Production line balancing by P-graphs

Aniko Bartos¹ · Botond Bertok¹

Received: 22 March 2019 / Revised: 30 July 2019 / Accepted: 30 July 2019 / Published online: 6 August 2019
© The Author(s) 2019

Abstract

Assembly industry plays a key role in Central and Eastern Europe. Large companies and their subcontractors manufacture automotive and electronic products from components, employing a significant number of human resources. Due to the growing labor shortage, it is critical that the production lines should be optimally loaded, i.e., the tasks have to be evenly distributed among the workstations according to their cycle times. In this article a novel formulation of the problem by process graphs or P-graphs is presented leading in an easy to follow visual definition of the potential task to employee allocations, as well as the options to generate a mathematical programming model algorithmically, to be solved by general purpose solvers or get the optimal and alternative N -best allocations by P-graph software. In addition to the theoretical presentation, the article shows the results achieved by applying the proposed methodology in a real-world environment in a computer assembly plant. The P-graph approach provides visual modeling by graphs in a graphical editor and helps understand the relations of decision variables while generating the corresponding mathematical model, which can be generalized for a class of problems and rebuilt according to actual data. As a result, the basis for rigorous mathematical optimization-based decision support can be built up according to graphical models easily understandable by end users as well.

Keywords Line balancing · Optimization · P-graph · Production line

1 Introduction

In Central and Eastern Europe assembly plants have a major role in the industry. In automotive and electronic assembly factories the products are assembled from ready-made parts in production lines mainly by human resources. Typically, various steps must be performed sequentially one after the other done by trained workers.

✉ Aniko Bartos
bartos@dcs.uni-pannon.hu

¹ Department of Computer Science and Systems Technology, University of Pannonia, 10, Egyetem Street, Veszprem 8200, Hungary

Although such work does not require special qualifications, the factories are faced with significant labor shortages. It is important for the fulfillment of orders that the plant gets the most out of its resources, thus optimize the task-worker allocation. The literature calls such optimization as line balancing, and it is highly beneficial where mass production occurs and the products have been manufactured for a longer period of time in the same production line (Sivasankaran and Shahabudeen 2014).

Although the conveyor belt was first used by meat factories in Chicago, Ford made it known in 1913. Even in these early times, they tried to schedule production (Wilson 2014). In 1955, Salveson identified the “balance delay”, or wasted time, as an objective (Salveson 1955). Optimization was aimed at a local and global level. Optimization was extended not only to the type of work but also to the physical placement of machines and workforce. With the years and the development of factories, an increasing number of optimization solutions have been introduced, depending on the placement of the producing belts and the variation of the manufactured products (Kumar and Mahto 2013).

Among them, the simplest one is the single line model which produces one product on a single line, the mixed assembly line (Thomopoulos 1967), when more than one model of the same general product are intermixed on one assembly line, the multiproduct, where there are several products occurs on one line (Roberts and Villa 1970), and the two-sided assembly line where there are workstations on both sides of the conveyor belt (Bartholdi 1993). Taking into account the sales, solutions were created where overproduction was punished by extra cost, as well as stochastic and heuristic solutions exist (Fazlollahi et al. 2011).

In the last years, the health of workers next to the production line has become increasingly important, for example, physical (ergonomic) or even mental health, and this is already being considered when subtasks are distributed among workers (Otto and Battaia 2017). Fortunately, recycling is becoming more and more widespread, so many publications provide a solution for not only assembly but disassembly as well (Özceylan et al. 2018). Moreover, in recent years different methods are provided for finding the optimum, especially for complex classes of the line-balancing problem. Such methods include swarm optimization (Delice et al. 2017) or gravitational search algorithm (Ren et al. 2017).

