaded into a single vehicle.
- Each customer must be visited by one, and only one vehicle, once.
- Every vehicle starts from, and ends at, the central depot.
- The weight of the items loaded in a vehicle must not exceed the capacity of the vehicle \(Q\).
- All the items in a vehicle must be loaded in area \(A\). Overlapping loading is not permitted.
- When a vehicle is visiting customer \(i\), all of the items in the set of \(D_i\) can be unloaded from the vehicle by means of forklift trucks parallel to the length dimension of the vehicle surface, without moving other items required by other customers.

The sequential 2L-CVRP objective is to determine a set of routes of minimal total length that satisfy the following constraints: (a) every route starts and ends at the central depot, (b) the demand of every customer is totally covered, (c) each customer is visited once, (d) the total weight of all items demanded by the set of customers covered by a route must not exceed the capacity of the vehicle \(Q\), (e) there must be a non-overlapping loading of all items demanded by the set of customers covered by a route into the \(L \times W\) loading surface of the vehicles and (f) the loading of the items must ensure that whenever a customer \(i\) is visited, all items in the set \(D_i\) can be unloaded by employing a sequence of straight movements (one per item) parallel to the length dimension of the vehicle surface. In other words, no item of customer \(j\), visited after customer \(i\), can be placed between items of customer \(i\) and the rear part (loading door) of the vehicle.

3. Proposed algorithm
The proposed algorithm for the solution of the 2L-CVRP employs nearest neighbor (NN) to solve route determination problem. Regarding loading constraints of the problem, the bundle of packing heuristics designed in Zachariadis et al. [7] are applied.

3.1. Route determination
Route-first-cluster-second concept is used in this research. The meaning of route-first concept is that the determination of the route made at the beginning, in which nearest neighbor method is applied
here. After that, routes are cut to form clusters as the cluster-second concept whenever one of these following two constraints are violated: (1) if the weight of customers items exceed the container weight capacity (weight capacity constraint) and if (2) the item cannot be loaded into the same container (area capacity constraint). If one of these two constraints violation happened, a new container are needed and consequently a new route cluster is created.

We are proposing two technique to form route clusters, which are called regular cutting NN and skip cutting NN. In regular cutting NN, routes are cutting off if the next customer cannot fulfill one of the two constraints that has been explained above, which is the weight of customers items exceed the container weight capacity or the item that should be delivered to the customer is not enough anymore if loaded on the same container. In skip cutting NN, the route should be cutting off if two constraints feasibility from all customers have been checked. If in that checking process the customer can fulfill two constraints, so that customer’s item can be loaded into the same container.

![Distribution Route](image1)

**Regular Cutting Nearest Neighbor**

![Regular Cutting Nearest Neighbor](image2)

**Skip Cutting Nearest Neighbor**

![Skip Cutting Nearest Neighbor](image3)

**Figure 2.** The difference between regular and skip cutting NN

The bottom part of figure 2 (skip cutting NN) illustrates that customer 2 cannot fulfill one of two existing loading constraints, but customer 3 can fulfill those two constraints and then the condition of
customer 4 are same with customer 2. So, this condition affect the routes cutting process, customer 1 and 3 are loaded on the same container, while customer 2 and 4 are loaded on the different container. From figure 2 we can see that in skip cutting NN, all of customer are checked for its loading feasibility.

3.2. The bundle of packing heuristics

For loading items in the container, there are five heuristic methods used in this research which are combined with the two proposed route determination methods. These five heuristics are bottom-left fill ($W$ axis), bottom-left fill ($L$ axis), maximum touching perimeter, maximum touching perimeter (no walls), and minimum area. Every heuristic method have its own characteristics for determining the location of an item to be loaded into the container. Each customer have at least 1 item and not more than 5 item to be loaded. Order of each customers item that should be loaded are determined from area of the item which sorted from smallest area.

When new item loaded into the container, there will be a new $posList$. Let $posList$ denote a list of available loading positions for the items. In the beginning, the only available loading position lies in the front left corner $(0, 0)$ of the vehicle, so $posList = \{(0, 0)\}$. Whenever an item is inserted, its loading position is erased from the $posList$, while new loading positions are generated and added into the $posList$. In this way, the positions of holes that may be created between the placed items are stored into the $posList$ and may be later filled by the subsequent items.

The position for the placement of an item is selected from the list of available positions $posList$ and must not lead to any loading constraint violation (overlapping or sequential constraint). As later explained, it is determined by the packing heuristic currently employed. If all items are packed onto the loading surface, the route is considered to be feasible in terms of the loading constraints of the problem. If, on the other hand, the insertion of an item into any available position leads to loading constraint violations, the method empties the loading surface, $posList$ is set equal to $(0, 0)$, and the next packing heuristic is employed from the beginning. If none of the five available packing heuristics manages to produce a feasible loading, the heuristic bundle is applied to the second ordering of the items. If again, no feasible loading is obtained, the examined route is considered to be infeasible regarding the loading constraints.

As mentioned earlier, the loading position of an inserted item is determined by the packing heuristic currently in use. This position must be feasible, i.e., it must not lead to any overlaps or sequence constraint violations. Each of the proposed five packing heuristics $Heur_i (i = 1...5)$ employs a different criterion for selecting the loading position of an item:

$Heur_1 : Bottom-Left Fill (W$-axis$) [31]$

From the feasible available loading positions of $posList$, the position selected is the one with the minimum $W$-axis coordinate, breaking ties by minimum $L$-axis coordinate. Using this heuristic, the packing tends to evolve in the form of strips parallel to the $W$-axis.

$Heur_2 : Bottom-Left Fill (L$-axis$) [31]$

From the feasible available loading positions of $posList$, the position selected is the one with the minimum $L$-axis coordinate, breaking ties by minimum $W$-axis coordinate. Using this heuristic, the packing tends to evolve in the form of strips parallel to the $W$-axis.
Figure 3. Calculating the perimeter for Maximum Touching Perimeter heuristic.

Heur$_3$ : Maximum Touching Perimeter [32]

For each of the feasible available positions of posList, the total touching perimeter of the inserted item is calculated. The total touching perimeter is evaluated as the sum of the common edges of the inserted item with the edges of the already inserted items, and the edges of the loading surface of the vehicle as seen in figure 3: the total touching perimeter of item $C$ placed in position $(w_A, 0)$ is demonstrated by the bold dotted lines, and is equal to $l_C + w_C + (w_B - w_A)$. Term $l_C$ corresponds to the common edges of items $C$ and $A$, term $w_C$ corresponds to the common edges of item $C$ and the loading surface, and term $(w_B - w_A)$ corresponds to the common edges of items $C$ and $B$. The item is placed into the loading position that maximises the value of touching perimeter.

Figure 4. Calculating the perimeter for Maximum Touching Perimeter No Walls heuristic.

Heur$_4$ : Maximum Touching Perimeter No Walls [32]

As in the case of the Max Touching Perimeter heuristic, for each of the feasible available positions of posList, the total touching perimeter of the inserted item is calculated. In this case, the total touching perimeter is evaluated as the sum of the common edges of the inserted item with the edges of the already inserted items. The common edges of the item and the loading surface are not taken into account. The evaluation of the touching perimeter is presented in figure 4: the total touching perimeter
of item $C$ placed in position $(w_A, 0)$ is demonstrated by the bold dotted lines, and is equal to $l_C + (w_B - w_A)$. Term $l_C$ corresponds to the common edges of items $C$ and $A$ and term $(w_B - w_A)$ corresponds to the common edges of items $C$ and $B$. The item is placed into the loading position that maximises the value of touching perimeter.

$\text{Heur}_5: \text{Minimum Area}$ [30]

For each of the feasible available positions of $\text{posList}$, the area of its corresponding rectangular surface is calculated, as demonstrated in figure 5. The area of the rectangular surface determined by the loading position $(w_A, 0)$ equal to $(W - w_A) \times l_A$, the area corresponding to loading position $(w_B, l_A)$ is equal to $(W - w_B) \times (L - l_A)$, and the area of the surface corresponding to position $(0, l_A + l_B)$ is equal to $W \times (L - l_A - l_B)$. The loading position selected is the one yielding the minimum surface area.

