Simulations in Decision Making

Bojan Rupnik
University of Maribor, Faculty of Logistics, Celje, Slovenia
bojan.rupnik@um.si

DOI
https://doi.org/10.18690/um.fl.2.2026.5

ISBN
978-961-299-074-9

Keywords:
simulations, discrete event simulation, digital twin, material flow, queuing systems

Logistical, production, transportation, and all related issues in the industry follow similar processes, with time being the crucial factor. While some processes can be relatively easily analysed due to their simplicity, the more interconnected the processes are, the more challenging it becomes to describe them accurately using traditional analytical approaches. Simulations, in this regard, provide a deeper insight into the flow of such processes. They enable the analysis of efficiency, shortcomings, and, most importantly, allow for the examination of existing systems under different conditions without interfering with their operation. Besides having a good understanding of the processes, data support is crucial for simulation. This support can involve the recording of historical data and predicting future events with possible alternative scenarios. By enabling real-time data logging during process execution and providing the data to an active simulation that processes it in real-time, a digital twin can be created. Within the scope of this subject, participants familiarize themselves with server systems, queuing systems, discrete event simulations, and the tools that support them, along with examples of their application in manufacturing, logistics, and transportation scenarios.

University of Maribor Press

1 Introduction

With simulations, we try to map real-world events into a mathematical or computer model, with which we can repeat these events, change them, and observe how they behave under different conditions. Most areas of the business world can be analyzed with simulations, such as material flows in production plants or warehouses, simulations of transport flows, information or financial flows. Simulations are therefore used not only for the analysis of such systems, but also for optimization - especially when systems are too computationally complex to be optimized in a timely manner using classical optimization methods.

Depending on the type of problems we are solving and the purpose of optimization, there are several different simulation approaches:

- 3D/real-time simulations (e.g. pilot training simulations),
- system dynamics (simulations of complex, comprehensive systems),
- agent simulation (observing people, entities interacting in space and time),
- discrete event simulation.

The latter approach is at the forefront of this work. Discrete event simulations allow for the description of any systems where individual events influence the further behavior of the events. The method itself is fundamentally simple. The entire system is designed with states that are changed only by events at predetermined times. Unlike continuous simulations, the state is always unchanged between individual events, regardless of the elapsed time. Events can be defined in advance (e.g., expected customer arrivals), or they can generate new events themselves.

2 Process modeling and simulation

Regardless of the field, all processes include a time component. Thus, based on behavior, we observe what is happening in a particular system and how long something takes. Here we can consider input flows, such as customer arrivals to the store, duration of purchase, waiting in front of the cash register. This example can be mapped to many others, where we talk about inputs, processing and finally output from the system.

2.1 Server Systems and Queues

A basic example (Figure 5.1) of a server system (Thomopoulos, 2012) includes a queue in which entities wait for processing and a server that processes these tasks, and the operation of the system depends on the server's processing capacity, the intensity of task arrivals and the capacity of the queue. Depending on the nature of the simulation, entities can represent tasks, packages, customers, information, workpieces or practically any element that affects the events within the simulation.

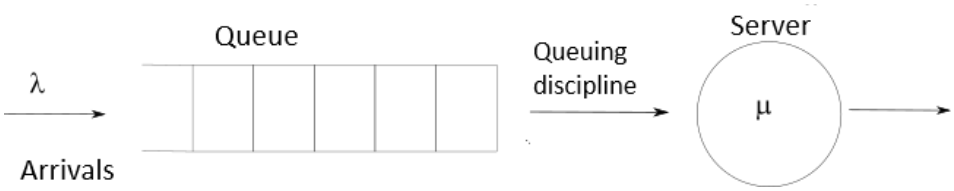

Figure 5.1: Basic server system with queue
Source: own.

The arrival rate determines how often entities arrive in the queue. In general, the arrival rate can be given as:

$$\lambda = \frac{N}{T} \tag{1}$$

where λ is the intensity, N the number of arrivals and T the time interval of arrivals.

According to the process, each entity is placed in a queue from which it is forwarded to the server, if it is available. In the case of multiple waiting entities, the selection of the next one to be forwarded can be done using different approaches:

- FIFO (First-in, First-out) approach, where each task is submitted to the server in the order in which it arrives.
- LIFO (Last-in, Fast-out) approach, where the last task to enter the queue is submitted to the server first.
- Priority queues allow for priority treatment to be set for certain tasks or groups of tasks. Thus, entities with higher priorities are submitted to the server before those with lower ones.

