Resource and Process Management With a Decision Model Based on Fuzzy Logic

J. T. Fornerón Martínez[1], F. Agostini[2], D. L. La Red Martínez[2]

[1] Faculty of Applied Sciences, National University of Pilar, Pilar (Paraguay)
[2] Faculty of Exact and Natural Sciences and Surveying, Northeastern National University, Corrientes (Argentine)

Received 28 June 2020 | Accepted 3 February 2023 | Published 14 February 2023

Abstract

The allocation of the resources to be shared in the context of a distributed processing system needs to be coordinated through the mutual exclusion mechanism, which will decide the order in which the shared resources will be allocated to those processes that require them. This paper proposes an aggregation operator, which can be used by a module that manages the shared resources, whose function is to assign the resources to the processes according to their requirements (shared resources) and the status of the distributed nodes in which the processes operate (computational load), by using 2-tuple associated to linguistic labels.

Keywords

Aggregation Operators, Communication Between Groups Of Processes, Computing With Word, Fuzzy Logic, Mutual Exclusion, Operating Systems.

DOI: 10.9781/ijimai.2023.02.009

I. Introduction

Distributed systems, composed of multiple nodes and multiple processes, cooperatively performing a given function, require the use of decision models that allow the use of shared resources to groups of processes that require them, accessed through the mutual exclusion mode.

Solutions proposed for this problem are found in [1] and [2], where the main synchronization algorithms in distributed systems are described. [3] presents an efficient solution, also fault-tolerant, for the problem of distributed mutual exclusion. [4], [5] and [6] present algorithms for the management of mutual exclusion in computer networks. [7] focuses on the main algorithms for the management of distributed processes, distributed mutual exclusion and distributed global states.

Solutions proposed for this problem are found in [1] and [2], where the main synchronization algorithms in distributed systems are described. [3] presents an efficient solution, also fault-tolerant, for the problem of distributed mutual exclusion. [4], [5] and [6] present algorithms managing mutual exclusion in computer networks. [7] focuses on the main algorithms for the management of distributed processes, distributed mutual exclusion and distributed global states.

In distributed systems, the allocation of resources to processes must be done considering the priorities of the processes and the workload status of the computational nodes in which the processes are executed.

Besides, solutions that we could call classic for several types of distributed systems have been proposed in [8], [9], [10], [11] and [12]. Also, in [13] and [14] works focused on ensuring mutual exclusion are presented. [15] presents an interesting distributed solution based on permissions and [16] a solution based on process priorities. A solution using consensus in resource allocation is presented in [17].

There are practical situations in which problems must be solved by having vague and imprecise information. This means that information is not always evaluated accurately with quantitative values, but with qualitative values. This was solved by [18], by incorporating the concept of the linguistic variable and applying it to decision making as well as explained in [19]. Also [20] expressed that computing with word (CWW) is a methodology in which words are used instead of numbers for computation and reasoning and where fuzzy logic plays a fundamental role in CWW and vice versa.

As mentioned in [21], based on the concept of symbolic translation, Herrera and Martínez proposed the 2-tuple linguistic representation model, which expresses linguistic assessment information using the linguistic 2-tuple (s_i, α), where the semantic element s_i is a linguistic label from a predefined linguistic variable, S, and α is a numerical value that represents the symbolic translation. Also, Zhang et al. introduced interval-valued hesitant fuzzy soft sets by combining the interval-valued hesitant fuzzy set and soft set models and evaluated their operations. Tao et al. presented the 2-tuple linguistic soft set method, incorporating the 2-tuple linguistic term set and soft set, to solve complex group decision-making problems. Today, soft set methods are widely applied to solve real-life decision-making problems (e.g., Ali and Shabir; Chang; Deli and Cagman; Tang; Chang; Chang et al.). The use of various algorithms to counteract uncertainty and incomplete

* Corresponding author.
E-mail addresses: jforneron@aplicadas.edu.py (J. Fornerón), fagostini@conicet.gov.ar (F. Agostini), lrmdavid@exa.unne.edu.ar (D. La Red Martínez)

information when trying to solve multi-criteria decision-making problems (MCDM), where generally precise values and a single set of linguistic terms are insufficient to deal with the complexity of selection problems, where experts hesitate among sets of linguistic terms to determine the values of evaluation attributes in said problems.

Different words can have different meanings for different people. Answer about how can a Computing With Words (CWW) engine be validated, what Fuzzy Set models should be used or what choices should be made to keep the design of the CWW engine as simple as possible, are analyzed and founded in [22].

The new decision models for allocating shared resources could be executed in the context of a shared resource manager for the distributed system, which would receive the shared resource requirements of the processes running on the different distributed nodes, as well as the computational load state of the nodes.

It has been worked with fuzzy variables using linguistic labels and 2-tuple to avoid losing precision in computing with words.

A computational model has been presented in [23], called 2-tuple linguistic computational model, in which a parameter was incorporated to the basic linguistic representation to improve the accuracy of linguistic calculations.

The fuzzy linguistic approach, although with the limitations at the moment of being used in fusion processes on the linguistic values, is used successfully in the resolution of many problems and presents tools to improve the application of the fuzzy linguistic approach, in relation to the loss of information caused by the need to express the results in the initial expression domain, which is discrete through an approximate process and implies a lack of precision in the final results of the fusion of linguistic information. Linguistic information is expressed through a 2-tuple, composed of a linguistic term and a numerical value evaluated at [-0.5, 0.5], which allows to represent the information obtained in an aggregation process. Together with the 2-tuple representation, a computational technique for word computing (CWW) is developed [24].

A clear explanation about main CW concepts can be found in [25]: granules and linguistic variables. A granule is defined as a clump of objects (or points) which are drawn together by indistinguishability, similarity, proximity, or functionality.

An example of granularity is a system that is composed of several smaller subsystems and these smaller subsystems are in turn divided into even smaller ones. The decomposition of the whole into parts (granulation) is, in general, hierarchical in nature.

A linguistic variable is a variable whose values are not numbers but words or sentences in a natural or artificial language. The main purpose of using linguistic values (words or sentences) instead of numbers is that linguistic characterizations are, in general, less specific than numerical ones, but much closer to the way that humans express and use their knowledge.

Other models have been presented in [23], [26], [27], [28] and [29]. These works show different advantages of this formalism to represent linguistic information over classical models.

It is a continuous linguistic domain, where the linguistic calculation model is based on linguistic tuples and performs word computation processes easily and without loss of information, therefore, the results of the word computation processes are always expressed in the initial linguistic domain.

Due to these advantages, this model of linguistic representation will be used to achieve the development of a procedure for the fusion of linguistic and numerical information.

The 2-tuple model of linguistic representation represents linguistic information by means of a 2-tuple, (s, α). In this work it will be used to represent the load of the nodes, nodal preferences, and final priorities. An example of this can be seen in Table I.

The symbolic translation of a linguistic term $s_i \in S = \{s_o, ..., s_g\}$ consists of a numerical value $\alpha_i \in [-.5, .5)$ that supports the "information difference" between an information count β evaluated in [0, g] obtained after a symbolic aggregation operation (acting on the order index of the labels) and the closest value in $[0, ..., g]$ that indicates the index of the closest linguistic term in S (s_i).

TABLE I. 2-Tuple Linguistic Weighted Average Score

	Criteria					*Aggregate Weight*
	C1	***C2***	***C3***	***C4***	***C5***	
A1	(S5, 0.00)	(S5, 0.31)	(S6, 0.33)	(S5, 0.50)	(S5, 0.00)	
A2	(S5, 0.00)	(S6, 0.00)	(S2, 0.00)	(S5, 0.31)	(S6, 0.00)	(S5, -0.21)
A3	(S5, 0.31)	(S5, 0.50)	(S3, 0.00)	(S6, 0.00)	(S5, 0.00)	(S5, -0.31)
A4	(S5, 0.00)	(S5, 0.50)	(S5, 0.50)	(S5, 0.00)	(S5, 0.31)	(S5, -0.15)
A5	(S4, 0.00)	(S5, 0.50)	(S4, 0.00)	(S5, 0.50)	(S4, 0.00)	(S4, 0.19)

Distributed systems are used in multiple solutions around the world, smart container management, connected smart plants, banking networks, the world wide web, etc., as seen in Fig 1.

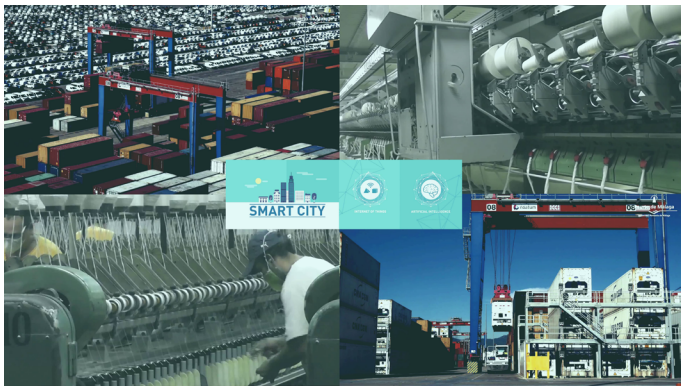

Fig. 1. Samples of the internet of things, smart cities, industry 4.0 (industrial robotics), etc.

In this paper, a new aggregation operator will be presented specifically for the problem. This falls under the category of OWA operators, more specifically Neat OWA. This will present an innovative method for shared resource management in distributed systems.

The structure of this document is as follows: section II gives guidelines about the premises and data structures to be used, section III describes the steps of the proposed aggregation operator, section IV explains details of an example of application of the proposed aggregation operator, section V presents the conclusions and future work and section VI mentions acknowledgement.

II. Data Structures to be Used

The following premises and data structures will be used.

These are groups of processes 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 in the processes, what the priorities will be for assigning the resources to the processes that require them (in order to be assigned in the processes, only the available resources will be taken into account, that is, those that have not yet been assigned in certain

processes. All the premises, resources and processes running in the different nodes, groups, cardinals, criteria, and categories to evaluate the different weights and calculations required are those mentioned in [30].

III. Description of the Aggregation Operator

The proposed operator consists of the following steps:

A. Calculation of the current computational load of the nodes.

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

C. Calculation of the priorities or preferences of the processes considering the state of the node (they are calculated in each node for each process).

D. Expression of the calculated values in terms of 2-tuple using a set of linguistic labels.

E. Calculation of the priorities or preferences of the processes to access the shared resources available and determination of the order and to which process the resources will be allocated.

Each of the steps above is described below.

A. Calculation of the Current Computational Load of the Nodes

To obtain an indicator of the current computational load of each node, different criteria can be adopted; in this proposal the criteria will be the percentage of CPU usage, the percentage of memory usage and the percentage of use of input / output operation. The computational load of each node, the number of criteria to determine the load of the nodes, the criteria that apply and the calculation of the computational load of each node, are those mentioned in [30].

B. Establishment of the Categories of Computational Load and of the Vectors of Weights Associated Thereto

The current computational load categories of each node, the number of categories to determine the load of the nodes, the categories that apply, the vectors of weights associated with the current computational load categories of each node. In this proposal, the criteria will be those used in [30].

Establishment of vectors of weights (same for all nodes): weights = $\{w_{ij}\}$ con $i = 1, ..., a$ (categories number of computational load) y $j = 1, ..., e$ (maximum number of criteria).

C. Calculation of the Priorities or Preferences of the Processes Considering the Status of the Node (They Are Calculated in Each Node for Each Process and Could Be Called Nodal Priorities)

These priorities are calculated at each node for each resource request originated in each process; the calculation considers the corresponding weight vector according to the current load of the node and the vector of the values granted by the node according to the evaluation criteria of the request.