Despite the extensive efforts to provide computer aid to solve the line-balancing problem, each of the proposed computational methods results a single assignment considered to be the optimal one, and none of the well-known techniques is capable to provide the N -best alternative optimal or suboptimal solutions even for the simplest single-line problem. To meet the demand for a computational tool generating alternative solutions for engineering process design problems the process graph or P-graph framework was introduced in the early 90's (Friedler et al. 1992a). P-graph is an unambiguous representation of the potentials to structural design options (Friedler et al. 1992b). P-graph is a bipartite graph, where the M-type nodes represents materials qualities appropriate to serve as inputs to outputs from the operating units, while O-type nodes identify the operating units transforming one or more material qualities to another or others. What is crucial for automated structure generation: if multiple operating units are adequate to produce a material quality of interest, they are immediately

considered as alternatives to each other. Consequently, all the feasible process structures can be algorithmically enumerated (Friedler et al. 1995) or the best once found effectively (Friedler et al. 1996).

The scope of process synthesis by P-graphs were extended to supply chains (Fan et al. 2009), occasionally relying on uncertain resources (Sule et al. 2011). The structure generation algorithms have been extended to increase process reliability by incorporating redundancy (Bertok et al. 2012b), multiperiod operations (Heckl et al. 2015), long term capacity planning (Tan and Aviso 2016), or take into consideration multiple objectives (Vance et al. 2012).

The P-graph was found to be suitable for the composition of workflows (Tick et al. 2006) and business processes (Tick et al. 2013) as well. The so-called superstructure approach, graphically revealing all the candidate routes leading to the process targets, has been established for process scheduling (Bertok et al. 2011) according to both discrete time (Garcia-Ojeda et al. 2012) and precedence based formulation (Barany et al. 2011). Finally, a general methodology for batch process scheduling has been introduced (Frits and Bertok 2014) by time constrained P-graphs (Kalauz et al. 2012).

In this article, a single line problem with rigorous constraints on the order of tasks and a case study serves for illustrating a novel approach for production line balancing by P-graphs resulting in multiple alternative best solutions. The P-graph model is constructed first following the general guidelines of batch process scheduling, then simplified in line with the distinctive features of the industrial problem of interest. Finally, the development of a decision support system at a manufacturer, evaluations, and feedbacks are displayed.

2 Problem statement

In the case study of interest, multiple different products are manufactured on a single production line from numerous parts by human resources in an assembly plant in Hungary. In spite of the broad variations of products, the set of parts plausible to be incorporated is finite. For optimization, it is necessary to know what components the product to be manufactured consists of, i.e., what assembly steps have to be performed to complete the product, and how long these steps would take. The duration of the steps was measured for each product by the factory and made available in the form of a time matrix; see Table 1 for example.

As the number of employees available is uncertain, and it is revealed only at the beginning of a shift, task distribution has to be revised at the beginning of each shift to achieve maximal efficiency. Therefore, there was a demand for an optimization algorithm and software to help task distribution, i.e., line balancing. See the input-output schema of the software in Fig. 1.

The goal was to build up a software aware of the type of product to be manufactured and the number of workers present, which is capable to determine the optimal task distribution, where the cycle time is as low as possible. It is important to note, that due to the physical locations of components inside the computer case, their insertion has to follow a fixed predefined order. Moreover if a worker is assigned to

Table 1 Time matrix

#	Name of subtask	Subtask time (s)			
		Prod.1	Prod.2	Prod.3	Prod.4
1	PuttingOn	1	2	2	2
2	Unboxing	2	2	3	3
3	Sorting	–	2	4	4
4	Checking1	–	2	2	2
5	Assembling1	5	2	10	2
6	Assembling2	–	–	4	4
7	Screwdriving	–	2	6	6
8	Wiring1	–	2	7	7
9	Wiring2	–	–	2	2
10	Checking2	–	2	2	2
11	Boxing	–	2	2	2
12	Closing	4	2	1	1
13	Labeling	–	2	1	1
14	TakingOff	3	2	2	2

Fig. 1 Schematic structure of the software

a specific workstation at the production line, he or she remains there till the end of the shift. Consequently, each worker can only perform a series of subsequent tasks on a product.

3 Methodology: problem formulation and solution by P-graphs

The rigorous structural representation of process structures proposed by Friedler and Fan is the Process Graph or P-graph in short, which graphically represents operations or activities by rectangles and their results and preconditions by solid circles as a bipartite graph; see Fig. 2. Any directed arc in the graph leads either to an operation from one of its precondition or from an operation to one of its result. It is important to note, that the primary aim of forming P-graphs not visualization but executing effective graph algorithms as part of computer-aided process design.

