Metric for Calculation of System Complexity based on its Connections

João R. B. Paiva, Viviane M. Gomes, Bernardo A. Rodrigues, Lais F. A. Silva, Bruno C. M. Aniceto, Geovanne P. Furriel, Wesley P. Calixto

Abstract—**This paper proposes a methodology based on system connections to calculate its complexity. Two study cases are proposed: the dining Chinese philosophers' problem and the distribution center. Both studies are modeled using the theory of Discrete Event Systems and simulations in different contexts were performed in order to measure their complexities. The obtained results present i) the static complexity as a limiting factor for the dynamic complexity, ii) the lowest cost in terms of complexity for each unit of measure of the system performance and iii) the output sensitivity to the input parameters. The associated complexity and performance measures aggregate knowledge about the system.**

Index Terms—**complexity, connections, discrete events systems, modeling, simulation.**

I. Introduction

Systems are studied in different areas by observing their parts and their behavior. A system consists of elements, arranged in a natural or controlled manner to fulfill a certain goal. To Rechtin and Maier [1], a system is a collection of things or elements, that by working together produce results that are impossible to be obtained by the elements themselves individually. Each part interacts directly or indirectly with each other and performs functions on behalf of the whole.

The systems' behavior in time can be investigated to verify patterns, relationships, hierarchy and other features. Many systems have a high variability in their parameters. To Per Bak [2], such characteristic defines them as complex. This type of systems are non-linear, hierarchical, emerging and self-organizing [3], i.e., they have variables that may emerge at any time or assume different levels of importance according to the system dynamics, producing a behavior that is difficult to predict.

In complex systems, the whole is considered as more than just the sum of the parts, referring to the fact that the set of properties is not easily inferred from the properties of the parts and the laws of their interactions [4]. In a wide range of scales, the variability may express itself, leading to the identification of the separate parts. This fact can be observed in the relationship between humans, who are able to recognize other humans because they are all different. In this context, the brain can be considered as the most complex system of all, because it can form a representation of the complex external world [2].

Faced with the need to quantify the complexity of systems, several authors have proposed different metrics. According to Lloyd [5], these metrics are developed to respond questions about the system with respect to: i) difficulty of description, ii) difficulty in creating or iii) degree of organization.

The system complexity can be evaluated for its dynamic operation or for its a static configuration. The static complexity measure is independent of the simulation and is considered as a reference. In this work, the maximum number of active connections between elements of the system is adopted to calculate the static complexity. The dynamic complexity measure considers the evolution of the system over time, in which the number of active connections varies at every state change [6].

Events contribute to the systems' dynamics, provoking changes of state [7]. During the process of simulating discrete events systems, the input parameters are changed and the output is observed. The simulation provides data to the analysis of parameter sensitivity, and the goal is to verify each input variable's contribution to some output. According to Saltelli [8], the sensitivity analysis is the study of how uncertainty in the model can be divided to different sources of doubt in the inputs.

The sensitivity analysis is relevant to the study of complexity because certain variables can eventually emerge and cause significant impact to the system. To Holland [3], this emergence characterizes complex systems and help distinguish such systems from others.

The purpose of this paper is to present a methodology for measuring the static and dynamic complexities of systems,

The authors would like to thank the National Council for Scientific and Technological Development (CNPq), the Foundation for Research in the State of Goiás (FAPEG) and the Higher Education Personnel Improvement Coordination (CAPES) for financial assistance to this research.

João R. B. Paiva, Viviane M. Gomes, Bernardo A. Rodrigues, Bruno C. M. Aniceto, Geovanne P. Furriel and Wesley P. Calixto are with the Experimental and Technological Research and Study Group (NExT) from Federal Institute of Goias (IFG), Goiânia, GO 74055-110 Brazil (e-mail: jricardobpaiva@yahoo.com.br) and School of Electrical, Mechanical and Computer Engineering (EMC) from Federal University of Goiás (UFG), Goiânia, GO 74605-010 Brazil (e-mail: wpcalixto@gmail.com).

