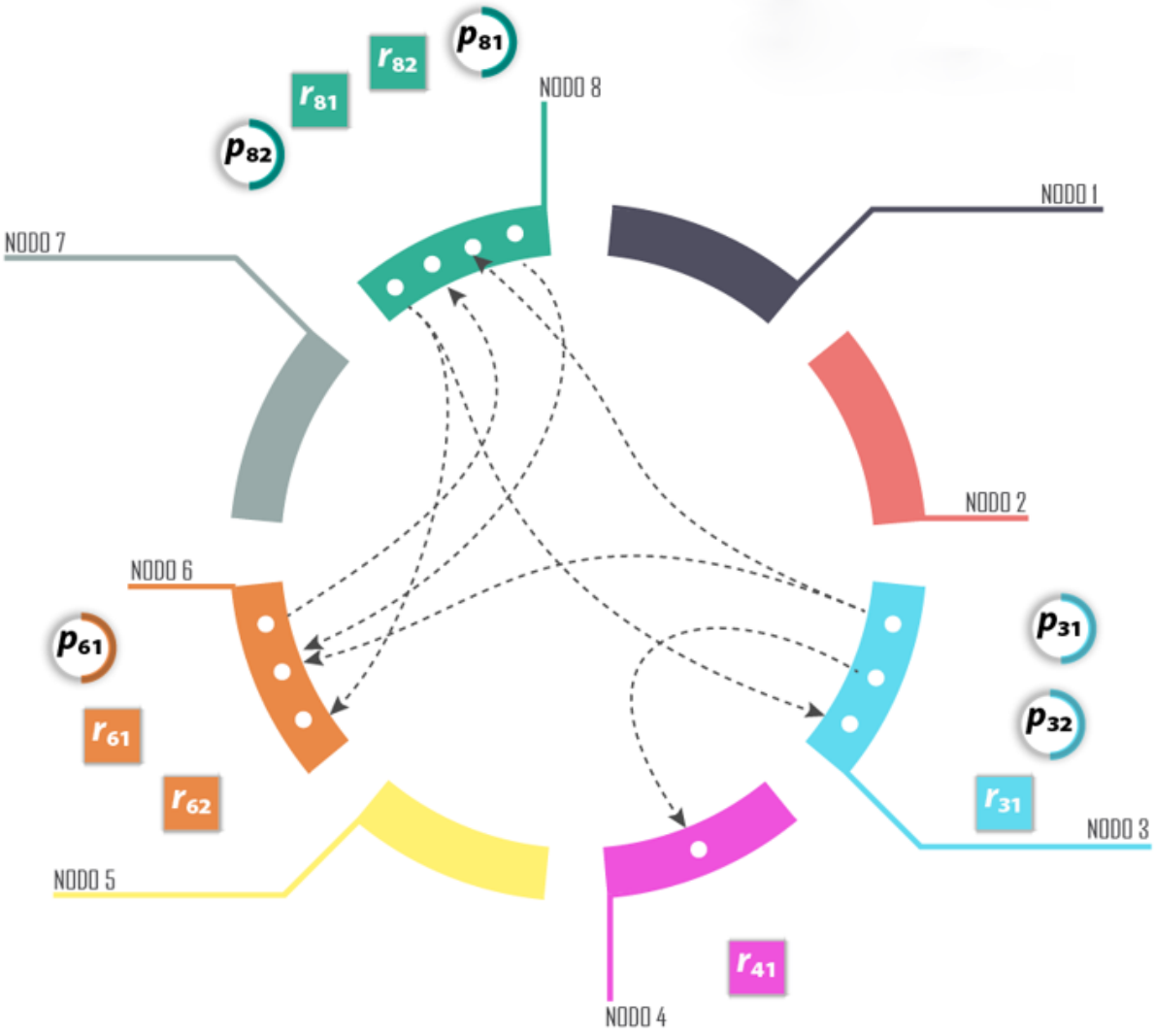
REVISTA ReCientE **NÚMERO ESPECIAL- SETIEMBRE DE 2023**
ISSN: 2524-1451

UNIVERSIDAD NACIONAL DE PILAR – FACULTAD DE CIENCIAS APLICADAS
GRUPO DE INVESTIGACION DE SISTEMAS DISTRIBUIDOS

Tesis de Maestría en Informática y Computación



UNIVERSIDAD NACIONAL DE PILAR
FACULTAD DE CIENCIAS APLICADAS

Presentación

Esta publicación forma parte de una serie de números especiales de la revista ReCientE, dedicadas a la publicación de tesis de postgrados que han sido presentadas y aprobadas en la Facultad de Ciencias Aplicadas de la Universidad Nacional de Pilar, constituyéndose en un canal que permite difundir nuevos conocimientos como un aporte a la comunidad científica internacional.

La tesis de postgrado es la culminación del proceso que todo alumno debe presentar y aprobar como pre-requisito para la obtención del título de postgrado que otorga el programa del cursado desarrollado. En general, es un documento extenso y bien documentado que presenta los hallazgos de una investigación original en un tema específico. Es la culminación de un trabajo de investigación que dura por lo general varios años y representa una inversión significativa de tiempo y esfuerzo por parte del estudiante. Con la publicación de las Tesis se pretende difundir los conocimientos más allá de nuestra unidad académica para apoyar decisiones informadas en el campo del trabajo realizado.

Iniciamos la serie de publicaciones con las tesis aprobadas en el contexto del programa de Maestría en Informática y Computación, presentadas por integrantes que conforman el Grupo de Investigación en Sistemas Distribuidos de la Facultad de Ciencias Aplicadas.

En el presente número especial, que es el primero de la serie, se publica la tesis del Magíster Jorge Tomás Fornerón Martínez, referido al tema “Gestión de Recursos y Procesos con un Modelo de Decisión basado en Lógica Difusa”.

Dra. Nélida Soria Rey

Editora

Editorial

En el mundo de la informática y la tecnología, la gestión eficiente de recursos y procesos en sistemas distribuidos es fundamental para garantizar un rendimiento óptimo. Los sistemas distribuidos, donde múltiples procesos deben tomar decisiones relacionadas con el acceso a recursos compartidos, presentan desafíos únicos que requieren soluciones innovadoras. El Magíster Jorge Tomás Fornerón Martínez, en su tesis de maestría titulada "Gestión de Recursos y Procesos con un Modelo de Decisión basado en Lógica Difusa," aborda estos desafíos de manera sobresaliente.

Fornerón Martínez inicia su tesis señalando la importancia crítica de la toma de decisiones en sistemas distribuidos en lo que respecta al acceso a recursos compartidos. Para garantizar la exclusión mutua en el acceso a estos recursos, el autor propone un innovador modelo de decisión que se adapta a una amplia variedad de requisitos y entornos de ejecución.

Uno de los aspectos más destacados de esta investigación es la consideración de entornos de ejecución distribuida, donde los procesos pueden estar agrupados o ser independientes. El acceso a los recursos compartidos se gestiona de acuerdo con diversas exigencias de consenso, permitiendo asignaciones de recursos a procesos por rondas o de manera consecutiva. Lo que distingue a este modelo es su capacidad para generar secuencias de asignación de recursos a procesos de manera dinámica, utilizando métodos de agregación apropiados para cada escenario particular, empleando etiquetas lingüísticas y 2-tuplas.

Un elemento esencial de la propuesta de Fornerón Martínez es la capacidad del sistema para autoregularse a través de ciclos y subciclos de retroalimentación. Esta característica permite lograr un equilibrio dinámico en la carga de trabajo y en el sistema en su conjunto. La utilización de etiquetas lingüísticas para clasificar los datos del estado del sistema, nodos, procesos y recursos agrega un nivel de inteligencia y adaptabilidad adicionales a la gestión de recursos.

La tesis también analiza y compara modelos clásicos de acceso a recursos compartidos, como el algoritmo centralizado, el algoritmo distribuido de Lamport, el algoritmo de Ricart y Agrawala, y el algoritmo de anillo de fichas. Esta comparación demuestra la eficacia del modelo propuesto por Fornerón Martínez en términos de adaptabilidad y rendimiento.

En resumen, la tesis de maestría de Jorge Tomás Fornerón Martínez representa un importante avance en la gestión de recursos y procesos en sistemas distribuidos. Su modelo de decisión basado en lógica difusa ofrece una solución versátil y eficaz para un problema cada vez más relevante en el mundo de la tecnología de la información. Es de esperar que este trabajo inspire a futuros investigadores y contribuya al avance continuo de la informática distribuida en Paraguay y más allá.

Corrientes, 16 de septiembre de 2023.

Dr. David Luis la Red Martínez

UNIVERSIDAD NACIONAL DE PILAR

FACULTAD DE CIENCIAS APLICADAS

Gestión de Recursos y Procesos con un Modelo de Decisión basado en Lógica Difusa

Jorge Tomás Fornerón Martínez

Orientador: **Prof. Mgter. Julio César Acosta**

Coorientador: **Prof. Mgter. Federico Agostini**

Tesis presentada a la Facultad de Ciencias Aplicadas de la Universidad Nacional de Pilar, para la obtención del Grado de Magíster en Informática y Computación, correspondiente al Programa de Maestría en Informática y Computación.

PILAR – PARAGUAY

SETIEMBRE 2020

Gestión de Recursos y Procesos con un Modelo de Decisión basado en Lógica Difusa

Jorge Tomás Fornerón Martínez

Orientador: **Prof. Mgter. Julio César Acosta**

Coorientador: **Prof. Mgter. Federico Agostini**

Tesis presentada a la Facultad de Ciencias Aplicadas de la Universidad Nacional de Pilar, para la obtención del Grado de Magíster en Informática y Computación, correspondiente al Programa de Maestría en Informática y Computación.

PILAR – PARAGUAY

SETIEMBRE 2020

Fornerón Martínez, Jorge Tomás. (2020). **Gestión de Recursos y Procesos con un Modelo de Decisión basado en Lógica Difusa.** Jorge Tomás Fornerón Martínez. 206 p.: il.

Orientador: Prof. Mgter. Julio César Acosta
Coorientador: Prof. Mgter. Federico Agostini

Tesis académica de Magíster en Informática y Computación - Facultad de Ciencias Aplicadas - Universidad Nacional de Pilar

GESTIÓN DE RECURSOS Y PROCESOS CON UN MODELO DE DECISIÓN BASADO EN LÓGICA DIFUSA

JORGE TOMÁS FORNERÓN MARTÍNEZ

Aprobado en fecha: 08/09/2020 (Res. N° 79/2020 CD-FCA).

Tribunal Examinador: Dr. Sergio Gálvez Rojas

Dr. Javier Fornari

Dr. Marcelo Marinelli

_____

Prof. Mgter. JULIO CÉSAR ACOSTA

Orientador

_____

Prof. Mgter. FEDERICO AGOSTINI

Coorientador

Sello

_____

Prof. Ing. ROGELIO ENCINA ROMÁN

Vicedecano

Dedicatoria

Dedico esta Tesis a mi familia, por ser una de las principales fuentes de motivación para la conclusión de esta tarea académica:

A mi esposa Liza, gracias por la comprensión, paciencia y el apoyo que siempre está dispuesta a dar.

A Viviana, Alejandra, Natalia, Pablo y Renzo, motores de mi vida, con la intención también de motivarlos contínuamente con el ejemplo, alentándolos a aceptar siempre los desafíos y perseguir sin pausa sus sueños.

Agradecimiento

En primer lugar, expreso un profundo agradecimiento al Prof. Mgter. Julio César Acosta (UNNE), Orientador de Tesis, por su esfuerzo e indicaciones oportunas desde el primer momento.

Agradecimiento muy especial al Prof. Mgter. Federico Agostini (UNNE), Coorientador de Tesis, quien, con su esfuerzo, conocimientos y dedicación, con un gran sentido de responsabilidad y rigor científico, ha apoyado decididamente el desarrollo de este trabajo.

Al Prof. Dr. David Luís La Red Martínez (UNNE), director del programa Maestría en Informática y Computación de la Facultad de Ciencias Aplicadas de la UNP, quien, por sus conocimientos, experiencia, su manera de trabajar, su paciencia, sus consejos e indicaciones y su motivación, fueron incentivando y moldeando mi formación como investigador. Para él, mis respetos, admiración y especial gratitud.

Índice

Listado de tablas

Lista de figuras

GESTIÓN DE RECURSOS Y PROCESOS CON UN MODELO DE DECISIÓN BASADO EN LÓGICA DIFUSA

AUTOR: JORGE TOMÁS FORNERÓN MARTÍNEZ
ORIENTADOR: PROF. MGTER. JULIO CÉSAR ACOSTA
COORIENTADOR: PROF. MGTER. FEDERICO AGOSTINI

Resumen

El funcionamiento de los sistemas distribuidos obliga a que grupos de procesos deban tomar decisiones en base a acuerdos en relación con el acceso a los recursos. Esas decisiones pueden estar relacionadas al hecho de compartir o no recursos y con el nivel de consenso o acuerdo necesario para lograr el acceso de grupos de procesos que así lo requieran. La utilización de los recursos compartidos se debe realizar en la modalidad de exclusión mutua. Esta investigación propone un nuevo modelo de decisión con la funcionalidad de adaptarse a distintos requerimientos, se considera el entorno de ejecución distribuida de procesos, que podrían estar agrupados o ser independientes, el acceso a recursos compartidos se establece de acuerdo a distintas exigencias de consenso (las asignaciones se pueden satisfacer por rondas de asignación o de manera consecutiva), para generar la secuencia de asignación de los recursos a los procesos que los solicitan mediante el uso del método de agregación más adecuado a cada escenario posible, respetando la exclusión mutua en el acceso a dichos recursos. Este método innovador, permite que el sistema se autoregule, proporcionando información de manera iterativa (mediante ciclos y subciclos de retroalimentación) para lograr el balanceo dinámico de la carga de trabajo, y el equilibrio del sistema, utilizando etiquetas lingüísticas que permitan clasificar los datos del estado del sistema, nodos, procesos y recursos. Se consideraron al algoritmo centralizado, al algoritmo distribuido de Lamport, Ricart y Agrawala, al algoritmo de anillo de fichas como modelos clásicos para acceder a recursos compartidos. La investigación se desarrolló en Pilar, República del Paraguay, en la Facultad de Ciencias Aplicadas (FCA) de la Universidad Nacional de Pilar (UNP).

Palabras clave: Modelos de Decisión, Operadores de Agregación, Control de Concurrencia, Consenso, Etiquetas Lingüísticas, Comunicación entre Grupos de Procesos, Exclusión Mutua, Sistemas Operativos Distribuidos, Lógica Difusa, Computación con Palabras, Sistemas Autorregulados, Balanceo de Carga.

GESTIÓN DE RECURSOS Y PROCESOS CON UN MODELO DE DECISIÓN BASADO EN LÓGICA DIFUSA

AUTOR: JORGE TOMÁS FORNERÓN MARTÍNEZ
ORIENTADOR: PROF. MGTER. JULIO CÉSAR ACOSTA
COORIENTADOR: PROF. MGTER. FEDERICO AGOSTINI

Abstract

The operation of distributed systems means that groups of processes must make decisions based on agreements regarding access to resources. These decisions may relate to whether or not resources are to be shared and the level of consensus or agreement needed to achieve access for groups of processes that require it. The use of shared resources should be done on a mutually exclusive mode. This research proposes a new decision model with the functionality to adapt to different requirements, considering the environment of distributed execution of processes, which could be grouped or independent, the access to shared resources is established according to different consensus requirements (the assignments can be satisfied by rounds of assignment or in a consecutive way), to generate the sequence of assignment of the resources to the processes that request them by means of the use of the most suitable method of aggregation to each possible scenario, respecting the mutual exclusion in the access to these resources. This innovative method allows the system to self-regulate, providing information in an iterative way (through feedback cycles and sub-cycles) to achieve dynamic workload balancing, and system balancing, using linguistic labels that allow the classification of the data of the system state, nodes, processes and resources. The centralized algorithm, the distributed algorithm of Lamport, Ricart and Agrawala, and the token ring algorithm were considered as classical models to access shared resources. The research was conducted in Pilar, Republic of Paraguay, at the Faculty of Applied Sciences (FCA) of the National University of Pilar (UNP).

Keywords: Decision Models, Aggregation Operators, Concurrency Control, Consensus, Linguistic Labels, Communication between Process Groups, Mutual Exclusion, Distributed Operating Systems, Fuzzy Logic, Computing with Words, Self-Regulated Systems, Load Balancing.

Capítulo I - Introducción

1.1. Introducción

Los sistemas distribuidos, compuestos de múltiples nodos y múltiples procesos, que de manera cooperativa realizan una determinada función, requieren la utilización de modelos de decisión que manera que permitan la utilización de los recursos compartidos a los grupos de procesos que los requieran, accedidos mediante la modalidad de exclusión mutua.

Los nuevos modelos de decisión para la asignación de recursos compartidos podrían ejecutarse en el contexto de un gestor de recursos compartidos para el sistema distribuido, que recibiría los requisitos de recursos compartidos de los procesos que se ejecutan en los diferentes nodos distribuidos, así como el estado de carga computacional de los nodos.

La formulación matemática de diferentes conceptos sociales, como la teoría de la elección social y la justicia, Gaertner (2009), como base para una equitativa distribución de recursos entre los miembros de sistemas distribuidos, como se observa en Pitt et al. (2015), han tomado fuerza a la hora de solucionar problemas de ingeniería teniendo como punto de partida el éxito de las comunidades.

Algunos métodos utilizados para comprender los procesos biológicos se han aplicado a problemas de la ingeniería, de acuerdo con Gómez-Cruz y Maldonado (2011). La aplicación de estos paradigmas a las redes neuronales, lógica difusa, o en inteligencia artificial, es un claro ejemplo del éxito de los mismos.

En el mecanismo de elección utilizado en redes de telecomunicaciones, es necesario tener en cuenta la perspectiva desde dónde se está tomando la decisión, desde el punto de vista del nodo y desde el punto de vista de la red, como una articulación de las opiniones de los miembros, de manera similar a lo que ocurre en las relaciones humanas, en donde las acciones públicas están determinadas

por los miembros de la sociedad; estas opiniones son agregadas constituyendo una preferencia social que refleja la opinión general de la sociedad (Gaertner, 2009), en nuestro caso, de un conjunto de nodos que alojan recursos y procesos compartidos.

Para ello, se debe elegir uno o más criterios para poder resaltar esa alternativa de las demás. Entre mayor número de criterios haya, el problema de elección se vuelve un poco más complejo y se pueden resolver con métodos denominados de optimización multiobjetivo. Si lo posible viene definido por un conjunto discreto de alternativas existen métodos multicriterio discretos para resolver el problema, como el método multicriterio discreto scoring mencionado en Maher y Maysam (2016) y en Bouyssou et al. (2000).

En los sistemas distribuidos, la asignación de recursos a los procesos debe ser realizada considerando las prioridades de los procesos y el estado de la carga de trabajo de los nodos computacionales en los cuales se ejecutan los procesos.

Además, se han propuesto soluciones que podríamos llamar clásicas para diversos tipos de sistemas distribuidos en Andrews (2000), Guerraoui y Rodríguez (2006), Lynch (1996), Tel (2000) y en Attiya y Welch (2004). También en Saxena y Rai (2003) y en Velázquez (1993) se presentan trabajos centrados en asegurar la exclusión mutua. Lin et al. (2004) presentan una interesante solución distribuida basada en permisos y en Sha et al. (1990) una solución basada en prioridades de proceso. Una interesante propuesta se presentó en Ríos y La Red Martínez (2019).

Como se mencionó en Wen et al. (2020), sobre la base del concepto de traducción simbólica, Herrera y Martínez propusieron el modelo de representación lingüística 2-tupla, que expresa la información de evaluación lingüística usando la 2-tupla lingüística (s_i, α), donde el elemento semántico s_i es una etiqueta lingüística de una variable lingüística predefinida, s, y α es un valor numérico que representa la traducción simbólica. En Zhang et al. (2015) introdujeron los conjuntos

imprecisos difusos dudosos y valorado a intervalos, combinando el modelo de conjuntos difusos dudosos y el de conjuntos imprecisos valorado a intervalos. En Tao et al. (2015) presentaron el método de conjunto impreciso 2-tupla, que incorpora el conjunto de términos lingüísticos 2-tupla y el conjunto impreciso para resolver complejos problemas de toma de decisiones en grupo.

Hoy en día, los métodos del conjunto incompleto se aplican ampliamente para resolver problemas de adopción de decisiones en la vida real (por ejemplo, Ali y Shabir; Chang; Deli y Cagman; Tang; Chang; Chang y otros), citados en Wen et al. (2020).

Diferentes palabras pueden tener un significado diferente para diferentes personas. Las respuestas sobre cómo se puede validar un motor de Computación con Palabras (CWW-*Computing With Word*s), qué modelos de conjuntos difusos se deben utilizar o qué opciones se deben tomar para mantener el diseño del motor de CWW lo más simple posible, se analizan y fundan en Zadeh (1975 y 1996) y en Mendel (2007). Algunos conceptos relacionados se pueden encontrar en Herrera et al. (2009).

Un modelo similar se ha presentado en Herrera y Martínez (2000), Zapata et al. (2015), Jiménez y Zulueta (2017), Ying (2002), Liu et al. (2018) y Martínez et al. (2006). Estos trabajos muestran diferentes ventajas de este formalismo para representar la información lingüística sobre los modelos clásicos.

Se trata de un dominio lingüístico continuo, en el que el modelo de cálculo lingüístico se basa en tuplas lingüísticas y realiza los procesos de cálculo de palabras con facilidad y sin pérdida de información, por lo que los resultados de los procesos de cálculo de palabras se expresan siempre en el dominio lingüístico inicial.

El uso de variables borrosas utilizando etiquetas lingüísticas y 2-tuplas permite evitar la pérdida de precisión en el cálculo con palabras.

Debido a estas ventajas, este modelo de representación lingüística se utilizará para lograr el desarrollo de un procedimiento de fusión de la información lingüística y numérica.

El modelo de representación lingüística de 2-tuplas representa la información lingüística mediante una 2-tupla, (s, α). En este trabajo se utilizará para representar la carga de los nodos, las preferencias nodales y las prioridades finales.

La traducción simbólica, según Herrera et al. (2000), de un término lingüístico $s_i \in S = \{s_0,...,s_g\}$, consiste en un valor numérico $\alpha_i \in [-.5, .5)$ que soporta la "diferencia de información" entre una cuenta de información β evaluada en $[0, g]$ obtenida después de una operación de agregación simbólica (actuando sobre el índice de orden de las etiquetas) y el valor más cercano en $[0, ..., g]$ que indica el índice del término lingüístico más cercano en S (s_i).

En el presente documento se presentará un nuevo operador de agregación específico para el problema mencionado. Este operador entra dentro de la categoría de operadores de *OWA*, más específicamente de *Neat OWA*. Se presentará un método innovador para la gestión de recursos compartidos en sistemas distribuidos.

En los últimos años, el avance en las tecnologías informáticas y las telecomunicaciones han permitido una gran expansión de los sistemas de información. Las telecomunicaciones permiten la conectividad de un gran número de usuarios ubicados en cualquier parte del mundo por medio de la transmisión de voz, datos o video a través de una gran variedad de dispositivos. Diferentes redes de comunicación pueden ser accedidas a través de Internet, lo que ha permitido la implementación de instalaciones de cómputo donde pueden ser desplegadas aplicaciones para realizar procesamiento distribuido de tareas. Estas nuevas facilidades ofrecen a los usuarios una gran flexibilidad para estructurar sus propios sistemas de información de una manera eficiente, así como la oportunidad de interactuar con otros sistemas de información de una manera distribuida.

Como consecuencia, esto ha generado un aumento de la utilización y una gran dependencia de los sistemas distribuidos para poder transmitir o procesar información.

Este aumento de la utilización de sistemas informáticos distribuidos, en los cuales existen múltiples procesos que cooperan para el logro de una determinada función, hace necesario disponer de modelos de decisión que permitan a los procesos intervinientes en los distintos grupos de procesos, tomar decisiones en las que son necesarios diferentes niveles de acuerdo, especialmente cuando se trata del acceso a recursos computacionales compartidos y el sistema debe autorregular la forma en que se compartirán dichos recursos.

Debe considerarse de una manera fundamental el caso del acceso a las denominadas regiones críticas de memoria por parte de distintos procesos, que pueden estar operando en equipos distribuidos, donde el acceso a esas regiones críticas debe hacerse en la modalidad de acceso exclusivo y contando con el consentimiento de los demás procesos del grupo.

Una variedad de ejemplos de lo mencionado precedentemente se encuentran en Tanenbaum (1996 y 2009), donde se describen los principales algoritmos de sincronización en sistemas distribuidos, uno de ellos en Lamport (1978), en Agrawal et al. (1991), donde se presenta una solución eficiente y tolerante a fallas para el problema de la exclusión mutua distribuida, en Ricart y Agrawala (1981), Pissinou et al. (1996), Raymond (1989), en Lodha y Kshemkalyani (2000) y en Cao y Singhal (2001), donde se presentan unos algoritmos para gestionar la exclusión mutua en redes de computadoras, en La Red Martínez (2004), donde se describen los principales algoritmos de sincronización en sistemas distribuidos, en Stallings (2005), donde se detallan los principales algoritmos para la gestión distribuida de procesos, los estados globales distribuidos y la exclusión mutua distribuida, en Silberschatz et al. (2006), La Red Martínez et al. (2017), La Red Martínez et al. (2018), Agostini et al. (2019a, 2019b y 2019c), donde se presentan modelos de

decisión innovadores para la gestión de recursos en sistemas distribuidos.

En Wiener (1961, 1985), Brix (1970), Parsegian (1973), Varela (1990), Bateson (1991) y Martínez et al. (2007) se explican los principales fundamentos de la cibernética.

Trabajos realacionados con la comunicación en sistemas distribuidos se mencionan en Birman et al. (1991) y Birman (2005). En Macedonia et al. (1995) se describe una arquitectura de red para entornos virtuales de gran escala para soportar la comunicación entre grupos de procesos distribuidos.

En Yager (1988, 1993) se presentan y analizan los operadores *OWA* (*Ordered Weighted Averaging*: Promedio de Pesos Ordenados) aplicados a la toma de decisiones multicriterio. En Yager et al. (1997, 2002) se presentan los operadores *OWA* y sus aplicaciones en la toma de decisiones multi-agente.

Operadores de agregación utilizables para la toma de decisiones en grupos se exponen en Chiclana et al. (2000), Chiclana et al. (2001), Chiclana et al. (2004), Chao et al. (2016) y Dong et al. (2016). Además, en Kaashoek (1992) se estudia en profundidad la comunicación entre grupos de procesos analizándose diferentes protocolos. En Lu et al. (2007) y Martínez et al. (2006 y 2007) se estudia la toma de decisiones en grupo con multi-objetivos. En La Red Martínez (2017) se desarrollan operadores de agregación para la asignación de recursos en sistemas distribuidos.

En cuanto a los sistemas de soporte de decisión se estudian en Fodor et al. (1994), Fullér (1996), Chen (2001), Greco et al. (2002), Marakas (2002), Peláez et al. (2003, 2004, 2007, 2009) y Doña et al. (2011).

