23rd International Conference on Modeling & Applied Simulation, 013
21st International Multidisciplinary Modeling & Simulation Multiconference

2724-0037 © 2024 The Authors.
doi: 10.46354/i3m.2024.mas.013

Aggregated evaluation of production flows in SMEs production systems

Ivo Formánek[1], David Kruml[2,*] and Jan Paseka[2]

[1]Department of Entrepreneurship and Management, Pan-European University, Spálená 76/14, Prague, 110 00, Czech Republic
[2]Department of Mathematics and Statistics, Masaryk University, Kotlářská 2, Brno, 611 37, Czech Republic

*Corresponding author. Email address: kruml@math.muni.cz

Abstract

This paper proposes a practical approach for developing production planning tools in Small and Medium-sized Enterprises (SMEs). We focus on modeling material flow (products and intermediates) to keep things simple for production workers who will use the tool. Since these workers may not have specialized modeling knowledge, we represent the production flow as a network of processes, containers (objects), and connecting transport routes. Arrows on the connections show flow direction. This structure resembles a graph or Petri net, which we leverage for our development. We illustrate that every production flow failure or waste can be quantified as either an underflow or an overflow within an appropriate stack. To achieve this, we introduce the concept of a defect associated with a planning scheme. Specifically, the defect of a composed scheme is expressed as the sum of particular defects. This approach allows us to measure and address inefficiencies in the production process effectively.

Keywords: Production flow; process; stack; defect; aggregated constraints

1. Introduction

Simply said, Economics, at its core, revolves around the flow of goods, services, energy, information, money etc. within a system. To understand these complex interactions, economists utilize various flow models. These models simplify the economic landscape by visualizing the interconnected exchanges between different sectors, such as:

- Households: Consumers who spend their income on goods and services.
- Firms: Businesses that produce goods and services and employ workers.
- Government: The public sector that collects taxes and provides public goods.
- Financial Institutions: Banks and other institutions that facilitate financial transactions.

Flow models offer a powerful tool for analyzing:

- Circular Flow: The basic flow of money and resources between households and firms, illustrating how spending generates income and vice versa. (e.g., Al-Fedaghi (2008), Samuelson (1961), Mankiw (2021))
- Leakages and Injections: Factors that disrupt the circular flow, such as savings (leakage) or government spending (injection). (e.g., Krugman and Obstfeld (2020), Blanchard and Johnson (2021))
- Economic Impacts: How changes in one sector (e.g., increased government spending) can ripple through the entire economy. (e.g., Bodie et al. (2021), Auerbach

and Kotlikoff (2018))

Flow modelling of production plays a crucial role in optimizing and analyzing production processes within various industries. It involves creating a digital representation of the flow of materials, products, and information throughout the production system. This allows manufacturers to:

- Visualize the production process: Flow models map out the various stages of production, from raw materials entering the system to finished goods exiting. This visual representation helps identify bottlenecks, inefficiencies, and potential areas for improvement. (e.g., Bodie et al. (2021))
- Analyze production performance: By simulating the flow of materials and products, manufacturers can gain insights into production time, resource utilization, and overall efficiency. This helps identify areas where production can be streamlined or bottlenecks can be eliminated. (e.g., Smith and Johnson (2019))
- Optimize production planning: Flow models can be used to test different production scenarios and optimize production schedules. This allows manufacturers to make informed decisions about resource allocation, inventory levels, and production capacity. (e.g., Lee and Chen (2018))
- Identify potential problems: By simulating potential disruptions or changes in demand, flow models can help manufacturers identify potential problems before they occur. This proactive approach allows for better preparedness and mitigation strategies. (e.g., Wang and Liu (2020))

Different types of flow models exist, ranging from simple diagrams to complex simulations. Each model offers varying levels of detail and analysis capabilities, depending on the specific needs and goals of the production system. (e.g., Gunal (2019))

Our focus is on modeling production flows specifically for SMEs. We're not interested in complex models for large manufacturers. For SMEs, a simplified approach that breaks down production into general processes and stacks (objects) connected by transport paths (arrows) is often beneficial. Arrows on these connections indicate the flow direction. This simplified structure can be effectively represented as a graph or a Petri net, a mathematical tool for modeling concurrent systems (e.g., Tuncel and Bayhan (2007)).

