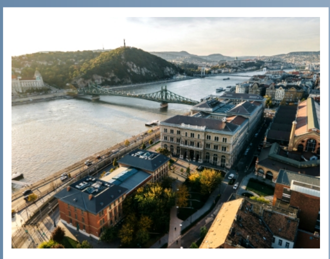

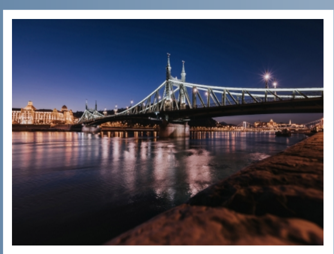

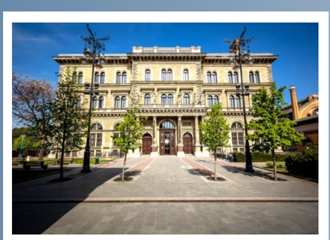

VOCAL2024

10th VOCAL Optimization Conference: Advanced Algorithms

Budapest, Hungary, June 5-7, 2024

Short Papers

VOCAL 2024
10th VOCAL Optimization Conference: Advanced Algorithms

Budapest, Hungary, June 5-7, 2024

Short Papers

Published by Hungarian Operations Research Society
Kende utca 13-17., 1111 Budapest, Hungary
E-mail: motelnok@inf.szte.hu
Website: http://www.mot.org.hu/
Contact: Boglárka G.-Tóth
Single volume, electronic publication, format: pdf.

ISBN 978-615-02-1221-0

VOCAL 2004. 9th VOCAL Optimization Conference:
Advanced Algorithms. Budapest, Hungary, June 5-7, 2024. Short Papers

Contents

Analysis of the Workload of Assembly Stations when the Makespan is Minimized in the Presence of Learning Effects

Zakaria Zine El Abidine[1], Imre Dimény[2], and Tamás Koltai[3]

Budapest University of Technology and Economics, Department of Management and Business Economics, Budapest, Hungary
zakariazineelabidine@edu.bme.hu, dimeny.imre@gtk.bme.hu, koltai.tamas@gtk.bme.hu

Abstract. Efficient assembly lines aim to achieve a shorter makespan to meet customer demands by reducing delivery times. Equally important is the balancing of the workloads of stations to prevent worker overload and to ensure a smooth production flow. Nonetheless, minimizing the makespan and balancing the workload are two conflicting objectives in the presence of learning, as a shorter makespan requires an unbalanced workload distribution. This study introduces a Mixed Integer Linear Programming (MILP) model to analyze the tradeoff between makespan and balanced workload. Using a practical problem, the possible compromises of the two conflicting objectives are explored, and valuable insights for managerial decision-making are provided.

Keywords: Assembly line balancing, learning effect, learning curve, operations management, mathematical programming.

1. Introduction

Assembly lines are flow-oriented production systems with workstations arranged along a conveyor belt. Parts are gradually assembled by moving from station to station. The complete work to make a product is split into simple, indivisible operations called tasks. Balancing an assembly line implies assigning tasks to workstations to improve one or more performance metrics without violating precedence relations between the tasks. Since assembly line tasks are intrinsically repetitive, task times decrease with repetition due to the effect of learning, especially in small to medium batch production of up to a few hundred units [3]. The growing demand for product customization drives the significance of small lot size production compared to mass production, underscoring the importance of the effect of learning in assembly lines [1].

One of the objectives of assembly line balancing (ALB) is to minimize the makespan. The makespan (Also referred to as throughput time or total production time) is the time necessary to complete a predetermined production batch. Achieving a shorter makespan implies a shorter delivery time for orders. Another critical measure is the workload distribution among the stations. A balanced workload reduces the risk of worker's overload and contributes to a smoother production flow [7]. In the presence of learning, however, minimizing the makespan and balancing the workload among the stations are two conflicting objectives as a smaller makespan requires an uneven workload allocation across the stations [2]. This paper aims to identify an optimal workload distribution strategy that minimizes the makespan and workload disparity among stations when the effect of learning is considered.

The remainder of this paper is structured as follows: Section 2 outlines the formulation of a MILP model designed to minimize the makespan within a specific workload difference range. Section 3 demonstrates the application of the model with the help of a benchmark problem and discusses the obtained results. Finally, the main results are summarized, and some conclusions are given.

2. Formulation of the model

Three MILP models are used to calculate the possible combinations of optimal makespan and workload spread, where the workload range *(R)* defined by the difference between the minimum and maximum initial station times describes the workload spread. The notation used in the following part of the paper is given in Table 1, while model formulations are summarized in Table 2.

The following mixed integer linear programming model assigns tasks (denoted by index $i=1,\dots, I$) to workstations ($j=1,\dots, J$), considering task precedence relations. The objective is to minimize the makespan (M) of the production of a defined number of parts (N). Task time progressively decreases as a consequence of the accumulated learning. t_{in} expresses the time necessary to execute task i in the n^{th} repetition. In this model, we assume that task execution time decreases exponentially with the number of repetitions according to the Wright exponential learning curve ($t_{in} = t_{i,1}n^b$) [6].

