SOLVING NON-QUADRATIC MATRICES IN ASSIGNMENT PROBLEMS WITH AN IMPROVED VERSION OF VOGEL'S APPROXIMATION METHOD

Maximilian Selmair
Alexander Swinarew
BMW Group
80788 Munich, Germany
maximilian.selmair@bmw.de

Klaus-Jürgen Meier
University of Applied Sciences Munich
80335 Munich, Germany
klaus-juergen.meier@hm.edu

Yi Wang
University of Plymouth
PL4 8AA Plymouth, United Kingdom
yi.wang@plymouth.ac.uk

Keywords*—*Vogel's Approximation Method; Hungarian Method; Generalised Assignment Problem

Abstract*—The efficient allocation of tasks to vehicles in a fleet of self-driving vehicles (SDV) becomes challenging for large-scale systems (e. g. more than hundred vehicles). Operations research provides different methods that can be applied to solve such assignment problems. Integer Linear Programming (ILP), the Hungarian Method (HM) or Vogel's Approximation Method (VAM) are frequently used in related literature (Paul 2018; Dinagar and Keerthivasan 2018; Nahar et al. 2018; Ahmed et al. 2016; Korukoğlu and Ballı 2011; Balakrishnan 1990). The underlying paper proposes an adapted version of VAM which reaches better solutions for non-quadratic matrices, namely Vogel's Approximation Method for non-quadratic Matrices (VAM-nq). Subsequently, VAM-nq is compared with ILP, HM and VAM by solving matrices of different sizes in computational experiments in order to determine the proximity to the optimal solution and the computation time. The experimental results demonstrated that both VAM and VAM-nq are five to ten times faster in computing results than HM and ILP across all tested matrix sizes. However, we proved that VAM is not able to generate optimal solutions in large quadratic matrices constantly (starting at approx. 15×15) or small non-quadratic matrices (starting at approx. 5×6). In fact, we show that VAM produces insufficient results especially for non-quadratic matrices. The result deviate further from the optimum if the matrix size increases. Our proposed VAM-nq is able to provide similar results as the original VAM for quadratic matrices, but delivers much better results in non-quadratic instances often reaching an optimum solution. This is especially important for practical use cases since quadratic matrices are rather rare.

LIST OF ABBREVIATIONS

GAP Generalised Assignment Problem
HM Hungarian Method
ILP Integer Linear Programming
KPI Key Performance Indicator
SDV Self-driving Vehicle
VAM Vogel's Approximation Method
VAM-nq Vogel's Approximation Method for non-quadratic Matrices

I. INTRODUCTION

The transportation problem is an extensively studied topic in operational research (Díaz-Parra et al. 2014). The methods for solving the mentioned problem aim to minimise the total transportation cost while bringing goods from several supply points (e. g. warehouses) to demand locations (e. g. customers). In general, each transport origin features a fixed amount of goods that can be distributed. Correspondingly, every point of transport destination requires a certain amount of units (Shore 1970). The underlying use case, where tasks have to be assigned to self-driving vehicles (SDVs), differs in some regards from the classical transportation problem. In our case, each vehicle has a capacity restriction of one, i. e. a maximum of one load carrier can be transported at a time. Furthermore, each task corresponds to a demand of one. This basically means that every task can only be allocated to one single vehicle. Additionally, the amount of available vehicles does rarely match the number of unassigned tasks in practice. Since the size of the matrices depends on those two factors, non-quadratic matrices (e. g. 10×50) are common. There are different approaches that can be applied to solve this kind of problem, e. g. ILP, HM and VAM. While ILP and HM manage to always generate an optimal solution, VAM often fails to do so. Furthermore, those methods vary greatly in the computational demand necessary to solve assignment problems. There are two major reasons involved in the motivation for improving the original VAM. For one, the authors wanted to keep the great performance (computational time) of the original VAM. Secondly, the insufficient results for non-quadratic matrices should be improved significantly, i. e. reaching the optimum. Following these considerations, an improved VAM version, as proposed in this paper, was developed and compared with the three established methods. The goal was to find a solution that provides optimal or near-optimal results while at the same requiring a small amount of resources (computing power).