The production systems of SMEs are usually characterized by the following characteristics. Let's list the main ones.

- **Customization Reigns Supreme:** Unlike large manufacturers, SMEs often specialize in producing smaller quantities of highly varied products. These products can be individual, unique pieces or made-to-order.
- **Low Repeatability:** The exact production process for a specific item might not be repeated often, or the repetition might be irregular.
- **Flexibility is Key:** Production systems in SMEs need to be adaptable to handle frequent adjustments in machines and lines. This necessitates careful and long-term storage of actual machine settings.

While Petri nets are our focus for modeling production flows, our software application offers a wider range of functionalities. This includes powerful tools for optimizing and preserving production parameters over the long term. These parameters encompass production programs, routes, machines, and technological settings crucial for Total Quality Management (TQM). Our application leverages big data processing to achieve these functionalities.

In this paper, we propose a general method for calculating the imperfection of a production plan as a weighted sum of specific defects. This method yields a ***quality measure across all possible plans***, which is essential for formulating and solving any optimization problem. We have adopted this method as a core function within our developed planning software. However, optimizing a plan remains a complex task, and we are currently exploring solutions using simulated annealing (e.g., Laarhoven and Aarts (1987)).

The article is organized as follows: In the remaining part of Section 1, we describe a production process where various resources (machines, workers) come together to work on a product. These resources form a temporary "composite state" that dissolves upon job completion. The process alters the state of the flow, which can involve working on the product, transporting materials, or preparing the machine.

Section 2 introduces a method for measuring and evaluating imperfections (defects) in production plans. Defects are viewed as deviations from an optimal flow of materials in the production process. These deviations are measured as underflows or overflows in "stacks" representing resource availability.

Section 3 describes illustrative examples of buffer stacks, of evaluating on-time delivery performance, particularly for products with critical deadlines where both early and late arrivals are undesirable, of scaled penalties in production environments, of aggregated constraints and synergy (both positive and negative), and of the trade-off between flexibility and optimality in production planning strategies.

Finally, Section 4 concludes the paper.

1.1. Processes

The process *alters the state* of the flow, which can involve working on a piece, transporting materials, preparing a machine, etc. Multiple sources can join the process simultaneously. Typically, a product enters the process along with a machine and a worker. These elements form a temporary *"composite state"* that dissolves upon completion of the job. While the product improves, the machine may