Lais F. A. Silva is with Experimental and Technological Research and Study Group (NExT) from Federal Institute of Goias (IFG), Goiânia, GO 74055-110 Brazil.

using as case studies two systems modeled by discrete events. Section II presents the problems that underlie these systems modeling and Section III brings the complexity metrics. Section IV presents the methodology used for the computer simulation and the complexity measure of the proposed systems. In section V, the results are presented.

II. Problems

A. *Dining Chinese Philosophers*

The Dining Chinese Philosophers problem, proposed by the Dutch computer scientist Edsger Dijkstra in 1965, became a classic representation of synchronization in Concurrent Programming, where there is competition for limited resources [9]. The problem consists of a situation in which there are five philosophers sitting at a circular table, there is a plate of pasta for each philosopher and a *hashi* between each pair of plates, as illustrated in Fig 1.

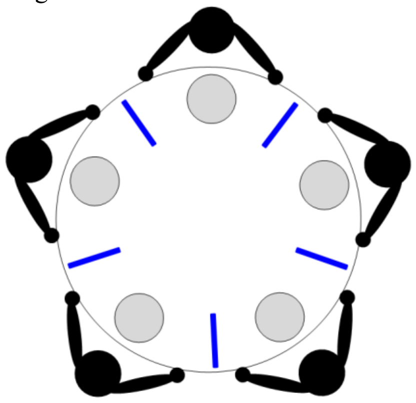

Fig. 1. Dining Chinese Philosophers.

Each philosopher needs two *hashis* to eat. Thus, when a philosopher wants to eat, he tries to catch the *hashis* that are adjacent to his plate, one at a time. In case both are free, he starts to eat for a predetermined time. When he finishes eating, he releases the *hashis* on the table so other philosophers may use them. If only one *hashi* is available, he holds it and keeps waiting for the release of the missing one.

This problem has features of Discrete Event Systems (DES), as it presents events and states that can be represented by discrete sets. The states of philosophers in this system are: i) philosophizing; ii) waiting and iii) eating. Based on the states, there are the following events: a) begin philosophizing; b) end philosophizing; c) begin waiting; d) end waiting; e) start eating; f) end eating.

B. *The Distribution Center*

The problem of the Distribution Center consists of a delivery logistics for requests according to the demand, which can form a queue. The freight for each order comes to the dock and is loaded into the truck by a team of four people [10].

The distribution process works accordingly to the following logical sequence: 1) requests arriving at the distribution center in time intervals t_1; 2) requests waiting in queue while the resources are not available; 3) each truck staying for a time t_2 on the dock while the loading process is performed; 4) truck leaving to make a delivery, while dock and boots are released for new loading; 5) request being transported to the destination during a period of time t_3; 6) truck returning to the distribution center in a time interval t_4 after the delivery. After step 6, the empty truck is available for new deliveries. In the model, values t_1, t_2, t_3, and t_4 are random values given by probability distributions that best represent each time.

The Distribution Center is a typical discrete event system, which features entities, queues, and resources. The entities are the requests, which are waiting in line for the availability of the resources: docks, trucks, and group of loaders. The set of discrete states concerning requests are: i) waiting in the queue; ii) being loaded; iii) being transported. Based on the states, it is determined how many resources are being used at every instant of time *t*. The events in this system are: a) receiving a new order; b) allocating a group of loaders to perform loading; c) starting to use the dock; d) allocating truck; e) starting to load truck; f) finishing to load truck; g) finishing to use dock; h) deallocating group of loaders; i) transporting request to the recipient and j) deallocating truck.

III. Complexity Metric

Several metrics to calculate the complexity have been developed based on the size of the system, entropy, information, cost, hierarchy, organization and other criteria [5]. In many cases, the complexity measure is dimensionless, so it only makes sense if compared to another measured value in the system itself or in a separate system, as long as the nature of the analyzed systems allows comparison [11].

Before applying the chosen complexity metric, the focus of analysis must be defined and a system model that objectively represents the various interactions of its parts must be created [10].