- Random approach determines a random entity in the queue.

Depending on the intensity of arrivals and the availability of the server, the entities in the queue can accumulate, decrease or wait in the queue for an average uniform amount of time. In system modeling, the latter variant is usually sought, as it allows for stable systems.

In addition to the intensity of arrivals, a key factor is also the service rate μ, which is given by the number of entities that the server can process per unit of time. The service rate is thus given as the reciprocal of the service time.

$$\mu = \frac{1}{S} \tag{2}$$

S represents the service time. Like arrivals, service time can also be subject to randomness. Thus, we distinguish service speeds into:

- deterministic,
- stochastic.

In some cases, the service time is constant and known in advance, while in others it depends on factors and is random. The modeling of service times is usually appropriate for exponential or normal distributions, depending on the type of process.

The presented model (Figure 5.2) allows for the simulation of a very basic process with one queue and one server. However, imitating real-world cases requires the construction of more complex networks, where each building block can have multiple inputs and outputs. Depending on the complexity of the case we want to model, complex models can be created where the flow is influenced not only by the connections between the building blocks, but also by the rules for sorting by individual, conditionally determined arrivals and by the serving rules. It is therefore sensible to model and simulate such cases in appropriate dedicated tools.

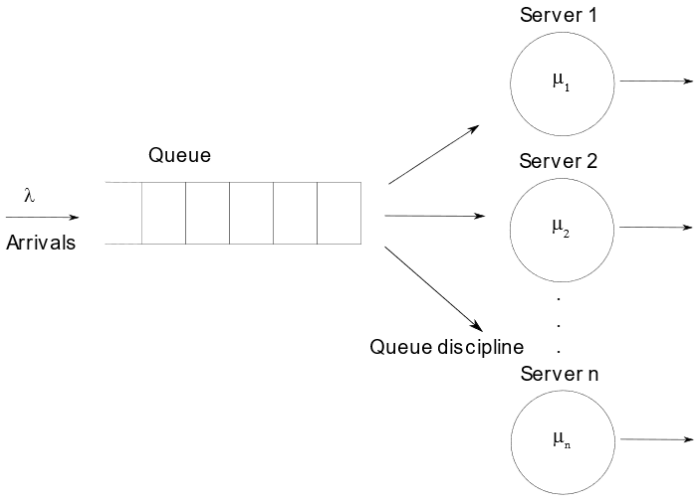

Figure 5.2: Server system with multiple servers
Source: own.

Kendall's notation is used to describe the main characteristics of queueing systems (Bolch et al., 2006). The basic notation is in the following form:

$$A/B/c/K/m/Q \tag{3}$$

where represents:

A – arrival time distribution,
B – service time distribution,
c – number of servers,
K – queue capacity,
m – population size,
Q – service strategy.

The values thus taken by components A and B are:

- M – exponential distribution,
- D – deterministic distribution,
- E – Erlang distribution,
- G – general distribution.

The number of servers c p specifies how many servers can be used to perform parallel services.

Capacity K determines the maximum number of clients in the system, both in the queues and on the server, while m represents the expected number of clients. The previously mentioned server strategies (FIFO, LIFO) determine how entities are delivered from the queues.

2.2 Discrete event simulation

While queueing systems are primarily an abstract representation, a more advanced approach is needed to model more complex systems. Discrete event simulations (Fishman, 2002) are one of the most widely used approaches, alongside e.g. system dynamics or agent simulation. They are commonly used in simulating problems in manufacturing, healthcare, transportation and logistics, energy systems, supply chains, and related fields.

Queuing systems assume a straightforward flow between arrivals and processing. In discrete event modeling, in addition to the entities, queues, and servers themselves, characteristics, rules, resources, and events are also considered. As part of the simulation, a list of all events and their expected time are built based on the model. An event represents any change in the system, such as a customer entering a queue, the start or end of processing a product on a machine, or a change in the properties of an entity. Each event changes the state of the simulated system.

The simulation is performed in simulation time, which does not run in real time, but discretely skips the times between individual events. Individual events can also create new events, which are also placed in the list of future events, which can cause some already planned events to be postponed.

During the simulation itself, statistics on queues, server utilization, throughput and other parameters are recorded, which can be used to provide an appropriate analysis of simulated systems.

Two more key elements in the implementation of simulations are verification and validation. Verification checks the correctness of individual implemented functionalities, calculation formulas, logic. Validation checks how well the model imitates the real system. For this purpose, the simulation results are compared with the expected behavior of the real system, which is obtained from measurements or expert assessment. Verification and validation are repetitive processes that lead to greater accuracy and reliability of the created model. Sensitivity analysis can also be used to assess areas of uncertainty.