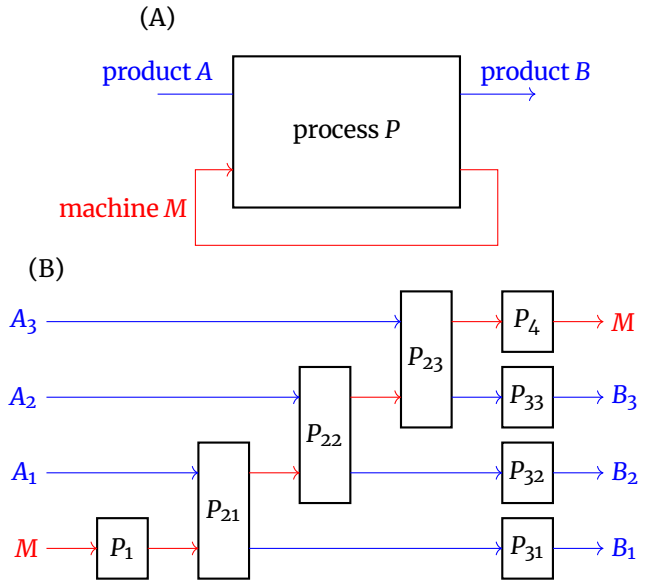

Figure 1. (A) A simplified scheme of iterated process P displays the process as a box with input stream of products A, output stream of (worked) products B, and a loop of "recycling" machine M. (B) An extended scheme of the same situation decomposes the process P to particular subprocesses. Here, the decomposition even splits the main working process to its iterations P_{21}, P_{22}, P_{23}. Since machine M is ready to accept a next item sooner than the previous one is finished (typical behaviour of line production), the difference between process time and cycle time is modeled by subprocesses P_{31}, P_{33}, P_{33}. The path of machine M (red) starts with preparation subprocessP_1 (warming up, adjustment) and finishing subprocess P_4 (cleaning, maintenance). Thus working each item $A_i \rightarrow B_i$ takes process time given by path$P_{2i} + P_{3i}$, and the machine M is occupied for time given by path $P_1 + P_{21} + P_{22} + P_{23} + P_4$.

experience minor degradation (dulling, overheating, etc.).

In most cases, the process consumes time. For regular production, this delay is assumed to be constant or have minimal randomness. Therefore, the time consumption can be represented by either a single value or a random variable. The key parameters are:

- ***Process Time:*** This expresses the time a product spends within the process.
- ***Cycle Time:*** This expresses the time between the entry of two consecutive products.
- ***Changeover Time:*** This expresses the time needed to reconfigure the machine for a different production run.

Since sources can enter and leave the process independently (forming composite states gradually), we can measure delays between *specific events*. This allows us to decompose the process into a network of simpler *"subprocesses"* with predictable behavior (see Figure 1). This approach resembles tracing partitions using Feynman diagrams.

1.2. Stacks

Unlike processes that transform items, stacks simply store them for an indefinite period. In simpler terms, items

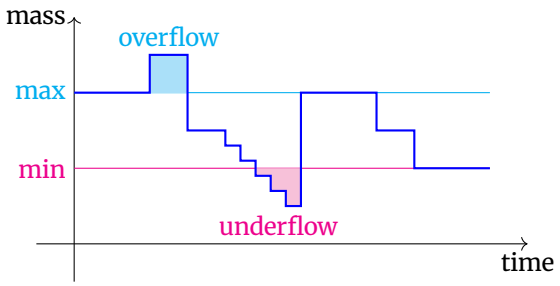

Figure 2. Overflow and underflow

within a stack only *"move in time"* without changing their position along their production path.

In production settings, stacks serve as strategic storage points for materials, finished products, or buffers to mitigate production fluctuations. Their capacity is defined by minimum and maximum stock levels.

In our previous work (Kruml and Paseka (2018)), we proposed an algorithm to quickly determine if a production plan is feasible within the constraints of a specific stack capacity. An overflow condition indicates a blockage on input streams, while an underflow indicates starvation on output streams. The algorithm can also be adapted to calculate the size of the error associated with exceeding capacity (see Figure 2).

Beyond physical storage, stacks offer a powerful tool for representing various types of *constraints* within a production system.

For example, a shipping deadline can be modeled as a process triggered on a specific date that simply retrieves ordered goods. The goods are either prepared or not, and this state is tracked using a dedicated boolean stack. If the order is ready, the stack value becomes 1. The shipping process consumes this value, resulting in a decrease to 0. If the shipping process starts prematurely (when the stack value is 0), the value becomes -1, indicating an error outside the acceptable range of 0 and 1 (the stack is underflowed).

Another example involves detecting potential job collisions on a machine. A stack representing the machine's readiness can be created. The machine is pushed onto the stack after finishing a job (and any necessary procedures). If a job attempts to utilize the machine while it's not ready, the readiness stack would acquire a negative value, similar to the shipping deadline example.

These examples demonstrate how stacks can effectively model various constraints within production systems, providing valuable insights into potential bottlenecks and resource limitations.