4. Computational results

4.1. Benchmark instances characteristics

All algorithms were implemented and tested on benchmark instances from the literature. In order to verify the performance of the proposed algorithm, the proposed algorithm was tested by 180 sequential 2L-CVRP benchmark problem instances that was used in [26], [28], [29], and [30]. The datasets are available at http://www.or.deis.unibo.it/research.html. These instances were derived from 36 CVRP instances, whose the description can be found in [33], by expressing the customer demand as a set of two-dimensional, weighted and rectangular items. To generate the aforementioned item sets, five classes of the item demand characteristics are introduced

![Figure 6. Data set contents](image-url)
• Class 1: with each customer is associated a single item of width and length equal to nil. The problems of Class 1 are in fact pure CVRP instances, as every customer sequence is feasible in terms of the loading constraints of the problem examined. They are used to test the algorithmic effectiveness in terms of the routing aspects of the problem examined.

• Classes 2–5: with each customer, a set of \( m_i \) is uniformly distributed within a given range (see table 1, column 2). Each item is classified into one of the three shape categories, with equal probability. The three categories are vertical (the relative lengths are greater than the relative widths), homogeneous (the relative lengths and widths are generated in the same intervals), and horizontal (the relative lengths are smaller than the relative widths). The dimensions (width and length) of an item are uniformly distributed into the ranges determined by this item’s shape category (see table 1, column 3-8).

Table 1. The characteristics of items of classes 2-5 instances

| Class | \( m_i \) | Vertical | Homogenous | Horizontal |
|-------|-----------|----------|-----------|------------|
|       |           | Length   | Width     | Length     | Width     | Length   | Width     |
| 2     | [1, 2]    | [0.4L, 0.9L] | [0.1W, 0.2W] | [0.2L, 0.5L] | [0.2W, 0.5W] | [0.1L, 0.2L] | [0.4W, 0.9W] |
| 3     | [1, 3]    | [0.3L, 0.8L] | [0.1W, 0.2W] | [0.2L, 0.4L] | [0.2W, 0.4W] | [0.1L, 0.2L] | [0.3W, 0.8W] |
| 4     | [1, 4]    | [0.2L, 0.7L] | [0.1W, 0.2W] | [0.1L, 0.4L] | [0.1W, 0.4W] | [0.1L, 0.2L] | [0.2W, 0.7W] |
| 5     | [1, 5]    | [0.1L, 0.6L] | [0.1W, 0.2W] | [0.1L, 0.3L] | [0.1W, 0.3W] | [0.1L, 0.2L] | [0.1W, 0.6W] |

Table 2. The characteristics of classes of 2-5 instances

| Inst | \( n \) | Number of items of classes 2-5 | Number of items of classes 2-5 |
|------|-------|---------------------------------|---------------------------------|
|      |       | 2 | 3 | 4 | 5 | Inst | \( n \) | 2 | 3 | 4 | 5 |
| 1    | 15    | 24 | 31 | 37 | 45 | 19 | 50 | 82 | 103 | 134 | 157 |
| 2    | 15    | 25 | 31 | 40 | 48 | 20 | 71 | 104 | 151 | 178 | 226 |
| 3    | 20    | 29 | 46 | 49 | 49 | 21 | 75 | 114 | 164 | 168 | 202 |
| 4    | 20    | 32 | 43 | 50 | 62 | 22 | 75 | 112 | 154 | 198 | 236 |
| 5    | 21    | 31 | 37 | 41 | 57 | 23 | 75 | 112 | 155 | 179 | 225 |
| 6    | 21    | 33 | 40 | 57 | 56 | 24 | 75 | 124 | 152 | 195 | 215 |
| 7    | 22    | 32 | 41 | 51 | 55 | 25 | 100 | 157 | 212 | 254 | 311 |
| 8    | 22    | 29 | 42 | 48 | 52 | 26 | 100 | 147 | 198 | 247 | 310 |
| 9    | 25    | 40 | 61 | 63 | 91 | 27 | 100 | 152 | 211 | 245 | 320 |
| 10   | 29    | 43 | 49 | 72 | 86 | 28 | 120 | 183 | 242 | 299 | 384 |
| 11   | 29    | 43 | 62 | 74 | 91 | 29 | 134 | 197 | 262 | 342 | 422 |
| 12   | 30    | 50 | 56 | 82 | 101 | 30 | 150 | 225 | 298 | 366 | 433 |
| 13   | 32    | 44 | 56 | 78 | 102 | 31 | 199 | 307 | 402 | 513 | 602 |
| 14   | 32    | 47 | 57 | 65 | 87 | 32 | 199 | 299 | 404 | 497 | 589 |
| 15   | 32    | 48 | 59 | 84 | 114 | 33 | 199 | 301 | 407 | 499 | 577 |
| 16   | 35    | 56 | 74 | 93 | 114 | 34 | 240 | 370 | 490 | 604 | 720 |
| 17   | 40    | 60 | 73 | 96 | 127 | 35 | 252 | 367 | 507 | 634 | 762 |
| 18   | 44    | 66 | 87 | 112 | 122 | 36 | 255 | 387 | 511 | 606 | 786 |

\( n \) number of customer

The values \( L = 40 \) and \( W = 20 \) were chosen for the dimensions of the loading area. The numbers of customers and items, in the instances for Classes 2–5, are shown in table 2. For details of the datasets,
the reader is referred to Gendreau et al. [7] and Zachariadis et al. [8]. As the aforementioned characteristics of data set, each data set contains number of customers, number of items, vehicle’s capacity, length & width of container, depot coordinate, customers coordinate, number of items/customer, and length & width of each items that different with the other data set, as we can see in figure 6.

Figure 6 shows that on that data set, there are 15 customers to be served and total items from those customers are 31 item. To know the location for each vertices (depot and customers), $X$ and $Y$ coordinate are used on the data set, for example, depot are in (30,40) position, customer 1 are in (37,52) position, et cetera. Customer 1 have 2 items that should be delivered which each item have length and width $4 \times 11$ and $13 \times 6$, respectively, with total weight for those two item are 7. Total weight capacity for the container are 90 with container area is $40 \times 20$.

### 4.2. Results on benchmark instances

There are ten algorithms combination for this research, first five algorithms combination are the combination between general cutting NN and five loading heuristics, while last five algorithms combination are the combination between skip cutting NN and five loading heuristics. From all combination of algorithms, the best algorithm combination will be found in accordance with the performance measurements that has been stated. There are 3 performance measurement that will be used to find the best algorithm combination: $K$ (the number of vehicles needed), $D$ (total vehicle’s mileage), and $U$ (containers utility). Good performance are indicated from those three performance measurements value, the smaller $K$ and $D$ value, the better performance are achieved, same as the bigger $U$ value will achieve the better performance.