First, the minimum feasible workload range is calculated without learning to minimize the difference between the minimum and maximum workload of stations. This is described by objective (2) and constraints (3), (4), (8) in Table 2. As per constraint (3), each task must be allocated to a specific station. Constraint (4) ensures that all task precedence relations are satisfied. Constraint (8) defines the objective value function as the maximum workload differences between stations.

The maximum workload range is provided by the allocation that minimizes the makespan when learning is considered and with no restriction on the allowed range. The second model minimizes the makespan with objective function (1) and applying constraints (3), (4), (5), (6), (7). A batch with size N on a defined number of stations J is produced in $J+N-1$ cycles. In the first $J-1$ cycle, the first part will enter the station with the same cycle number as the station (in the first cycle to station 1, in the second cycle to station 2, and so on). Starting from cycle J, a new piece is completed in each cycle, as long as the required N cycles are finished, that is, the required batch size N is completed. In the last $J-1$ cycles, gradually, all stations will stop working, and the last $J-1$ pieces leave the stations step by step. The cycle time of each cycle is changing, and the bottleneck station may also change. The cycle time in each cycle equals the total execution time of tasks assigned to the bottleneck station of the cycle. Constraint (5) describes the cycle times during the run-up period when no part is completed yet, and the first part is on stations $s<J$. According to constraint (6), the cycle time when completing the n^{th} part is greater than the total execution

time of tasks at any of the stations at that time, considering again that the execution time at each station depends on the number of parts currently finished at that station. The makespan or the total production time will be the sum of the cycle times related to the $J+N-1$ cycles as described in constraint (7).

Finally, in the third model, objective value function (1) and constraints (3), (4), (5), (6), (7), and (8) describe the model for minimizing the makespan when there is a restriction on the workload range.

Table 1. Summary of notation applied in the model.

Indices:

i, k	- index of tasks (i=1…, I; k=1…, I),
j, s	- index of workstations (j=1,…, J; s=1,…, J),
n	- index of parts produced (n=1,…, N).

Parameters:

b	- parameter of the exponential learning curve expressing learning,
I	- number of tasks,
J	- number of workstations,
N	- number of parts produced,
t_{in}	- time necessary to execute task i in the n^{th} repetition.

Sets:

P_i	- set of indices of those tasks which must be finished before task i is started.

Decision variables:

x_{ij}	- 0-1 decision variable; if it is equal to 1, then task i is assigned to station j, otherwise, it is equal to 0,
u_s	- cycle time when executing the first part on station s, where $s < J$,
v_n	- cycle time when completing part n,
M	- makespan or total production time,
R	- range of station workloads.

Table 2. MILP models for minimizing the makespan with learning.

$$Min(M) \tag{1}$$

$$Min(R) \tag{2}$$

$$\sum_j x_{i,j} = 1 \quad \forall\, i \tag{3}$$

$$\sum_j j\left(x_{i,j} - x_{k,j}\right) \geq 0 \quad \forall\, (i,k) |\, k \in P_i \tag{4}$$

$$u_s \geq \sum_i x_{i,j} * t_{i,(s-j+1)} \quad \forall\, (j,s) | s < J, s - N + 1 \leq\, j \leq s \tag{5}$$

$$v_n \geq \sum_i x_{i,j} * t_{i,(n-j+J)} \quad \forall\, (j,n) | 1 \leq n + J - j \leq N \tag{6}$$

$$M = \sum_n v_n + \sum_{s<J} u_s \tag{7}$$

$$R \geq \sum_i x_{i,j} * t_{i,1} - \sum_i x_{i,s} * t_{i,1}\ \forall\, (j,s) \tag{8}$$

3. Practical illustration of the presented model

3.1 Problem description

To illustrate the performance of the proposed model, let us consider a well-known benchmark problem from Scholl (1993) named by Gunther [5]. The precedence graph of tasks and the task times (in minutes) of the problem are given in Fig. 1.

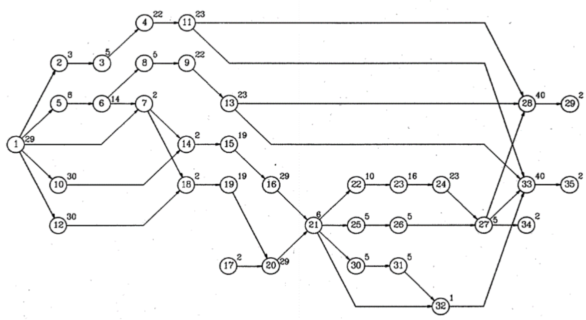

Fig. 1. Gunther precedence graph.

The model presented in Table 2 is applied to determine the optimal makespan of the benchmark problem within a specific initial workload difference range when the learning effect is considered. We implemented the algorithm and generated the results using the AIMMS Prescriptive Analytics Platform [4]. The MILP model is solved with CPLEX version 12.7.1.