3.1 P-graph algorithms and software

The P-graph framework involves three major algorithms. The one to be applied first on a P-graph is algorithm MSG, which generates the maximal structure, i.e., the superstructure involving each plausible operation from those defined in the problem and potentially capable to contribute to at least one structurally feasible process network according to a set of axioms (Friedler et al. 1992b).

The second one is algorithm SSG, which can generate each combinatorially feasible process structure leading from the resources to the final targets through a network of operations and their preconditions and results. And finally, the third one is algorithm ABB (Friedler et al. 1993), which returns not only a single optimal but the N -best process structures satisfying additional constraints besides the structural ones, e.g., limitations on the availability of resources, volumes of operations, or conservation laws (Bertok et al. 2012a). Having multiple solutions is crucial to make decision-makers more aware of the diversity among different process alternatives and the consequences of selecting one of them.

Decision support systems aiding large scale process design or optimization can be constructed by setting up model prototypes in Software P-graph studio

Fig. 2 Basic P-graph notations

(Barany et al. 2018), exporting the initial model, and finally updating and resolving according to actual data; see for example the RegiOpt conceptual planner-identifying possible energy network solutions for regions (Kettl et al. 2011). As illustrated in Fig. 3, the model export and revision can be realized either in the language of process-network synthesis (PNS) or in the language of mathematical programming (MILP), but daily update in a decision support system typically requires implementations of data export and plan import interfaces from and to the enterprise resource planning (ERP) system of the customer. The consumer can decide whether the optimization is to be achieved by the help of the P-graph solver shipped together with the P-graph Studio free of charge resulting in the N-best process structures effectively (Bartos and Bertok 2019), or by a mathematical programming software carrying out a single optimal solution but being more scalable and having customer support for an annual fee. Note that the mathematical programming model can also be generated from the initial P-graph model (Bertok et al. 2012a) and exported by the software P-graph Studio as depicted in Fig. 3.

First the P-graph model then the mathematical programming is to be introduced.

Fig. 3 Decision support

3.2 Formulating the line balancing problem by P-graphs

Before developing the optimization model for periodical update and solution for daily decision support, model prototypes are created in software P-graph Studio first, as introduced in the previous section. As an initial step, a general scheduling model is constructed and then tailored due to the specialties of the line balancing problem of interest.

Let us assume that P1 and P2 are two consecutive tasks of a manufacturing process at an assembly line, and R1 and R2 are two employees. Figure 4 shows the general layout proposed by (Frits and Bertok 2014) for a two-step sequence of tasks to be realized by two potential resources R1 and R2, each of which is suitable to complete any of the two tasks P1 and P2. At the top of the P-graph in Fig. 4 R1 and R2 depicted as initial resources. Any of resources R1 and R2 can initially be assigned to any of tasks P1 and P2 by the help of operations R1toP1, R1toP2 and R2toP1, R2toP2; respectively. R1beforeP1 and R2beforeP2 are consequences of task resource assignments expressing that a resource is ready to start a task. If R1 or R2 performs task P1 through operation P1byR1 or P1byR2, then P1done appears as a consequence, which is a precondition to any operation for executing task P2, since P1 and P2 ordered elements of a two-step task sequence, i.e., their execution order is fixed. Meanwhile, the resource is released as well, and ready to continue with another task; see outcomes R1afterP1 and R2afterP1. The changeover assigning resource R1 to task P1, after completing task P1 is represented by operation R1P1P2. Similarly, R1P2P1, R2P1P2, and R2P2P1 represent changeovers of the resources from one task to another. The blue part is related to the assignments of resource R1, while the green part to the assignments of resource R2. If task P2 is

Fig. 4 General superstructure for two potential resources capable to execute any of two consecutive tasks in any order

finished by any of the operations P2byR1 or P2byR2, P2done results as the final target of the process formulated.

