Evaluation of Algorithm Performance for Simulated Square and Non-Square Logistic Assignment Problems

Maximilian Selmair
Sascha Hamzehi

BMW Group
Email: maximilian.selmair@bmw.de

Klaus-Jürgen Meier

University of Applied Sciences Munich
Email: klaus-juergen.meier@hm.edu

***Abstract*—The optimal allocation of transportation tasks to a fleet of vehicles, especially for large-scale systems of more than 20 Autonomous Mobile Robots (AMRs), remains a major challenge in logistics. Optimal in this context refers to two criteria: how close a result is to the best achievable objective value and the shortest possible computational time. Operations research has provided different methods that can be applied to solve this assignment problem. Our literature review has revealed six commonly applied methods to solve this problem. In this paper, we compared three optimal methods (Integer Linear Programming, Hungarian Method and the Jonker Volgenant Castanon algorithm) to three three heuristic methods (Greedy Search algorithm, Vogel's Approximation Method and Vogel's Approximation Method for non-quadratic Matrices). The latter group generally yield results faster, but were not developed to provide optimal results in terms of the optimal objective value. Every method was applied to 20.000 randomised samples of matrices, which differed in scale and configuration, in simulation experiments in order to determine the results' proximity to the optimal solution as well as their computational time. The simulation results demonstrate that all methods vary in their time needed to solve the assignment problem scenarios as well as in the respective quality of the solution. Based on these results yielded by computing quadratic and non-quadratic matrices of different scales, we have concluded that the Jonker Volgenant Castanon algorithm is deemed to be the best method for solving quadratic and non-quadratic assignment problems with optimal precision. However, if performance in terms of computational time is prioritised for large non-quadratic matrices (50×300 and larger), the Vogel's Approximation Method for non-quadratic Matrices generates faster approximated solutions.**

***Keywords*—Assignment Problem; Jonker Volgenant Castanon; Vogel's Approximation Method; Hungarian Method; Greedy Search; Autonomous Mobile Robots;**

I. Introduction

The transportation of goods and materials, which can take place within a plant as well as to and from it, is an integral part of every supply chain and contributes substantially to an enterprise's expenses. As such, every logistics manager who has to deal with matters of transport aims to find the most cost-efficient solution while meeting high customer service standards, i. e. meeting demands at the lowest possible expense. This is referred to as the Assignment Problem (AP) and has been the subject of extensive operational research (Díaz-Parra et al., 2014). Researchers, who attempt to solve this problem, also aim to minimise the total transportation costs whilst moving goods from several supply points (e. g. warehouses) to demand locations (e. g. customers). Theoretical and practical use cases are based on two conditions. Each transport origin features a fixed amount of goods that can be distributed. Correspondingly, every point of transport destination requires a certain number of goods (Shore, 1970).

Each point of origin can be connected to every designated destination, thereby creating several routes that are often associated with different costs. These varying costs can be visualised in matrices, where each cell represents a potential effort. Depending on the objective of the optimisation, these efforts can be a monetary value, the transport distance or the duration that an agent requires to fulfil a corresponding task. Matrices like this are common practice in studies dealing with the AP and are used to make allocation decisions.

The underlying use case, where tasks have to be assigned to vehicles of a fleet consisting of AMRs, differs from the classical AP. In our case, each AMR has a capacity restriction of 1, i. e. a maximum of one load carrier can be transported at a time. Furthermore, each task corresponds to a demand of 1. This means that every task can only be allocated to one single AMR. Additionally, the number of available AMRs rarely matches the number of to-be-assigned tasks in practice. Since the size of the matrices depends on this factor, non-quadratic matrices (e. g. 40×50) are common. There are different approaches to solve this particular assignment problem, e. g. Integer Linear Programming, machine-learning-based methods or the application of algorithms (see section II).

For the latter approach, algorithms, one needs to distinguish between those that solve every problem scenario optimally, here referred to as optimal methods, and heuristics, which in some cases fail to find the best solution and instead result in approximations. Furthermore, all methods vary greatly in terms of the computation time required to solve a given assignment problem.

This paper is structured as follows: the next section provides an overview of the different methods that can be applied to solving the AP. These being Integer Linear Programming (ILP), the Hungarian Method (HM), the Jonker Volgenant Castanon (JVC) algorithm, a variation of a Greedy Search (GS) algorithm and Vogel's Approximation Method (VAM) with its associated adapted version for non-quadratic matrices, VAM-nq. The third

Communications of the ECMS, Volume 35, Issue 1,
Proceedings, ©ECMS Khalid Al-Begain, Mauro Iacono,
Lelio Campanile, Andrzej Bargiela (Editors)
ISBN: 978-3-937436-72-2 / 78-3-937436-73-9(CD) ISSN 2522-2414

section introduces the use case that gave rise to this study. The simulation study and the discussion of the results are presented in section IV, which is followed by the conclusion.

II. The History of Solving the Assignment Problem

The Generalised Assignment Problem (GAP) describes the generalisation of both the knapsack problem and the bin packing problem (Cattrysse and van Wassenhove, 1992). In its original characteristic, one is given m agents with a corresponding capacity B_j for each agent j, and n tasks such that each task i has size s_{ij} and yields profit p_{ij} when assigned to agent j. The objective is to find an optimal assignment of agents to tasks by maximising the total profit. Applications of GAP can be found in multiple areas including fixed charge location problems, grouping and loading for vehicle routing, flexible manufacturing systems, scheduling variable length commercials, allocating storage space, designing communication networks, scheduling payments on accounts, assigning software development tasks to programmers, scheduling projects, assigning jobs to computers in networks, and assigning ships to overhaul facilities (Krumke and Thielen, 2013; Cattrysse and van Wassenhove, 1992).

In this paper, we consider a variation of the problem in which all the agents' budgets and all tasks' costs are equal to 1 (Pentico, 2007). This section provides a description of the established solution methods for the AP. The first attempt at solving the AP dates back to 1955 with Kuhn's Hungarian Method. In the subsequent decades, numerous researchers proposed a multitude of methods and solutions as listed hereafter in their chronological order: (Kuhn, 1955; Munkres, 1957; Reinfeld and Vogel, 1958; Witzgall and Zahn, 1965; Edmonds, Johnson and Lockhart, 1969; Dinic and Kronrod, 1969; Hopcroft and Karp, 1973; Pulleyblank, 1973; Gabow, 1976; Lawler, 1976; Karzanov, 1976; Cunningham and Marsh, 1978; Micali and Vazirani, 1980; Burkard and Derigs, 1980; Kazakidis, 1980; Derigs, 1981; Shimshak, Kaslik and Barclay, 1981; Galil, Micali and Gabow, 1982; Minoux, 1982; Havel, Kuntz and Crippen, 1983; Gabow, 1985; Grötschel and Holland, 1985; Derigs, 1986; Derigs and Metz, 1986; Trick, 1987; Jonker and Volgenant, 1987; Derigs, 1988; Lessard, Rousseau and Minoux, 1989; Gabow, Galil and Spencer, 1989; Gabow, 1990; Gabow and Tarjan, 1991; Derigs and Metz, 1991; Gerngross, 1991; Applegate and Cook, 1993; Atamtürk, 1993; Miller and Pekny, 1995; Feder and Motwani, 1995; Cheriyan, Hagerup and Mehlhorn, 1996; Goldberg and Kennedy, 1997; Goldberg and Karzanov, 2004; Mucha and Sankowski, 2004; Harvey, 2006; Duan and Pettie, 2014; Cygan, Gabow and Sankowski, 2015; Gabow, 2017; Selmair, Meier and Wang, 2019)

The analysis of the relevant literature has yielded a total of four methods that are frequently applied in research (Paul, 2018; Dinagar and Keerthivasan, 2018; Nahar, Rusyaman and Putri, 2018; Ahmed et al., 2016; Korukoğlu and Ballı, 2011; Freling, Wagelmans and Paixão, 2001; Li, Mirchandani and Borenstein, 2007; Balakrishnan, 1990). These are Integer Linear Programming (ILP), the Hungarian Method (HM), Vogel's Approximation Method (VAM) and the Jonker Volgenant Castanon (JVC) algorithm. For the purpose of this paper, we chose the three optimal methods – ILP, HM and the JVC algorithm – and three heuristic approximation methods: Vogel's Approximation Method with its variation for non-quadratic matrices, VAM-nq, as well as a trivial Greedy Search algorithm for benchmark reasons. In the following, each method will be described in detail.