Estos temas y otros que guardan relación también han sido considerados en Joseph y Birman (1989), Bagrodia (1989), Alagarsamy (2003), Cao et al. (2004), Birman (2005), Joshi y Holzmann (2007), Razzaque y Hong (2008), Taheri et al. (2011), Neamatollahi et al. (2012),

Lejeune et al. (2012, 2015), Naimi y Thiare (2013), Rodrigues et al. (2013), La Red Martínez y Acosta (2014), Cicirelli et al. (2016), Kamei y Kakugawa (2017), La Red Martínez (2017), Sakamoto y Ogata (2018), Yuan et al. (2018), Agostini et al. (2018), Agostini et al. (2019a, 2019b, 2019c), etc.

1.2. El problema

Planteamiento del problema

En los sistemas de procesamiento distribuido es necesario que los procesos que actúan en grupos deban tomar decisiones basados en acuerdos respecto del acceso a recursos; las decisiones pueden estar relacionadas con la realización de determinada actividad que requiera o no la sincronización de los procesos, es decir, que los procesos del grupo estén activos en los mismos lapsos de tiempo en sus respectivos procesadores, requiriendo el uso de recursos compartidos en la modalidad de exclusión mutua mediante consensos estrictos o no, con la posibilidad de que ciertas variables relacionadas con el estado de los nodos sean expresadas en formato lingüístico.

Teniendo en consideración lo expresado, puede plantearse la siguiente pregunta: ¿cuáles son los modelos de decisión y los operadores de agregación que habrá que generar para la toma de decisiones en la gestión de grupos de procesos, que trasciendan el enfoque tradicional de las ciencias de la computación, teniendo en cuenta la autorregulación?

Se deberá tener en cuenta diferentes situaciones en relación con el hecho de compartir o no recursos y con el nivel de acuerdo o consenso que sea requerido y que podrá ser estricto o no, con posibles requisitos de sincronización.

La evaluación de los modelos desarrollados se hará comparando sus características con los modelos que son habitualmente utilizados.

Se consideran modelos clásicos para acceder a recursos compartidos en la modalidad de

exclusión mutua utilizando regiones críticas al algoritmo centralizado, al algoritmo distribuido de Lamport, Ricart y Agrawala, al algoritmo de anillo de fichas, entre otros.

Enunciado preciso del eje de la situación problemática

En los sistemas computacionales de procesamiento distribuido es frecuentemente necesario que los procesos que actúan en grupos deban tomar decisiones basados en el acuerdo. Dichos procesos podrán operar en un mismo equipo informático o en varios equipos distribuidos interconectados.

Las decisiones para las cuales deben alcanzar algún nivel de acuerdo pueden estar relacionadas con la realización de determinada actividad que no requiera el uso de recursos compartidos en la modalidad de exclusión mutua, o con la realización de determinada actividad que sí requiera el uso de recursos compartidos en la modalidad de exclusión mutua, para lo cual generalmente las exigencias de niveles de acuerdo son mayores que para el caso anterior, pudiendo darse además que los procesos integren grupos que requieran (o no) sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo). En todos estos casos, las variables que determinen la carga computacional de los nodos, etc., podrán expresarse utilizando etiquetas lingüísticas y lógica difusa.

Fuentes de interés

Las principales fuentes de interés son las siguientes:

a) Los métodos considerados tradicionales para la asignación de recursos en sistemas distribuidos respetando la exclusión mutua en el acceso a los recursos compartidos, en base a cuyas limitaciones se propondrán nuevas soluciones son descriptos por Ricart y Agrawala (1981), Cao y Singhal (2001), Pissinou et al. (1996), Raymond (1989), Lodha y Kshemkalyani (2000) y otros.

b) La lógica difusa y los conjuntos de etiquetas lingüísticas para representar el estado de

variables se mencionan en Chen (2001), Martínez et al. (2006), Lu et al. (2007), Doña et al. (2011), La Red Martínez y Acosta (2014), La Red Martínez y Acosta (2015a), Chao et al. (2016), Dong et al. (2016) y otros.

c) Los operadores de agregación, entre ellos los operadores de la familia *OWA*, que podrán utilizarse para obtener valores agregados de variables lingüísticas difusas se explican en Yager (1988 y 1993), Fodor y Roubens (1994), Fullér (1996), Yager y Kacprzyk (1997), Chiclana et al. (2000, 2001 y 2004), Yager y Pasi (2002), Peláez y Doña (2003), La Red Martínez et al. (2011 y 2015b) y otros.

d) Los modelos de decisión que finalmente obtendrán la solución adecuada para cada escenario previsto, utilizando el operador de agregación adecuado se detallan en Chiclana et al. (2001), Peláez et al. (2004 y 2007), Dong et al. (2016) y otros.

Formulación del problema

¿Cuáles son los nuevos modelos de decisión que habrá que desarrollar incorporando lógica difusa y etiquetas lingüísticas para la toma de decisiones en grupos de procesos, que trasciendan el enfoque tradicional de las ciencias de la computación, teniendo en cuenta la autorregulación?

Sistematización del problema

¿Qué modelos de decisión pueden generarse incorporando lógica difusa y etiquetas lingüísticas para la toma de decisiones para la gestión de procesos o grupos de procesos y recursos, que trasciendan el enfoque tradicional de las ciencias de la computación, de manera que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua sin constituir grupos de procesos que requieran sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y sin exigencias estrictas de consenso para lograr el acceso?

¿Qué modelos de decisión pueden generarse incorporando lógica difusa y etiquetas

lingüísticas para la toma de decisiones de gestión de procesos y recursos, que trasciendan el enfoque tradicional de las ciencias de la computación, de manera que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua sin constituir grupos de procesos que requieran sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y con exigencias estrictas de consenso para lograr el acceso?

¿Qué modelos de decisión pueden generarse incorporando lógica difusa y etiquetas lingüísticas para la toma de decisiones en grupos de procesos y recursos, que trasciendan el enfoque tradicional de las ciencias de la computación, de manera que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua constituyendo grupos de procesos que requieren sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y con exigencias estrictas de consenso para lograr el acceso?

¿Cuáles son las diferencias entre los modelos que se desarrollaron con los modelos conocidos y generalmente aceptados de las ciencias de la computación?

1.3. Objetivos

Objetivo General

Generar modelos de decisión basados en lógica difusa y etiquetas lingüísticas para la gestión en grupos de procesos y recursos en sistemas distribuidos, teniendo en cuenta la autorregulación.

Objetivos específicos

• Generar modelos de decisión incorporando lógica difusa y etiquetas lingüísticas para la toma de decisiones de gestión de procesos y recursos, que trasciendan el enfoque tradicional de las ciencias de la computación, de manera que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua sin constituir grupos de procesos que requieran sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y sin exigencias estrictas de

consenso para lograr el acceso.

• Generar modelos de decisión incorporando lógica difusa y etiquetas lingüísticas para la toma de decisiones de gestión de procesos y recursos, que trasciendan el enfoque tradicional de las ciencias de la computación, de manera que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua sin constituir grupos de procesos que requieran sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y con exigencias estrictas de consenso para lograr el acceso.

• Generar modelos de decisión incorporando lógica difusa y etiquetas lingüísticas para la toma de decisiones en grupos de procesos y recursos, que trasciendan el enfoque tradicional de las ciencias de la computación, de manera que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua constituyendo grupos de procesos que requieren sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y con exigencias estrictas de consenso para lograr el acceso.

• Comparar los modelos que se desarrollaron con los modelos conocidos y generalmente aceptados de las ciencias de la computación.

1.4. Justificación

En los sistemas de procesamiento distribuido es necesario que los procesos que actúan en grupos deban tomar decisiones basados en acuerdos respecto del acceso a recursos. Las decisiones pueden estar relacionadas con la realización de determinada actividad que requiera o no la sincronización de los procesos, es decir, que los procesos del grupo estén activos en los mismos lapsos de tiempo en sus respectivos procesadores, requiriendo el uso de recursos compartidos en la modalidad de exclusión mutua mediante consensos estrictos o no.

Así surge el siguiente interrogante: ¿cuáles son los modelos de decisión y los operadores de

agregación que habrá que generar incorporando la perspectiva cognitiva a los modelos clásicos para la toma de decisiones en la gestión de grupos de procesos, que trasciendan el enfoque tradicional de las ciencias de la computación, teniendo en cuenta la autorregulación y permitiendo la utilización de lógica difusa y conjuntos de etiquetas lingüísticas?

Para ello habrá que considerar diferentes situaciones relacionadas con el hecho de compartir o no recursos y con el nivel de acuerdo o consenso requerido, que podrá ser estricto o no, con posibles requisitos de sincronización.

Los modelos considerarán la posibilidad de imputación de datos faltantes y la fuzzyficación de ciertas variables, utilizando operadores *OWA*, buscando generar operadores de agregación específicos utilizando etiquetas lingüísticas.

1.5. Hipótesis y objeto

Hipótesis

Si se utiliza lógica difusa y etiquetas lingüísticas se podrá desarrollar nuevos modelos de decisión aplicables a grupos de procesos, que trasciendan el enfoque tradicional basado en la prioridad de procesos.

Objeto

El objeto de la investigación son los Sistemas Distribuidos, sus recursos y sus procesos, considerando que los procesos y recursos podrán estar alojados en diferentes nodos distribuidos y que los procesos deberán acceder a los recursos compartidos en la modalidad de exclusión mutua.

1.6. Antecedentes

Para la presente investigación se ha realizado una amplia búsqueda de antecedentes referidos a:

a) Comunicación en sistemas distribuidos.

b) Sincronización en sistemas distribuidos.

c) Uso compartido de recursos y exclusión mutua en sistemas distribuidos.

d) Toma de decisiones en grupo y modelos de decisión.

e) Operadores de agregación lingüísticos en el contexto de modelos de decisión.

f) Operaciones con lógica difusa y etiquetas lingüísticas para expresar y evaluar la carga de trabajo de los nodos de procesamiento en un sistema distribuido.

Como resultado de dicha búsqueda se han encontrado varios antecedentes sustantivos, los cuales figuran en la Bibliografía, pero si bien el número de antecedentes es considerable, no se han encontrado antecedentes que traten de manera simultánea estos temas para la toma de decisiones relacionada con la gestión de recursos compartidos en sistemas distribuidos bajo diferentes condiciones, utilizando lógica difusa y etiquetas lingüísticas. Por esta razón, se considera que el trabajo de investigación reúne las condiciones de originalidad necesarias, siendo además relevante académicamente ya que:

a) Se espera producir conocimiento científico específico no disponible actualmente.

b) El desarrollo de modelos de decisión lingüísticos para toma de decisiones en grupos de procesos en el contexto de sistemas complejos y esquemas de autorregulación constituye un aporte significativo a las ciencias de la computación, en especial en relación con la gestión del acceso a recursos compartidos desde procesos distribuidos, mejorando la gestión de los mismos por parte de los sistemas operativos.

c) Se podría utilizar el modelo de decisión y sus operadores de agregación para la asignación de recursos a procesos, contribuyendo así a mejorar el desempeño de los sistemas distribuidos considerando información global del sistema expresada lingüísticamente, que no se contempla en los métodos tradicionales.

1.7. Encuadre o Marco Teórico

En este apartado se mencionan a continuación los principales aportes al estado del conocimiento en las distintas áreas relacionadas que están relacionadas con el presente trabajo de investigación.

Cibernética.

En los trabajos de Wiener (1961), Brix (1970) y Parsegian (1973) se explican los principales fundamentos de la cibernética y en Wiener (1985) se definen los conceptos fundamentales de la cibernética. Se manifiesta que la cibernética constituye todo campo de la teoría de control y de la comunicación, tanto en la máquina como en animales; en Varela (1990) se presentan los principales conceptos de enacción y emergencia y una reseña de la evolución de los conceptos centrales de la cibernética. En el trabajo de Bateson (1991), el mismo manifiesta la importancia del contexto en la comunicación. Toda comunicación exige un contexto, porque si la comunicación no está enmarcada en contexto no hay significado de la misma, no hay valor diferencial que genere información. Se presenta una visión desde el punto de vista sistémico y a la vez interdisciplinario de los procesos comunicativos, en donde concibe los mismos con un carácter circular y evolutivo y donde la retroalimentación es de vital importancia

Comunicación en sistemas distribuidos.

En Joseph et al. (1989) se exponen protocolos de comunicación confiables, en la modalidad de broadcast, desde la perspectiva de las ciencias de la computación; en Birman et al. (1991) se presenta un protocolo de comunicación multicast para grupos de procesos en la modalidad atómica, desde la óptica de las ciencias de la computación; en Kaashoek (1992) se examina detalladamente la comunicación entre grupos de procesos. Se analizan protocolos como el *FLIP: Fast Local Internet Protocol* y el *BP: Broadcast Protocol*, analizandose la comunicación confiable y eficiente entre

grupos, la programación paralela, la programación tolerante a fallos utilizando broadcasting. Se esboza una arquitectura de tres niveles: el nivel inferior para el protocolo *FLIP*, el nivel medio para la comunicación entre grupos de procesos y el nivel superior para las aplicaciones; en Macedonia et al. (1995) se describe una arquitectura de red para entornos virtuales de gran escala para soportar la comunicación entre grupos de procesos distribuidos; en La Red Martínez (2004) se describen los principales algoritmos de comunicación utilizados en los sistemas distribuidos (referidos como algoritmos clásicos de las ciencias de la computación); en Birman (2005) se estudian tecnologías, servicios web, así como aplicaciones de sistemas distribuidos confiables; en Silberschatz et al. (2006) se presentan los principales algoritmos de coordinación distribuida y gestión de la exclusión mutua (algoritmos clásicos de las ciencias de la computación); en Tanenbaum y Van Steen (2008) y en Tanenbaum (1996, 2009) se describen también los principales algoritmos de comunicación en sistemas distribuidos (referidos como algoritmos clásicos de las ciencias de la computación).

Sincronización en sistemas distribuidos.

En Lamport (1978) se realiza el estudio del tiempo, los relojes y el ordenamiento de eventos en sistemas distribuidos, se analiza el ordenamiento parcial, los relojes lógicos, el ordenamiento total de eventos, el comportamiento anormal, los relojes físicos y otros aspectos; en Ricart et al. (1981), en Lodha y Kshemkalyani (2000) y en Cao y Singhal (2001) se presentan unos algoritmos diseñados para gestionar la exclusión mutua en redes de computadoras, acorde a las ciencias de la computación; en Bagrodia (1989) se analiza el diseño y la evaluación del rendimiento de algoritmos distribuidos utilizados en la sincronización de procesos. Se muestra una solución sencilla para el problema de coordinación de comité, que abarca los problemas de sincronización y exclusión que se asocian con la implementación de encuentro de múltiples vías; en Agrawal et al. (1991) se presenta un procedimiento eficiente y tolerante a fallos para resolver el problema de la exclusión mutua

distribuida, desde la óptica de las ciencias de la computación; en Tanenbaum (1996, 2009) están explicados los principales algoritmos de sincronización en sistemas distribuidos (algoritmos clásicos de las ciencias de la computación); en Alagarsamy (2003) se analizan los principales algoritmos de exclusión mutua; en La Red Martínez (2004) se describen los principales algoritmos de sincronización en sistemas distribuidos (algoritmos clásicos de las ciencias de la computación); en Stallings (2005) se detallan los principales algoritmos de las ciencias de la computación para la gestión distribuida de procesos, los estados globales distribuidos y la exclusión mutua distribuida; en Joshi y Holzmann (2007) se analiza la verificabilidad en un sistema de archivos; en La Red Martínez (2017) y en La Red Martínez et al. (2017, 2018) se desarrollan operadores de agregación para asignaciones de recursos en sistemas distribuidos.

Sistemas de soporte a la decisión.

En Fodor et al. (1994) se analiza el modelado de preferencias difuso para el soporte de decisiones multicriterio; en Fullér (1996) se presenta la utilización de operadores de agregación de la familia *OWA* para la toma de decisiones; en Chen (2001) se estudia la aplicación de métodos lingüísticos a la toma de decisiones para tratar el problema de evaluación de calidad de servicio; en Greco et al. (2002) se presentan varias metodologías para resolver problemas, considerando múltiples atributos y criterios; en Marakas (2002) se estudian sistemas de soporte a la decisión; en Peláez et al. (2003, 2004, 2007, 2009) se analizan operadores de agregación (de mayoría) en grupo que buscan la representación de la mayoría; en Doña et al. (2011) se presenta un modelo de decisión en grupo, utilizando operadores de agregación de la familia *OWA*.

Toma de decisiones en grupo.

En Yager (1988, 1993) se presentan y analizan los operadores *OWA* aplicados a la toma de decisiones multicriterio; en Yager et al. (1997, 2002) se exponen los operadores *OWA* y sus

posibilidades de aplicación en la toma de decisiones multi-agente; En Chiclana et al. (2000), Chiclana et al. (2001) y Chiclana et al. (2004) se presentan varios operadores de agregación que pueden utilizarse para la toma de decisiones en grupos; en Peláez et al. (2003, 2004, 2007, 2009) se analizan operadores de agregación de mayoría y las posibles aplicaciones de estos a la toma de decisiones en grupo; en Lu et al. (2007) y Martínez et al. (2006, 2007) se estudia la toma de decisiones en grupo con multi-objetivos, además del tratamiento de información heterogénea en procesos de ingeniería de evaluación y en sistemas de ingeniería; en La Red Martínez et al. (2011) se presenta el operador WKC-OWA para agregar información en problemas de decisión democrática; en La Red Martínezy Acosta (2015b) se presentan las medidas de comportamiento y las principales propiedades matemáticas relacionadas con los operadores de agregación; en Chao et al. (2016) se estudia de qué manera puede obtenerse un vector de prioridades colectivo a partir de diferentes formatos de expresión de las preferencias por parte de los decisores. El modelo posee la capacidad de reducir la complejidad de la toma de decisiones y evitar la pérdida de información cuando se transforman los diferentes formatos en un formato único de expresión de las preferencias; en Dong et al. (2016) se expone un problema complejo y dinámico de toma de decisiones en grupo, con múltiples atributos. Se propone además un método de resolución que utiliza un proceso de consenso para grupos de atributos, de alternativas y de preferencias, donde se presenta luego un modelo de decisión aplicable a problemas del mundo real.

Lógica Difusa.

Zadeh (1988) realiza una exposición condensada de algunas ideas básicas que subyacen a la lógica difusa y describe algunas aplicaciones representativas. Abarca los principios básicos, el significado de representación e inferencia, las reglas básicas de inferencia y las variables lingüísticas; Zadeh (1999) trata la metodología de la Computación con Palabras en la que los objetos de cálculo

son palabras y proposiciones extraídas de un idioma. Explica conceptos como granularidad, variables lingüísticas y otros; En Zadeh (2008), se exponen la Lógica difusa, los Conjuntos Difusos, el Razonamiento aproximado, la Computación con Palabras, la Computación con Percepciones y la Teoría generalizada de la Incertidumbre.

En Singh et al. (2013), se explican los fundamentos, aplicaciones y avances de la lógica difusa y Meier et al. (2017) trata de la la utilización de lógica difusa para la toma de decisiones; el trabajo de Zhang et al. (2017) hace referencia a Conjuntos Difusos, Conjuntos Rugosos, Redes Neuronales, Computación Evolutiva, Razonamiento Probabilístico, Lógica Multivaluada y areas relacionadas.

Kacprzyk et al., (2017) presenta una nueva visión de los "sistemas inteligentes", a través de un montón de modelos para el análisis de datos, la toma de decisiones, la modelización de sistemas y el control, con puntos de vista innovadores desde el enfoque de la computación granular, la soft computing y la lógica difusa.

También Petrosino et al. (2016) y Collan et al. (2016) mencionan trabajos que tratan los Fundamentos de Análisis de Datos, Toma de Decisiones y Modelado de Sistemas, así como Lógica Difusa en la práctica empresarial e industrial y Carvalho de Barros et al. (2017) trata sobre Teoría de Conjuntos Difusos, nociones de Lógica Difusa, Relaciones Difusas, Sistemas Dinámicos Difusos y otros.

1.8. Relevancia

Académica

Esta investigación apunta a una fuerte innovación y que llama constantemente a la reflexión sobre las formas de encarar la solución a problemas de las ciencias de la computación desde una

óptica no convencional, en la cual se propone la utilización de distintas disciplinas en las que ya se han realizado trabajos y publicaciones y que ahora se tiene previsto integrar en la búsqueda de nuevas soluciones a los problemas planteados de gestión de recursos compartidos en grupos de procesos operando en sistemas distribuidos.

Además, se tiene como objetivo incorporar al aula de grado tanto de la Universidad de Pilar como de la Universidad Nacional del Nordeste, especialmente en la asignatura Sistemas Operativos, los principales resultados del proyecto, enriqueciendo de esta manera la labor docente y el aprendizaje de los alumnos.

Social

El propósito de la presente investigación es el de desarrollar modelos de decisión que permitan la toma de decisiones basadas en etiquetas lingüísticas en vez de valores numéricos, aplicables a grupos de procesos computacionales. En tal sentido podría considerarse que la relevancia social está dada, ya que la informática y las comunicaciones se constituyeron en parte de la vida cotidiana de millones de personas alrededor del mundo, considerándose que la informática se ha vuelto ubícua y pervasiva: ubícua en el sentido de estar presente de manera muchas veces inadvertida en gran cantidad de actos de la vida cotidiana y pervasiva al permitir el acceso a los sistemas informáticos casi desde cualquier lugar y en cualquier momento, haciendo uso de redes de comunicaciones en muchos casos inalámbricas.

Es un hecho también que actualmente la mayoría de los sistemas informáticos utilizados funcionan de manera distribuida sobre redes de comunicaciones, debiendo realizar permanentemente operaciones de coordinación y toma de decisiones en grupos de procesos.

Los sistemas descriptos anteriormente son masivamente utilizados por las personas de manera directa o indirecta, por lo que considera que cualquier forma de mejorar el desempeño de

dichos sistemas redundaría en beneficio de la sociedad, en el contexto de la llamada Sociedad de la Información y el Conocimiento, con aplicaciones relacionadas por ejemplo con el e-government (gobierno electrónico), el e-commerce (comercio electrónico), el e-learning (aprendizaje electrónico), la e-democracy (democracia electrónica) y la m-cognocracy (gobierno electrónico móvil en la sociedad del conocimiento).

Relevancia Científica

Desarrollar modelos de decisión para toma de decisiones en grupos de procesos en el contexto de sistemas complejos y esquemas de auto-regulación, constituye un aporte significativo de las ciencias cognitivas a las ciencias de la computación, especialmente en relación con la gestión del acceso a recursos compartidos desde procesos distribuidos, permitiendo mejorar la gestión de dichos accesos por parte de los sistemas operativos.

1.9. Metodología

Dimensión de la Estrategia General

a) Tipo de Estrategia y Fundamentación

Teniendo en cuenta los componentes objeto-problema-objetivos, se utilizó la investigación cuantitativa, con desarrollo de modelos teóricos, medición de variables, formulación de hipótesis y validación de hipótesis.

b) Universo

El universo lo constituyeron recursos y procesos alojados en distintos nodos en sistemas distribuidos, donde se producen requerimientos de accesos a los recursos compartidos en la modalidad de exclusión mutua.

c) Unidad de Análisis

La unidad de análisis estuvo integrada por la relación proceso-recurso requerido, donde los procesos

y recursos están en diferentes nodos del sistema distribuido.

d) Selección de Casos

Se trabajó con los escenarios indicados en los objetivos específicos.

e) Énfasis en el Proceso Lineal

Conforme a las características de la investigación propuesta, se ha optado por la lógica cuantitativa, basada en la búsqueda y verificación de relaciones causa-efecto, que se puedan generalizar; es decir, hay un proceso lineal de relación teoría-empiria, con diferentes espacios y momentos para la definición de las hipótesis, la obtención de los datos, su análisis y su posterior interpretación.

Dimensión de las Técnicas de Obtención y Análisis de Información Empírica

Técnicas de Obtención y Análisis

De los procesos de simulación de los modelos de decisión y operadores de agregación que fueron desarrollados se obtuvo la información para su análisis. Se describen a continuación las estructuras de datos con las cuales se realizó el trabajo y el análisis y tratamiento de los datos efectuados.

Estructuras de datos

Se utilizó un sistema de matrices de datos construidas en base a las siguientes premisas: Se manejan grupos de procesos distribuidos en nodos de procesos con acceso a recursos críticos compartidos en modo de exclusión mútua distribuida y que, ante la solicitud de recursos por parte de los procesos, debe decidirse cuáles serán las prioridades para la asignación de los recursos a los procesos que lo solicitan. Debe resaltarse que los recursos a ser asignados deberán estar disponibles (no asignado previamente a otro proceso determinado):

• El permiso de acceso a los recursos compartidos que son propios de un nodo no estará en función de que el nodo los esté solicitando o no, sino del valor de agregación de las opiniones

(prioridades) de los distintos nodos respecto de otorgar el acceso a los recursos compartidos (alternativas).

• La consideración del valor de las variables que representan el estado de cada uno de los distintos nodos definirán las opiniones (prioridades) de los distintos nodos para otorgar el acceso a los recursos compartidos (alternativas). Cada nodo debe indicar sus prioridades a fin de que se le puedan asignar los distintos recursos compartidos con relación a los requerimientos de recursos de cada proceso de cada grupo.

La nomenclatura de las variables que fueron utilizadas es la siguiente:

Nodos que alojan procesos: 1,…, *n*.

Procesos alojados en cada uno de los *n* nodos: 1,…, *p*.

Grupos de procesos distribuidos: 1,…, *g*.

Tamaño de cada uno de los *g* grupos de procesos: 1,…, *t*.

Recursos críticos compartidos en la modalidad de exclusión mutua distribuida disponibles en cada uno de los *n* nodos: 1,…, *r*.

• Estados posibles de cada uno de los *p* procesos:

Los posibles estados de cada uno de los *p* procesos son: a) Grupo al que pertenece el proceso (0 significa proceso independiente); b) Requiere sincronización con los demás procesos del grupo al que pertenece (los procesos del grupo deben estar activos en sus respectivos procesadores en un mismo lapso de tiempo), (0 significa que no requiere sincronización, 1 significa que sí la requiere); c) En espera de un recurso compartido con el grupo de procesos al que pertenece; d) En espera de un recurso no compartido con el grupo de procesos al que pertenece; e) En ejecución con permiso de acceso a un recurso compartido con el grupo de procesos al que pertenece: f) En ejecución sin permiso de acceso a un recurso compartido con el grupo de procesos al que pertenece; y g) Inactivo.

• Estado posible de cada uno de los *n* nodos:

Los posibles estados de cada uno de los *n* nodos son: a) Número de procesos; b) Prioridades de los procesos; c: Uso de CPU; d) Uso de memoria principal; y e) Uso de memoria virtual.

• Estado posible de cada uno de los *r* recursos:

Los posibles estados de cada uno de los *r* recursos críticos compartidos en la modalidad de exclusión mutua distribuida existentes en el nodo: a) Asignado a un proceso local; b) Asignado a un proceso remoto.; c) Disponible.