The valuation vectors that will be applied for each request of a resource by a process, according to the criteria established for the determination of the priority that in each case and moment will fix the node in which the request occurs, are the following: *valuations* $(r_{ij}\, p_{kl}) = \{cp_m\}$ con $i = 1, ..., n$ (node where the resource resides), $j = 1, ..., r$ (resource on node i), $k = 1, ..., n$ (node where the process resides), $l = 1, ..., p$ (process at node k) and $m = 1, ..., e$ (valuation criteria of the requirement priority). As can be seen in Table II.

TABLE II. Valuations Assigned to the Criteria to Calculate the Priority or Preference That Each Node Will Give to Each Requirement of Each Process According to the Node Load

Resources - Processes	Criteria				
$r_{11}\, p_{11}$	cp_1	...	cp_m	...	cp_e
...	...	...	...	...	...
$r_{ij}\, p_{kl}$	cp_1	...	cp_m	...	cp_e
...	...	...	...	...	...
$r_{nr}\, p_{np}$	cp_1	...	cp_m	...	cp_e

D. Expression of the Calculated Values in Terms of 2-Tuple Using a Set of Linguistic Labels

The valuations expressed in a linguistic format using the linguistic and semantic labels mentioned can be seen in Fig. 2, where in the abscissa are indicated the linguistic labels and in the ordinates the values of probability of belonging to them.

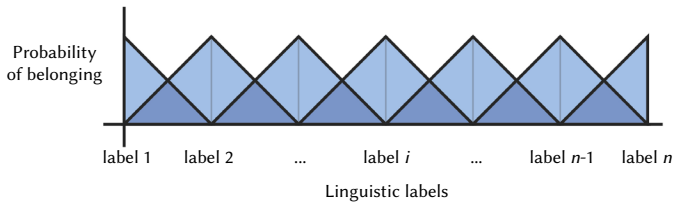

Fig. 2. Representation of the label set.

The next step is to transform these values into the 2-tuple format, considering the linguistic labels proposed above. Therefore, each criteria value will have to be compared with the average value of each label, the minimum difference of that comparison will be the appropriate label [24].

The first element of the 2-tuple will be the linguistic value of that label. The second element will be the difference between the value of the searched criteria and the average value of the selected label.

dm = the minimum difference between the cp_m differences and the most representative value of each language label.

Label valuations $(r_{ij}\, p_{kl})$ = 2-tuple = $T(label_m; dm_m)$ where the sub-index m corresponds to the linguistic labels defined above, as can be seen in Table III.

TABLE III. Valuations Assigned to the Criteria for Calculating the Priority or Preference That Each Node Will Give to Each Requirement of Each Process According to the 2-Tuple Load of the Node

Resources / Processes	2-tuple criteria				
$r_{11}\, p_{11}$	$T(label_1; dm_1)$	...	$T(label_m; dm_m)$	...	$T(label_e; dm_e)$
...	...	...	...	...	...
$r_{ij}\, p_{kl}$	$T(label_1; dm_1)$	...	$T(label_m; dm_m)$	...	$T(label_e; dm_e)$
...	...	...	...	...	...
$r_{nr}\, p_{np}$	$T(label_1; dm_1)$	...	$T(label_m; dm_m)$	...	$T(label_e; dm_e)$

To sum up, the nodal priority (to be calculated at the node where the request occurs) of a process to access a given resource (which can be at any node) is calculated by the scalar product of the mentioned vectors: nodal priority $(r_{ij}\, p_{kl}) = \Sigma\, w_{om} * T(label_m; dm_m) = T(label_n; dm_n) = NPT_{ijkl}$ (Nodal Priority Tuple) with o indicating the weights vector according to the load of the node, all other sub-index maintaining the meanings explained above. With m and n indicating the corresponding linguistic label within the adopted set defined above.

This nodal priority must be transformed into the 2-tuple format, considering the linguistic labels already mentioned. Therefore, it will be necessary to compare each nodal priority value with the average value of each label, the minimum difference of these comparisons will indicate the corresponding label.

E. Calculation of Process Priorities or Preferences to Access Available Shares. In Addition, Determining the Order in Which the Resources Will Be Allocated, and to Which Process Each Resource Will Be Allocated

Table IV is used to calculate the final priorities, in which the priorities or nodal preferences calculated in the previous stage are placed; in this table each row contains the information of the nodal priorities of the different processes to access a certain resource.

Next, it is necessary to calculate the vector of final weights that will be used in the process of aggregation to determine the order or priority of access to the resources.

final weights = $\{wf_{kl}\}$ con $k = 1, ..., n$ (number of nodes) and $l = 1, ..., p$ (Maximum number of processes per node), where np is the number of processes in the system and prg_i is the priority of the process group to which the process belongs (explained in the previous section).

TABLE IV. Nodal Priorities of the Processes to Access Each Resource in 2-Tuple

Resources	2-tuple				
r_{11}	NPT_{1111}	...	NPT_{11kl}	...	NPT_{11np}
...	...	...	...	...	...
r_{ij}	NPT_{ij11}	...	NPT_{ijkl}	...	NPT_{ijnp}
...	...	...	...	...	...
r_{nr}	NPT_{nr11}	...	NPT_{npkl}	...	NPT_{nrnp}

The next step is to normalize the newly obtained weights by dividing each by the sum of all of them.

Thus, a normalized weight vector (in the range of 0 to 1 inclusive) is obtained and with the restriction that the sum of the elements of the vector must give 1:

$\Sigma \{nwf_{kl}\} = 1$ with $k = 1, ..., n$ (number of nodes) and $l = 1,..., p$ (maximum number of processes per node).

The nodal priorities taken row by row for each resource will be scalar multiplied by the normalized final weight vector. In this way, it is possible to obtain each process's final global access priorities to each resource. It is indicated below how the order or priority with which the resources will be allocated is obtained and to which process each one will be assigned.

overall final priority $(r_{ij}\, p_{kl}) = NPT_{ijkl} = FGPT_{ijkl}$ (Final Global Priority Tuple) with r_{ij} indicating the resource j of node i, NPT_{ijkl} is the 2-tuple format, ij indicating the resource j of node i, kl the process l of node k and the product of the overall final priority of the process to access such resource, as can be seen in Table V.

TABLE V. Final Global Priority Tuple

Resources	Nodal Process Priorities				
r_{11}	$FGPT_{1111}$	...	$FGPT_{11kl}$	...	$FGPT_{11np}$
...	...	...	...	...	...
r_{ij}	$FGPT_{ij11}$	...	$FGPT_{ijkl}$	...	$FGPT_{ijnp}$
...	...	...	...	...	...
r_{nr}	$FGPT_{r11}$	...	$FGPT_{pkl}$	...	$FGPT_{rnp}$

The next step is to normalize Table V between extreme values. This will be done using the maximum, minimum and range values calculated from Table VI and represented in Table VII.

TABLE VI. Calculation of the Maximum, Minimum and Range Values

Label	Value
Maximum Value	Maximum ($FGPT_{ijkl}$)
Minimum value	Minimum ($FGPT_{ijkl}$)
Range	Maximum ($FGPT_{ijkl}$) – Minimum ($FGPT_{ijkl}$)

TABLE VII. Normalized Final Global Priority Tuple

Resources	2-tuple				
r_{11}	$NFGPT_{1111}$	...	$NFGPT_{11kl}$	...	$NFGPT_{11np}$
...	...	...	...	...	...
r_{ij}	$NFGPT_{ij11}$	...	$NFGPT_{ijkl}$	...	$NFGPT_{ijnp}$
...	...	...	...	...	...
r_{nr}	$NFGPT_{nr11}$	...	$NFGPT_{npkl}$	...	$NFGPT_{nrnp}$

The greater of these products made for the different processes in relation to the same resource will indicate which of the processes will have access to the resource.

The addition of all these products in relation to the same resource will indicate the priority that will have that resource to be assigned, in relation to the other resources that will also have to be assigned. This is what will be called Linguistic Distributed Systems Assignment Function (*LDSAF*). Refer to (1).

$$LDSAF(r_{ij}) = \Sigma\, NFGPT_{ijkl} = \text{resource allocation priority } r_{ij} \quad (1)$$

By calculating the *LDSAF* for all resources a 2-tuple vector will be obtained, and by ordering its elements from highest to lowest, the priority order of allocation of resources will be obtained. These must be normalized guaranteeing that the 2-tuples obtained are in the interval [0, 1]. The maximum, minimum and range values can be seen in Table VIII.

TABLE VIII. Valuations to Normalize the LDSAF

Label	Value
Maximum Value	Maximum ($LDSAF_{ijkl}$)
Minimum value	Minimum ($LDSAF_{ijkl}$)
Range	Maximum ($LDSAF_{ijkl}$) – Minimum ($LDSAF_{ijkl}$)

In addition, as already indicated, the largest of the products $NFGPT_{ijkl}$ for each resource will indicate the process to which the resource will be assigned.

This is what will be called Normalized Linguistic Distributed Systems Assignment Function (*NLDSAF*). Refer to (2).

$$NLDSAF(r_{ij}) = \Sigma\, (NFGPT_{ijkl}/(\text{maximum } (NFGPT_{ijkl}) - \text{minimum } (NFGPT_{ijkl}))) = r_{ij} \text{ resource allocation priority normalized between extreme values} \quad (2)$$

This can be seen in Table IX.

TABLE IX. Order or Final Priority of Assignment of Resources and Process to Which Is Allocated Each Resource in the First Iteration

Order of allocation of resources	*Process to which the resource will be assigned*
1°: *r*ij of the Max(*NLDSAF*(*r*ij))	p_{kl} of the Max(NFGPT$_{ijkl}$) for the selected r_{ij}
2°: *r*ij of the Max(*NLDSAF*(*r*ij)) for unallocated r_{ij}	p_{kl} of the Max(NFGPT$_{ijkl}$) for the selected r_{ij}
...	...
last: r_{ij} no assigned	p_{kl} of the Max(NFGPT$_{ijkl}$) for the selected r_{ij}

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. Table VIII should be recalculated by omitting the resource allocations already done.

F. 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 by [31] and [32].

The values of the variables are expressed by sets of linguistic labels and 2-tuples [33], thus generalizing the model proposed in [30].

IV. Example and Discussion of Results

This section will explain in detail an example of application of the proposed aggregation operator. The distributed processing system, premises, resources, and processes running in the different nodes, groups, cardinals, criteria, and categories to evaluate the different loads and calculations needed, are those mentioned in [30], corresponding to steps A and B.

Calculation of the priorities or preferences of the processes taking the status of the node into account (they are calculated in each node for each process and could be called nodal priorities).

The valuation vectors are applied for each requirement of a resource made by a process, according to the criteria established for the determination of the priority that in each case and moment fixes the node in which the request occurs.

A. Expression of the Calculated Values in Terms of 2-Tuple Using a Set of Linguistic Labels

The valuations expressed in a linguistic format using the linguistic and semantic labels mentioned, with minimum, medium and maximum values, can be seen in Fig. 2 and Table X.

TABLE X. Proposals for Priority Assessment

EH: Extremely High	0.83	1.00	1.00
VH: Very High	0.67	0.83	1.00
H: High	0.50	0.67	0.83
M: Medium	0.33	0.50	0.67
L: Low	0.17	0.33	0.50
VL: Very Low	0.00	0.17	0.33
EL: Extremely Low	0.00	0.00	0.17

The next step is to transform these values into the 2-tuple format, considering the linguistic labels proposed above. Therefore, it will be necessary to compare each criterion value with the average value of each label, the minimum difference of this comparison will be the appropriate label.

The minimum difference between the differences of each criterion and the most representative value of each language label will be the most representative value.

The first element of the 2-tuple will be the linguistic value of that label, while the second element will be the difference between the value of the searched criterion and the average value of the selected label. This can be seen in Table XI, where "Process Priority" is represented by the methods considered traditional [2], [3] and [4].

As mentioned in the previous stage, each vector of evaluations of each requirement is scalar multiplied by the vector of weights corresponding to the current load category of the node to obtain the priority according to each criterion and the nodal priority granted to each requirement. This can be seen in Table XII.

