Modeling of the Consensus in the Allocation of Resources in Distributed Systems

Federico Agostini[1], David L. La Red Martínez[2], Julio C. Acosta[3]
Faculty of Exact and Natural Sciences and Surveying, North-eastern National University
Corrientes, Argentina

***Abstract*—When it comes to processes distributed in process nodes that access critical resources shared in the modality of distributed mutual exclusion, it is important to know how these are managed and the order in which the demand for resources is resolved by the processes. Being in a shared environment, it is necessary to comply with certain rules, for instance, access to resources must be achieved through mutual exclusion. In this work, through an aggregation operator, a consensus mechanism is proposed to establish the order of allocation of resources to the processes. The consensus is understood as the agreement that must be achieved for the allocation of all the resources requested by each process. To model this consensus, it must be taken into account that the processes can form group of processes or be independent, the state of the nodes where each of them is located, the computational load, the number of processes, the priorities of the processes, CPU usage, use of main memory, virtual memory, etc. These characteristics allow the evaluation of the conditions to agree on the order in which allocations of resources to processes will be made.**

Keywords*—Aggregation operators; communication between groups of processes; mutual exclusion; operating systems; processor scheduling

I. Introduction

The proliferation of computer systems, many of them distributed in different nodes with multiple processes that cooperate for the achievement of a particular function, requires decision models that allows groups of processes to use shared resources that can only be accessed in the modality of mutual exclusion.

The traditional solutions for this problem are found in [1] and [2], both papers describe the main synchronization algorithms in distributed systems. The author in [3] presents an efficient and fault tolerant solution for the problem of distributed mutual exclusion. The authors in [4], [5] and [6] present algorithms to manage the mutual exclusion in computer networks. In [7] are detailed the main algorithms for distributed processes management, distributed global states and distributed mutual exclusion.

The allocation of resources in processes should be performed taking into account the priorities of the processes and also the state in terms of workload of the computational nodes in which the processes are executed.

Also, solutions (which may be considered traditional) have been proposed for different types of distributed systems in [8], [9], [10], [11] and [12]. Other works that focused on ensuring mutual exclusion have been presented in [13] and [14]. An interesting distributed solution based on permissions is presented in [15] and a solution based on process priorities can be found in [16].

In this paper, a new aggregation operator will be presented specifically for solving the aforementioned problem. This falls under the category of OWA (Ordered Weighted Averaging) operators, more specifically Neat OWA. The use of aggregation operators in decision models has been widely studied. For example, [17], develops methodologies that solve problems in the presence of multiple attributes and criteria and in [18] the way to obtain a priority vector is collectively studied, which is created from different formats of expression of the preferences of decision makers. The model can reduce the complexity of decision-making and avoid the loss of information when the different formats are transformed into a single format of expression of preferences. In addition, [19] presents the main mathematical properties and behavioural measures related to the aggregation operators. A review of aggregation operators, especially those of the OWA family, is presented in [20], [21] and [22]. OWA operators applied to multicriteria decision making are presented and analysed in [23], and [24] analyse the OWA operators and their applications in the decision making process. In turn, in [25] a complex and dynamic problem of group decision making with multiple attributes is defined and a resolution method is proposed, which uses a consensus process for groups of attributes, alternatives and preferences, resulting in a decision model for problems of the real world.

This study will present a variant of an innovative method for the management of shared resources in distributed systems, based on [26] and [27], in which an aggregation operator is developed to assign resources in distributed systems. Here, we establish a consensus model that favours the sequential access of the processes to all the requested resources. The premises, data structures and the operator mentioned in [26] and [27], are used as a starting point to create a new operator in the scenario described next.

This paper, which presents an innovative method for the management of shared resources in distributed systems is structured as follows: Section 2 explains the data structures that the proposed operator will use, Section 3 describes the aggregation operator, in Section 4 a detailed example of this is shown, then the Conclusions and the Future lines of work are presented, and then the Acknowledgments, the References and the appendix are shown.

II. Data Structures to be Used

The proposed scenario considers the following conditions: In first place, the processes must have access to shared resources in the mutual exclusion modality. In second place, they must be able to form groups of processes (independent processes are considered as unitary groups). In third place, the processes must not require synchronization (that is, to be active in their respective processors at the same time) and they must have strict consensus requirements in order to gain access to the resources (an agreement is required in order to consecutively allocate the resources requested by a process, that is, once the resources allocation sequence is started, it cannot be interrupted to allocate resources to other processes, until the active process releases the resources).

These are groups of processes that are distributed in process nodes that access critical resources. These resources are shared in the form of distributed mutual exclusion and it must be decided, according to the demand for resources by the processes, what the priorities to allocate the resources to the processes that require them will be (only the resources that are available to be assigned in the processes will be taken into account, that is, those not yet allocated in certain processes).

- The access permission to the shared resources of a node will not only depend on whether the nodes are using them or not, but on the aggregation value of the preferences (priorities) of the different nodes regarding granting access to shared resources (alternatives) as well.
- The opinions (priorities) of the different nodes regarding granting access to shared resources (alternatives) will depend on the consideration of the value of variables that represent the state of each one of the different nodes. Each node must express its priorities for assigning the different shared resources according to the resource requirements of each process (which may be part of a group of processes).

These available shared resources hosted on different nodes of the distributed system may be required by the processes (clustered or independent) running on the nodes.

Possible states of each process:

- Independent process.
- Process belonging to a group of processes.
- Possible state of each one of the nodes:
- Number of processes.
- Priorities of the processes.
- CPU usage.
- Main memory usage.
- Use of virtual memory.
- Additional memory required for each resource requested by each process (depending on the availability of the data).
- Additional estimated processor load required for each resource requested by each process (depending on data availability).
- Additional estimated input / output load required for each resource requested by each process (depending on data availability).
- Status of each one of the shared resources in the distributed mutual exclusion mode in the node:
- Assigned to a local or remote process.
- Available.
- Predisposition (nodal priority) to grant access to each of the r shared resources in the mode of distributed mutual exclusion (will result from the consideration of the variables representing the node status, the priority of the processes and the additional computational load, which would mean allocating the resource to the requesting process).
- Current load of the node, which can be calculated as the average CPU, memory and input / output usage percentages at any given time (these load indicators may vary depending on the case, some may be added or changed); the current load categories, for example, High, Medium and Low, should also be defined, with value ranges for each category being indicated.

The scenario proposed in this study considers resources and processes in distributed operating systems, applied to the telecommunications environment, but without being limited to any specific communications protocol, meaning that it is a generic scheme. It is considered that the application of the proposed method would result in an increase in the traffic of control information, but the overall performance of the system would improve by allocating resources to the processes according to a holistic and cognitive decision-making scheme that also guarantees mutual exclusion in access to shared resources.