#### 4.2.1. Loading heuristic methods comparison

Main program which is the combination between nearest neighbor and five loading heuristic algorithms can be used to solve sequential 2L-CVRP problem properly. Summary of data test results for algorithm combination between regular cutting NN and five loading heuristics can be seen on table 3. SUM row indicated as the total of all results from performance measurements for each methods from 180 data. While AVERAGE row indicated as the average of all results from performance measurements for each methods from 180 data.

| Heuristic | Regular Cutting NN |
|-----------|---------------------|
|           | Heuristic 1 | Heuristic 2 | Heuristic 3 | Heuristic 4 | Heuristic 5 |
| SUM       | 4352 | 3954 | 3860 | 3995 | 4316 |
|           | 374742 | 352806 | 347892 | 354842 | 372566 |
| U         | 68.2 | 76.04 | 77.84 | 75.44 | 69.75 |
| AVERAGE   | 24.18 | 21.97 | 21.44 | 22.19 | 23.98 |
|           | 2082 | 1960 | 1933 | 1971 | 2070 |
|           | 38% | 42% | 43% | 42% | 39% |

Table 3 shows that from five algorithm combination between regular cutting NN and five loading heuristics, heuristic 3 (maximum touching perimeter) have the best solution from the other heuristics. Because of that, to prove that these heuristic methods is the best method, five loading heuristics are also combined with skip cutting NN algorithm that can be seen in table 4 which produce the same result as before.
Table 4. Data test results summary for algorithm combination between skip cutting NN and five loading heuristics

|               | Skip Cutting NN |               |               |               |               |
|---------------|----------------|---------------|---------------|---------------|---------------|
| SUM           | Heuristic 1   | Heuristic 2   | Heuristic 3   | Heuristic 4   | Heuristic 5   |
| K             | 3420          | 3055          | **3030**      | 3077          | 3348          |
| D             | 434026        | 416526        | **410276**    | 414222        | 430344        |
| U             | 85.12         | 95.79         | **97.08**     | 96.03         | 86.95         |
| AVERAGE       | 19            | 16.97         | **16.83**     | 17.09         | 18.6          |
|               | 2411          | 2314          | **2279**      | 2301          | 2391          |
|               | 47%           | 53%           | **54%**       | 53%           | 48%           |

Table 4 shows that from five algorithm combination between skip cutting NN and five loading heuristics, heuristic 3 (maximum touching perimeter) have the best solution from the other heuristics. Maximum touching perimeter are said as the best loading heuristics because have the smallest value of $K$ (the number of vehicles needed) and $D$ (total vehicle’s mileage), and have the biggest value of $U$ (containers utility).

4.2.2. Route determination methods comparison. There are two route determination methods, namely regular and skip cutting NN. The comparison between these two methods are summarized in table 5. For each methods of nearest neighbor, averaging five loading results are done for doing this comparison.

Table 5. Data test results summary for route determination methods

|               | Regular Cutting NN | Skip Cutting NN |
|---------------|---------------------|-----------------|
| SUM           | K 4095              | 3186            |
|               | D 360569.6          | 421078.8        |
|               | U 73.45             | 92.19           |
| AVERAGE       | K 22.75             | 17.7            |
|               | D 2003              | 2339            |
|               | U 0.408             | **0.512**       |

If viewed from the value of $K$ (the number of vehicles needed) and $U$ (containers utility), skip cutting NN has the better performance than regular cutting NN. But, if viewed from the value of $D$ (total vehicle’s mileage), regular cutting NN has the better performance than skip cutting NN.

4.2.3. Algorithms combination comparison. From comparisons above, loading heuristics have been compared at each NN, regular and skip cutting NN. Because of that, last comparison is needed, which is will compare all algorithms combination that used to solve sequential 2L-CVRP problem in this research. The comparison results between those ten algorithms combination for each performance measurement, $K$ (the number of vehicles needed), $D$ (total vehicle’s mileage), and $U$ (containers utility), can be seen in table 6.

Table 6 shows that if viewed from performance measurement $K$ (the number of vehicles needed) and $U$ (containers utility), then combination of algorithm between skip cutting nearest neighbor and maximum touching perimeter have a better performance from the others. Whereas, if viewed from performance measurement $D$ (total vehicle’s mileage), then combination of algorithm
between regular-cutting nearest neighbor and maximum touching perimeter have a better performance from the others.

Table 6. Data test results summary for sequential 2L-CVRP

| Heuristic | Regular Cutting NN | Skip Cutting NN |
|-----------|--------------------|-----------------|
| Heuristic 1 | 4352 | 3954 |
| Heuristic 2 | 3860 | 3995 |
| Heuristic 3 | 4316 | 3416 |
| Heuristic 4 | 3420 | 3055 |
| Heuristic 5 | 3030 | 3077 |
| SUM | 3482 | 3398 |
| D | 374742 | 347892 |
| U | 68.20 | 76.04 |

Comparison results from those combination of algorithms depicts that the less container is needed will affect the bigger container utility. Beside, for performance measurement $K$ and $U$, skip cutting NN method have a better performance because all customers are checked, so, the container can be used more optimal than regular cutting NN method which directly cut off the route if one of two loading constraints are not fulfilled. Otherwise, for performance measurement $D$, regular cutting NN method have a better performance because this method directly do the cut off if one of two loading constraints are not fulfilled.

5. Conclusions

In this paper, we study a generalisation of the VRP, in which the demand of customers consists of weighted, two-dimensional, rectangular items. This problem is called sequential two dimensional loading constraints capacitated vehicle routing problem (sequential 2L-CVRP) and have three performance measurements to choose the best combination of algorithms that used to test the instances. The 2L-CVRP is of particular theoretical interest as it combines two frequently studied combinatorial optimisation problems, namely the Vehicle Routing Problem, and the two-dimensional bin packing problem. Although 2L-CVRP has several real-life applications in the field of transportation logistics.

Regarding the packing features of the problem examined, our algorithm makes use of a bundle of packing heuristics, producing diverse packing structures in order to increase the probability of obtaining a feasible loading. Those bundle of packing heuristics consist of Bottom Left Fill, Maximum Touching Perimeter, and Minimum area heuristics. The routing aspects of the problem are handled by a Nearest Neighbor method that divided into two cluster, namely regular and skip cutting nearest neighbor.

The conclusion for this research is that from 10 combination algorithms, if viewed from performance measurement $K$ (the number of vehicles needed) and $U$ (containers utility), then combination of algorithm between skip cutting nearest neighbor and maximum touching perimeter have a better performance from the others. Whereas, if viewed from performance measurement $D$ (total vehicle’s mileage), then combination of algorithm between regular-cutting nearest neighbor and maximum touching perimeter have a better performance from the other method.
References

[1] Chopra S and Meindl P 2007 Supply Chain Management Strategy, Planning, and Operation, 3rd edition (New Jersey: Prentice Hall)

[2] Miltenburg J 2005 Manufacturing Strategy: How to Formulate and Implement a Winning Plan, 2nd edition (Oregon: Productivity Press)

[3] Pujawan I N and Mahendrawati E R 2010 Supply Chain Management, edisi 2 (Surabaya: Guna Widya)

[4] Miao L, Ruan Q, Woghiren K and Ruq Q 2012 A hybrid genetic algorithm for the vehicle routing problem with three-dimensional loading constraints RAIRO-Operations Research 46(01) 63-82

[5] Leung S C, Zheng J, Zhang D and Zhou X 2010 Simulated annealing for the vehicle routing problem with two-dimensional loading constraints Flexible services and manufacturing journal 22(1-2) 61-82

[6] Laporte G and Semet F 2001 Classical heuristics for the capacitated VRP The vehicle routing problem - Society for Industrial and Applied Mathematics 109-128

[7] Gendreau M, Laporte G and Potvin J Y 2001 Metaheuristics for the capacitated VRP The vehicle routing problem - Society for Industrial and Applied Mathematics 129-154

[8] Cordeau J F, Gendreau M, Laporte G, Potvin J Y and Semet F 2002 A guide to vehicle routing heuristics Journal of the Operational Research society 512-522

[9] Cordeau J F, Gendreau M, Hertz A, Laporte G and Sormany J S 2005 New heuristics for the vehicle routing problem Logistics systems: design and optimization - Springer US 279-297

[10] Gendreau M, Iori M, Laporte G and Martello S 2006 A tabu search algorithm for a routing and container loading problem Transportation Science 40(3) 342-350

[11] Fekete S P, Schepers J and Van der Veen J C 2007 An exact algorithm for higher-dimensional orthogonal packing problems Operations Research 55(3) 569-587

[12] Martello S and Vigo D 1998 Exact solution of the two-dimensional finite bin packing problem Management science 44(3) 388-399

[13] Fekete S P and Schepers J 2001 New classes of fast lower bounds for bin packing problems Mathematical Programming 115(2) 351-385
Lodi A, Martello S and Vigo D 1999 Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems *INFORMS Journal on Computing* **11**(4) 345-357

Alvarez-Valdés R, Parreño F and Tamarit J M 2005 A GRASP algorithm for constrained two-dimensional non-guillotine cutting problems *Journal of the Operational Research Society* **56**(4) 414-425