2.3 Modeling simulation parameters

The methods of modeling the input parameters of the simulation depend on the type of simulated system and the available data. In this, a good understanding of the processes based on which it is possible to model the material flow is required in the first phase. Thus, it is necessary to identify all the factors (processes or parameters) that can affect the behavior of the system, such as:

Entities and their properties – what are the key elements of the simulation, how can their properties affect the material flow (entities with different properties have different flows through the network, for example).

Simulation objects – any building blocks of the simulation tool that affect the state of the system – sources, sinks, servers or processors, queues, objects for combining or uncombining entities, objects for changing entity properties, event generators.

Material flow – connections between all objects from or to which entities can move. In this case, it is necessary to carefully determine the conditions for redirection from individual objects to successors.

Input intensities – the example given at the beginning of the chapter is just one of the options for modeling inputs. When modeling inputs from real systems, we can use:

- Deterministic values – in systems where quantities and times are well-defined (e.g. train schedules, meeting schedules, etc.).

- Dynamic arrivals – input loads can depend on various factors such as the number of vehicles during rush hour.
- Fitting to statistical distributions – when we have appropriate data available, input loads can be modeled by fitting to statistical distributions.
- Historical data – where we have records of events in the systems (e.g. MES systems), we can perform an analysis by fitting to statistical distributions.
- Expert estimates – in the absence of records, the assessment of the behavior of individual building blocks can be estimated based on the empirical assessments of experts.
- Random values – randomness is a key element of simulations. In arrival modeling, random values are used within appropriate ranges or random values are generated according to appropriate distributions.
- Sensitivity analysis - input parameters can be varied to assess how the system behaves under different initial settings under certain assumptions.
- Service speeds – obtaining service speeds is like input intensities. It is often possible to obtain service speeds from knowledge of process durations such as production machine specifications, transport speeds, etc.

Regardless of which approach is used, it is necessary to carefully examine all selected parameters (model validation) depending on the modeled system.

2.4 Random values

The generation of random values is one of the fundamental concepts in simulations, which is why we dedicate a chapter to it. Generating a random number (L'Ecuyer, 2007) is a mathematically simple operation, but if approached incorrectly, it can lead to the appearance of patterns. The appearance of patterns in the generation of random numbers can lead to inappropriate behavior of the simulation, as unwanted dependencies may appear in the simulation flow, which would otherwise not be expected in a real system.

Computer systems for generating random values use pseudo-random number generators, where the calculation of the random value is performed by a function with an input variable. An example of a simple linear congruence generator is given by the formula:

$$X_{n+1} = (aX_n + c)\%m \tag{4}$$

Here they represent:

X_n- generator seed,
a – multiplier – determines the period and quality of randomness,
c – increment – sequence shift for greater variety of generated numbers,
m – divisor – determines the range of generated numbers.

The properties of the sequence of random numbers generated by such a generator depend on the choice of given parameters. The purpose of generators is to create as much entropy or unpredictability of states as possible, so the choice of seed is also important. When using the same seed, the function will always generate the same sequence of pseudo-random values. Depending on the needs, this may be desirable, such as when implementing different configurations with the same initial inputs or for verification. In most cases, however, it is desirable to disperse the random values as much as possible. In such cases, it makes sense to choose the generator seed as randomly as possible, for example from the current processor time when generating the random value. A linear congruence generator generates integers on the interval $[0, m-1]$, but often the generation of real numbers on the interval $[0,1]$, is desired, mainly for the purpose of normalizing the values. For this purpose, the new number is divided by m.

In modeling most real-world problems, the intensity of arrivals occurs randomly, but this randomness can usually be limited. The intensity of arrivals is thus often modeled by distributions where the arrivals are independent and follow each other at equal intervals on average. Modeling of real-world random processes is often done using the Poisson distribution:

$$P(X = k) = \frac{(\lambda^k e^{-\lambda})}{k!} \tag{5}$$

where ($P(X = k)$ is the probability of occurrence of k events, λ is the average intensity of arrivals in the time interval, and k is the number of events for which we want to find the probability. The Poisson distribution is useful in describing events such as:

- modeling customer arrivals to a store over a certain period of time,
- analysis of the number of production defects,
- forecasting the number of accidents on a section within a time period,
- arrivals of e-mail messages.