TABLE XI. The Valuations Assigned to the Criteria to Calculate the Priority or Nodal Preference That Each Node Will Grant Each Requirement of Each Process According to the Node Load

Res./Proc.	Criteria				
	%CPU	...	Process Priority	...	%VM
$r_{11}p_{11}$	T(EB;0.0250)	...	T(EB;0.0800)	...	T(M;-0.0500)
$r_{12}p_{11}$	T(EB;0.0350)	...	T(EB;0.0300)	...	T(MB;0.0833)
$r_{21}p_{11}$	T(EB;0.0200)	...	T(MB;-0.0767)	...	T(MB;-0.0667)
$r_{22}p_{11}$	T(EB;0.0250)	...	T(EB;0.0800)	...	T(MB;0.0333)
$r_{23}p_{11}$	T(EB;0.0300)	...	T(MB;-0.0717)	...	T(B;0.0667)
$r_{24}p_{11}$	T(EB;0.0250)	...	T(EB;0.0600)	...	T(MB;-0.0667)
...	...	...	...	...	...
$r_{11}p_{13}$	T(EB;0.0350)	...	T(EB;0.0600)	...	T(M;-0.0500)
$r_{12}p_{13}$	T(EB;0.0400)	...	T(MB;-0.0767)	...	T(MB;0.0833)
$r_{13}p_{13}$	T(EB;0.0300)	...	T(MB;-0.0767)	...	T(MB;-0.0667)
$r_{21}p_{13}$	T(EB;0.0200)	...	T(EB;0.0500)	...	T(MB;0.0333)
$r_{22}p_{13}$	T(EB;0.0450)	...	T(EB;0.0500)	...	T(B;0.0667)
$r_{31}p_{13}$	T(EB;0.0350)	...	T(EB;0.0800)	...	T(MB;-0.0667)
$r_{32}p_{13}$	T(EB;0.0450)	...	T(EB;0.0400)	...	T(M;-0.0500)
$r_{33}p_{13}$	T(EB;0.0100)	...	T(MB;-0.0767)	...	T(MB;0.0833)
...	...	...	...	...	...
$r_{12}p_{23}$	T(MB;-0.0467)	...	T(MB;-0.0667)	...	T(EB;0.0300)
$r_{24}p_{23}$	T(EB;0.0400)	...	T(EB;0.0600)	...	T(EB;0.0700)
$r_{31}p_{23}$	T(EB;0.0200)	...	T(MB;-0.0267)	...	T(EB;0.0800)
$r_{32}p_{23}$	T(EB;0.0800)	...	T(EB;0.0800)	...	T(EB;0.0200)
$r_{33}p_{23}$	T(MB;-0.0467)	...	T(MB;0.0133)	...	T(EB;0.0200)
...	...	...	...	...	...
$r_{12}p_{34}$	T(B;-0.0333)	...	T(EB;0.0700)	...	T(MB;-0.0667)
$r_{13}p_{34}$	T(MB;0.0733)	...	T(EB;0.0800)	...	T(MB;-0.0067)
$r_{22}p_{34}$	T(MB;0.0133)	...	T(MB;-0.0767)	...	T(MB;-0.0067)
$r_{23}p_{34}$	T(MB;-0.0467)	...	T(MB;-0.0767)	...	T(EB;0.0800)
$r_{24}p_{34}$	T(MB;0.0133)	...	T(EB;0.0700)	...	T(MB;0.0133)
$r_{31}p_{34}$	T(MB;0.0133)	...	T(EB;0.0700)	...	T(MB;-0.0667)
$r_{32}p_{34}$	T(MB;0.0133)	...	T(EB;0.0600)	...	T(MB;-0.0667)
$r_{33}p_{34}$	T(B;-0.0333)	...	T(MB;-0.0767)	...	T(MB;-0.0667)
...	...	...	...	...	...
$r_{11}p_{37}$	T(MB;0.0133)	...	T(MB;-0.0767)	...	T(MB;-0.0467)
$r_{12}p_{37}$	T(B;-0.0633)	...	T(EB;0.0500)	...	T(MB;-0.0267)
$r_{21}p_{37}$	T(MB;0.0433)	...	T(EB;0.0600)	...	T(MB;-0.0467)
$r_{32}p_{37}$	T(B;-0.0633)	...	T(EB;0.0800)	...	T(EB;0.0600)
$r_{33}p_{37}$	T(MB;-0.0467)	...	T(EB;0.0800)	...	T(MB;-0.0067)

TABLE XII. The Valuations Assigned to the Criteria to Calculate the Priority or Nodal Preference That Each Node Will Grant Each Requirement of Each Process According to the Node Load

Resources/ Processes	Criteria				
	%CPU	...	Process Priority	...	%VM
$r_{11}p_{11}$	T(M;0.000)	...	T(VH;-0.0333)	...	T(VH;0.0667)
$r_{12}p_{11}$	T(H;0.0333)	...	T(L;-0.0333)	...	T(M;0.0000)
$r_{21}p_{11}$	T(L;0.0667)	...	T(VH;0.0667)	...	T(VL;0.0333)
$r_{22}p_{11}$	T(M;0.0000)	...	T(VH;-0.0333)	...	T(L;0.0667)
$r_{23}p_{11}$	T(H;-0.0667)	...	T(EH;-0.05)	...	T(VH;-0.0333)
$r_{24}p_{11}$	T(M;0.0000)	...	T(H;-0.0667)	...	T(VL;0.0333)
...	...	...	...	...	...
$r_{11}p_{13}$	T(H;0.0333)	...	T(H;-0.0667)	...	T(VH;-0.0333)
$r_{12}p_{13}$	T(VH;-0.0333)	...	T(VH;0.0667)	...	T(L;0.0667)
$r_{13}p_{13}$	T(H;-0.0667)	...	T(VH;0.0667)	...	T(VH;-0.0333)
$r_{21}p_{13}$	T(L;0.0667)	...	T(M;0.0000)	...	T(L;-0.0333)
$r_{22}p_{13}$	T(VH;0.0667)	...	T(M;0.0000)	...	T(L;-0.0333)
$r_{31}p_{13}$	T(H;0.0333)	...	T(VH;-0.0333)	...	T(H;-0.0667)
$r_{32}p_{13}$	T(VH;0.0667)	...	T(L;0.0667)	...	T(H;-0.0667)
$r_{33}p_{13}$	T(VL;0.0333)	...	T(VH;0.0667)	...	T(H;-0.0667)
...	...	...	...	...	...
$r_{12}p_{23}$	T(H;-0.0667)	...	T(M;0.0000)	...	T(L;-0.0333)
$r_{24}p_{23}$	T(VL;0.0333)	...	T(L;-0.0333)	...	T(H;0.0333)
$r_{31}p_{23}$	T(VL;-0.0667)	...	T(H;0.0333)	...	T(VH;-0.0333)
$r_{32}p_{23}$	T(L;0.0667)	...	T(L;0.0667)	...	T(VL;0.0333)
$r_{33}p_{23}$	T(H;-0.0667)	...	T(VH;0.0667)	...	T(VL;0.0333)
...	...	...	...	...	...
$r_{12}p_{34}$	T(EH;0.0000)	...	T(H;0.0333)	...	T(M;0.0000)
$r_{13}p_{34}$	T(EH;0.0000)	...	T(VH;-0.0333)	...	T(M;0.0000)
$r_{22}p_{34}$	T(VH;-0.0333)	...	T(VH;0.0667)	...	T(VH;-0.0333)
$r_{23}p_{34}$	T(H;-0.0667)	...	T(VH;0.0667)	...	T(VH;-0.0333)
$r_{24}p_{34}$	T(L;0.0667)	...	T(H;0.0333)	...	T(L;0.0667)
$r_{31}p_{34}$	T(H;-0.0667)	...	T(H;0.0333)	...	T(VH;0.0667)
$r_{32}p_{34}$	T(H;-0.0667)	...	T(H;-0.0667)	...	T(M;0.0000)
$r_{33}p_{34}$	T(H;-0.0667)	...	T(VH;0.0667)	...	T(M;0.0000)
...	...	...	...	...	...
$r_{11}p_{37}$	T(H;-0.0667)	...	T(VH;0.0667)	...	T(H;-0.0667)
$r_{12}p_{37}$	T(VH;0.0667)	...	T(M;0.0000)	...	T(H;0.0333)
$r_{21}p_{37}$	T(H;0.0333)	...	T(H;-0.0667)	...	T(H;-0.0667)
$r_{32}p_{37}$	T(VH;0.0667)	...	T(VH;-0.0333)	...	T(L;-0.0333)
$r_{33}p_{37}$	T(L;0.0667)	...	T(VH;-0.0333)	...	T(VH;-0.0333)

In summary, the nodal priority (to be calculated at the node where the request occurs) of a process to access a given resource (which can be at any node) is calculated by the scalar product of the vectors mentioned above.

This nodal priority must be transformed into the 2-tuple format, considering the linguistic labels already mentioned. Therefore, it will be necessary to compare each nodal priority value with the average value of each label, the minimum difference of these comparisons will indicate the corresponding label.

B. Calculation of the Priorities or Preferences of the Processes to Access the Shared Resources Available (Calculated in the Centralized Resource Manager) and Determining the Order in Which the Resources Will Be Allocated, and Which Process Each Resource Will Be Assigned

Table XIII and Table XIV are used to calculate the final priorities, in which the priorities or nodal preferences calculated in the previous stage are placed; in this table each row contains information of the nodal priorities of the different processes to access a certain resource.

TABLE XIII. Nodal Priorities of the Processes to Access Each Resource in 2-Tuple (P_{11}, P_{13}, P_{23})

Resources	Nodal Process Priorities				
	p_{11}	...	p_{13}	...	p_{23}
r_{11}	NPT(H;0.0483)	...	NPT(H;0.0483)	...	-
r_{12}	NPT(M;-0.0050)	...	NPT(M;0.0350)	...	NPT(H;-0.070)
r_{13}	-	...	NPT(H;0.0733)	...	-
r_{21}	NPT(L;0.0217)	...	NPT(M;-0.060)	...	-
r_{22}	NPT(M;-0.0150)	...	NPT(M;-0.050)	...	-
r_{23}	NPT(VH;-0.0483)	...	-	...	-
r_{24}	NPT(L;0.0717)	...	-	...	NPT(L;-0.0008)
r_{31}	-	...	NPT(H;-0.0370)	...	NPT(M;-0.0400)
r_{33}	-	...	NPT(H;-0.070)	...	NPT(M;0.0200)
r_{33}	-	...	NPT(H;-0.0517)	...	NPT(H;-0.0317)

TABLE XIV. Nodal Priorities of the Processes to Access Each Resource in 2-Tuple (P_{34}, P_{37})

Resources	Nodal Process Priorities				
	...	p_{34}	...	p_{37}	...
r_{11}	...	-	...	NPT(H;0.0533)	...
r_{12}	...	NPT(H;0.0058)	...	NPT(VH;-0.0733)	...
r_{13}	...	NPT(H;0.0658)	...	-	...
r_{21}	...	-	...	NPT(H;0.0283)	...
r_{22}	...	NPT(VH;-0.0708)	...	-	...
r_{23}	...	NPT(H;0.0083)	...	-	...
r_{24}	...	NPT(M;0.0700)	...	-	...
r_{31}	...	NPT(H;0.0258)	...	-	...
r_{33}	...	NPT(H;0.0083)	...	NPT(H;-0.0192)	...
r_{33}	...	NPT(H;-0.0492)	...	NPT(H;-0.0541)	...

Next, the final weight vector to be used in the final aggregation process to determine the order or priority of access to resources must be calculated. In addition, the recently obtained weights will have to be normalized by dividing each one by the sum of all of them.

The nodal priorities indicated in Table XIII and Table XIV taken row by row, i.e., for each resource, will be multiplied by the final standardized weight vector mentioned above. See in Table XV and Table XVI.