Despite its age, the approximation method proposed by those authors is still in use nowadays and is subject to recent operations research as the contributions by Banik and Hasan (2018), Ezekiel and Edeki (2018), Hlatká et al. (2017), Ahmed

Communications of the ECMS, Volume 33, Issue 1,
Proceedings, ©ECMS Mauro Iacono, Francesco Palmieri,
Marco Gribaudo, Massimo Ficco (Editors)
ISBN: 978-3-937436-65-4/978-3-937436-66-1(CD) ISSN 2522-2414

et al. (2016), and Gani and Baskaran (2014) show.

Already Shimshak et al. (1981) extended the original VAM with rules that apply in case of ties, e. g. the same maximum cost differences occur. Out of the four cases using either individual rules or a combination of them, only one case manages to generate slightly better solutions than the original VAM in regards to costs. Furthermore, they used only small matrices (5×5, 5×10 and 10×10) which does not provide any information on the results achieved in large-scale applications. Goyal (1984) further tried to improve Shimshak's approach in case of unbalanced transportation problems, i. e. the total supply does not correspond to the total demand. Again, only a small 3×3 matrix was used, thus, lacking any real informative value. Balakrishnan (1990) realised this drawback and tested Goyal's approach with other not specified examples concluding that it is not always better than Shimshak's approach. He in turn proposed an extended approach which was tested in different scenarios and compared with those of the other authors mentioned above leading to even better solutions. In a more recent contribution, Korukoğlu and Ballı (2011) improved the original VAM by constructing a total opportunity cost matrix which they obtained through the addition of the row and column opportunity cost matrix. A row opportunity matrix is for example generated by subtracting the smallest cost value in each row from all other values in the same row. The column opportunity matrix is obtained in the same way. Korukoğlu and Ballı (2011) further deviate from the classical approach by selecting the three rows or columns with the highest penalty costs instead of choosing only the highest one. Out of those three, the cell with the lowest transportation cost is consequently selected and used for resource allocation.

This paper is structured as follows: in the second chapter a detailed description of VAM as well as a brief explanation of the HM and the ILP are given. The third chapter features the description of the proposed VAM-nq. Chapter four will provide an overview of the experiments as well as the discussion of the corresponding results. The last chapter contains a brief conclusion to this paper.

II. Established Solution Methods for the Generalised Assignment Problem

This chapter is intended to provide a description on established solution methods for the Generalised Assignment Problem. These are in particular the basic VAM as well as the HM and the ILP. Since the use case at hand differs in some areas from conventional examples (e. g. the vehicles can only transport one load carrier at a time and have thus a supply of one), these variations will be considered in the description of VAM and the HM. The ILP approach will be adapted to the underlying use case as well, i. e. an appropriate objective function as well as necessary constraints will be formulated.

A. Vogel's Approximation Method

The following description of VAM is based on the original proposal by Reinfeld and Vogel (1958). VAM solves transport matrices by repeating the steps as seen below until a feasible solution is found. The cells of the matrices are filled with costs c_{ij} associated with allocating a task to a vehicle. Those costs occur when a vehicle brings goods from a point of origin i to a destination j. 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). 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 there is a tie, choose 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 a vehicle.
4) Eliminate the row and column that has been used for the allocation.
5) Check if there are still vehicles and tasks left to allocate, and repeat steps 1 - 4 in case that there are.

Apart from the later proposed adoption of VAM in this paper, there are different authors that tried to improve or change the classic VAM in order to achieve better results and move closer to an optimal solution which can be achieved for example by ILP or HM (Paul 2018; Dinagar and Keerthivasan 2018; Nahar et al. 2018; Ahmed et al. 2016; Korukoğlu and Ballı 2011; Balakrishnan 1990; Goyal 1984; Shimshak et al. 1981).

An example for VAM can be found in Table I through Table III. Here, the rows are represented by vehicles (V_i) and the columns by tasks (T_j). The costs (c_{ij}) are the corresponding cell values. The row differences can be found in Δi while the column differences are saved in Δj. Starting with the initial matrix (Table I), it is evident that the biggest difference can be found in the third row featuring the lowest value in the third column (Table II). Accordingly, task 3 is assigned to vehicle 3. After the allocation, the third row and column are eliminated (Table III).