Since in the line balancing problem of interest in the computer assembly plant, manufacturing tasks are to be completed one after the other and any employee can work on a single product at a time, there are no parallel steps performed on a single product, no synchronization of parallel tasks is needed. Consequently, the calculation of exact time windows for each operation is not required, the total time of completing successive jobs assigned to an employee can be counted as the sum of individual completion times of the jobs. In the P-graph model in Fig. 5 all the synchronization operations including initial assignments and changeovers are removed, but for each resource working times are incorporated and utilized as preconditions to any task performed by the resource. As a result, the process described by the P-graph consists of two resource, four operations, an intermediate target and a final target only.

In Fig. 6 elements of Fig. 5 are replaced without changing their relations to be easier to understand. It can be read as: task P1 is completed by either resource R1 or R2 first, and task P2 is finished by either resource R1 or R2 afterward achieving the final target of the process. Working times of both resource R1 and R2 are counted in parallel.

During the manufacturing of a single product, each worker is delegated to work stations at the production line. Therefore, workers can pass products to the next workstation in the forward direction of the line only. Consequently, if resource R1 is assigned to workstation #1 and R2 to workstation #2, then R1 can pass a semi-finished product to R2 but not vice versa. Process structures including operations P1byR1 and P2byR1; or P1byR2 and P2byR2; or P1byR1, pass and P2

Fig. 5 Superstructure focusing on the execution of two consecutive tasks by two potential resources in any order

Fig. 6 Rearranged superstructure of two consecutive tasks executed by two potential resources in any order

could be feasible and automatically generated by algorithms SSG or ABB from the P-graph in Figure 7. Note that the structure including operations P1byR2 and P2byR1 but not P1byR1 cannot be feasible since precondition P1done1 to operation P2byR1 is not satisfied.

Finally, to clarify the aim of line balancing, the maximal working time of any resource is introduced as an input to the process structure, which utilization is to be minimized during the optimization; see Fig. 8. Through operation Max, the maximum working time is defined as a precondition to the availability of each working time for any resources performing any assembly task. The model prototype in Fig. 8 is to be extended for multiple assembly tasks and more employees and serves the bases for the decision support system detailed in the forthcoming paragraphs.

P-graph representation of the work distribution of assembling Product1 given in the Table 1 with 3 worker is shown in Fig. 9. Tasks in the same column are assigned to the same worker while rows contain subtasks to be completed by at least one worker. From left to right the workers are listed, while the subtasks appear from top to down in their required order.

Fig. 7 Rearranged superstructure of two consecutive tasks executed by two potential resources in fixed order

Fig. 8 Rearranged superstructure for minimizing the maximal time needed for any of two resource during performing two consecutive tasks in fixed order

Fig. 9 P-graph representation of the potential steps of the assembly process: maximal structure

3.3 Optimization by P-graph algorithms

Algorithm ABB results 7 s of cycle time for the process synthesis problem introduced in the preceding session, where the first worker completes the steps 1 and 2,

the second worker the step 4, and finally, the third worker completes the last two tasks; see Fig. 10.

Ideally, every worker spends $(\sum_{j=1}^m s_j)/n$ time with assembling, but this is in most cases not feasible, because of the discrete values of times with large variations required by performing subtasks. For Product2, distribution is easy, because every step takes 2 s, so for two people the assembly time is 12–12 s, for three people is 8–8–8, and four people spend 6–6–6–6 s with assembling in the optimal case. Clearly visible is that in an ideal case the number of the cycle time is inversely proportional to the number of workers.

It is worth paying attention to alternative solutions in Table 2. The second best assignment for Product1 with 3 workers is as good as the first one when the second worker does the second subtask as shown in Fig. 11. From the third to the sixth best assignments, the cycle time is 8 s, where multiple subsets of tasks are allocated to practically two of the three workers. Consequently, the optimal allocation for Product1 with two workers is 8 s. For those who are interested in reproducing the results, the P-graph Studio file is available for download at <http://www.p-graph.org/wp-content/uploads/2019/07/Example1Product1ThreeWorkers.zip>.

By increasing the number of workers further, it turns out that 4 workers can complete the process within the cycle time of 5 s, and the cycle time can no longer be reduced by increasing the number of workers because no results better than 5 s are available. It comes from the time of the longest subtask, i.e., the $cycletime \geq \max(s_j)$.