Modeling of input flows or service speeds is performed by fitting to statistical distributions (Johnson, 1987), such as Poisson or normal. These can be determined using statistical tests, histogram shape estimation, least squares, and other approaches. Once the process distributions are known, they can be used to generate random events that follow the same statistical characteristics as the systems under study.

An example of calculating randomly generated values according to a Poisson distribution with mean λ is shown in the following procedure:

$$
\begin{aligned}
&\boldsymbol{function}\ Poisson(\lambda)\\
&L = e^{-\lambda}\\
&k \leftarrow 0\\
&p \leftarrow 1\\
&\boldsymbol{while}\ (p > L)\ \textbf{do}\\
&k \leftarrow k + 1\\
&p = p * rand()\\
&\textbf{end}\\
&Poisson = k
\end{aligned}
$$

Pseudocode 1: Poisson random value generator

3 Simulation example

For a simulation example, let's take a store where customers enter, search for products for different lengths of time, and finally purchase them at the checkout. Let's define the system properties:

- 5 customers enter on average per minute,
- number of cashiers: 5,
- average purchase duration: 15 min,

– the transaction at the checkout takes an average of 5 minutes.

According to Kendall's notation, a basic server system could be described by:

$$M/M/5/30/3000/FIFO \tag{6}$$

assuming exponential customer arrivals and service, 5 cash registers, an estimated store capacity of 30 customers, and a total number of customers rounded to 3000 (estimated for one business day). We choose FIFO as the serving strategy, meaning that customers are served according to their arrival (and purchase) time.

The input parameter here represents the average arrival time between two consecutive customers. Assuming that customer arrivals are a Poisson process, we can model random arrival times as follows:

$$t_i = \frac{-\ln rnd()}{\lambda} \tag{7}$$

Table 1: Example of randomly determined arrivals according to an exponential distribution

	Random time [s]	Next arrival [s]
1	0,010422473	0,625348364
2	0,44356782	27,23941755
3	0,047033142	30,06140609
4	0,561568412	63,7555108
5	0,416494108	88,74515728
6	0,083158277	93,73465391
7	0,023527478	95,14630261
8	0,052567808	98,30037111
9	0,130142537	106,1089233
10	0,055501926	109,4390389
11	0,010422473	144,9275279
12	0,44356782	152,8945772
	…	…

Source: own

The given simulation example can be analyzed with server systems with queues, but the complexity increases with each added element. Therefore, it is advisable to use appropriate simulation tools for such problems. Simulation tools cannot be expected to produce simulation results that are completely consistent with theoretical calculations due to rounding errors and randomness, but with a well-designed simulation model, the results should come close to theoretical calculations.

From the given simulation example, we can quickly see that the system is not sustainable; with an average number of 5 customers per minute and 5 cash registers with a service speed of 5 minutes. The shopping time here represents an example that customers perform simultaneously. If we were to use Kendall's notation to describe only this part, we could describe it as a process, which can be simplified as:

$$M/M/\infty \tag{8}$$

because when shopping, each customer makes their own purchase and does not even need to enter the queue. Therefore, this segment can be considered unlimited (each customer has their own immediately available server). Customer arrivals represent arrivals as generated, and for the service speed, we consider an average of 15 minutes per customer according to the given parameters. After making a purchase, customers enter the queue (or queues in front of individual cash registers). In a concrete simulation, we should of course take into account various factors, such as working hours, breaks and snacks, loads at different times during the day, etc.

The presented example can be modeled in simulation tools (Figure 5.3) and avoids the multitude of calculations involved in increasing the complexity of server systems with queues.

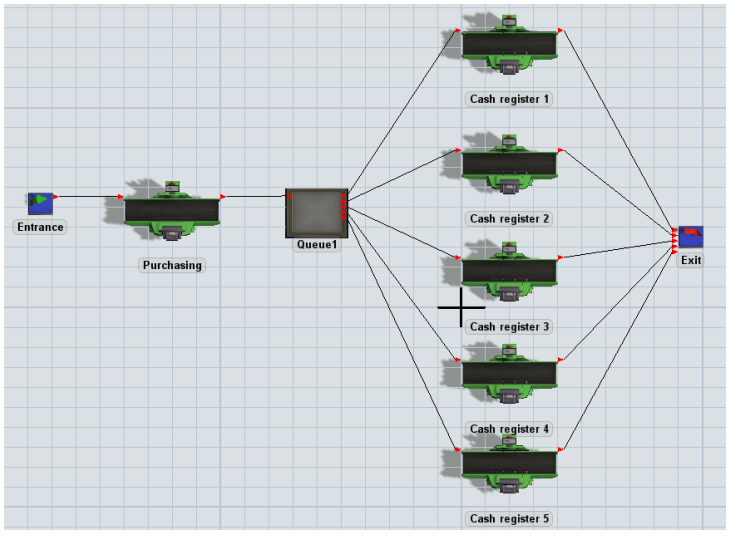

Figure 5.3: Trade simulation model in FlexSim

Source: own.