TABLE XV. Final Global Priority Tuple (P_{11}, P_{13}, P_{23})

Resources	Nodal Process Priorities				
	p_{11}	...	p_{13}	...	p_{23}
r_{11}	NPT(EL;0.069)	...	NPT(EL;0.069)	...	-
r_{12}	NPT(EL;0.048)	...	NPT(EL;0.052)	...	NPT(EL;0.058)
r_{13}	-	...	NPT(EL;0.072)	...	-
r_{21}	NPT(EL;0.034)	...	NPT(EL;0.043)	...	-
r_{22}	NPT(EL;0.047)	...	NPT(EL;0.044)	...	-
r_{23}	NPT(EL;0.076)	...	-	...	-
r_{24}	NPT(EL;0.039)	...	-	...	NPT(EL;0.032)
r_{31}	-	...	NPT(EL;0.061)	...	NPT(EL;0.045)
r_{33}	-	...	NPT(EL;0.058)	...	NPT(EL;0.050)
r_{33}	-	...	NPT(EL;0.06)	...	NPT(EL;0.061)

The next step is to normalize Table XV and Table XVI between the extreme values. To do this, subtract the numerical value of the 2-tuple from the minimum value of both tables and divide it by the range, which is the difference between the maximum and minimum values of the two. As can see in Table XVII and Table XVIII.

The largest of these products made for the different processes in relation to the same resource will indicate which of the processes will have access to the resource.

TABLE XVI. Final Global Priority Tuple (P_{34}, P_{37})

Resources	...	Nodal Process Priorities p_{34}	...	p_{37}
r_{11}	...	-	...	NPT(EL;0.07)
r_{12}	...	NPT(EL;0.065)	...	NPT(EL;0.074)
r_{13}	...	NPT(EL;0.071)	...	-
r_{21}	...	-	...	NPT(EL;0.067)
r_{22}	...	NPT(EL;0.074)	...	-
r_{23}	...	NPT(EL;0.065)	...	-
r_{24}	...	NPT(EL;0.055)	...	-
r_{31}	...	NPT(EL;0.067)	...	-
r_{33}	...	NPT(EL;0.065)	...	NPT(EL;0.063)
r_{33}	...	NPT(EL;0.060)	...	NPT(EL;0.059)

TABLE XVII. Normalized Final Global Priority Tuple (P_{11}, P_{13}, P_{23})

Resources	Nodal Process Priorities p_{11}	...	p_{13}	...	p_{23}
r_{11}	NFGPT(VH;0.06)	...	NFGPT(VH;0.06)	...	-
r_{12}	NFGPT(M;0.060)	...	NFGPT(H;-0.046)	...	NFGPT(VH;-0.052)
r_{13}	-	...	**NFGPT(EH;-0.068)**	...	-
r_{21}	NFGPT(L;0.014)	...	NFGPT(M;-0.023)	...	-
r_{22}	NFGPT(M;0.045)	...	NFGPT(M;-0.008)	...	-
r_{23}	**NFGPT(EH;0.00)**	...	-	...	-
r_{24}	NFGPT(M;-0.076)	...	-	...	NFGPT(L;0.007)
r_{31}	-	...	NFGPT(VH;-0.068)	...	NFGPT(M;0.051)
r_{33}	-	...	NFGPT(H;0.053)	...	NFGPT(H;-0.017)
r_{33}	-	...	NFGPT(H;0.075)	...	**NFGPT(VH;-0.061)**

TABLE XVIII. Normalized Final Global Priority Tuple (P_{34}, P_{37})

Resources	...	Nodal Process Priorities p_{34}	...	p_{37}
r_{11}	...	-	...	**NFGPT(VH;0.068)**
r_{12}	...	NFGPT(VH;-0.004)	...	**NFGPT(EH;-0.038)**
r_{13}	...	NFGPT(EH;-0.080)	...	-
r_{21}	...	-	...	**NFGPT(VH;0.030)**
r_{22}	...	**NFGPT(EH;-0.034)**	...	-
r_{23}	...	NFGPT(VH;-0.00)	...	-
r_{24}	...	**NFGPT(H;0.007)**	...	-
r_{31}	...	**NFGPT(VH;0.026)**	...	-
r_{33}	...	**NFGPT(VH;-0.00)**	...	NFGPT(VH;-0.042)
r_{33}	...	NFGPT(H;0.079)	...	NFGPT(H;0.072)

The summation of all these products in relation to the same resource will indicate the priority that this resource will have to be assigned, in relation to the other resources that will also have to be assigned. This constitutes the Linguistics Distributed System Assignment Function (*LDSAF*). Refer to (3).

$$LDSAF(r_{ij}) = \Sigma\ NFGPT_{ijkl} = r_{ij} \text{ resource assignment priority} \quad (3)$$

By calculating the *LDSAF* for all resources, a 2-tuple vector will be obtained and, by ordering its elements from highest to lowest, the priority order of resource allocation will be obtained, which should be normalized ensuring that the 2-tuples obtained are in the interval [0, 1]. As can be seen in Table XIX.

TABLE XIX. Valuations to Normalize the *LDSAF* (First Iteration)

Label	Value
Maximum Value	5.2364
Minimum value	1.8255
Range	3.4109

In addition, as indicated above, the largest of the $NFGPT_{ijkl}$ for each resource will indicate the process to which the resource will be assigned.

The result of normalizing the 2 tuples constitutes what will be called Normalized Linguistics Distributed System Assignment Function a (*NLDSAF*). Refer to (4)

$$NLDSAF(r_{ij}) = \Sigma\ (NFGPT_{ijkl} / (\text{Maximum } (NFGPT_{ijkl}) - \text{Minimum } (NFGPT_{ijkl}))) = r_{ij} \text{ resource assignment priority normalized between extreme values} \quad (4)$$

This can be seen in Table XX.

TABLE XX. Normalized Linguistics Distributed System Assignment Function Ordered By Highest Priority Resource (First Iteration)

Assignment order of resources	Priority of the resource	Process at which assign the resource
r_{33}	T(EH;0.0000)	p_{23}
r_{12}	T(EH;-0.0326)	p_{37}
r_{31}	T(H;-0.0028)	p_{34}
r_{11}	T(H;-0.0355)	p_{37}
r_{22}	T(M;0.0423)	p_{34}
r_{21}	T(M;0.0337)	p_{37}
r_{13}	T(M;0.0274)	p_{13}
r_{32}	T(M;-0.0334)	p_{34}
r_{23}	T(L;-0.0591)	p_{11}
r_{24}	T(EL;0.0000)	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. Table XIX should be recalculated by omitting the resource allocations already done. As can see in Fig. 3.

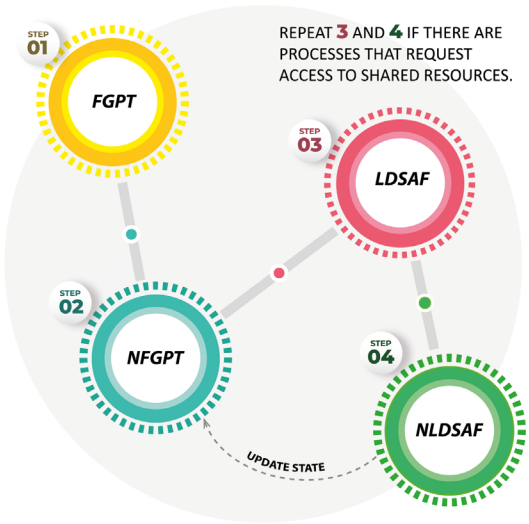

Fig. 3. Process of self-regulation and calculation of *NLDSAF*.

Since the system regulates itself by releasing the resources already assigned to the processes in the previous step, and because there are resource requests from the processes that have not yet been satisfied, the calculations in Table XIX and XX are repeated with their respective values, omitting the processes already completed.

Normalized the 2-tuples in the second iteration can be seen in Table XXI.

TABLE XXI. VALUATIONS TO NORMALIZE THE *LDSAF* (SECOND ITERATION)

Label	Value
Maximum Value	4.8503
Minimum value	1.2514
Range	3.5989

The result of normalizing the 2-tuples for the second iteration can be seen in Table XXII.

TABLE XXII. NORMALIZED LINGUISTICS DISTRIBUTED SYSTEM ASSIGNMENT FUNCTION ORDERED BY HIGHEST PRIORITY RESOURCE (SECOND ITERATION)

Assignment order of resources	Priority of the resource	Process at which assign the resource
r_{33}	T(EH;0.0000)	p_{34}
r_{12}	T(VH;0.0758)	p_{34}
r_{31}	T(H;-0.0392)	p_{13}
r_{11}	T(M;0.0813)	p_{11}
r_{21}	T(M;-0.0076)	p_{25}
r_{22}	T(M;-0.0298)	p_{11}
r_{13}	T(M;-0.0347)	p_{34}
r_{32}	T(M;-0.0676)	p_{37}
r_{23}	T(VL;0.0173)	p_{34}
r_{24}	T(EL;0.0000)	p_{11}

The final tables with the results of all the iterations will be shown below.

The valuations to normalize the *LDSAF* of each iteration can be seen in Table XXIII.

TABLE XXIII. VALUATIONS TO NORMALIZE THE *LDSAF* (OF EACH ITERATION).

Maximum Value	Minimum value	Range	Iteration
5.2364	1.8255	3.4109	1
4.8503	1.2514	3.5989	2
4.1598	0.8147	3.3451	3
4.0308	0.5616	3.4692	4
3.9955	0.4695	3.526	5
3.6488	0.2681	3.3807	6
2.6775	0.0000	2.6775	7
2.4275	0.0000	2.4275	8
2.9932	0.0000	2.9932	9
1.5946	0.0000	1.5946	10
1.0000	0.0000	1.0000	11
0.0169	0.0000	0.0169	12

The result of normalizing the 2 tuples for each iteration can be seen in Table XXIV, ordered by highest priority resource (of each iteration).

V. EXAMPLE OF A DECISION MODEL APPLIED TO ONE OF THE TRADITIONAL ALGORITHMS

A particular case of the proposed decision model is to visualize how some of the methods considered traditional in this work, are a particular case of this method.

As the traditional methods do not consider groups of processes, the calculation will only be done with independent processes and the column that considers whether a process is part of a group of processes should be disabled, see Table XXV.

TABLE XXV. WEIGHTS ASSIGNED TO THE PROCESSES FOR THE CALCULATION OF PRIORITIES

Processes	Group of processes	Independent processes
p_{11}	-	$wf_{11}=1/np$
....	-	....
p_{kl}	-	$wf_{kl}=1/np$
....	-	
p_{np}	-	$wf_{np}=1/np$

The methods considered traditional do not consider most of the criteria contemplated in the proposed model (in addition to not considering the representation by means of linguistic labels or 2-tuples), they are only based on the calculation of the "Process Priority" criterion. The weights assigned to the criteria in Table XXVI, to calculate the global priority, only the "Process Priority" criterion will be considered, disabling the other criteria.

For each requirement of a resource made by a process, the assessment vectors are applied according to the criteria established for the determination of the priority. This is done in the node where the requirement occurs. To obtain the node priority, each rating vector of each requirement must be scaled and multiplied by the weight vector corresponding to the current load category of the node.

TABLE XXVI. WEIGHTS ASSIGNED TO THE CRITERIA FOR CALCULATING PRIORITY

Categories	Process priority	Other Criteria
High	0.1000	-
Medium	0.2000	-
Low	0.1000	-

Although the decision model obtains the information of all the criteria, it should be noted that for the traditional methods, from the weight vector Table XXVI, only the criterion "Process Priority" will affect the calculation of the priority.