Fig. 10 Optimal task allocation for Problem1

Table 2 10 best assignments for Product1, Product2, Product3 and Product4 with three and four workers

Solution	Cycle time (s)							
	Product1		Product2		Product3		Product4	
	w3	w4	w3	w4	w3	w4	w3	w4
#1	7	5	8	6	20	14	17	12
#2	7	7	10	8	21	15	17	12
#3	8	7	10	8	21	16	17	12
#4	8	7	10	8	21	16	17	13
#5	8	7	10	8	21	17	17	13
#6	8	7	10	8	22	17	17	13
#7	9	7	10	8	23	17	17	13
#8	11	7	10	8	23	17	17	13
#9	12	7	10	8	23	17	18	13
#10	12	7	10	8	23	17	19	13

The large-assembling subtask time problem causes the longer cycle times for Product3 compared to Product4, even if the long time required to perform the 5th step does not limit the cycle time directly.

In daily decision support, graphical modeling is not a must to utilize the power of the P-graph framework as described in Sect. 3.1, but the P-graph model can be saved in text files with .pns extensions, updated, and resolved by P-graph solver. Alternatively, algorithmic generation of the mathematical programming model is offered with the input of the number of workers and the list of the tasks to be executed. In chapter 3.4, such a model generation will be presented.

3.4 MILP formulation of the line balancing problem defined by P-graphs

The line balance can be achieved by solving a linear programming problem where the objective is given by a linear function and the constraints also appear as linear inequalities. As the basics of linear programming go back to the 1940s (Dantzig 1948), therefore, many open source software implementations are now available (Orchard-Hays 1990). However, the input structures of these solvers are often difficult to understand and program. Thus language Zimpl was selected to translate the mathematical model of a problem into a linear or mixed-integer mathematical program expressed in .lp or .mps file format to be read and solved by an LP solver (Koch 2001).

For numbers n of workers and number m of tasks, the cycle time $cycleTime$, the working times $workingTimeOfWorker_i$, $WorkerPerformTask_{i,j}$ and $PassWorkerAfterTask_{i,j}$ are defined as variables for each worker $i \in \{1, 2, \dots, n\}$ and task $j \in \{1, 2, \dots, m\}$ in the mathematical programming model. The cycle time in a nonnegative number:

$$cycleTime \geq 0 \quad (1)$$

as well as the working times of workers:

$$\forall i \in \{1, 2, \dots, n\} : workingTimeOfWorker_i \geq 0 \quad (2)$$

Fig. 11 Second and third best task allocations for Problem1

workerPerformTask is a binary matrix expressing the assignment of workers to tasks by values of 1's:

$$\forall i \in \{1, 2, \dots, n\}, j \in \{1, 2, \dots, m\} : \text{workerPerformTask}_{ij} \in \{0, 1\} \quad (3)$$

while *passWorkerAfterTask* is another binary matrix telling us whether worker *i* passes the semi-finished product to the worker *i* + 1 after finishing task *j* instead of further processing it:

$$\forall i \in \{1, 2, \dots, n\}, j \in \{1, 2, \dots, m\} : \text{passWorkerAfterTask}_{ij} \in \{0, 1\} \quad (4)$$

The objective function is to minimize the cycle time:

$$\text{cycleTime} \rightarrow \text{Min} \quad (5)$$

As a constraints, the following are incorporated. Each subtask must be completed by exactly one worker:

$$\forall j \in \{1, 2, \dots, m\} : \sum_{i=1}^n workerPerformTask_{i,j} = 1 \quad (6)$$

The cycle time cannot be shorter than the working time of any worker:

$$\forall i \in \{1, 2, \dots, n\} : cycleTime \geq workingTimeOfWorker_i \quad (7)$$