1.3. Petri nets

Both processes and stacks admit *aggregation*, i. e. they can be composed to complex processes or complex stacks. Then two consequent processes can be either composed

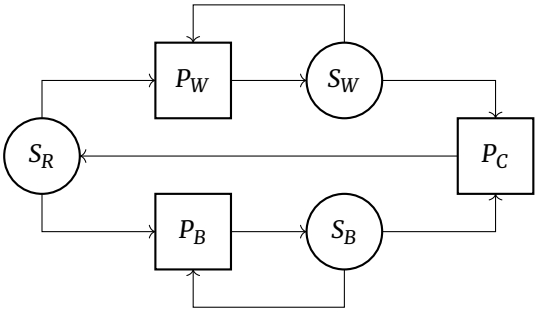

Figure 3. The scheme represents working cycles of a brush. In the beginning it is at state S_R "ready" and can be used either for white painting P_W or black painting P_C. After that, the brush is in one of states S_W "dirty white" or S_C "dirty black" and can be used again for the same painting P_W or P_C. After finishing all jobs in one colour, the brush is cleaned in process P_C and returned to the "ready" state S_R. Since stacks and processes alternate, one can consider the scheme as a Petri net. But in reality, the cleaning process P_C is triggered only by one of the inputs. We fix the problem by splitting P_C to two subprocesses P_{CW} "cleaning white brush" and P_{CB} "cleaning black brush".

or separated by a stack as well as two stacks can be united or separated by a process (e. g. transport). This results to a diagram where processes and stacks alternate making the graph *bipartite*.

Flows in such graphs can be effectively modeled by *Petri nets* or *timed Petri nets*. Let us recall that a process of Petri net is *fired* if all input stacks are non-empty. After firing the process stock of every input stack is reduced by one token and stock of every output stack is increased by one token.

In this way, Petri nets allows to model more complex processes, e. g. assembly of more components in manufacturing or making temporary composite states as discussed earlier (see Tuncel and Bayhan (2007)). Even controled processes can be simulated as Petri nets with an extra information input. The source is *boolean*. It is filled by a single token if the process is approved or empty if the process is disapproved.

On the other hand, the mechanism of firing processes might be too restrictive in situations, in which not all input sources are needed. The problem can be solved by a formal refinement of stacks or processes which can be less clear (see Figure 3). However, we assume that every production network could be *in principle* modelled by a Petri net.

2. Defect

Our intention is to demonstrate that every *failure* or *waste* of the flow can be measured as an underflow or overflow of an appropriate stack.

We define a *defect* as a product of *size* and *weight* of such deviation from optimality. The size is an area of overlaps of the stock quantity curve. The weight is defined by the planner and should express seriousness of the defect. Values of weight can differ for overflow and for underflow (see Figure 4).

Formally, the defect is given by formula

$$d = w \cdot \int_{t_0}^{t_1} |f(t) - m| dt$$

where w is weight, m is a bound of optimal stock, f the stock function, and $[t_0, t_1]$ the interval on which stock is out of optimal range.

Here's a reformulated version of the text explaining breakdown structures in manufacturing:

In manufacturing, we use a concept called *breakdown structures* to organize complex processes. These structures break down entire processes into smaller, more manageable sub-processes. Just like you can zoom in on a map to see more details, breakdown structures allow us to see the individual steps that make up a larger production process.

This breakdown can be reversed as well. By combining sub-processes, we can recreate the whole process. This works similarly to how building blocks can be assembled to form a larger structure.

An important assumption of breakdown structures is that any constraints or limitations affecting the sub-processes will also apply to the larger process they are part of. Likewise, the defects (or imperfections) of the sub-processes are expected to contribute to the overall defect of the final product. There might be some exceptions to this rule, but we'll explore those in a later section (section 3.4).

