http://doi.org/10.35784/iapgos.6555 received: 08.09.2024 | revised: 30.10.2024 | accepted: 15.03.2025 | available online: 31.03.2025

A STOCHASTIC INTERVAL ALGEBRA FOR SMART FACTORY PROCESSES

Piotr Dziurzanski[1], Karol Kabała[2], Agnieszka Konrad[3,4]
[1]West Pomeranian University of Technology, Faculty of Computer Science and Information Technologies, Szczecin, Poland, [2]Numlabs Ltd., Krakow, Poland, [3]Polish Academy of Sciences, Institute of Bioorganic Chemistry, Poznan, Poland, [4]Papukurier Sp. z o.o., Poznan, Poland

Abstract. *This paper presents a stochastic interval algebra specifically developed to evaluate the time and cost properties of smart factories. This algebra models production tasks as intervals and treats allocation and scheduling as algebraic operations on these intervals, with the goal of analysing the impact of resource allocation decisions on total production time or economic cost. The theoretical foundations of this notation are introduced, and then several simple examples of their use are presented. The proposed algebra can be also applied to describe multi-stage production and service processes, recorded with an activity-on-arrow type of graphs, In addition, it was analysed a real-life application of the described technique to planning and scheduling the activities in restaurants preparing takeaway meals. The data was collected in 30 restaurants throughout Poland, using a bespoken software/hardware Kitchen Delivery System, in which over 65,000 orders were registered. Time criteria for the correctness of individual stages of meal preparation were proposed and, after filtering out incorrect orders, the appropriate probability distributions were fitted to the remaining measured activity durations. The resulting probabilities can then be used in practice to improve the accuracy of predicting the completeness of food preparation, which in turn should improve food delivery planning with greater accuracy and enable more accurate order delivery times to be provided to end customers*

Keywords: interval algebra, smart factories, kitchen delivery systems

STOCHASTYCZNA ALGEBRA INTERWAŁOWA OPISUJĄCA PROCESY INTELIGENTNEJ FABRYKI

Streszczenie. *W artykule przedstawiono stochastyczną algebrę interwałową stworzoną specjalnie w celu oceny właściwości czasowych i kosztowych inteligentnych fabryk. Algebra ta modeluje zadania produkcyjne jako interwały i traktuje alokację i planowanie jako operacje algebraiczne na tych interwałach, mając na celu analizę wpływu decyzji o alokacji zasobów na całkowity czas produkcji lub koszt ekonomiczny. Wprowadzono podstawy teoretyczne tej notacji, a następnie zaprezentowano kilka prostych przykładów ich użycia. Podano także przykłady zastosowania proponowanej algebry do opisu kilkuetapowych procesów produkcyjnych i usługowych, zapisanych za pomocą grafu z aktywnościami na przejściach. Ponadto przeanalizowano zastosowanie opisanej techniki do planowania i usług w restauracjach przygotowujących posiłki na wynos. Dane zostały zebrane w 30 restauracjach w całej Polsce z wykorzystaniem zaproponowanego programowo-sprzętowego systemu dostaw w kuchni, w którym zarejestrowano ponad 65 000 zamówień. Zaproponowano czasowe kryteria poprawności poszczególnych etapów przygotowania posiłku i po odfiltrowaniu niepoprawnych zamówień, do pozostałych zmierzonych czasów trwania czynności dopasowano odpowiednie rozkłady prawdopodobieństwa. Otrzymane prawdopodobieństwa można następnie wykorzystać w praktyce do poprawy dokładności przewidywania kompletności przygotowania żywności, co z kolei powinno poprawić planowanie dostaw żywności z większą dokładnością i umożliwienie podawania klientom końcowym dokładniejszego czasu terminu dostawy zamówienia.*

Słowa kluczowe: algebra interwałowa, inteligentne fabryki, systemy dostaw w kuchni

Introduction

In the contemporary economy, sometimes referred to as Society 5.0, the goal is to finish tasks faster with the help of machines for the benefit of humans [4]. Following this direction, even relatively small companies, operating in the service industries, shall benefit from automation and optimisation of their processes. This is in contrast with more popular notion of Industry 4.0, which were related to manufacturing and production industries [10].