For each worker, the working time is to sum of the subtask times performed by the worker:

$$\begin{aligned} \forall i \in \{1, 2, \dots, n\} : workingTimeOfWorker_i \\ = \sum_{j=1}^m workerPerformTask_{i,j} subtaskTime_j \end{aligned} \quad (8)$$

where the $subtaskTime_j$ constants are considered to be know from the problem definition. The first subtask is assumed to done by the first worker:

$$\forall i \in \{1, 2, \dots, n\} : workerPerformTask_{i,1} = 1 \quad (9)$$

If a task is performed by a worker or the product is just received from the previous worker, it is questionable whether the next task is to be completed by the same worker, or the semi-finished product is to be passed further to the next worker:

$$\begin{aligned} \forall i \in \{2, 3, \dots, n\}, \forall j \in \{2, 3, \dots, m\} : workerPerformTask_{i,j-1} + passWorkerAfterTask_{i-1,j-1} \\ = workerPerformTask_{i,j} + passWorkerAfterTask_{i,j-1} \end{aligned} \quad (10)$$

Since the first worker cannot receive a semi finished product from a previous one, it is a special case:

$$\begin{aligned} \forall j \in \{2, 3, \dots, m\} : workerPerformTask_{1,j-1} \\ = workerPerformTask_{1,j} + passWorkerAfterTask_{1,j-1} \end{aligned} \quad (11)$$

Equations 9–11 guarantees, that no tasks can be omitted, and workers can perform consecutive steps only. Note, that the mathematical programming model presented herein is identical to the one exported by software P-graph Studio on the bases of the P-graph model presented in Sect. 3.2.

3.5 Solution of the MILP formulation

After translating the mathematical model given in ZIMPL language in a .zpl file the result will be a file with .lp extension, which can be handled by LP solvers. Such an LP solver is COIN-OR CBC, which is an open-source mixed integer programming solver written in C++ (Forrest and Lougee-Heimer 2005). Of course, it is also an option to omit ZIMPL and directly formulate the model in LP format

in an .lp file, but considering the composition of .lp, it would be much more complicated. For example, the MILP model for determining the optimal assignment for the production of Product4 contains 50 lines in .zpl file and 350 lines in .lp format.

Solving the .lp files by COIN-OR CBC, the optimal task-worker assignment and cycle time, as a result, are displayed in a text file. The results are consistent with those reported by the P-graph software, but only a single optimal assignment can be obtained by CBC, see Fig. 2.

4 Software implementation and feedback

Since production managers are typically not experts in neither process synthesis by P-graphs nor mathematical programming, it is necessary to create software for generating and solving either the P-graph or the mathematical programming models described in Sects. 3.2 and 3.4

The software developed for daily balancing a serial production line was tailored according to the practice of an assembly plant, and visible in Fig. 1. In this decision support software both P-graph model generation connected to the P-graph solver and MILP model generation providing input to a MILP solver has been implemented, user can switch between the two alternative optimization method.

First, the user selects the product to be manufactured and the number of workers considered to be available. Subtask times are stored in a table, which can be specified through the menu. This table is freely expandable. The software loads the table at startup and offers only those products for selection which are included in it. After the generation and solution of the optimization model, the software displays the optimal assignments visually in detail in an easy to read table form for the decision-makers.

The software was developed in consultation with a Hungarian assembly company, so the model was also tested on real data. The company indicated that although the names of the products to be manufactured are identical but depending on the order some assembly steps can be included or excluded on demand, thus the final version of the interface involves options for tailoring the steps relevant for the order of interest. The input received from the company contained 21 different products and measurement data of at most 128 but at least 48 assembly steps.

When using the software, the user has to select the product type and the number of workers first. After that, the list of subtasks can be tailored based on a reference list on the right-hand side of the interface. Clicking on the “Solve problem” button, the software starts both model generation and solution, then retrieves the result and displays it in an Excel file on two worksheets.

The company gave positive feedback on the software, the result speaks for itself, as the company has achieved an average efficiency improvement of 20–25% on all types of products by introducing the optimization software in daily practice; see Fig. 12.

Fig. 12 Part of the feedback: assembly with 13 worker

5 Future work

The model and software could be extended in the future to identify bottlenecks and take into account the possibility of parallelizing long lasting subtasks. It could be realized by setting up work stations in parallel, e.g., at both sides of the production line, where multiple workers could share semi-finished products waiting for the same operation. Such extension could lead to an even better line balance by cutting the length of the longest tasks, and eliminating bottlenecks highlighted.