A. Integer Linear Programming

As stated previously, ILP is one if the approaches that is able to ascertain an optimal solution for different, even large-scale scenarios. Its application begins by formulating an objective function as well as applicable restrictions in order to receive correct results. In accordance with Osman (1995) and by adapting ILP to the use case at hand, the objective function reads as follows:

$$\min \sum_{t \in T} \sum_{a \in A} d_{ta} \cdot c_{ta} \tag{1}$$

$$\sum_{a \in A} d_{ta} = 1 \qquad \forall t \in T \tag{2}$$

$$\sum_{t \in T} d_{ta} \leq 1 \qquad \forall a \in A \tag{3}$$

$$d_{ta} \in \{0, 1\} \qquad \forall t \in T, \forall a \in A \tag{4}$$

The goal of the objective function 1 is to minimise the sum of all costs c_{ja} for all tasks $T = 1, \ldots, m$ and for all agents $A = 1, \ldots, n$, which is the result of multiplying the decision variable d_{ja} by the corresponding costs that arise when a task t is assigned to an agent a. For the underlying use case, an agent refers to an AMR. The first Constraint (Equation 2) ensures that every task is actually assigned to an agent whereas the second Constraint (Equation 3) makes sure that each agent's capacity of 1 is not exceeded, i. e. each agent can only perform one task at a time. The last Constraint (Equation 4), applies to both tasks and agents and restricts the decision variable d_{ja} to binary values.

B. Hungarian Method

The Hungarian Method was initially proposed by Kuhn (1955) to solve the AP. Similar to ILP and JVC, the HM is able to find an optimal solution to any given problem. The algorithm solves $n \times n$ matrices (e. g. 10×10) by carrying out the following steps until an optimum solution is found:

1) Identify the minimum value in each column and subtract this value from all other values in the corresponding column.
2) Identify the minimum value in each row and subtract this value from all other values in the corresponding row.
3) All zeros in the matrix must be covered by marking as few rows and/or columns as possible.
4) Check if the number of lines equals n. If it does, an optimal allocation of the zero-values is possible. If the number of lines is smaller than n, an optimal allocation is not yet feasible and Step 5 has to be carried out.

5) Find the smallest value which is not covered by a line and a) subtract this value from each uncovered row and b) add it to each covered column.
6) Return to Step 3.

It has to be noted that non-quadratic matrices, $n \times m$ matrices (e. g. 10×40), are converted to $n \times n$ matrices (e. g. 40×40), as the Hungarian Method can only be applied to quadratic matrices. The additional value cells are allocated with the highest value identified in the original matrix. This adaptation requires additional computing power since the algorithm has to consider, for instance, 1.600 cells of values (40×40) instead of 400 (10×40). It is evident that this is a drawback when non-quadratic matrices are to be solved, yet this is always the case when more tasks than agents have to be considered or vice versa.

C. Jonker Volgenant Castanon Algorithm

Skiena (1990) defined the augmenting path as a simple path – thus, a path that does not contain cycles – through a network using only edges with positive capacity from the source to the sink. Compared to the Hungarian Method, which finds any augmenting paths out of all feasible ones, the JVC algorithm finds the shortest augmenting path among all options. Although the JVC algorithm is based on the implementation of the Hungarian Method, the following additional pre-processing steps are required:

1) *initialization step* with sub-steps such as a) column reduction, b) reduction transfer, c) reduction of unassigned rows, similar to auction algorithms,
2) *termination step*, if row assignment is complete,
3) *augmentation step*, where an auxiliary network is generated to determine an alternating path with minimal total objective cost, from unassigned row i to unassigned column j,
4) *update step of dual solution*, where the complementary slackness variables are updated, and finally a
5) *return step to the termination Step 2.*

where the algorithm output is an integer sequence that assigns each column to each corresponding minimal row element denoted by $v_j = \min_i(c_{ij})$. Note, that some rows may not be assigned, which is particularly important for the processing of non-quadratic matrices. Moreover, the JVC algorithm minimises the primary objective:

$$\min \sum_i \sum_j c_{ij} x_{ij}; \tag{5}$$

subject to the constraints:

$$\sum_i x_{ij} = 1, \; \sum_j x_{ij} = 1 \tag{6}$$

$$x_{ij} \leq 0, \; \forall (i,j) \in E \; , \tag{7}$$

and maximises the dual objective as follows:

$$\max \Big\{ \sum_i u_i + \sum_j v_j \Big\} \text{ s.t. } c_{ij} - u_i - v_j \leq 0 \; . \tag{8}$$

where the reduction transfer is completed from unassigned to assigned rows where for each row i.

$$j_1 = x_i; \tag{9}$$

$$\mu = \min\{c_{ij} - v_j : j = 1, ..., n; \forall j \neq j - 1\} \tag{10}$$

$$v_{j1} = v_{j1} - (\mu - u_i) \, ; u_i = \mu \tag{11}$$

After augmentation Step 3, the partial solution assignments are updated, while the dual values are updated to restore the complementary slackness conditions in the following:

$$c_{ik} - u_i - v_k = 0, \text{if } x_i = k \; \forall \text{ assigned columns and} \tag{12}$$

$$k, i = 1, .., n \text{ and} \tag{13}$$

$$c_{ik} - u_i - v_k \leq 0 \tag{14}$$

Generally, the complexity of the first two initialisation procedures is reported to be $\mathcal{O}(n^2)$, whereas it has been shown that the augmenting reduction procedure has a complexity of $\mathcal{O}(R \cdot n^2)$. Here, R refers to the range of the cost coefficients (Jonker and Volgenant, 1987).

D. Greedy Search Heuristic

In order to compare the optimal method algorithms to a simple heuristic, we have used a Greedy Search method, which is described as follows:

1) Input the cost matrix C and determine the maximum cost value m_c,
2) while a minimum value can be found that is lower than m_c, locate the minimum value across all rows i and columns j,
3) update the objective value o_c^k, of current iteration k by adding its own value to the previous iterative objective value o_c^{k-1}.
4) for all columns j, locate the minimum value row and column position (i_{min}, j_{min}), assign the task to the assignment or matching list, and set the minimum value to maximum cost m_c for all residual row entries,
5) for all rows i, locate the minimum value, assign the task, and set the minimum value to the maximum cost m_c for all residual column entries,
6) until no minimum values are left that are smaller than the maximum value m_c.