Based on Shannon's studies [12] about entropy in information exchange, some complexity metrics were developed [11] [13] [14] [15] [16]. Lemes [13] adapted Shannon modeling to measure the complexity of the system connections using (1):

$$\Gamma(s) = -\sum_{i,j=1}^{|s|} p(x)_{i,j} \log_2 p(x)_{i,j} \qquad (1)$$

where: $\Gamma(s)$ is the complexity of the system connections equivalent to the entropy in information exchange, s is the set of connections between elements of the system, $|s|$ is the total of connections between the elements of the system, $p(x)$, $\forall x \in s$, is the frequency in which the connections between elements i and j occur.

IV. Methodology

A. *Proposed Metric*

The proposed method is based on Lemes' modeling [13] in order to measure the complexity of real systems, but it ignores the exchange of information among its members. The proposed metric maps the active connections between entities, resources and queues at any given time t, expressed through the relationship matrix M. Thus, one can measure the static and dynamic complexity of any real system using (2) and (3).

$$\gamma(s) = -\sum_{i=1}^{\rho} p(c)_i \, . \, \log_2 p(c)_i \tag{2}$$

$$p(c)_i = \frac{1}{e \, .(n_r + n_q)} \tag{3}$$

where: $\gamma(s)$ is the system complexity, e is the number of entities that each resource can attend, n_r is the number of resources, n_q is the number of queues, $p(c)_i$ is the probability that the connection i occurs and ρ is the number of active connections at the moment t, expressed by (4).

$$\rho = \sum_{j=1}^{k} n_{cj} \, .n_{ej} \tag{4}$$

where: k is the number of entity states, n_{cj} is the number of active connections per entity at the jth state and n_{ej} is the number of entities at the j state.

B. Model for the Dining Chinese Philosophers Problem

The problem of the Dining Philosophers is simulated from a computational routine that implements the three proposed states in [17] for the philosophers: i) a_p philosophizing; ii) a_w waiting and iii) a_e eating. To perform the simulation, it is necessary to define 1) the number of philosophers n; 2) the philosopher P_i that starts the simulation, with $i = 1, 2, 3, ..., n$; 3) the orientation of the simulation (clockwise or counterclockwise); 4) the amount m of periods of time t that the philosophers can stay in state a_e; 5) the vector T_j consisted of periods of time t to the state a_e with $j = 1, 2, 3, ..., m$, which can be stochastically generated.

The philosopher who is in the state a_e uses two *hashis*. If the philosopher has just one *hashi*, he is waiting to eat, therefore he is in state a_w. In case the philosopher has no *hashi* with him, meaning that he is philosophizing, he in state a_p. Considering clockwise orientation, the philosopher P_i takes the *hashi* released from philosopher P_{i-1}, from his right, and the other *hashi* released from philosopher P_{i+1}, from his left, as explained in (5).

$$\begin{cases} P_{i-1} = P_n & if \quad i-1=0 \\ P_{i+1} = P_1 & if \quad i+1=n+1 \end{cases} \tag{5}$$

Each philosopher P_i stays in state a_e during the period of time given by T_j, in its j-th position, which corresponds to the j-th time that a philosopher assumes the state a_e. When the philosopher is finished eating, he mandatorily switches to state a_p and must wait, at least, a whole round through the table to come back to state a_e. The simulation ends when the vector T_j is totally gone through, with $j = m$, and when all philosophers come out of the state a_e. By the end of the simulation, there are the times t_{ae}, t_{aw} and t_{ap} during which the entity P_i kept in the states a_e, a_w and a_p, respectively. The total time of simulation $T(P_i)$ is the sum of times t_{ae}, t_{aw} and t_{ap} for the entity P_i.

Two approaches are considered in the model of the Dining Chinese Philosophers: 1) the philosophers always want to eat and 2) the philosophers do not always want to eat. In the first one, the resources are required at all times, unlike the second approach, in which philosophers may decide not to eat even if the resources (*hashis*) are available. The simulation clock varies in a time unit, *[ut]*, one by one, corresponding to one round. At the beginning of each round, the philosophers can change their status according to the rules of the simulation.