Job scheduling in service vendors or industrial factories resembles task scheduling in contemporary many-core devices and thus similar techniques can be applied to solve both [7]. In real-time computer systems, schedulability analysis is used to prove that all tasks will be executed before their deadlines. Despite the numerous techniques used for this purpose, surveyed for example in [3], their accuracy depends on the knowledge of so-called worst-cast execution time (WCET) of each task, whose determination by itself is a challenge [11]. This restrictive assumption can be relieved after applying statistical analysis of WCET [6]. Hower, in both soft real-time systems and non-real time systems, the average execution time (AET) is usually more important than the notoriously pessimistic WCET. In this paper, WCET, AET and even best-case execution time (BCET) are considered simultaneously, as the execution time is modelled with its probability density function (pdf). This function may be constructed based on large-scale measurements in situ, as described in section *Use case*.

1. Related work

Interval Algebra (IA) for task scheduling was introduced in [7]. Its principles are based on the calculus for temporal reasoning named Allen's interval algebra [1]. In the Allen's algebra, elements are intervals, i.e., an ordered pair of points with the first point lower than the second. In this algebra, a model consisting of a fully ordered set of points of time is assumed. Allen denoted assuming the lesser and greater endpoint of interval x by $x-$ and $x+$, respectively. Seven binary relations have been defined, for example interval X meets interval Y, whose equivalent relation on endpoints can be described as $x+ = y-$.

In [6], the following three-step approach for process scheduling has been proposed. In the first step, an estimate ω_i of the WCET of task τ_i with a specified confidence level ϱ_i is constructed using the Gumbel distribution function approximating the maximum of a set of sample data. In the second step, these estimates are used to construct a schedule. At the third stage, both the sample data and schedule are used to calculate confidence ξ with which the whole system is expected to meet all task deadlines. In that paper, it is also described a technique for deriving individual ϱ_i from system confidence ξ.

Bernaet et al. in [2] stressed that combining random variables representing execution time can be performed with the convolution operation only for tasks whose execution time is independent and identically distributed (iid). Otherwise, the inaccuracy of the combination is proportional to the inter-task dependencies. Since they argued that execution times of code sections are not iid, a new Execution Time Profiles (ETP) algebra were introduced. This algebra included so-called joint execution profile capturing these dependencies and an equivalent operator for the convolution for the joint execution profiles. Another safe yet pessimistic operator for combining ETPs was provided in case the existence of the inter-task dependence is unknown.

Convolution computing of two pdfs is costly in the time domain. As in the IA reference implementation described in [7] this operation was performed in time domain, only analysis of simple cases was presented. In [9], FFTW library has been used to perform convolutions based in the frequency domain on the Fast Fourier Transform. This approach has been shown practical for real-world-size problems.

2. Deterministic interval algebra

For the sake of self-consistency of this paper, the most important features of IA introduced in [7] are described in this section. For more information regarding this algebra, the reader is referred to that paper.

A process is viewed as a set of jobs, a job set $\Gamma = \{\tau_1, \tau_2, \tau_3, \ldots\}$. These jobs appear exactly once during the whole considered time horizon and, hence, they are often referred to as singletons. These jobs can be represented with intervals while using IA. As the jobs are intended to be allocated to plant resources, the interval length is equal to the processing time on the resource the job is allocated to. In [7], the following hash-based notation is used for a singleton τ_1: $\#\tau_1\#0\#10$, where the first element of the tuple is the unique job identifier, the second element determines the release time of the job and the third element is the processing time of τ_1 on the target resource. In the example, job τ_1 is released (i.e., ready for processing) at time instant 0, and its processing requires 10 time units.

The release time may depend on other job, which is described in the hash-notation with $\#\tau_2\#\tau_1\#20$, meaning that task τ_2, whose processing time equals 20 time units, can be released only after task τ_1 finishes its processing. This notation is then a shortcut of the τ_1 *earlier than* τ_2 from the Allen's algebra. The hash-notation is capable of a compact description of a dependency on an interval set, for example $\#\tau_3\#\{\tau_1, \tau_2\}\#100$ means that τ_3 can be released only after the processing of both τ_1 and τ_2 has been finished.

All the intervals introduced above appeared only once during the entire analysed timeline, but IA is expressive enough to describe both the periodic and sporadic jobs. As such tasks do not appear in the analysed use case, these job types are treated out of the scope of the papers. The reader is referred to [7] for more information regarding IA representation of those recurring jobs.