The effect of learning on the optimal makespan is analyzed using the Wright LC model [5]. For the analysis, we selected a learning rate (*L*) of 0.85 (*L*=0.85), corresponding to fast learning, assuming that the learning rate is the same for all the stations. Quantity-dependent task execution times are precalculated and rounded to 2 decimals. For illustration purposes, a line with four stations (*J*=4) and a batch size of 50 units (*N*=50) were selected; however, this approach can be valid for any line length and batch size.

3.2 Analysis of the optimal makespan and workload distribution

Table 3 summarizes the initial workload per station, workload difference range, initial workload differences, total workload differences in minutes and percentages, and the corresponding optimal makespans obtained for the selected learning rate (*L*=0.85), line length (*J*=4), and quantity (*N*=50).

Table 3. Results summary

Initial workload per station (mn)				Initial workload difference range (mn)	Initial workload difference (mn)	Total workload difference (mn)	Total workload difference (%)	Optimal Makespan (mn)
Station 1	Station 2	Station 3	Station 4					
121.00	121.00	121.00	120.00		1.00	25.51	0.83%	3426.95
122.00	121.00	120.00	120.00		2.00	51.00	1.64%	3424.98
122.00	121.00	121.00	119.00	[1,5]	3.00	76.50	2.46%	3415.85
122.00	122.00	121.00	118.00		4.00	102.10	3.28%	3411.42
123.00	122.00	120.00	118.00		5.00	127.60	4.07%	3409.39

The results indicate that the lowest initial workload difference is 1 minute, corresponding to a total workload difference of 25.51minutes (0.83%) and an optimal makespan of 3426.95 minutes. The results also show that the lowest optimal makespan is 3409.39 minutes, corresponding to an initial workload difference of 5 minutes and a total workload difference of 127.69 minutes (4.07%). These two points specify the initial workload difference range of 1 to 5. Going below this range does not provide any feasible solution, and going above this range is not beneficial as it only generates solutions for a higher workload difference and makespan. As Table 2 shows, five possible solutions are possible within the obtained range [1,5].

The results demonstrate that the optimal makespan decreases as the initial workload difference and total workload increase. A shorter makespan implies a shorter production time and faster delivery time, with some operators working more than others. On the other hand, a minimal workload difference between the stations and operators implies an even workload distribution between the workers and, therefore, a reduced risk of workers overload and a smoother production flow with a longer makespan. The decision rests with the managers in making a compromise between the workload distribution and makespan, depending on their objectives.

4. Conclusion

In this paper, a MILP model is formulated and applied when the makespan and workload difference between operators is minimized in the presence of learning. The MILP model developed in section 2 is illustrated with a benchmark problem in section 3. The effect of learning is characterized using the Wright exponential learning model, assuming that the learning rate is the same for all the stations. The model, however, can be extended to any learning curve model or empirical data. In the illustrative example, a learning rate of 0.85, a batch size of 50 units, and a four-station line were chosen. The approach, however, can be valid for any learning rate, line length, and batch size.

The present research investigates the evolution of the optimal makespan within a specific workload difference range. As a result, the possible optimal makespans, the corresponding initial workloads per station, and total workload differences within the specified range were provided. The findings indicate that the optimal makespan decreases as the difference between the initial and total workloads increases, highlighting a tradeoff between the optimal makespan and workload distribution. The findings can be valuable for managers as they can select the workload distribution and corresponding makespan according to their objectives.

References

1. Bukchin, Y., Wexler, E.: The effect of buffers and work sharing on makespan improvement of small batches in assembly lines under learning effects. IIE Transactions 48(5), 403–414 (2016).
2. Cohen, Y., Dar-El, M.E., Vitner, G., Sarin, S.C.: Allocation of work to the stations of an assembly line with buffers between stations and three general learning patterns. International Journal of Intelligent Systems Technologies and Applications 4(1–2), 123–140 (2008).
3. Globerson, S., Shtub, A.: The impact of learning curves on the design of long cycle time lines. Industrial Management 26(3), 5–10 (1984).
4. Roelofs, M., & Bisschop, J.: AIMMS Language Reference, 2010. URL: http://www.aimms.com/aimms/download/manuals/aimms_ref.pdf (2018).
5. Scholl, Armin. Data of assembly line balancing problems. Techn. Hochsch., Inst. für Betriebswirtschaftslehre (1995).
6. Tamás, A., Koltai, T.: Application of Learning Curves in Operations Management Decisions. Periodica Polytechnica Social and Management Sciences 28(1), 81–90 (2020).
7. Zeltzer, L., Aghezzaf, E.-H., & Limère, V.: Workload balancing and manufacturing complexity leveling in mixed-model assembly lines. International Journal of Production Research, 55(10), 2829–2844 (2017).