Alvarez-Valdés R, Parreño F and Tamarit J M 2007 A tabu search algorithm for a two-dimensional non-guillotine cutting problem *European Journal of Operational Research* **183**(3) 1167-1182

Khebbache S, Prins C and Yalaoui A 2008 Iterated local search algorithm for the constrained two-dimensional non-guillotine cutting problem *Journal of Industrial and Systems Engineering* **2**(3) 164-179

Iori M 2004 *Metaheuristic algorithms for combinatorial optimization problems* PhD thesis DEIS (University of Bologna)

Iori M 2005 Metaheuristic algorithms for combinatorial optimization problems *4OR* **3**(2) 163-166

Iori M, Salazar-González J J and Vigo D 2007 An exact approach for the vehicle routing problem with two-dimensional loading constraints *Transportation Science* **41**(2) 253-264

Gendreau M, Iori M, Laporte G and Martello S 2008 A Tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints *Networks* **51**(1) 4-18

Zachariadis E E, Tarantilis C D and Kiranoudis C T 2009 A guided tabu search for the vehicle routing problem with two-dimensional loading constraints *European Journal of Operational Research* **195**(3) 729-743

Chazelle B 1983 The bottom-left bin-packing heuristic: An efficient implementation *Computers, IEEE Transactions on* **100**(8) 697-707

Lodi A, Martello S and Vigo D 1999 Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems *INFORMS Journal on Computing* **11**(4) 345-357

Toth P and Vigo D 2002 The vehicle routing problem. Monographs on discrete mathematics and applications SIAM (Philadelphia)
## Appendix