To denote the job mapping to processing resources in the hash notation, an algebraic operator γ is employed. This operator specifies the way of resource sharing between the jobs mapped to that resource as well as the impact of the sharing on the job processing time. The γ operator is applied to the set of intervals surrounded by brackets, for example $\gamma(\#\tau_1\#0\#10, \#\tau_2\#0\#20)$. To evaluate the operator, another symbol, &, has been introduced in the hash notation, which precedes the completion time of the jobs. Hence, the operator in the previous example can be evaluated in the manner presented in formula (1) assuming the FIFO dispatching heuristics.

$$\gamma(\#\tau_1\#0\#10, \#\tau_2\#0\#20) = \gamma(\#\tau_1\&10, \#\tau_2\&30) = \gamma([0, 30)) \quad (1)$$

In the evaluation above, in the first step, the symbol & is used which preserves the information regarding jobs and their dispatching order. The second step loses the information regarding the individual jobs and determines the busy period(s) of the resource with one or more intervals. The latter way of evaluation is then named information-collapsing.

IA is also capable of modelling job affinities, i.e., the situation when a job can be processed by only a subset of available resources. Affinity can limit the processing to only one resource type (for a single-typed jobs) or several enumerated resource types, with the same or different processing times. In the hash notation, it is expressed with pointy brackets, which specify different resource types and different processing times. For example, in formula (2), task τ_4 can be processed on resource α_1 only, whereas task τ_5 can be processed on resources α_2, α_3, α_8 that would require 5, 10 and 15 time units, respectively.

$$\gamma(\#\tau_4 < \alpha_1 > \#0\#15, \#\tau_5 < \alpha_2, \alpha_3, \alpha_8 > \#0\# <5,10,15>) \quad (2)$$

In the absence of pointy brackets, a job can be processed on any type of resources.

3. Representing a manufacturing process with deterministic interval algebra

The stages used in a plant to process some goods can be presented visually using so-called Activity on Arrow (AoA) notation. Its example is provided in Fig. 1. In AoA, synchronisation points in a manufacturing process are represented with states (labelled with numbers in Fig. 1), whereas the transitions between these states denote the manufacturing activities that needs to be executed in the order described by the graph, to reach a following stage in the entire manufacturing process (the activities are denoted with capital letters in Fig. 1). In the example given in Fig. 1, it is assumed that manufacturing actions A, …, F require d_A, …, d_F time units to be executed, respectively. The modelled manufacturing process starts at time t_1 in state 1. At this synchronisation point, the first manufacturing activity A is carried out which transforms the entire manufacturing process to synchronisation point 2 after d_A time units. State 2 is reached then at $t_2 = t_1 + d_A$. State 2 has two successors, state 3 and 4. However, state 4 also depends on state 3, hence it may be reached only after both manufacturing actions B and D have been completed. State 3 is then reached at $t_3 = t_2 + d_C$ and state 4 at $t_4 = \max(t_2 + d_B, t_3 + d_D)$. The last manufacturing action E completes at $t_5 = t_4 + d_E$.

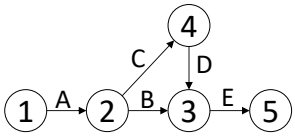


Fig. 1. Activity on Arrow representation of a plant

The whole manufacturing process from Fig. 1 can be evaluated using IA presented in (3).

$$\gamma(\#A\#\, t_1\#d_A, \#B\#A\#d_B, \#C\#A\#d_C, \#D\#C\#d_D, \#E\ \#\{B, D\}\#d_E\ ) = \\ = \gamma([t_1, t_1 + d_E + \max(d_A + d_B, d_A + d_C + d_D)) \quad (3)$$

Assuming that the manufacturing process commences at $t_1 = 0$ and the processing times for manufacturing actions A to E are fixed as given in Table 1, the whole manufacturing time can be evaluated as $\gamma([0, 0 + 10 + \max(5 + 10, 5 + 15 + 5)) = \gamma([0, 35))$.