• Predisposición (prioridad nodal) para otorgar el acceso a cada uno de los *r* recursos críticos compartidos (alternativas) en la modalidad de exclusión mutua distribuida (resultará de la consideración de las variables representativas del estado del nodo, para cada recurso crítico compartido existente). Se obtendrá una tupla por cada uno de los *n* nodos, cada tupla contendrá *r* valores (prioridades nodales) para compartir los recursos críticos.

Estado global del sistema:

a) Número de grupos de procesos y tamaño (número *t* de procesos) de cada uno de los *g* grupos.

b) Porcentajes de consenso requeridos para otorgar el acceso a cada uno de los *r* recursos críticos disponibles en cada uno de los *n* nodos.

c) Predisposición (prioridad global) para otorgar el acceso a cada uno de los *r* recursos críticos compartidos (alternativas) en la modalidad de exclusión mutua distribuida (resultará de la agregación de las prioridades nodales para cada recurso crítico compartido existente (alternativas). Se obtendrá una tupla con *r* valores normalizados (prioridades globales) para compartir los recursos críticos.

d) Decisión de acceso a los *r* recursos críticos en función de contrastar las prioridades

globales normalizadas para compartir los mismos con los porcentajes de consenso requeridos para otorgar los respectivos accesos.

El estado global del sistema deberá actualizarse reiteradamente mientras haya alguno o algunos de los *p* procesos que requieran acceder a alguno o algunos de los *r* recursos compartidos.

El sistema se autoregula reiteradamente en función del estado local de los *n* nodos y del estado global del sistema, produciéndose una actualización de los estados locales de los nodos como consecuencia de la evolución de sus respectivos procesos y de las decisiones de acceso a los recursos críticos producidas teniendo en cuenta el estado global del sistema: el sistema distribuido en el que se ejecutan grupos de procesos que acceden a recursos críticos, se observa a sí mismo y produce decisiones de accesos a recursos que modifican el estado del sistema y lo reajustan reiterativamente.

Es preciso señalar que se tiene previsto trabajar con variables difusas utilizando etiquetas lingüísticas para su representación. En tales casos las valoraciones deben expresarse de manera lingüística utilizando etiquetas lingüísticas y sus semánticas correspondientes.

Tratamiento y análisis de datos

Se efectuaron simulaciones con los modelos de decisión propuestos y los modelos de decisión computacionales clásicos, a los efectos de analizar el comportamiento de estos para las mismas condiciones de carga de trabajo y de consumo de recursos, dando lugar a un esquema iterativo de modificación de los modelos propuestos para intentar lograr un desempeño de los mismos al menos equivalente al de los modelos computacionales clásicos.

1.10. Estructura de la Tesis

Se ha presentado el problema que dio origen a la idea para la realización de este trabajo de investigación, así como los antecedentes más relevantes y los conceptos teóricos que constituyen el marco conceptual. También se mencionó la metodología a utilizar. Seguidamente se indican los

capítulos restantes con los cuales se ha estructurado esta tesis.

Capítulo II - Operador de agregación definido para el Escenario 1 (E1). En este Capítulo se han efectuado los siguientes pasos: se ha estudiado el escenario general, se ha formalizado el operador de agregación, se ha validado empíricamente el operador de agregación se ha definido y validado teóricamente el modelo global de decisión, se ha definido detalladamente las etapas, niveles o capas del modelo de decisión y validado empíricamente y, se ha evaluado y comparado los resultados del modelo de decisión.

Capítulo III - Operador de agregación definido para el Escenario 2 (E2). En este capítulo se han efectuado los siguientes pasos: se ha estudiado el escenario, se ha formalizado el operador de agregación, se ha validado empíricamente el operador de agregación, se ha definido y validado teóricamente el modelo global de decisión, se ha definido detalladamente las etapas, niveles o capas del modelo de decisión y validado empíricamente y, se ha evaluado y comparado los resultados del modelo de decisión.

Capítulo IV - Operador de agregación definido para el escenario 3 (E3). En este capítulo se han efectuado los siguientes pasos: se ha estudiado el escenario, se ha formalizado el operador de agregación, se ha validado empíricamente el operador de agregación, se ha definido y validado teóricamente el modelo global de decisión, se ha definido detalladamente las etapas, niveles o capas del modelo de decisión y validado empíricamente y, se ha evaluado y comparado los resultados del modelo de decisión.

Capítulo V - Modelo de decisión: Se ha desarrollado y explicado el modelo de decisión propuesto para poder aplicar cada uno de los operadores de agregación para cada escenario.

Capítulo VI - Conclusiones y futuras líneas de investigación: se han comentado las principales conclusiones y se han indicado las posibles líneas futuras de investigación.

1.11. Discusiones y Comentarios

Se realizó una exposición de los antecedentes y el marco teórico en el cual se detallan los principales aspectos del estado del arte en las distintas áreas relacionadas con la presente tesis (cibernética, comunicación y sincronización en sistemas distribuidos, toma de decisiones en grupo, etc). Se planteó la hipótesis y los objetivos de la investigación, donde se contemplan situaciones en las que los procesos puedan acceder a recursos compartidos en la modalidad de exclusión mutua constituyendo o no grupos de procesos, que requieran o no sincronización y que puedan tener o no exigencias estrictas de consenso para obtener el acceso a los recursos.

También se describieron las estructuras de datos, los elementos que intervienen dentro del sistema, los nodos, recursos y procesos, con prioridades y cargas de trabajo y que en diferentes situaciones permiten que el sistema se autorregule de manera reiterada en función del estado local de los n nodos y del estado global del sistema.

Finalmente se ha hecho una descripción de los capítulos con que cuenta la estructura de la tesis. Además de lo relatado precedentemente, continúa con tres capítulos referentes a los distintos escenarios en los cuales se hace explícito el desarrollo de varios operadores de agregación, un capítulo sobre el modelo de decisión propuesto, finalizándose luego con las conclusiones y líneas futuras de investigación.

Lo indicado en la descripción de la estructura de la tesis inicia en el siguiente capítulo.

Capítulo II – Operador de agregación definido para el escenario 1 (E1)

2.1. Trabajos previos

En La Red Martínez (2017) se desarrollan operadores de agregación para la asignación de recursos en sistemas distribuidos.

2.2. Premisas

Se trata de grupos de procesos distribuidos en nodos de procesos que acceden a recursos críticos compartidos en la modalidad de exclusión mutua distribuida, debiendo decidirse, ante la demanda de recursos por parte de los procesos, cuáles serán las prioridades para asignar los recursos a determinador procesos o grupos de procesos que los requieren (sólo intervendrán como alternativas de asignación a los procesos aquellos recursos disponibles, es decir, no asignados aún a determinados procesos):

- El permiso de acceso a los recursos compartidos propios de un nodo no dependerá sólo de si los nodos los están utilizando o no, sino del valor de agregación de las preferencias (prioridades) de los distintos nodos respecto de otorgar el acceso a los recursos compartidos (alternativas).

- Las opiniones (prioridades) de los distintos nodos respecto de otorgar el acceso a los recursos compartidos (alternativas) dependerá de la consideración del valor de variables que representen el estado de cada uno de los distintos nodos. Cada nodo deberá expresar sus prioridades para la asignación de los distintos recursos compartidos respecto de los requerimientos de recursos de cada proceso.

En algunas situaciones se considerar necesario establecer un consenso, que se entiende por priorizar a los procesos o grupo de procesos que resulten con mayor prioridad global promedio en

sus asignaciones, según las tablas de asignaciones.

El escenario E1 (pasos del 1 al 9 de la Fig. 1) consiste en que el *Runtime* o Coordinador Central, reciba la información proveniente de los distintos nodos y a través de un operador de agregación determina y soluciona la siguiente premisa:

- Que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua **sin constituir grupos de procesos**, los procesos no requieren sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y **sin exigencias estrictas de consenso** para lograr el acceso (no se requiere un consenso para asignar de manera consecutiva los recursos solicitados por un proceso o grupo de procesos, es decir, que iniciada la secuencia de asignación de recursos a un proceso, la misma puede ser interrumpida para asignar recursos a otro proceso).

El escenario E2, comienza a partir del resultado del escenario anterior (pasos del 10 al 12 de la Fig. 1), utiliza la tabla final de prioridades (paso 9 de la Fig. 1), y a través de un operador de agregación propio de este escenario determina y soluciona la premisa establecida y desarrollada en el capítulo III.

El escenario E3 comienza a partir del resultado del escenario anterior (pasos del 13 al 16 de la Fig. 1), utiliza la tabla final de prioridades (paso 12 de la Fig. 1), y a través de un operador de agregación propio de este escenario determina y soluciona la premisa establecida y desarrollada en el capítulo IV.

La autoregulación que se muestra en la Fig. 1, corresponde a una ronda de asignación de recursos a procesos según el cálculo realizado en la ***FASDLNO*** (Función de Asignación del Sistema Distribuido Lingüística Normalizada), Tabla 30, P. 52.

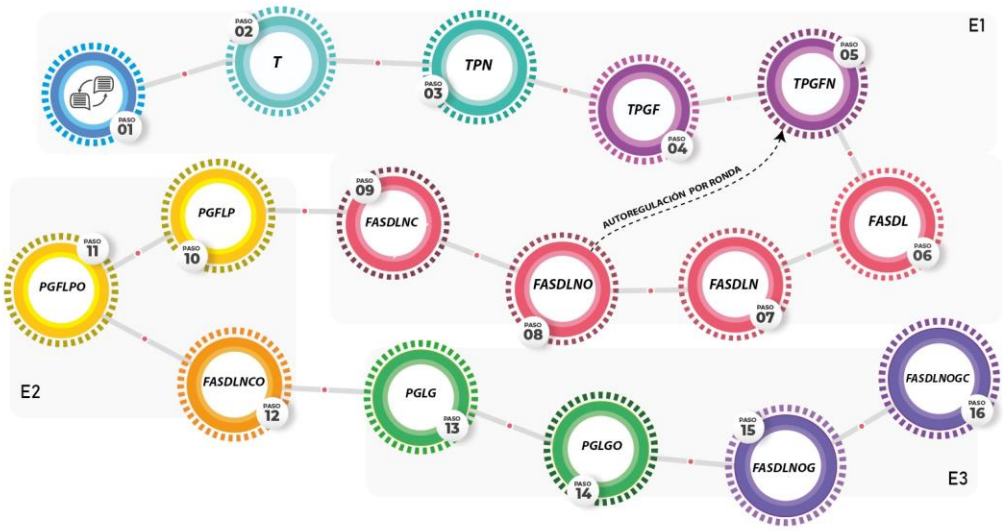


Fig. 1. Representación de un ciclo necesario para aplicar los operadores de agregación utilizados en los 3 escenarios propuestos.

Fuente: Elaboración propia.

2.3. Propuesta de solución

Se presentará una variante de un método innovador para la gestión de recursos compartidos en sistemas distribuidos, basado en La Red Martínez (2017) y La Red Martínez et al. (2018), donde se desarrolla un operador de agregación, para asignar recursos en sistemas distribuidos, pero en este caso, sin un modelo de consenso que favorezca el acceso secuencial de los procesos a los recursos solicitados. Las premisas, las estructuras de datos y el operador mencionado en las publicaciones mencionadas se utilizan como punto de partida para crear un nuevo operador en el escenario descrito a continuación.

El escenario propuesto considera las siguientes condiciones: en primer lugar, los procesos deben tener acceso a recursos compartidos en la modalidad de exclusión mutua, en segundo lugar, deben ser capaces de formar grupos de procesos (los procesos que no pertenezcan a grupos se consideran independientes), en tercer lugar, los procesos no deben requerir sincronización (es decir, estar activos en sus respectivos procesadores al mismo tiempo) y no hay requisitos de

consenso para acceder a los recursos (no se requiere un acuerdo para asignar los recursos solicitados a los procesos de cada grupo, es decir, una vez iniciada la secuencia de asignación de recursos a un proceso, no se puede interrumpir para otorgársela a otro proceso, hasta que el proceso activo los libere).

2.4. Descripción del operador de agregación

El operador propuesto consta de las siguientes etapas:

1. Expresión de los valores calculados en términos de 2-tupla utilizando un conjunto de etiquetas lingüísticas.

2. Cálculo de la carga computacional actual de los nodos.

3. Establecimiento de las categorías de carga computacional y de los vectores de pesos asociados a las mismas.

4. Cálculo de las prioridades o preferencias de los procesos teniendo en cuenta el estado del nodo (se las calcula en cada nodo para cada proceso).

5. Cálculo de las prioridades o preferencias de los procesos para acceder a los recursos compartidos disponibles (se las calcula en el administrador centralizado de recursos compartidos) y determinación del orden en que se asignarán los recursos y a qué proceso será asignado cada recurso.

Expresión de los valores calculados en términos de 2-tupla usando un conjunto de etiquetas lingüísticas

El conjunto de etiquetas utilizado puede variar en cada nodo, es decir, que el conjunto es independiente del nodo central, y es este el encargado de realizar la traslación simbólica, según se

indica en Dutta, Labella, Rodríguez y Martínez (2019), proceso que consiste en convertir los conjuntos de etiquetas recibidos en el nodo central, en el conjunto que este último trabaja, el cual debe ser el de mayor número de etiquetas. De igual manera, esa traslación simbólica se realiza cuando el nodo central devuelve la información final de prioridades a cada nodo, como se puede ver en la Fig. 2.

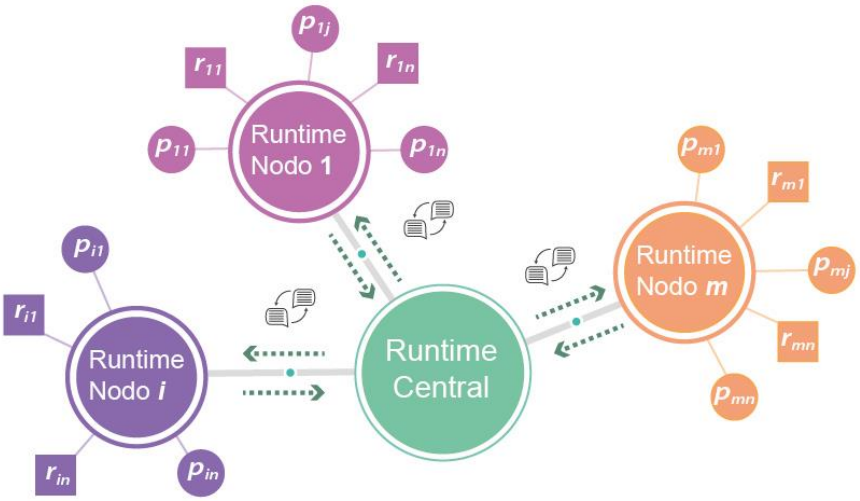


Fig. 2. Proceso de traslación de etiquetas que recibe y envía el nodo central.

Fuente: Elaboración propia.

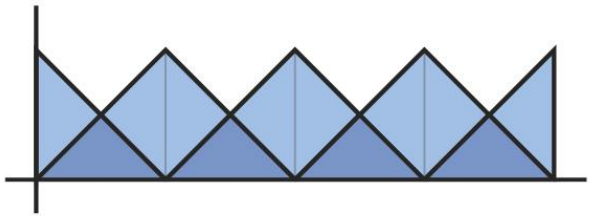


Fig. 3. Representación del conjunto de etiquetas.

Fuente: Elaboración propia.

Tabla 1. Semántica correspondiente al conjunto de etiquetas de ejemplo

Etiquetas	Semánticas
MA: Muy Alta prioridad	0.75; 1; 1
A: Alta prioridad	0.50; 0.75; 1
ME: Media prioridad	0.25; 0.50; 0.75
B: Baja prioridad	0; 0.25; 0.50
MB: Muy Baja prioridad	0; 0; 0.25

Fuente: Elaboración propia.

Un ejemplo de las valoraciones se expresan en un formato lingüístico utilizando las etiquetas lingüísticas y semánticas mencionadas se pueden ver en Fig. 3 y Tabla 1, aunque estos conjuntos de etiquetas y su significado podrían variar.

En cada nodo se define una interfaz entre las aplicaciones y el sistema operativo, que a través de un *Runtime* (software en tiempo de ejecución complementario al sistema operativo) incluido en esa interfaz, gestiona los procesos y recursos compartidos y define el escenario correspondiente. Además, los *Runtime* interactúan entre sí para intercambiar información y en uno de los nodos hay un coordinador global de *Runtime* que evalúa y ejecuta el modelo de decisión y el operador de agregación correspondiente. A continuación, se describe cada uno de los pasos anteriores.

Cálculo de la carga actual de los nodos

Para obtener un indicador de la carga computacional actual de cada nodo, se pueden adoptar diferentes criterios; en esta propuesta los criterios serán el porcentaje de uso de la CPU, el porcentaje de uso de la memoria y el porcentaje de uso de la operación de entrada/salida. La carga computacional de cada nodo, el número de criterios para determinar la carga de los nodos, los criterios que se aplican y el cálculo de la carga computacional de cada nodo, son los mencionados en La Red Martínez (2017).

El establecimiento de las categorías de carga de cálculo y de los vectores de pesos asociados a ellos

Las categorías de carga computacional actuales de cada nodo, el número de categorías para determinar la carga de los nodos, las categorías que se aplican, los vectores de pesos asociados a las categorías de carga computacional actuales de cada nodo. En esta propuesta, los criterios serán

los utilizados en La Red Martínez (2017).

Establecimiento de vectores de pesos (iguales para todos los nodos): pesos = $\{w_{ij}\}$ con i = 1,..., a (número de categorías de carga computacional) y j = 1, ..., e (número máximo de criterios).

Cálculo de las prioridades o preferencias de los procesos teniendo en cuenta el estado del nodo (se calculan en cada nodo para cada proceso y podrían denominarse prioridades nodales)

Estas prioridades se calculan en cada nodo para cada solicitud de recursos originada en cada proceso; el cálculo considera el vector de peso correspondiente según la carga actual del nodo y el vector de los valores concedidos por el nodo según los criterios de evaluación de la solicitud.

Los vectores de valoración que se aplicarán para cada solicitud de un recurso por un proceso, de acuerdo con los criterios establecidos para la determinación de la prioridad que en cada caso y momento fijará el nodo en el que se produce la solicitud, son los siguientes: valoraciones $(r_{ij}\ p_{kl}) = \{cp_m\}$ con i = 1, ..., n (nodo donde reside el recurso), j = 1, ..., r (recurso en el nodo i), k = 1, ..., n (nodo donde reside el proceso), l = 1, ..., p (proceso en el nodo k) y m = 1, ..., e (criterio de valoración de la prioridad de la solicitud).

Estos valores son representados en el formato de 2-tupla, considerando las etiquetas lingüísticas propuestas anteriormente. Por lo tanto, el valor de cada criterio deberá compararse con el valor medio de cada etiqueta, la diferencia mínima de esa comparación será la etiqueta apropiada.

El primer elemento de la 2-tupla será el valor lingüístico de esa etiqueta. El segundo elemento será la diferencia entre el valor de los criterios buscados y el valor medio de la etiqueta seleccionada.

dm = la diferencia mínima entre las diferencias de cp_m y el valor más representativo de cada etiqueta lingüística.

Tabla 2. Valoraciones asignadas a los criterios de cálculo de la prioridad o preferencia que cada nodo dará a cada requerimiento de cada proceso según la carga del nodo en 2-tuplas

Recursos - Procesos	Criterio 2-tupla				
$r_{11}\ p_{11}$	T(*etiqueta*$_1$; dm$_1$)	…	T(*etiqueta*$_m$; dm$_m$)	…	T(*etiqueta*$_e$; dm$_e$)
….	….	…	…	…	….
$r_{ij}\ p_{kl}$	T(*etiqueta*$_1$; dm$_1$)	…	T(*etiqueta*$_m$; dm$_m$)	…	T(*etiqueta*$_e$; dm$_e$)
….	….	…	…	…	….
$r_{nr}\ p_{np}$	T(*etiqueta*$_1$; dm$_1$)	…	T(*etiqueta*$_m$; dm$_m$)	…	T(*etiqueta*$_e$; dm$_e$)

Fuente: Elaboración propia.

Valoraciones de la etiqueta ($r_{ij}\ p_{kl}$) = 2-tupla = T(etiqueta$_m$; dm$_m$) donde el subíndice *m* corresponde a las etiquetas lingüísticas definidas anteriormente, como puede verse en la Tabla 2.

En resumen, la prioridad nodal (que debe calcularse en el nodo en el que se produce la solicitud) de un proceso de acceso a un recurso determinado (que puede ser en cualquier nodo) se calcula mediante el producto escalar de los vectores mencionados: prioridad nodal ($r_{ij}\ p_{kl}$) = $\Sigma\ w_{om}$ * T(etiqueta$_m$; dm$_m$) = T(etiqueta$_n$; dm$_n$) = TPN$_{ijkl}$ (Tupla de Prioridad Nodal) con la *o* indicando el vector de pesos según la carga del nodo, manteniendo todos los demás subíndices los significados explicados anteriormente. Con *m* y *n* indicando la etiqueta lingüística correspondiente dentro del conjunto adoptado definido anteriormente.

Esta prioridad nodal debe transformarse en el formato de 2-tuplas, teniendo en cuenta las etiquetas lingüísticas ya mencionadas. Por lo tanto, será necesario comparar cada valor de prioridad nodal con el valor medio de cada etiqueta, la diferencia mínima de estas comparaciones indicará la etiqueta correspondiente.

Cálculo de las prioridades o preferencias de los procesos para acceder a los recursos compartidos disponibles (se las calcula en el administrador centralizado de recursos compartidos) y determinación del orden en que se asignarán los recursos y a qué proceso será

asignado cada recurso

La Tabla 3 se utiliza para calcular las prioridades finales, en el que se colocan las prioridades o preferencias nodales calculadas en la etapa anterior; en esta tabla cada fila contiene la información de las prioridades nodales de los diferentes procesos para acceder a un determinado recurso.

Tabla 3. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla

Recursos	2-tupla				
r_{11}	TPN_{1111}	...	TPN_{11kl}	...	TPN_{11np}
...	...	...	...	...	...
r_{ij}	TPN_{ij11}	...	TPN_{ijkl}	...	TPN_{ijnp}
...	...	...	...	...	...
r_{nr}	TPN_{nr11}	...	TPN_{npkl}	...	TPN_{nrnp}

Fuente: Elaboración propia.

A continuación, es necesario calcular el vector de pesos finales que se utilizará en el proceso de agregación para determinar el orden o la prioridad de acceso a los recursos.

Pesos finales = $\{wf_{kl}\}$ con k = 1, ..., n (número de nodos) y l = 1, ..., p (número máximo de procesos por nodo), donde np es el número de procesos en el sistema y prg_i es la prioridad del grupo de procesos al que pertenece el proceso.

El siguiente paso es normalizar los pesos recién obtenidos dividiendo cada uno de ellos por la suma de todos. Así se obtiene un vector de peso normalizado (en el rango de 0 a 1 inclusive) y con la restricción de que la suma de los elementos del vector debe dar 1:

$\Sigma \{nwf_{kl}\} = 1$ con k = 1,..., n (número de nodos) y l = 1,..., p (número máximo de procesos por nodo).

Las prioridades nodales tomadas fila por fila para cada recurso serán multiplicadas escalarmente por el vector de peso final normalizado. De esta manera es posible obtener la prioridad global final de acceso de cada proceso a cada recurso. A continuación, se indica cómo

se obtiene el orden o la prioridad con que se asignarán los recursos y a qué proceso se asignará cada uno.

Prioridad global final ($r_{ij}\ p_{kl}$) = TPN_{ijkl} = $TPGF_{ijkl}$ (Tupla de prioridad global final) con el r_{ij} indicando el recurso *j* del nodo *i*, TPN_{ijkl} es el formato de 2-tuplas, *ij* indicando el recurso *j* del nodo *i*, *kl* el proceso *l* del nodo *k* y el producto de la prioridad global final del proceso para acceder a dicho recurso, como se puede ver en la Tabla 4.

Tabla 4. Tupla de Prioridad Global Final (***TPGF***)

Recursos	Prioridades Nodales de los Procesos				
r_{11}	$TPGF_{1111}$	...	$TPGF_{11kl}$	...	$TPGF_{11np}$
...	...	...	...	...	...
r_{ij}	$TPGF_{ij11}$	...	$TPGF_{ijkl}$	...	$TPGF_{ijnp}$
...	...	...	...	...	...
r_{nr}	$TPGF_{r11}$	...	$TPGF_{pkl}$	...	$TPGF_{rnp}$

Fuente: Elaboración propia.

El siguiente paso es normalizar la Tabla 4 entre los valores extremos. Esto se hará utilizando los valores máximo, mínimo y rango calculados de la Tabla 4 y representados en la Tabla 5.

Tabla 5. Cálculo de los valores máximo, mínimo y el rango

Etiqueta	Valor
Valor Máximo	Máximo ($TPGF_{ijkl}$)
Valor mínimo	Mínimo ($TPGF_{ijkl}$)
Rango	Máximo ($TPGF_{ijkl}$) - Mínimo ($TPGF_{ijkl}$)

Fuente: Elaboración propia.

Para normalizar, el valor numérico de cada 2-tupla debe restarse del valor mínimo y dividirse por el rango, que es la diferencia entre el valor máximo y el valor mínimo, para obtener el *TPGFN* (Tupla de Prioridad Global Final Normalizada). Esto puede verse en la Tabla 6.

Tabla 6. Tupla de Prioridad Global Final Normalizada (***TPGFN***)

Recursos	2-tupla				
r_{11}	$TPGFN_{1111}$	...	$TPGFN_{11kl}$	...	$TPGFN_{11np}$
…	…	…	…	…	…
r_{ij}	$TPGFN_{ij11}$	...	$TPGFN_{ijkl}$	...	$TPGFN_{ijnp}$
…	…	…	…	…	…
r_{nr}	$TPGFN_{nr11}$	...	$TPGFN_{npkl}$	...	$TPGFN_{nrnp}$

Fuente: Elaboración propia.

El mayor de estos productos realizados para los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que se le asignará a ese recurso, en relación con los demás recursos que también deberán asignarse. Esto es lo que se denominará Función de Asignación de Sistemas Distribuidos Lingüística (*FASDL*):

$FASDL\ (r_{ij}) = \Sigma\ TPGFN_{ijkl}$ = prioridad de asignación de recursos r_{ij}.

Calculando el *FASDL* para todos los recursos se obtendrá un vector 2-tupla, y ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos. Estos deben ser normalizados garantizando que las 2-tuplas obtenidas se encuentran en el intervalo [0, 1], para ello, se utilizarán los valores máximo, mínimo y rango, como puede verse en la Tabla 7.

Tabla 7. Valoraciones para normalizar el ***FASDL***

Etiqueta	Valor
Valor Máximo	Máximo ($FASDL_{ijkl}$)
Valor mínimo	Mínimo ($FASDL_{ijkl}$)
Rango	Máximo ($FASDL_{ijkl}$) – Mínimo ($FASDL_{ijkl}$)

Fuente: Elaboración propia.