### Table A1

Results obtained for the regular cutting nearest neighbor

| Instance | Class | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------|-------|-------------|-------------|-------------|-------------|-------------|
| 1        |       | 4 432 0%   | 4 432 0%   | 4 432 0%   | 4 432 0%   | 4 432 0%   |
| 2        |       | 5 498 47%  | 5 462 47%  | 5 480 47%  | 5 492 47%  | 5 444 47%  |
| 3        |       | 5 468 49%  | 5 492 49%  | 5 470 49%  | 5 464 49%  | 5 460 49%  |
| 4        |       | 6 500 47%  | 5 446 56%  | 6 472 47%  | 5 446 56%  | 6 520 47%  |
| 5        |       | 0 0 0%     | 0 0 0%     | 0 0 0%     | 0 0 0%     | 0 0 0%     |
| 1        |       | 6 494 0%   | 6 494 0%   | 6 494 0%   | 6 494 0%   | 6 494 0%   |
| 2        |       | 6 500 44%  | 6 480 52%  | 6 506 44%  | 6 480 52%  | 6 484 44%  |
| 3        |       | 7 540 42%  | 6 496 49%  | 7 540 42%  | 6 502 49%  | 6 490 49%  |
| 4        |       | 6 442 44%  | 5 416 53%  | 5 416 53%  | 5 416 53%  | 5 444 53%  |
| 5        |       | 5 456 39%  | 4 444 49%  | 3 374 65%  | 4 424 49%  | 3 400 65%  |
| 1        |       | 5 584 1%   | 5 584 1%   | 5 584 1%   | 5 584 1%   | 5 584 1%   |
| 2        |       | 8 676 43%  | 7 650 49%  | 7 618 49%  | 7 650 49%  | 7 636 49%  |
| 3        |       | 9 730 43%  | 7 650 55%  | 7 650 55%  | 7 616 55%  | 8 736 48%  |
| 4        |       | 7 584 44%  | 7 638 44%  | 7 638 44%  | 7 622 44%  | 8 644 39%  |
| 5        |       | 6 570 45%  | 5 566 54%  | 5 552 54%  | 5 566 54%  | 6 580 45%  |
| 1        |       | 7 626 0%   | 7 626 0%   | 7 626 0%   | 7 626 0%   | 7 626 0%   |
| 2        |       | 8 698 43%  | 6 586 58%  | 6 590 58%  | 6 586 58%  | 7 624 49%  |
| 3        |       | 8 690 44%  | 6 578 58%  | 6 562 58%  | 7 620 50%  | 8 714 44%  |
| 4        |       | 10 742 38% | 9 702 42%  | 9 718 42%  | 9 704 42%  | 9 714 42%  |
| 5        |       | 8 666 35%  | 7 664 40%  | 8 666 35%  | 7 664 40%  | 7 664 40%  |
| 1        |       | 1 7 640 1% | 5 640 1%   | 5 640 1%   | 5 640 1%   | 5 640 1%   |
| 2        |       | 6 620 48%  | 6 680 48%  | 5 642 58%  | 6 650 48%  | 6 692 48%  |
| 3        |       | 6 672 50%  | 5 698 60%  | 5 664 60%  | 5 628 60%  | 5 652 60%  |
| 4        |       | 7 800 43%  | 5 618 61%  | 5 670 61%  | 6 698 50%  | 5 652 61%  |
| 5        |       | 8 820 33%  | 5 608 53%  | 5 614 53%  | 6 680 44%  | 6 660 44%  |
| 1        |       | 7 678 0%   | 7 678 0%   | 7 678 0%   | 7 678 0%   | 7 678 0%   |
| 2        |       | 8 798 40%  | 8 810 40%  | 8 794 40%  | 8 810 40%  | 8 798 40%  |
| 3        |       | 6 672 63%  | 8 764 47%  | 7 712 54%  | 7 706 54%  | 10 890 38% |
| 4        |       | 9 832 48%  | 8 754 54%  | 8 808 54%  | 8 754 54%  | 8 728 54%  |
| 5        |       | 6 670 46%  | 5 588 56%  | 5 546 56%  | 5 590 56%  | 5 572 56%  |
| 1        |       | 4 1030 1%  | 4 1030 1%  | 4 1030 1%  | 4 1030 1%  | 4 1030 1%  |
| 2        |       | 7 1376 51% | 7 1418 51% | 7 1376 51% | 7 1418 51% | 7 1376 51% |
| 3        |       | 7 1354 47% | 6 1216 55% | 6 1264 55% | 6 1250 55% | 6 1294 55% |
| 4        |       | 8 1418 42% | 7 1222 48% | 7 1214 48% | 9 1340 37% | 7 1200 48% |
| 5        |       | 7 1248 42% | 5 1144 58% | 5 1132 58% | 5 1082 58% | 6 1254 49% |
| Instance | Class | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------|-------|-------------|-------------|-------------|-------------|-------------|
| 1        | 4     | 1030        | 1030        | 1030        | 1030        | 1030        |
| 2        | 7     | 1236        | 1298        | 1380        | 1170        | 1164        |
| 3        | 7     | 1370        | 1372        | 1372        | 1372        | 1372        |
| 4        | 6     | 1128        | 1368        | 1224        | 1334        | 1364        |
| 5        | 5     | 1108        | 1096        | 1086        | 964         | 1046        |
| 6        | 9     | 886         | 886         | 886         | 886         | 886         |
| 7        | 10    | 1470        | 1470        | 1470        | 1470        | 1470        |
| 8        | 9     | 936         | 880         | 800         | 838         | 856         |
| 9        | 10    | 1470        | 1470        | 1470        | 1470        | 1470        |
| 10       | 9     | 1348        | 1294        | 1168        | 1260        | 1254        |
| 11       | 5     | 1296        | 1166        | 1196        | 1262        | 1254        |
| 12       | 10    | 874         | 874         | 874         | 874         | 874         |
| 13       | 4     | 1640        | 1580        | 1594        | 1700        | 1570        |
| 14       | 10    | 1438        | 1548        | 1544        | 1466        | 1518        |
| 15       | 9     | 792         | 792         | 792         | 792         | 792         |
| 16       | 10    | 1470        | 1316        | 1254        | 1240        | 1464        |
| 17       | 9     | 936         | 900         | 840         | 964         | 1222        |
| 18       | 11    | 1418        | 1532        | 1520        | 1598        | 1518        |
| 19       | 4     | 1640        | 1580        | 1594        | 1700        | 1570        |
| 20       | 5     | 1296        | 1166        | 1196        | 1262        | 1254        |
| 21       | 1     | 874         | 874         | 874         | 874         | 874         |
| 22       | 2     | 1470        | 1316        | 1254        | 1240        | 1464        |
| 23       | 3     | 936         | 900         | 840         | 964         | 1222        |
| 24       | 12    | 1640        | 1580        | 1594        | 1700        | 1570        |
| 25       | 13    | 1470        | 1316        | 1254        | 1240        | 1464        |
| 26       | 14    | 936         | 900         | 840         | 964         | 1222        |
| Instance | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------|------------|------------|------------|------------|------------|
|          | K | D | U | K | D | U | K | D | U | K | D | U |
| 1        | 5 | 1214 | 1% | 5 | 1214 | 1% | 5 | 1214 | 1% | 5 | 1214 | 1% |
| 2        | 9 | 2062 | 50% | 8 | 1768 | 57% | 9 | 2022 | 50% | 8 | 1866 | 57% |
| 3        | 12 | 2570 | 42% | 9 | 2016 | 56% | 10 | 2234 | 50% | 9 | 2090 | 56% |
| 4        | 11 | 2264 | 52% | 10 | 2120 | 57% | 10 | 2068 | 57% | 10 | 2172 | 57% |
| 15       | 11 | 2374 | 52% | 11 | 2178 | 52% | 10 | 2118 | 57% | 11 | 2220 | 52% |
|          | 13 | 1030 | 0% | 13 | 1030 | 0% | 13 | 1030 | 0% | 13 | 1030 | 0% |
| 2        | 15 | 1128 | 41% | 16 | 1168 | 38% | 15 | 1148 | 41% | 16 | 1194 | 38% |
| 3        | 16 | 1186 | 39% | 15 | 1134 | 42% | 16 | 1166 | 39% | 15 | 1134 | 42% |
| 4        | 15 | 1120 | 47% | 15 | 1136 | 47% | 16 | 1198 | 44% | 16 | 1190 | 44% |
| 16       | 13 | 1032 | 41% | 13 | 1030 | 41% | 13 | 1030 | 41% | 13 | 1030 | 41% |
|          | 16 | 1216 | 0% | 16 | 1216 | 0% | 16 | 1216 | 0% | 16 | 1216 | 0% |
| 2        | 18 | 1310 | 36% | 18 | 1302 | 36% | 18 | 1310 | 36% | 17 | 1262 | 38% |
| 3        | 17 | 1252 | 34% | 16 | 1216 | 37% | 16 | 1206 | 37% | 17 | 1266 | 34% |
| 4        | 18 | 1336 | 40% | 16 | 1216 | 45% | 16 | 1216 | 45% | 19 | 1390 | 38% |
| 17       | 17 | 1280 | 33% | 16 | 1216 | 35% | 16 | 1240 | 35% | 16 | 1210 | 35% |
|          | 1   | 1484 | 1%  | 5  | 1484 | 1%  | 5  | 1484 | 1%  | 5  | 1484 | 1%  |
| 2        | 14 | 2034 | 46% | 12 | 1836 | 54% | 13 | 1838 | 50% | 12 | 1822 | 54% |
| 3        | 16 | 2056 | 47% | 14 | 2044 | 54% | 14 | 2042 | 54% | 13 | 1986 | 58% |
| 4        | 17 | 2374 | 46% | 14 | 2042 | 55% | 14 | 2010 | 55% | 15 | 2032 | 52% |
| 18       | 12 | 1886 | 49% | 10 | 1734 | 58% | 8  | 1434 | 73% | 10 | 1748 | 58% |
|          | 1   | 1020 | 1%  | 6  | 1020 | 1%  | 6  | 1020 | 1%  | 6  | 1020 | 1%  |
| 2        | 18 | 1522 | 47% | 17 | 1438 | 50% | 17 | 1462 | 50% | 16 | 1432 | 53% |
| 3        | 18 | 1528 | 50% | 15 | 1356 | 61% | 16 | 1506 | 57% | 16 | 1404 | 57% |
| 4        | 19 | 1544 | 50% | 20 | 1662 | 47% | 19 | 1606 | 50% | 19 | 1612 | 50% |
| 19       | 16 | 1448 | 46% | 13 | 1306 | 56% | 13 | 1268 | 56% | 13 | 1310 | 56% |
|          | 1   | 390  | 2%   | 5  | 390  | 2%   | 5  | 390  | 2%   | 5  | 390  | 2%   |
| 2        | 22 | 1034 | 54% | 22 | 1020 | 54% | 22 | 1020 | 54% | 24 | 1104 | 49% |
| 3        | 24 | 1022 | 52% | 23 | 1002 | 54% | 22 | 954  | 57% | 24 | 992  | 52% |
| 4        | 24 | 1038 | 54% | 26 | 1120 | 50% | 21 | 956  | 62% | 24 | 1066 | 54% |
| 20       | 22 | 970  | 50% | 19 | 886  | 58% | 18 | 868  | 61% | 17 | 822  | 65% |
|          | 7  | 1070 | 1%   | 7  | 1070 | 1%   | 7  | 1070 | 1%   | 7  | 1070 | 1%   |
| 2        | 26 | 2168 | 45% | 24 | 2110 | 49% | 24 | 2042 | 49% | 23 | 2060 | 51% |
| 3        | 29 | 2336 | 48% | 27 | 2348 | 51% | 26 | 2342 | 53% | 28 | 2392 | 50% |
| 4        | 25 | 2096 | 48% | 23 | 1998 | 52% | 21 | 1808 | 57% | 22 | 1988 | 54% |
| 21       | 22 | 1914 | 47% | 18 | 1704 | 57% | 18 | 1726 | 57% | 19 | 1772 | 54% |
| Instance Class | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------------|------------|------------|------------|------------|------------|
| 1              | K 8       | 8          | K 1184 1%  | K 1184 1%  | K 1184 1%  |
| 2              | D 26      | 2142 47%   | D 1930 56% | D 1974 53% | D 1832 59% |
|                | K 26      | 2082 50%   | 2148 50%   | 2012 57%   | 2024 57%   |
| 3              | U 31      | 2376 45%   | 2018 56%   | 2082 54%   | 2122 52%   |
|                | 23        | 2130 49%   | 19        | 1770 59%   | 1738 53%   |
| 4              | 5         | 23103 49%  | 19        | 1770 59%   | 1843 53%   |
| 22             | 5         | 23        | 1940 51%   | 22        | 2274 50%   |
| 1              | 11        | 1294 1%    | 1294 1%    | 1294 1%    | 1294 1%    |
| 2              | 26        | 2188 46%   | 2274 46%   | 2088 50%   | 2178 48%   |
|                | 25        | 2168 53%   | 2068 55%   | 2050 55%   | 2048 58%   |
| 3              | 27        | 2228 49%   | 2138 53%   | 2088 53%   | 2132 56%   |
|                | 23        | 1928 50%   | 1843 52%   | 1792 57%   | 1940 51%   |
| 23             | 5         | 23        | 2036 46%   | 22        | 2156 48%   |
| 1              | 16        | 1658 1%    | 1658 1%    | 1658 1%    | 1658 1%    |
| 2              | 26        | 2196 53%   | 2184 53%   | 2194 51%   | 2216 51%   |
|                | 28        | 2198 47%   | 2042 54%   | 2052 54%   | 2042 54%   |
| 3              | 27        | 2468 45%   | 2006 58%   | 2082 53%   | 2132 56%   |
|                | 23        | 1948 51%   | 1808 59%   | 1816 62%   | 1878 56%   |
| 24             | 5         | 23        | 2036 46%   | 22        | 2156 48%   |
| 1              | 8         | 1478 2%    | 1478 2%    | 1478 2%    | 1478 2%    |
| 2              | 26        | 2784 50%   | 2784 51%   | 2798 51%   | 2888 50%   |
|                | 33        | 2944 46%   | 3028 48%   | 2700 56%   | 2952 48%   |
| 3              | 27        | 3022 49%   | 3038 51%   | 2710 59%   | 2990 51%   |
|                | 25        | 2572 49%   | 2354 57%   | 2256 60%   | 2462 53%   |
| 25             | 5         | 25        | 2448 51%   | 2572 49%   | 2354 57%   |
| 1              | 10        | 1362 1%    | 1362 1%    | 1362 1%    | 1362 1%    |
| 2              | 32        | 2784 47%   | 2574 55%   | 2800 50%   | 2684 53%   |
|                | 33        | 2942 50%   | 2564 59%   | 2582 59%   | 2512 59%   |
| 3              | 34        | 2868 51%   | 2858 53%   | 2914 55%   | 2710 55%   |
|                | 25        | 2540 50%   | 2678 58%   | 2058 68%   | 2472 62%   |
| 26             | 5         | 2572 51%   | 2368 58%   | 2472 53%   | 2448 48%   |
| 1              | 15        | 1830 1%    | 1830 1%    | 1830 1%    | 1830 1%    |
| 2              | 25        | 2790 46%   | 2686 50%   | 2624 52%   | 2718 47%   |
|                | 30        | 2794 45%   | 2798 51%   | 2734 53%   | 2786 50%   |
| 3              | 34        | 2748 50%   | 2742 50%   | 2524 59%   | 2590 55%   |
|                | 27        | 2782 50%   | 2462 58%   | 2384 60%   | 2490 56%   |
| 27             | 5         | 2782 50%   | 2462 58%   | 2384 60%   | 2490 56%   |
| 1              | 8         | 1700 2%    | 1700 2%    | 1700 2%    | 1700 2%    |
| 2              | 34        | 5114 56%   | 5096 56%   | 5084 58%   | 5404 53%   |
|                | 45        | 6082 46%   | 5580 51%   | 5290 55%   | 5542 53%   |
| 3              | 42        | 5694 50%   | 5510 52%   | 5422 55%   | 5630 50%   |
|                | 28        | 5070 45%   | 4678 60%   | 4668 62%   | 4988 57%   |
| 28             | 5         | 5070 45%   | 4678 60%   | 4668 62%   | 4988 57%   |
**Table A1 continued**