E. Vogel's Approximation Method and VAM-nq

The following description of Vogel's Approximation Method is based on the original proposal by Reinfeld and Vogel (1958). VAM solves transport matrices by repeating the steps presented below until a feasible solution is found. The cells of the matrices are assigned with the costs c_{ij} associated with allocating a task to an agent. These costs occur when an agent transports goods from a point of origin i to a destination j. Depending on the objective of the optimisation, c_{ij} can be a monetary value, the transport distance or the duration that an agent requires to fulfil the corresponding task. Each source (origin) features a specific amount of goods that can be allocated (supply). Correspondingly, each sink (destination) usually requires a certain number of units (demand). The underlying case is special in regard to supply and demand. Each agent can only

provide a supply of 1. Correspondingly, each task equals a demand of 1. As mentioned earlier, this fact describes the special case of the Generalised Assignment Problem, refereed to the Assignment Problem. In order to carry out the allocation under these circumstances, the following steps are necessary:

1) Calculate the difference between the smallest and the second-smallest cell value for each row and each column.
2) Select the row or column which features the biggest difference. If the calculations yield the same value in two or more columns or rows, select the row or column containing the smallest cell value.
3) Choose the smallest cell value of the selected row or column and allocate the corresponding task to an agent.
4) Eliminate the row and column that has been used for the allocation.
5) Check if there are still agents and tasks left to allocate. If so, repeat Steps 1 - 4.

Different authors have previously attempted to improve the classic VAM in order to progress towards an optimal solution, which is generally achieved by applying either ILP or the HM. These are, for instance, (Paul, 2018; Dinagar and Keerthivasan, 2018; Nahar, Rusyaman and Putri, 2018; Ahmed et al., 2016; Korukoğlu and Ballı, 2011; Balakrishnan, 1990; Goyal, 1984; Shimshak, Kaslik and Barclay, 1981).

An example of the VAM can be found in Tables I a through c. Here, the agents V_i are assigned to the different rows and the tasks T_j to the columns. The individual cells show the costs c_{ij} for each possible task-agent combination. The row differences, which refer to those between the least and the second-least expensive option, can be found in the column Δi, whereas the column differences are shown in Δj. Table I a shows that the most substantial difference is associated with the third row, which features the lowest value in the third column (Table I b). Accordingly, Task 3 is assigned to Agent 3. After the allocation, the third row and column are eliminated (Table I c).

Selmair, Meier and Wang (2019) have shown that the original VAM is not always suitable when it comes to determining optimal or near-optimal results for non-quadratic matrices (see Figure 5 a). In fact, the calculated objective values are in some cases more than 100 % higher than the optimal objective value. For the purpose of comprehensibility, any matrix is described as having two dimensions, yet these are not identical in length for non-quadratic matrices. In line with this, it was proposed that these insufficient results arise if either the selected row or column, which features the maximum difference, is from the shorter dimension. This may mean that if a matrix contains more columns than rows, choosing a column with the maximum difference (which is achieved by subtracting cell values in the smaller dimension / rows) might result in values that deviate from the optimal objective value. The same also applies if there are more rows than columns. This can be explained by the fact that the dimension of rows features more values and the chance is therefore higher to find a smaller cell value within those. In order to mitigate the above stated disadvantage of VAM, an improved version of VAM was developed.

Figure 5 a shows that the results of the VAM start to deteriorate as soon as the matrix size is increased in only one dimension, i. e. a non-quadratic matrix is created. While VAM is able to generate optimal solutions in some cases, however, in those instances in which it fails, the calculated objective values are up to 5 times higher than the optimal objective value. The deviations from the optimal solution increase continuously as the difference between the number of rows and columns increases. This even applies to small non-quadratic instances with the dimensions of 4×5. For instance, in the case of 5×10 matrices, the calculated objective values can be three times as high as the actual optimal objective value.

Selmair, Meier and Wang (2019) provided an enhanced version of VAM, namely Vogel's Approximation Method for non-quadratic Matrices, which yields results that are much closer to the optimal objective value for said non-quadratic matrices. The description below is based on a scenario in which a matrix contains more columns than rows. If a matrix were to contain more rows than columns, these steps can be adapted accordingly by replacing "rows" with "columns" and vice versa. The Vogel's Approximation Method for non-quadratic Matrices (VAM-nq) solves allocation matrices featuring more columns than rows by carrying out the following steps:

1) Calculate the difference between the smallest and the second-smallest cell value for each row.
2) Select the row featuring the biggest difference. If there is a tie between rows, choose the row containing the smallest cell value.
3) Determine the smallest cell value for the selected row and allocate the corresponding task to an agent.
4) Eliminate the corresponding row and column that have been used for the allocation.
5) Check if there are still agents and tasks left to allocate, and repeat Steps 1 - 4 in case that there are.

On comparison of the original VAM and the enhanced VAM-nq, some aspects point towards a refinement of the original approach. For one, VAM-nq considers only the rows if there are more columns than rows (Step 1). Accordingly, only the largest differences within each of the rows and the corresponding smallest cell values are considered (Step 2 and 3). Applying this method to the other option, that is, to matrices that feature more rows than columns, results in the following approach. Steps 1 through 3 only apply to columns, their biggest differences within each column and smallest cell values. With Table II, the example of subsection II-E is solved by means of all three heuristics, illustrating that the proposed VAM-nq provides substantially better results even in small non-quadratic cases.

III. The Use Case at Hand

The use case presented in this study is part of a BMW project in the context of Industry 4.0, to which we are contributing. In order to automate its internal material flow, the BMW Group has developed its own AMR, the so-called Smart Transport Robot (STR) shown in Figure 1, which is designed to substitute

c_{ij}	T1	T2	T3	T4	Δi
V1	200	100	400	50	50
V2	60	80	30	350	30
V3	210	300	70	150	80
V4	120	510	340	80	40
V5	70	80	40	400	30
Δj	10	0	10	30	

(a) Initial Matrix to be solved by VAM

c_{ij}	T1	T2	T3	T4	Δi
V1	200	100	400	50	50
V2	60	80	30	350	30
V3	210	300	70	150	80
V4	120	510	340	80	40
V5	70	80	40	400	30
Δj	10	0	10	30	

(b) Matrix featuring the identified biggest difference (80)

c_{ij}	T1	T2	T3	T4	Δi
V1	200	100	400	50	50
V2	60	80	30	350	30
V3	210	300	70	150	80
V4	120	510	340	80	40
V5	70	80	40	400	10
Δj	10	0	10	30	

(c) Matrix after eliminating assigned row and column

Table I: Exemplary procedure of Vogel's Approximation Method

c_{ij}	T1	T2	T3	T4
V1	200	100	400	**50**
V2	60	80	**30**	350
V3	210	**300**	70	150
V4	120	510	340	80
V5	**70**	80	40	400

(a) Solution of Greedy Search with an objective value of 450

c_{ij}	T1	T2	T3	T4
V1	200	100	400	**50**
V2	60	80	30	350
V3	210	300	**70**	150
V4	**120**	510	340	80
V5	70	**80**	40	400

(b) Solution of the original VAM with an objective value of 320

c_{ij}	T1	T2	T3	T4
V1	200	100	400	**50**
V2	60	**80**	30	350
V3	210	300	70	150
V4	**120**	510	340	80
V5	70	80	**40**	400

(c) Solution of VAM-nq with an objective value of 290

Table II: Exemplary solutions of GS, VAM and VAM-nq with different resulting objective values

commonly used tucker trains. These play a central role in the automotive material handling processes. The industrial use-case of the BMW Group specifies the following requirements:

1) Every agent can carry one load carrier at a time.
2) All transportation requests are unknown prior to their receipt.
3) Agents without a task are required to either recharge or park.

The allocation of tasks to AMR is formulated as the AP. In this context, the costs are defined as the driving effort of each AMR to the source-point of a task. This study aims to identify the most efficient method to solve this assignment problem. Although it is highly likely that this specific problem results in non-quadratic matrices, we have also applied all methods to quadratic matrices.