Table 1. Example processing times

Manufacturing action	A	B	C	D	E
Processing time	5	10	15	5	10

4. Stochastic interval algebra

All examples presented in the previous section were purely deterministic: both the release time and the processing time were expressed with constants. However, it is possible to apply so-called aleatory variables instead, which are associated with certain probability distributions. IA notion does not impose any limitation on the choice of probability distributions and their types and parameters in the IA notation follows the traditional probability notation used in statistics. For example, if task τ_5 in released according to the normal distribution with mean $\mu = 3$ and variance $\sigma^2 = 1$, denoted traditionally, can as $N(\mu, \sigma^2)$, and its execution time is described with distribution $N(10, 1)$, its IA-based notation assumes form as in (4):

$$\#\tau_5\#\text{normal}(3, 1)\#\text{normal}(10, 1) \quad (4)$$

Task τ_5 finishes its execution at the time instant following the probability distribution computed as the convolution of two Gaussians: $N(3, 1) * N(10, 1)$.

It is also well-suited to represent time intervals as discrete random variables described by a probability mass function (PMF), which is a function giving the probability that a discrete random

variable is equal to a provided value. Every value of a PMF must be non-negative and add up to 1. If we use this function for the description of task execution time, the best-case execution time (BCET) and the worst-case execution time (WCET) are indicated by the first and the last probability value of the distribution, respectively. The probability of the other possible execution times is described between these two extreme cases. In the example PMF shown in Fig. 2, the BCET and WCET of the task execution time is 5 s and 9 s, appropriately. This task is described by the IA formula (5).

$$\#\tau_6\#0\#\text{pmf (5, 0.1), (6, 0.1), (7, 0.2), (8, 0.4), (9, 0.2)} \quad (5)$$

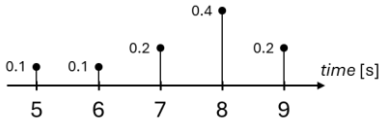


Fig. 2. Probability mass function (PMF) of a discrete random variable describing an example task execution time

In formula (6), it is shown how two tasks with stochastic timing can be mapped to a certain resource. Task τ_7 is released in time instant described by discrete uniform distribution U{0, 1} and is executed in 40 time units. Task τ_8 depends on τ_7 and is executed in U{1, 4} time units by a certain resource *a* with FIFO scheduling. , to demonstrate how.

$$a(\#\tau_7\#\text{pmf (0, 0.5), (1, 0.5)\#40, }\#\tau_8\#\tau_7\#\text{pmf (1, 0.25), (2, 0.25),}$$
$$\text{(3, 0.25), (4, 0.25)) } = a(\text{[pmf (0, 0.5), (1, 0.5), pmf (41, 0.125),}$$
$$\text{(42, 0.25), (43, 0.25), (44, 0.25), (45, .125)))} \quad (6)$$

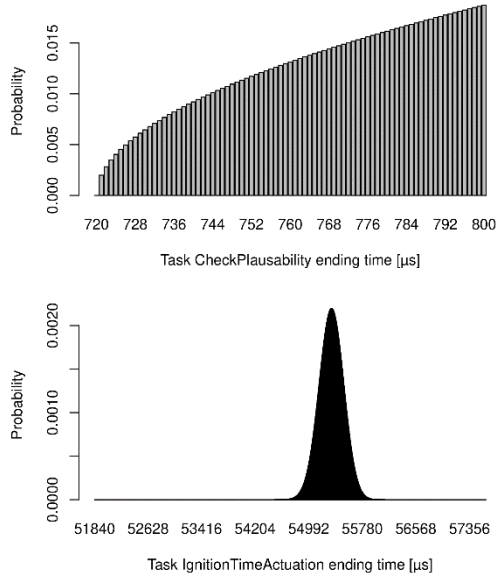


Fig. 3. Distributions of ending time for tasks CheckPlausability (above) and Ignition-TimeActuation (below) from DemoCar

As an example of a real-life stochastic model than can be described using IA, we refer to DemoCar, a simple automotive gasoline engine model described in [5]. In this model, all execution timings are treated as random variables and described with a discretised, lower- and upper-bounded Weibull distribution. The first task executed in this model is *CheckPlausability*. It started at 0ms and ended with the PMF presented in Fig. 3 (above). This task can be described using IA as #CheckPlausability#0#pmf (721, 0.00198), (722, 0.00282), ... (78 values), ... (800, 0.01872), where all timings are given in µs. The ending time of IgnitionTimeActuation, the last executed task in that model, has been evaluated with IA. The possible ending times, in a form of 5761 PMF nonzero values, are presented in Fig. 3 (below).