Fig. 1 shows the resources requests by the processes, the resources already assigned and the nodes in which they are located.

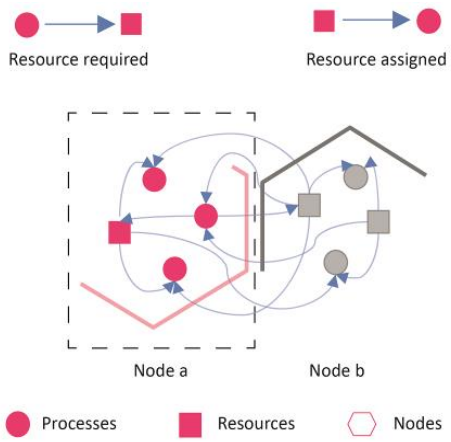

Fig. 1. Resources and Processes at Nodes in Distributed Systems.

III. Description of the Aggregation Operator

The proposed operator consists of the following steps:

1) Calculation of the current computational load of the nodes.

2) Establishment of the categories of computational load and the vectors of weights associated with them.

3) Calculation of the priorities or preferences of the processes considering the state of the node (in each node for each process).

4) Calculation of the priorities or preferences of the processes to access the available shared resources. (calculated in the centralized manager of shared resources) and determination of the allocation order and to which process the resources will be allocated.

Each one of the steps of Fig. 2 is described in [26] and [27].

In Fig. 2, there is a list of the necessary steps to obtain the final global priorities to assign the resources (*DSAF*, Distributed Systems Assignment Function).

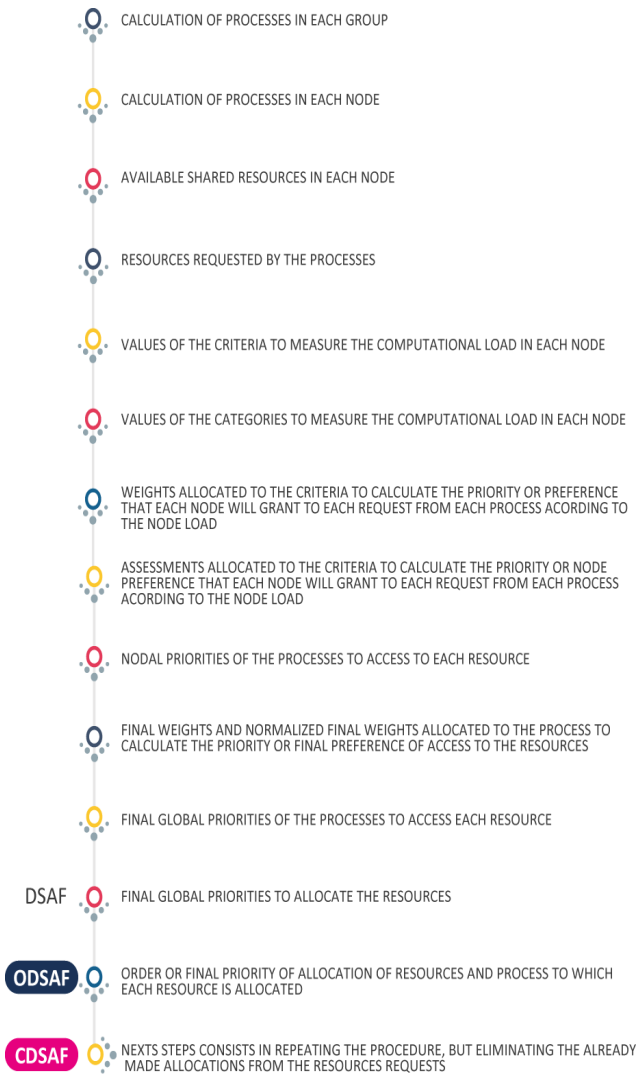

Fig. 2. Steps to Obtain the *DSAF*, *ODSAF* and *CDSAF* Functions.

TABLE I. Concatenation of the Ordered Assignment Tables (***ODSAF***) of Each One of the Iterations Corresponding to the General Method

ODSAF	***Iterations***
1st iteration	Rows from 1 to *n* *n*= number of rows of the ***ODSAF*** first iteration
2nd iteration	Rows from *n*+1 to *m* *m* = number of rows of the ***ODSAF*** second iteration
last iteration	*m* = number of rows of the ***ODSAF*** second iteration

The order or priority of allocation of the resources and the process to which each resource is assigned (*ODSAF*, Ordered Distributed System Assignment Function) can be seen in Table 1.

The last step is to repeat the procedure but removing the already made allocations from the resources requests (*CDSAF*, Concatenated Distribution Systems Assignment Function), as shown in Fig. 3.

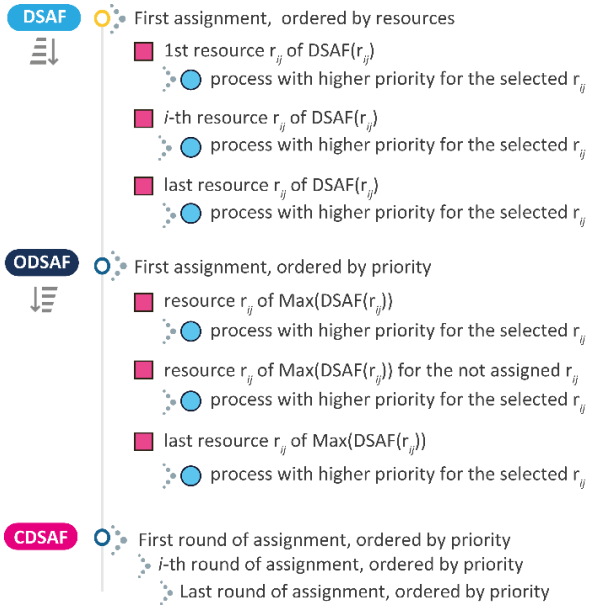

Fig. 3. Steps to Obtain the *DSAF*, *ODSAF* and *CDSAF* with their Corresponding Iterations.