Quantifying the impact of outlier management techniques on digital country rankings

Zoltán Bánhidi[1], Imre Dobos[1]

[1] Dept. of Economics, Budapest University of Technology and Economics, Budapest, Hungary
banhidi.zoltan@gtk.bme.hu

Abstract. The objective of our study is to create rankings of European Union (EU) member states based on objective weights that provide a comprehensive overview of their digital and economic development. We also aim to examine the impact of outlier management techniques, such as winsorizing, on these rankings. To accomplish this, we utilized a macro-level cross-sectional dataset that comprises the principal dimensions of the Digital Economy and Society Index, as published by the European Commission, along with the GDP per capita and AIC indicators from economic statistics. In one version of the dataset, extreme values in the macroeconomic data were treated with winsorizing, while in another version, they were left untreated. The efficiency indicators were used to rank EU Member States based on the synthesis of the digital and economic dimensions using the DEA/MaxMin model from decision theory. The rankings aim to characterize the digital-economic strengths of EU countries and the digital divide found within the EU, as well as evaluate the impact of outlier management.

Keywords: Winsorizing, DEA, DESI.

1 Introduction

The relationship between digital and economic development is complex and bidirectional. Digital transformation refers to the process by which information and communication technologies (ICTs) increasingly permeate the way the economy and society (businesses, governments, and citizens) function and live. Economic development, in turn, is linked to GDP growth, employment growth, and improvements in living standards, which in modern economies are closely linked to and interact with digital transformation.

Reliable measurement of the various aspects of digital development is crucial for governments to design and implement effective digital strategies that drive digital transformation. A sound measurement system can identify the areas with the greatest impact on digital transformation, as well as the gaps and weaknesses where government intervention and incentives are most needed.

Although the European Commission's Digital Economy and Society Index (DESI) [1], first published in 2014, has been the subject of intense expert debate since its inception, it is one of the most important and frequently cited indicators for characterizing the process of digitization in Europe. However, it is important to note that there are identified problems with the survey methodology and data collection systems on which it is based. Expert debates on this topic have primarily taken place in policy workshops and forums with EU institutions and national governments, but the database has also been utilized in academic research, as seen in the works of Kotarba [2], Moroz [3], Giannone and Santaniello [3] and Laitsu et al. [4] etc.

In this study, we utilize the MaxMin variant of the Data Envelopment Analysis method (DEA/MaxMin) from decision theory, to analyze the five primary dimensions of the 2020 edition of DESI. Our cross-sectional dataset includes country-level data from well-known macroeconomic indicators such as AIC and GDP per capita from Eurostat. The objective is to provide a comprehensive overview of the digital economic strengths of the European Union (EU) Member States. Our analysis was conducted for the 28-member EU (the EU-28 group of countries) prior to 'Brexit' since the datasets used still included the UK leaving the EU in 2020. (The 2020 DESI report was based on 2019 baseline data.)

In addition, we aim to analyze the effect of outlier management techniques, such as winsorizing, on these rankings. To achieve this, we utilize two versions of our dataset. In one version, the extreme values in the raw AIC and GDP per capita data were treated with winsorizing, while in the other version, they were left untreated.

Table 1 provides an overview of our variables (DESI dimensions and macroeconomic indicators), along with their shares in the original DESI composite index (which does not include GDP and AIC) and a brief description summarizing their content.

Table 1. The main dimensions of the DESI 2020 and their weights, as recommended by the European Commission, supplemented by the GDP/cap and AIC macroeconomic indicators.

Dimensions	**Weights**	**Description**
Connectivity (CNC)	25%	Fixed and mobile broadband networks and prices
Human Capital (HUC)	25%	Basic and advanced digital skills, digital literacy
Integration of Digital Technology (IDT)	20%	Digitization of enterprises and e-commerce
Digital Public Services (DPS)	15%	E-government services
Use of Internet (UOI)	15%	Citizens' online content consumption, communication and transactions
Gross Domestic Product (GDP)	N/A	The value of total final output of goods and services produced by the economy
Actual Individual Consumption (AIC)	N/A	A measure of all goods and services actually consumed by households

2 Methodology

Our research employs a decision theoretic method to investigate digital and economic development and rank the countries in our dataset, namely the DEA/MaxMin. We also use outlier management (winsorizing) to prepare our data for analysis.

DEA is a mathematical programming method used to measure the relative efficiency of decision-making units (DMUs) and can also serve as a ranking method. Its main objective is to find the weight vector that yields the maximum efficiency for each decision unit. In the original DEA model (DEA/CCR, Charnes-Cooper-Rhodes), efficiency is measured by a quotient-type indicator with the weighted value of the output criteria in the numerator and the weighted value of the inputs in the denominator. In contrast to DEA/CCR, the DEA model we use only considers output criteria, the inputs are assumed to be constant (specifically, equal to one). This approach is referred to as DEA Without Explicit Input (DEA/WEI) in the literature. Additionally, the DEA/CCR model assigns separate weights to each DMU, which is not desirable in our case due to fairness concerns, among other reasons. The DEA Common Weights Analysis (DEA/CWA) approach that we employ addresses this issue by utilizing uniform weights to rank all DMUs, i.e., the EU members states in our dataset.