Figure 1: BMW's STR in its natural habitat.

IV. Simulation Study

In order to evaluate the suitability of all previously described methods, simulations were performed to provide the necessary data. This section describes the methodology of the simulation study and presents its results.

A. Algorithm Performance Metric for Comparisons

We utilised the Mean Absolute Percentage Error (MAPE) metric, defined below, in order to measure the qualitative performances of the introduced methods.

$$E_{\text{MAPE}} = \frac{1}{n}\sum_{t=1}^{n}\frac{S_s - O_s}{O_s} \tag{15}$$

It indicates the percentage average and the relative deviation from an actual optimal objective value. Furthermore, in Equation 15, O_s denotes the optimal objective value of an optimal solution, whereas S_s is the objective value of an inferior solution. The optimal value therefore is based on the best solution achievable, whereas any deviation is considered to be sub-optimal.

B. Research Design

Simulation experiments based on different quadratic and non-quadratic matrices were carried out using an Intel Core i7-6820HQ 2.70 GHz featuring 32 GB RAM in order to compare the six methods discussed in this paper. The software *AnyLogic*

version 8.7.2 was utilised to generate and solve assignment problems.

For every experiment, a total of 20.000 assignment scenarios were randomly generated. In order to generate realistic sets of data, a map was randomly filled with nodes, which represent the AMRs and the tasks. Their distance from each other provided the cost values assigned to the cells in the matrices. The following overview shows which matrices were used to generate and evaluate the corresponding Key Performance Indicators (KPIs):

- Mean Computation Times for ILP, HM, JVC, GS, VAM and VAM-nq were computed for:
  a) 20 quadratic matrices starting at 10×10 and increasing to 200×200 in steps of 10
  b) 20 non-quadratic matrices starting at 50×50 and increasing to 50×525 in steps of 25
- Probability for the heuristic methods VAM, VAM-nq and Greedy Search to reach the actual optimal objective value was calculated for:
  a) 50 quadratic $n \times n$ matrices with $n = \{5, \ldots, 35\}$
  b) 50 non-quadratic $5 \times n$ matrices with $n = \{6, \ldots, 55\}$
- E_{MAPE}, Mean Absolute Percentage Error of VAM, VAM-nq and Greedy Search to map the deviation from the actual optimal solution was computed for 15 non-quadratic $5 \times n$ matrices with $n = \{6, \ldots, 20\}$
- E_{MAPE}, Mean Absolute Percentage Error of VAM, VAM-nq and Greedy Search to map the deviation from the optimal objective value were calculated for:
  a) 50 non-quadratic $50 \times n$ matrices with $n = \{51, \ldots, 100\}$
  b) 17 mixed matrices: 5×5, 5×50, 10×10, 10×20, 10×30, 10×40, 20×20, 10×60, 20×60, 30×30, 10×100, 40×40, 50×50, 50×100, 100×100, 100×200, 100×300

C. Discussion of Results

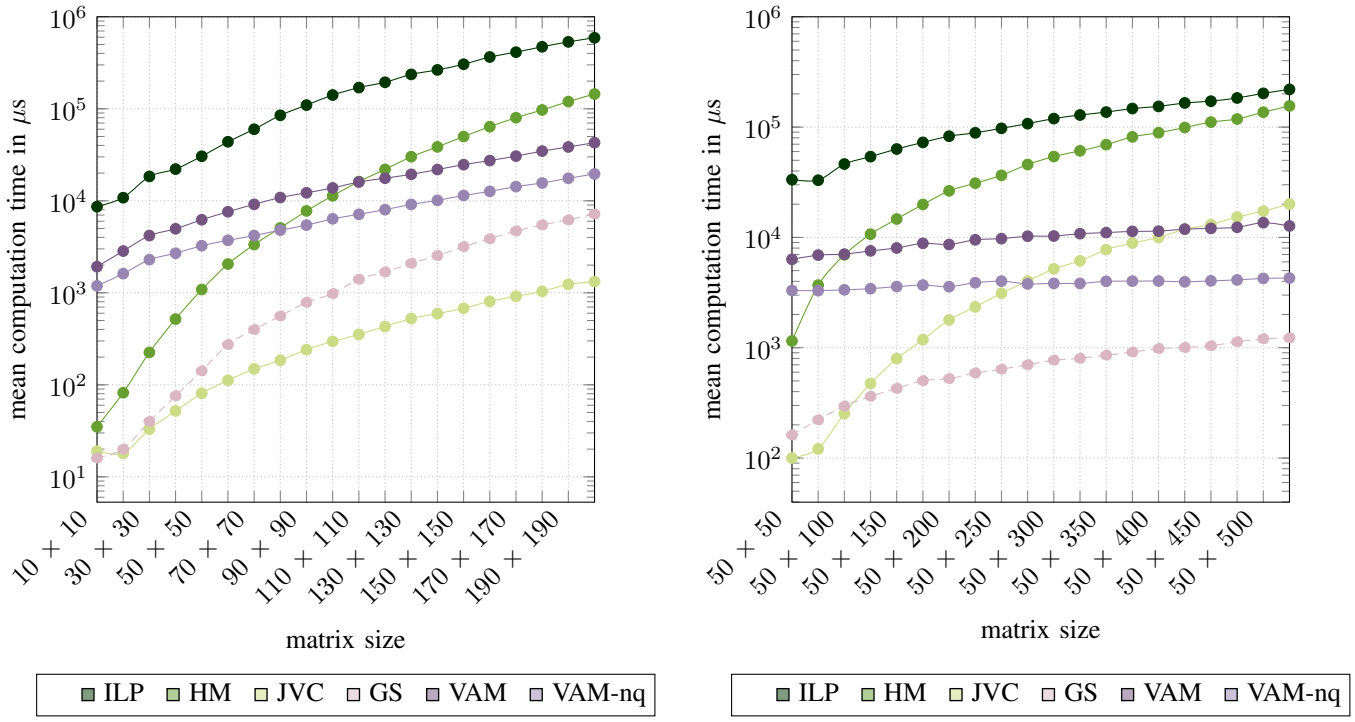
This section summarises the results of the simulation experiments. Figure 2 a shows clearly that ILP requires the most computation time for quadratic matrices of all sizes. For example, the time required to solve a 180×180 matrix is six times longer than that required by the Hungarian Method and even 1.000 times longer than for the JVC algorithm. JVC is by far the fastest method to solve quadratic matrices of all sizes. The results show JVC to be even faster than the Greedy Search heuristic even for large quadratic matrix sizes. For non-quadratic matrices, VAM-nq begins to outperform the JVC algorithm when matrices are equal to or larger than 50×300 as illustrated in Figure 2 b. More specifically, the computational time only increases slightly for VAM-nq as the matrix size increases even further, while also noting that the VAM-nq provides optimal objective values in most cases. In contrast, the computational time required by JVC, rises exponentially as the matrix size increases as presented in Figure 4.

HM, VAM and VAM-nq yielded computation times between 31 μs and 100.000 μs to solve matrices of all considered sizes. Even so, the results showed that the time required to compute results for matrices equal to or larger than 120×120 and 50×125, for non-quadratic matrices, was longer for the HM than both the VAM and VAM-nq. However, Figure 2 b shows that the computation time for the HM increases considerably when non-quadratic matrices are computed. This is most likely due to the fact that the HM has to generate additional rows or columns to produce quadratic matrices since it cannot, as previously mentioned, compute non-quadratic problems. VAM and VAM-nq, on the other hand, are able to compute quadratic and non-quadratic matrices regardless of their size in a relatively short amount of time (between 1.000 μs for small sizes and 30.000 μs for larger ones), which provides support for the superior scalability of VAM and VAM-nq.