The execution times of processes in the simple example of a plant presented in Fig. 1 have been described with probability distributions provided in Table 2. Assuming that $t_1 = 0$, the (stochastic) processing time can be determined, for example applying well-known property that the convolution of a pair of Gaussian PDFs are also Gaussian. For other distributions, arithmetical computations on random variables can be performed numerically, e.g. using the PaCal package [8]. This package has been also used to generate the probability of the ending time of job 5 shown in Fig. 4.

Table 2. Example execution times described with distributions

Job	A	B	C	D	E
Processing time	N(10,1)	N(20,2)	N(30,4)	N(40,5)	N(30,2)

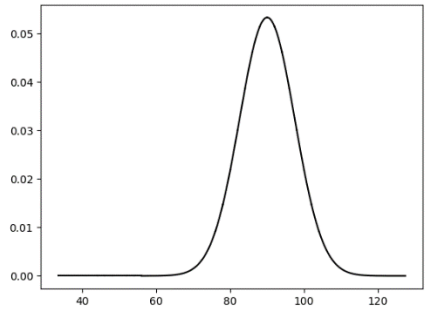


Fig. 4. Distributions of ending time for job 5 from the example given in Fig. 1 assuming job execution time distributions from Tab. 2

5. Use case

Society 5.0 has under its umbrella such businesses as restaurants or other food serving places. This branch, despite being huge and important in each economy, is usually immune to process automation. Consider that, according to Polish GUS, in the whole country the number of such food places was equal to 83.9 thousand in 2022, and their total income in Poland was 64.6 bln PLN [12].

In our work, we focus on restaurants that deliver food to the customer premise. In order to provide more reliable delivery time to the customers during their food order, a kitchen simulation model has been developed. To build such model, a considerable amount of food order and preparation processes have been recorded using a bespoke hardware module, named Kitchen Display System (KDS). An example of software and hardware integration within a prototype of a data collection system is presented in Fig. 5. As this system was intended to be used by the actual kitchen employee, it was designed to be extremely simple and user friendly. As KDS was intended to be used by kitchen staff focusing rather on performing their main duties rather than recording data for further analysis, a significant probability of human errors has been assumed and hence the automatic validation of the correctness of data collected by the kitchen data acquisition system has been introduced.

Fig. 5. An example of software and hardware integration within a prototype of a data collection system papu.io

5.1. Data acquisition

Gathering reliable data from the kitchen posed a significant challenge. The initial KDS system, with its low level of ergonomics, was used very reluctantly and incorrectly by the restaurant staff. Decomposing the process into stages and preparing the graphical interface required several iterations in collaboration with the kitchen staff. Thanks to these improvements, the data collection system (KDS) could be operated in an intuitive manner that did not disrupt daily work. Despite the optimizations, there were instances where a restaurant employee forgot to enter information or entered it incorrectly into the system. To monitor this phenomenon, a service was prepared that checked the statistical parameters of the collected data on a daily and weekly cycle. Analysis of these reports allowed for addressing issues such as reminding employees to use the system and training new employees. Together with a group of experts in the gastronomy field, the results were analysed, and filters were established to separate outliers caused by human error from realistic values

The validated features come from tables collected by the kitchen data acquisition system and contain information about the times of order state changes and their parameters, including location, associated user, delivery method. During validation, the presence of all necessary implementation phases is checked and whether their duration does not deviate from the assumed threshold and detection of duplicate order states. Only four activities have been assigned to a food order, namely:

A. Order is waiting in the queue.
B. Cold preparation of the order (e.g. preparing a pizza slice).
C. Hot preparation of the order (e.g. baking, cooking etc.).

The four states have the following meaning

1. Order accepted by the restaurant.
2. Start of order cold processing.
3. Start of order hot processing (e.g. placing a pizza in the oven).
4. Completion of the order which is ready to be transferred to the courier.

The AoA of this process is presented in Fig. 6.