In order to apply the complexity metric to the Dining Chinese Philosophers, it is considered that the number of active connections ρ in this system in a given moment is provided by (6), where: n_{ap}, n_{aw} and n_{ae} are the number of philosophers in a_p, a_e and a_w, respectively. When the philosopher is eating, two connections are added to the system, (one for *hashi* used); when the philosopher is waiting, one connection is added (with *hashi* that was already allocated); and when the philosopher is just philosophizing, they do not add any connection to the system. Hence, (4) assumes the following configuration:

$$\rho = 2 \, .n_{ae} + 1 \, .n_{aw} + 0 \, .n_{ap} \tag{6}$$

The probability of occurrence for a connection in this system is obtained by applying (3), in which the number of entities e that each resource can attend is equal to two, as each *hashi* attends up to two different philosophers, and n_r corresponds to the number of resources (*hashis*). Since this is a closed system, the number of queues n_q is not taken into consideration.

The calculation of complexity is performed based on the active connection at each instant of simulation. In the Dining Chinese Philosophers problem, it is possible to have the configuration represented by the relationship matrix M at some instant t expressed in (7), in which the columns represent the philosophers from P_1 to P_5 and the lines represent the *hashis* from H_1 to H_5. In this matrix, the philosophers P_1 and P_4 are philosophizing, P_3 is waiting to eat and P_2 and P_5 are eating.

$$M = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

C. Model for the Distribution Center Problem

Different performance measures can be chosen for the problem of the Distribution Center, e.g. the average waiting time in queue, the average time for transportation, or the percentage of use of the system resources. It is important to note that unlike the Dining Chinese Philosophers problem, the distribution center is an open system because new entities can be integrated into the system during operation. Thus, the number of entities (requests) and therefore, the demand for resources (docks, trucks and loaders) can vary over time.

For this case, the delivery time T_d of requests is considered as a performance measure, which is the time taken from the arrival of the request at the Distribution Center to the moment it is delivered to the recipient. This time is composed of the sum of the times i) waiting in queue until it gets carried away t_{rq}, ii) loading the request into the truck t_{rl} and iii) transporting the request from the Distribution Center to the recipient t_{rt}, as in (8):

$$T_d = t_{rq} + t_{rl} + t_{rt} \tag{8}$$

The measure for complexity is calculated by (2). The active connections ρ are mapped based in the states p_q, p_l and p_t, which correspond to the queued requests, in loading process and being transported, respectively. The number of active connections is determined by (4), being that each request in state p_q adds one connection to the system (with the request ahead of it), each request in state p_l, adds three connections (one with the dock, one with the truck and another with the loaders group) and finally a request in state p_t contributes with one connection to the system (with the truck). Thus, (4) results in (9).

$$\rho = 1 \,.\, n_{rq} + 3 \,.\, n_{rl} + 1 \,.\, n_{rt} \tag{9}$$

where: n_{rq}, n_{rl} and n_{rt} are the number of requests in queue, being loaded and being transported, respectively.

The probability of occurrence for a connection in the Distribution Center problem in given by (3), in which the number of entities e that each resource can attend is equal to the total number of requests in the system at the instant t, considering that each entity (request) can be attended by any system resource. Therefore, e is the sum of n_{rq}, n_{rl} and n_{rt}.

In this model, n_r is equal to the sum of the number of docks, trucks and loaders groups. The value of n_q corresponds to the number of queues adopted in the system modeling, which is in this case, one queue. Considering a configuration of *6* requests with *2* docks, *3* trucks and *2* groups of loaders, it is possible at some instant t of this system, to set up the following relationship matrix M expressed in (10), in which the columns represent the requests, from R_1 to R_6, and the lines represent the queue (Q), docks (D_1 and D_2), trucks (from T_1 to T_3) and loaders group (G_1 and G_2), in this order.

$$M = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

In (10), it is verified that the requests R_1 and R_2 are being transported in trucks T_2 and T_3, respectively, the request R_3 is being carried by the loaders group G_1 into the truck T_1 parked in the dock D_2 and the requests R_4, R_5 and R_6 are waiting in line. It is noteworthy that the number of active connections can be computed from the matrix M

V. Results

A. Study Case 1: Dining Philosophers

The static and dynamic complexities were calculated from the model presented in Section IV-B. For calculating the static complexity, the maximum number of active connections was considered, which is only possible in closed systems. This is the case of the Dining Philosophers, where new entities cannot be added during its operation. In this model, the maximum number of active connections ρ is equal to the number of resources.