As previously mentioned, the three heuristic methods GS, VAM and VAM-nq are not always able to produce an optimal solution. Figure 3 discusses the probability of these three methods to reach for the actual optimal objective value. For quadratic matrices, this probability approaches 0 % for matrix sizes of 20×20 and upwards. In contrast, for relatively small quadratic scenarios, such as 9×9, the chance for VAM and VAM-nq to calculate the best possible solution is only 24.7 %. For a 16×16 scenario, the chance is only 3.3 %. The examined GS method has an even lower probability than both VAM versions when applied to quadratic matrices. For the 9×9 scenario, the likelihood of attaining the optimal objective value lies at 4.8 %, for 16×16 matrices it is infinitesimal small at 0.2 %. For non-quadratic scenarios, VAM-nq and GS reveal a different picture: as the difference between dimensions increases, both methods approach a 100 % probability for reaching the optimal objective value, VAM-nq earlier than GS. For example, the probability of solving a 5×13 scenario optimally lies at 94.5 % for VAM-nq and at 68.5 % for the GS method. A 5×55 scenario is solved optimally 99.7 % of the time when applying the VAM-nq and 92.9 % of the time when the Greedy Search method is utilised.

Having presented the computational times and the probability of reaching the actual optimum for the examined combinations of variables, this section present the results associated with MAPE (see Figure 4). The mean deviation to the optimum of VAM increase continuously for non-quadratic matrices as the difference between the dimensions increases successively. That is, while the deviation from the optimal objective value of the original VAM continuous to grow as the matrix size increases, the deviation of the VAM-nq approaches 0 %. This means an optimal solution is generated more often as the difference between the dimensions increase. This clearly demonstrates that the VAM-nq is more suitable to compute non-quadratic instances than the original VAM.

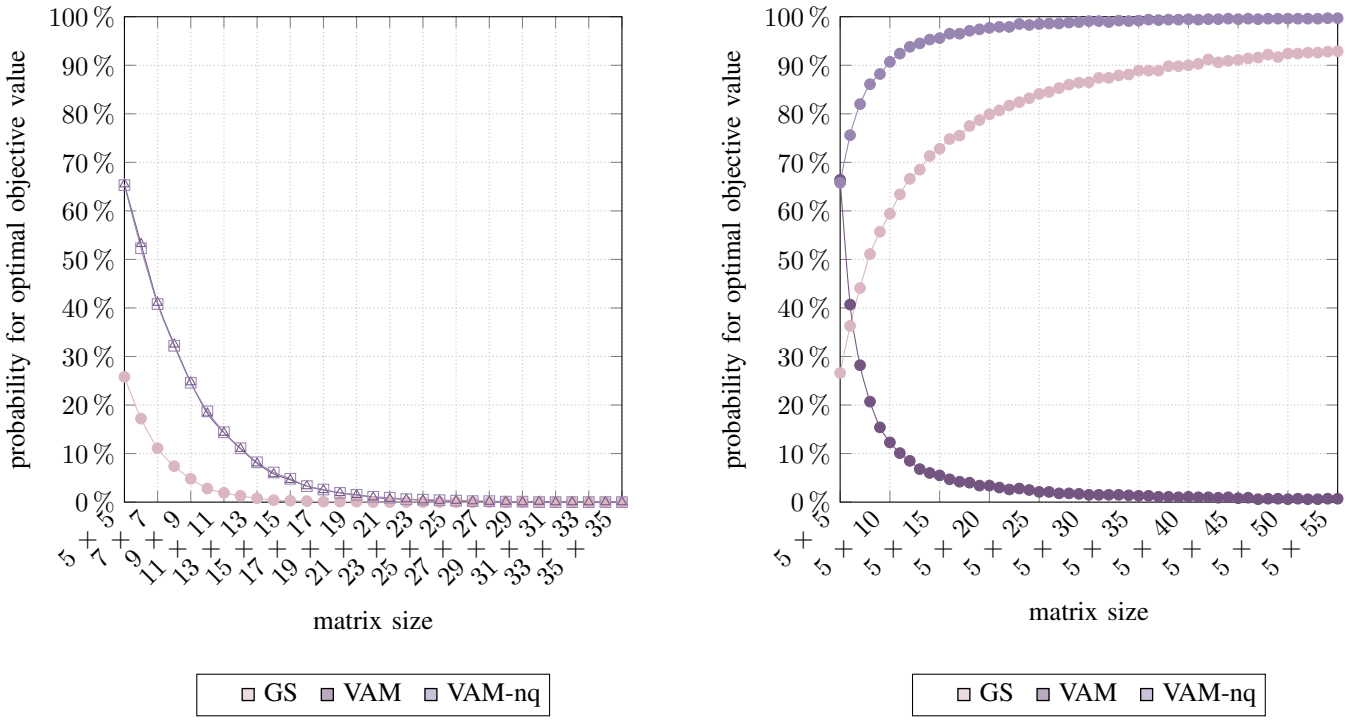
Figure 6 shows the results of simulation experiments performed by applying the original VAM, the VAM-nq and the GS algorithm to different assignment scenarios. For non-quadratic matrices, it is also evident that the original VAM produces objective values that are up to 300 % higher than the corresponding optimal solution. The results yielded by VAM-nq, on the other hand, display almost no deviation



(a) For quadratic matrices

(b) For non-quadratic matrices

Figure 2: Mean computation time for ILP (CPLEX-solver), HM, JVC, GS, VAM and VAM-nq for quadratic and non-quadratic matrices in microseconds (20.000 averaged samples for each test point)



(a) for quadratic matrices

(b) for non-quadratic matrices

Figure 3: Probability for reaching the optimal objective value for quadratic and non-quadratic matrices (20.000 averaged samples for each test point)