TABLE I. 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	

TABLE II. 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	30
Δj	10	0	10	30	

TABLE III. MATRIX AFTER ELIMINATING ASSIGNED ROW AND COLUMN

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	

B. Integer Linear Programming

As already stated previously, ILP is able to find an optimal solution for different scenarios, even large-scale problems. Initially, one has to formulate an objective function as well as applicable restrictions in order to receive correct results. According to Osman (1995) and following the adoption of the ILP to fit the use case at hand, the objective function reads as follows:

$$\min \sum_{j \in J} \sum_{v \in V} d_{jv} \cdot c_{jv} \tag{1}$$

$$\sum_{v \in V} d_{jv} = 1 \qquad \forall j \in J \tag{2}$$

$$\sum_{j \in J} d_{jv} \leq 1 \qquad \forall v \in V \tag{3}$$

$$d_{jv} \in \{0, 1\} \qquad \forall j \in J, \forall v \in V \tag{4}$$

The goal of the objective function (1) is to minimise the sum of all costs (c_{jv}) for all jobs $J = 1, \ldots, m$ and for all vehicles $V = 1, \ldots, n$ which is the result of multiplying the decision variable (d_{jv}) with the corresponding costs which arise when a job j is assigned to a vehicle v. The first constraint (2) ensures that every job is assigned to a vehicle while the second constraint (3) makes sure that each vehicle's capacity of 1 is not exceeded, i. e. each vehicle can execute a maximum of one job at a time. The last constraint (4), which applies for both jobs and vehicles, restricts the decision variable d_{jv} to binary values.

C. Hungarian Method

The Hungarian Method was initially proposed by Kuhn (1955) to solve the Generalised Assignment Problem (GAP). Similar to the ILP, the HM is able to find an optimal solution to said 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) Find the minimum value in each column and subtract this value from all other values in the corresponding column.
2) Find the minimum value in each row and subtract this value from all other values in the corresponding row.
3) Draw lines through the columns and rows so that all zero values of the matrix are covered by as few lines 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 not covered row and b) add it to each covered column.
6) Return to step 3.

It has to be noted that in case of $n \times m$ matrices (e. g. 10×40), an extension takes places to generate $n \times n$ matrices (e. g. 40×40) since the method only works with quadratic matrices. The additional cells are filled with values that are of the same size as the highest value of the original matrix. This extension requires additional computing power since instead of 400 cells (10×40), the algorithm has to consider 1600 cells (40×40). It is evident that this is a drawback when non-quadratic matrices are to be solved. This is always the case when more tasks than vehicles have to be considered or vice versa.

D. Comparison ILP / VAM / HM

In order to compare the three methods, experiments have been carried out with different quadratic and non-quadratic matrices using an Intel Core i7-6820HQ 2.70 GHz featuring 32 GB RAM. Figure 1 shows clearly that ILP requires the most computational time for quadratic matrices. Especially, in large matrices the time it takes to finish the calculations rises significantly. HM and VAM on the other hand do not require a lot of time to finish calculating the matrices. In fact, there is almost no difference between them up until 80×80 matrices where the HM starts to take longer than VAM. From this point forward, the difference between HM and VAM grows continuously with increasing matrix size. This might lead to the conclusion that it is more sensible to use the HM since it is able to produce optimal solutions while maintaining a relatively low computation time. However, looking at Figure 2 shows that the computation time for HM increases significantly if non-quadratic matrices are involved. This is due to the fact that HM has to generate additional rows or columns to produce quadratic matrices since it is not able to deal with non-quadratic problem instances (see section II). VAM on the other hand can deal with quadratic and non-quadratic matrices regardless of their size in a relatively small amount of time which shows VAMs great scalability.