Our concept of defect offers several advantages for evaluating production plans:

- **Simple Calculations:** Defects are additive. This means the overall health of a plan is simply the sum of the defects in its individual components. This makes it easy to understand how different parts of the plan contribute to the overall outcome.
- **Independent Evaluation**: Independent parts of a production plan can be assessed separately. This allows planners to focus on specific areas without getting

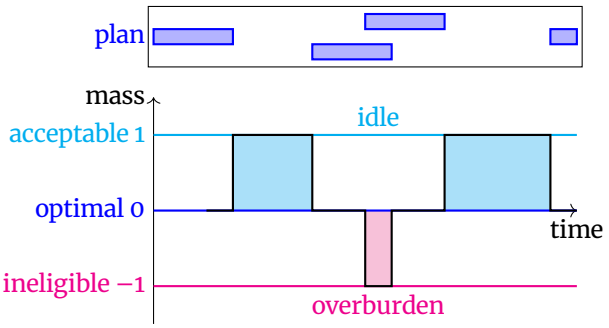

Figure 4. Blue rectangles are jobs planned for a machine. Activity of the machine is indicated on stack of its readiness. If there is no job then the machine is in an idle state. If there are two overlapping jobs then there is "negative number of ready machines". Both cases are not optimal. Size of the defects is calculated from the area of cyan or magenta rectangles. The overburden is clearly worse and should be penalized with a higher weight.

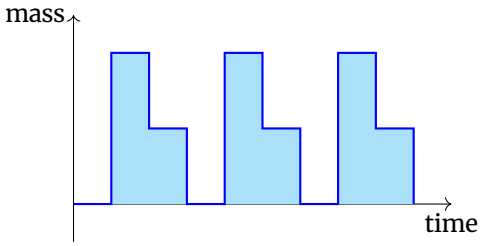

Figure 5. The blue stock curve on a buffer stack is created by two consequent processes. Here the input (feeding) process with three times larger working batch than the output (consuming) process.

bogged down in complex interactions.

- **Efficient Updates:** When one makes localized changes to a plan, only the defects in nearby stacks and within a specific timeframe are affected. This means one can efficiently recalculate the overall defect without having to re-evaluate the entire plan from scratch.

3. Examples

3.1. Buffer stack for different batches

Buffer stacks become essential when consecutive production processes handle different batch sizes (as illustrated in Figure 5). These stacks act as a buffer zone, temporarily storing items between processes to ensure smooth production flow.

While items sit in the buffer, they are technically idle, which isn't ideal. However, this idling is unavoidable due to the difference in batch sizes. We can consider this a "natural" defect, meaning it's not something that can be optimized away without completely restructuring the production line.

It's important to remember that even though this idling is unavoidable, the buffer stack still introduces a delay in the overall lead time (the time it takes to complete an order). Additionally, the stored items contribute to the total production defect, even if it's not due to errors. Therefore, when evaluating production plans, we should assign a non-zero weight to account for the impact of buffer stacks.

3.2. Order in time

For deadlines that are critical, like delivering a wedding cake, both early and late deliveries are undesirable. The cake should be fresh, but arriving too soon might mean it sits out for too long.

To account for this, we can consider a delivery performance measure with three possible values: -1 (missed deadline, the worst outcome), 0 (delivered on time, the ideal scenario), and 1 (delivered too earlyl, which can also be negative depending on the product). The situation resembles one from Figure 4.

We can assign different penalty weights to each of these values. For instance, a late wedding cake (penalty -1) would likely be considered much worse than an early one (penalty 1).

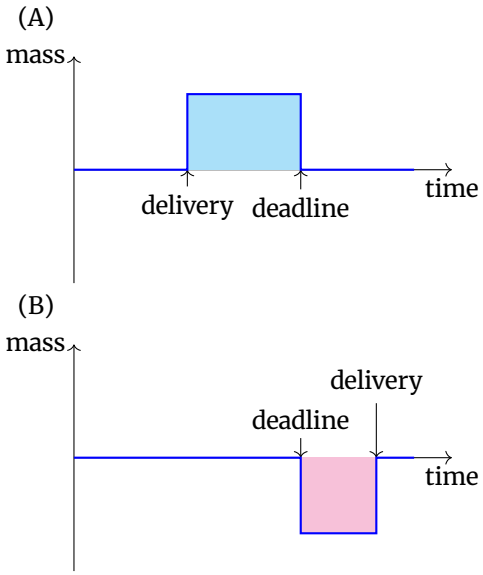

Figure 6. Sooner delivery creates an interval of "idle" state (A). Later delivery creates an interval of "overburden" state (B).