The *CDSAF* table is obtained from the concatenation of the *ODSAF* tables of each iteration, as shown in Table 1.

a) Final global priority of the process

Once the *CDSAF* table is completed (Table 1), the final global priorities of the processes will be calculated in order to access all of its resources, and the order in which each one will be allocated will be established, receiving all the requested resources. For this, the *CDSAF* table will be considered, the priorities of all the resource/process assignments will be added for each process, and they will be divided by the number of assignments of that process. The process with the higher final global priority will be the first one to get the requested resources. This constitutes what will be called the Final Global Priority of the Process (*FGPP*), as shown in Fig. 4.

$$FGPP_i = i = 1,\ldots,h = \frac{\sum \boldsymbol{CDSAF}\, j}{\text{Number of global assignments of the } i \text{ process}}$$

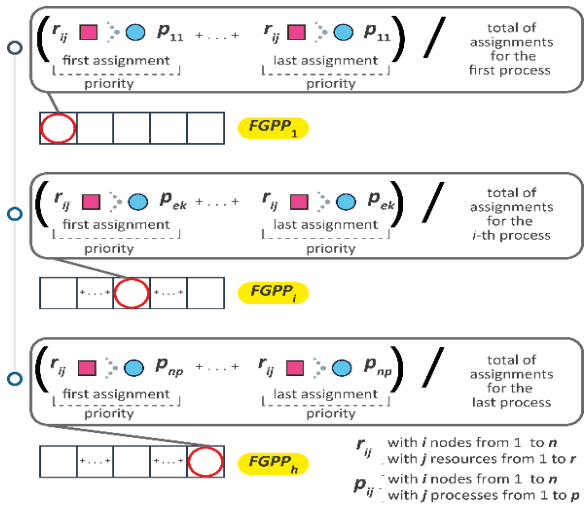

Fig. 4. Calculation of the *FGPP* of each process.

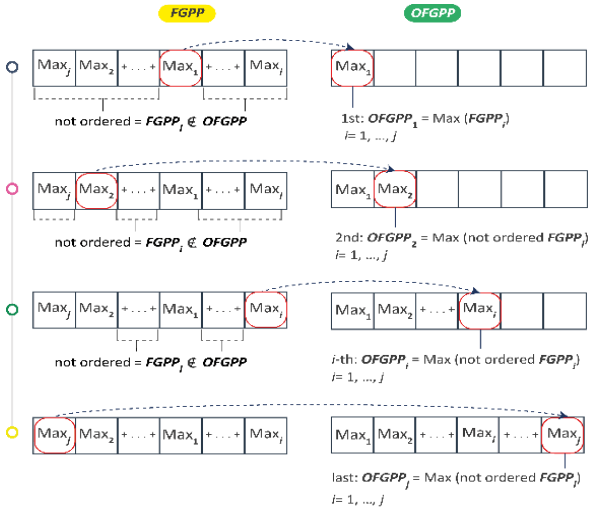

Fig. 5. An example of the calculation of the OFGPP of each process.

h= total number of processes in the system (summation of processes of the nodes); *j*=number of resources allocated to the *i* process.

The elements of the *FGPP* vector must be ordered from highest to lowest to obtain the global priority order of allocation of resources to processes, as shown in Fig. 5.

Ordered Final Global Priority of the Process (*OFGPP*)

j= cardinality of *FGPP* (number of processes in the system)

OFGPPi= Max (not ordered *FGPPi*) *i*= 1, …, *j*

not ordered = *FGPPi* ∉ *OFGPP*

1st: $OFGPP_1$ = Max (*FGPPi*) *i*= 1, …, *j*

2nd: $OFGPP_2$ = Max (not ordered *FGPPi*) *i*= 1, …, *j*

last: OFGPPj = Max (not ordered FGPPi) i= 1, …, j

b) Ordered concatenated distributed system assignment function (ocdsaf)

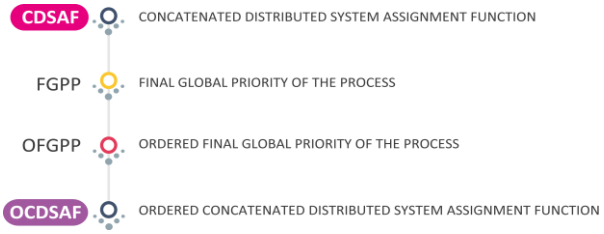

Fig. 6. Steps to go from the *CDSAF* to the *OCDSAF*.

The *OCDSAF* will establish the order of the final global priority allocation of processes to access its resources, and the order in which each one will be allocated, getting all the requested resources. For this, the *CDSAF* and *OFGPP* tables will be considered, as shown in Fig. 6.

The cardinalities (number of allocation of resources to each process) obtained from each one of the processes of the *OFGPP* vector in the *CDASF* table will be calculated.

CP_i = process cardinality ($OFGPP_i$) in *CDSAF*.

Then, each one of the allocations of resources to processes in the *CDSAF* table of each one of the *OFGPP* vector processes will be obtained. The total number of allocations for each process will be determined by the cardinality calculated in the previous step, as shown in Fig. 7.

In Fig 7, the first step is to calculate the priority of the process $\boldsymbol{p_{ek}}$, considering all rounds at *CDSAF*. The second step is to obtain the position in the *OFGPP* vector according to the calculated priority. The third step is to find all the assignments of the $\boldsymbol{p_{ek}}$ process in the *CDSAF* and place them in the *OCDSAF* in the order in which the p_{ek} process appears in the *OFGPP*. The representation of resources r_{ij} indicate the resources (whose first sub-index represents the node where it is and the second sub-index represents the resource number itself) that are assigned to the $\boldsymbol{p_{ek}}$ process (whose first sub-index represents the node where it is and the second sub-index represents the process number itself) in each round. Although the resources have the same sub-indexes, they are not necessarily the same resources, but they can represent different resources that are assigned several times in the different rounds, but always to the same $\boldsymbol{p_{ek}}$ process. The location in the *FASDCO* table will depend on the location in the *PGFPO* vector.

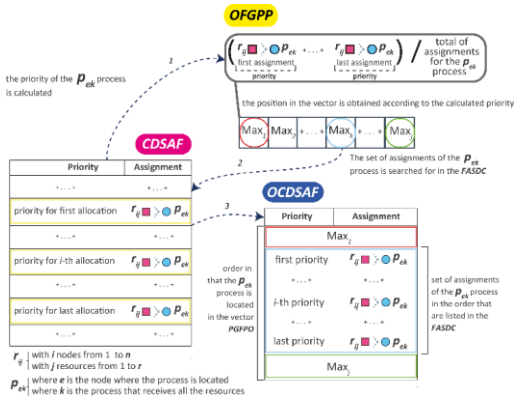

Fig. 7. Calculation of Priorities for the $\boldsymbol{P_{ek}}$ Process with the Highest Priority in *PGFPO*.