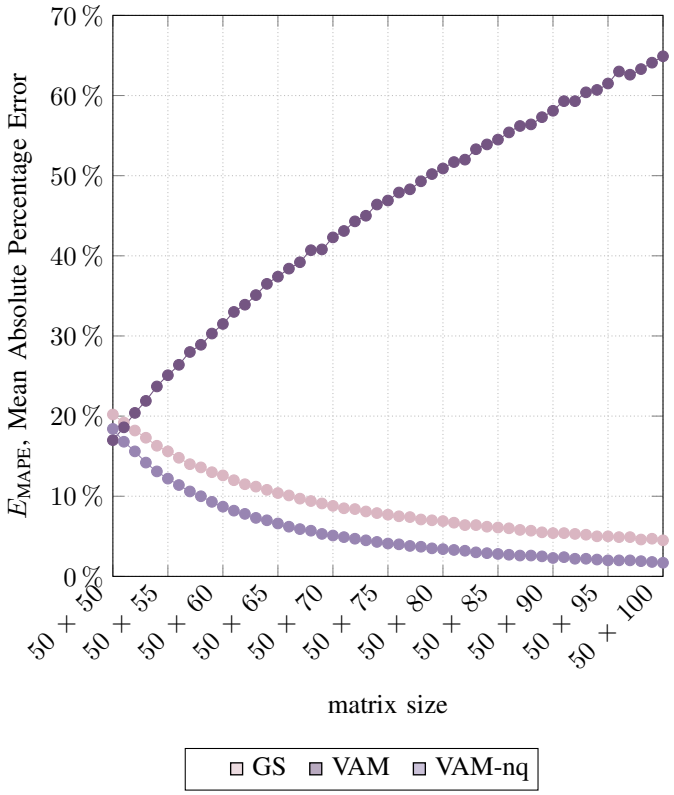


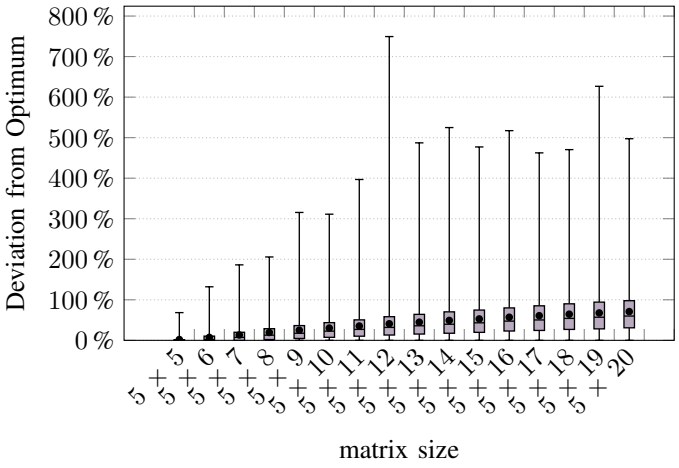
Figure 4: E_{MAPE}, Mean Absolute Percentage Error of the original VAM, VAM-nq and Greedy Search from the optimal objective value for non-quadratic matrices (20.000 averaged samples for each test point)

from the optimal objective value and this method, on average, manages to generate optimal solutions in a majority of non-quadratic cases. However, the results also show that the VAM produces slightly better results for quadratic matrices than the VAM-nq, but the differences in those cases are considered to be negligible.

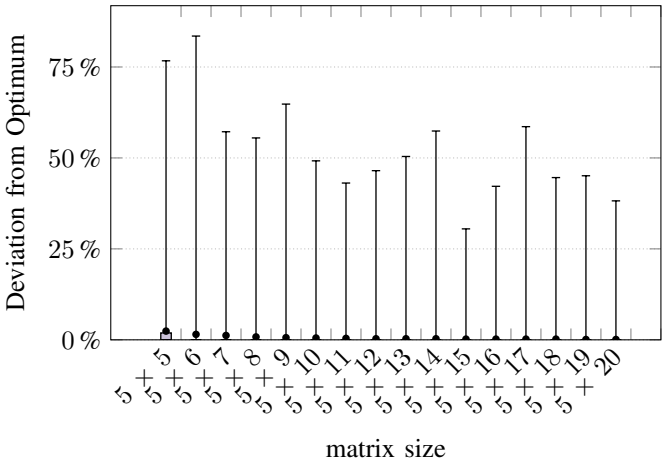
V. Conclusion

The results of our experiments have shown that JVC is superior to all other methods discussed in this paper. JVC is able to provide optimal solutions to each assignment scenario in a relatively short time. For large non-quadratic matrices, the approximation method VAM-nq yields better results than JVC in terms of computational time, even if some scenarios may not be solved optimally. As such, the selection of a method for practical application depends on the circumstances and prioritised objective.

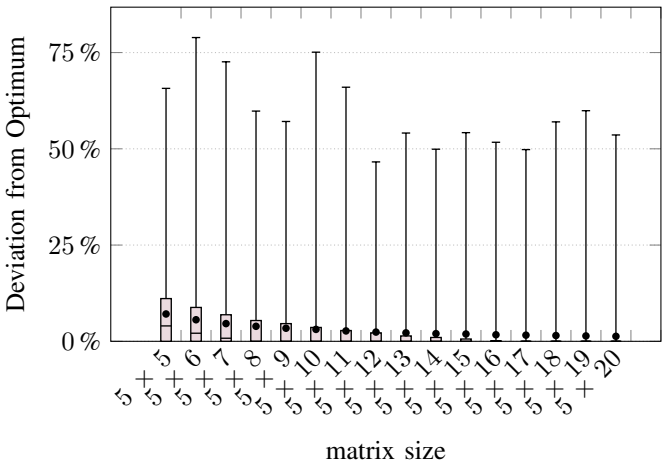
VAM is substantially faster at calculating results than the HM and ILP across all matrix sizes. However, the VAM results in a MAPE of 20 % from the optimal objective value for quadratic matrices (starting with approx. 15×15). In contrast, the MAPE increases to 65 % for larger non-quadratic matrices (starting at approx. 50×100). On the other side, VAM produces insufficient results for all non-quadratic matrice sizes and deviates considerably from the optimum. The adapted version



(a) VAM



(b) VAM-nq



(c) Greedy Search

Figure 5: Percentage Error of VAM, VAM-nq and Greedy Search from the optimal objective value with increasing matrix size (each box-plot represents a set of 20.000 samples)

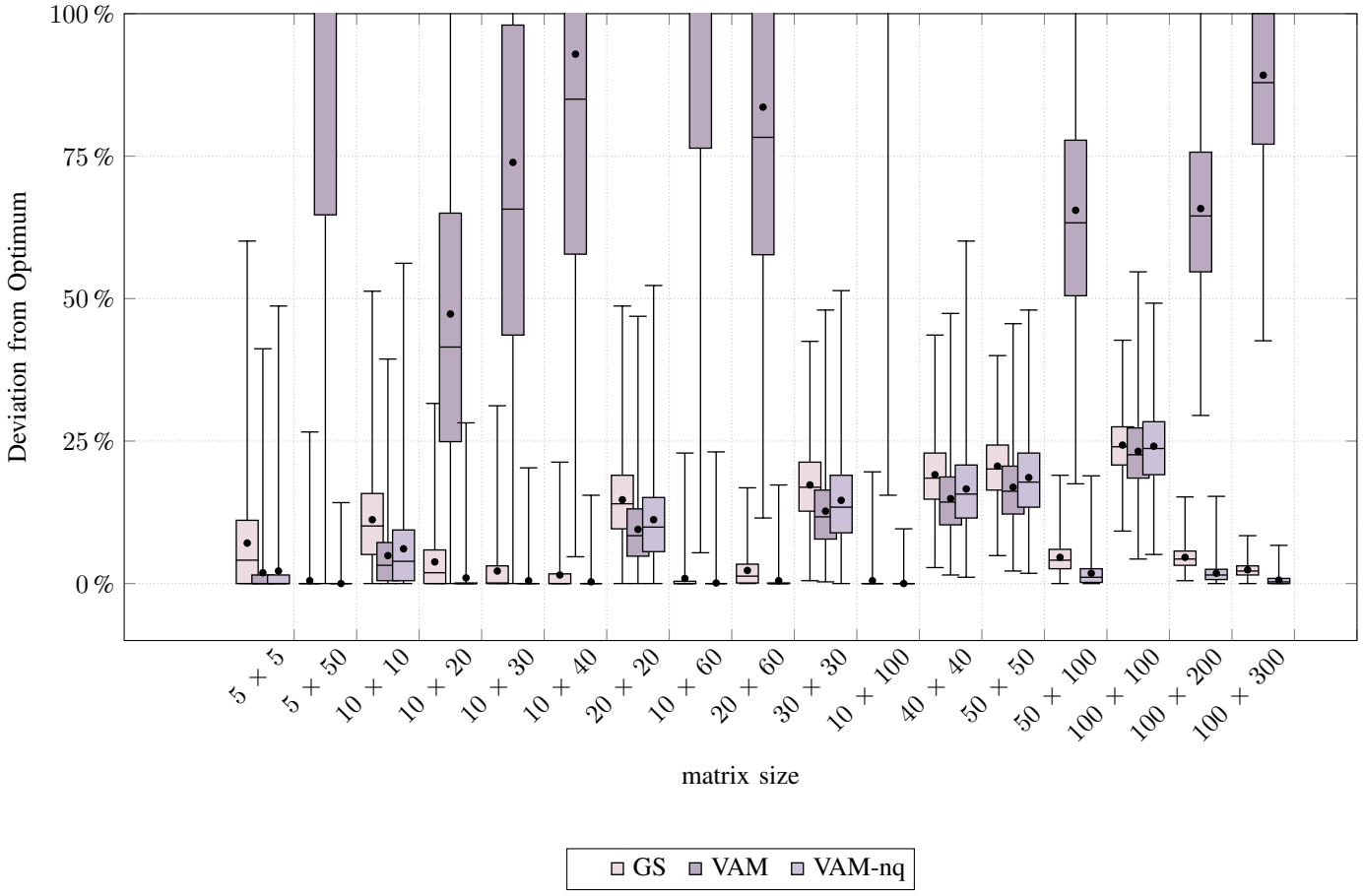


Figure 6: Deviation from optimum of Greedy Search the original VAM and VAM-nq from the optimal objective value for different matrix sizes (each box-plot represents a set of 20.000 samples)