Esto es lo que se denominará Función de Asignación de Sistemas Distribuidos Lingüísticos Normalizados (*FASDLN*):

$FASDLN(r_{ij}) = \Sigma\ (TPGFN_{ijkl}$ / (máximo($TPGFN_{ijkl}$) - mínimo($TPGFN_{ijkl}$))) = prioridad de asignación de recursos r_{ij} normalizada entre valores extremos.

La Tabla 8 muestra el proceso que le corresponde a cada recurso de acuerdo al orden en el que están en el vector *FASDLN*.

Tabla 8. Prioridad final de asignación de recursos y proceso al que se asigna cada recurso en la primera iteración (***FASDLN***)

Orden de asignación de los recursos	*Proceso al que se asignará el recurso*
1°: r_{ij} del primer(*FASDLN*(r_{ij}))	p_{kl} del Max(*TPGFN*$_{ijkl}$) para el r_{ij} selecionado
2°: r_{ij} del segundo(*FASDLN*(r_{ij})) para los r_{ij} no asignados	p_{kl} del Max(*TPGFN*$_{ijkl}$) para el r_{ij} selecionado
….	….
último: r_{ij} no asignado	p_{kl} del Max(*TPGFN*$_{ijkl}$) para el r_{ij} selecionado

Fuente: Elaboración propia.

Como ya se ha indicado, el mayor de los productos *TPGFN*$_{ijkl}$ para cada recurso indicará el proceso al que se asignará el recurso, como se puede ver en la Fig. 4.

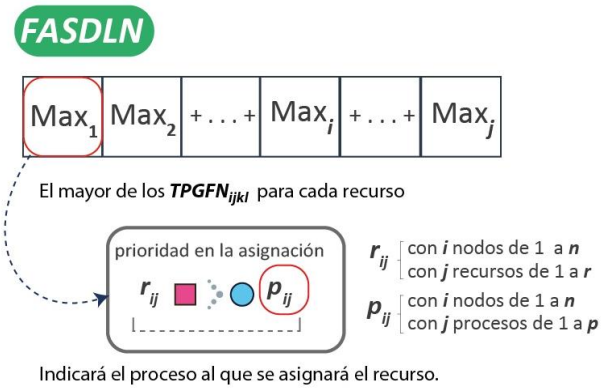


Fig. 4. Orden de prioridad de cada proceso sobre cada recurso.

Fuente: Elaboración propia.

El siguiente paso es ordenar de mayor a menor las prioridades del vector *FASDLN* de la Tabla 8, para obtener la *FASDLNO*, lo cual se puede observar en la Tabla 9.

Tabla 9. Prioridad final ordenada de asignación de recursos y proceso al que se asigna cada recurso en la primera iteración (***FASDLNO***)

Orden de asignación de los recursos	*Proceso al que se asignará el recurso*
1°: r_{ij} del Max(*FASDLN*(r_{ij}))	p_{kl} del Max(*TPGFN*$_{ijkl}$) para el r_{ij} selecionado
2°: r_{ij} del Max(*FASDLN*(r_{ij})) para los r_{ij} no asignados	p_{kl} del Max(*TPGFN*$_{ijkl}$) para el r_{ij} selecionado
….	….
último: r_{ij} no asignado	p_{kl} del Max(*TPGFN*$_{ijkl}$) para el r_{ij} selecionado

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos. Se debe recalcular

la Tabla 7, Tabla 8, Tabla 9 omitiendo las asignaciones de recursos ya hechas.

2.5. Ejemplo

En esta sección se explicará en detalle un ejemplo de aplicación del operador de agregación propuesto. El sistema de procesamiento distribuido, las estructuras de datos, los recursos y los procesos que se ejecutan en los diferentes nodos, grupos, cardinalidades, criterios y categorías para evaluar las diferentes cargas y cálculos necesarios, son los mencionados en La Red Martínez (2017), Agostini et al. (2018) y Agostini et al. (2019a, 2019b, 2019c).

Expresión de los valores calculados en términos de 2-tupla usando un conjunto de etiquetas lingüísticas

Las valoraciones expresadas en un formato lingüístico utilizando las etiquetas lingüísticas y semánticas mencionadas pueden verse en la Tabla 10.

Tabla 10. Propuestas para valorar la prioridad

Nombre de la etiqueta	Valor mínimo	Valor más probable	Valor máximo
EA: Extremadamente Alta	0.83	1	1
MA: Muy Alta prioridad	0.67	0.83	1
A: Alta prioridad	0.5	0.67	0.83
ME: Media prioridad	0.33	0.5	0.67
B: Baja prioridad	0.17	0.33	0.5
MB: Muy Baja prioridad	0	0.17	0.33
EB: Extremadamente Baja	0	0	0.17

Fuente: Elaboración propia.

Todos los valores son representados en el formato de 2-tupla, considerando las etiquetas lingüísticas propuestas anteriormente. Por lo tanto, será necesario comparar cada valor del criterio con el valor medio de cada etiqueta, la diferencia mínima de esta comparación será la etiqueta apropiada.

El primer elemento de la 2-tupla será el valor lingüístico de esa etiqueta, mientras que el segundo elemento será la diferencia entre el valor del criterio buscado y el valor medio de la

etiqueta seleccionada. Esto puede verse en la Tabla 11 y Tabla 12, donde la "prioridad del proceso" está representada por los métodos considerados tradicionales, Tanenbaum (2009), Agrawal y El Abbadi (1991) y Ricart y Agrawala (1981).

Tabla 11. Valoraciones asignadas a los criterios para calcular la prioridad o preferencia que cada nodo otorgará a cada requerimiento de cada proceso según la carga del nodo (*n° proc., %cpu, %mem., %mv*)

Rec.	N°Proc	%CPU	%Mem	%MV
$p_{11}r_{11}$	T(A;0.0333)	T(M;0.0000)	T(A;0.0333)	T(MA;0.0667)
$p_{11}r_{12}$	T(MA;-0.0333)	T(A;0.0333)	T(B;0.0667)	T(M;0.0000)
$p_{11}r_{21}$	T(B;-0.0333)	T(B;0.0667)	T(M;0.0000)	T(MB;0.0333)
$p_{11}r_{22}$	T(M;0.0000)	T(M;0.0000)	T(A;0.0333)	T(B;0.0667)
$p_{11}r_{23}$	T(M;0.0000)	T(A;-0.0667)	T(MA;-0.0333)	T(MA;-0.0333)
$p_{11}r_{24}$	T(B;-0.0333)	T(M;0.0000)	T(MA;0.0667)	T(MB;0.0333)
$p_{12}r_{11}$	T(B;0.0667)	T(A;0.0333)	T(M;0.0000)	T(MA;0.0667)
$p_{12}r_{12}$	T(MB;0.0333)	T(A;0.0333)	T(B;-0.0333)	T(A;0.0333)
$p_{12}r_{21}$	T(A;0.0333)	T(B;0.0667)	T(B;-0.0333)	T(A;0.0333)
$p_{12}r_{22}$	T(MA;0.0667)	T(A;-0.0667)	T(A;0.0333)	T(A;0.0333)
$p_{12}r_{31}$	T(MB;0.0333)	T(M;0.0000)	T(A;0.0333)	T(A;0.0333)
$p_{12}r_{33}$	T(B;0.0667)	T(M;0.0000)	T(A;0.0333)	T(MA;0.0667)
$p_{13}r_{11}$	T(M;0.0000)	T(A;0.0333)	T(A;0.0333)	T(MA;-0.0333)
$p_{13}r_{12}$	T(A;0.0333)	T(MA;-0.0333)	T(A;0.0333)	T(B;0.0667)
$p_{13}r_{13}$	T(A;0.0333)	T(A;-0.0667)	T(A;0.0333)	T(MA;-0.0333)
$p_{13}r_{21}$	T(A;0.0333)	T(B;0.0667)	T(MA;0.0667)	T(B;-0.0333)
$p_{13}r_{22}$	T(M;0.0000)	T(MA;0.0667)	T(MA;-0.0333)	T(B;-0.0333)
$p_{13}r_{31}$	T(M;0.0000)	T(A;0.0333)	T(B;-0.0333)	T(A;-0.0667)
$p_{13}r_{32}$	T(A;-0.0667)	T(MA;0.0667)	T(B;-0.0333)	T(A;-0.0667)
$p_{13}r_{33}$	T(A;-0.0667)	T(MB;0.0333)	T(B;0.0667)	T(A;-0.0667)
$p_{21}r_{12}$	T(MB;0.0333)	T(MB;-0.0667)	T(B;-0.0333)	T(MA;-0.0333)
$p_{21}r_{13}$	T(MB;0.0333)	T(B;0.0667)	T(MA;-0.0333)	T(MA;0.0667)
$p_{21}r_{22}$	T(B;-0.0333)	T(M;0.0000)	T(MA;-0.0333)	T(MA;0.0667)
$p_{21}r_{23}$	T(A;0.0333)	T(M;0.0000)	T(A;-0.0667)	T(MA;-0.0333)
$p_{21}r_{31}$	T(A;0.0333)	T(M;0.0000)	T(MA;-0.0333)	T(A;-0.0667)
$p_{21}r_{33}$	T(A;0.0333)	T(B;0.0667)	T(MA;0.0667)	T(MA;-0.0333)
$p_{22}r_{21}$	T(M;0.0000)	T(B;0.0667)	T(A;0.0333)	T(MA;-0.0333)
$p_{22}r_{22}$	T(M;0.0000)	T(MA;0.0667)	T(A;-0.0667)	T(MA;-0.0333)
$p_{22}r_{31}$	T(M;0.0000)	T(B;0.0667)	T(M;0.0000)	T(MB;0.0333)
$p_{22}r_{33}$	T(MA;-0.0333)	T(B;0.0667)	T(B;-0.0333)	T(MB;0.0333)
$p_{23}r_{12}$	T(M;0.0000)	T(A;-0.0667)	T(MA;-0.0333)	T(B;-0.0333)
$p_{23}r_{24}$	T(A;-0.0667)	T(MB;0.0333)	T(MB;-0.0667)	T(A;0.0333)
$p_{23}r_{31}$	T(B;-0.0333)	T(MB;-0.0667)	T(B;0.0667)	T(MA;-0.0333)
$p_{23}r_{32}$	T(B;0.0667)	T(B;0.0667)	T(MA;-0.0333)	T(MB;0.0333)
$p_{23}r_{33}$	T(B;-0.0333)	T(A;-0.0667)	T(MA;-0.0333)	T(MB;0.0333)
$p_{24}r_{11}$	T(B;0.0667)	T(A;-0.0667)	T(A;0.0333)	T(MA;0.0667)
$p_{24}r_{12}$	T(B;-0.0333)	T(A;-0.0667)	T(A;0.0333)	T(MA;-0.0333)
$p_{24}r_{23}$	T(B;0.0667)	T(MA;0.0667)	T(B;0.0667)	T(MA;-0.0333)
$p_{24}r_{24}$	T(M;0.0000)	T(MA;-0.0333)	T(A;0.0333)	T(MA;0.0667)
$p_{25}r_{21}$	T(MB;0.0333)	T(MA;-0.0333)	T(MA;-0.0333)	T(MA;0.0667)
$p_{31}r_{13}$	T(A;-0.0667)	T(MA;0.0667)	T(A;-0.0667)	T(MA;0.0667)
$p_{31}r_{31}$	T(MA;-0.0333)	T(B;-0.0333)	T(MA;0.0667)	T(M;0.0000)
$p_{31}r_{33}$	T(B;0.0667)	T(A;0.0333)	T(A;0.0333)	T(M;0.0000)
$p_{32}r_{11}$	T(A;-0.0667)	T(MA;0.0667)	T(MA;-0.0333)	T(M;0.0000)

$p_{32}r_{12}$	T(A;0.0333)	T(B;0.0667)	T(A;-0.0667)	T(MA;0.0667)
$p_{32}r_{13}$	T(MA;-0.0333)	T(MA;0.0667)	T(EA;0.0000)	T(A;0.0333)
$p_{32}r_{23}$	T(MA;-0.0333)	T(MA;-0.0333)	T(EA;0.0000)	T(MA;0.0667)
$p_{33}r_{11}$	T(MB;0.0333)	T(A;0.0333)	T(MA;0.0667)	T(MA;-0.0333)
$p_{33}r_{12}$	T(MA;0.0667)	T(EA;0.0000)	T(MA;0.0667)	T(A;0.0333)
$p_{33}r_{13}$	T(MA;0.0667)	T(M;0.0000)	T(A;0.0333)	T(MA;0.0667)
$p_{33}r_{21}$	T(B;0.0667)	T(A;-0.0667)	T(A;0.0333)	T(MA;-0.0333)
$p_{33}r_{22}$	T(MA;0.0667)	T(B;0.0667)	T(A;0.0333)	T(MA;-0.0333)
$p_{33}r_{23}$	T(B;0.0667)	T(A;-0.0667)	T(MA;0.0667)	T(EA;0.0000)
$p_{33}r_{32}$	T(A;-0.0667)	T(A;-0.0667)	T(A;0.0333)	T(MA;-0.0333)
$p_{33}r_{33}$	T(A;-0.0667)	T(EA;0.0000)	T(A;0.0333)	T(B;0.0667)
$p_{34}r_{12}$	T(MA;-0.0333)	T(EA;0.0000)	T(B;-0.0333)	T(M;0.0000)
$p_{34}r_{13}$	T(MB;0.0333)	T(EA;0.0000)	T(MA;-0.0333)	T(M;0.0000)
$p_{34}r_{22}$	T(MA;0.0667)	T(MA;-0.0333)	T(A;-0.0667)	T(MA;-0.0333)
$p_{34}r_{23}$	T(B;0.0667)	T(A;-0.0667)	T(A;0.0333)	T(MA;-0.0333)
$p_{34}r_{24}$	T(MA;0.0667)	T(B;0.0667)	T(A;0.0333)	T(B;0.0667)
$p_{34}r_{31}$	T(M;0.0000)	T(A;-0.0667)	T(A;0.0333)	T(MA;0.0667)
$p_{34}r_{32}$	T(MA;-0.0333)	T(A;-0.0667)	T(MA;0.0667)	T(M;0.0000)
$p_{34}r_{33}$	T(B;0.0667)	T(A;-0.0667)	T(MA;-0.0333)	T(M;0.0000)
$p_{35}r_{12}$	T(MB;0.0333)	T(B;0.0667)	T(A;0.0333)	T(B;0.0667)
$p_{35}r_{13}$	T(MA;-0.0333)	T(MA;0.0667)	T(A;0.0333)	T(B;-0.0333)
$p_{35}r_{22}$	T(B;-0.0333)	T(MA;-0.0333)	T(MA;0.0667)	T(B;0.0667)
$p_{35}r_{24}$	T(M;0.0000)	T(MA;-0.0333)	T(MA;0.0667)	T(B;0.0667)
$p_{35}r_{31}$	T(MA;0.0667)	T(MA;-0.0333)	T(A;0.0333)	T(B;0.0667)
$p_{35}r_{32}$	T(MA;0.0667)	T(A;0.0333)	T(MA;-0.0333)	T(A;-0.0667)
$p_{35}r_{33}$	T(M;0.0000)	T(A;0.0333)	T(A;-0.0667)	T(MA;0.0667)
$p_{36}r_{11}$	T(MA;-0.0333)	T(MA;0.0667)	T(A;-0.0667)	T(M;0.0000)
$p_{36}r_{12}$	T(A;0.0333)	T(MA;0.0667)	T(B;0.0667)	T(MA;0.0667)
$p_{36}r_{13}$	T(MA;0.0667)	T(MA;0.0667)	T(MA;-0.0333)	T(A;0.0333)
$p_{36}r_{21}$	T(MA;-0.0333)	T(MA;0.0667)	T(M;0.0000)	T(B;-0.0333)
$p_{36}r_{22}$	T(MA;-0.0333)	T(MB;0.0333)	T(MB;-0.0667)	T(B;-0.0333)
$p_{36}r_{24}$	T(B;0.0667)	T(A;0.0333)	T(MA;0.0667)	T(B;-0.0333)
$p_{36}r_{31}$	T(M;0.0000)	T(MA;0.0667)	T(MA;0.0667)	T(A;0.0333)
$p_{36}r_{32}$	T(MA;0.0667)	T(MA;-0.0333)	T(A;-0.0667)	T(A;0.0333)
$p_{36}r_{33}$	T(MB;0.0333)	T(A;0.0333)	T(B;0.0667)	T(MA;-0.0333)
$p_{37}r_{11}$	T(MA;0.0667)	T(A;-0.0667)	T(MA;-0.0333)	T(A;-0.0667)
$p_{37}r_{12}$	T(A;0.0333)	T(MA;0.0667)	T(MA;-0.0333)	T(A;0.0333)
$p_{37}r_{21}$	T(MA;0.0667)	T(A;0.0333)	T(MA;-0.0333)	T(A;-0.0667)
$p_{37}r_{32}$	T(MA;-0.0333)	T(MA;0.0667)	T(M;0.0000)	T(B;-0.0333)
$p_{37}r_{33}$	T(MA;-0.0333)	T(B;0.0667)	T(A;-0.0667)	T(MA;-0.0333)

Fuente: Elaboración propia.

Tabla 12. Valoraciones asignadas a los criterios para calcular la prioridad o preferencia que cada nodo otorgará a cada requerimiento de cada proceso según la carga del nodo (*prior. proc., sobrec. mem., sobrec. proc., sobre e/s*)

Recurso	Priorid Proc	Sobrec Mem	Sobrec Proc	Sobre E/S
$p_{11}r_{11}$	T(MA;-0.0333)	T(MB;0.0333)	T(B;-0.0333)	T(B;0.0667)
$p_{11}r_{12}$	T(B;-0.0333)	T(A;0.0333)	T(MB;0.0333)	T(B;0.0667)
$p_{11}r_{21}$	T(MA;0.0667)	T(MB;0.0333)	T(M;0.0000)	T(A;0.0333)
$p_{11}r_{22}$	T(MA;-0.0333)	T(B;-0.0333)	T(M;0.0000)	T(A;-0.0667)
$p_{11}r_{23}$	T(EA;-0.0500)	T(MA;0.0667)	T(A;0.0333)	T(A;-0.0667)
$p_{11}r_{24}$	T(A;-0.0667)	T(A;-0.0667)	T(A;0.0333)	T(B;0.0667)
$p_{12}r_{11}$	T(EA;0.0000)	T(MA;0.0667)	T(MA;-0.0333)	T(MA;-0.0333)
$p_{12}r_{12}$	T(MA;-0.0333)	T(B;-0.0333)	T(MA;-0.0333)	T(MA;0.0667)
$p_{12}r_{21}$	T(MA;-0.0333)	T(MA;0.0667)	T(M;0.0000)	T(MB;0.0333)

$p_{12}r_{22}$	T(MA;-0.0333)	T(MB;0.0333)	T(M;0.0000)	T(B;0.0667)
$p_{12}r_{31}$	T(B;-0.0333)	T(MB;0.0333)	T(A;0.0333)	T(MA;-0.0333)
$p_{12}r_{33}$	T(B;-0.0333)	T(B;0.0667)	T(M;0.0000)	T(MA;-0.0333)
$p_{13}r_{11}$	T(A;-0.0667)	T(M;0.0000)	T(A;0.0333)	T(MA;-0.0333)
$p_{13}r_{12}$	T(MA;0.0667)	T(MB;0.0333)	T(MA;0.0667)	T(A;0.0333)
$p_{13}r_{13}$	T(MA;0.0667)	T(B;0.0667)	T(MA;-0.0333)	T(A;0.0333)
$p_{13}r_{21}$	T(M;0.0000)	T(A;0.0333)	T(MB;0.0333)	T(B;-0.0333)
$p_{13}r_{22}$	T(M;0.0000)	T(B;0.0667)	T(MA;0.0667)	T(B;-0.0333)
$p_{13}r_{31}$	T(MA;-0.0333)	T(MA;0.0667)	T(MA;0.0667)	T(M;0.0000)
$p_{13}r_{32}$	T(B;0.0667)	T(MA;-0.0333)	T(A;0.0333)	T(MA;-0.0333)
$p_{13}r_{33}$	T(MA;0.0667)	T(MA;-0.0333)	T(M;0.0000)	T(MA;-0.0333)
$p_{21}r_{12}$	T(A;0.0333)	T(A;0.0333)	T(M;0.0000)	T(MA;-0.0333)
$p_{21}r_{13}$	T(A;0.0333)	T(B;0.0667)	T(M;0.0000)	T(MB;0.0333)
$p_{21}r_{22}$	T(M;0.0000)	T(A;-0.0667)	T(MB;0.0333)	T(B;0.0667)
$p_{21}r_{23}$	T(M;0.0000)	T(MB;0.0333)	T(MA;0.0667)	T(B;0.0667)
$p_{21}r_{31}$	T(MA;0.0667)	T(B;0.0667)	T(MA;0.0667)	T(MA;0.0667)
$p_{21}r_{33}$	T(B;0.0667)	T(MA;-0.0333)	T(A;-0.0667)	T(A;-0.0667)
$p_{22}r_{21}$	T(A;-0.0667)	T(B;0.0667)	T(A;-0.0667)	T(MA;0.0667)
$p_{22}r_{22}$	T(MA;0.0667)	T(B;0.0667)	T(MB;0.0333)	T(MB;-0.0667)
$p_{22}r_{31}$	T(B;0.0667)	T(MA;-0.0333)	T(MB;0.0333)	T(MA;0.0667)
$p_{22}r_{33}$	T(MA;-0.0333)	T(MA;0.0667)	T(M;0.0000)	T(MA;-0.0333)
$p_{23}r_{12}$	T(M;0.0000)	T(A;0.0333)	T(M;0.0000)	T(M;0.0000)
$p_{23}r_{24}$	T(B;-0.0333)	T(MA;-0.0333)	T(MA;0.0667)	T(B;0.0667)
$p_{23}r_{31}$	T(A;0.0333)	T(MA;0.0667)	T(B;0.0667)	T(A;-0.0667)
$p_{23}r_{32}$	T(B;0.0667)	T(A;-0.0667)	T(A;-0.0667)	T(A;-0.0667)
$p_{23}r_{33}$	T(MA;0.0667)	T(A;-0.0667)	T(B;0.0667)	T(MB;0.0333)
$p_{24}r_{11}$	T(M;0.0000)	T(A;0.0333)	T(MA;0.0667)	T(M;0.0000)
$p_{24}r_{12}$	T(MA;0.0667)	T(A;0.0333)	T(A;-0.0667)	T(MA;0.0667)
$p_{24}r_{23}$	T(MA;-0.0333)	T(A;0.0333)	T(A;-0.0667)	T(B;-0.0333)
$p_{24}r_{24}$	T(B;-0.0333)	T(B;0.0667)	T(MA;-0.0333)	T(A;0.0333)
$p_{25}r_{21}$	T(B;0.0667)	T(M;0.0000)	T(A;-0.0667)	T(MA;-0.0333)
$p_{31}r_{13}$	T(A;0.0333)	T(B;0.0667)	T(MA;0.0667)	T(MA;-0.0333)
$p_{31}r_{31}$	T(A;0.0333)	T(B;-0.0333)	T(MA;0.0667)	T(A;0.0333)
$p_{31}r_{33}$	T(MA;0.0667)	T(MA;0.0667)	T(A;0.0333)	T(A;0.0333)
$p_{32}r_{11}$	T(MA;0.0667)	T(A;0.0333)	T(B;-0.0333)	T(A;0.0333)
$p_{32}r_{12}$	T(MA;-0.0333)	T(EA;0.0000)	T(MA;0.0667)	T(A;0.0333)
$p_{32}r_{13}$	T(MA;0.0667)	T(B;-0.0333)	T(M;0.0000)	T(MA;0.0667)
$p_{32}r_{23}$	T(A;-0.0667)	T(MA;-0.0333)	T(B;0.0667)	T(MA;0.0667)
$p_{33}r_{11}$	T(A;-0.0667)	T(MA;0.0667)	T(EA;0.0000)	T(B;-0.0333)
$p_{33}r_{12}$	T(B;-0.0333)	T(M;0.0000)	T(A;0.0333)	T(B;-0.0333)
$p_{33}r_{13}$	T(B;-0.0333)	T(B;0.0667)	T(M;0.0000)	T(MA;-0.0333)
$p_{33}r_{21}$	T(MA;-0.0333)	T(B;0.0667)	T(MA;-0.0333)	T(MA;-0.0333)
$p_{33}r_{22}$	T(A;0.0333)	T(B;0.0667)	T(MA;0.0667)	T(A;0.0333)
$p_{33}r_{23}$	T(A;-0.0667)	T(B;0.0667)	T(A;0.0333)	T(MA;-0.0333)
$p_{33}r_{32}$	T(B;0.0667)	T(M;0.0000)	T(MA;-0.0333)	T(MA;-0.0333)
$p_{33}r_{33}$	T(A;-0.0667)	T(MA;-0.0333)	T(MA;-0.0333)	T(MA;0.0667)
$p_{34}r_{12}$	T(A;0.0333)	T(B;-0.0333)	T(MA;-0.0333)	T(A;0.0333)
$p_{34}r_{13}$	T(MA;-0.0333)	T(A;-0.0667)	T(MA;0.0667)	T(A;0.0333)
$p_{34}r_{22}$	T(MA;0.0667)	T(MA;-0.0333)	T(M;0.0000)	T(A;-0.0667)
$p_{34}r_{23}$	T(MA;0.0667)	T(M;0.0000)	T(MA;0.0667)	T(A;-0.0667)
$p_{34}r_{24}$	T(A;0.0333)	T(M;0.0000)	T(MA;0.0667)	T(A;0.0333)
$p_{34}r_{31}$	T(A;0.0333)	T(MA;0.0667)	T(A;-0.0667)	T(A;0.0333)
$p_{34}r_{32}$	T(A;-0.0667)	T(MA;0.0667)	T(B;-0.0333)	T(MA;0.0667)
$p_{34}r_{33}$	T(MA;0.0667)	T(MA;0.0667)	T(B;0.0667)	T(B;-0.0333)
$p_{35}r_{12}$	T(MA;0.0667)	T(M;0.0000)	T(A;0.0333)	T(MA;0.0667)
$p_{35}r_{13}$	T(MA;0.0667)	T(MA;-0.0333)	T(A;0.0333)	T(B;0.0667)
$p_{35}r_{22}$	T(MA;-0.0333)	T(A;-0.0667)	T(B;-0.0333)	T(A;-0.0667)

$p_{35}r_{24}$	T(A;-0.0667)	T(MA;0.0667)	T(B;0.0667)	T(A;0.0333)
$p_{35}r_{31}$	T(MA;-0.0333)	T(B;-0.0333)	T(B;0.0667)	T(MA;-0.0333)
$p_{35}r_{32}$	T(B;0.0667)	T(MA;0.0667)	T(B;0.0667)	T(A;0.0333)
$p_{35}r_{33}$	T(MA;-0.0333)	T(M;0.0000)	T(MA;0.0667)	T(A;0.0333)
$p_{36}r_{11}$	T(MA;-0.0333)	T(MA;0.0667)	T(MA;0.0667)	T(A;-0.0667)
$p_{36}r_{12}$	T(MA;-0.0333)	T(MA;0.0667)	T(B;0.0667)	T(A;0.0333)
$p_{36}r_{13}$	T(MA;-0.0333)	T(A;0.0333)	T(MA;0.0667)	T(A;0.0333)
$p_{36}r_{21}$	T(A;0.0333)	T(A;0.0333)	T(MA;0.0667)	T(MA;0.0667)
$p_{36}r_{22}$	T(MA;-0.0333)	T(A;0.0333)	T(A;-0.0667)	T(MA;0.0667)
$p_{36}r_{24}$	T(MA;-0.0333)	T(M;0.0000)	T(MA;-0.0333)	T(MA;0.0667)
$p_{36}r_{31}$	T(MA;-0.0333)	T(MA;-0.0333)	T(MA;-0.0333)	T(A;0.0333)
$p_{36}r_{32}$	T(MA;-0.0333)	T(M;0.0000)	T(A;-0.0667)	T(A;0.0333)
$p_{36}r_{33}$	T(MA;-0.0333)	T(MA;0.0667)	T(B;0.0667)	T(M;0.0000)
$p_{37}r_{11}$	T(MA;0.0667)	T(A;0.0333)	T(MA;0.0667)	T(MA;-0.0333)
$p_{37}r_{12}$	T(M;0.0000)	T(MA;-0.0333)	T(MA;-0.0333)	T(A;-0.0667)
$p_{37}r_{21}$	T(A;-0.0667)	T(MA;0.0667)	T(M;0.0000)	T(B;0.0667)
$p_{37}r_{32}$	T(MA;-0.0333)	T(A;0.0333)	T(B;0.0667)	T(A;-0.0667)
$p_{37}r_{33}$	T(MA;-0.0333)	T(A;-0.0667)	T(MA;0.0667)	T(B;-0.0333)

Fuente: Elaboración propia.