$OCDSAF_1$ = first allocation of the *CDSAF* for the ($OFGPP_1$) process

$OCDSAF_{cp1}$ = last allocation of the *CDSAF* for the ($OFGPP_1$) process

$OCDSAF_{cp1+1}$ = first allocation of the *CDSAF* for the ($OFGPP_2$) process

$OCDSAF_{cp1+cp2}$ = last allocation of the *CDSAF* for the ($OFGPP_2$) process

$OCDSAF_{cp1+cp2+\ldots+cpk-1+1}$ = first allocation of the *CDSAF* for the ($OFGPP_k$) process

$OCDSAF_{cp1+cp2+\ldots+cpk}$ = last allocation of the *CDSAF* for the ($OFGPP_k$) process

$OCDSAF_{cp1+cp2+\ldots+cpn-1+1}$ = first allocation of the *CDSAF* for the ($OFGPP_n$) process

$OCDSAF_{cp1+cp2+\ldots+cpn}$ = last allocation of the *CDSAF* for the ($OFGPP_n$) *process*

c) Considerations for aggregation operations

The characteristics of the aggregation operations described allow to consider that the proposed method belongs to the family of aggregation operators Neat-OWA, which are characterized as follows:

The definition of OWA operators indicates

$$f(a_1, a_2, \ldots, a_n) = \sum_{j=1}^{n} w_j \cdot b_j \tag{1}$$

Where b_j is the j_{th} highest value of the a_n, with the restriction for weights to satisfy

$$w_i \in [0,1] \tag{2}$$

$$\sum_{i=1}^{n} w_i = 1 \tag{3}$$

For the Neat OWA operator family, the weights will be calculated according to the elements that are added, or more exactly to the values to be orderly added, the b_j, maintaining conditions (2) and (3). In this case the weights are $w_i=f_i(b_1,\ldots,b_n)$, defining the operator:

$$F(a_1,\ldots a_n) = \sum_i f_i(b_1,\ldots,b_n)\cdot b_i \tag{4}$$

This family, in which the weights depend on the aggregation, do not require to meet all properties of OWA operators.

In addition, in order to be able to assert that an aggregation operator is *neat*, the final aggregation value needs to be independent of the order of the values. $\boldsymbol{A}=(a_1,\ldots,a_n)$ being the entries to add, $\boldsymbol{B}=(b_1,\ldots,b_n)$ being the ordered entries and $\boldsymbol{C}=(c_1,\ldots,c_n)= Perm(a_1,\ldots,a_n)$ being a permutation of the entries. An OWA operator is defined as *neat* if

It produces the same result for any assignment $C = B$

$$F(a_1, a_2, \ldots, a_n) = \sum_{i=1}^{n} w_i \cdot b_i \tag{5}$$

One of the characteristics to be pointed out by Neat OWA operators is that the values to be added do not need to be sorted out for their process. This implies that the formulation of a neat operator can be defined by the arguments instead of the orderly elements.

In the proposed aggregation operator, the weights are calculated according to context values. From this context, arise the values to be aggregated.

IV. Example and Discussion of Results

This section will explain in detail an example of application of the proposed aggregation operator. This example takes as a starting point the ordered *DSAF* vector from [26] and [27], and these steps are shown in Fig. 2.

The example seen in [26] shows the following calculations:

- The priorities or preferences of the processes to access the available shared resources.
- The vector of final weights that will be used in the final aggregation process to determine the order or priority of access to the resources.

The greatest of these products made for the different processes in relation to the same resource, will indicate which one of the processes will get access to the resource.

The summation of all these products in relation to the same resource will indicate the priority that said resource will have in order to be assigned. This constitutes the Distributed System Assignment Function (*DSAF*) that can be seen in Table 2.

The final order of allocation of the resources and the recipient processes is obtained by ordering Table 2, as shown in Table 3.

TABLE II. Final Global Priorities for Allocating the Resources (*DSAF*) in the First Iteration

Resources	***Priority***	***Assignment***
r_{11}	0.35120968	r_{11} to p_{37}
r_{12}	0.47306452	r_{12} to p_{37}
r_{13}	0.32862903	r_{13} to p_{13}
r_{21}	0.33000000	r_{21} to p_{37}
r_{22}	0.34403226	r_{22} to p_{34}
r_{23}	0.24919355	r_{23} to p_{11}
r_{24}	0.18951613	r_{24} to p_{34}
r_{31}	0.37048387	r_{31} to p_{34}
r_{32}	0.30322581	r_{32} to p_{34}
r_{33}	0.46798387	r_{33} to p_{23}

TABLE III. ORDER OR FINAL PRIORITY OF ASSIGNMENT OF RESOURCES AND PROCESS TO WHICH IS ALLOCATED EACH RESOURCE (***ODSAF***) IN THE FIRST ITERATION

Ordered Final Global Priority	***Assignment***
0.47306452	r_{12} to p_{37}
0.46798387	r_{33} to p_{23}
0.37048387	r_{31} to p_{34}
0.35120968	r_{11} to p_{37}
0.34403226	r_{22} to p_{34}
0.33000000	r_{21} to p_{37}
0.32862903	r_{13} to p_{13}
0.30322581	r_{32} to p_{34}
0.24919355	r_{23} to p_{11}
0.18951613	r_{24} to p_{34}

The next step is to repeat the procedure, but removing the requests of already made allocations; it must be noted that the assigned resources will be available once they are released by the processes, and can therefore be allocated to other processes.

In this way, all the resources requests from all the processes will be answered, considering mutual exclusion and priorities of the processes, nodal priorities and final priorities, according to the scenario presented in [26] and [27].

The scenario presented next, starts from the concatenation of the ordered assignment of each one of the iterations corresponding to the above mentioned scenario.

The *CDSAF* table will be obtained from the concatenation of the *ODSAF* table of each iteration, as shown in Table 4.

TABLE IV. ORDER OR FINAL PRIORITY OF ASSIGNMENT OF RESOURCES AND PROCESS TO WHICH IS ALLOCATED EACH RESOURCE IN ALL ITERATIONS (*CDSAF*)