Since the given example is unstable (constantly growing queue and constant occupancy of the cash registers), let's check how we could change the system to be sustainable. We can mainly use two approaches. We can add additional cash registers or replace them with faster ones. For this scenario, we leave all settings and characteristics the same, only we speed up the cash registers by a factor of 5 (still according to an exponential distribution).

The goal of each simulation is to determine the capabilities of the modeled system, which includes various characteristics. In this case, we focus on the size of the queue (Figure 2.4) and the waiting times in it (Figure 2.5) and the utilization of the cash registers (Figure 2.6).

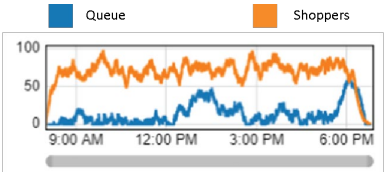

Figure 2.4: Queue length
Source: own.

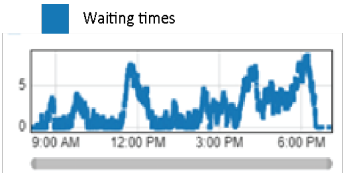

Figure 2.5: Waiting times in the queue
Source: own.

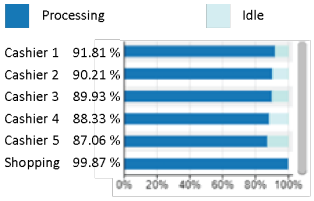

Figure 2.6: Cash register utilization
Source: own.

The simulation results show a (relatively) stable system with changed characteristics, as we do not have constantly increasing queues and waiting times. In simulations of complex systems, it is often necessary to find solutions that avoid bottlenecks and unused resources.

7 Conclusion

The presented example shows only a fraction of the capabilities that simulations offer. The great usefulness of simulations is especially evident in the study of complex systems, where seemingly unrelated parameters are involved. Thus,

simulations are used in logistics, finance, information, production, or in fact any related field. In the field of logistics, simulations represent a cost-effective approach to the analysis of production processes, transport routes and routing, traffic patterns, etc. By changing the parameters of the simulation or simulation scenarios, it is possible to observe complex systems from different perspectives, which enables effective decision-making based on rational analyses.

Performing simulations allows a cost-effective approach to the analysis of complex systems without the need to interrupt processes. In today's technological systems, there is an increasing integration of various solutions, from ERP, MES, PLC, SCADA systems and others. While such systems mainly record and can also largely manage processes, simulation represents an alternative option for optimization by comparing alternatives and supporting decision-making. Recently, digital twins have come to the fore, providing a comprehensive insight into various company processes. Digital twins capture the events of a company's processes in real time from sensors, machines, devices and other sources (Internet of Things) and build a virtual representation based on them. Simulations performed on a virtual image of a real system provide a deeper insight into operations and business and represent added value in business decision-making at all levels.

References

Bolch, G., Greiner, S., Meer, H. de, & Trivedi, K. S. (2006). *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications, 2nd Edition | Wiley* (2nd ed.). John Wiley & Sons. https://www.wiley.com/en-us/Queueing+Networks+and+Markov+Chains%3A+Modeling+and+Performance+Evaluation+with+Computer+Science+Applications%2C+2nd+Edition-p-9780471565253

Fishman, G. (2002). Discrete-event Simulation: Modeling, Programming, and Analysis. In *J. Artificial Societies and Social Simulation* (Vol. 5). https://doi.org/10.1007/978-1-4757-3552-9

Johnson, M. E. (1987). *Multivariate Statistical Simulation: A Guide to Selecting and Generating Continuous Multivariate Distributions* (1st edition). Wiley.

L'Ecuyer, P. (2007). Random Number Generation. In *Springer Handbooks of Computational Statistics* (pp. 93–137). https://doi.org/10.1002/9780470172445.ch4

Thomopoulos, N. T. (2012). *Fundamentals of Queuing Systems: Statistical Methods for Analyzing Queuing Models*. Springer Science & Business Media.