III. IMPROVED VAM FOR NON-QUADRATIC MATRICES (VAM-NQ)

Prior experiments have shown, that the original VAM is not able to produce optimal or at least near-optimal results for non-quadratic matrices (see Figure 3). In fact, the results are in some cases more than 100 % worse than the optimum. It was determined that choosing the row or column featuring the maximum difference from the smaller dimension leads to those insufficient results. This means for example that if the matrix contains more columns than rows, choosing a column with the maximum difference (which is achieved by subtracting cell values in the smaller dimension) might result in worse outcomes. The same obviously applies vice versa if there are more rows than columns. This can be explained with the fact that the bigger dimension obviously features more values and

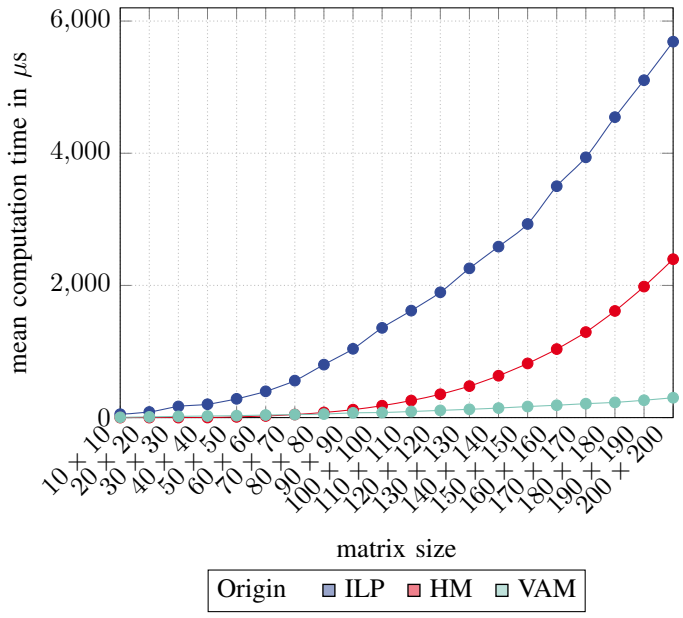

Fig. 1. Mean computational time for ILP (CPLEX-solver), HM and VAM for quadratic matrices in microseconds (5.000 samples each)

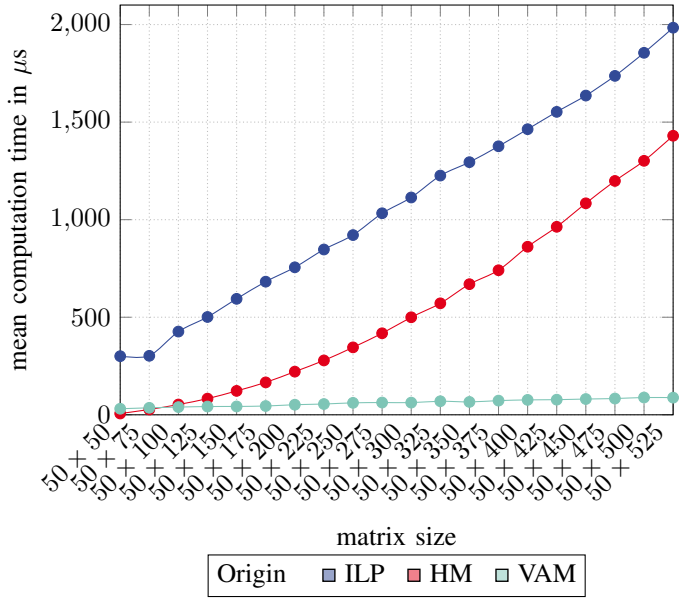

Fig. 2. Mean computational time for ILP (CPLEX-solver), HM and VAM for non-quadratic matrices in microseconds (5.000 samples each)

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 3 shows that the results produced by VAM start to deteriorate immediately if the matrix size is increased in only one dimension, i. e. a non-quadratic matrix is created. It is evident that while VAM is able to generate optimal solutions in some cases, the cases where it fails are up to 200 % worse than the optimum (see Figure 3). The deviations increase continuously with increasing matrix size, even in rather small instances. In case of 5×10 matrices for example, the results can be twice as bad as the optimum value.