Cálculo de las prioridades o preferencias de los procesos teniendo en cuenta el estado del nodo (se calculan en cada nodo para cada proceso y podrían denominarse prioridades nodales)

Los vectores de valoración se aplican para cada requerimiento de un recurso realizado por un proceso, según los criterios establecidos para la determinación de la prioridad que en cada caso y momento fija el nodo en el que se produce el requerimiento.

Cada vector de valoración de cada requerimiento se multiplica escalarmente por el vector de pesos correspondiente a la categoría de carga actual del nodo para obtener la prioridad según cada criterio y la prioridad nodal otorgada a cada requerimiento. Esto se puede ver en la Tabla 13 y Tabla 14.

Tabla 13. Las valoraciones asignadas a los criterios para calcular la prioridad o preferencia nodal que cada nodo concederá a cada requisito de cada proceso según la carga del nodo (*n° proc., %cpu, %mem., %mv*)

Recurso	N°Proc	%CPU	%Mem	%MV
$p_{11}r_{11}$	T(EB;0.0350)	T(EB;0.0250)	T(EB;0.0700)	T(M;-0.0500)
$p_{11}r_{12}$	T(EB;0.0400)	T(EB;0.0350)	T(EB;0.0400)	T(MB;0.0833)
$p_{11}r_{21}$	T(EB;0.0150)	T(EB;0.0200)	T(EB;0.0500)	T(MB;-0.0667)
$p_{11}r_{22}$	T(EB;0.0250)	T(EB;0.0250)	T(EB;0.0700)	T(MB;0.0333)
$p_{11}r_{23}$	T(EB;0.0250)	T(EB;0.0300)	T(EB;0.0800)	T(B;0.0667)
$p_{11}r_{24}$	T(EB;0.0150)	T(EB;0.0250)	T(MB;-0.0767)	T(MB;-0.0667)
$p_{12}r_{11}$	T(EB;0.0200)	T(EB;0.0350)	T(EB;0.0500)	T(M;-0.0500)
$p_{12}r_{12}$	T(EB;0.0100)	T(EB;0.0350)	T(EB;0.0300)	T(B;0.0167)

$p_{12}r_{21}$	T(EB;0.0350)	T(EB;0.0200)	T(EB;0.0300)	T(B;0.0167)
$p_{12}r_{22}$	T(EB;0.0450)	T(EB;0.0300)	T(EB;0.0700)	T(B;0.0167)
$p_{12}r_{31}$	T(EB;0.0100)	T(EB;0.0250)	T(EB;0.0700)	T(B;0.0167)
$p_{12}r_{33}$	T(EB;0.0200)	T(EB;0.0250)	T(EB;0.0700)	T(M;-0.0500)
$p_{13}r_{11}$	T(EB;0.0250)	T(EB;0.0350)	T(EB;0.0700)	T(B;0.0667)
$p_{13}r_{12}$	T(EB;0.0350)	T(EB;0.0400)	T(EB;0.0700)	T(MB;0.0333)
$p_{13}r_{13}$	T(EB;0.0350)	T(EB;0.0300)	T(EB;0.0700)	T(B;0.0667)
$p_{13}r_{21}$	T(EB;0.0350)	T(EB;0.0200)	T(MB;-0.0767)	T(MB;-0.0167)
$p_{13}r_{22}$	T(EB;0.0250)	T(EB;0.0450)	T(EB;0.0800)	T(MB;-0.0167)
$p_{13}r_{31}$	T(EB;0.0250)	T(EB;0.0350)	T(EB;0.0300)	T(B;-0.0333)
$p_{13}r_{32}$	T(EB;0.0300)	T(EB;0.0450)	T(EB;0.0300)	T(B;-0.0333)
$p_{13}r_{33}$	T(EB;0.0300)	T(EB;0.0100)	T(EB;0.0400)	T(B;-0.0333)
$p_{21}r_{12}$	T(EB;0.0200)	T(EB;0.0200)	T(MB;-0.0767)	T(EB;0.0800)
$p_{21}r_{13}$	T(EB;0.0200)	T(EB;0.0800)	T(MB;0.0733)	T(MB;-0.0767)
$p_{21}r_{22}$	T(EB;0.0300)	T(MB;-0.0667)	T(MB;0.0733)	T(MB;-0.0767)
$p_{21}r_{23}$	T(EB;0.0700)	T(MB;-0.0667)	T(MB;0.0133)	T(EB;0.0800)
$p_{21}r_{31}$	T(EB;0.0700)	T(MB;-0.0667)	T(MB;0.0733)	T(EB;0.0600)
$p_{21}r_{33}$	T(EB;0.0700)	T(EB;0.0800)	T(B;-0.0633)	T(EB;0.0800)
$p_{22}r_{21}$	T(EB;0.0500)	T(EB;0.0800)	T(MB;0.0433)	T(EB;0.0800)
$p_{22}r_{22}$	T(EB;0.0500)	T(MB;0.0133)	T(MB;0.0133)	T(EB;0.0800)
$p_{22}r_{31}$	T(EB;0.0500)	T(EB;0.0800)	T(MB;-0.0167)	T(EB;0.0200)
$p_{22}r_{33}$	T(EB;0.0800)	T(EB;0.0800)	T(MB;-0.0767)	T(EB;0.0200)
$p_{23}r_{12}$	T(EB;0.0500)	T(MB;-0.0467)	T(MB;0.0733)	T(EB;0.0300)
$p_{23}r_{24}$	T(EB;0.0600)	T(EB;0.0400)	T(EB;0.0300)	T(EB;0.0700)
$p_{23}r_{31}$	T(EB;0.0300)	T(EB;0.0200)	T(MB;-0.0467)	T(EB;0.0800)
$p_{23}r_{32}$	T(EB;0.0400)	T(EB;0.0800)	T(MB;0.0733)	T(EB;0.0200)
$p_{23}r_{33}$	T(EB;0.0300)	T(MB;-0.0467)	T(MB;0.0733)	T(EB;0.0200)
$p_{24}r_{11}$	T(EB;0.0400)	T(MB;-0.0467)	T(MB;0.0433)	T(MB;-0.0767)
$p_{24}r_{12}$	T(EB;0.0300)	T(MB;-0.0467)	T(MB;0.0433)	T(EB;0.0800)
$p_{24}r_{23}$	T(EB;0.0400)	T(MB;0.0133)	T(MB;-0.0467)	T(EB;0.0800)
$p_{24}r_{24}$	T(EB;0.0500)	T(MB;-0.0067)	T(MB;0.0433)	T(MB;-0.0767)
$p_{25}r_{21}$	T(EB;0.0200)	T(MB;-0.0067)	T(MB;0.0733)	T(MB;-0.0767)
$p_{31}r_{13}$	T(EB;0.0600)	T(B;-0.0633)	T(MB;-0.0467)	T(MB;0.0133)
$p_{31}r_{31}$	T(EB;0.0800)	T(MB;-0.0767)	T(MB;0.0133)	T(MB;-0.0667)
$p_{31}r_{33}$	T(EB;0.0400)	T(MB;0.0433)	T(MB;-0.0267)	T(MB;-0.0667)
$p_{32}r_{11}$	T(EB;0.0600)	T(B;-0.0633)	T(MB;-0.0067)	T(MB;-0.0667)
$p_{32}r_{12}$	T(EB;0.0700)	T(MB;-0.0467)	T(MB;-0.0467)	T(MB;0.0133)
$p_{32}r_{13}$	T(EB;0.0800)	T(B;-0.0633)	T(MB;0.0333)	T(MB;-0.0267)
$p_{32}r_{23}$	T(EB;0.0800)	T(MB;0.0733)	T(MB;0.0333)	T(MB;0.0133)
$p_{33}r_{11}$	T(EB;0.0200)	T(MB;0.0433)	T(MB;0.0133)	T(MB;-0.0067)
$p_{33}r_{12}$	T(MB;-0.0767)	T(B;-0.0333)	T(MB;0.0133)	T(MB;-0.0267)
$p_{33}r_{13}$	T(MB;-0.0767)	T(MB;-0.0167)	T(MB;-0.0267)	T(MB;0.0133)
$p_{33}r_{21}$	T(EB;0.0400)	T(MB;0.0133)	T(MB;-0.0267)	T(MB;-0.0067)
$p_{33}r_{22}$	T(MB;-0.0767)	T(MB;-0.0467)	T(MB;-0.0267)	T(MB;-0.0067)
$p_{33}r_{23}$	T(EB;0.0400)	T(MB;0.0133)	T(MB;0.0133)	T(MB;0.0333)
$p_{33}r_{32}$	T(EB;0.0600)	T(MB;0.0133)	T(MB;-0.0267)	T(MB;-0.0067)
$p_{33}r_{33}$	T(EB;0.0600)	T(B;-0.0333)	T(MB;-0.0267)	T(EB;0.0800)
$p_{34}r_{12}$	T(EB;0.0800)	T(B;-0.0333)	T(EB;0.0600)	T(MB;-0.0667)
$p_{34}r_{13}$	T(EB;0.0200)	T(B;-0.0333)	T(MB;-0.0067)	T(MB;-0.0667)
$p_{34}r_{22}$	T(MB;-0.0767)	T(MB;0.0733)	T(MB;-0.0467)	T(MB;-0.0067)
$p_{34}r_{23}$	T(EB;0.0400)	T(MB;0.0133)	T(MB;-0.0267)	T(MB;-0.0067)
$p_{34}r_{24}$	T(MB;-0.0767)	T(MB;-0.0467)	T(MB;-0.0267)	T(EB;0.0800)
$p_{34}r_{31}$	T(EB;0.0500)	T(MB;0.0133)	T(MB;-0.0267)	T(MB;0.0133)
$p_{34}r_{32}$	T(EB;0.0800)	T(MB;0.0133)	T(MB;0.0133)	T(MB;-0.0667)
$p_{34}r_{33}$	T(EB;0.0400)	T(MB;0.0133)	T(MB;-0.0067)	T(MB;-0.0667)
$p_{35}r_{12}$	T(EB;0.0200)	T(MB;-0.0467)	T(MB;-0.0267)	T(EB;0.0800)
$p_{35}r_{13}$	T(EB;0.0800)	T(B;-0.0633)	T(MB;-0.0267)	T(EB;0.0600)

$p_{35}r_{22}$	T(EB;0.0300)	T(MB;0.0733)	T(MB;0.0133)	T(EB;0.0800)
$p_{35}r_{24}$	T(EB;0.0500)	T(MB;0.0733)	T(MB;0.0133)	T(EB;0.0800)
$p_{35}r_{31}$	T(MB;-0.0767)	T(MB;0.0733)	T(MB;-0.0267)	T(EB;0.0800)
$p_{35}r_{32}$	T(MB;-0.0767)	T(MB;0.0433)	T(MB;-0.0067)	T(MB;-0.0467)
$p_{35}r_{33}$	T(EB;0.0500)	T(MB;0.0433)	T(MB;-0.0467)	T(MB;0.0133)
$p_{36}r_{11}$	T(EB;0.0800)	T(B;-0.0633)	T(MB;-0.0467)	T(MB;-0.0667)
$p_{36}r_{12}$	T(EB;0.0700)	T(B;-0.0633)	T(EB;0.0800)	T(MB;0.0133)
$p_{36}r_{13}$	T(MB;-0.0767)	T(B;-0.0633)	T(MB;-0.0067)	T(MB;-0.0267)
$p_{36}r_{21}$	T(EB;0.0800)	T(B;-0.0633)	T(MB;-0.0667)	T(EB;0.0600)
$p_{36}r_{22}$	T(EB;0.0800)	T(EB;0.0600)	T(EB;0.0200)	T(EB;0.0600)
$p_{36}r_{24}$	T(EB;0.0400)	T(MB;0.0433)	T(MB;0.0133)	T(EB;0.0600)
$p_{36}r_{31}$	T(EB;0.0500)	T(B;-0.0633)	T(MB;0.0133)	T(MB;-0.0267)
$p_{36}r_{32}$	T(MB;-0.0767)	T(MB;0.0733)	T(MB;-0.0467)	T(MB;-0.0267)
$p_{36}r_{33}$	T(EB;0.0200)	T(MB;0.0433)	T(EB;0.0800)	T(MB;-0.0067)
$p_{37}r_{11}$	T(MB;-0.0767)	T(MB;0.0133)	T(MB;-0.0067)	T(MB;-0.0467)
$p_{37}r_{12}$	T(EB;0.0700)	T(B;-0.0633)	T(MB;-0.0067)	T(MB;-0.0267)
$p_{37}r_{21}$	T(MB;-0.0767)	T(MB;0.0433)	T(MB;-0.0067)	T(MB;-0.0467)
$p_{37}r_{32}$	T(EB;0.0800)	T(B;-0.0633)	T(MB;-0.0667)	T(EB;0.0600)
$p_{37}r_{33}$	T(EB;0.0800)	T(MB;-0.0467)	T(MB;-0.0467)	T(MB;-0.0067)

Fuente: Elaboración propia.

Tabla 14. Las valoraciones asignadas a los criterios para calcular la prioridad o preferencia nodal que cada nodo concederá a cada requisito de cada proceso según la carga del nodo (*priorid proc., sobrec. mem., sobrec. proc., sobre E/S*).

Recurso	Priorid Proc	Sobrec Mem	Sobrec Proc	Sobre E/S
$p_{11}r_{11}$	T(EB;0.0800)	T(EB;0.0200)	T(EB;0.0150)	T(EB;0.0200)
$p_{11}r_{12}$	T(EB;0.0300)	T(EB;0.0700)	T(EB;0.0100)	T(EB;0.0200)
$p_{11}r_{21}$	T(MB;-0.0767)	T(EB;0.0200)	T(EB;0.0250)	T(EB;0.0350)
$p_{11}r_{22}$	T(EB;0.0800)	T(EB;0.0300)	T(EB;0.0250)	T(EB;0.0300)
$p_{11}r_{23}$	T(MB;-0.0717)	T(MB;-0.0767)	T(EB;0.0350)	T(EB;0.0300)
$p_{11}r_{24}$	T(EB;0.0600)	T(EB;0.0600)	T(EB;0.0350)	T(EB;0.0200)
$p_{12}r_{11}$	T(MB;-0.0667)	T(MB;-0.0767)	T(EB;0.0400)	T(EB;0.0400)
$p_{12}r_{12}$	T(EB;0.0800)	T(EB;0.0300)	T(EB;0.0400)	T(EB;0.0450)
$p_{12}r_{21}$	T(EB;0.0800)	T(MB;-0.0767)	T(EB;0.0250)	T(EB;0.0100)
$p_{12}r_{22}$	T(EB;0.0800)	T(EB;0.0200)	T(EB;0.0250)	T(EB;0.0200)
$p_{12}r_{31}$	T(EB;0.0300)	T(EB;0.0200)	T(EB;0.0350)	T(EB;0.0400)
$p_{12}r_{33}$	T(EB;0.0300)	T(EB;0.0400)	T(EB;0.0250)	T(EB;0.0400)
$p_{13}r_{11}$	T(EB;0.0600)	T(EB;0.0500)	T(EB;0.0350)	T(EB;0.0400)
$p_{13}r_{12}$	T(MB;-0.0767)	T(EB;0.0200)	T(EB;0.0450)	T(EB;0.0350)
$p_{13}r_{13}$	T(MB;-0.0767)	T(EB;0.0400)	T(EB;0.0400)	T(EB;0.0350)
$p_{13}r_{21}$	T(EB;0.0500)	T(EB;0.0700)	T(EB;0.0100)	T(EB;0.0150)
$p_{13}r_{22}$	T(EB;0.0500)	T(EB;0.0400)	T(EB;0.0450)	T(EB;0.0150)
$p_{13}r_{31}$	T(EB;0.0800)	T(MB;-0.0767)	T(EB;0.0450)	T(EB;0.0250)
$p_{13}r_{32}$	T(EB;0.0400)	T(EB;0.0800)	T(EB;0.0350)	T(EB;0.0400)
$p_{13}r_{33}$	T(MB;-0.0767)	T(EB;0.0800)	T(EB;0.0250)	T(EB;0.0400)
$p_{21}r_{12}$	T(MB;-0.0267)	T(EB;0.0350)	T(EB;0.0125)	T(EB;0.0200)
$p_{21}r_{13}$	T(MB;-0.0267)	T(EB;0.0200)	T(EB;0.0125)	T(EB;0.0050)
$p_{21}r_{22}$	T(MB;-0.0667)	T(EB;0.0300)	T(EB;0.0050)	T(EB;0.0100)
$p_{21}r_{23}$	T(MB;-0.0667)	T(EB;0.0100)	T(EB;0.0225)	T(EB;0.0100)
$p_{21}r_{31}$	T(MB;0.0133)	T(EB;0.0200)	T(EB;0.0225)	T(EB;0.0225)
$p_{21}r_{33}$	T(EB;0.0800)	T(EB;0.0400)	T(EB;0.0150)	T(EB;0.0150)
$p_{22}r_{21}$	T(MB;-0.0467)	T(EB;0.0200)	T(EB;0.0150)	T(EB;0.0225)
$p_{22}r_{22}$	T(MB;0.0133)	T(EB;0.0200)	T(EB;0.0050)	T(EB;0.0025)
$p_{22}r_{31}$	T(EB;0.0800)	T(EB;0.0400)	T(EB;0.0050)	T(EB;0.0225)
$p_{22}r_{33}$	T(MB;-0.0067)	T(EB;0.0450)	T(EB;0.0125)	T(EB;0.0200)
$p_{23}r_{12}$	T(MB;-0.0667)	T(EB;0.0350)	T(EB;0.0125)	T(EB;0.0125)

$p_{23}r_{24}$	T(EB;0.0600)	T(EB;0.0400)	T(EB;0.0225)	T(EB;0.0100)
$p_{23}r_{31}$	T(MB;-0.0267)	T(EB;0.0450)	T(EB;0.0100)	T(EB;0.0150)
$p_{23}r_{32}$	T(EB;0.0800)	T(EB;0.0300)	T(EB;0.0150)	T(EB;0.0150)
$p_{23}r_{33}$	T(MB;0.0133)	T(EB;0.0300)	T(EB;0.0100)	T(EB;0.0050)
$p_{24}r_{11}$	T(MB;-0.0667)	T(EB;0.0350)	T(EB;0.0225)	T(EB;0.0125)
$p_{24}r_{12}$	T(MB;0.0133)	T(EB;0.0350)	T(EB;0.0150)	T(EB;0.0225)
$p_{24}r_{23}$	T(MB;-0.0067)	T(EB;0.0350)	T(EB;0.0150)	T(EB;0.0075)
$p_{24}r_{24}$	T(EB;0.0600)	T(EB;0.0200)	T(EB;0.0200)	T(EB;0.0175)
$p_{25}r_{21}$	T(EB;0.0800)	T(EB;0.0250)	T(EB;0.0150)	T(EB;0.0200)
$p_{31}r_{13}$	T(EB;0.0700)	T(EB;0.0100)	T(EB;0.0225)	T(EB;0.0400)
$p_{31}r_{31}$	T(EB;0.0700)	T(EB;0.0075)	T(EB;0.0225)	T(EB;0.0350)
$p_{31}r_{33}$	T(MB;-0.0767)	T(EB;0.0225)	T(EB;0.0175)	T(EB;0.0350)
$p_{32}r_{11}$	T(MB;-0.0767)	T(EB;0.0175)	T(EB;0.0075)	T(EB;0.0350)
$p_{32}r_{12}$	T(EB;0.0800)	T(EB;0.0250)	T(EB;0.0225)	T(EB;0.0350)
$p_{32}r_{13}$	T(MB;-0.0767)	T(EB;0.0075)	T(EB;0.0125)	T(EB;0.0450)
$p_{32}r_{23}$	T(EB;0.0600)	T(EB;0.0200)	T(EB;0.0100)	T(EB;0.0450)
$p_{33}r_{11}$	T(EB;0.0600)	T(EB;0.0225)	T(EB;0.0250)	T(EB;0.0150)
$p_{33}r_{12}$	T(EB;0.0300)	T(EB;0.0125)	T(EB;0.0175)	T(EB;0.0150)
$p_{33}r_{13}$	T(EB;0.0300)	T(EB;0.0100)	T(EB;0.0125)	T(EB;0.0400)
$p_{33}r_{21}$	T(EB;0.0800)	T(EB;0.0100)	T(EB;0.0200)	T(EB;0.0400)
$p_{33}r_{22}$	T(EB;0.0700)	T(EB;0.0100)	T(EB;0.0225)	T(EB;0.0350)
$p_{33}r_{23}$	T(EB;0.0600)	T(EB;0.0100)	T(EB;0.0175)	T(EB;0.0400)
$p_{33}r_{32}$	T(EB;0.0400)	T(EB;0.0125)	T(EB;0.0200)	T(EB;0.0400)
$p_{33}r_{33}$	T(EB;0.0600)	T(EB;0.0200)	T(EB;0.0200)	T(EB;0.0450)
$p_{34}r_{12}$	T(EB;0.0700)	T(EB;0.0075)	T(EB;0.0200)	T(EB;0.0350)
$p_{34}r_{13}$	T(EB;0.0800)	T(EB;0.0150)	T(EB;0.0225)	T(EB;0.0350)
$p_{34}r_{22}$	T(MB;-0.0767)	T(EB;0.0200)	T(EB;0.0125)	T(EB;0.0300)
$p_{34}r_{23}$	T(MB;-0.0767)	T(EB;0.0125)	T(EB;0.0225)	T(EB;0.0300)
$p_{34}r_{24}$	T(EB;0.0700)	T(EB;0.0125)	T(EB;0.0225)	T(EB;0.0350)
$p_{34}r_{31}$	T(EB;0.0700)	T(EB;0.0225)	T(EB;0.0150)	T(EB;0.0350)
$p_{34}r_{32}$	T(EB;0.0600)	T(EB;0.0225)	T(EB;0.0075)	T(EB;0.0450)
$p_{34}r_{33}$	T(MB;-0.0767)	T(EB;0.0225)	T(EB;0.0100)	T(EB;0.0150)
$p_{35}r_{12}$	T(MB;-0.0767)	T(EB;0.0125)	T(EB;0.0175)	T(EB;0.0450)
$p_{35}r_{13}$	T(MB;-0.0767)	T(EB;0.0200)	T(EB;0.0175)	T(EB;0.0200)
$p_{35}r_{22}$	T(EB;0.0800)	T(EB;0.0150)	T(EB;0.0075)	T(EB;0.0300)
$p_{35}r_{24}$	T(EB;0.0600)	T(EB;0.0225)	T(EB;0.0100)	T(EB;0.0350)
$p_{35}r_{31}$	T(EB;0.0800)	T(EB;0.0075)	T(EB;0.0100)	T(EB;0.0400)
$p_{35}r_{32}$	T(EB;0.0400)	T(EB;0.0225)	T(EB;0.0100)	T(EB;0.0350)
$p_{35}r_{33}$	T(EB;0.0800)	T(EB;0.0125)	T(EB;0.0225)	T(EB;0.0350)
$p_{36}r_{11}$	T(EB;0.0800)	T(EB;0.0225)	T(EB;0.0225)	T(EB;0.0300)
$p_{36}r_{12}$	T(EB;0.0800)	T(EB;0.0225)	T(EB;0.0100)	T(EB;0.0350)
$p_{36}r_{13}$	T(EB;0.0800)	T(EB;0.0175)	T(EB;0.0225)	T(EB;0.0350)
$p_{36}r_{21}$	T(EB;0.0700)	T(EB;0.0175)	T(EB;0.0225)	T(EB;0.0450)
$p_{36}r_{22}$	T(EB;0.0800)	T(EB;0.0175)	T(EB;0.0150)	T(EB;0.0450)
$p_{36}r_{24}$	T(EB;0.0800)	T(EB;0.0125)	T(EB;0.0200)	T(EB;0.0450)
$p_{36}r_{31}$	T(EB;0.0800)	T(EB;0.0200)	T(EB;0.0200)	T(EB;0.0350)
$p_{36}r_{32}$	T(EB;0.0800)	T(EB;0.0125)	T(EB;0.0150)	T(EB;0.0350)
$p_{36}r_{33}$	T(EB;0.0800)	T(EB;0.0225)	T(EB;0.0100)	T(EB;0.0250)
$p_{37}r_{11}$	T(MB;-0.0767)	T(EB;0.0175)	T(EB;0.0225)	T(EB;0.0400)
$p_{37}r_{12}$	T(EB;0.0500)	T(EB;0.0200)	T(EB;0.0200)	T(EB;0.0300)
$p_{37}r_{21}$	T(EB;0.0600)	T(EB;0.0225)	T(EB;0.0125)	T(EB;0.0200)
$p_{37}r_{32}$	T(EB;0.0800)	T(EB;0.0175)	T(EB;0.0100)	T(EB;0.0300)
$p_{37}r_{33}$	T(EB;0.0800)	T(EB;0.0150)	T(EB;0.0225)	T(EB;0.0150)

Fuente: Elaboración propia.

En resumen, la prioridad nodal (que debe calcularse en el nodo en que se produce la

solicitud) de un proceso para acceder a un recurso determinado (que puede estar en cualquier nodo) se calcula mediante el producto escalar de los vectores mencionados anteriormente.