| Instance Class | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------------|------------|------------|------------|------------|------------|
| 1              | 8 1972 2% | 8 1972 2% | 8 1972 2% | 8 1972 2% | 8 1972 2% |
| 2              | 39 4366 52% | 36 4210 57% | 38 4294 54% | 38 4272 54% | 38 4376 54% |
| 3              | 47 4650 46% | 39 4280 55% | 40 4292 54% | 41 4470 52% | 43 4388 50% |
| 4              | 47 4732 51% | 42 4580 57% | 44 4858 54% | 44 4410 54% | 51 5194 47% |
| 29             | 46 4794 45% | 33 3918 63% | 34 4130 61% | 36 4100 58% | 38 4224 55% |
| 1              | 12 1730 2% | 12 1730 2% | 12 1730 2% | 12 1730 2% | 12 1730 2% |
| 2              | 48 3782 50% | 47 3812 51% | 45 3674 54% | 45 3632 54% | 50 3888 48% |
| 3              | 53 4126 49% | 52 4110 50% | 49 3828 53% | 49 3864 53% | 53 4064 49% |
| 4              | 51 4052 51% | 49 3826 53% | 45 3864 57% | 50 3908 52% | 52 4102 50% |
| 30             | 41 3422 51% | 34 3014 62% | 33 3024 64% | 35 3134 60% | 40 3334 52% |
| 1              | 17 2072 1% | 17 2072 1% | 17 2072 1% | 17 2072 1% | 17 2072 1% |
| 2              | 67 4930 48% | 58 4296 55% | 61 4652 53% | 58 4394 55% | 67 4984 48% |
| 3              | 69 4940 49% | 63 4764 54% | 62 4626 55% | 65 4826 52% | 71 5146 48% |
| 4              | 70 5194 52% | 68 4998 53% | 63 4744 57% | 67 4904 54% | 75 5468 48% |
| 31             | 61 4638 48% | 50 3988 59% | 52 4312 56% | 49 4042 60% | 55 4364 53% |
| 1              | 17 2072 1% | 17 2072 1% | 17 2072 1% | 17 2072 1% | 17 2072 1% |
| 2              | 68 4984 47% | 63 4690 51% | 61 4718 53% | 65 4782 49% | 66 4788 49% |
| 3              | 65 4916 51% | 61 4612 54% | 59 4504 56% | 64 4800 52% | 64 4820 52% |
| 4              | 68 5060 50% | 63 4650 53% | 62 4686 54% | 64 4800 53% | 72 5258 47% |
| 32             | 57 4492 50% | 48 3958 59% | 44 3740 65% | 49 4026 58% | 53 4222 54% |
| 1              | 17 2112 1% | 17 2112 1% | 17 2112 1% | 17 2112 1% | 17 2112 1% |
| 2              | 64 4944 49% | 62 4606 51% | 61 4530 52% | 65 4896 49% | 67 5010 47% |
| 3              | 71 5228 49% | 63 4746 55% | 62 4696 56% | 64 4824 54% | 69 5182 50% |
| 4              | 75 5470 47% | 68 5018 52% | 62 4638 57% | 68 5034 52% | 70 5214 51% |
| 33             | 58 4480 50% | 48 3922 61% | 46 3772 63% | 47 3868 62% | 56 4330 52% |
| 1              | 23 1074 1% | 23 1074 1% | 23 1074 1% | 23 1074 1% | 23 1074 1% |
| 2              | 82 2542 48% | 73 2302 54% | 75 2388 52% | 76 2456 52% | 81 2458 48% |
| 3              | 87 2742 48% | 77 2428 54% | 74 2366 57% | 81 2590 52% | 86 2668 49% |
| 4              | 86 2626 50% | 80 2448 54% | 77 2390 56% | 81 2504 53% | 84 2560 51% |
| 34             | 76 2384 48% | 64 2060 57% | 62 2010 58% | 65 2084 56% | 70 2176 52% |
| 1              | 27 1128 1% | 27 1128 1% | 27 1128 1% | 27 1128 1% | 27 1128 1% |
| 2              | 79 2908 49% | 73 2700 53% | 74 2748 52% | 75 2776 51% | 84 3042 46% |
| 3              | 95 3362 45% | 78 2876 54% | 75 2738 57% | 80 2900 53% | 91 3204 47% |
| 4              | 91 3418 49% | 86 3190 52% | 80 2990 56% | 90 3326 50% | 96 3472 47% |
| 35             | 73 2742 51% | 63 2424 59% | 63 2430 59% | 61 2400 61% | 68 2614 54% |
### Table A1 continued

| Instance | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------|-------------|-------------|-------------|-------------|-------------|
|          | K D U       | K D U       | K D U       | K D U       | K D U       |
| 1        | 14 918 2%  | 14 918 2%  | 14 918 2%  | 14 918 2%  | 14 918 2%  |
| 2        | 87 3954 47%| 79 3612 51%| 79 3562 51%| 79 3658 51%| 86 3874 47%|
| 3        | 89 3976 49%| 82 3770 53%| 80 3598 55%| 81 3704 54%| 87 3928 50%|
| 4        | 86 3744 49%| 78 3428 54%| 72 3228 59%| 78 3494 54%| 88 3828 48%|
| 5        | 78 3504 49%| 63 2872 60%| 64 2972 59%| 67 3046 56%| 71 3226 53%|
| 36       |             |             |             |             |             |