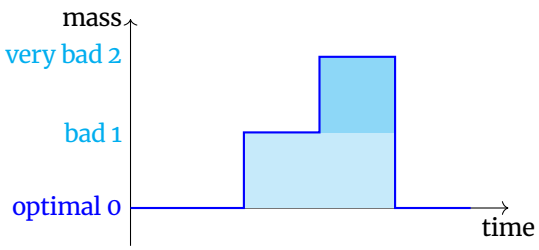

Figure 7. An extra bound (level) expresses an area in which a defect is weighted with a higher penalty.

3.3. Scaled penalties

In real-world factories (industrial practice), penalties for production issues often increase more severely as the severity of the issue grows. A small delay might have a minor impact, but a large delay can be much more disruptive.

We can capture this idea by creating different zones or levels for evaluating these issues. Each zone would have a higher weight assigned to it, reflecting the greater consequences of larger deviations from the ideal (see Figure 7).

Different weights can be also defined for time axis. An example could be "extended deadline" — missing regular deadline is "bad", but missing extended deadline would be "very bad".

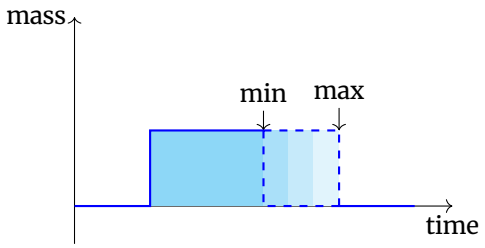

Figure 8. For jobs with variable processing times, the likelihood of a defect occurring depends on when the job finishes. Jobs that haven't finished by the minimum allowed time are certain to be incomplete, resulting in the highest defect value (multiplied by 1). As time progresses, the chance of the job finishing increases. This means the defect value gradually decreases. By the maximum allowed time, the job is guaranteed to be finished, and so the defect value reaches zero. "Density" of the defect in the middle area is expressed by cumulative distribution function of the probability distribution.

3.4. Aggregated constrains

Simple addition doesn't always capture the full picture. Imagine an order: it's not valuable until all the pieces are complete. Similarly, a single right shoe or a single left shoe isn't very useful. These examples show that just summing the value of individual parts doesn't reflect the true worth of the whole.

To account for these interactions between components, we can introduce the concept of synergy. Synergy happens when the combined value of two or more things is greater than the sum of their individual values. We can imagine representing these synergies with virtual stacks, like counters that keep track of specific combinations.

The importance of each combination can be reflected by assigning a weight to its virtual stack. This weight indicates how much that particular combination contributes to the overall value.

Synergy can be positive or negative. For example, assembling a complete product creates positive synergy. On the other hand, multiple machine failures happening at the same time (critical coincidence of accidents) would be a negative synergy.

3.5. Buffer stack for probabilistic variation

Buffer stacks act as a buffer zone in production lines, helping to smooth out inconsistencies. This is particularly useful when there's unpredictable behavior at the beginning or end of the production process, such as variations in processing time by machines.

Imagine these variations in processing time as different weights. The likelihood of each variation (represented by a probability distribution and illustrated in Figure 8) can be seen as another factor influencing the overall impact on production (similar to how a heavier weight would cause a bigger disruption).

3.6. Flexibility vs. optimality

More complex example concerns comparison of planning strategies. One strategy prioritizes minimizing production times, regardless of cost. This approach keeps inventory of in-progress work pieces very low, but it might come at the expense of underutilizing machines.