In order to obtain the dynamic complexity $\gamma_d(p)$ during the simulation of the system, (2) was used at each event occurrence, which alters the state of the system. Therefore, the configuration of the system is considered at each instant of time t.

For this problem, two policies were used to express the behavior of philosophers concerning their desire to eat. In the first one, philosophers want to eat whenever possible. The second approach considers that each philosopher may or may not want to eat at each round.

The policy in which the philosophers always wants to eat was presented in [6], where it was observed that at every instant of time t, the dynamic complexity was maximum, equal to the static complexity $\gamma_s(p)$. Thus, (11) is obtained.

$$\gamma_d(p) = \gamma_s(p) : \forall t \in T_j \tag{11}$$

In this approach, it is noted that the philosophers (entities) always use all *hashis* (resources) at every round. However, in most real systems, the resources are not fully utilized at all times, varying the number of active connections in time as the system evolves.

For the case in which philosophers can decide whether or not to eat, a new simulation was performed because the number of connections varied. All philosophers started in the state a_p, philosophizing, so all the *hashis* were free. The simulation clock forwarded one unit *ut* at a time, which corresponds to a round when each philosopher has its state assessed.

The simulation was performed during *200 ut* for the scenario with five philosophers and five *hashis*. At each change of state of the system, (2) was used to assess the dynamic complexity, observing the number of active connections. In Fig. 2, it is noted the behavior of the system relative to the static complexity $\gamma_s(p)$, dynamic $\gamma_d(p)$ and average $\gamma_a(p)$.

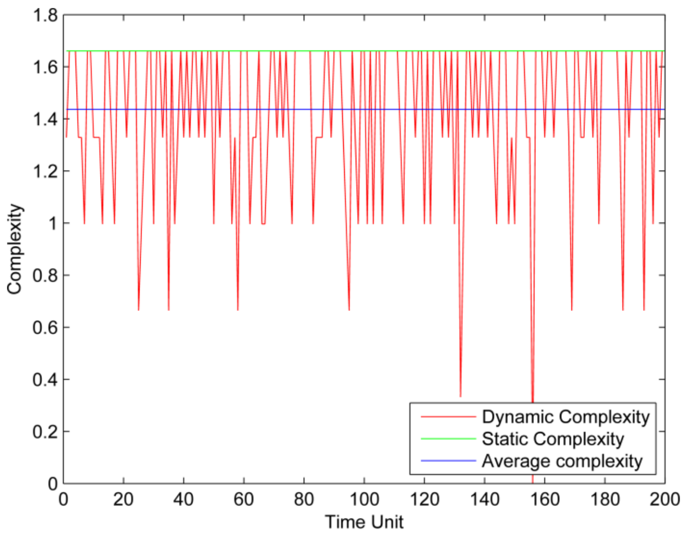

Fig. 2. Static, dynamic and average complexities in the Dining Chinese Philosophers problem.

In Fig. 2, static and average complexities were approximately *1.66* and *1.42*, respectively. The dynamic complexity varied during the simulation process, reaching the maximum value in few instants. As each philosopher could choose to eat or not, the number of active connections in the system changed every round. It was also noted also that dynamic complexity has a ceiling limit defined by the value of the static complexity. Another observation can be made regarding the mode values, being approximately: *0.38*; *0.68*; *1* and *1.35*. The repeated values are repeated are an indicative of the number of resources used.

B. Study Case 2: Distribution Center

The model of the Distribution Center, presented in Section IV-C, was simulated based on the following operation dynamics: 1) the arrival of the requests happens in interval of time t_1 with mean average of *120* minutes, according to an exponential distribution; 2) the loading process of each truck lasts a certain time interval t_2, following a normal distribution with mean value of *100* minutes and standard deviation of *30* minutes; 3) when a truck leaves to delivery, the dock and the loaders group are released to a new loading process; 4) the transportation of each request to the recipient is performed in a period of time t_3, evenly distributed from *120* to *240* minutes; 5) after delivery, the empty truck takes an amount of time t_4 to return to the Distribution Center, and t_4 follows the same probabilistic distribution of t_3.