### Table A2

Results obtained for the skip cutting nearest neighbor

| Instance | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------|-------------|-------------|-------------|-------------|-------------|
|          | K D U       | K D U       | K D U       | K D U       | K D U       |
| 1        | 3 472 1%   | 3 472 1%   | 3 472 1%   | 3 472 1%   | 3 472 1%   |
| 2        | 4 488 59%  | 4 550 59%  | 4 506 59%  | 3 506 78%  | 4 516 59%  |
| 3        | 5 532 49%  | 4 500 61%  | 4 526 61%  | 4 474 61%  | 4 526 61%  |
| 4        | 5 588 56%  | 5 596 56%  | 4 562 70%  | 4 576 70%  | 6 592 47%  |
| 5        | 0 0 0%     | 0 0 0%     | 0 0 0%     | 0 0 0%     | 0 0 0%     |
| 1        | 5 504 0%   | 5 504 0%   | 5 504 0%   | 5 504 0%   | 5 504 0%   |
| 2        | 5 590 52%  | 4 480 65%  | 5 590 52%  | 4 480 65%  | 5 592 52%  |
| 3        | 6 544 49%  | 5 638 59%  | 5 574 59%  | 5 592 59%  | 6 520 49%  |
| 4        | 5 584 53%  | 4 574 66%  | 5 588 53%  | 4 574 66%  | 5 536 53%  |
| 2        | 5 458 49%  | 3 424 65%  | 3 428 65%  | 3 448 65%  | 4 442 49%  |
| 1        | 4 540 1%   | 4 540 1%   | 4 540 1%   | 4 540 1%   | 4 540 1%   |
| 2        | 6 684 57%  | 6 708 57%  | 6 654 57%  | 6 718 57%  | 6 722 57%  |
| 3        | 7 776 55%  | 6 672 64%  | 6 652 64%  | 6 670 64%  | 6 678 64%  |
| 4        | 5 676 62%  | 5 722 62%  | 5 720 62%  | 5 766 62%  | 6 770 52%  |
| 3        | 5 738 54%  | 4 672 68%  | 4 558 68%  | 4 624 68%  | 5 714 54%  |
| 1        | 6 654 0%   | 6 654 0%   | 6 654 0%   | 6 654 0%   | 6 654 0%   |
| 2        | 6 680 58%  | 6 716 58%  | 6 670 58%  | 6 660 58%  | 5 702 69%  |
| 3        | 7 770 50%  | 6 676 58%  | 5 650 70%  | 6 630 58%  | 7 714 50%  |
| 4        | 10 810 38% | 9 766 42%  | 7 676 54%  | 9 770 42%  | 9 752 42%  |
| 4        | 5 666 35%  | 7 664 40%  | 8 666 35%  | 7 664 40%  | 7 664 40%  |
| 1        | 4 688 1%   | 4 688 1%   | 4 688 1%   | 4 688 1%   | 4 688 1%   |
| 2        | 5 742 58%  | 5 876 58%  | 4 862 72%  | 5 782 58%  | 6 774 48%  |
| 3        | 5 744 60%  | 5 760 60%  | 4 714 75%  | 5 752 60%  | 5 728 60%  |
| 4        | 5 888 61%  | 5 740 61%  | 5 764 61%  | 4 742 76%  | 5 912 61%  |
| 5        | 6 732 44%  | 5 672 53%  | 5 614 53%  | 5 636 53%  | 6 702 44%  |
### Table A2 continued

| Instance | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------|-------------|-------------|-------------|-------------|-------------|
| K        | D           | U           | K           | D           | U           |
| 1        | 6           | 750 0%      | 6           | 750 0%      | 6           | 750 0%      | 6           | 750 0%      |
| 2        | 8           | 752 40%     | 8           | 826 40%     | 8           | 842 40%     | 8           | 826 40%     | 8           | 808 40%     |
| 3        | 6           | 864 63%     | 6           | 918 63%     | 7           | 898 54%     | 6           | 888 63%     | 7           | 888 54%     |
| 4        | 7           | 790 61%     | 7           | 938 61%     | 7           | 926 61%     | 7           | 870 61%     | 7           | 938 61%     |
| 5        | 6           | 726 46%     | 5           | 856 56%     | 4           | 544 70%     | 4           | 726 70%     | 5           | 778 56%     |
| 6        | 3           | 994 1%      | 3           | 994 1%      | 3           | 994 1%      | 3           | 994 1%      | 3           | 994 1%      |
| 7        | 5           | 1402 58%    | 4           | 1296 56%    | 5           | 1326 67%    | 6           | 1404 56%    | 6           | 1502 56%    |
| 8        | 4           | 1288 56%    | 4           | 1288 56%    | 4           | 1288 56%    | 3           | 1092 75%    | 4           | 1148 56%    |
| 9        | 8           | 1114 53%    | 6           | 972 71%     | 6           | 1038 71%    | 6           | 900 71%     | 6           | 1042 71%    |
| 10       | 9           | 1590 49%    | 6           | 1328 73%    | 6           | 1146 73%    | 6           | 1394 73%    | 7           | 1666 63%    |
| 11       | 8           | 1610 55%    | 7           | 1416 63%    | 7           | 1382 63%    | 7           | 1440 63%    | 7           | 1628 63%    |
| 12       | 5           | 1074 58%    | 8           | 1016 66%    | 8           | 1160 66%    | 8           | 1030 66%    | 9           | 1046 58%    |
| Instance | Class | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------|-------|-------------|-------------|-------------|-------------|-------------|
| 1        | 3     | 4106 1%    | 4106 1%    | 4106 1%    | 4106 1%    | 4106 1%    |
| 2        | 8     | 6752 61%   | 6774 61%   | 6306 69%   | 5772 69%   | 6494 54%   |
| 3        | 8     | 4706 59%   | 5814 68%   | 5576 68%   | 5324 68%   | 5570 59%   |
| 4        | 9     | 5698 60%   | 6118 60%   | 6332 67%   | 5750 67%   | 5798 67%   |
| 13       | 5     | 5880 59%   | 6038 67%   | 6180 67%   | 6088 67%   | 5674 59%   |
| 14       | 5     | 1898 56%   | 1736 65%   | 1624 65%   | 1726 65%   | 1770 65%   |
| 15       | 9     | 2192 64%   | 2262 72%   | 2254 64%   | 2156 64%   | 2154 53%   |
| 16       | 5     | 13044 41%  | 1160 44%   | 1172 44%   | 1138 44%   | 1206 41%   |
| 17       | 5     | 1286 37%   | 1350 37%   | 1298 40%   | 1350 37%   | 1298 40%   |
| 18       | 5     | 2194 65%   | 2156 73%   | 1882 73%   | 2396 65%   | 2110 73%   |

Table A2 continued
Table A2 continued

| Instance Class | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------------|------------|------------|------------|------------|------------|
| 1              | 4          | 4          | 4          | 4          | 4          |
| 2              | 17         | 17         | 17         | 17         | 17         |
| 3              | 20         | 17         | 17         | 17         | 17         |
| 4              | 20         | 18         | 18         | 18         | 18         |
|                | 20         | 20         | 20         | 20         | 20         |
| 5              | 18         | 1100       | 15         | 1140       | 19         |
|                | 17         | 1252       | 17         | 1274       | 17         |
|                | 20         | 1262       | 18         | 1246       | 17         |
|                | 20         | 1260       | 15         | 1120       | 15         |
| 21             | 1           | 1384       | 8          | 1384       | 8          |
|                | 2           | 2602       | 23         | 2524       | 23         |
|                | 3           | 2806       | 20         | 2798       | 20         |
|                | 4           | 2842       | 18         | 2838       | 17         |
|                | 22          | 2616       | 14         | 2390       | 14         |
|                |             |             |             |             |             |
| 22             | 1           | 1764       | 10         | 1764       | 10         |
|                | 2           | 2752       | 19         | 2668       | 19         |
|                | 3           | 2780       | 18         | 2664       | 18         |
|                | 4           | 2800       | 14         | 2742       | 14         |
|                | 21          | 2600       | 16         | 2296       | 16         |
|                |             |             |             |             |             |
| 23             | 1           | 1886       | 14         | 1886       | 14         |
|                | 2           | 2726       | 20         | 2516       | 19         |
|                | 3           | 2846       | 18         | 2524       | 19         |
|                | 4           | 2688       | 19         | 2814       | 19         |
|                | 23          | 2460       | 16         | 2642       | 16         |
|                |             |             |             |             |             |
| 24             | 1           | 1596       | 8          | 1596       | 8          |
|                | 2           | 3416       | 24         | 3334       | 24         |
|                | 3           | 3654       | 25         | 3448       | 24         |
|                | 4           | 3652       | 27         | 3762       | 25         |
|                | 25          | 3538       | 19         | 3186       | 19         |
|                |             |             |             |             |             |
| 25             | 1           | 1402       | 10         | 1402       | 10         |
|                | 2           | 3274       | 21         | 3274       | 22         |
|                | 3           | 3726       | 24         | 3762       | 23         |
|                | 4           | 3692       | 24         | 3546       | 23         |
|                | 26          | 3098       | 20         | 2894       | 19         |
Table A2 continued