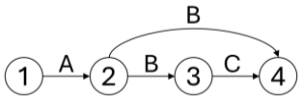


Fig. 6. Activity on Arrow representation of the pizza (path 1-2-3-4) or bottled drink (path 1-2-4) preparation process

As shown in the AoA diagram, two state sequences listed above are considered correct: 1-2-4 or 1-2-3-4. Notice the nondeterminism in state 2 with two outgoing arrows with symbol B. The 1-2-4 state sequence is considered simplified. It occurs, for example, in salads that do not need to be cooked. An order is considered incorrectly marked on the kitchen side when at least one of the states is duplicated (e.g. 1-1-2-4) or there is no meaningful transition between the states, i.e. there is no beginning or end of the order (e.g. 2- 2, 3-2).

Assumptions made during calculations to detect anomalies (we also record the order as incorrect):

- To consider the order as fully successful, the time between states 3 and 2 (baking and started cold processing) must be at least 20 seconds, and the time between states 4 and 3 (completed and started hot processing) must be 10 seconds.
- To consider the order as fully good, the time between states 1 and 2 cannot be longer than 50 minutes, between 2 and 3–10 minutes, between 3 and 4–2 minutes.
- For simplified orders (without the hot processing activity): transition time from state 2 to 4 between 25 and 660 seconds (11 minutes).
- The upper threshold values were determined based on the 90th percentile of a given parameter.

During stage 1 (until February 28, 2022), the data collection system was launched in 10 restaurants and later (as of August 31, 2022) also successively with other partners, in a total number of approximately 30 (a partner is understood as a restaurant that registered in the KDS system until that date minimum 1000 orders). The demo (including partner restaurants) was shown to a total of (as of August 31, 2022) 207 catering outlets. The restaurants selected for the study were located throughout Poland. They represent the specificity of large urban centres, smaller towns and urban-rural areas. Among the restaurants, pizzerias and sushi establishments dominated. Each time, employees and the owner were trained by the person implementing the system. The system was implemented by a restaurateur who had a KDS set running and independently tested in his restaurant.

The number of orders collected using the KDS system in the period 1/12/2021 - 30/04/2022 was: 35,297 (average: 2,941 orders per restaurant). In turn, in the period from December 1, 2021 to June 1, 2022, it amounted to: 65,845 orders (average: 4,115 orders per restaurant).

5.2. Example results and IA applicability

Data from the restaurant with internal ID 222 was used to generate the graphs presented in this subsection. Due to the large number of pizza orders in that restaurant, only the orders related to this kind of dish were analysed. The data has been collected between 2022-02-17 and 2023-06-09. Before the data processing, its preprocessing was carried out in which wrong orders were filtered out. Only two state orders were accepted: full state orders with states 1-2-3-4 and simplified state order with states 1-2-4.

In addition, only orders with the following timing properties remained unfiltered out: (1) with time interval between 10 and 10000 seconds for a transition between states 1 and 2 and (2) the time from 10 to 1500 seconds for any remaining state transition (2 -> 3, 3 -> 4, 2 -> 4).

The histogram of the time between the first and second states for the selected restaurant, i.e. the time from accepting the order in the restaurant to starting its execution in the kitchen, is presented in Fig. 7. In Fig. 8, the histogram of the time between the second and third state, i.e. the time from the starting of the food processing to the instant when the order is ready to be handed over to the courier.

In Fig. 9, the same histograms are shown but this time the data have been generated based on the data acquired in all restaurants taking part in the experiment. The histogram of the durations between states 1 and 2, presented in Fig. 9 (above), is best described with the Pareto distribution with parameters $b = 4.12$, $loc = -2278.34$, $scale = 2288.36$, whereas the distribution between states 2 and 4, shown in from Fig. 9 (below), can be best described with the Laplace distribution with $loc = 523.45$ and $scale = 151.11$.

Having fitted the probability distributions of all the activities in the process following AoA from Fig. 6, it is then possible to apply IA in the same way as presented in section 4 and evaluate the process ending time using numerical methods, for example with the PaCal package mentioned earlier. The straightforward practical application is, by using the suitable distributions, determine the probability distribution of the completion time of the entire order. This value can be utilised to schedule food delivery with better precision and to supply the end-customers more exact promise time for food delivery.