The number of docks used in the simulation ranges from *1* to *10*, the amount of trucks, ranges *1* to *15,* and the number of loaders groups ranges *1* to *10.* Therefore, there are *1,500* different scenarios, obtained by combining the amounts of resource values. The simulation was performed for *180* days for each scenario, considering *24* daily hours of operation and that every truck is loaded with only one request at a time. In the *1,500* simulated scenarios, the delivery time T_d in minutes and the complexity $\gamma(d)$ of the system were calculated.

The normalized values of T_d and $\gamma(d)$ for all simulation scenarios are presented in Fig. 3. As the simulation is the combination of resources, docks, trucks and loaders groups, in Fig. 3 the peaks of T_d (in blue) correspond to the combinations in which they had only one truck. At every change in the number of docks, a truck was used for *10* scenarios, leading to higher values of T_d and $\gamma(d)$ (in red). The oscillations that occur between the values *0.1* and *0.3* of $\gamma(d)$ are derived from changes in the number of trucks, indicating greater sensitivity in this system parameter.

Table I has some of the shortest delivery time values T_d obtained in the simulation. By analyzing Fig. 3, it is possible to see that for values of T_d close to *zero*, there are several system configurations that result in the same value of T_d. However, Table I presents the five scenarios with the lowest complexity values found for the five smallest times T_d.

TABLE I
SCENARIOS WITH SMALLEST DELIVERY TIME AND CORRESPONDENT COMPLEXITY

T_d	$\gamma(d)$	Docks	Trucks	L. Groups
276.9357	0.2963	10	15	6
276.9569	0.3038	6	14	10
277.0154	0.2697	10	15	10
277.0368	0.2821	9	14	10
277.0613	0.3041	5	15	10

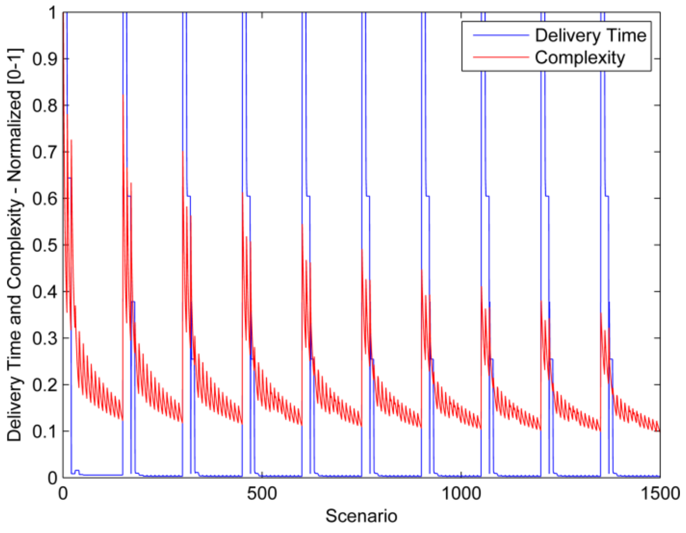

Fig. 3. Relationship between delivery time and complexity.

The smallest complexity that was found in the *1500* scenarios is $\gamma(d) = 0.2697$, which corresponds to the third smallest delivery time and to the scenario with the maximum number of available resources, as disposed in Table I. The worst delivery time T_d was *340* times bigger than the smallest time occurred in *100* different scenarios. That is illustrated in the peaks of Fig. 3, where each peak contains *10* scenarios. The biggest complexity was $\gamma(d) = 2.7906$, which relates to the worse delivery time since the calculation for both the complexity and delivery time considers the permanence of the requests in line. As each request in queue corresponds to one connection, the longer the queue, the bigger will be the time T_d and the complexity $\gamma(d)$.

It was noted that the value $\gamma(d)$ is intrinsically related to the configuration of the system. By putting the results from the *1500* scenarios in descendant order of $\gamma(d)$, among the five highest values, there is $T_d \approx 815$. This represents a value close to *3* times the smallest T_d that was found, as displayed in Table II. It is verified that for T_d = 815.7777, approximately *84%* fewer resources were used when compared to the scenario of smallest T_d.