| Instance Class | Kontainer 1 K D U | Kontainer 2 K D U | Kontainer 3 K D U | Kontainer 4 K D U | Kontainer 5 K D U |
|----------------|------------------|------------------|------------------|------------------|------------------|
| 1              | 14 2346 1%       | 14 2346 1%       | 14 2346 1%       | 14 2346 1%       | 14 2346 1%       |
| 2              | 25 3444 64%      | 22 2918 73%      | 23 3254 70%      | 23 3330 70%      | 24 3366 67%      |
| 3              | 29 3878 62%      | 25 3882 72%      | 25 3776 72%      | 25 3664 72%      | 30 3982 60%      |
| 4              | 26 3478 66%      | 23 3658 74%      | 23 3580 74%      | 24 3796 71%      | 26 3702 66%      |
| 27             | 5 3256 63%       | 21 2934 72%      | 22 3432 71%      | 21 3116 74%      | 24 3422 65%      |
| 1              | 7 1824 2%        | 7 1824 2%        | 7 1824 2%        | 7 1824 2%        | 7 1824 2%        |
| 2              | 27 5574 71%      | 26 5296 74%      | 26 5320 74%      | 25 5000 76%      | 28 5980 68%      |
| 3              | 33 6256 63%      | 28 5606 74%      | 29 6076 71%      | 28 6218 74%      | 32 6118 65%      |
| 4              | 33 7532 63%      | 29 6982 72%      | 29 6642 72%      | 29 7026 72%      | 30 6682 70%      |
| 28             | 5 6310 60%       | 23 5064 79%      | 23 5178 79%      | 24 5826 75%      | 27 5920 67%      |
| 1              | 7 2444 2%        | 7 2444 2%        | 7 2444 2%        | 7 2444 2%        | 7 2444 2%        |
| 2              | 30 5244 68%      | 28 5272 73%      | 28 5686 73%      | 29 5262 70%      | 31 5654 66%      |
| 3              | 34 6154 63%      | 29 5102 74%      | 28 4832 77%      | 29 5320 74%      | 33 5980 65%      |
| 4              | 36 5952 66%      | 31 6114 77%      | 32 5438 75%      | 32 5800 75%      | 35 6126 68%      |
| 29             | 5 5470 60%       | 27 4362 77%      | 28 4868 74%      | 28 4602 74%      | 31 4732 67%      |
| 1              | 12 2190 2%       | 12 2190 2%       | 12 2190 2%       | 12 2190 2%       | 12 2190 2%       |
| 2              | 35 4728 69%      | 32 4170 75%      | 31 4044 78%      | 33 4290 73%      | 35 4722 69%      |
| 3              | 39 4734 67%      | 34 4854 77%      | 37 4750 70%      | 35 4622 74%      | 38 4938 69%      |
| 4              | 39 5102 66%      | 35 5044 74%      | 33 4720 79%      | 36 4778 72%      | 38 5346 68%      |
| 30             | 5 4308 66%       | 27 4026 78%      | 28 4172 75%      | 28 4370 75%      | 31 4432 68%      |
| 1              | 16 2620 2%       | 16 2620 2%       | 16 2620 2%       | 16 2620 2%       | 16 2620 2%       |
| 2              | 46 5624 70%      | 43 5282 75%      | 42 5300 76%      | 42 5412 76%      | 46 5542 70%      |
| 3              | 51 6488 67%      | 46 5998 74%      | 45 6114 76%      | 46 5664 74%      | 49 5726 70%      |
| 4              | 53 6484 68%      | 48 6242 75%      | 46 6094 78%      | 49 5892 74%      | 55 6240 66%      |
| 31             | 5 4770 57%       | 39 4104 75%      | 39 4232 75%      | 41 4264 71%      | 46 4684 64%      |
| 1              | 16 2620 2%       | 16 2620 2%       | 16 2620 2%       | 16 2620 2%       | 16 2620 2%       |
| 2              | 44 5348 73%      | 42 5488 76%      | 41 5326 78%      | 43 5908 75%      | 45 5668 71%      |
| 3              | 49 6326 67%      | 43 5896 77%      | 43 5700 77%      | 43 5782 77%      | 48 6068 69%      |
| 4              | 49 6218 69%      | 44 6044 77%      | 43 5814 78%      | 45 5810 75%      | 50 6102 67%      |
| 32             | 5 4894 58%       | 39 4308 73%      | 38 4004 75%      | 40 4142 71%      | 46 4532 62%      |
| 1              | 17 2112 1%       | 17 2112 1%       | 17 2112 1%       | 17 2112 1%       | 17 2112 1%       |
| 2              | 49 4574 64%      | 46 4488 69%      | 45 4560 70%      | 45 4412 70%      | 50 4782 63%      |
| 3              | 53 6710 65%      | 46 5712 75%      | 46 5874 75%      | 46 5864 75%      | 51 5928 68%      |
| 4              | 52 6662 68%      | 47 6588 76%      | 47 6038 76%      | 47 6656 76%      | 54 6524 66%      |
| 33             | 5 48 6126 61%    | 40 4260 73%      | 39 4220 75%      | 40 4178 73%      | 45 4350 65%      |
Table A2 continued

| Instance | Class | Kontainer 1 | Kontainer 2 | Kontainer 3 | Kontainer 4 | Kontainer 5 |
|----------|-------|-------------|-------------|-------------|-------------|-------------|
|          |       | K | D | U | K | D | U | K | D | U | K | D | U |
| 1        | 22    | 1192 | 1% | 22 | 1192 | 1% | 22 | 1192 | 1% | 22 | 1192 | 1% |
| 2        | 62    | 2356 | 63% | 58 | 2194 | 68% | 57 | 2220 | 69% | 58 | 2282 | 68% |
| 3        | 69    | 2538 | 61% | 60 | 2266 | 70% | 61 | 2360 | 69% | 63 | 2400 | 67% |
| 4        | 68    | 2474 | 63% | 62 | 2482 | 70% | 61 | 2404 | 71% | 63 | 2494 | 68% |
| 34       | 5     | 62   | 2364 | 58% | 51 | 2106 | 71% | 49 | 2076 | 74% | 52 | 2134 | 70% |
| 1        | 26    | 1354 | 1% | 26 | 1354 | 1% | 26 | 1354 | 1% | 26 | 1354 | 1% |
| 2        | 57    | 3740 | 67% | 50 | 3524 | 77% | 51 | 3494 | 75% | 51 | 3676 | 75% |
| 3        | 67    | 4222 | 63% | 58 | 4138 | 73% | 56 | 3830 | 76% | 58 | 4178 | 73% |
| 4        | 64    | 4464 | 70% | 59 | 4226 | 76% | 56 | 4118 | 80% | 60 | 4220 | 75% |
| 35       | 5     | 58   | 3866 | 64% | 46 | 3812 | 80% | 47 | 3364 | 79% | 46 | 3834 | 80% |
| 1        | 14    | 1062 | 2% | 14 | 1062 | 2% | 14 | 1062 | 2% | 14 | 1062 | 2% |
| 2        | 59    | 3988 | 69% | 53 | 3692 | 77% | 53 | 3514 | 77% | 54 | 3562 | 75% |
| 3        | 66    | 4188 | 66% | 60 | 3858 | 73% | 58 | 3840 | 75% | 58 | 3906 | 75% |
| 4        | 64    | 4112 | 66% | 55 | 3822 | 77% | 55 | 3658 | 77% | 56 | 3706 | 75% |
| 36       | 5     | 64   | 3552 | 59% | 51 | 2982 | 74% | 50 | 2964 | 76% | 53 | 3056 | 71% |