6 Conclusion

The assembly industry in Central and Eastern Europe is considerably present. However, due to the growing demand and labor shortages, it is necessary to optimize production in as many places as possible. This article presents an exact modeling and optimization approach based on the P-graph framework during model development and prototyping, as well as, the implementation of the resultant formulation by a mathematical programming solver as part of decision support software in a real-life environment. The resultant software has been installed and introduced to the daily practice in a Hungarian assembly plant. According to real life measurements, 20–25% increase in the yields of the shifts has been achieved in comparison with the previous line assignment best practice.

Acknowledgements Open access funding provided by University of Pannonia (PE). Authors acknowledge the financial support of Széchenyi 2020 program under the Project EFOP-3.6.1-16-2016-00015.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution,

and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- Barany M, Bertok B, Kovacs Z, Friedler F, Fan LT (2011) Solving vehicle assignment problems by process-network synthesis to minimize cost and environmental impact of transportation. *Clean Technol Environ Policy* 13(4):637–642. <https://doi.org/10.1007/s10098-011-0348-2>
- Barany M, Bartos A, Bertok B, Horvath A, Horvath A (2018) Software P-graph studio. <http://p-graph.org/>. Accessed 5 Aug 2019
- Bartholdi J (1993) Balancing two-sided assembly lines: a case study. *Int J Prod Res* 31(10):2447–2461
- Bartos A, Bertok B (2019) Parameter tuning for a cooperative parallel implementation of process-network synthesis algorithms. *Cent Eur J Oper Res* 27(2):551–572
- Bertok B, Adonyi R, Friedler F, Fan L (2011) Superstructure approach to batch process scheduling by S-graph representation. *Comput Aided Chem Eng* 29:1105–1109
- Bertok B, Barany M, Friedler F (2012a) Generating and analyzing mathematical programming models of conceptual process design by P-graph software. *Ind Eng Chem Res* 52(1):166–171
- Bertok B, Kalauz K, Sule Z, Friedler F (2012b) Combinatorial algorithm for synthesizing redundant structures to increase reliability of supply chains: application to biodiesel supply. *Ind Eng Chem Res* 52(1):181–186
- Dantzig G (1948) A theorem on linear inequalities. (**unpublished report**)
- Delice Y, Aydogan EK, Özcan U, Ilkay MS (2017) A modified particle swarm optimization algorithm to mixed-model two-sided assembly line balancing. *J Intell Manuf* 28(1):23–36
- Fan L, Kim Y, Yun C, Park SB, Park S, Bertok B, Friedler F (2009) Design of optimal and near-optimal enterprise-wide supply networks for multiple products in the process industry. *Ind Eng Chem Res* 48(4):2003–2008
- Fazlollahtabar H, Hajmohammadi H, Es'haghzadeh A (2011) A heuristic methodology for assembly line balancing considering stochastic time and validity testing. *Int J Adv Manuf Technol* 52(1–4):311–320. <https://doi.org/10.1007/s00170-010-2708-1>
- Forrest J, Lougee-Heimer R (2005) CBC user guide. *Emerg Theory Methods Appl*. <https://doi.org/10.1287/educ.1053.0020>
- Friedler F, Tarjan K, Huang Y, Fan L (1992a) Combinatorial algorithms for process synthesis. *Comput Chem Eng* 16:S313–S320. [https://doi.org/10.1016/S0098-1354\(09\)80037-9](https://doi.org/10.1016/S0098-1354(09)80037-9)
- Friedler F, Tarjan K, Huang Y, Fan L (1992b) Graph-theoretic approach to process synthesis: axioms and theorems. *Chem Eng Sci* 47(8):1973–1988. [https://doi.org/10.1016/0009-2509\(92\)80315-4](https://doi.org/10.1016/0009-2509(92)80315-4)
- Friedler F, Tarjan K, Huang Y, Fan L (1993) Graph-theoretic approach to process synthesis: polynomial algorithm for maximal structure generation. *Comput Chem Eng* 17(9):929–942. [https://doi.org/10.1016/0098-1354\(93\)80074-W](https://doi.org/10.1016/0098-1354(93)80074-W)
- Friedler F, Varga J, Fan L (1995) Decision-mapping: a tool for consistent and complete decisions in process synthesis. *Chem Eng Sci* 50(11):1755–1768
- Friedler F, Varga JB, Fehér E, Fan LT (1996) Combinatorially accelerated branch-and-bound method for solving the mip model of process network synthesis. In: Floudas CA, Pardalos PM (eds) *State of the art in global optimization. Nonconvex optimization and its applications*, vol 7. Springer, pp 609–626
- Frits M, Bertok B (2014) Process scheduling by synthesizing time constrained process-networks. *Comput Aided Chem Eng* 33:1345–1350. <https://doi.org/10.1016/B978-0-444-63455-9.50059-3>
- Garcia-Ojeda J, Bertok B, Friedler F (2012) Planning evacuation routes with the P-graph framework. *Chem Eng Trans*. <https://doi.org/10.3303/CET1229256>
- Heckl I, Halasz L, Szlama A, Cabezas H, Friedler F (2015) Process synthesis involving multi-period operations by the P-graph framework. *Comput Chem Eng* 83:157–164
- Kalauz K, Sule Z, Bertok B, Friedler F, Fan L (2012) Extending process-network synthesis algorithms with time bounds for supply network design. *Chem Eng Trans* 29:259–264. <https://doi.org/10.3303/CET1229044>