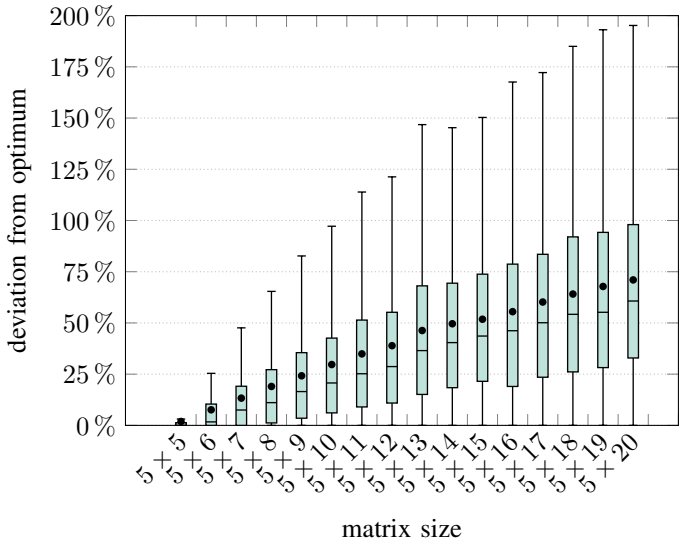

Fig. 3. Deviation of VAM from the optimal solution with increasing matrix size (5.000 samples each)

In general, there are two possible versions of non-quadratic matrices. Either there are more columns than rows or more rows than columns. The description below is based on the first case when a matrix contains more columns than rows. Accordingly, the rows and columns in the description have to be switched when the second case occurs. 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 among 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 a vehicle.
4) Eliminate the corresponding row and column that have been used for the allocation.
5) Check if there are still vehicles and tasks left to allocate, and repeat steps 1 - 4 in case that there are.

Upon comparison of the original and the adapted VAM, it becomes evident that there are some variations and simplifications. For one, VAM-nq considers only the rows in case that there are more columns than rows (step 1). Accordingly, only the biggest differences in the rows and the corresponding smallest cell values are considered (step 2 and 3). Applying those variations to the second case (more rows than columns) would mean that only columns, their biggest differences and smallest cell values are considered in steps 1 through 3. With Table IV and Table V, the example of subsection II-A is solved with both versions showing that the proposed VAM-nq provides significant better results even in small non-quadratic cases.

TABLE IV. SOLUTION OF THE ORIGINAL METHOD (VAM) WITH OBJECTIVE 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

TABLE V. SOLUTION OF VAM-NQ WITH OBJECTIVE OF 290

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

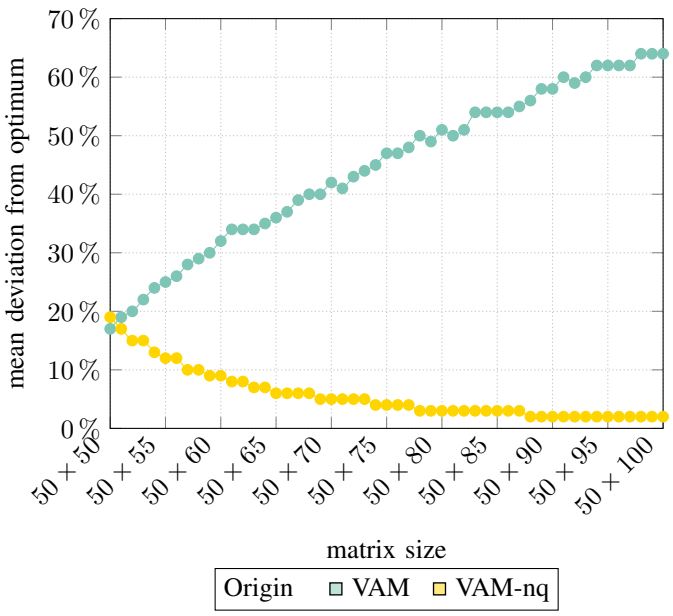

Fig. 4. Mean deviation of the original VAM and VAM-nq from the optimal solution for non-quadratic matrices in percent (5.000 samples each)

IV. EXPERIMENTS

In order to evaluate the performance of VAM-nq as well as its ability to reach optimal solutions, experiments have been carried out by using *AnyLogic* to generate matrices of different sizes.