The valuations assigned to the criteria for calculating the priority or preference that each node will give to each requirement of each process according to the node load, will be those used in TABLE XII. To calculate the priorities or preferences of the processes, taking into account the state of the node, Table XX will be used, but disabling all the criteria, except "Process Priority". Nodal priorities, final weights and overall process priorities for accessing resources must be calculated.

Table XXVII, Table XXVIII, Table XXIX, Table XXX and Table XXXI are constructed from the nodal priority values, which for this example matches the "Process Priority" criteria.

TABLE XXIV. Concatenated Normalized Linguistics Distributed System Assignment Function (*CNLDSAF*)

Assignment order of resources	Priority of the resource	Process at which assign the resource	Iteration
r_{33}	T(EH;0.0000)	p_{23}	1
r_{12}	T(EH;-0.0326)	p_{37}	1
r_{31}	T(H;-0.0028)	p_{34}	1
r_{11}	T(H;-0.0355)	p_{37}	1
r_{22}	T(M;0.0423)	p_{34}	1
r_{21}	T(M;0.0337)	p_{37}	1
r_{13}	T(M;0.0274)	p_{13}	1
r_{32}	T(M;-0.0334)	p_{34}	1
r_{23}	T(L;-0.0591)	p_{11}	1
r_{24}	T(EL;0.0000)	p_{34}	1
r_{33}	T(EH;0.0000)	p_{34}	2
r_{12}	T(VH;0.0758)	p_{34}	2
r_{31}	T(H;-0.0392)	p_{13}	2
r_{11}	T(M;0.0813)	p_{11}	2
r_{21}	T(M;-0.0076)	p_{25}	2
r_{22}	T(M;-0.0298)	p_{11}	2
r_{13}	T(M;-0.0347)	p_{34}	2
r_{32}	T(M;-0.0676)	p_{37}	2
r_{23}	T(VL;0.0173)	p_{34}	2
r_{24}	T(EL;0.0000)	p_{11}	2
r_{33}	T(EH;0.0000)	p_{34}	3
r_{12}	T(VH;0.0758)	p_{34}	3
r_{31}	T(H;-0.0392)	p_{13}	3
r_{11}	T(M;0.0813)	p_{11}	3
r_{21}	T(M;-0.0076)	p_{25}	3
r_{22}	T(M;-0.0298)	p_{11}	3
r_{13}	T(M;-0.0347)	p_{34}	3
r_{32}	T(M;-0.0676)	p_{37}	3
r_{23}	T(VL;0.0173)	p_{34}	3
r_{24}	T(EL;0.0000)	p_{11}	3
r_{33}	T(EH;0.0000)	p_{37}	4
r_{12}	T(VH;0.0255)	p_{13}	4
r_{31}	T(H;-0.0739)	p_{23}	4
r_{22}	T(M;-0.0178)	p_{13}	4
r_{21}	T(M;-0.0721)	p_{12}	4
r_{11}	T(L;0.0082)	p_{12}	4
r_{13}	T(L;-0.0337)	p_{21}	4
r_{32}	T(L;-0.0762)	p_{23}	4
r_{23}	T(EL;0.0489)	p_{32}	4
r_{24}	T(EL;0.0000)	p_{35}	4
r_{33}	T(EH;0.0000)	p_{12}	5
r_{12}	T(VH;0.0431)	p_{11}	5
r_{31}	T(H;-0.0777)	p_{31}	5
r_{22}	T(M;-0.0467)	p_{12}	5
r_{21}	T(L;0.0674)	p_{22}	5
r_{13}	T(L;-0.0824)	p_{32}	5
r_{11}	T(VL;0.0273)	p_{32}	5
r_{32}	T(VL;-0.0591)	p_{36}	5
r_{23}	T(EL;0.0230)	p_{33}	5
r_{24}	T(EL;0.0000)	p_{36}	5
r_{33}	T(EH;0.0000)	p_{31}	6
r_{12}	T(VH;-0.0133)	p_{12}	6
r_{31}	T(M;0.0638)	p_{12}	6
r_{22}	T(L;0.0275)	p_{21}	6
r_{21}	T(L;-0.0106)	p_{11}	6
r_{13}	T(VL;0.0815)	p_{36}	6
r_{11}	T(VL;0.0443)	p_{36}	6
r_{32}	T(VL;-0.0604)	p_{35}	6
r_{23}	T(EL;0.0032)	p_{24}	6
r_{24}	T(EL;0.0000)	p_{24}	6
r_{33}	T(EH;0.0000)	p_{21}	7
r_{12}	T(VH;-0.0348)	p_{21}	7
r_{31}	T(M;0.0040)	p_{22}	7
r_{13}	T(VL;0.0742)	p_{35}	7
r_{21}	T(VL;0.0682)	p_{33}	7
r_{11}	T(VL;0.0631)	p_{33}	7
r_{22}	T(VL;0.0580)	p_{35}	7
r_{32}	T(VL;-0.0553)	p_{33}	7
r_{33}	T(EH;0.0000)	p_{22}	8
r_{12}	T(EH;-0.0210)	p_{33}	8
r_{31}	T(M;-0.0299)	p_{36}	8
r_{21}	T(VL;0.0191)	p_{36}	8
r_{13}	T(VL;0.0110)	p_{33}	8
r_{22}	T(VL;0.0078)	p_{33}	8
r_{11}	T(VL;-0.0035)	p_{24}	8
r_{12}	T(EH;0.0000)	p_{36}	9
r_{33}	T(VH;-0.0117)	p_{33}	9
r_{31}	T(L;-0.0534)	p_{35}	9
r_{22}	T(EL;0.0000)	p_{36}	9
r_{12}	T(EH;0.0000)	p_{24}	10
r_{33}	T(VH;0.0734)	p_{35}	10
r_{12}	T(EH;0.0000)	p_{32}	11
r_{33}	T(H;-0.0196)	p_{36}	11
r_{12}	T(EH;0.0000)	p_{35}	12

TABLE XXVII. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{11}, p_{12}, p_{13})

	p_{11}	p_{12}	p_{13}
r_{11}	NPT(EL;0.0800)	NPT(VL;-0.0667)	NPT(EL;0.0600)
r_{12}	NPT(EL;0.0300)	NPT(EL;0.0800)	NPT(VL;-0.0767)
r_{13}	-	-	NPT(VL;-0.0767)
r_{21}	NPT(VL;-0.0767)	NPT(EL;0.0800)	NPT(EL;0.0500)
r_{22}	NPT(EL;0.0800)	NPT(EL;0.0800)	NPT(EL;0.0500)
r_{23}	NPT(VL;-0.0717)	-	-
r_{24}	NPT(EL;0.0600)	-	-
r_{31}	-	NPT(EL;0.0300)	NPT(EL;0.0800)
r_{32}	-	-	NPT(EL;0.0400)
r_{33}	-	NPT(EL;0.0300)	NPT(VL;-0.0767)

TABLE XXVIII. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{21}, p_{22}, p_{23})

	p_{21}	p_{22}	p_{23}
r_{11}	-	-	-
r_{12}	NPT(VL;-0.0267)	-	NPT(VL;-0.0667)
r_{13}	NPT(VL;-0.0267)	-	-
r_{21}	-	NPT(VL;-0.0467)	-
r_{22}	NPT(VL;-0.0667)	NPT(VL;0.0133)	-
r_{23}	NPT(VL;-0.0667)	-	-
r_{24}	-	-	NPT(EL;0.0600)
r_{31}	NPT(VL;0.0133)	NPT(EL;0.0800)	NPT(VL;-0.0267)
r_{32}	-	-	NPT(EL;0.0800)
r_{33}	NPT(EL;0.0800)	NPT(VL;-0.0067)	NPT(VL;0.0133)

TABLE XXIX. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{24}, p_{25}, p_{31})

	p_{24}	p_{25}	p_{31}
r_{11}	NPT(VL;-0.0667)	-	-
r_{12}	NPT(VL;0.0133)	-	-
r_{13}	-	-	NPT(EL;0.0700)
r_{21}	-	NPT(EL;0.0800)	-
r_{22}	-	-	-
r_{23}	NPT(VL;-0.0067)	-	-
r_{24}	NPT(EL;0.0600)	-	-
r_{31}	-	-	NPT(EL;0.0700)
r_{32}	-	-	-
r_{33}	-	-	NPT(VL;-0.0767)

TABLE XXX. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{32}, p_{33}, p_{34})

	p_{32}	p_{33}	p_{34}
r_{11}	NPT(VL;-0.0767)	NPT(EL;0.0600)	-
r_{12}	NPT(EL;0.0800)	NPT(EL;0.0300)	NPT(EL;0.0700)
r_{13}	NPT(VL;-0.0767)	NPT(EL;0.0300)	NPT(EL;0.0800)
r_{21}	-	NPT(EL;0.0800)	-
r_{22}	-	NPT(EL;0.0700)	NPT(VL;-0.0767)
r_{23}	NPT(EL;0.0600)	NPT(EL;0.0600)	NPT(VL;-0.0767)
r_{24}	-	-	NPT(EL;0.0700)
r_{31}	-	-	NPT(EL;0.0700)
r_{32}	-	NPT(EL;0.0400)	NPT(EL;0.0600)
r_{33}	-	NPT(EL;0.0600)	NPT(VL;-0.0767)

TABLE XXXI. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{35}, p_{36}, p_{37})

	p_{35}	p_{36}	p_{37}
r_{11}	-	NPT(EL;0.0800)	NPT(VL;-0.0767)
r_{12}	NPT(VL;-0.0767)	NPT(EL;0.0800)	NPT(EL;0.0500)
r_{13}	NPT(VL;-0.0767)	NPT(EL;0.0800)	-
r_{21}	-	NPT(EL;0.0700)	NPT(EL;0.0600)
r_{22}	NPT(EL;0.0800)	NPT(EL;0.0800)	-
r_{23}	-	-	-
r_{24}	NPT(EL;0.0600)	NPT(EL;0.0800)	-
r_{31}	NPT(EL;0.0800)	NPT(EL;0.0800)	-
r_{32}	NPT(EL;0.0400)	NPT(EL;0.0800)	NPT(EL;0.0800)
r_{33}	NPT(EL;0.0800)	NPT(EL;0.0800)	NPT(EL;0.0800)

As mentioned, traditional methods do not consider groups of processes. The calculation will only consider that the processes are independent. In the example there are 15 processes and the calculation of the weights is wf_{ij} equal to $1/np$ for independent processes, where np is the number of processes in the system (15), the calculation for the weights of each process (wp_{ij}) is equal to 1/15. For the calculation of the standardized weights (nwp_{ij}) each wp_{ij} value is divided by the sum of all wp_{ij}, this can be seen in the tables above. The final weight vector to be used in the final aggregation process should be calculated to determine the order or priority of access to resources. In addition, the recently obtained weights should be normalized by dividing each one of them by the sum of all of them. For this particular situation, all processes will have the same weight value, since they are only considered as independent processes. The normalized weight vector will have the same value for all processes, this value will be 1/15, which results in 0.0066.

The nodal priorities indicated in Table XXVII, Table XXVIII, Table XXIX, Table XXX and Table XXXI taken row by row, that is, for each resource, will be multiplied by the normalized weight vector (nwp_{ij}). This can be seen in Table XXXII, Table XXXIII, XXXIV, Table XXXV and Table XXXVI.