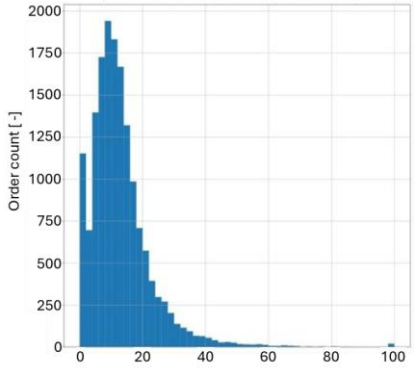


Fig. 7. Histogram of the time between the first and second status for the selected restaurant, i.e. the time from accepting the order in the system to starting its execution in the kitchen

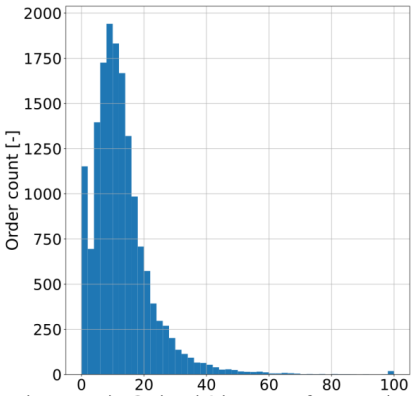


Fig. 8. Histogram of the time between the second and third state for the selected restaurant, i.e. the time since the start of implementation in the kitchen until ready to be handed over to the courier for "pizza" category dishes

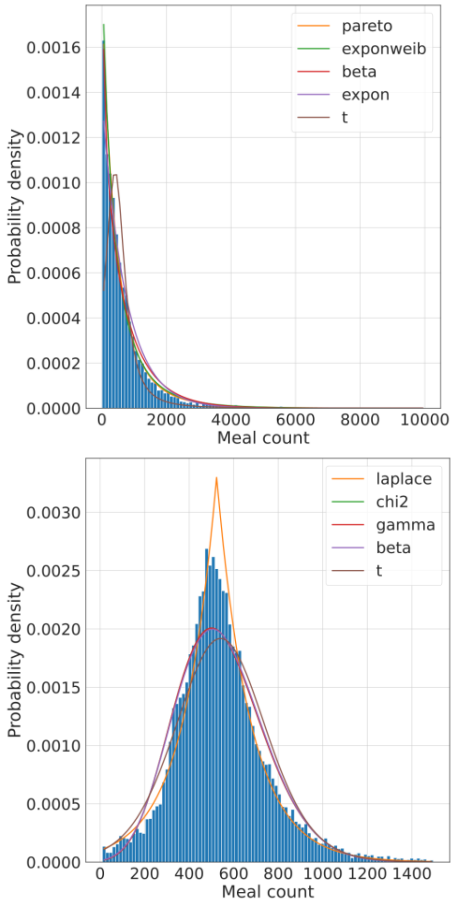


Fig. 9. Distribution fitting to the time duration of the order preparation stages between states 1 and 2 (above) and 2 and 4 (below)

6. Conclusion

A stochastic extension to an interval algebra based on the Allen's algebra has been presented. In particular, the paper focused on its application to the evaluation of the timing of manufacturing goods or delivering a service in smart factories. The interval algebra has been applied to a real-life use case of restaurants delivering food for take-away consumption. The entire process of food preparation has been described in a form of a non-deterministic Activity on Arrow graph. A software/hardware-based Kitchen Delivery Systems have been deployed to 30 restaurants throughout Poland and more than 65,000 orders were recorded. The orders not fulfilling the assumed timing constraints have been filtered out from the set. For the measured time durations for activities, appropriate probability distributions have been fitted. These probabilities can be described in the form of the presented stochastic algebra and then various temporal relations can be evaluated numerically using dedicated software packages, such as the one presented in [8]. For example, by adding the appropriate distributions, the probability distribution of the entire time of the food preparation can be determined. This value can be then used in practice to improve the accuracy of the prediction of the food preparation completeness, which in turn can be employed to schedule food delivery with a higher accuracy and to provide to the end-customers more accurate promise time for food delivery.

In future, we are planning to employ the described stochastic interval algebra to a job shop scheduling problem, as a formalism in a fitness function in a search-based optimisation process.

Acknowledgement

Project No. POIR.01.01.01-00-0799/21 co-financed by the European Union from the European Regional Development Fund under the Smart Growth Programme. The project was implemented by Papukurier Sp. z o.o. as part of the National Centre for Research and Development's Fast Track call.