A. Design

The matrices have been randomly generated and randomly filled with uniformly distributed costs ranging from 0 to 1.400. Each matrix has been solved 5.000 times to provide meaningful results. The following overview shows which matrices have been used to generate and evaluate the corresponding key performance indicators (KPIs):

- Mean Deviation of VAM from the optimal solution as seen in Figure 3: 15 non-quadratic $5 \times n$ matrices with $n = \{6, \ldots, 20\}$
- Mean Computation Times for VAM, HM and ILP as seen in Figure 1 and Figure 2:
  a) 20 quadratic matrices starting with 10×10 and rising to 200×200 in steps of 10
  b) 20 non-quadratic matrices starting with 50×50 and rising to 50×525 in steps of 25
- Mean Deviation of VAM and VAM-nq from the optimal solution:
  a) 50 non-quadratic $50 \times n$ matrices with $n = \{51, \ldots, 100\}$
  b) 17 different 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)

B. Results of the Experiments

As can be seen from Figure 4, both the original VAM and VAM-nq are not always able to produce an optimal solution, but are instead on average deviating from it. It is also evident that in non-quadratic instances (as seen in Figure 4) the deviation gap between the original VAM and VAM-nq rises continuously when the size of the non-quadratic matrix is increased. While the deviation of the original VAM continuous to grow, the deviation of VAM-nq approaches 0 %, i. e. an optimal solution is generated more often. This shows clearly that the proposed method is more suitable to deal with non-quadratic instances than the original method. Figure 5 shows the results of experiments performed by using the original VAM and VAM-nq for different problem instances. In this case, it is also evident that in non-quadratic instances the original VAM produces results that are up to 300 % worse than the corresponding optimal solution. VAM-nq on the other hand displays almost no deviation and manages on average to generate an optimal solution in all non-quadratic cases. However, it is also recognisable that in quadratic matrices the original VAM is always slightly better than VAM-nq, but the differences in those cases are negligible.

V. CONCLUSION

Experiments have shown that VAM is substantially faster in calculating results than HM and CPLEX-solver (ILP) across all matrix sizes. However, VAM is not able to generate optimum solutions in large quadratic matrices (starting with approx. 15×15) or small non-quadratic matrices (starting with approx. 5×6). In fact, VAM produces insufficient results in those cases and deviates greatly from the optimum. The proposed adapted version of VAM, introduced as VAM-nq, is able to provide slightly worse results than the original VAM for quadratic instances, but delivers much better results in non-quadratic instances reaching an optimum solution in most of the cases. Based on those findings, the authors propose to use an algorithm that includes both the original VAM and the improved VAM-nq and which is able to switch between those two according to the underlying situation. In case that the underlying matrix is quadratic, the original VAM method

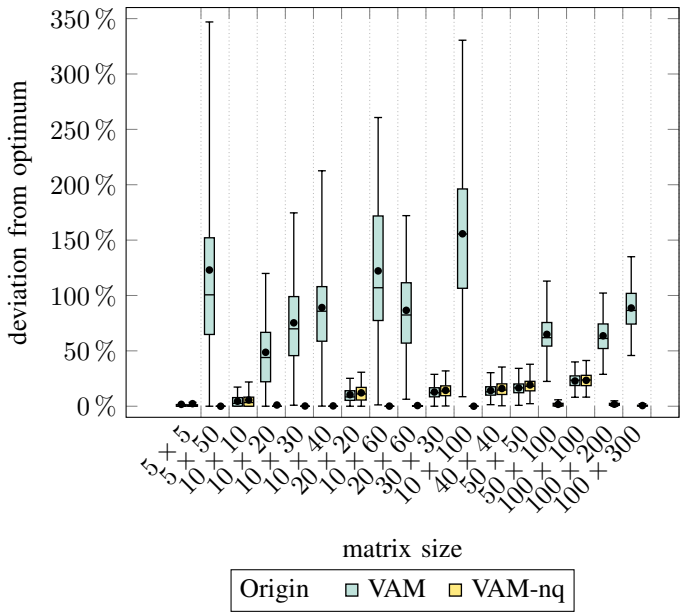

Fig. 5. Deviation of the original VAM and VAM-nq from the optimal solution for different matrix sizes (5.000 samples each)

should be used by the algorithm. For non-quadratic problem instances however, the algorithm should switch to the improved VAM-nq.

REFERENCES

Ahmed, Mesbahuddin, Aminur Khan, Faruque Ahmed, and Md Uddin (2016). "Customized Vogel's Approximation Method (CVAM) for Solving Transportation Problems". In: *Buletinul Institutului Politehnic* 62 (66), pp. 31–44.