TABLE XXXII. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{11}, p_{12}, p_{13})

	p_{11}	p_{12}	p_{13}
r_{11}	NPT(EL;0.005)	NPT(EL;0.007)	NPT(EL;0.004)
r_{12}	NPT(EL;0.002)	NPT(EL;0.005)	NPT(EL;0.006)
r_{13}	-	-	NPT(EL;0.006)
r_{21}	NPT(EL;0.006)	NPT(EL;0.005)	NPT(EL;0.003)
r_{22}	NPT(EL;0.005)	NPT(EL;0.005)	NPT(EL;0.003)
r_{23}	NPT(EL;0.006)	-	-
r_{24}	NPT(EL;0.004)	-	-
r_{31}	-	NPT(EL;0.002)	NPT(EL;0.005)
r_{32}	-	-	NPT(EL;0.003)
r_{33}	-	NPT(EL;0.002)	NPT(EL;0.006)

TABLE XXXIII. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{21}, p_{22}, p_{23})

	p_{21}	p_{22}	p_{23}
r_{11}	-	-	-
r_{12}	NPT(EL;0.009)	-	NPT(EL;0.007)
r_{13}	NPT(EL;0.009)	-	-
r_{21}	-	NPT(EL;0.008)	-
r_{22}	NPT(EL;0.007)	NPT(EL;0.012)	-
r_{23}	NPT(EL;0.007)	-	-
r_{24}	-	-	NPT(EL;0.004)
r_{31}	NPT(EL;0.012)	NPT(EL;0.005)	NPT(EL;0.009)
r_{32}	-	-	NPT(EL;0.005)
r_{33}	NPT(EL;0.005)	NPT(EL;0.011)	NPT(EL;0.012)

TABLE XXXIV. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{24}, p_{25}, p_{31})

	p_{24}	p_{25}	p_{31}
r_{11}	NPT(EL;0.007)	-	-
r_{12}	NPT(EL;0.012)	-	-
r_{13}	-	-	NPT(EL;0.005)
r_{21}	-	NPT(EL;0.005)	-
r_{22}	-	-	-
r_{23}	NPT(EL;0.011)	-	-
r_{24}	NPT(EL;0.004)	-	-
r_{31}	-	-	NPT(EL;0.005)
r_{32}	-	-	-
r_{33}	-	-	NPT(EL;0.006)

TABLE XXXV. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{32}, p_{33}, p_{34})

	p_{32}	p_{33}	p_{34}
r_{11}	NPT(EL;0.006)	NPT(EL;0.004)	-
r_{12}	NPT(EL;0.005)	NPT(EL;0.002)	NPT(EL;0.005)
r_{13}	NPT(EL;0.006)	NPT(EL;0.002)	NPT(EL;0.005)
r_{21}	-	NPT(EL;0.005)	-
r_{22}	-	NPT(EL;0.005)	NPT(EL;0.006)
r_{23}	NPT(EL;0.004)	NPT(EL;0.004)	NPT(EL;0.006)
r_{24}	-	-	NPT(EL;0.005)
r_{31}	-	-	NPT(EL;0.005)
r_{32}	-	NPT(EL;0.003)	NPT(EL;0.004)
r_{33}	-	NPT(EL;0.004)	NPT(EL;0.006)

TABLE XXXVI. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE (p_{35}, p_{36}, p_{37})

	p_{35}	p_{36}	p_{37}
r_{11}	-	NPT(EL;0.005)	NPT(EL;0.006)
r_{12}	NPT(EL;0.006)	NPT(EL;0.005)	NPT(EL;0.003)
r_{13}	NPT(EL;0.006)	NPT(EL;0.005)	-
r_{21}	-	NPT(EL;0.005)	NPT(EL;0.004)
r_{22}	NPT(EL;0.005)	NPT(EL;0.005)	-
r_{23}	-	-	-
r_{24}	NPT(EL;0.004)	NPT(EL;0.005)	-
r_{31}	NPT(EL;0.005)	NPT(EL;0.005)	-
r_{32}	NPT(EL;0.003)	NPT(EL;0.005)	NPT(EL;0.005)
r_{33}	NPT(EL;0.005)	NPT(EL;0.005)	NPT(EL;0.005)

The next step is to normalize Table XXXII, Table XXXIII, Table XXXIV, Table XXXV and Table XXXVI between the extreme values. To do this, you must subtract the numerical value of the 2-tuple by the minimum value of all of them and divide it by the range. The range is the difference between the maximum value and minimum value of the tables already mentioned. This can be seen in Table XXXVII.

TABLE XXXVII. NORMALIZATION ASSESSMENTS

Label	Value
Maximum Value	0.012
Minimum value	0.002
Range	0.010

The result of this standardization can be seen in Table XXXVIII, Table XXXIX, Table XL, Table XLI and Table XLII. The largest of these products made for the different processes in relation to the same resource will indicate which of the processes will have access to the resource.

TABLE XXXVIII. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE NORMALIZED (p_{11}, p_{12}, p_{13})

	p_{11}	p_{12}	p_{13}
r_{11}	NFGPT(L;0.000)	NFGPT(M;-0.033)	NFGPT(VL;0.033)
r_{12}	NFGPT(EL;0.000)	NFGPT(L;0.000)	NFGPT(L;0.067)
r_{13}	-	-	NFGPT(L;0.067)
r_{21}	NFGPT(L;0.067)	NFGPT(L;0.000)	NFGPT(VL;-0.033)
r_{22}	NFGPT(L;0.000)	NFGPT(L;0.000)	NFGPT(VL;-0.033)
r_{23}	NFGPT(M;-0.067)	-	-
r_{24}	NFGPT(VL;0.033)	-	-
r_{31}	-	NFGPT(EL;0.000)	NFGPT(L;0.000)
r_{32}	-	-	NFGPT(EL;0.067)
r_{33}	-	NFGPT(EL;0.000)	NFGPT(L;0.067)

TABLE XXXIX. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE NORMALIZED (p_{21}, p_{22}, p_{23})

	p_{21}	p_{22}	p_{23}
r_{11}	-	-	-
r_{12}	NFGPT(H;0.067)	-	NFGPT(M;-0.033)
r_{13}	NFGPT(H;0.067)	-	-
r_{21}	-	NFGPT(H;-0.067)	-
r_{22}	NFGPT(M;-0.033)	NFGPT(EH;0.000)	-
r_{23}	NFGPT(M;-0.033)	-	-
r_{24}	-	-	NFGPT(VL;0.033)
r_{31}	NFGPT(EH;0.000)	NFGPT(L;0.000)	NFGPT(H;0.067)
r_{32}	-	-	NFGPT(L;0.000)
r_{33}	NFGPT(L;0.000)	NFGPT(VH;0.033)	NFGPT(EH;0.000)

TABLE XL. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE NORMALIZED (p_{24}, p_{25}, p_{31})

	p_{24}	p_{25}	p_{31}
r_{11}	NFGPT(M;-0.033)	-	-
r_{12}	NFGPT(EH;0.000)	-	-
r_{13}	-	-	NFGPT(L;-0.067)
r_{21}	-	NFGPT(L;0.000)	-
r_{22}	-	-	-
r_{23}	NFGPT(VH;0.033)	-	-
r_{24}	NFGPT(VL;0.033)	-	-
r_{31}	-	-	NFGPT(L;-0.067)
r_{32}	-	-	-
r_{33}	-	-	NFGPT(L;0.067)

TABLE XLI. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE NORMALIZED (p_{32}, p_{33}, p_{34})

	p_{32}	p_{33}	p_{34}
r_{11}	NFGPT(L;0.067)	NFGPT(VL;0.033)	-
r_{12}	NFGPT(L;0.000)	NFGPT(EL;0.000)	NFGPT(L;-0.067)
r_{13}	NFGPT(L;0.067)	NFGPT(EL;0.000)	NFGPT(L;0.000)
r_{21}	-	NFGPT(L;0.000)	-
r_{22}	-	NFGPT(L;-0.067)	NFGPT(L;0.067)
r_{23}	NFGPT(VL;0.033)	NFGPT(VL;0.033)	NFGPT(L;0.067)
r_{24}	-	-	NFGPT(L;-0.067)
r_{31}	-	-	NFGPT(L;-0.067)
r_{32}	-	NFGPT(EL;0.067)	NFGPT(VL;0.033)
r_{33}	-	NFGPT(VL;0.033)	NFGPT(L;0.067)

TABLE XLII. NODAL PRIORITIES OF THE PROCESSES TO ACCESS EACH RESOURCE IN 2-TUPLE NORMALIZED (p_{35}, p_{36}, p_{37})

	p_{35}	p_{36}	p_{37}
r_{11}	-	NFGPT(L;0.000)	NFGPT(L;0.067)
r_{12}	NFGPT(L;0.067)	NFGPT(L;0.000)	NFGPT(VL;-0.033)
r_{13}	NFGPT(L;0.067)	NFGPT(L;0.000)	-
r_{21}	-	NFGPT(L;-0.067)	NFGPT(VL;0.033)
r_{22}	NFGPT(L;0.0000)	NFGPT(L;0.000)	-
r_{23}	-	-	-
r_{24}	NFGPT(VL;0.033)	NFGPT(L;0.000)	-
r_{31}	NFGPT(L;0.000)	NFGPT(L;0.000)	-
r_{32}	NFGPT(EL;0.067)	NFGPT(L;0.000)	NFGPT(L;0.000)
r_{33}	NFGPT(L;0.000)	NFGPT(L;0.000)	NFGPT(L;0.000)

The sum of all these products in relation to the same resource will indicate the priority that should be assigned to this resource, in relation to the other resources that should also be assigned. This constitutes the Linguistic Distributed System Assignment Function (*LDSAF*). Refer to (5).

$$LDSAF(r_{ij}) = \Sigma\ NFGPT_{ijkl}\ with\ r_{ij}\ \text{resource allocation priority} \quad (5)$$

When calculating the *LDSAF* for all resources, a 2-tuple vector will be obtained. Sorting their elements from highest to lowest, you will get the priority order of resource allocation. This should be normalized by ensuring that the 2-tuples obtained are in the range [0, 1]. This can be seen in Table XLIII.

In addition, as indicated above, the largest of the $NFGPT_{ijkl}$ of each resource will indicate the process to which the resource will be assigned.

TABLE XLIII. ASSESSMENTS TO NORMALIZE THE *LDSAF*

Label	Value
Maximum Value	4.5999
Minimum value	1.4003
Range	3.1996

The result of the standardization of the 2-tuples constitutes what will be called the Normalized Linguistic Distributed System Assignment Function (*LDSAF*). Refer to (6).

$$LDSAF(r_{ij}) = \Sigma\ (NFGPT_{ijkl} / (\text{Maximum}\ (NFGPT_{ijkl}) - \text{Minimum}\ (NFGPT_{ijkl})) = r_{ij}\ \text{priority of resource allocation normalized between extreme values} \quad (6)$$

This is shown in Table XLIV.

TABLE XLIV. ASSESSMENTS TO NORMALIZE THE *LDSAF*

Order of resource assignment	Priority of assignment	Process selected
r_{33}	T(EH;0.0000)	p_{23}
r_{12}	T(EH;-0.0625)	p_{24}
r_{31}	T(H;0.0209)	p_{21}
r_{22}	T(H;0.0209)	p_{22}
r_{13}	T(M;-0.0418)	p_{21}
r_{11}	T(M;-0.0626)	p_{12}
r_{21}	T(L;0.0417)	p_{22}
r_{23}	T(L;0.0311)	p_{24}
r_{24}	T(EL;0.0000)	p_{36}
r_{32}	T(EL;0.0000)	p_{23}

The next step is to repeat the procedure but eliminating the requests for assignments already made. It should be noted that the allocated resources will be available once the processes release them and therefore, they can be allocated to other processes. The system is self-regulating by releasing the resources already assigned to the processes in the previous step.

The resource requests from the processes that have not yet been satisfied, that is, the calculations in Table XLIII and Table XLIV are repeated with their respective values, omitting the processes already completed.

The result of the concatenation of all allocation rounds for this example can be seen in Table XLV. The diagram in Fig. 4 shows a graph that allows the flow and relationship between the different rounds of resource allocation to processes to be represented.