Cálculo de las prioridades o preferencias de los procesos para acceder a los recursos compartidos disponibles (calculados en el gestor centralizado de recursos) y determinación del orden de asignación de los recursos y del proceso de asignación de cada uno de ellos

La Tabla 15 y Tabla 16 y Tabla 17 se utiliza para calcular las prioridades finales, en el que se colocan las prioridades o preferencias nodales calculadas en la etapa anterior; en esta tabla cada fila contiene la información de las prioridades nodales de los diferentes procesos para acceder a un determinado recurso.

Tabla 15. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla (p_{11}, p_{12}, p_{13}, p_{21}, p_{22})

Recursos	Prioridades Nodales de los Procesos				
	p_{11}	p_{12}	p_{13}	p_{21}	p_{22}
r_{11}	TPN(A;0.0483)	TPN(MA;-0.0083)	TPN(A;0.0483)	-	-
r_{12}	TPN(M;-0.0050)	TPN(A;-0.0467)	TPN(M;0.035)	TPN(M;-0.0825)	-
r_{13}	-	-	TPN(A;0.0733)	TPN(A;-0.0592)	-
r_{21}	TPN(B;0.0217)	TPN(A;-0.0267)	TPN(M;-0.06)	-	TPN(A;-0.0692)
r_{22}	TPN(M;-0.015)	TPN(A;-0.0267)	TPN(M;-0.05)	TPN(A;-0.0617)	TPN(A;0.0308)
r_{23}	TPN(MA;-0.0483)	-	-	TPN(M;0.0725)	-
r_{24}	TPN(B;0.0717)	-	-	-	-
r_{31}	-	TPN(M;0.080)	TPN(A;-0.0367)	TPN(A;0.0483)	TPN(M;-0.0525)
r_{32}	-	-	TPN(A;-0.0667)	-	-
r_{33}	-	TPN(A;0.0333)	TPN(A;-0.0517)	TPN(A;-0.0167)	TPN(M;0.0075)

Fuente: Elaboración propia.

Tabla 16. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla (p_{23}, p_{24}, p_{25}, p_{31}, p_{32})

Recursos	Prioridades Nodales de los Procesos				
	p_{23}	p_{24}	p_{25}	p_{31}	p_{32}
r_{11}	-	TPN(A;-0.0367)	-	-	TPN(A;0.0733)
r_{12}	TPN(A;-0.0667)	TPN(A;0.0258)	-	-	TPN(A;-0.0142)
r_{13}	-	-	-	TPN(MA;-0.0608)	TPN(MA;0.0117)
r_{21}	-	-	TPN(A;-0.0167)	-	-
r_{22}	-	-	-	-	-
r_{23}	-	TPN(A;-0.0292)	-	-	TPN(MA;0.0017)
r_{24}	TPN(B;-0.0008)	TPN(A;-0.0392)	-	-	-
r_{31}	TPN(M;-0.0400)	-	-	TPN(A;-0.0817)	-
r_{32}	TPN(M;0.0200)	-	-		-
r_{33}	TPN(A;-0.0317)	-	-	TPN(A;-0.0117)	-

Fuente: Elaboración propia.

Tabla 17. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla (p_{33}, p_{34}, p_{35}, p_{36}, p_{37})

Recursos	Prioridades Nodales de los Procesos				
	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}
r_{11}	TPN(A;0.0258)	-		TPN(A;0.0583)	TPN(A;0.0533)
r_{12}	TPN(MA;-0.0483)	TPN(A;0.0058)	TPN(M;0.025)	TPN(A;0.0808)	TPN(MA;-0.0733)
r_{13}	TPN(A;-0.0142)	TPN(A;0.0658)	TPN(A;0.0308)	TPN(MA;-0.0183)	-
r_{21}	TPN(A;0.0033)	-	-	TPN(A;-0.0017)	TPN(A;0.0283)
r_{22}	TPN(A;-0.0192)	TPN(MA;-0.0708)	TPN(A;-0.0042)	TPN(B;0.0442)	-
r_{23}	TPN(A;0.0608)	TPN(A;0.0083)	-	-	-
r_{24}	-	TPN(M;0.0700)	TPN(A;0.0108)	TPN(A;-0.0192)	-
r_{31}	-	TPN(A;0.0258)	TPN(A;0.0208)	TPN(MA;-0.0383)	-
r_{32}	TPN(A;-0.0142)	TPN(A;0.0083)	TPN(A;0.0208)	TPN(A;0.0658)	TPN(A;-0.0192)
r_{33}	TPN(A;0.0583)	TPN(A;-0.0492)	TPN(A;0.0433)	TPN(A;-0.0592)	TPN(A;-0.0542)

Fuente: Elaboración propia.

A continuación, se debe calcular el vector de peso final que se utilizará en el proceso de agregación final para determinar el orden o la prioridad de acceso a los recursos. Además, los pesos recientemente obtenidos deberán normalizarse dividiendo cada uno de ellos por la suma de todos ellos.

Tabla 18. Vector de pesos (p_{11}, p_{12}, p_{13}, p_{21}, p_{22})

	p_{11}	p_{12}	p_{13}	p_{21}	p_{22}
wp_{ij}	0.2000	0.1333	0.2000	0.1333	0.1333
nwp_{ij}	0.0968	0.0645	0.0968	0.0645	0.0645

Fuente: Elaboración propia.

Tabla 19. Vector de pesos (p_{23}, p_{24}, p_{25}, p_{31}, p_{32})

	p_{23}	p_{24}	p_{25}	p_{31}	p_{32}
wp_{ij}	0.2000	0.0667	0.2000	0.1333	0.0667
nwp_{ij}	0.0968	0.0323	0.0968	0.0645	0.0323

Fuente: Elaboración propia.

Tabla 20. Vector de pesos (p_{33}, p_{34}, p_{35}, p_{36}, p_{37})

	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}	Total
wp_{ij}	0.0667	0.2000	0.0667	0.0667	0.2000	2.0667
nwp_{ij}	0.0323	0.0968	0.0323	0.0323	0.0968	1

Fuente: Elaboración propia.

Las prioridades nodales indicadas en la Tabla 15 y Tabla 16 y Tabla 17 tomadas fila por fila, es decir, para cada recurso, se multiplicarán por el vector de peso normalizado (nwp_{ij}) de la Tabla 18 Tabla 19 y Tabla 20. Esto se puede ver en la Tabla 21, Tabla 22 y Tabla 23.

Tabla 21. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla, multiplicado por su vector de peso (p_{11}, p_{12}, p_{13}, p_{21}, p_{22})

Recursos	Prioridades Nodales de los Procesos				
	p_{11}	p_{12}	p_{13}	p_{21}	p_{22}
r_{11}	TPN(EB;0.069)	TPN(EB;0.053)	TPN(EB;0.069)	-	-
r_{12}	TPN(EB;0.048)	TPN(EB;0.040)	TPN(EB;0.052)	TPN(EB;0.027)	-
r_{13}	-	-	TPN(EB;0.072)	TPN(EB;0.039)	-
r_{21}	TPN(EB;0.034)	TPN(EB;0.041)	TPN(EB;0.043)	-	TPN(EB;0.039)
r_{22}	TPN(EB;0.047)	TPN(EB;0.041)	TPN(EB;0.044)	TPN(EB;0.039)	TPN(EB;0.045)
r_{23}	TPN(EB;0.076)	-	-	TPN(EB;0.037)	-
r_{24}	TPN(EB;0.039)	-	-	-	-
r_{31}	-	TPN(EB;0.037)	TPN(EB;0.061)	TPN(EB;0.046)	TPN(EB;0.029)
r_{32}	-	-	TPN(EB;0.058)	-	-
r_{33}	-	TPN(EB;0.045)	TPN(EB;0.06)	TPN(EB;0.042)	TPN(EB;0.033)

Fuente: Elaboración propia.

Tabla 22. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla, multiplicado por su vector de peso (p_{23}, p_{24}, p_{25}, p_{31}, p_{32})

Recursos	Prioridades Nodales de los Procesos				
	p_{23}	p_{24}	p_{25}	p_{31}	p_{32}
r_{11}	-	TPN(EB;0.020)	-	-	TPN(EB;0.024)
r_{12}	TPN(EB;0.058)	TPN(EB;0.022)	-	-	TPN(EB;0.021)
r_{13}	-	-	-	TPN(EB;0.05)	TPN(EB;0.027)
r_{21}	-	-	TPN(EB;0.063)	-	-
r_{22}	-	-	-	-	-
r_{23}	-	TPN(EB;0.021)	-	-	TPN(EB;0.027)
r_{24}	TPN(EB;0.032)	TPN(EB;0.020)	-	-	-
r_{31}	TPN(EB;0.045)	-	-	TPN(EB;0.038)	-
r_{32}	TPN(EB;0.050)	-	-	-	-
r_{33}	TPN(EB;0.061)	-	-	TPN(EB;0.042)	-

Fuente: Elaboración propia.

Tabla 23. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla, multiplicado por su vector de peso (p_{33}, p_{34}, p_{35}, p_{36}, p_{37})

Recursos	Prioridades Nodales de los Procesos				
	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}
r_{11}	TPN(EB;0.022)	-	-	TPN(EB;0.023)	TPN(EB;0.070)
r_{12}	TPN(EB;0.025)	TPN(EB;0.065)	TPN(EB;0.017)	TPN(EB;0.024)	TPN(EB;0.074)
r_{13}	TPN(EB;0.021)	TPN(EB;0.071)	TPN(EB;0.023)	TPN(EB;0.026)	-
r_{21}	TPN(EB;0.022)	-	-	TPN(EB;0.021)	TPN(EB;0.067)
r_{22}	TPN(EB;0.021)	TPN(EB;0.074)	TPN(EB;0.021)	TPN(EB;0.012)	-
r_{23}	TPN(EB;0.023)	TPN(EB;0.065)	-	-	-
r_{24}	-	TPN(EB;0.055)	TPN(EB;0.022)	TPN(EB;0.021)	-
r_{31}	-	TPN(EB;0.067)	TPN(EB;0.022)	TPN(EB;0.026)	-
r_{32}	TPN(EB;0.021)	TPN(EB;0.065)	TPN(EB;0.022)	TPN(EB;0.024)	TPN(EB;0.063)
r_{33}	TPN(EB;0.023)	TPN(EB;0.06)	TPN(EB;0.023)	TPN(EB;0.02)	TPN(EB;0.059)

Fuente: Elaboración propia.

El siguiente paso es normalizar la Tabla 21, Tabla 22 y Tabla 23 entre los valores extremos. Para ello, se debe restar el valor numérico de la 2-tupla por el valor mínimo de todas ellas y

dividirlo por el rango, que es la diferencia entre el valor máximo y el valor mínimo de las tablas ya mencionadas. Esto se puede ver en la Tabla 24.

Tabla 24. Valoraciones para normalizar la ***FASDL***

Etiqueta	Valor
Valor Máximo	0.0760
Valor Mínimo	0.0122
Rango	0.0638

Fuente: Elaboración propia.

El resultado de esta normalización se puede ver en la Tabla 25, Tabla 26 y Tabla 27.

Tabla 25. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla normalizada (p_{11}, p_{12}, p_{13}, p_{21}, p_{22})

Rec.	Prioridades Nodales de los Procesos				
	p_{11}	p_{12}	p_{13}	p_{21}	p_{22}
r_{11}	TPGFN(MA;0.06)	TPGFN(A;-0.023)	TPGFN(MA;0.060)	-	TPGFN(MA;0.06)
r_{12}	TPGFN(M;0.060)	TPGFN(M;-0.064)	TPGFN(A;-0.046)	TPGFN(MB;0.065)	TPGFN(M;0.060)
r_{13}	-	-	TPGFN(EA;-0.068)	TPGFN(M;-0.076)	-
r_{21}	TPGFN(B;0.014)	TPGFN(M;-0.044)	TPGFN(M;-0.023)	-	TPGFN(B;0.014)
r_{22}	TPGFN(M;0.045)	TPGFN(M;-0.044)	TPGFN(M;-0.008)	TPGFN(M;-0.079)	TPGFN(M;0.045)
r_{23}	TPGFN(EA;0.00)	-	-	TPGFN(B;0.055)	TPGFN(EA;0.00)
r_{24}	TPGFN(M;-0.076)	-	-	-	TPGFN(M;-0.076)
r_{31}	-	TPGFN(B;0.062)	TPGFN(MA;-0.068)	TPGFN(M;0.032)	-
r_{32}	-	-	TPGFN(A;0.053)	-	-
r_{33}	-	TPGFN(M;0.017)	TPGFN(A;0.075)	TPGFN(M;-0.034)	-

Fuente: Elaboración propia.

Tabla 26. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla normalizada (p_{23}, p_{24}, p_{25}, p_{31}, p_{32})

Rec.	Prioridades Nodales de los Procesos				
	p_{23}	p_{24}	p_{25}	p_{31}	p_{32}
r_{11}	-	TPGFN(MB;-0.039)	-	-	TPGFN(MB;0.017)
r_{12}	TPGFN(MA;-0.052)	TPGFN(MB;-0.007)	-	-	TPGFN(MB;-0.028)
r_{13}	-	-	-	TPGFN(A;-0.076)	TPGFN(MB;0.070)
r_{21}	-	-	TPGFN(MA;-0.038)	-	-
r_{22}	-	-	-	-	-
r_{23}	-	TPGFN(MB;-0.035)	-	-	TPGFN(MB;0.065)
r_{24}	TPGFN(B;0.007)	TPGFN(MB;-0.040)	-	-	-
r_{31}	TPGFN(M;0.051)	-	-	TPGFN(B;0.067)	-
r_{32}	TPGFN(A;-0.017)	-	-	-	-
r_{33}	TPGFN(MA;-0,061)	-	-	TPGFN(M;-0.028)	-

Fuente: Elaboración propia.

El mayor de estos productos realizados para los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

Tabla 27. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla normalizada (p_{33}, p_{34}, p_{35}, p_{36}, p_{37})

Rec.	Prioridades Nodales de los Procesos				
	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}
r_{11}	TPGFN(MB;-0.007)	-	-	TPGFN(MB;0.009)	TPGFN(MA;0.068)
r_{12}	TPGFN(MB;0.039)	TPGFN(MA;-0.004)	TPGFN(EB;0.075)	TPGFN(MB;0.020)	TPGFN(EA;-0.038)
r_{13}	TPGFN(MB;-0.028)	TPGFN(EA;-0.080)	TPGFN(MB;-0.005)	TPGFN(MB;0.055)	-
r_{21}	TPGFN(MB;-0.019)	-	-	TPGFN(MB;-0.021)	TPGFN(MA;0.030)
r_{22}	TPGFN(MB;-0.030)	TPGFN(EA;-0.034)	TPGFN(MB;-0.023)	TPGFN(EB;0.000)	-
r_{23}	TPGFN(MB;0.010)	TPGFN(MA;0.000)	-	-	-
r_{24}	-	TPGFN(A;0.007)	TPGFN(MB;-0.015)	TPGFN(MB;-0.030)	-
r_{31}	-	TPGFN(MA;0.026)	TPGFN(MB;-0.010)	TPGFN(MB;0.044)	-
r_{32}	TPGFN(MB;-0.028)	TPGFN(MA;0.000)	TPGFN(MB;-0.010)	TPGFN(MB;0.013)	TPGFN(MA;-0.042)
r_{33}	TPGFN(MB;0.009)	TPGFN(A;0.079)	TPGFN(MB;0.0010)	TPGFN(MB;-0.050)	TPGFN(A;0.072)

Fuente: Elaboración propia.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse. Esto constituye la Función de Asignación del Sistema Distribuido Lingüística (*FASDL*):

$FASDL(r_{ij}) = \Sigma\ TPGFN_{ijkl} = r_{ij}$ prioridad de asignación de recursos.

Al calcular la ***FASDL*** para todos los recursos, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 28.

Tabla 28. Valoraciones para normalizar la ***FASDL*** en la primera iteración.

Etiqueta	Valor
Valor Máximo	5.2364
Valor Mínimo	1.8255
Rango	3.4109

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los $TPGFN_{ijkl}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (*FASDLN*):

$FASDLN(r_{ij}) = \Sigma\ (TPGFN_{ijkl}$ / (Máximo ($TPGFN_{ijkl}$) - Mínimo ($TPGFN_{ijkl}$)) = r_{ij} prioridad de asignación de recursos normalizada entre valores extremos, ver Tabla 29.

Tabla 29. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la primera iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{11}	T(A;-0.0355)	p_{37}
r_{12}	T(EA;-0.0326)	p_{37}
r_{13}	T(M;0.0274)	p_{13}
r_{21}	T(M;0.0337)	p_{37}
r_{22}	T(M;0.0423)	p_{34}
r_{23}	T(B;-0.0591)	p_{11}
r_{24}	T(EB;0.0000)	p_{34}
r_{31}	T(A;-0.0028)	p_{34}
r_{32}	T(M;-0.0334)	p_{34}
r_{33}	T(EA;0.0000)	p_{23}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 30.

Tabla 30. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la primera iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.0000)	p_{23}
r_{12}	T(EA;-0.0326)	p_{37}
r_{31}	T(A;-0.0028)	p_{34}
r_{11}	T(A;-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(B;-0.0591)	p_{11}
r_{24}	T(EB;0.0000)	p_{34}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Dado que el sistema se autorregula liberando los recursos ya asignados a los procesos en el paso anterior, y dado que hay solicitudes de recursos de los procesos que aún no han sido

satisfechas, los cálculos de la ***FASDL***, ***FASDLN*** y ***FASDLNO*** de las Tabla 28, Tabla 29 y Tabla 30 se repiten con sus respectivos valores, omitiendo los procesos ya completados.

En la Fig. 5 se pueden observar los pasos para obtener la prioridad final, en cada ronda de asignaciones se deberán repetir los pasos 6, 7 y 8 dado que a medida que las solicitudes de recursos se satisfacen, se liberaran los recursos para que otros procesos los utilicen, al finalizar todas las rondas ordenadas, se obtendrá el listado final que se corresponde con la ***FASDLNC***.

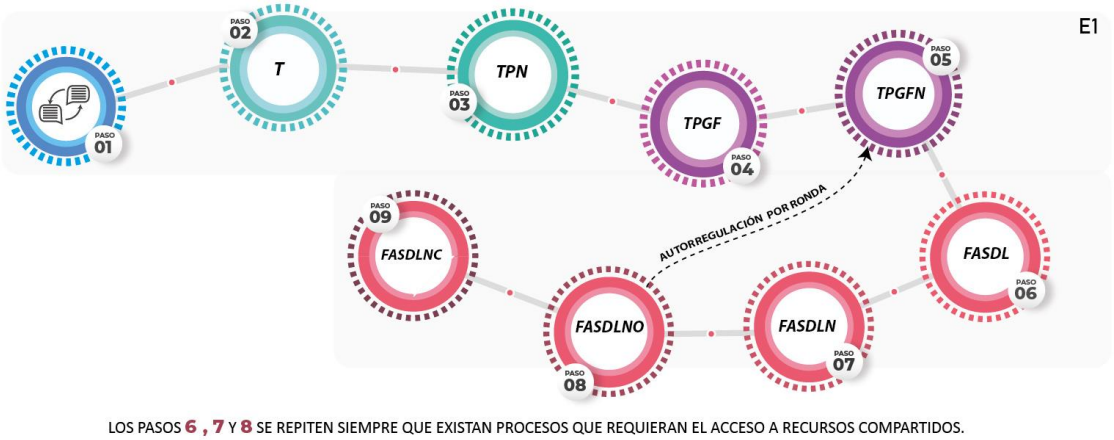


Fig. 5. Proceso de autorregulación y cálculo de la ***FASDLNC.***

Fuente: Elaboración propia.

El resultado de las sucesivas iteraciones se muestra en las siguientes tablas.

Segunda iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la segunda ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 31.

Tabla 31. Valoraciones para normalizar la ***FASDL*** en la segunda iteración

Etiqueta	Valor
Valor Máximo	4.8503
Valor Mínimo	1.2514
Rango	3.5989

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso. El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver tabla Tabla 32.

Tabla 32. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la segunda iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{11}	T(M;0.0813)	p_{11}
r_{12}	T(MA;0.0758)	p_{34}
r_{13}	T(M;-0.0347)	p_{34}
r_{21}	T(M;-0.0076)	p_{25}
r_{22}	T(M;-0.0298)	p_{11}
r_{23}	T(MB;0.0173)	p_{34}
r_{24}	T(EB;0.0000)	p_{11}
r_{31}	T(A;-0.0392)	p_{13}
r_{32}	T(M;-0.0676)	p_{37}
r_{33}	T(EA;0.0000)	p_{34}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 33.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Tabla 33. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la segunda iteración ***(FASDLNO)***

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.0000)	p_{34}
r_{12}	T(MA;0.0758)	p_{34}
r_{31}	T(A;-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(MB;0.0173)	p_{34}
r_{24}	T(EB;0.0000)	p_{11}

Fuente: Elaboración propia.

Tercera iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la tercera ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la

Tabla **34.**

Tabla 34. Valoraciones para normalizar la ***FASDL*** en la tercera iteración

Etiqueta	Valor
Valor Máximo	4.1598
Valor Mínimo	0.8147
Rango	3.3451

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver tabla Tabla 35.

Tabla 35. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la tercera iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{11}	T(M;-0.0133)	p_{13}
r_{12}	T(MA;0.0381)	p_{23}
r_{13}	T(B;0.0159)	p_{31}
r_{21}	T(M;-0.0789)	p_{13}
r_{22}	T(M;-0.0197)	p_{22}
r_{23}	T(EB;0.0668)	p_{21}
r_{24}	T(EB;0.0000)	p_{23}
r_{31}	T(M;0.0810)	p_{21}
r_{32}	T(B;0.0227)	p_{13}
r_{33}	T(EA;0.0000)	p_{13}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 36.

Tabla 36. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la tercera iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.0000)	p_{13}
r_{12}	T(MA;0.0381)	p_{23}
r_{31}	T(M;0.0810)	p_{21}
r_{11}	T(M;-0.0133)	p_{13}
r_{22}	T(M;-0.0197)	p_{22}
r_{21}	T(M;-0.0789)	p_{13}
r_{32}	T(B;0.0227)	p_{13}
r_{13}	T(B;0.0159)	p_{31}
r_{23}	T(EB;0.0668)	p_{21}
r_{24}	T(EB;0.0000)	p_{23}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez

que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Cuarta iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso. La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la cuarta ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 37.

Tabla 37. Valoraciones para normalizar el ***FASDL*** en la cuarta iteración

Etiqueta	Valor
Valor Máximo	4.0308
Valor Mínimo	0.5616
Rango	3.4692

Fuente: *Elaboración propia.*

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso. El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver tabla Tabla 38.

Tabla 38. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la cuarta iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{11}	T(B;0.0082)	p_{12}
r_{12}	T(MA;0.0255)	p_{13}
r_{13}	T(B;-0.0337)	p_{21}
r_{21}	T(M;-0.0721)	p_{12}
r_{22}	T(M;-0.0178)	p_{13}
r_{23}	T(EB;0.0489)	p_{32}
r_{24}	T(EB;0.0000)	p_{35}
r_{31}	T(A;-0.0739)	p_{23}
r_{32}	T(B;-0.0762)	p_{23}

r_{33}	T(EA;0.0000)	p_{37}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 39.

Tabla 39. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la cuarta iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.0000)	p_{37}
r_{12}	T(MA;0.0255)	p_{13}
r_{31}	T(A;-0.0739)	p_{23}
r_{22}	T(M;-0.0178)	p_{13}
r_{21}	T(M;-0.0721)	p_{12}
r_{11}	T(B;0.0082)	p_{12}
r_{13}	T(B;-0.0337)	p_{21}
r_{32}	T(B;-0.0762)	p_{23}
r_{23}	T(EB;0.0489)	p_{32}
r_{24}	T(EB;0.0000)	p_{35}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Quinta iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la quinta ronda, se obtendrá un vector de

2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 40.

Tabla 40. Valoraciones para normalizar la ***FASDL*** en la quinta iteración

Etiqueta	Valor
Valor Máximo	3.9955
Valor Mínimo	0.4695
Rango	3.5260

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver Tabla 41.

Tabla 41. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la quinta iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{11}	T(MB;0.0273)	p_{32}
r_{12}	T(MA;0.0431)	p_{11}
r_{13}	T(B;-0.0824)	p_{32}
r_{21}	T(B;0.0674)	p_{22}
r_{22}	T(M;-0.0467)	p_{12}
r_{23}	T(EB;0.0230)	p_{33}
r_{24}	T(EB;0.0000)	p_{36}
r_{31}	T(A;-0.0777)	p_{31}
r_{32}	T(MB;-0.0591)	p_{36}
r_{33}	T(EA;0.0000)	p_{12}

Fuente: Elaboración propia.

Tabla 42. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la quinta iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.0000)	p_{12}
r_{12}	T(MA;0.0431)	p_{11}
r_{31}	T(A;-0.0777)	p_{31}
r_{22}	T(M;-0.0467)	p_{12}
r_{21}	T(B;0.0674)	p_{22}
r_{13}	T(B;-0.0824)	p_{32}
r_{11}	T(MB;0.0273)	p_{32}
r_{32}	T(MB;-0.0591)	p_{36}
r_{23}	T(EB;0.0230)	p_{33}
r_{24}	T(EB;0.0000)	p_{36}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 42.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Sexta iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la sexta ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 43.

Tabla 43. Valoraciones para normalizar la ***FASDL*** en la sexta iteración

Etiqueta	Valor
Valor Máximo	3.6488
Valor Mínimo	0.2681
Rango	3.3807

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso

indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver Tabla 44.

Tabla 44. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la sexta iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{11}	T(MB;0.0443)	p_{32}
r_{12}	T(MA;-0.0133)	p_{11}
r_{13}	T(MB;0.0815)	p_{32}
r_{21}	T(B;-0.0106)	p_{22}
r_{22}	T(B;0.0275)	p_{12}
r_{23}	T(EB;0.0032)	p_{33}
r_{24}	T(EB;0.0000)	p_{36}
r_{31}	T(M;0.0638)	p_{31}
r_{32}	T(MB;-0.0604)	p_{36}
r_{33}	T(EA;0.0000)	p_{12}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 45.

Tabla 45. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la sexta iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.0000)	p_{12}
r_{12}	T(MA;-0.0133)	p_{11}
r_{31}	T(M;0.0638)	p_{31}
r_{22}	T(B;0.0275)	p_{12}
r_{21}	T(B;-0.0106)	p_{22}
r_{13}	T(MB;0.0815)	p_{32}
r_{11}	T(MB;0.0443)	p_{32}
r_{32}	T(MB;-0.0604)	p_{36}
r_{23}	T(EB;0.0032)	p_{33}
r_{24}	T(EB;0.0000)	p_{36}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Séptima iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la séptima ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 46.

Tabla 46. Valoraciones para normalizar la ***FASDL*** en la séptima iteración

Etiqueta	Valor
Valor Máximo	2.6775
Valor Mínimo	0.0000
Rango	2.6775

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver Tabla 47.