Ordered Final Priority	Assignment	Round
0.47306452	r_{12} al p_{37}	1
0.46798387	r_{33} al p_{23}	1
0.37048387	r_{31} al p_{34}	1
0.35120968	r_{11} al p_{37}	1
0.34403226	r_{22} al p_{34}	1
0.33000000	r_{21} al p_{37}	1
0.32862903	r_{13} al p_{13}	1
0.30322581	r_{32} al p_{34}	1
0.24919355	r_{23} al p_{11}	1
0.18951613	r_{24} al p_{34}	1
0.40653226	r_{33} al p_{34}	2
0.39951613	r_{12} al p_{34}	2
0.30346774	r_{31} al p_{13}	2
0.28153226	r_{11} al p_{11}	2
0.27024194	r_{22} al p_{11}	2
0.26274194	r_{21} al p_{25}	2
0.25701613	r_{13} al p_{34}	2
0.23790323	r_{32} al p_{37}	2
0.17322581	r_{23} al p_{34}	2
0.13435484	r_{24} al p_{11}	2
0.34677419	r_{33} al p_{13}	3
0.33443548	r_{12} al p_{23}	3
0.24250000	r_{31} al p_{21}	3
0.22330645	r_{22} al p_{13}	3
0.21233871	r_{11} al p_{13}	3
0.19983871	r_{21} al p_{13}	3
0.18612903	r_{13} al p_{31}	3
0.17524194	r_{32} al p_{13}	3
0.10790323	r_{23} al p_{21}	3
0.09516129	r_{24} al p_{23}	3
0.28725806	r_{33} al p_{37}	4
0.27637097	r_{12} al p_{13}	4
0.19637097	r_{31} al p_{23}	4
0.17975806	r_{22} al p_{12}	4
0.15725806	r_{21} al p_{12}	4
0.14314516	r_{11} al p_{12}	4
0.13629032	r_{13} al p_{21}	4
0.11717742	r_{32} al p_{23}	4
0.07096774	r_{23} al p_{32}	4
0.06298387	r_{24} al p_{35}	4
0.22798387	r_{33} al p_{12}	5
0.22459677	r_{12} al p_{11}	5
0.15185484	r_{31} al p_{31}	5
0.13846774	r_{22} al p_{21}	5
0.11596774	r_{21} al p_{22}	5
0.09709677	r_{13} al p_{32}	5
0.08991935	r_{11} al p_{32}	5
0.06685484	r_{32} al p_{36}	5
0.04403226	r_{23} al p_{33}	5
0.04112903	r_{24} al p_{36}	5
0.18282258	r_{33} al p_{31}	6
0.17669355	r_{12} al p_{12}	6
0.11411290	r_{31} al p_{12}	6
0.09943548	r_{22} al p_{22}	6
0.07741935	r_{21} al p_{11}	6
0.06983871	r_{13} al p_{36}	6
0.06604839	r_{11} al p_{36}	6
0.04322581	r_{32} al p_{35}	6
0.02056452	r_{23} al p_{24}	6
0.02024194	r_{24} al p_{24}	6
0.14056452	r_{33} al p_{21}	7
0.13669355	r_{12} al p_{21}	7
0.07669355	r_{31} al p_{22}	7
0.05443548	r_{22} al p_{35}	7
0.04354839	r_{13} al p_{35}	7
0.04306452	r_{21} al p_{33}	7
0.04266129	r_{11} al p_{33}	7
0.02104839	r_{32} al p_{33}	7
0.10975806	r_{12} al p_{33}	8
0.09862903	r_{33} al p_{22}	8
0.04782258	r_{31} al p_{36}	8
0.03306452	r_{22} al p_{33}	8
0.02145161	r_{21} al p_{36}	8
0.02104839	r_{13} al p_{33}	8
0.02032258	r_{11} al p_{24}	8
0.08443548	r_{12} al p_{36}	9
0.06588710	r_{33} al p_{33}	9
0.02217742	r_{31} al p_{35}	9
0.01217742	r_{22} al p_{36}	9
0.06032258	r_{12} al p_{24}	10
0.04250000	r_{33} al p_{35}	10
0.03798387	r_{12} al p_{32}	10
0.01959677	r_{33} al p_{36}	10
0.01693548	r_{12} al p_{35}	11

TABLE V. FINAL GLOBAL PRIORITY ORDERED BY PROCESS

Final Global Priority	Resources	Process	Round
0.24919355	r_{23}	p_{11}	1
0.28153226	r_{11}	p_{11}	2
0.27024194	r_{22}	p_{11}	2
0.13435484	r_{24}	p_{11}	2
0.22459677	r_{12}	p_{11}	5
0.07741935	r_{21}	p_{11}	6
0.17975806	r_{22}	p_{12}	4
0.15725806	r_{21}	p_{12}	4
0.14314516	r_{11}	p_{12}	4
0.22798387	r_{33}	p_{12}	5
0.17669355	r_{12}	p_{12}	6
0.11411290	r_{31}	p_{12}	6
0.32862903	r_{13}	p_{13}	1
0.30346774	r_{31}	p_{13}	2
0.34677419	r_{33}	p_{13}	3
0.22330645	r_{22}	p_{13}	3
0.21233871	r_{11}	p_{13}	3
0.19983871	r_{21}	p_{13}	3
0.17524194	r_{32}	p_{13}	3
0.27637097	r_{12}	p_{13}	4
0.24250000	r_{31}	p_{21}	3
0.10790323	r_{23}	p_{21}	3
0.13629032	r_{13}	p_{21}	4
0.13846774	r_{22}	p_{21}	5
0.14056452	r_{33}	p_{21}	7
0.13669355	r_{12}	p_{21}	7
0.11596774	r_{21}	p_{22}	5
0.09943548	r_{22}	p_{22}	6
0.07669355	r_{31}	p_{22}	7
0.09862903	r_{33}	p_{22}	8
0.46798387	r_{33}	p_{23}	1
0.33443548	r_{12}	p_{23}	3
0.09516129	r_{24}	p_{23}	3
0.19637097	r_{31}	p_{23}	4
0.11717742	r_{32}	p_{23}	4
0.02056452	r_{23}	p_{24}	6
0.02024194	r_{24}	p_{24}	6
0.02032258	r_{11}	p_{24}	8
0.06032258	r_{12}	p_{24}	10
0.26274194	r_{21}	p_{25}	2
0.18612903	r_{13}	p_{31}	3
0.15185484	r_{31}	p_{31}	5
0.18282258	r_{33}	p_{31}	6
0.07096774	r_{23}	p_{32}	4
0.09709677	r_{13}	p_{32}	5
0.08991935	r_{11}	p_{32}	5
0.03798387	r_{12}	p_{32}	10
0.04403226	r_{23}	p_{33}	5
0.04306452	r_{21}	p_{33}	7
0.04266129	r_{11}	p_{33}	7
0.02104839	r_{32}	p_{33}	7
0.10975806	r_{12}	p_{33}	8
0.03306452	r_{22}	p_{33}	8
0.02104839	r_{13}	p_{33}	8
0.06588710	r_{33}	p_{33}	9
0.37048387	r_{31}	p_{34}	1
0.34403226	r_{22}	p_{34}	1
0.30322581	r_{32}	p_{34}	1
0.18951613	r_{24}	p_{34}	1
0.40653226	r_{33}	p_{34}	2
0.39951613	r_{12}	p_{34}	2
0.25701613	r_{13}	p_{34}	2
0.17322581	r_{23}	p_{34}	2
0.06298387	r_{24}	p_{35}	4
0.04322581	r_{32}	p_{35}	6
0.05443548	r_{22}	p_{35}	7
0.04354839	r_{13}	p_{35}	7
0.02217742	r_{31}	p_{35}	9
0.04250000	r_{33}	p_{35}	10
0.01693548	r_{12}	p_{35}	11
0.06685484	r_{32}	p_{36}	5
0.04112903	r_{24}	p_{36}	5
0.06983871	r_{13}	p_{36}	6
0.06604839	r_{11}	p_{36}	6
0.04782258	r_{31}	p_{36}	8
0.02145161	r_{21}	p_{36}	8
0.08443548	r_{12}	p_{36}	9
0.01217742	r_{22}	p_{36}	9
0.01959677	r_{33}	p_{36}	10
0.47306452	r_{12}	p_{37}	1
0.35120968	r_{11}	p_{37}	1
0.33000000	r_{21}	p_{37}	1
0.23790323	r_{32}	p_{37}	2
0.28725806	r_{33}	p_{37}	4