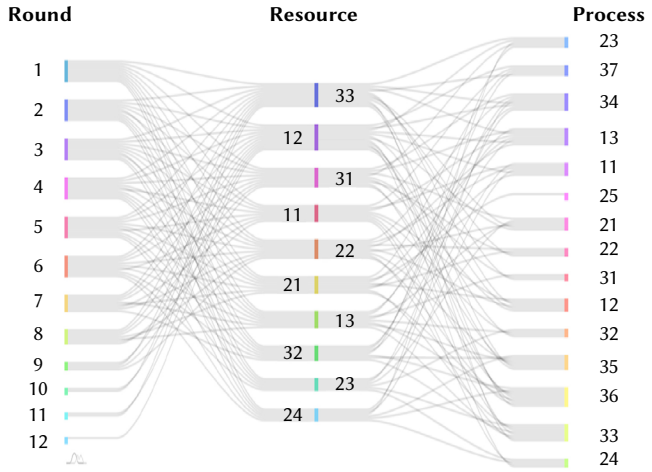

Fig. 4 Flow and relationships between different rounds of resource allocation.

VI. COMPARISON OF RESULTS OBTAINED WITH TRADITIONAL METHODS

Table XLV of this scenario is compared to Table XXIV of scenario E1 and look for each first round allocation of the latter in Table XLV.

In Table XLVI you can see the order of assignments for the first round of the E1 scenario, while Table XLVII represents the order in which the same assignments appear but for the traditional method, in which round they appear and in which position with respect to each round.

TABLE XLVI. VALUES CORRESPONDING TO THE FIRST ROUND OF ITERATION OF SCENARIO E1 (*ONLDSAF*)

Pos.	Resource	2-tuple	Process	Round
1	r_{33}	T(EH;0.0000)	p_{23}	1
2	r_{12}	T(EH;-0.0326)	p_{37}	1
3	r_{31}	T(H;-0.0028)	p_{34}	1
4	r_{11}	T(H;-0.0355)	p_{37}	1
5	r_{22}	T(M;0.0423)	p_{34}	1
6	r_{21}	T(M;0.0337)	p_{37}	1
7	r_{13}	T(M;0.0274)	p_{13}	1
8	r_{32}	T(M;-0.0334)	p_{34}	1
9	r_{23}	T(L;-0.0591)	p_{11}	1
10	r_{24}	T(EL;0.0000)	p_{34}	1

TABLE XLV. CONCATENATION OF ALL ASSIGNMENT ROUNDS (*CNLDSAF*) FOR TRADITIONAL METHODS

Resource	2-tuple	Process	Round
r_{33}	T(EH;0.0000)	p_{23}	1
r_{12}	T(EH;-0.0625)	p_{24}	1
r_{31}	T(H;0.0209)	p_{21}	1
r_{22}	T(H;0.0209)	p_{22}	1
r_{13}	T(M;-0.0418)	p_{21}	1
r_{11}	T(M;-0.0626)	p_{12}	1
r_{21}	T(L;0.0417)	p_{22}	1
r_{23}	T(L;0.0311)	p_{24}	1
r_{24}	T(EL;0.0000)	p_{36}	1
r_{32}	T(EL;0.0000)	p_{23}	1
r_{33}	T(EH;0.0000)	p_{22}	2
r_{12}	T(EH;-0.079)	p_{21}	2
r_{31}	T(H;-0.0613)	p_{23}	2
r_{22}	T(H;-0.0614)	p_{21}	2
r_{11}	T(M;-0.0001)	p_{24}	2
r_{13}	T(M;-0.0791)	p_{13}	2
r_{21}	T(L;0.0351)	p_{11}	2
r_{23}	T(VL;0.0832)	p_{21}	2
r_{24}	T(EL;0.0000)	p_{34}	2
r_{32}	T(EL;0.0000)	p_{36}	2
r_{33}	T(EH;0.0000)	p_{13}	3
r_{12}	T(EH;-0.0333)	p_{23}	3
r_{22}	T(H;0.0335)	p_{34}	3
r_{31}	T(M;0.0669)	p_{13}	3
r_{11}	T(M;0.0666)	p_{32}	3
r_{13}	T(M;-0.0001)	p_{32}	3
r_{21}	T(M;-0.0665)	p_{12}	3
r_{23}	T(VL;0.0832)	p_{11}	3
r_{24}	T(EL;0.0333)	p_{11}	3
r_{32}	T(EL;0.0000)	p_{37}	3
r_{33}	T(EH;0.0000)	p_{31}	4
r_{12}	T(EH;-0.069)	p_{13}	4
r_{22}	T(H;0.0231)	p_{11}	4
r_{31}	T(H;-0.0803)	p_{22}	4
r_{11}	T(M;0.0517)	p_{37}	4
r_{13}	T(M;-0.0172)	p_{35}	4
r_{21}	T(M;-0.0516)	p_{25}	4
r_{23}	T(VL;0.0402)	p_{34}	4
r_{24}	T(VL;-0.0632)	p_{23}	4
r_{32}	T(EL;0.0000)	p_{34}	4
r_{33}	T(EH;0.0000)	p_{34}	5
r_{12}	T(EH;-0.0769)	p_{35}	5
r_{22}	T(H;0.0257)	p_{12}	5
r_{31}	T(M;0.077)	p_{35}	5
r_{11}	T(M;0.0001)	p_{11}	5
r_{13}	T(M;-0.0769)	p_{34}	5
r_{21}	T(M;-0.0769)	p_{33}	5
r_{23}	T(VL;-0.0513)	p_{32}	5
r_{24}	T(VL;-0.0513)	p_{24}	5
r_{32}	T(EL;0.0000)	p_{13}	5
r_{33}	T(EH;0.0000)	p_{21}	6
r_{12}	T(VH;0.0714)	p_{12}	6
r_{22}	T(H;0.0000)	p_{35}	6
r_{31}	T(M;0.0238)	p_{36}	6
r_{11}	T(M;-0.0714)	p_{36}	6
r_{13}	T(L;0.0000)	p_{36}	6
r_{21}	T(L;0.0000)	p_{36}	6
r_{23}	T(EL;0.0476)	p_{33}	6
r_{24}	T(EL;0.0476)	p_{35}	6
r_{32}	T(EL;0.0000)	p_{33}	6
r_{33}	T(EH;0.0000)	p_{35}	7
r_{12}	T(VH;0.0556)	p_{32}	7
r_{22}	T(H;-0.0556)	p_{36}	7
r_{31}	T(M;-0.0556)	p_{31}	7
r_{11}	T(L;0.0000)	p_{13}	7
r_{21}	T(L;-0.0556)	p_{37}	7
r_{13}	T(VL;0.0556)	p_{31}	7
r_{32}	T(EL;0.0556)	p_{35}	7
r_{33}	T(EH;0.0000)	p_{36}	8
r_{12}	T(VH;0.0128)	p_{36}	8
r_{22}	T(M;-0.0385)	p_{33}	8
r_{31}	T(L;-0.0256)	p_{34}	8
r_{11}	T(VL;0.0641)	p_{33}	8
r_{21}	T(VL;-0.0128)	p_{13}	8
r_{33}	T(EH;0.0000)	p_{37}	9
r_{12}	T(VH;-0.0833)	p_{34}	9
r_{22}	T(L;-0.0833)	p_{13}	9
r_{33}	T(EH;0.0000)	p_{33}	10
r_{12}	T(H;0.0000)	p_{37}	10
r_{12}	T(EH;0.0000)	p_{11}	11
r_{13}	T(M;0.0000)	p_{33}	11
r_{31}	T(M;0.0000)	p_{12}	11
r_{33}	T(M;0.0000)	p_{12}	11
r_{12}	T(EH;0.0000)	p_{33}	12

TABLE XLVII. VALUES CORRESPONDING TO THE SAME PROCESS RESOURCE ASSIGNMENTS FROM THE *ONLDSAF* TABLE OF THE FIRST E1 ITERATION FOUND IN THE *CNLDSAF* TABLE OF THE TRADITIONAL METHODS

Pos.	Resource	2-tuple	Process	Round
1	r_{33}	T(EH;0.0000)	p_{23}	1
2	r_{12}	T(H;0.0000)	p_{37}	10
4	r_{31}	T(L;-0.0256)	p_{34}	8
5	r_{11}	T(M;0.0517)	p_{37}	4
3	r_{22}	T(H;0.0335)	p_{34}	3
6	r_{21}	T(L;-0.0556)	p_{37}	7
6	r_{13}	T(M;-0.0791)	p_{13}	2
10	r_{32}	T(EL;0.0000)	p_{34}	4
8	r_{23}	T(VL;0.0832)	p_{11}	3
9	r_{24}	T(EL;0.0000)	p_{34}	2

The first element of Table XLVII, assignment of r_{33} to p_{23}, is the only one that occurs in the same iteration (first), all other assignments in the example of traditional methods occur in different rounds and in different positions. It has been seen that in this comparison, the results of assignments in the traditional methods are not the same as those in the proposed model. This is because traditional methods consider only one type of criterion (process priority), and do not consider the number of processes, %CPU, %Mem, %MV, etc., that is, the load of each node and the overall state of the system.

In this sense, it can be said that besides being the traditional methods, a particular case of the proposed method. This new model allows a more approximate evaluation to the real state of the system, which would allow to obtain better results in the assignments.

The global model, for the example of the traditional methods, does not consider the collection of information on the overall state of the system, nor the predisposition (nodal priority), nor the load of the node, only the process priority is considered.

It should be noted that the results obtained are adjusted to each particular scenario. That is, when the conditions of the scenario change, the results obtained in the application of the Decision Model may be different.

The Fig. 5 shows the different values of the nodal loads. It is observed that the process p_{36} requests the resource r_{13}, whose nodal load has a value of 6.4. By the intensity of the color, you can see that this node is highly loaded.

In the traditional methods, the order of assignment is made only considering the initial priority of the processes. Following this premise, the assignment of resource r_{13} to process p_{36} is made in the first position.

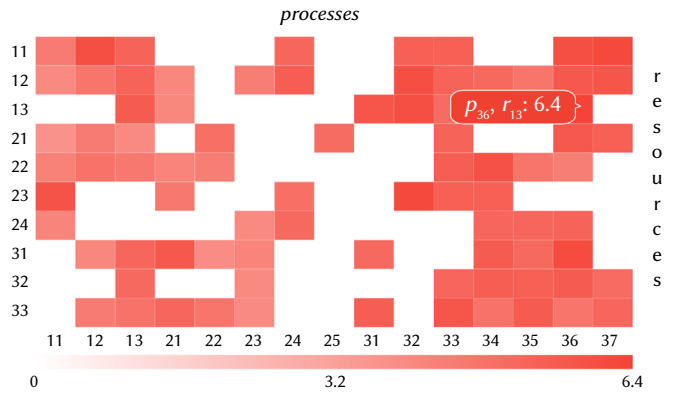

Fig. 5. Heat map showing sample values of nodal loads. The heat map has been created with AMCharts v4 JavaScript library.

As explained above, the proposed method evaluates a set of criteria (including initial priority and nodal load) to determine the order of allocation. Considering that, the same assignment of the previous example, for this method, is in position 56. This is because the node where the assignment is made is heavily loaded. This can be seen in the Table XLVIII.

TABLE XLVIII. COMPARISON OF THE TRADITIONAL METHOD WITH THE PROPOSED METHOD

Nodal Preference	Traditional Method			Proposed Method		
	Process	Resource	Pos.	Process	Resource	Pos.
6.4	p_{36}	r_{13}	1	p_{36}	r_{13}	56
6.2	p_{32}	r_{23}	2	p_{32}	r_{23}	48
6.2	p_{37}	r_{11}	3	p_{37}	r_{11}	35
6.1	p_{36}	r_{31}	4	p_{36}	r_{31}	54
6.0	p_{32}	r_{12}	5	p_{32}	r_{12}	62
6.0	p_{32}	r_{13}	6	p_{32}	r_{13}	26
6.0	p_{36}	r_{11}	7	p_{36}	r_{11}	55
6.0	p_{12}	r_{11}	8	p_{12}	r_{11}	6
5.9	p_{34}	r_{22}	9	p_{34}	r_{22}	23
5.9	p_{11}	r_{23}	10	p_{11}	r_{23}	28

VII. DISCUSSIONS AND COMMENTS

It highlights the dynamism, the magnitude of the nodes, processes and the number of requirements that can be applicable to large systems, through a global solution, or to systems with fewer nodes and requirements. The load of traffic, processes and requirements varies, so that systems that were operating stop operating because they end and others appear, or because nodes with resources and processes that are needed are activated. The nodes can be active, but they are incorporated to the algorithm when some process requests some resource, or some resource is requested by another process of another node. A node can be active but not part of the assignment evaluations.