Tabla 47. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la séptima iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{11}	T(MB;0.0631)	p_{33}
r_{12}	T(MA;-0.0348)	p_{21}
r_{13}	T(MB;0.0742)	p_{35}
r_{21}	T(MB;0.0682)	p_{33}
r_{22}	T(MB;0.0580)	p_{35}
r_{31}	T(M;0.0040)	p_{22}
r_{32}	T(MB;-0.0553)	p_{33}
r_{33}	T(EA;0.0000)	p_{21}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 48.

Tabla 48. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la séptima iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.000)	p_{21}
r_{12}	T(MA;-0.0348)	p_{21}
r_{31}	T(M;0.0040)	p_{22}
r_{13}	T(MB;0.0742)	p_{35}
r_{21}	T(MB;0.0682)	p_{33}
r_{11}	T(MB;0.0631)	p_{33}
r_{22}	T(MB;0.0580)	p_{35}
r_{32}	T(MB;-0.0553)	p_{33}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Octava iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la octava ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de

asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 49.

Tabla 49. Valoraciones para normalizar la ***FASDL*** en la octava iteración

Etiqueta	Valor
Valor Máximo	2.4275
Valor Mínimo	0.0000
Rango	2.4275

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver tabla Tabla 50.

Tabla 50. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la octava iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{11}	T(MB;-0.0035)	p_{24}
r_{12}	T(EA;-0.0210)	p_{33}
r_{13}	T(MB;0.0110)	p_{33}
r_{21}	T(MB;0.0191)	p_{36}
r_{22}	T(MB;0.0078)	p_{33}
r_{31}	T(M;-0.0299)	p_{36}
r_{33}	T(EA;0.0000)	p_{22}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 51.

Tabla 51. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la octava iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.0000)	p_{22}
r_{12}	T(EA;-0.0210)	p_{33}
r_{31}	T(M;-0.0299)	p_{36}
r_{21}	T(MB;0.0191)	p_{36}
r_{13}	T(MB;0.0110)	p_{33}
r_{22}	T(MB;0.0078)	p_{33}
r_{11}	T(MB;-0.0035)	p_{24}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Novena iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la novena ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 52.

Tabla 52. Valoraciones para normalizar la ***FASDL*** en la novena iteración

Etiqueta	Valor
Valor Máximo	2.9932
Valor Mínimo	0.0000
Rango	2.9932

Fuente: *Elaboración propia.*

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver tabla Tabla 53.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 54.

Tabla 53. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la novena iteración (***FASDLN***.

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{12}	T(EA;0.0000)	p_{36}
r_{22}	T(EB;0.0000)	p_{36}
r_{31}	T(B;-0.0534)	p_{35}
r_{33}	T(MA;-0.0117)	p_{33}

Fuente: Elaboración propia.

Tabla 54. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la novena iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{12}	T(EA;0.0000)	p_{36}
r_{33}	T(MA;-0.0117)	p_{33}
r_{31}	T(B;-0.0534)	p_{35}
r_{22}	T(EB;0.0000)	p_{36}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Décima iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la décima ronda, se obtendrá un vector de

2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 55.

Tabla 55. Valoraciones para normalizar la ***FASDL*** en la décima iteración

Etiqueta	Valor
Valor Máximo	1.5946
Valor Mínimo	0.0000
Rango	1.5946

Fuente: *Elaboración propia.*

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver Tabla 56.

Tabla 56. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la décima iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{12}	T(EA;0.0000)	p_{24}
r_{33}	T(MA;0.0734)	p_{35}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 57.

Tabla 57. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la décima iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{12}	T(EA;0.0000)	p_{24}
r_{33}	T(MA;0.0734)	p_{35}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de

asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Onceava iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la undécima ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 58.

Tabla 58. Valoraciones para normalizar la ***FASDL*** en la undécima iteración

Etiqueta	Valor
Valor Máximo	1.0000
Valor Mínimo	0.0000
Rango	1.0000

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver tabla Tabla 59.

Tabla 59. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la undécima iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso

r_{12}	T(EA;0.0000)	p_{32}
r_{33}	T(A;-0.0196)	p_{36}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 60.

Tabla 60. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la undécima iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{12}	T(EA;0.0000)	p_{32}
r_{33}	T(A;-0.0196)	p_{36}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Duodécima iteración

Para calcular la siguiente ronda de asignaciones, se actualiza la tabla ***TPGFN***, retirando las asignaciones ya realizadas. El mayor de los valores de los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán asignarse.

Al calcular la ***FASDL*** para todos los recursos de la duodécima ronda, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, por haber un solo proceso restante, la ***FASDLN*** tendrá el valor máximo de la 2-tupla.

Además, como se ha indicado anteriormente, el mayor de los $\boldsymbol{TPGFN_{ijkl}}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada (***FASDLN***), ver Tabla 61.

Tabla 61. Orden ascendente de recursos y proceso al cual se asigna cada recurso en la duodécima iteración (***FASDLN***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{12}	T(EA;0.0000)	p_{35}

Fuente: Elaboración propia.

Ordenando la ***FASDLN*** se obtendrá el vector final ordenado, que constituye la Función de Asignación del Sistema Distribuido Lingüística Normalizada Ordenada (***FASDLNO***), como se puede ver en la Tabla 62.

Tabla 62. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en la duodécima iteración (***FASDLNO***)

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{12}	T(EA;0.0000)	p_{35}

Fuente: Elaboración propia.

La reiteración de todas las rondas de asignaciones da como resultado la Función de Asignación para Sistemas Distribuidos Lingüística Concatenada (***FASDLNC***) como se puede ver en la Tabla 63.

Tabla 63. Concatenación de todas las rondas de asignaciones (***FASDLNC***)

*R*ecurso	2-tuplas	*P*roceso	Ronda
r_{33}	T(EA;0.0000)	p_{23}	1
r_{12}	T(EA;-0.0326)	p_{37}	1
r_{31}	T(A;-0.0028)	p_{34}	1
r_{11}	T(A;-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(B;-0.0591)	p_{11}	1
r_{24}	T(EA;0.0000)	p_{34}	1
r_{33}	T(EA;0.0000)	p_{34}	2
r_{12}	T(MA;0.0758)	p_{34}	2
r_{31}	T(A;-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(MB;0.0173)	p_{34}	2
r_{24}	T(EA;0.0000)	p_{11}	2
r_{33}	T(EA;0.0000)	p_{13}	3
r_{12}	T(MA;0.0381)	p_{23}	3
r_{31}	T(M;0.0810)	p_{21}	3
r_{11}	T(M;-0.0133)	p_{13}	3
r_{22}	T(M;-0.0197)	p_{22}	3
r_{21}	T(M;-0.0789)	p_{13}	3
r_{32}	T(B;0.0227)	p_{13}	3
r_{13}	T(B;0.0159)	p_{31}	3
r_{23}	T(EB;0.0668)	p_{21}	3
r_{24}	T(EA;0.0000)	p_{23}	3
r_{33}	T(EA;0.0000)	p_{37}	4
r_{12}	T(MA;0.0255)	p_{13}	4
r_{31}	T(A;-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(B;0.0082)	p_{12}	4
r_{13}	T(B;-0.0337)	p_{21}	4
r_{32}	T(B;-0.0762)	p_{23}	4
r_{23}	T(EB;0.0489)	p_{32}	4
r_{24}	T(EA;0.0000)	p_{35}	4
r_{33}	T(EA;0.0000)	p_{12}	5
r_{12}	T(MA;0.0431)	p_{11}	5
r_{31}	T(A;-0.0777)	p_{31}	5
r_{22}	T(M;-0.0467)	p_{12}	5
r_{21}	T(B;0.0674)	p_{22}	5
r_{13}	T(B;-0.0824)	p_{32}	5
r_{11}	T(MB;0.0273)	p_{32}	5
r_{32}	T(MB;-0.0591)	p_{36}	5
r_{23}	T(EB;0.023)	p_{33}	5
r_{24}	T(EA;0.0000)	p_{36}	5
r_{33}	T(EA;0.0000)	p_{31}	6
r_{12}	T(MA;-0.0133)	p_{12}	6
r_{31}	T(M;0.0638)	p_{12}	6
r_{22}	T(B;0.0275)	p_{21}	6
r_{21}	T(B;-0.0106)	p_{11}	6
r_{13}	T(MB;0.0815)	p_{36}	6
r_{11}	T(MB;0.0443)	p_{36}	6
r_{32}	T(MB;-0.0604)	p_{35}	6
r_{23}	T(EB;0.0032)	p_{24}	6
r_{24}	T(EA;0.0000)	p_{24}	6
r_{33}	T(EA;0.0000)	p_{21}	7
r_{12}	T(MA;-0.0348)	p_{21}	7
r_{31}	T(M;0.0040)	p_{22}	7
r_{13}	T(MB;0.0742)	p_{35}	7
r_{21}	T(MB;0.0682)	p_{33}	7
r_{11}	T(MB;0.0631)	p_{33}	7
r_{22}	T(MB;0.058)	p_{35}	7
r_{32}	T(MB;-0.0553)	p_{33}	7
r_{33}	T(EA;0.0000)	p_{22}	8
r_{12}	T(EA;-0.021)	p_{33}	8

r_{31}	T(M;-0.0299)	p_{36}	8
r_{21}	T(MB;0.0191)	p_{36}	8
r_{13}	T(MB;0.011)	p_{33}	8
r_{22}	T(MB;0.0078)	p_{33}	8
r_{11}	T(MB;-0.0035)	p_{24}	8
r_{12}	T(EA;0.0000)	p_{36}	9
r_{33}	T(MA;-0.0117)	p_{33}	9
r_{31}	T(B;-0.0534)	p_{35}	9
r_{22}	T(EA;0.0000)	p_{36}	9
r_{12}	T(EA;0.0000)	p_{24}	10
r_{33}	T(MA;0.0734)	p_{35}	10
r_{12}	T(EA;0.0000)	p_{32}	11
r_{33}	T(A;-0.0196)	p_{36}	11
r_{12}	T(EA;0.0000)	p_{35}	12

Fuente: Elaboración propia.

De esta manera se han atendido todas las solicitudes de recursos de todos los procesos respetando la exclusión mutua y las prioridades de los procesos, las prioridades nodales y las prioridades finales.

2.6. Evaluación

El modelo propuesto logra que el sistema distribuido se auto regule reiteradamente en función del estado local de los *n* nodos, produciéndose una actualización de los estados locales de los mismos como consecuencia de la evolución de sus respectivos procesos y de las decisiones de acceso a los recursos: el sistema distribuido en el que se ejecutan grupos de procesos que acceden a recursos críticos, se observa a sí mismo y produce decisiones de accesos a recursos que modifican el estado del sistema y lo reajustan reiterativamente, garantizándose además la exclusión mutua en el acceso a los recursos compartidos, indicándose la prioridad de otorgamiento de acceso a cada recurso y el proceso al cual se lo asigna; este proceso se repite mientras haya procesos que soliciten acceso a recursos compartidos.

2.7. Discusiones y comentarios

El modelo propuesto permite que el sistema distribuido se autorregule repetidamente según el estado local de los nodos, lo que resulta en una actualización de sus estados locales, como

consecuencia de la evolución de sus respectivos procesos y de las decisiones de acceso a los recursos: el sistema distribuido en cuyos grupos de procesos se ejecuta el acceso a los recursos críticos, produce decisiones de acceso a los recursos que modifican el estado del sistema y lo reajustan repetidamente, garantizando además la exclusión mutua en el acceso a los recursos compartidos, indicando la prioridad de concesión de acceso a cada recurso y el proceso al que está asignado. Este proceso se repite si existen procesos que solicitan el acceso a los recursos compartidos.

El uso del modelo lingüístico de 2-tupla permite mejorar la precisión y facilitar el procesamiento de textos tratando el dominio lingüístico como continuo, pero manteniendo la base lingüística (sintáctica y semántica), a través de la traducción simbólica.

El contenido de este capítulo sirvió de base para las siguientes publicaciones:

- Fornerón, J., Agostini, F., La Red Martínez, D. (2020). Resource and Process Management with a Decision Model based on Fuzzy Logic. *World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI 2020).* (ACEPTADO)

- Fornerón, J., Agostini, F., La Red Martínez, D. (2020). Load Balancing in Distributed Systems with Fuzzy Logic. *International Journal of Interactive Multimedia and Artificial Intelligence (IJIMAI)*. (ENVIADO)

Capítulo III - Operador de agregación definido para el escenario 2 (E2)

3.1. Trabajos previos

En La Red Martínez (2017) y Agostini et al. (2018, 2019c) se desarrollan operadores de agregación para la asignación de recursos en sistemas distribuidos.

3.2. Premisas

Que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua **pudiendo constituir grupos** de procesos (los procesos independientes son considerados grupos unitarios); los procesos no requieren sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y **con exigencias estrictas de consenso** para lograr el acceso (se requiere un consenso para asignar de manera consecutiva los recursos solicitados por un proceso, es decir, que iniciada la secuencia de asignación de recursos a un proceso, la misma no puede ser interrumpida para asignar recursos a otro proceso).

3.3. Propuesta de solución

Se presentará una variante de un método innovador para la gestión de recursos compartidos en sistemas distribuidos, basado en La Red Martínez (2017) y La Red Martínez et al. (2018), donde se desarrolla un operador de agregación, para asignar recursos en sistemas distribuidos, pero en este caso, con un modelo de consenso que favorezca el acceso secuencial de los procesos a los recursos solicitados. Las premisas, las estructuras de datos y el operador mencionado en las publicaciones mencionadas se utilizan como punto de partida para crear un nuevo operador en el escenario descrito a continuación.

El escenario propuesto considera las siguientes condiciones: en primer lugar, los procesos

deben tener acceso a recursos compartidos en la modalidad de exclusión mutua, en segundo lugar, pueden ser capaces de formar grupos de procesos (los procesos que no pertenezcan a grupos se consideran independientes), en tercer lugar, los procesos no deben requerir sincronización (es decir, estar activos en sus respectivos procesadores al mismo tiempo) y hay estrictos requisitos de consenso para acceder a los recursos (se requiere un acuerdo para asignar todos los recursos solicitados a los procesos, es decir, una vez iniciada la secuencia de asignación de recursos a un proceso, no se puede interrumpir para otorgársela a otro proceso, hasta que el proceso activo los libere).

El operador de agregación de este escenario utiliza los puntos del 1 al 5 que se han desarrollado en el operador del capítulo anterior. Para satisfacer los requerimientos y aplicar el operador correspondiente a este escenario, se parte de la tabla ***FASDLNC***, correspondiente al paso 9 como se puede ver en la Fig. 6.

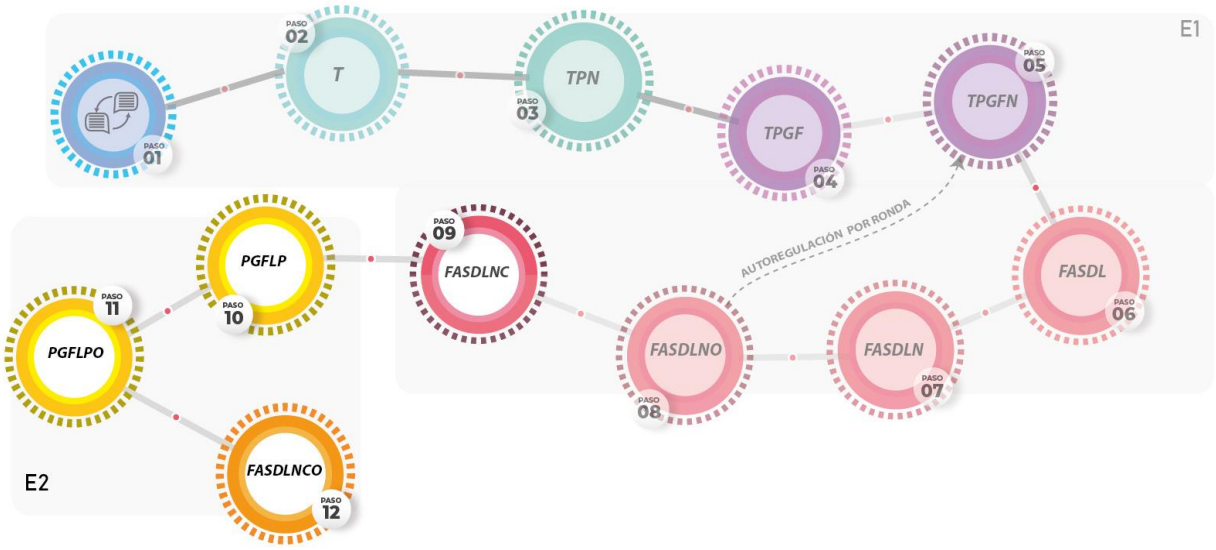


Fig. 6. Pasos para obtener la Función de Asignación para Sistemas Distribuidos Linguística Normalizada Concatenada Ordenada (***FASDLNCO***)

Fuente: Elaboración propia.

3.4. Descripción del operador de agregación

El operador propuesto consta de las siguientes etapas:

1) Expresión de los valores calculados en términos de 2-tupla utilizando un conjunto de etiquetas lingüísticas.

2) Cálculo de la carga computacional actual de los nodos.

3) Establecimiento de las categorías de carga computacional y de los vectores de pesos asociados a las mismas.

4) Cálculo de las prioridades o preferencias de los procesos teniendo en cuenta el estado del nodo (se las calcula en cada nodo para cada proceso).

5) Cálculo de las prioridades o preferencias de los procesos para acceder a los recursos compartidos disponibles (se las calcula en el administrador centralizado de recursos compartidos) y determinación del orden en que se asignarán los recursos y a qué proceso será asignado cada recurso.

6) Cálculo de las prioridades o preferencias de los procesos para acceder secuencialmente, a todos sus recursos compartidos.

Cálculo de las prioridades o preferencias de los procesos para acceder secuencialmente, a todos sus recursos compartidos.

El primer paso es obtener las prioridades globales definitivas para asignar los recursos (***Función de Asignación para Sistemas Distribuidos Lingüística Normalizada - FASDLN)***. Seguidamente establecer el orden o prioridad de asignación de los recursos y el proceso al que se asigna cada recurso (***Función de Asignación para Sistemas Distribuidos Lingüística***

Normalizada Ordenada - FASDLNO). La unión de todas las asignaciones ordenadas, realizadas en todas las rondas (***FASDLNO)*** conforman lo que se denomina (***Función de Asignación para Sistemas Distribuidos Lingüística Normalizada Concatenada - FASDLNC)***), como se muestra en la Fig. 7.

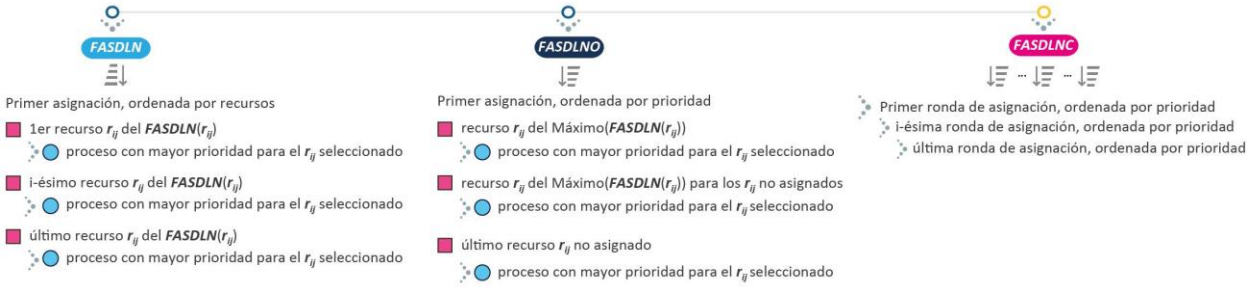


Fig. 7. Pasos para obtener la ***FASDLN***, ***FASDLNO*** y ***FASDLNC***.

Fuente: Elaboración propia.

La tabla ***FASDLNC***, se obtendrá a partir de la concatenación de las tablas ***FASDLNO*** de cada iteración, según se muestra en la Tabla 64.

Tabla 64. Concatenación de las tablas de asignación ordenada (***FASDLNC***)

FASDLNO	Iteraciones
FASDLNO 1er iteración	Filas de 1 a *n; n* = número de filas de la ***FASDLNO*** en la primer iteración
FASDLNO 2da iteración	Filas de *n*+1 a *m; m* = número de filas de la ***FASDLNO*** en la segunda iteración
…	…
FASDLNO última iteración	Filas de *m*+1 a *t; t* = número de filas de la ***FASDLNO*** en la última iteración

Fuente: Elaboración propia.

Prioridad global final del proceso

Una vez completada la tabla ***FASDLNC*** (Tabla 64), se calculará las **prioridades globales finales** de los procesos para acceder a sus recursos y se establecerá el orden en el que se ejecutará cada uno, recibiendo todos los recursos solicitados.

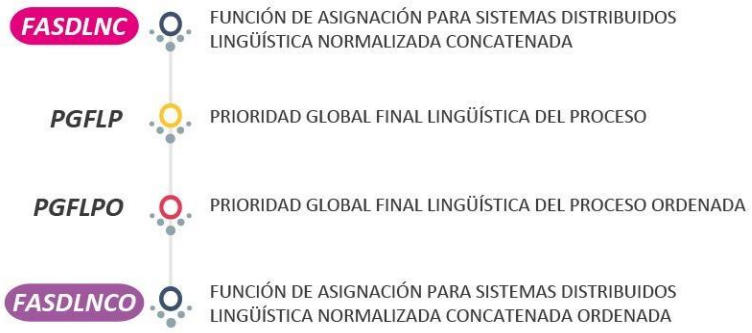


Fig. 8. Pasos para obtener desde la ***FASDLNC*** a la ***FASDLNCO***

Fuente: Elaboración propia.

En la Fig. 8 se muestra la serie de pasos que se debe realizar, se tendrá en cuenta la tabla ***FASDLNC***, con todos los procesos y recursos vinculados. Se sumará la prioridad de todas las asignaciones recurso/proceso, para cada proceso, y se las dividirá por la cantidad de asignaciones del mismo. El proceso que tenga mayor prioridad global final recibirá primero los recursos solicitados. Esto constituye lo que se denominará Prioridad Global Final Lingüística del Proceso (***PGFLP***), como se muestra en Fig. 9:

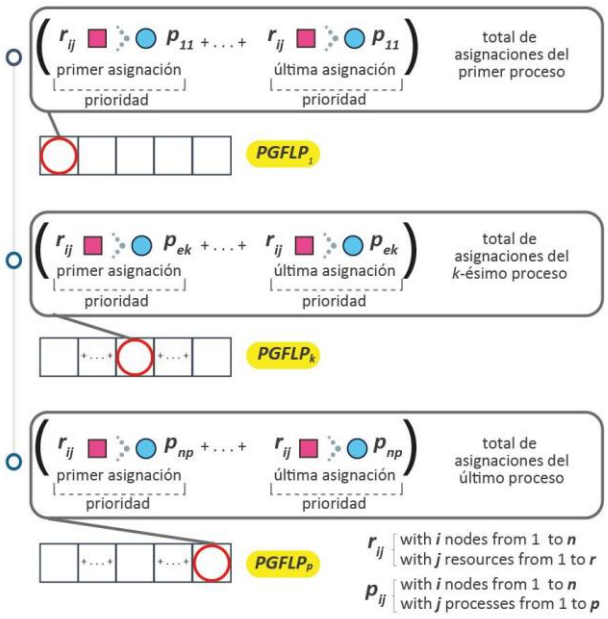


Fig. 9. Cálculo de la ***PGFLP*** de cada proceso.

Fuente: Elaboración propia.

$\boldsymbol{PGFLP}_i = i = 1,\ldots, h = \frac{\sum FASDLNCj}{Cantidad\ de\ asignaciones\ global\ del\ proceso\ i}$ *h= total de procesos en el sistema (sumatoria de procesos de los nodos); j=n° de recursos asignados al proceso i.*

Del vector ***PGFLP*** se debe ordenar sus elementos de mayor a menor para obtener el orden prioritario global de asignación de recursos a procesos, como se muestra en Fig. 10.

PGFLPO$_i$ = Prioridad Global Final Lingüística del Proceso Ordenada

j= cardinalidad de ***PGFLP*** *(total de procesos en el sistema)*

PGFLPO$_i$ = Max (***PGFLP$_i$*** no ordenado) *i*= 1,…, *j*

No ordenado = ***PGFLP$_i$*** $\notin$ ***PGFLPO***

1ero: ***PGFLPO$_1$*** *= Max (****PGFLP$_i$****) i= 1,..., j*

2do: ***PGFLPO$_2$*** *= Max (no ordenado* ***PGFLP$_i$****) i= 1,..., j*

último: ***PGFLPO$_j$*** *= Max (no ordenado* ***PGFLP$_i$****) i= 1, ..., j*

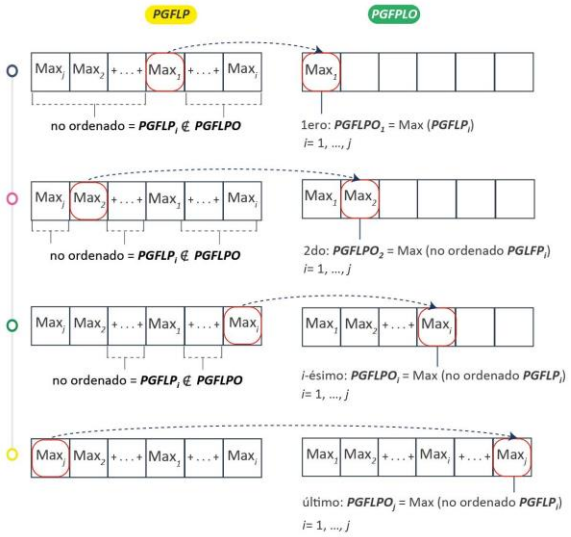


Fig. 10. Ejemplo para calcular la ***PGFLPO*** de cada proceso.

Fuente: Elaboración propia.

*Función de Asignación para Sistemas Distribuidos Lingüística Normalizada Concatenada Ordenada (**FASDLNCO**)*

La ***FASDLNCO*** establecerá el **orden de asignación por prioridades globales finales** de los procesos para acceder a sus recursos y se establecerá el orden en el que se asignará cada uno, recibiendo todos los recursos solicitados. Para ello, se tendrá en cuenta la tabla ***FASDLNC*** y la ***PGFLPO***.

Se calcularán las cardinalidades (número de asignaciones de recursos a cada proceso) obtenidas de cada uno de los procesos del vector ***PGFLPO*** en la tabla ***FASDLNC***.

cp_i= cardinalidad proceso (***PGFLPO***$_i$) en ***FASDLNC***

Luego, habrá que obtener cada una de las asignaciones de recursos a procesos en la tabla ***FASDLNC*** de cada uno de los procesos del vector ***PGFLPO***. El total de elementos por cada proceso estará determinado por la cardinalidad calculada en el paso anterior, como se puede ver en Fig. 11.

FASDLNCO$_1$ formado por el primer elemento de la ***FASDLNC*** para el proceso (***PGFLPO***$_1$) y ***FASDLNCO***$_{cp1}$ el último elemento de la ***FASDLNC*** para el proceso (***PGFLPO***$_1$).