The vector of possible weights of the DEA/MaxMin model can be determined by the system of equations (1) to (2). Inequalities (1) shows the upper limit of DEA efficiency, i.e. one, while inequality (2) defines the non-negativity of weights. The number of decision-making units is p, and vector $\mathbf{y}_j$ comprises the values of the jth decision making unit, in this case country. The vectors $\mathbf{y}_j$ can be summarized in the $\mathbf{Y}$ matrix. Vector $\mathbf{u}$ is the DEA weights.

$$\mathbf{u} \cdot \mathbf{y}_j \leq 1;\ j = 1,2,...,p \tag{1}$$

$$\mathbf{u} \geq \mathbf{0}. \tag{2}$$

$$F(\mathbf{u}) = \min_{1 \leq j \leq p} \mathbf{u} \cdot \boldsymbol{y_j} \rightarrow \max \tag{3}$$

The DEA efficiency of the jth DMU (in our case, jth EU member state) for a given weight $\mathbf{u}$ is equal to $E_j(\mathbf{u}) = \mathbf{u} \cdot \mathbf{y}_j$, and the efficiencies of all DMU with common weight vector $\mathbf{u}$ in vector form is equal to $F(\mathbf{u}) = \min_{1 \leq j \leq p} \mathbf{u} \cdot \boldsymbol{y_j}$.

In this methodology section, we should also briefly describe the methodology for managing outliers and the factors that make this necessary. Multinational companies' tax optimization decisions in Luxembourg and Ireland may skew macroeconomic data, particularly the GDP per capita indicator. Therefore, the GDP per capita figures for Luxembourg and Ireland might not accurately reflect the true strength of their respective economies. A study on the Central Bank of Ireland's website, authored by the bank’s former governor, acknowledges the distortion present in Ireland [5].

For this reason, we have decided to apply outlier management using the winsorization method. Winsorization performs a symmetric "truncation", i.e. the values rounded from the bottom and the top are increased from the bottom and decreased from the top by the same number. The method assumes that the same number of pieces of data are changed in the data distribution, and that the values above and below the specified upper and lower values are kept constant. This means that if we perform a 90 percent winsorization, as we did, we replace the observations in the bottom 5 percent with one constant and the values in the top 95 percent with another constant.

The formula applied to the data is given by equation (4):

$$x_{\lfloor (n-1)p+1 \rfloor} + [(n-1)p + 1 - \lfloor (n-1)p+1 \rfloor]\left(x_{\lfloor (n-1)p+1 \rfloor + 1} - x_{\lfloor (n-1)p+1 \rfloor}\right) \quad (4)$$

Countries whose AIC and/or GDP values have been treated with winsorization because they are deemed to be too high or too low are highlighted in Table 2, indicating a reduction of the high value for Luxembourg, Ireland, and Germany and an increase for Bulgaria and Croatia. The GDP and AIC data of the other countries, as well as the data related to the DESI dimensions, were left untreated.

Table 2. The original and winsorized GDP and AIC values (for the treated countries)

Country	GDP/cap PPS (EU = 100)	AIC/cap PPS (EU = 100)	GDP/cap PPS (winsorized)	AIC/cap PPS (winsorized)
Bulgaria	53	59	65.9	66.3
Croatia	65	66	65.9	66.3
Germany	121	123	121.0	121.5
Ireland	191	97	172.4	97.0
Luxembourg	261	135	172.4	121.5

3 Results

In Table 3, we present the joint DEA/MaxMin weights obtained on the raw ("basic") data as well as those obtained by calculations on the winsorized data. As can be seen from the table, the UOI and IDT dimensions do not play a significant role in determining DEA efficiency or rankings in either version, but the weights of the other dimensions show differences depending on the version. Although the DPS dimension plays the largest role in determining efficiency in both models, the role of AIC is equally important in the winsorized model. In both models, the HUC dimension also plays a (smaller) role, but CNC plays a role only in the base model, while GDP plays a role only in the winsorized model. The numbers in the table were rounded to three decimal places, but the numbers in gray are equal to zero even without rounding.

Table 3. The weights with and without winsorizing

Weights	**CNC**	**HUC**	**UOI**	**IDT**	**DPS**	**GDP**	**AIC**
Basic	0.001	0.001	0.000	0.000	0.005	0.000	0.003
Winsorized	0.000	0.002	0.000	0.000	0.004	0.001	0.004

Once the weights have been determined, we proceed to the determination of the efficiencies and the ranking itself, which is presented in two versions. As can be observed from Table 4, winsorizing did not result in a decline in the rankings for Luxembourg and Ireland, even though their GDP values (and Luxembourg's AIC value) were reduced. On the contrary, while the former maintained their lead, Ireland advanced four places by incorporating GDP into the weighting.