of VAM, introduced as VAM-nq, yields objective values that deviate slightly more from the optimal objective value than the original VAM for quadratic instances, but provides much better results for non-quadratic scenarios, reaching an optimum solution in most of the cases. Moreover, the likelihood of solving a non-quadratic matrix optimally approaches 100 %, as the difference between the dimensions increases.

Based on these findings, we propose that the JVC algorithm proved to be most suitable for quadratic assignment problems. For non-quadratic scenarios, JVC is currently deemed to be the best method for solving quadratic and non-quadratic assignment problems with optimal precision. If performance in terms of computation time is more important than the quality of the solution, VAM-nq can be used for large non-quadratic matrices to generate a faster approximated solution. The probability for VAM-nq to calculate an optimal solution for non-quadratic problem instances larger than 5×25 approaches 100 % as the scale increases. Additionally, the time required by VAM-nq to calculate solutions is substantially faster than JVC for such scenarios and the computational time remains feasible even for large scale assignment problems. In summary, we propose that VAM-nq can be applied to streamline the computational effort and duration required to solve large non-quadratic scenarios at only minor to negligible detriment to the solution's quality.

References

Ahmed, M., Khan, A., Ahmed, F. and Uddin, M. (2016), 'Customized Vogel's Approximation Method (CVAM) for Solving Transportation Problems', *Buletinul Institutulu Politehnic* 62 (66), pp. 31–44.

Applegate, D. and Cook, W. (1993), 'Solving large-scale matching problems', *Network Flows and Matching: First DIMACS Implementation Challenge*, ed. by D. Johnson and C. McGeoch, vol. 12, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Providence, Rhode Island: American Mathematical Society, pp. 557–576, ISBN: 9780821865989.

Atamtürk, A. (1993), *Efficient algorithms for the minimum cost perfect matching problem on general graphs*, Bilkent University.

Balakrishnan, N. (1990), 'Modified Vogel's approximation method for the unbalanced transportation problem', *Applied Mathematics Letters* 3 (2), pp. 9–11, ISSN: 08939659.

Burkard, R.E. and Derigs, U. (1980), *Assignment and Matching Problems: Solution Methods with FORTRAN-Programs*, vol. 184, Lecture notes in economics and mathematical systems, Berlin, Heidelberg usw.: Springer, ISBN: 9783540102670.

Cattrysse, D.G. and van Wassenhove, L.N. (1992), 'A survey of algorithms for the generalized assignment problem', *European Journal of Operational Research* 60 (3), pp. 260–272, ISSN: 0377-2217.

Cheriyan, J., Hagerup, T. and Mehlhorn, K. (1996), 'An $o(n^3)$-Time Maximum-Flow Algorithm', *SIAM Journal on Computing* 25 (6), pp. 1144–1170.

Cunningham, W.H. and Marsh, A.B. (1978), 'A primal algorithm for optimum matching', *Polyhedral Combinatorics*, ed. by M.L. Balinski, E.M.L. Beale, G.B. Dantzig, L. Kantorovich, T.C. Koopmans, A.W. Tucker, P. Wolfe, V. Chvátal, R.W. Cottle, H.P. Crowder, J.E. Dennis, B.C. Eaves, R. Fletcher, M. Iri, E.L. Johnson, C. Lemarechal, C.E. Lemke, G.P. McCormick, G.L. Nemhauser, W. Oettli, M.W. Padberg, M.J.D. Powell, J.F. Shapiro, L.S. Shapley, K. Spielberg, H. Tuy, D.W. Walkup, R. Wets, C. Witzgall and A.J. Hoffman, vol. 8, Mathematical Programming Studies, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 50–72, ISBN: 978-3-642-00789-7.

Cygan, M., Gabow, H.N. and Sankowski, P. (2015), 'Algorithmic Applications of Baur-Strassen's Theorem: Shortest Cycles, Diameter and Matchings', *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, IEEE, pp. 531–540, ISBN: 978-0-7695-4874-6.

Derigs, U. (1981), 'A shortest augmenting path method for solving minimal perfect matching problems', *Networks* 11 (4), pp. 379–390, ISSN: 00283045.

Derigs, U. (1986), 'Solving large-scale matching problems efficiently: A new primal matching approach', *Networks* 16 (1), pp. 1–16, ISSN: 00283045.

Derigs, U. (1988), 'Solving non-bipartite matching problems via shortest path techniques', *Annals of Operations Research* 13 (1), pp. 225–261.

Derigs, U. and Metz, A. (1986), 'On the use of optimal fractional matchings for solving the (integer) matching problem', *Computing* 36 (3), pp. 263–270, ISSN: 0010-485X.

Derigs, U. and Metz, A. (1991), 'Solving (large scale) matching problems combinatorially', *Mathematical Programming* 50 (1-3), pp. 113–121, ISSN: 0025-5610.

Díaz-Parra, O., Ruiz-Vanoye, J.A., Bernábe Loranca, B., Fuentes-Penna, A. and Barrera-Cámara, R.A. (2014), 'A Survey of Transportation Problems', *Journal of Applied Mathematics* 2014 (3), pp. 1–17, ISSN: 1110-757X.

Dinagar, D.S. and Keerthivasan, R. (2018), 'Solving Fuzzy Transportation Problem Using Modified Best Candidate Method', *Journal of Computer and Mathematical Sciences* 9 (9), pp. 1179–1186.

Dinic, E.A. and Kronrod, M.A. (1969), 'An algorithm for the solution of the assignment problem', *Soviet Math. Dokl*, vol. 6, pp. 1324–1326.

Duan, R. and Pettie, S. (2014), 'Linear-Time Approximation for Maximum Weight Matching', *Journal of the ACM* 61 (1), pp. 1–23.

Edmonds, J., Johnson, E.L. and Lockhart, S.C. (1969), 'Blossom I: a computer code for the matching problem', *IBM TJ Watson Research Center*.

Feder, T. and Motwani, R. (1995), 'Clique Partitions, Graph Compression and Speeding-Up Algorithms', *Journal of Computer and System Sciences* 51 (2), pp. 261–272.

Freling, R., Wagelmans, A.P.M. and Paixão, J.M.P. (2001), 'Models and Algorithms for Single-Depot Vehicle Scheduling', *Transportation Science* 35 (2), pp. 165–180, ISSN: 0041-1655.

Gabow, H.N. (1976), 'An Efficient Implementation of Edmonds' Algorithm for Maximum Matching on Graphs', *Journal of the ACM* 23 (2), pp. 221–234.