Balakrishnan, Nagraj (1990). "Modified Vogel's approximation method for the unbalanced transportation problem". In: *Applied Mathematics Letters* 3.2, pp. 9–11.

Banik, Debapriya and Md. Zahid Hasan (2018). "Transportation Cost Optimization of an Online Business Applying Vogel's Approximation Method". In: *World Scientific News* 96.

Díaz-Parra, Ocotlán, Jorge A. Ruiz-Vanoye, Beatriz Bernábe Loranca, Alejandro Fuentes-Penna, and Ricardo A. Barrera-Cámara (2014). "A Survey of Transportation Problems". In: *Journal of Applied Mathematics* 2014.3, pp. 1–17.

Dinagar, D. Stephen and R. Keerthivasan (2018). "Solving Fuzzy Transportation Problem Using Modified Best Candidate Method". In: *Journal of Computer and Mathematical Sciences* 9.9, pp. 1179–1186.

Ezekiel, I. D. and S. O. Edeki (2018). "Modified Vogel approximation method for balanced transportation models towards optimal option settings". In: *International Journal of Civil Engineering and Technology* 9, pp. 358–366.

Gani, Nagoor and Baskaran (2014). "Improved Vogel's Approximation method to Solve Fuzzy Transshipment Problem". In: *Fuzzy Mathematical Archive* 4, pp. 80–87.

Goyal, S. K. (1984). "Improving VAM for Unbalanced Transportation Problems". In: *Journal of the Operational Research Society* 35.12, pp. 1113–1114.

Hlatká, Martina, Ladislav Bartuška, and Ján Ližbetin (2017). "Application of the Vogel Approximation Method to Reduce Transport-logistics Processes". In: *MATEC Web of Conferences* 134.4, p. 00019.

Korukoğlu, Serdar and Serkan Ballı (2011). "A Improved Vogel's Approximatio Method for the Transportation Problem". In: *Mathematical and Computational Applications* 16.2, pp. 370–381.

Kuhn, H. W. (1955). "The Hungarian method for the assignment problem". In: *Naval Research Logistics Quarterly* 2.1-2, pp. 83–97.

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

Osman, Ibrahim H. (1995). "Heuristics for the generalised assignment problem: simulated annealing and tabu search approaches". In: *OR Spectrum* 17.4, pp. 211–225.

Paul, Surjit (2018). "A novel initial basic feasible solution method for transportion problem". In: *International Journal of Advanced Research in Computer Science* 9.1, pp. 472–474.

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

Shimshak, Daniel, Alan James Kaslik, and Thomas Barclay (1981). "A Modification Of Vogel'S Approximation Method Through The Use Of Heuristics". In: *INFOR: Information Systems and Operational Research* 19.3, pp. 259–263.

Shore, Harvey H. (1970). "The Transportation problem and the Vogel Approximation Method". In: *Decision Sciences* 1.3-4, pp. 441–457.

AUTHOR BIOGRAPHIES

Maximilian Selmair is doctoral student at the University of Plymouth. Recently employed at the SimPlan AG, he was in charge of projects in the area of material flow simulation. Currently he is working on his doctoral thesis with a fellowship of the BMW Group. His email address is: maximilian.selmair@bmw.de and his website can be found at maximilian.selmair.de.

Alexander Swinarew is a masters graduate in logistics at OTH Regensburg who worked at BMW on his masters thesis during which the proposed method was developed. His email address is: alexander.swinarew@gmail.com.

Prof. Dr. Klaus-Jürgen Meier holds the professor-ship for production planning and logistic systems in the Department of Engineering and Management at the University of Applied Sciences Munich and he is the head of the Institute for Production Management and Logistics (IPL). His e-mail address is: klaus-juergen.meier@hm.edu.

Dr. Yi Wang is a lecturer in business decision making in the Faculty of Business, University of Plymouth, UK. He has special research interests in supply chain management, logistics, operation management, culture management, information systems, game theory, data analysis, semantics and ontology analysis, and neuromarketing. His e-mail address is: yi.wang@plymouth.ac.uk.