Table 4. The DEA/MaxMin efficiencies and rankings with and without winsorizing

Countries	Basic data	Ranking with basic data	Winsorized data	Ranking with wins. data
Luxembourg	1.000	1	1.000	1
Denmark	1.000	1	1.000	1
Finland	1.000	1	1.000	1
Austria	0.946	6	0.975	4
Netherlands	0.958	4	0.972	5
Sweden	0.955	5	0.963	6
Germany	0.898	7	0.927	7
Ireland	0.884	12	0.926	8
United Kingdom	0.889	9	0.916	9
Belgium	0.885	11	0.909	10
France	0.885	10	0.891	11
Spain	0.894	8	0.849	12
Estonia	0.865	13	0.818	13
Lithuania	0.841	14	0.809	14
Malta	0.825	15	0.798	15
Italy	0.784	18	0.783	16
Cyprus	0.770	19	0.774	17
Portugal	0.792	17	0.757	18
Slovenia	0.762	20	0.744	19
Czechia	0.725	21	0.731	20
Latvia	0.802	16	0.723	21
Poland	0.724	22	0.693	22
Slovakia	0.643	24	0.632	23
Greece	0.609	27	0.616	24
Hungary	0.649	23	0.615	25
Croatia	0.622	26	0.611	26
Romania	0.623	25	0.610	27
Bulgaria	0.609	28	0.610	28

Apart from Ireland, the two countries with the most significant changes in ranking (due to the change in vector weights) are Latvia and Spain, whose data remained unchanged after the winsorization, with a drop of five and four positions, respectively. The Pearson correlation coefficient between the DEA scores is 0.981, indicating that winsorization of the data has minimal impact on the order obtained by the underlying data. The rank-to-rank correlation is measured by Kendall's tau-b, with a value of 0.884, also demonstrating a high correlation. The two correlations yield comparable results, indicating that the treatment of outlier data does not significantly affect this sample. Therefore, it is not worthwhile to perform winsorization on the orders.

4 Conclusions

In our study, we employed objective weights and models based on the statistical characteristics of our dataset, which encompassed digital and economic dimensions. These models were utilized to create country rankings that we believe provide a comprehensive picture of the digital economic strengths and development of the European Union (EU) Member States. Based on the five digital indicators (DESI dimensions) and two macro indicators (AIC, GDP/capita), we established rankings for the EU-28 (including the UK) using the DEA/MaxMin method.

The weights obtained from the raw and winsorized data exhibited minimal discrepancy from one another, yet exhibited notable divergence from the weights set by the European Commission. Nevertheless, the resulting country rankings remained largely consistent with one another and with the original rankings. In the winsorized rankings, only a few countries show noticeable differences in their ranking, which (except for Ireland) were not necessarily the countries whose outliers were adjusted. Overall, therefore, we can conclude that the main effect of winsorization on the rankings is through the modification of the DEA/MaxMin weights. However, even this effect is relatively minor.

References

1. Digital Economy and Society Index 2020, https://digital-strategy.ec.europa.eu/en/library/digital-economy-and-society-index-desi-2020, last accessed 2024/04/14
2. Kotarba, M.: Measuring digitalization–key metrics. Foundations of Management 9(1), 123–138 (2017).
3. Moroz, M.: The level of development of the digital economy in Poland and selected European countries: a comparative analysis. Foundations of Management 9(1), 175–190 (2017).
4. Giannone, D., Santaniello, M.: Governance by indicators: the case of the Digital Agenda for Europe. Information, Communication & Society 22(13), 1889–1902 (2019).
5. Laitsou, E., Kargas, A., Varoutas, D.: Digital competitiveness in the European Union era: The Greek case. Economies, 8(4), 85 (2020).
6. Honohan, P.: Is Ireland really the most prosperous country in Europe?. Economic Letters 1 (2021).

Constraint Programming formulation for a real-world final exam scheduling problem with parallel sessions based on short time intervals

László Kálmán Trautsch[1] and Bence Kovari[1]

[1] Department of Automation and Applied Informatics, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary
trautschl@edu.bme.hu

Abstract. Scheduling final exams is subject to various requirements that differ by country and university. A range of personal, institutional, and regulatory factors should be considered at the same time for creating an optimal schedule. We propose a Constraint Programming model for scheduling final examinations at the Department of Automation and Applied Informatics, Budapest University of Technology and Economics. The requirements of this scheduling problem regard an examination period which is divided into 5-minute intervals. Heterogeneous student groups and instructors with various roles are scheduled to parallel sessions based on these time intervals. The cost function is defined by multiple types of constraints, such as balancing the workload, optimizing start and end times of sessions and breaks, and penalizing gaps in the schedules of instructors. We present the formulation of the various complex requirements of final exam scheduling and demonstrate the results of applying a Constraint Programming solver to our model to find feasible solutions for a real-world scheduling problem involving 101 students.

Keywords: Constraint Programming, Scheduling, Final exam scheduling

1 Problem description

Final exam scheduling is a combinatorial optimization problem in which various constraints restrict the search space and the optimization is based on multiple, often conflicting objectives. The task of our addressed problem is to schedule exams during the examination period which is divided into short time intervals. Parallel examination is enabled with possible assignments to different rooms. The exams during a day in a room should form consecutive sessions, with the lunch break being the only gap during these blocks of exams. Instructors should be assigned to the exams to fulfill different roles. Exams are heterogeneous which means that their length and the required instructor roles can differ.