- Kettl KH, Niemetz N, Sandor N, Eder M, Heckl I, Narodoslawsky M (2011) Regional optimizer (regi-opt)—sustainable energy technology network solutions for regions. *Comput Aided Chem Eng* 29:1959–1963
- Koch T (2001) ZIMPL. Citeseer
- Kumar N, Mahto D (2013) Assembly line balancing: a review of developments and trends in approach to industrial application. *Glob J Res Eng* 13(2)
- Orchard-Hays W (1990) History of the development of LP solvers. *Interfaces* 20(4):61–73. <https://doi.org/10.1287/inte.20.4.61>
- Otto A, Battaia O (2017) Reducing physical ergonomic risks at assembly lines by line balancing and job rotation: a survey. *Comput Ind Eng* 111:467–480
- Özceylan E, Kalayci CB, Güngör A, Gupta SM (2018) Disassembly line balancing problem: a review of the state of the art and future directions. *Int J Prod Res*. <https://doi.org/10.1080/00207543.2018.1428775>
- Ren Y, Yu D, Zhang C, Tian G, Meng L, Zhou X (2017) An improved gravitational search algorithm for profit-oriented partial disassembly line balancing problem. *Int J Prod Res* 55(24):7302–7316
- Roberts SD, Villa CD (1970) On a multiproduct assembly line-balancing problem. *AIIE Trans* 2(4):361–364
- Salveson M (1955) The assembly line balancing problem. *J Ind Eng* 6:18–25
- Sivasankaran P, Shahabudeen P (2014) Literature review of assembly line balancing problems. *Int J Adv Manuf Technol* 73(9–12):1665–1694. <https://doi.org/10.1007/s00170-014-5944-y>
- Sule Z, Bertok B, Fan FFL (2011) Optimal design of supply chains by P-graph framework under uncertainties. *Chem Eng* 25:453–458
- Tan RR, Aviso KB (2016) An extended P-graph approach to process network synthesis for multi-period operations. *Comput Chem Eng* 85:40–42
- Thomopoulos NT (1967) Line balancing-sequencing for mixed-model assembly. *Manag Sci* 14(2):B-59
- Tick J, Kovacs Z, Friedler F (2006) Synthesis of optimal workflow structure. *J UCS* 12(9):1385–1392
- Tick J, Imreh C, Kovács Z (2013) Business process modeling and the robust PNS problem. *Acta Polytech Hung* 10(6):193–204
- Vance L, Cabezas H, Heckl I, Bertok B, Friedler F (2012) Synthesis of sustainable energy supply chain by the P-graph framework. *Ind Eng Chem Res* 52(1):266–274
- Wilson J (2014) Henry ford vs. assembly line balancing. *Int J Prod Res* 52(3):757–765. <https://doi.org/10.1080/00207543.2013.836616>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.