Once the ***CDSAF*** table is completed, the Final Global Priorities of the Process (***FGPP***) will be calculated:

$\boldsymbol{FGPP}_1$ = (0.24919355 + 0.28153226 + 0.27024194 + 0.13435484 + 0.22459677 + 0.07741935) / 6

$\boldsymbol{FGPP}_2$ = (0.17975806 + 0.15725806 + 0.14314516 + 0.22798387 + 0.17669355 + 0.11411290) / 6

$\boldsymbol{FGPP}_3$ = (0.32862903 + 0.30346774 + 0.34677419 + 0.22330645 + 0.21233871 + 0.19983871 + 0.17524194 + 0.27637097) / 7

$\boldsymbol{FGPP}_4$ = (0.24250000 + 0.10790323 + 0.13629032 + 0.13846774 + 0.14056452 + 0.13669355) / 6

$\boldsymbol{FGPP}_5$ = (0.11596774 + 0.09943548 + 0.07669355 + 0.09862903) / 4

$\boldsymbol{FGPP}_6$ = (0.46798387 + 0.33443548 + 0.09516129 + 0.19637097 + 0.11717742) / 5

$\boldsymbol{FGPP}_7$ = (0.02056452 + 0.02024194 + 0.02032258 + 0.06032258) / 4

$\boldsymbol{FGPP}_8$ = 0.26274194 / 1

$\boldsymbol{FGPP}_9$ = (0.18612903 + 0.15185484 + 0.18282258) / 3

$\boldsymbol{FGPP}_{10}$ = (0.07096774 + 0.09709677 + 0.08991935 + 0.03798387) / 4

$\boldsymbol{FGPP}_{11}$ = (0.04403226 + 0.04306452 + 0.04266129 + 0.02104839 + 0.10975806 + 0.03306452 + 0.02104839 + 0.06588710) / 8

$\boldsymbol{FGPP}_{12}$ = (0.37048387 + 0.34403226 + 0.30322581 + 0.18951613 + 0.40653226 + 0.39951613 + 0.25701613 + 0.17322581) / 8

$\boldsymbol{FGPP}_{13}$ = (0.06298387 + 0.04322581 + 0.05443548 + 0.04354839 + 0.02217742 + 0.04250000 + 0.01693548) / 7

$\boldsymbol{FGPP}_{14}$ = (0.06685484 + 0.04112903 + 0.06983871 + 0.06604839 + 0.04782258 + 0.02145161 + 0.08443548 + 0.01217742 + 0.01959677) / 9

$\boldsymbol{FGPP}_{15}$ = (0.47306452 + 0.35120968 + 0.33000000 + 0.23790323 + 0.28725806) / 5

The ***CDSAF*** table ordered by process, as shown in **Table 5**.

By calculating the ***FGPP*** for all the processes, as shown in **Table 5**, a vector will be obtained, as shown in **Table 6**.

The elements of the ***FGPP*** vector must be ordered from highest to lowest, in order to obtain the global priority order of allocation of resources to processes, as can be seen in **Table 7**.

TABLE VI. FINAL GLOBAL PRIORITY OF THE PROCESS (***FGPP***)

Ordered Final Global Priority	***Assignment***
0.20622312	p_{11}
0.16649193	p_{12}
0.25824597	p_{13}
0.15040323	p_{21}
0.09768145	p_{22}
0.24222581	p_{23}
0.03036291	p_{24}
0.26274194	p_{25}
0.17360215	p_{31}
0.07399193	p_{32}
0.04757057	p_{33}
0.30544355	p_{34}
0.04082949	p_{35}
0.04770609	p_{36}
0.33588710	p_{37}

TABLE VII. ORDERED FINAL GLOBAL PRIORITY OF THE PROCESS (***OFGPP***)

Ordered Final Global Priority	***Process***
0.33588710	p_{37}
0.30544355	p_{34}
0.26274194	p_{25}
0.25824597	p_{13}
0.24222581	p_{23}
0.20622312	p_{11}
0.17360215	p_{31}
0.16649193	p_{12}
0.15040323	p_{21}
0.09768145	p_{22}
0.07399193	p_{32}
0.04770609	p_{36}
0.04757057	p_{33}
0.04082949	p_{35}
0.03036291	p_{24}

The cardinalities (number of allocation of resources to each process) obtained from each one of the ***OFGPP*** vector processes in the ***CDSAF*** table will be calculated.

CP_{37}= process cardinality($\boldsymbol{OFGPP}_1$) in ***CDSAF*** = 5

CP_{34}= process cardinality($\boldsymbol{OFGPP}_2$) in ***CDSAF*** = 8

CP_{25}= process cardinality($\boldsymbol{OFGPP}_3$) in ***CDSAF*** = 1

CP_{13}= process cardinality($\boldsymbol{OFGPP}_4$) in ***CDSAF*** = 8

CP_{23}= process cardinality($\boldsymbol{OFGPP}_5$) in ***CDSAF*** = 5

CP_{11}= process cardinality($\boldsymbol{OFGPP}_6$) in ***CDSAF*** = 6

CP_{31}= process cardinality($\boldsymbol{OFGPP}_7$) in ***CDSAF*** = 3

CP_{12}= process cardinality($\boldsymbol{OFGPP}_8$) in ***CDSAF*** = 6

CP_{21}= process cardinality($\boldsymbol{OFGPP}_9$) in ***CDSAF*** = 6

CP_{22}= process cardinality($\boldsymbol{OFGPP}_{10}$) in ***CDSAF*** = 4

CP_{32}= process cardinality($\boldsymbol{OFGPP}_{11}$) in ***CDSAF*** = 4

CP_{36}= process cardinality($\boldsymbol{OFGPP}_{12}$) in ***CDSAF*** = 9

CP_{33}= process cardinality($\boldsymbol{OFGPP}_{13}$) in ***CDSAF*** = 8

CP_{35}= process cardinality($\boldsymbol{OFGPP}_{14}$) in ***CDSAF*** = 7

CP_{24}= process cardinality($\boldsymbol{OFGPP}_{15}$) in ***CDSAF*** = 4

Then, each one of the allocation of resources to processes in the ***CDSAF*** table of each process of ***OFGPP*** vector must be obtained. The total number of elements for each process will be determined by the cardinality calculated in the previous step.