The Constraint Programming formulation of final exam scheduling is presented through an actual complex problem of the Department of Automation and Applied

Informatics, Budapest University of Technology and Economics. For the sake of brevity only the more important parameters, variables, and constraints are introduced. The actual formulation of the problem was done using other similar restricting constraints and an extended objective function.

The main idea to reduce the number of variables and constraints of the problem is to use exam slots and formulate the constraints based on these slots whenever possible, instead of the many 5-minute time intervals of the planning horizon. The maximum number of exams that every continuous session of exams can contain is specified for the problem, therefore that many exam slots should be created for every block of exams on each day and each room. An exam slot can be empty or exactly one exam could be assigned to it. The proposed Constraint Programming model is presented in this section.

1.1 Parameters

- D: Set of days
- T: Set of 5-minute intervals in the planning period
- R: Set of rooms
- E: Set of exams, one exam corresponding to each student
- I: Set of instructors
- $MaxB$: Number of blocks that can be scheduled to a room in a day
- B: Set of possible exam blocks in the planning period considering every room
- $MaxS$: Number of exam slots that can be scheduled to a block
- S: Set of possible slots for exams in the planning period considering every room. The slots are counted starting from number 1.
- B_s: Block of slot s
- P_s: Set of the slots that can be scheduled in parallel with slot s
- $PStart_s$: Set of possible start intervals of slot s
- $AllowedEnd_b$: Set of allowed end intervals of block b, for which no soft constraint is violated
- $length_e \in \{8,9\}$: Length of exam e, which is based on the major of the corresponding student
- C: Set of soft constraints
- w_c: Weight corresponding to soft constraint c
- U: Set of workload upper limit soft constraints
- L: Set of workload lower limit soft constraints
- $StartC$: Set of session start time soft constraints
- EndC: Set of session end time soft constraints
- $h_{r,i}$: Indicator whether instructor i can hold role r

- $h_{r,e}$: Indicator whether exam e needs role r
- d_i: Workload demand for instructor i
- p_u: Upper limit of workload constraint u
- p_l: Lower limit of workload constraint l
- startL_{sc}: Minimum start interval of session start time constraint sc
- endL_{ec}: Maximum end interval of session end time constraint ec

1.2 Decision variables

- $x_{s,e} \in \{0,1\}$: 1 if exam e is assigned to slot s, 0 otherwise
- $x_{s,i} \in \{0,1\}$: 1 if instructor i is assigned to slot s, 0 otherwise
- $start_s \in PStart_s$: Start interval of slot s
- $length_s \in \{0,8,9\}$: Length of slot s
- $end_b \in \mathbb{Z}_{\geq 0}$: End interval of block b
- $isGap_{s,i} \in \{0,1\}$: 1 if instructor i has a gap in the schedule starting from slot s, 0 otherwise
- $workladV_{l,i} \in \{0,1\}$: 1 if the workload of instructor i violates the workload limit soft constraint l, 0 otherwise
- $startV_{sc,b} \in \{0,1\}$: 1 if the start time of block b violates the session start time soft constraint sc, 0 otherwise
- $endV_{ec,b} \in \{0,1\}$: 1 if the end time of block b violates the session end time soft constraint ec, 0 otherwise

1.3 Constraints

Each slot can hold maximum one exam:

$$\sum_{e \in E} x_{s,e} \leq 1 \qquad \forall s \in S \tag{1}$$

Each exam is scheduled to exactly one slot:

$$\sum_{s \in S} x_{s,e} = 1 \qquad \forall e \in E \tag{2}$$

The next slot of a block starts after the previous one ended:

$$start_{s+1} = length_s + start_s \qquad \forall s \in S : s \bmod MaxS > 0 \tag{3}$$

The block ends when its last slot ends:

$$end_b = length_s + start_s \qquad \forall s \in S : s \bmod MaxS = 0\,, b = B_s \tag{4}$$

A slot has the same length as the exam scheduled in it:

$$\neg x_{s,e} \lor (length_s = length_e) \qquad \forall s \in S\,, \forall e \in E \tag{5}$$

The length of a slot is 0 when it holds no exam:

$$\left(\bigvee_{e \in E} x_{s,e}\right) \vee (length_s = 0) \qquad \forall s \in S \tag{6}$$

At least one instructor is scheduled for each role of the exam held in a slot:

$$\neg x_{s,e} \vee \neg h_{r,e} \bigvee_{i \in I} (x_{s,i} \wedge h_{r,i}) \qquad \forall s \in S, \forall e \in E, \forall r \in R \tag{7}$$

1.4 Objective function

The objective function is the minimization of violations of the various constraints, weighted by the corresponding parameters. e_c is the expression for the number of violations of a given constraint.