Considering R_C in (12) and that $\gamma(d)$ is calculated from the connections, R_C contains the relationship between time and the configuration of the system.

$$R_C = \frac{T_d}{\gamma_{(d)}} \tag{12}$$

TABLE II
RELATIONSHIP BETWEEN THE DELIVERY TIME AND THE COMPLEXITY

T_d	$\gamma(d)$	R_C	Docks	Trucks	L. Groups
276.9357	0.2963	934.6463	10	15	6
291.6034	0.4083	714.1128	2	11	7
297.8970	0.5892	505.5827	3	7	2
322.1149	0.4690	686.8121	2	6	9
815.7777	2.0263	402.5947	1	3	1

Table II displays the values found for R_C. It is noted that the lowest value refers to the time of *815.7777*, whose scenario presented a reduced amount of resources. Even so, the efficiency of the system was *115* times greater than the worst case and only *3* times smaller than in the best case. This shows that the use of resources was optimized, as the ratio value R_C indicates the lowest cost in terms of complexity for each minute of permanence of the request in the system.

High complexity values may reflect significant sizes of the queue if the delivery time is high. However, high complexity can also be indicative of optimal settings for the system. Therefore, the number of active connections observed in the calculation of complexity can express both the queue formation and the use of resources for loading and transportation requests. Thus, it becomes necessary to use R_C to check if the high value of complexity is indicative of queue or of full operation, as described in (13):

$$\begin{cases} \gamma \uparrow & T_d \uparrow \quad \textit{queue} \\ \gamma \uparrow & T_d \downarrow \quad \textit{operation} \end{cases} \tag{13}$$

C. *Sensitivity Analysis of Distribution Center*

This case study aims to confirm the hypothesis that the *truck* parameter has the largest sensitivity. From the simulated results, the following input parameters were analyzed: *dock, truck, and loaders group.* Delivery time and complexity were considered as output parameters.

The following values were considered: 5 docks, 7 trucks and 5 loaders groups. These values were chosen because they correspond to the average points of the following resource ranges: 1 to 10 docks, 1 to 15 trucks and 1 to 10 groups of loaders.

In Fig. 4 and Fig. 5, each parameter was varied from its base value of -100% to 100%, according to a univariate analysis. Fig. 4 represents the relation between the amount of resources and delivery time, and Fig. 5 represents the relation between resources and complexity.

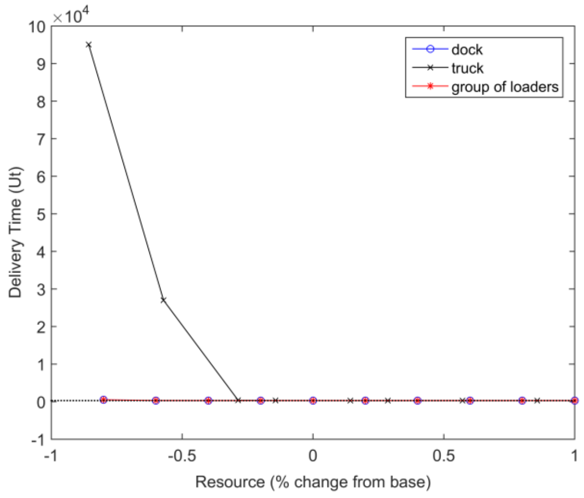

Fig. 4. Changes in multiple parameters (resources) for a single output variable (time delivery).

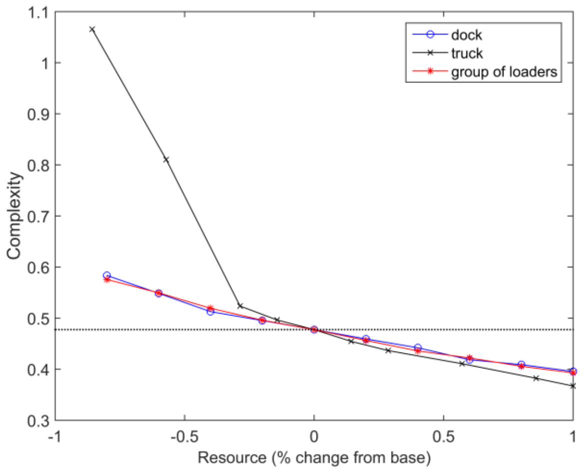

Fig. 5. Changes in multiple parameters (resources) for a single output variable (complexity).