$\boldsymbol{OCDSAF}_1$ = first element of the ***CDSAF*** for the process ($\boldsymbol{OFGPP}_1$)

$\boldsymbol{OCDSAF}_5$ = last element of the ***CDSAF*** for the process ($\boldsymbol{OFGPP}_1$)

$\boldsymbol{OCDSAF}_{5+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_2$)

$\boldsymbol{OCDSAF}_{5+8}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_2$) process

$\boldsymbol{OCDSAF}_{5+8+1}$ = the element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_3$)

$\boldsymbol{OCDSAF}_{5+8+1+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_4$) process

$\boldsymbol{OCDSAF}_{5+8+1+8}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_4$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_5$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_5$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_6$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_6$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{7}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{7}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{8}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{8}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{9}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{9}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{10}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{10}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{11}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{11}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{12}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4+9}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{12}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4+9+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{13}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4+9+8}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{13}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4+9+8+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{14}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4+9+8+7}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{14}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4+9+8+7+1}$ = first element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{15}$) process

$\boldsymbol{OCDSAF}_{5+8+1+8+5+6+3+6+6+4+4+9+8+7+4}$ = last element of the ***CDSAF*** for the ($\boldsymbol{OFGPP}_{15}$) process.

Table 8 shows the order of all the resource allocations for each process, which one is the first process with greater global priority, and is the one to which the resources are assigned first. The complete table continues for each one of the requests for each process (***OCDSAF***).

TABLE VIII. FINAL ORDER OF ALLOCATION OF EACH ONE OF THE RESOURCES TO EACH OF ONE PROCESSES OF THE ***(OCDSAF)***

Priority	Resource	Process	Round
0.4730645	r_{12}	p_{37}	1
0.3512097	r_{11}	p_{37}	1
0.3300000	r_{21}	p_{37}	1
0.2379032	r_{32}	p_{37}	2
0.2872581	r_{33}	p_{37}	4
0.3704839	r_{31}	p_{34}	1
0.3440323	r_{22}	p_{34}	1
0.3032258	r_{32}	p_{34}	1
0.1895161	r_{24}	p_{34}	1
0.4065323	r_{33}	p_{34}	2
0.3995161	r_{12}	p_{34}	2
0.2570161	r_{13}	p_{34}	2
0.1732258	r_{23}	p_{34}	2
0.2627419	r_{21}	p_{25}	2
0.3286290	r_{13}	p_{13}	1
0.3034677	r_{31}	p_{13}	2
0.3467742	r_{33}	p_{13}	3
0.2233065	r_{22}	p_{13}	3
0.2123387	r_{11}	p_{13}	3
0.1998387	r_{21}	p_{13}	3
0.1752419	r_{32}	p_{13}	3
0.2763710	r_{12}	p_{13}	4
0.4679839	r_{33}	p_{23}	1
0.3344355	r_{12}	p_{23}	3
0.0951613	r_{24}	p_{23}	3
0.1963710	r_{31}	p_{23}	4
0.1171774	r_{32}	p_{23}	4
0.2491936	r_{23}	p_{11}	1
0.2815323	r_{11}	p_{11}	2
0.2702419	r_{22}	p_{11}	2
0.1343548	r_{24}	p_{11}	2
0.2245968	r_{12}	p_{11}	5
0.0774194	r_{21}	p_{11}	6
0.1861290	r_{13}	p_{31}	3
0.1518548	r_{31}	p_{31}	5
0.1828226	r_{33}	p_{31}	6
0.1797581	r_{22}	p_{12}	4
0.1572581	r_{21}	p_{12}	4
0.1431452	r_{11}	p_{12}	4
0.2279839	r_{33}	p_{12}	5
0.1766936	r_{12}	p_{12}	6
0.1141129	r_{31}	p_{12}	6
0.2425000	r_{31}	p_{21}	3
0.1079032	r_{23}	p_{21}	3
0.1362903	r_{13}	p_{21}	4
0.1384677	r_{22}	p_{21}	5
0.1405645	r_{33}	p_{21}	7
0.1366936	r_{12}	p_{21}	7
0.1159677	r_{21}	p_{22}	5
0.0994355	r_{22}	p_{22}	6
0.0766936	r_{31}	p_{22}	7
0.0986290	r_{33}	p_{22}	8
0.0709677	r_{23}	p_{32}	4
0.0970968	r_{13}	p_{32}	5
0.0899194	r_{11}	p_{32}	5
0.0379839	r_{12}	p_{32}	10
0.0668548	r_{32}	p_{36}	5
0.0411290	r_{24}	p_{36}	5
0.0698387	r_{13}	p_{36}	6
0.0660484	r_{11}	p_{36}	6
0.0478226	r_{31}	p_{36}	8
0.0214516	r_{21}	p_{36}	8
0.0844355	r_{12}	p_{36}	9
0.0121774	r_{22}	p_{36}	9
0.0195968	r_{33}	p_{36}	10
0.0440323	r_{23}	p_{33}	5
0.0430645	r_{21}	p_{33}	7
0.0426613	r_{11}	p_{33}	7
0.0210484	r_{32}	p_{33}	7
0.1097581	r_{12}	p_{33}	8
0.0330645	r_{22}	p_{33}	8
0.0210484	r_{13}	p_{33}	8

0.0658871	r_{33}	p_{33}	9
0.0629839	r_{24}	p_{35}	4
0.0432258	r_{32}	p_{35}	6
0.0544355	r_{22}	p_{35}	7
0.0435484	r_{13}	p_{35}	7
0.0221774	r_{31}	p_{35}	9
0.0425000	r_{33}	p_{35}	10
0.0169355	r_{12}	p_{35}	11
0.0205645	r_{23}	p_{24}	6
0.0202419	r_{24}	p_{24}	6
0.0203226	r_{11}	p_{24}	8
0.0603226	r_{12}	p_{24}	10

In this way, all the requests of resources from all the processes were answered, considering the mutual exclusion and the priorities of the processes, the nodal priorities and the final priorities, taking into account the strict consensus requirements established for this scenario.

V. Evaluation

The data structure mentioned above and the aggregation method used are not fully covered by traditional methods.

This work considers the global average of priorities that each process has over all the resources of all its assignments in the different rounds, but for the final global allocation, it respects the same order of allocation of each resource in the different rounds in which they were assigned in the general scenario. That is, the choice of which process will be granted resources, is established with the global average of priorities in all assignments, but the order in which those assignments are to be made, respects the one in the table ***FASD***, for each process.

The proposed model manages to establish a consensus that allows processes to access all their resources sequentially and that these cannot be removed until the process that holds them releases them. The order of assignment will be determined by the overall average priority of all the assignments. The distributed system regulates and constantly updates the local state of each node, the decisions of access to resources modify these states so it must be readjusted repeatedly, guaranteeing mutual exclusion and reordering new priorities. The method must be repeated whenever there are processes that require shared resources.