In each node, an interface is defined between the applications and the operating system, which through a Runtime (software at runtime complementary to the operating system) included in that interface, manages the processes and shared resources, and defines the corresponding scenario, as can see in Fig. 6.

In addition, the Runtimes interact with each other to exchange information and in one of the nodes there is a global coordinating Runtime that evaluates and executes the decision model and the corresponding aggregation operator.

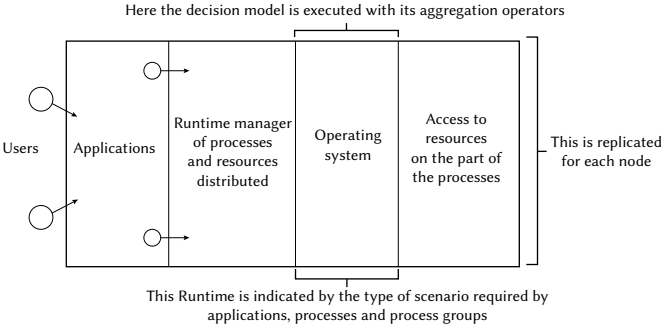

Fig. 6. Runtime global coordinator.

The proposed model manages to establish a consensus that allows groups of processes to access all their resources sequentially and that these cannot be removed until the same group of processes that maintains them, releases them. The order of allocation will be determined by the overall average priority of all the allocations of

each group. The distributed system constantly regulates and updates the local status of each node. The decisions of access to the resources modify these states so it must be readjusted repeatedly, guaranteeing the mutual exclusion, and reordering new priorities. The method should be repeated whenever there are groups of processes that require shared resources.

An important feature to note about the Neat OWA operators used is that the values to be added do not need to be ordered for processing. This implies that the formulation of a neat operator can be defined using the arguments directly instead of the sorted elements.

In the proposed aggregation operator, the weights are calculated based on the context values from which the values to be aggregated arise.

The characteristics of the Decision Model allow us to evaluate the possible alternatives and consequences and thus be able to clearly define the objectives. The best optimization has been achieved by selecting the best possible alternative in each particular case. As the main objective of the proposed model, the environment of distributed execution of processes was considered, the access to shared resources was established according to different consensus requirements. This allows the generation of the sequence of resource allocation to the processes that request them by using the most appropriate aggregation method for each possible scenario, respecting the mutual exclusion in the access to such resources. It has been explained that the decision model uses a Runtime that manages the shared processes and resources and defines the corresponding scenario. The traditional models have been compared by means of an example of application with respect to the proposed model and the considerations of the aggregation operators developed have been commented on.

The aggregation method used, and the data structure mentioned in this work are not fully covered by traditional methods, for example, do not contemplate the predisposition (nodal priority), node load, the nodal state (nodes, processes, groups, resources.) or the overall system state, for the calculation of priorities in resource allocations to processes.

VIII. Conclusions

The proposed model makes it possible for the distributed system to self-regulate repeatedly according to the local state of the n nodes, resulting in an update of their local states, as a consequence of the evolution of their respective processes and the decisions of access to resources: the distributed system in whose groups of processes access to critical resources is executed, produces access decisions to resources that modify the state of the system and readjusts it repetitively, also guaranteeing the mutual exclusion in access to the shared resources, indicating the priority of granting access to each resource and the process to which it is assigned. This process is repeated if there are processes that request access to shared resources.

In this work, fuzzy logic has been used as a tool to innovatively solve the management of resources and processes in distributed systems. The use of the 2-tuple linguistic model allows to improve accuracy and facilitate word processing by treating the linguistic domain as continuous but maintaining the linguistic base (syntax and semantics), through symbolic translation.

What makes the proposed method innovative is that it allows system self-regulation, respects the initial priority of the processes, maintains the status of the nodes updated through the self-regulation, the mutual exclusion is guaranteed, the symbolic translation is incorporated for nodes that use different types of tags, the collaborative nodal priority is established that collaborates in the self-regulation of the system and also includes traditional methods as particular cases of the proposed method.

A prototype simulator has been developed to evaluate the performance of the new decision models and aggregation operators proposed against the main traditional models.

In this research, a software has been developed that simulates the execution of a central runtime of a node located in a distributed system, it is a web application that has been developed with the php language.

When evaluating the results obtained with the simulator, it was possible to verify that the solution produced contemplates an adequate workload balancing, according to the theoretical support used for the development of the simulator. It was also possible to demonstrate that the proposed theoretical solution is more adequate than traditional algorithms that allocate resources to processes only according to the priority of the processes. The values for the figures 4 and 5 has been obtained from the simulator.

For future work, it is planned to develop variants of the proposed method considering other aggregation operators (especially the OWA family) and the possibility of being used by a resource manager shared (instead of centralized as in the proposed method).

It is also planned to continue the development of a simulator with other scenarios.

Acknowledgment

This work has been supported by the Project: "Decision models for resource and process management in distributed systems considering process migration, data imputation and fuzzy logic in new aggregation operators.", code 20F005 of Northeastern National University (Argentine), and the Project: "Development of a simulator for the evaluation of classical and new algorithms for the management of shared resources in distributed systems contemplating mutual exclusion.", code PI 126/20 of the National University of the Southern Chaco (Argentine).

References

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

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

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

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

[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, USA, 2001, pp. 1256-1268.

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

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

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

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

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

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

[12] H. Attiya, J. Welch, *Distributed Computing Fundamentals, Simulations, and Advanced Topics*, 2nd ed., New York, USA, John Wiley, 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". *Proceeding of the Third International Workshop on Peer-to-Peer Systems*, vol. 3279 of Lecture Notes in Computer Sciences, (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), pp 1175– 1185, 1990.

[17] F. Agostini, D. L. La Red Martínez, J. C. Acosta. "Modeling of the consensus in the allocation of resources in distributed systems". *International Journal of Advanced Computer Science and Applications* (IJACSA). The Science and Information (SAI) Organization, England, U.K. Vol. 9, no. 12., 2018.

[18] L. A. Zadeh, "The Concept of a Linguistic Variable and its Application to Approximate Reasoning-I", *Information Sciences*, Volume 8, Issue 3, pp 199-249, 1975, DOI: 10.1016/0020-0255(75)90036-5.

[19] C. González García, E. R. Núñez-Valdez, V. García-Díaz, B. C. Pelayo G Bustelo, J. M. Cueva Lovelle, "A Review of Artificial Intelligence in the Internet of Things", *International Journal of Interactive Multimedia and Artificial Intelligence - IJIMAI Journal*, 2019, DOI: 10.9781/ijimai.2018.03.004

[20] L. A. Zadeh, "Fuzzy Logic = Computing with Words", *IEEE Transactions On Fuzzy Systems*, VOL. 4, NO. 2, 103-111, 1996.

[21] Ta-Chun Wen, Kuei-Hu Chang, Hsin-Hung Lai, "Integrating the 2-tuple linguistic representation and soft set to solve supplier selection problems with incomplete information", *Engineering Applications of Artificial Intelligence*, Vol. 87, January 2020.

[22] Jerry M. Mendel, "Computing with Words: Zadeh, Turing, Popper and Occam", *IEEE Computational Intelligence Magazine*, pp 10-17, November 2007.

[23] F. Herrera, L. Martinez. "An Approach For Combining Linguistic And Numerical Information Based On The 2-Tuple Fuzzy Linguistic Representation Model In Decision-Making". *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, Vol. 8, No. 5, pp 539-562, 2000.

[24] F. Herrera, L. Martínez, "A 2-Tuple Fuzzy Linguistic Representation Model for Computing with Words". *IEEE Transactions On Fuzzy Systems*, VOL. 8, NO. 6, pp. 746-752, December 2000

[25] F. Herrera, E. Herrera-Viedma, S. Alonso, F. Chiclana: "Computing with Words in Decision Making: Foundations, Trends and Prospects", *Fuzzy Optimization and Decision Making*, 8, 337-364, 2009 (ISSN: 1568-4539). doi:10.1007/s10700-009-9065-2

[26] S. Zapata, D. Fuentealba, G. Valenzuela. "Aplicación del modelo de representación de información lingüística 2-tuplas con información multigranular". *Revista Trilogía: Ciencia, Tecnología y Sociedad*. Vol. 27, N° 37, pp 110-127, July 2015. Facultad de Ingeniería UTEM.

[27] Jiménez, G.E. and Zulueta, Y., "A 2-tuple linguistic multi-period decision making approach for dynamic green supplier selection". *DYNA*, 84(202), pp. 199-206, September 2017.

[28] M. Ying. "A Formal Model of Computing With Words". *IEEE Transactions On Fuzzy Systems*, VOL. 10, N° 5, October 2002.

[29] J. Liu, L. Yi and Z. Pei, "A new linguistic term transformation method in linguistic decision making". *Journal of Intelligent & Fuzzy Systems* 35, 2403–2412, IOS Press, 2018, DOI:10.3233/JIFS-17987.

[30] David L. la Red Martínez, "Aggregation Operator for Assignment of Resources in Distributed Systems", *(IJACSA) International Journal of Advanced Computer Science and Applications*, Vol. 8, No. 10, 2017.

[31] R. Yager. "On Ordered Weighted Averaging Aggregation Operators in Multi-Criteria Decision Making". *IEEE Transactions On Systems, Man and Cybernetics* 18: 183-190, 1988.

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

[33] F. Herrera and L. Martínez, "A model based on linguistic 2-tuples for dealing with multigranular hierarchical linguistic contexts in multi-expert decision-making," *IEEE Transactions on Systems, Man and Cybernetics. Part B (Cybernetics)*, vol. 31, no. 2, 2001, doi: 10.1109/3477.915345.

Jorge Tomás Fornerón Martínez

Jorge Tomás Fornerón Martínez received the diploma degree in Information Systems Analysis from the Autonomous University of Asunción (Paraguay) in 2015 and the MSc degree in Informatic and Computation from the National University of Pilar (Paraguay) in 2020. He is professor of General Systems Theory and is Thesis Director in National University of Pilar (Paraguay), since 2015. Currently, he holds the position of Dean at Faculty of Applied Sciences in the same University, and its lines of research are focused on distributed operating systems.

Federico Agostini

Federico Agostini received the diploma degree in Information Systems from the National University of the Northeast (Argentine) in 2013, and the MSc degree in Telecommunications Systems and Networks from the National University of the Northeast (Argentine), in 2019. He is professor of Data Communications and Operating Systems at the same University, since 2013. Currently, he is working at the Northeast Botanical Institute (National University of the Northeast - National Council for Scientific and Technical Research), and its lines of research are focused on bioinformatics.

David L. la Red Martínez

David L. la Red Martínez received the diploma degree in Information Systems from the National University of the Northeast (Argentine) in 1979, the MSc degree in Informatic and Computation from the National University of the Northeast (Argentine), in 2001, the Specialist degree in University Teaching from the National University of the Northeast (Argentine), in 2003 and the Doctoral degree in Computer Systems Engineering from the University of Malaga (Spain), in 2011. He made a postdoctoral research stay in cyber-physical security systems at Florida Atlantic University (USA), in 2019. He is professor of Databases, Data Communications and Operating Systems at the National University of the Northeast (Argentine), since 1983. Currently, its lines of research are focused on distributed systems and data mining.