References

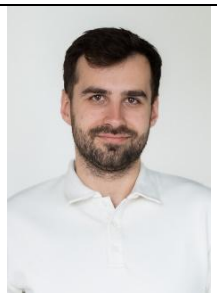
[1] Allen J. F.: Maintaining knowledge about temporal intervals. Communications of the ACM 26(11), 1983, 832–843.
[2] Bernat G. Colin A., Petters S. M.: WCET analysis of probabilistic hard real-time systems. 23rd IEEE Real-Time Systems Symposium (RTSS 2002), 2002.
[3] Davis R. I., Burns A.: A survey of hard real-time scheduling for multiprocessor systems. ACM computing surveys (CSUR) 43(4), 2011, 1–44.
[4] Deguchi A. et al.: What is society 5.0. Society 5.0, 2020): 1–24.
[5] Dziurzanski P., Singh A. K., Indrusiak L. S.: Energy-aware resource allocation in multi-mode automotive applications with hard real-time constraints. 19th International Symposium on Real-Time Distributed Computing (ISORC), York, UK, 2016, 100–107.
[6] Edgar S., Burns A.: Statistical analysis of WCET for scheduling. 22nd IEEE Real-Time Systems Symposium (RTSS 2001) (Cat. No.01PR1420), London, UK, 2001, 215–224.
[7] Indrusiak L. S., Dziurzanski P.: An interval algebra for multiprocessor resource allocation. International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS), Samos, Greece, 2015, 165–172.
[8] Korzeń M., Jaroszewicz S.: PaCAL: A Python package for arithmetic computations with random variables. Journal of Statistical Software 57, 2014, 1–34.
[9] Manolache S., Eles P., Peng Z.: Memory and time-efficient schedulability analysis of task sets with stochastic execution time. 13th Euromicro Conference on Real-Time Systems (ECRTS '01). IEEE Computer Society, USA, 2001.
[10] Ustundag A., Cevikcan E.: Industry 4.0: managing the digital transformation. Springer, Cham 2018.
[11] Wilhelm R. et al.: The worst-case execution-time problem – overview of methods and survey of tools. ACM Transactions on Embedded Computing Systems (TECS) 7(3). 2008, 1–53.
[12] Główny Urząd Statystyczny. Rynek wewnętrzny w 2022 roku. 2023.

D.Sc. Ph.D. Eng. Piotr Dziurzanski
e-mail: piotr.dziurzanski@zut.edu.pl

He received the Ph.D. and D.Sc. degrees in computer engineering from the West Pomeranian University of Technology in Szczecin, Poland, in 2003 and 2022, respectively. Currently, he is an associate professor at that university, and a visiting researcher at the University of York, UK. In the past, he worked at the University of York, UK, and Staffordshire University, UK. He was involved in 2 EU-funded research projects and a national project funded by EPSRC (UK). He was also PI or CI in 3 national projects funded by the National Science Centre in Poland.
His main research interest is related to resource allocation, cloud computing, parallel processing, and reconfigurable hardware.

https://orcid.org/0000-0001-9542-652X

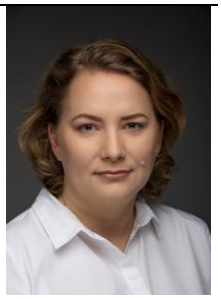
M.Sc. Eng. Karol Kabala
e-mail: k.kabala@numlabs.com



A graduate with a master's degree in automation and robotics from AGH University of Science and Technology in Kraków. For the past 5 years, he has been associated with Numlabs, a company that conducts research and development projects in the fields of information technology and artificial intelligence.
His areas of interest include the application of optimization techniques and computer vision in industry 4.0 and remote sensing.

https://orcid.org/0000-0003-0380-9270

M.Sc. Agnieszka Konrad
e-mail: aga.konrad@gmail.com



Agnieszka Konrad graduated with a degree in International Relations from Poznań University of Economics (Poland) and holds a degree in European Economy & Management from Abertay University Dundee (Scotland). Currently, she serves as the Deputy Director for Collaboration at the Institute of Bioorganic Chemistry, Polish Academy of Sciences in Poznań. Previously, she managed Project Support Office focusing on enhancing the administration and coordination of research projects. Her professional experience includes collaboration on numerous projects at the national and international levels.
Her interests include research collaboration and the development of strategic partnerships within the scientific community.

https://orcid.org/0000-0002-7900-3153