VI. Conclusions

The proposed model includes, as a particular case, a method that consists in considering the global priority of the processes, instead of a group of state variables of each node. As the processes are executed in different processors using all their resources, there is no conflict in running several processes in the same processor. In this scenario, no account is taken of the amount of time each process will use in a processor of a particular node. Nor is the amount of time in which each resource will be assigned to a particular process Another notable feature of the proposal is its ease of implementation in the environment of a centralized administrator of shared resources of a distributed system.

VII. Future Lines of Research

It is considered to develop decision models from the cognitive point of view for decision making in groups of processes, contemplating the principles of cybernetics of second order, in the context of complex systems of self-regulation, which transcend the traditional approach of computer science considering the possibility of imputation of missing data, for example, as a consequence of problems in communications between processes, and fuzzyfication of variables to support situations where it is not possible or convenient to express exact values.

In addition, the aim is to investigate the impact on data traffic of applying the proposed method and comparing it with other classical methods. To this end, a simulator will be developed in which the different possible scenarios will be considered to allow the system to predict, compare and optimise the behaviour of its simulated processes in a very short time without the cost or risk of carrying them out, making it possible to represent the processes, resources and nodes in a dynamic model.

Another possible line of research considers aspects related to security in the execution of processes, access to resources and communication between nodes.

Acknowledgment

This work has been supported by the Project: “Decision models and aggregation operators for process management in distributed systems”, code 16F001 of the Northeastern National University (Argentina).

References

[1] S. Tanenbaum, Sistemas Operativos Distribuidos. Prentice - Hall Hispanoamericana S.A., México, 1996.

[2] A. S. Tanenbaum, **Sistemas Operativos Modernos**. 3ra. Edición. Pearson Educación S. A., México, 2009.

[3] D. Agrawal, A. El Abbadi, “An Efficient and Fault-Tolerant Solution of Distributed Mutual Exclusion”. **ACM Trans. on Computer Systems**. Vol. 9, pp. 1-20, USA, 1991.

[4] G. Ricart, A. K. Agrawala, “An Optimal Algorithm for Mutual Exclusion in Computer Networks”. **Commun. of the ACM**. Vol. 24, pp. 9-17, 1981.

[5] G. Cao, M. Singhal, “A Delay-Optimal Quorum-Based Mutual Exclusion Algorithm for Distributed Systems”. **IEEE Transactions on Parallel and Distributed Systems.** Vol. 12, no. 12, pp. 1256-1268. USA, 2001.

[6] S. Lodha, A. Kshemkalyani, “A Fair Distributed Mutual Exclusion Algorithm”. **IEEE Trans. Parallel and Distributed Systems.** Vol. 11, no. 6, pp. 537-549, USA, 2000.

[7] W. Stallings, **Sistemas Operativos**. 5ta. Edición. Pearson Educación S.A., España, 2005.

[8] G. Andrews, Foundation of Multithreaded, Parallel, and Distributed Programming. Reading, MA: Addison-Wesley. USA, 2000.

[9] R. Guerraoui, L. Rodrigues, **Introduction to Reliable Distributed Programming.** Berlin, Springer-Verlag, 2006.

[10] N. Lynch, **Distributed Algorithms**. Morgan Kauffman, San Mateo, CA, USA, 1996.

[11] G. Tel, **Introduction to Distributed Algorithms**. Cambridge University Press, 2nd ed. Cambridge, UK, 2000.

[12] H. Attiya, J. Welch, **Distributed Computing Fundamentals, Simulations, and Advanced Topics**, John Wiley, 2nd ed., New York, USA, 2004.

[13] P. Saxena, J. Rai, “A Survey of Permission-based Distributed Mutual Exclusion Algorithms”. **Computer Standards and Interfaces**, vol. (25)2, pp 159-181, 2003.

[14] M. Velazquez, “A Survey of Distributed Mutual Exclusion Algorithms”. **Technical Report** CS-93-116, University of Colorado at Boulder, 1993.

[15] S.-D. Lin, Q. Lian, M. Chen, Z. Zhang, "A Practical Distributed Mutual Exclusion Protocol in Dynamic Peer-to-Peer Systems". **Proc. Third International Workshop on Peer-to-Peer Systems**, vol. 3279 of Lect. Notes Compo Sc., (La Jolla, CA). Springer-Verlag, Berlin, 2004.

[16] L. Sha, R. Rajkumar, J. P. Lehoczky, "Priority inheritance protocols: An approach to real-time synchronization". **Computers, IEEE Transactions on**, vol. 39(9), pp1175–1185, 1990.

[17] S. Greco, B. Matarazzo, R. Slowinski, "Rough sets methodology for sorting problems in presence of multiple attributes and criteria", **European Journal of Operational Research**, 2002, 138, pp. 247-259.

[18] X. Chao, G. Kou, Y. Peng, "An optimization model integrating different preference formats", **6th International Conference on Computers Communications and Control (ICCCC)**, 2016, pp. 228 - 231.

[19] D. L. La Red Martínez, J. C. Acosta, "Aggregation Operators Review - Mathematical Properties and Behavioral Measures", **International Journal of Intelligent Systems and Applications (IJISA)**, Hong Kong, 2015, 7, (10), pp. 63-76.

[20] D. L. La Red Martínez, N. Pinto, "Brief Review of Aggregation Operators", **Wulfenia Journal**, Austria, 2015, 22, (4), pp. 114-137.

[21] R. Yager, "On Ordered Weighted Averaging Aggregation Operators in Multi-Criteria Decision Making", **IEEE Trans. On Systems, Man and Cybernetics**, 1988, 18, (1), pp. 183-190.

[22] R. Yager, "Families of OWA Operators. Fuzzy Sets and Systems", 1993, 59, (2), pp. 125-148.

[23] R. Yager, Kacprzyk J.: "The Ordered Weighted Averaging Operators", **Theory and Applications, Kluwer Academic Publishers**, USA, 1997.

[24] R. Yager, Pasi, G.: "Modelling Majority Opinion in Multi-Agent Decision Making", **International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems**, 2002.

[25] Y. Dong, Zhang, H., Herrera-Viedma, E., "E. Consensus reaching model in the complex and dynamic MAGDM problem", **Knowledge Based Systems, Elsevier**, 2016, 106, pp. 206-219.

[26] D. L. La Red Martínez, "Aggregation Operator for Assignment of Resources in Distributed Systems." **International Journal of Advanced Computer Science and Applications (IJACSA)**. The Science and Information (SAI) Organization, England, U.K, 2017, 8, (10), pp. 406-419, ISSN N° 2156-5570.

[27] D. L. La Red Martínez, J. C. Acosta, F. Agostini, "Assignment of Resources in Distributed Systems". **9th International Multi-Conference on Complexity, Informatics and Cybernetics (IMCIC 2018)**, Orlando, USA, 2018.