On the other hand, some strategies aim for smoother production with larger buffers of partially completed work. This ensures better machine utilization but can lead to a build-up of inventory, which can be expensive to store and manage.

The strategy focused on short production times might lead to more defects accumulating on the machines themselves, due to potential strain or rushing the process. Conversely, the strategy prioritizing full machine utilization might create more defects in the raw materials or partially finished products that pile up in storage.

We can evaluate a production plan by considering two key metrics: the total number of machine defects and the total number of material defects. The ratio between these values indicates which strategy, short production times or high machine utilization, might be more favorable for a particular scenario.

Therefore, an ideal production plan wouldn't necessarily prioritize one extreme over the other. Instead, it would likely strike a balance.

4. Conclusion

We have proposed a method to evaluate production plans. This method assigns a score to each plan, reflecting the sum of its partial defects. These partial defects represent any undesirable state the production process might encounter.

In this paper, we have demonstrated a wide range of examples showcasing how production constraints can be implemented and their associated defects calculated. These defects encompass any production source, effectively covering all relevant partial measures of plan quality. The weighted sum of these defects can then be interpreted as a measure of the overall ***production plan inefficiency***.

The method can be considered as a sort of multiple-criteria decision and enables to define preference ordering on a space of possible plans. The enumeration of quality of the plan provides an essential function for optimization.

This research builds upon our previous work Kruml and Paseka (2018) on *plan validation*. The key innovation here is that we go beyond simply identifying production issues (overflows and underflows). We also quantify their severity. This allows us to view these issues not as dealbreakers, but as factors that can be penalized based on their relative importance, as determined by the planner.

5. Funding

The research was funded by the Technology Agency of the Czech Republic, project number TAČR FW03010296 "Practical use of big data for an intelligent production flow decision system," principal investigator Jan Štindl (data-Partner s.r.o). This funding source financially supported all authors.

References

Al-Fedaghi, S. (2008). Flow-based modeling of economic. In *Proceedings of the World Congress on Engineering and Computer Science 2008*, WCECS 2008, San Francisco, USA. World Congress on Engineering and Computer Science.

Auerbach, A. J. and Kotlikoff, L. J. (2018). *Dynamic Public Finance*. Cambridge University Press.

Blanchard, O. J. and Johnson, D. S. (2021). *Macroeconomics*. Pearson.

Bodie, Z., Kane, A., and Marcus, A. J. (2021). *Investments*. McGraw-Hill Education, 12th edition.

Gunal, M. M., editor (2019). *Simulation for Industry 4.0: Past, Present, and Future*. Springer Series in Advanced Manufacturing. Springer International Publishing.

Krugman, P. R. and Obstfeld, M. (2020). *International Economics: Theory and Policy*. Pearson.

Kruml, D. and Paseka, J. (2018). Categoric simulation of production flows. In *Proc. of the 17th International Conference on Modeling and Applied Simulation*, pages 68–75. CAL-TEK SRL.

Laarhoven, P. J. M. and Aarts, E. H. L. (1987). *Simulated Annealing: Theory and Applications*. Springer Dordrecht.

Lee, C.-H. and Chen, H. (2018). A decision support system for production planning based on flow modeling. *International Journal of Production Economics*, 172:120–135.

Mankiw, N. G. (2021). *Principles of Economics*. Cengage Learning.

Samuelson, P. A. (1961). *Economics: An Introductory Analysis*. McGraw-Hill.

Smith, J. and Johnson, R. (2019). Optimizing production processes using flow models. *Journal of Manufacturing Science*, 45(3):210–225.

Tuncel, G. and Bayhan, M. (2007). Applications of Petri nets in production scheduling: a review. *International Journal of Advanced Manufacturing Technology*, 34(7-8):762–773.

Wang, L. and Liu, Y. (2020). Predictive maintenance using flow models: A case study in automotive manufacturing. *IEEE Transactions on Industrial Informatics*, 16(5):3200–3210.