It can be observed that the *truck* parameter has the largest sensitivity in the Distribution Center, especially in the interval between -85.7% to 28.6% of its base value. Fig. 5 shows that the curve related to the *truck* resource has the largest distance from the base (dotted) line, which indicates the largest sensitivity in the analyzed scenarios.

This analysis reflects the fact that the *truck* has a larger demand for resources, because it is utilized during the states where the order is being i) loaded and ii) transported, differently from other resources that are requested only when the order is being loaded.

VI. Conclusion

This work presented a methodology for measuring the static and dynamic complexity of systems. The presented metric can be applied in real systems, provided that they are modeled in terms of events and states that express the existing connections. Static complexity was calculated only in the Dining Philosophers problem because it is a closed system. This measurement indicated the maximum complexity of the system, upwardly limiting the dynamic complexity. In the Distribution Center problem, it was found that the complexity associated with the performance measure provides knowledge about the system. The lowest value of the relationship between delivery time and complexity was in a system configuration that showed high complexity, although the demand was met with fewer resources. The *truck* resource presented the largest sensitivity, causing the highest impact on the system when its quantity is limited. The presented complexity metric can support decision-making policies related to resource management, optimization processes (as a constraint or goal), security policies, or appreciation of systems.

References

[1] E. Rechtin and M. W. Maier, *The art of systems architecting*. CRC Press, 2010.

[2] T. Per Bak and K. Wiesenfeld, "Self-organized criticality: and explanation of 1/f noise," *Phys. Rev. Let*, vol. 59, pp. 381–384, 1987.

[3] J. H. Holland, *Complexity: A very short introduction*. Oxford University Press, 2014.

[4] H. A. Simon, "The architecture of complexity," *Proceedings of the American Philosophical Society*, vol. 106, no. 6, pp. 467–482, 1962.

[5] S. Lloyd, "Measures of complexity: a nonexhaustive list," *IEEE Control Systems Magazine*, vol. 21, no. 4, pp. 7–8, 2001.

[6] L. Santos, C. Silva, J. Paiva, V. Gomes, S. Oliveira, A. Alves, and W. Calixto, "A methodology for calculation of complexity in systems: Case study," in *IEEE Congreso Chileno de Ingeniería Eléctrica, Electrônica, Tecnologías de la Informacíon y Comunicaciones*, pp. 213–218, 2015.

[7] C. G. Cassandras, S. Lafortune. *Introduction to discrete event systems*. Secaucus, NJ: Springer Science & Business Media, 2008.

[8] A. Saltelli, et al. *Global Sensitivity Analysis*. England: John Wiley & Sons, Ltd, 2008.

[9] E. W. Dijkstra, "Hierarchical ordering of sequential processes," *Acta Informatica*, vol. 1, no. 2, pp. 115–138, 1971.

[10] L. Chwif and A. C. Medina, *Modeling and simulation of discrete events (in portuguese)*. Campus-Elsevier, 4 ed., 2014.

[11] D. W. Repperger, R. G. Roberts, and C. G. Koepke, "Quantitative measurements of system complexity," US Patent 8,244,503 B1, Aug. 8, 2012.

[12] C. E. Shannon, "A mathematical theory of communication" *The Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[13] M. J. R. Lemes, "Complexity, coupling and criticality (C2A) as risk factors in system design (in portuguese)," PhD thesis, University of São Paulo, 2012.

[14] C. Gershenson and N. Fernández, "Complexity and information: Measuring emergence, self-organization, and homeostasis at multiple scales," *Complexity*, vol. 18, no. 2, pp. 29–44, 2012.

[15] C. Gershenson, "Harnessing the complexity of education with information technology," *Complexity*, vol. 20, no. 5, pp. 13–16, 2015.

[16] C. E. Maldonado, G. Cruz, and A. Nelson, "Biological hypercomputation: A new research problem in complexity theory," *Complexity*, vol. 20, no. 4, pp. 8–18, 2015.

[17] A. S. Tanenbaum and H. Bos, *Modern operating systems*. Prentice Hall Press, 2014.