Gabow, H.N. (1985), 'A scaling algorithm for weighted matching on general graphs', *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, IEEE, pp. 90–100, ISBN: 0-8186-0644-4.

Gabow, H.N. (1990), 'Data structures for weighted matching and nearest common ancestors with linking', *Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, pp. 434–443.

Gabow, H.N. (2017), 'A Data Structure for Nearest Common Ancestors with Linking', *ACM Transactions on Algorithms* 13 (4), pp. 1–28.

Gabow, H.N., Galil, Z. and Spencer, T.H. (1989), 'Efficient implementation of graph algorithms using contraction', *Journal of the ACM* 36 (3), pp. 540–572.

Gabow, H.N. and Tarjan, R.E. (1991), 'Faster scaling algorithms for general graph matching problems', *Journal of the ACM* 38 (4), pp. 815–853.

Galil, Z., Micali, S. and Gabow, H.N. (1982), 'Priority queues with variable priority and an O(EV log V) algorithm for finding a maximal weighted matching in general graphs', *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, IEEE, pp. 255–261.

Gerngross, P. (1991), *Zur Implementation von Edmonds' Matching Algorithmus: Datenstrukturen und verschiedene Varianten*, Universität Augsburg.

Goldberg, A.V. and Karzanov, A.V. (2004), 'Maximum skew-symmetric flows and matchings', *Mathematical Programming* 100 (3), ISSN: 0025-5610.

Goldberg, A.V. and Kennedy, R. (1997), 'Global Price Updates Help', *SIAM Journal on Discrete Mathematics* 10 (4), pp. 551–572.

Goyal, S.K. (1984), 'Improving VAM for Unbalanced Transportation Problems', *Journal of the Operational Research Society* 35 (12), pp. 1113–1114, ISSN: 1476-9360.

Grötschel, M. and Holland, O. (1985), 'Solving matching problems with linear programming', *Mathematical Programming* 33 (3), pp. 243–259, ISSN: 0025-5610.

Harvey, N.J.A. (2006), 'Algebraic Structures and Algorithms for Matching and Matroid Problems', *2006 47th Annual*

IEEE Symposium on Foundations of Computer Science (FOCS'06), IEEE, pp. 531–542, ISBN: 0-7695-2720-5.

Havel, T.F., Kuntz, I.D. and Crippen, G.M. (1983), 'The theory and practice of distance geometry', *Bulletin of Mathematical Biology* 45 (5), pp. 665–720, ISSN: 0092-8240.

Hopcroft, J.E. and Karp, R.M. (1973), 'An n5/2 Algorithm for Maximum Matchings in Bipartite Graphs', *SIAM Journal on Computing* 2 (4), pp. 225–231.

Jonker, R. and Volgenant, A. (1987), 'A shortest augmenting path algorithm for dense and sparse linear assignment problems', *Computing* 38 (4), pp. 325–340, ISSN: 0010-485X.

Karzanov, A.V. (1976), 'Efficient implementations of Edmonds' algorithms for finding matchings with maximum cardinality and maximum weight', *Studies in Discrete Optimization*, pp. 306–327.

Kazakidis, G. (1980), *Die Löung minimaler perfekter Matchingprobleme mittels kürzester erweiternder Pfade*.

Korukoğlu, S. and Ballı, S. (2011), 'A Improved Vogel's Approximatio Method for the Transportation Problem', *Mathematical and Computational Applications* 16 (2), pp. 370–381.

Krumke, S.O. and Thielen, C. (2013), 'The generalized assignment problem with minimum quantities', *European Journal of Operational Research* 228 (1), pp. 46–55, ISSN: 0377-2217.

Kuhn, H.W. (1955), 'The Hungarian method for the assignment problem', *Naval Research Logistics Quarterly* 2 (1-2), pp. 83–97, ISSN: 00281441.

Lawler, E.L. (1976), *Combinatorial optimization: Networks and matroids*, Unabridged reprint ... orig. publ. by Holt, Rinehart & Winston in 1976, Mineola, NY: Dover Publ, ISBN: 978-0486414539, URL: http://www.loc.gov/catdir/description/dover032/00060242.html.

Lessard, R., Rousseau, J.-M. and Minoux, M. (1989), 'A new algorithm for general matching problems using network flow subproblems', *Networks* 19 (4), pp. 459–479, ISSN: 00283045.

Li, J.-Q., Mirchandani, P.B. and Borenstein, D. (2007), 'The vehicle rescheduling problem: Model and algorithms', *Networks* 50 (3), pp. 211–229, ISSN: 00283045.

Micali, S. and Vazirani, V.V. (1980), 'An algorithm for finding maximum matching in general graphs', *21st Annual Symposium on Foundations of Computer Science (sfcs 1980)*, IEEE, pp. 17–27.

Miller, D.L. and Pekny, J.F. (1995), 'A Staged Primal-Dual Algorithm for Perfect b-Matching with Edge Capacities', *ORSA Journal on Computing* 7 (3), pp. 298–320, ISSN: 0899-1499.

Minoux, M. (1982), *A New Polynomial Cutting-plane Algorithm for Maximum Weight Matchings in General Graphs*.

Mucha, M. and Sankowski, P. (2004), 'Maximum Matchings via Gaussian Elimination', *45th Annual IEEE Symposium on Foundations of Computer Science*, IEEE, pp. 248–255, ISBN: 0-7695-2228-9.

Munkres, J. (1957), 'Algorithms for the Assignment and Transportation Problems', *Journal of the Society for Industrial and Applied Mathematics* 5 (1), pp. 32–38, ISSN: 0368-4245.

Nahar, J., Rusyaman, E. and Putri, S.D.V.E. (2018), 'Application of improved Vogel's approximation method in minimization of rice distribution costs of Perum BULOG', *IOP Conference Series: Materials Science and Engineering* 332.

Osman, I.H. (1995), 'Heuristics for the generalised assignment problem: simulated annealing and tabu search approaches', *OR Spectrum* 17 (4), pp. 211–225, ISSN: 0171-6468.

Paul, S. (2018), 'A novel initial basic feasible solution method for transportion problem', *International Journal of Advanced Research in Computer Science* 9 (1), pp. 472–474.

Pentico, D.W. (2007), 'Assignment problems: A golden anniversary survey', *European Journal of Operational Research* 176 (2), pp. 774–793, ISSN: 0377-2217.

Pulleyblank, W. (1973), *Faces of Matching Polyhedra*.

Reinfeld, N.V. and Vogel, W.R. (1958), *Mathematical Programming*, Prentice-Hall, Englewood Cliffs.

Selmair, M., Meier, K.-J. and Wang, Y. (2019), 'Solving non-quadratic Matrices in assignment Problems with an improved Version of Vogel's Approximation Method', *European Conference of Modelling and Simulation*.

Shimshak, D., Kaslik, A.J. and Barclay, T. (1981), 'A Modification Of Vogel'S Approximation Method Through The Use Of Heuristics', *INFOR: Information Systems and Operational Research* 19 (3), pp. 259–263, ISSN: 0315-5986.

Shore, H.H. (1970), 'The Transportation problem and the Vogel Approximation Method', *Decision Sciences* 1 (3-4), pp. 441–457, ISSN: 0011-7315.

Skiena, S. (1990), 'The Cycle Structure of Permutations', *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, pp. 20–24.

Trick, M. (1987), *Networks with Additional Structured Constraints*, Georgia Institute of Technology.

Witzgall, C. and Zahn, C.T. (1965), 'Modification of Edmonds' maximum matching algorithm', *J. Res. Nat. Bur. Standards Sect. B*.