$$min \sum_{c \in C} e_c * w_c \tag{8}$$

The basic idea of penalizing gaps in the schedules of instructors is as follows: If an instructor is scheduled to an exam slot, then that slot is either the start of a gap, or the instructor is also scheduled to another slot that begins immediately after the current slot ends (either in the same block or in a parallel section in a different room), or the slot ends at an allowed time:

$$\neg x_{s,i} \vee isGap_{s,i} \vee x_{s+1,i} \bigvee_{\hat{s} \in P_s} (x_{\hat{s},i} \wedge (start_{\hat{s}} = start_s + length_s))$$
$$\vee (start_s + length_s = AllowedEnd_b)$$
$$\forall \mathrm{i} \in \mathrm{I}, \ \forall \mathrm{s} \in \mathrm{S} : s \bmod MaxS > 0, b = B_S \tag{9}$$

The last slot of a block can also start a gap in the schedule of an instructor:

$$\neg x_{s,i} \vee isGap_{s,i} \bigvee_{\hat{s} \in P_s} (x_{\hat{s},i} \wedge (start_{\hat{s}} = end_b)) \vee (end_b = AllowedEnd_b)$$
$$\forall \mathrm{i} \in \mathrm{I}, \ \forall \mathrm{s} \in \mathrm{S} : s \bmod MaxS = 0, \ b = B_S \tag{10}$$

The number of gap variables is minimized by the objective function, using the expression for the number of violations:

$$e_{gap} = \sum_{s \in S} \sum_{i \in I} isGap_{s,i} \tag{11}$$

The workload demand for each instructor is the maximum of their role demands:

$$d_i = Max_{r \in R} \left(\frac{\sum_{e \in E} h_{\mathrm{r},e}}{\sum_{j \in I} h_{\mathrm{r},j}} : h_{r,i} \right) \qquad \forall \mathrm{i} \in \mathrm{I} \tag{12}$$

An instructor should be scheduled to fewer exam slots than the limit of a given workload demand, or that constraint is violated:

$$\left(\sum_{s \in S} x_{s,i} \le d_i * p_u\right) \vee workloadV_{u,i} \qquad \forall i \in I, \forall u \in U \tag{13}$$

An instructor should be scheduled to more exam slots than the limit of a given workload demand, or that constraint is violated:

$$(\sum_{s \in S} x_{s,i} \geq d_i * p_l) \lor workloadV_{l,i} \qquad \forall i \in I, \forall l \in L \tag{14}$$

The number of workload violations is minimized by the objective function, using different expressions for each upper and lower limit:

$$e_l = \sum_{i \in I} workloadV_{l,i} \qquad \forall l \in (L \cup U) \tag{15}$$

A constraint regarding the start time of a block is violated when the first slot of the block starts too early:

$$(start_s \geq startL_{sc}) \lor startV_{sc,b}$$

$$\forall sc \in StartC, \forall s \in S : s \bmod MaxS = 1, b = B_s \tag{16}$$

A constraint regarding the end time of a block is violated when the block ends later than allowed:

$$(end_b \leq endL_{ec}) \lor endV_{ec,b} \qquad \forall ec \in EndC, \forall b \in B \tag{17}$$

The number of start and end time violations are minimized similarly to the workload constraints, using dedicated expressions. These constraints are also used to optimize the start and end times of breaks between exam sessions.

$$e_{sc} = \sum_{b \in B} startV_{sc,b} \qquad \forall sc \in StartC \tag{18}$$

$$e_{ec} = \sum_{b \in B} endV_{ec,b} \qquad \forall ec \in EndC \tag{19}$$

2 Results

The presented Constraint Programming model was tested using CP-SAT solver [1] in a Python implementation. The dataset used for testing was an actual problem of final exam scheduling, containing information about a heterogeneous group of 101 students and 82 instructors during a 7-day planning period with two rooms for parallel scheduling. The test was executed for 7 hours of runtime, using a machine with an i5 @ 3.30 GHz processor and 16GB of RAM. During the experiment, 362 different feasible solutions were found. The initial upper bound of the cost function calculated by the solver was 84983, the final lower bound was 10. The first feasible solution found had a cost of 10205, and the best had a cost of 1179. The decrease in the cost of feasible solutions during the runtime can be seen in Fig. 1.

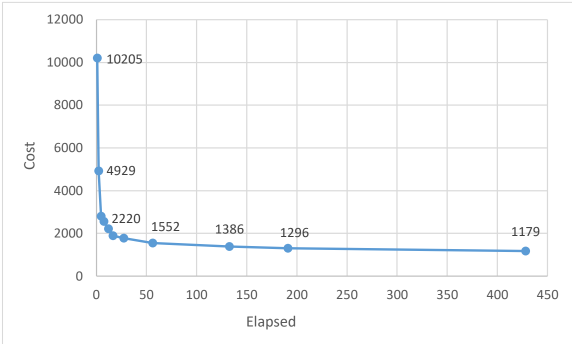

Fig. 1. The cost decrease of feasible solutions found by the solver during the runtime.

The results show that using our formulation the Constraint Programming solver was able to find feasible solutions for the presented complex real-world final exam scheduling problem.

Acknowledgement

The presented research has been financially supported by the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory. Financial supports are gratefully acknowledged.

References

1. OR-Tools CP-SAT v9.7. Laurent Perron and Frédéric Didier. https://developers.google.com/optimization/cp/cp_solver, last accessed 2024/04/017.