FASDLNCO$_{cp1+1}$ formado por el primer elemento de la ***FASDLNC*** para el proceso (***PGFLPO***$_2$) y ***FASDLNCO***$_{cp1+cp2}$ el último elemento de la ***FASDLNC*** para el proceso (***PGFLPO***$_2$).

Se continúa el cálculo para todos los procesos intermedios.

FASDLNCO$_{cp1+cp2+\ldots+cpk-1+1}$ formado por el primer elemento de la ***FASDLNC*** para el proceso (***PGFLPO***$_k$) y ***FASDLNCO***$_{cp1+cp2+\ldots+cpk}$ el último elemento de la ***FASDLNC*** para el proceso (***PGFLPO***$_k$).

FASDLNCO$_{cp1+cp2+\ldots+cpn-1+1}$ formado por el primer elemento de la ***FASDLNC*** para el proceso(***PGFLPO***$_n$) y ***FASDLNCO***$_{cp1+cp2+\ldots+cpn}$ el último elemento de la ***FASDLNC*** para el proceso(***PGFLPO***$_n$).

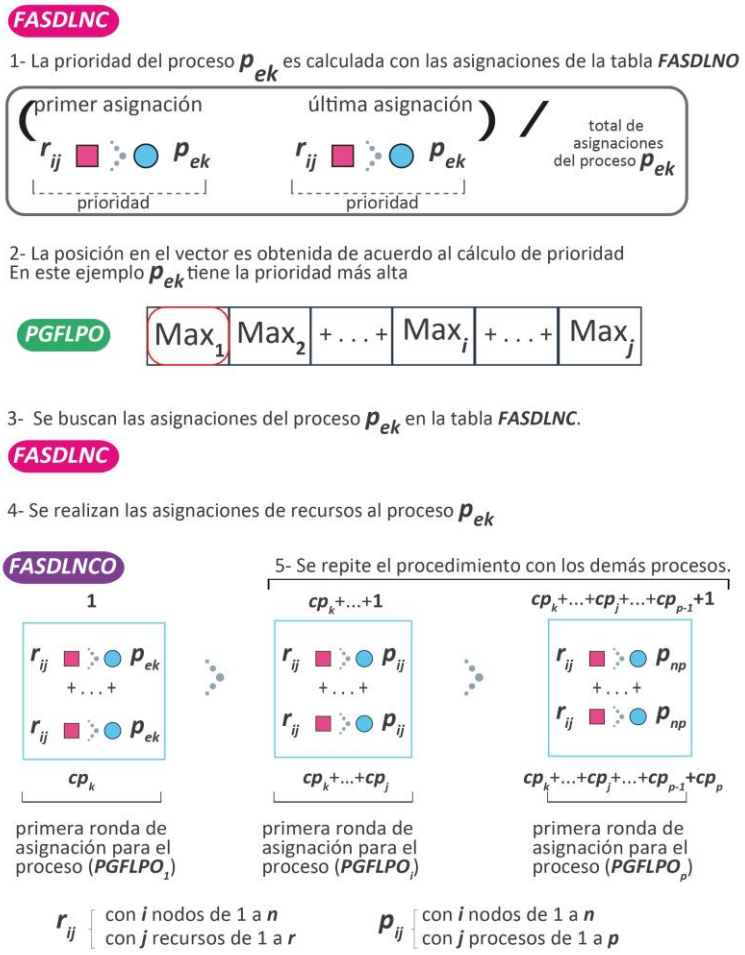


Fig. 11. Esquema para obtener la ***FASDLNCO*** de cada proceso.

Fuente: Elaboración propia.

De esta manera, se ha dado respuesta a todas las solicitudes de recursos de todos los procesos, respetando la exclusión mutua y las prioridades de los procesos, las prioridades nodales y las prioridades finales, teniendo en cuenta los estrictos requisitos de consenso establecidos para este escenario.

3.5. Ejemplo

En esta sección se explicará en detalle un ejemplo de aplicación del operador de agregación propuesto. El sistema de procesamiento distribuido, las estructuras de datos, los recursos y los procesos que se ejecutan en los diferentes nodos, grupos, cardinalidades, criterios y categorías para evaluar las diferentes cargas y cálculos necesarios, son los mencionados en La Red Martínez (2017), Agostini et al. (2018) y Agostini et al. (2019a, 2019b, 2019c).

El escenario que se presenta a continuación, parte de la concatenación de la asignación ordenada de cada una de las iteraciones correspondientes al escenario antes mencionado.

La tabla ***FASDLNC*** se obtendrá a partir de la concatenación de la tabla ***FASDLNO*** de cada iteración, como se muestra en la Tabla 65.

Tabla 65. Orden final de asignación de todas las iteraciones (***FASDLNC***)

Prioridad Final Ordenada	Asignación	Ronda
T(EA;0.0000)	r_{33} al p_{23}	1
T(EA;-0.0326)	r_{12} al p_{37}	1
T(A;-0.0028)	r_{31} al p_{34}	1
T(A;-0.0355)	r_{11} al p_{37}	1
T(M;0.0423)	r_{22} al p_{34}	1
T(M;0.0337)	r_{21} al p_{37}	1
T(M;0.0274)	r_{13} al p_{13}	1
T(M;-0.0334)	r_{32} al p_{34}	1
T(B;-0.0591)	r_{23} al p_{11}	1
T(EB;0.0000)	r_{24} al p_{34}	1
T(EA;0.0000)	r_{33} al p_{34}	2
T(MA;0.0758)	r_{12} al p_{34}	2
T(A;-0.0392)	r_{31} al p_{13}	2
T(M;0.0813)	r_{11} al p_{11}	2
T(M;-0.0076)	r_{21} al p_{25}	2
T(M;-0.0298)	r_{22} al p_{11}	2
T(M;-0.0347)	r_{13} al p_{34}	2
T(M;-0.0676)	r_{32} al p_{37}	2
T(MB;0.0173)	r_{23} al p_{34}	2
T(EB;0.0000)	r_{24} al p_{11}	2
T(EA;0.0000)	r_{33} al p_{13}	3
T(MA;0.0381)	r_{12} al p_{23}	3
T(M;0.0810)	r_{31} al p_{21}	3
T(M;-0.0133)	r_{11} al p_{13}	3
T(M;-0.0197)	r_{22} al p_{22}	3
T(M;-0.0789)	r_{21} al p_{13}	3
T(B;0.0227)	r_{32} al p_{13}	3
T(B;0.0159)	r_{13} al p_{31}	3
T(EB;0.0668)	r_{23} al p_{21}	3

T(EB;0.0000)	r_{24} al p_{23}	3
T(EA;0.0000)	r_{33} al p_{37}	4
T(MA;0.0255)	r_{12} al p_{13}	4
T(A;-0.0739)	r_{31} al p_{23}	4
T(M;-0.0178)	r_{22} al p_{13}	4
T(M;-0.0721)	r_{21} al p_{12}	4
T(B;0.0082)	r_{11} al p_{12}	4
T(B;-0.0337)	r_{13} al p_{21}	4
T(B;-0.0762)	r_{32} al p_{23}	4
T(EB;0.0489)	r_{23} al p_{32}	4
T(EB;0.0000)	r_{24} al p_{35}	4
T(EA;0.0000)	r_{33} al p_{12}	5
T(MA;0.0431)	r_{12} al p_{11}	5
T(A;-0.0777)	r_{31} al p_{31}	5
T(M;-0.0467)	r_{22} al p_{12}	5
T(B;0.0674)	r_{21} al p_{22}	5
T(B;-0.0824)	r_{13} al p_{32}	5
T(MB;0.0273)	r_{11} al p_{32}	5
T(MB;-0.0591)	r_{32} al p_{36}	5
T(EB;0.0230)	r_{23} al p_{33}	5
T(EB;0.0000)	r_{24} al p_{36}	5
T(EA;0.0000)	r_{33} al p_{31}	6
T(MA;-0.0133)	r_{12} al p_{12}	6
T(M;0.0638)	r_{31} al p_{12}	6
T(B;0.0275)	r_{22} al p_{21}	6
T(B;-0.0106)	r_{21} al p_{11}	6
T(MB;0.0815)	r_{13} al p_{36}	6
T(MB;0.0443)	r_{11} al p_{36}	6
T(MB;-0.0604)	r_{32} al p_{35}	6
T(EB;0.0032)	r_{23} al p_{24}	6
T(EB;0.0000)	r_{24} al p_{24}	6
T(EA;0.0000)	r_{33} al p_{21}	7
T(MA;-0.0348)	r_{12} al p_{21}	7
T(M;0.0040)	r_{31} al p_{22}	7
T(MB;0.0742)	r_{13} al p_{35}	7
T(MB;0.0682)	r_{21} al p_{33}	7
T(MB;0.0631)	r_{11} al p_{33}	7
T(MB;0.0580)	r_{22} al p_{35}	7
T(MB;-0.0553)	r_{32} al p_{33}	7
T(EA;0.0000)	r_{33} al p_{22}	8
T(EA;-0.0210)	r_{12} al p_{33}	8
T(M;-0.0299)	r_{31} al p_{36}	8
T(MB;0.0191)	r_{21} al p_{36}	8
T(MB;0.0110)	r_{13} al p_{33}	8
T(MB;0.0078)	r_{22} al p_{33}	8
T(MB;-0.0035)	r_{11} al p_{24}	8
T(EA;0.0000)	r_{12} al p_{36}	9
T(MA;-0.0117)	r_{33} al p_{33}	9
T(B;-0.0534)	r_{31} al p_{35}	9
T(EB;0.0000)	r_{22} al p_{36}	9
T(EA;0.0000)	r_{12} al p_{24}	10
T(MA;0.0734)	r_{33} al p_{35}	10
T(EA;0.0000)	r_{12} al p_{32}	11
T(A;-0.0196)	r_{33} al p_{36}	11
T(EA;0.0000)	r_{12} al p_{35}	12

Fuente: Elaboración propia.

La tabla de la ***FASDLNC*** ordenada por proceso, se muestra en la Tabla 66.

Tabla 66. Prioridad Global Final Lingüística Ordenada por proceso

Prioridad Final Global	Asignación	Rondas
T(B;-0.0591)	r_{23} al p_{11}	1
T(M;0.0813)	r_{11} al p_{11}	2
T(M;-0.0298)	r_{22} al p_{11}	2
T(EB;0.0000)	r_{24} al p_{11}	2
T(MA;0.0431)	r_{12} al p_{11}	5
T(B;-0.0106)	r_{21} al p_{11}	6
T(M;-0.0721)	r_{21} al p_{12}	4
T(B;0.0082)	r_{11} al p_{12}	4
T(EA;0.0000)	r_{33} al p_{12}	5
T(M;-0.0467)	r_{22} al p_{12}	5
T(MA;-0.0133)	r_{12} al p_{12}	6
T(M;0.0638)	r_{31} al p_{12}	6
T(M;0.0274)	r_{13} al p_{13}	1
T(A;-0.0392)	r_{31} al p_{13}	2
T(EA;0.0000)	r_{33} al p_{13}	3
T(M;-0.0133)	r_{11} al p_{13}	3
T(M;-0.0789)	r_{21} al p_{13}	3
T(B;0.0227)	r_{32} al p_{13}	3
T(MA;0.0255)	r_{12} al p_{13}	4
T(M;-0.0178)	r_{22} al p_{13}	4
T(M;0.0810)	r_{31} al p_{21}	3
T(EB;0.0668)	r_{23} al p_{21}	3
T(B;-0.0337)	r_{13} al p_{21}	4
T(B;0.0275)	r_{22} al p_{21}	6
T(EA;0.0000)	r_{33} al p_{21}	7
T(MA;-0.0348)	r_{12} al p_{21}	7
T(M;-0.0197)	r_{22} al p_{22}	3
T(B;0.0674)	r_{21} al p_{22}	5
T(M;0.0040)	r_{31} al p_{22}	7
T(EA;0.0000)	r_{33} al p_{22}	8
T(EA;0.0000)	r_{33} al p_{23}	1
T(MA;0.0381)	r_{12} al p_{23}	3
T(EB;0.0000)	r_{24} al p_{23}	3
T(A;-0.0739)	r_{31} al p_{23}	4
T(B;-0.0762)	r_{32} al p_{23}	4
T(EB;0.0032)	r_{23} al p_{24}	6
T(EB;0.0000)	r_{24} al p_{24}	6
T(MB;-0.0035)	r_{11} al p_{24}	8
T(EA;0.0000)	r_{12} al p_{24}	10
T(M;-0.0076)	r_{21} al p_{25}	2
T(B;0.0159)	r_{13} al p_{31}	3
T(A;-0.0777)	r_{31} al p_{31}	5
T(EA;0.0000)	r_{33} al p_{31}	6
T(EB;0.0489)	r_{23} al p_{32}	4
T(B;-0.0824)	r_{13} al p_{32}	5
T(MB;0.0273)	r_{11} al p_{32}	5
T(EA;0.0000)	r_{12} al p_{32}	11
T(EB;0.0230)	r_{23} al p_{33}	5
T(MB;0.0682)	r_{21} al p_{33}	7
T(MB;0.0631)	r_{11} al p_{33}	7
T(MB;-0.0553)	r_{32} al p_{33}	7
T(EA;-0.0210)	r_{12} al p_{33}	8
T(MB;0.0110)	r_{13} al p_{33}	8
T(MB;0.0078)	r_{22} al p_{33}	8
T(MA;-0.0117)	r_{33} al p_{33}	9
T(A;-0.0028)	r_{31} al p_{34}	1
T(M;0.0423)	r_{22} al p_{34}	1
T(M;-0.0334)	r_{32} al p_{34}	1
T(EB;0.0000)	r_{24} al p_{34}	1

T(EA;0.0000)	r_{33} al p_{34}	2
T(MA;0.0758)	r_{12} al p_{34}	2
T(M;-0.0347)	r_{13} al p_{34}	2
T(MB;0.0173)	r_{23} al p_{34}	2
T(EB;.0.0000)	r_{24} al p_{35}	4
T(MB;-0.0604)	r_{32} al p_{35}	6
T(MB;0.0742)	r_{13} al p_{35}	7
T(MB;0.0580)	r_{22} al p_{35}	7
T(B;-0.0534)	r_{31} al p_{35}	9
T(MA;0.0734)	r_{33} al p_{35}	10
T(EA;0.0000)	r_{12} al p_{35}	12
T(MB;-0.0591)	r_{32} al p_{36}	5
T(EB;0.0000)	r_{24} al p_{36}	5
T(MB;0.0815)	r_{13} al p_{36}	6
T(MB;0.0443)	r_{11} al p_{36}	6
T(M;-0.0299)	r_{31} al p_{36}	8
T(MB;0.0191)	r_{21} al p_{36}	8
T(EA;0.0000)	r_{12} al p_{36}	9
T(EB;0.0000)	r_{22} al p_{36}	9
T(A;-0.0196)	r_{33} al p_{36}	11
T(EA;-0.0326)	r_{12} al p_{37}	1
T(A;-0.0355)	r_{11} al p_{37}	1
T(M;0.0337)	r_{21} al p_{37}	1
T(M;-0.0676)	r_{32} al p_{37}	2
T(EA;0.0000)	r_{33} al p_{37}	4

Fuente: Elaboración propia.

Una vez que se haya completado la tabla de la ***FASDLNC***, se calcularán las Prioridades Finales Globales Lingüísticas del Proceso (***PGFLP***), para los cuales habrá que sumar el valor numérico representativo de cada 2-tupla de cada proceso y dividirlo por la cardinalidad de cada uno:

$\boldsymbol{PGFLP}_1$ = (T(B;-0.0591) + T(M;0.0813) + T(M;-0.0298) + T(EB;0.0000) + T(MA;0.0431) + T(B;-0.0106)) / 6

$\boldsymbol{PGFLP}_2$ = (T(M;-0.0721) + T(B;0.0082) + T(EA;0.0000) + T(M;-0.0467) + T(MA;-0.0133) + T(M;0.0638)) / 6

$\boldsymbol{PGFLP}_3$ = (T(M;0.0274) + T(A;-0.0392) + T(EA;0.0000) + T(M;-0.0133) + T(M;-0.0789) + T(B;0.0227) + T(MA;0.0255) + T(M;-0.0178)) / 8

$\boldsymbol{PGFLP}_4$ = (T(M;0.081) + T(EB;0.0668) + T(B;-0.0337) + T(B;0.0275) + T(EA;0.0000) + T(MA;-0.0348)) / 6

$\boldsymbol{PGFLP}_5$ = (T(M;-0.0197) + T(B;0.0674) + T(M;0.0040) + T(EA;0.0000)) / 4

$\boldsymbol{PGFLP}_6$ = (T(EA;0.0000) + T(MA;0.0381) + T(EB;0.0000) + T(A;-0.0739) + T(B;-0.0762)) / 5

$\boldsymbol{PGFLP}_7$ = (T(EB;0.0032) + T(EB;0.0000) + T(MB;-0.0035) + T(EA;0.0000)) / 4

$\boldsymbol{PGFLP}_8$ = (T(M;-0.0076)) / 1

$\boldsymbol{PGFLP}_9$ = (T(B;0.0159) + T(A;-0.0777) + T(EA;0.0000)) / 3

$\boldsymbol{PGFLP}_{10}$ = (T(EB;0.0489) + T(B;-0.0824) + T(MB;0.0273) + T(EA;0.0000)) / 4

$\boldsymbol{PGFLP}_{11}$ = (T(EB;0.0230) + T(MB;0.0682) + T(MB;0.0631) + T(MB;-0.0553) + T(EA;-0.021) + T(MB;0.011) + T(MB;0.0078) + T(MA;-0.0117)) / 8

$\boldsymbol{PGFLP}_{12}$ = (T(A;-0.0028) + T(M;0.0423) + T(M;-0.0334) + T(EB;0.0000) + T(EA;0.0000) + T(MA;0.0758) + T(M;-0.0347) + T(MB;0.0173)) / 8

$\boldsymbol{PGFLP}_{13}$ = (T(EB;0.0000) + T(MB;-0.0604) + T(MB;0.0742) + T(MB;0.0580) + T(B;-0.0534) + T(MA;0.0734) + T(EA;0.0000)) / 7

$\boldsymbol{PGFLP}_{14}$ = (T(MB;-0.0591) + T(EB;0.0000) + T(MB;0.0815) + T(MB;0.0443) + T(M;-0.0299) + T(MB;0.0191) + T(EA;0.0000) + T(EB;0.0000) + T(A;-0.0196)) / 9

$\boldsymbol{PGFLP}_{15}$ = (T(EA;-0.0326) + T(A;-0.0355) + T(M;0.0337) + T(M;-0.0676) + T(EA;0.0000)) / 5

Al calcular el ***PGFLP*** para todos los procesos de la Tabla 30, se obtendrá un vector, como se muestra en la Tabla 67. Del vector ***PGFLP*** se debe ordenar sus elementos de mayor a menor para obtener el orden prioritario global de asignación de recursos a procesos, como se puede observar en la Tabla 68.

Tabla 67. Prioridad Global Final Lingüística del Proceso (***PGFLP***)

Prioridad Global Final	Procesos
T(M;-0.0792)	p_{11}
T(A;-0.0656)	p_{12}
T(A;-0.0717)	p_{13}
T(M;0.0178)	p_{21}
T(A;-0.0704)	p_{22}
T(M;0.0443)	p_{23}
T(B;-0.0417)	p_{24}
T(M;-0.0076)	p_{25}
T(A;-0.0206)	p_{31}
T(B;0.0401)	p_{32}
T(B;0.0106)	p_{33}
T(M;0.0289)	p_{34}
T(B;0.0607)	p_{35}
T(B;-0.0145)	p_{36}
T(A;0.0463)	p_{37}

Fuente: Elaboración propia.

Tabla 68. Prioridad Global Final Lingüística del Proceso Ordenada (***PGFLPO***)

Prioridad Global Final Ordenada	Procesos
T(A;0.0463)	p_{37}
T(A;-0.0206)	p_{31}
T(A;-0.0656)	p_{12}
T(A;-0.0704)	p_{22}
T(A;-0.0717)	p_{13}
T(M;0.0443)	p_{23}
T(M;0.0289)	p_{34}
T(M;0.0178)	p_{21}
T(M;-0.0076)	p_{25}
T(M;-0.0792)	p_{11}
T(B;0.0607)	p_{35}
T(B;0.0401)	p_{32}
T(B;0.0106)	p_{33}
T(B;-0.0145)	p_{36}
T(B;-0.0417)	p_{24}

Fuente: Elaboración propia.

*Función de Asignación para Sistemas Distribuidos Lingüística Normalizada Concatenada Ordenada (**FASDLNCO**)*

La ***FASDLNCO*** establecerá el orden de asignación por prioridades globales finales lingüísticas de los procesos para acceder a sus recursos y se establecerá el orden en el que se asignará cada uno, recibiendo todos los recursos solicitados. Para ello, se tendrá en cuenta la tabla ***FASDLNC*** y la ***PGFLPO***.

Se calcularán las cardinalidades (número de asignaciones de recursos a cada proceso)

obtenidas de cada uno de los procesos del vector ***PGFLPO*** en la tabla ***FASDLNC***.

cp_i= cardinalidad proceso(***PGFLPO***$_i$) en ***FASCLNC***

cp_{37}= cardinalidad proceso(***PGFLPO***$_1$) en ***FASCLNC*** = 5

cp_{31}= cardinalidad proceso(***PGFLPO***$_2$) en ***FASCLNC*** = 3

cp_{12}= cardinalidad proceso(***PGFLPO***$_3$) en ***FASCLNC*** = 6

cp_{22}= cardinalidad proceso(***PGFLPO***$_4$) en ***FASCLNC*** = 4

cp_{13}= cardinalidad proceso(***PGFLPO***$_5$) en ***FASCLNC*** = 8

cp_{23}= cardinalidad proceso(***PGFLPO***$_6$) en ***FASCLNC*** = 5

cp_{34}= cardinalidad proceso(***PGFLPO***$_7$) en ***FASCLNC*** = 8

cp_{21}= cardinalidad proceso(***PGFLPO***$_8$) en ***FASCLNC*** = 6

cp_{25}= cardinalidad proceso(***PGFLPO***$_9$) en ***FASCLNC*** = 1

cp_{11}= cardinalidad proceso(***PGFLPO***$_{10}$) en ***FASCLNC*** = 6

cp_{35}= cardinalidad proceso(***PGFLPO***$_{11}$) en ***FASCLNC*** = 7

cp_{32}= cardinalidad proceso(***PGFLPO***$_{12}$) en ***FASCLNC*** = 4

cp_{33}= cardinalidad proceso(***PGFLPO***$_{13}$) en ***FASCLNC*** = 8

cp_{36}= cardinalidad proceso(***PGFLPO***$_{14}$) en ***FASCLNC*** = 9

cp_{24}= cardinalidad proceso(***PGFLPO***$_{15}$) en ***FASCLNC*** = 4

Luego, habrá que obtener cada una de las asignaciones de recursos a procesos en la tabla ***FASDLNC*** de cada uno de los procesos del vector ***PGFLPO***. El total de elementos por cada

proceso, estará determinado por la cardinalidad calculada en el paso anterior.

FASCLNCO$_1$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_1$) y ***FASCLNCO***$_5$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_1$).

FASCLNCO$_{5+1}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_2$) y ***FASCLNCO***$_{5+3}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_2$)

FASCLNCO$_{5+3+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_3$)

FASCLNCO$_{5+3+6}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_3$)

FASCLNCO$_{5+3+6+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_4$) y ***FASCLNCO***$_{5+3+6+4}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_4$)

FASCLNCO$_{5+3+6+4+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_5$) y ***FASCLNCO***$_{5+3+6+4+8}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_5$)

FASCLNCO$_{5+3+6+4+8+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_6$) y ***FASCLNCO***$_{5+3+6+4+8+5}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_6$)

FASCLNCO$_{5+3+6+4+8+5+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_7$) y ***FASCLNCO***$_{5+3+6+4+8+5+8}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_7$)

FASCLNCO$_{5+3+6+4+8+5+8+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_8$) y ***FASCLNCO***$_{5+3+6+4+8+5+8+6}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_8$)

FASCLNCO$_{5+3+6+4+8+5+8+6+1}$ el único elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_9$)

FASCLNCO$_{5+3+6+4+8+5+8+6+1+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{10}$) y ***FASCLNCO***$_{5+3+6+4+8+5+8+6+1+6}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{10}$)

FASCLNCO$_{5+3+6+4+8+5+8+6+1+6+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{11}$) y ***FASCLNCO***$_{5+3+6+4+8+5+8+6+1+6+7}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{11}$)

FASCLNCO$_{5+3+6+4+8+5+8+6+1+6+7+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{12}$) y ***FASCLNCO***$_{5+3+6+4+8+5+8+6+1+6+7+4}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{12}$)

FASCLNCO$_{5+3+6+4+8+5+8+6+1+6+7+4+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{13}$) y ***FASCLNCO***$_{5+3+6+4+8+5+8+6+1+6+7+4+8}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{13}$)

FASCLNCO$_{5+3+6+4+8+5+8+6+1+6+7+4+8+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{14}$) y ***FASCLNCO***$_{5+3+6+4+8+5+8+6+1+6+7+4+8+9}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{14}$)

FASCLNCO$_{5+3+6+4+8+5+8+6+1+6+7+4+8+9+1}$ el primer elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{15}$) *y* ***FASCLNCO***$_{5+3+6+4+8+5+8+6+1+6+7+4+8+9+4}$ el último elemento de la ***FASCLNC*** para el proceso (***PGFLPO***$_{15}$)

La Tabla 69 muestra el orden de todas las asignaciones de recursos para cada proceso, que es el primer proceso con mayor prioridad global y al que se asignan en primer lugar los recursos. La tabla completa continúa para cada una de las solicitudes de cada proceso (***FASDLNCO***).

Tabla 69. Orden Final de asignación en la ***FASDLNCO***

Prioridad	Asignación	Rondas
T(EA;-0.0326)	r_{12} al p_{37}	1
T(A;-0.0355)	r_{11} al p_{37}	1
T(M;0.0337)	r_{21} al p_{37}	1
T(M;-0.0676)	r_{32} al p_{37}	2
T(EA;0.0000)	r_{33} al p_{37}	4
T(B;0.0159)	r_{13} al p_{31}	3

T(A;-0.0777)	r_{31} al p_{31}	5
T(EA;0.0000)	r_{33} al p_{31}	6
T(M;-0.0721)	r_{21} al p_{12}	4
T(B;0.0082)	r_{11} al p_{12}	4
T(EA;0.0000)	r_{33} al p_{12}	5
T(M;-0.0467)	r_{22} al p_{12}	5
T(MA;-0.0133)	r_{12} al p_{12}	6
T(M;0.0638)	r_{31} al p_{12}	6
T(M;-0.0197)	r_{22} al p_{22}	3
T(B;0.0674)	r_{21} al p_{22}	5
T(M;0.0040)	r_{31} al p_{22}	7
T(EA;0.0000)	r_{33} al p_{22}	8
T(M;0.0274)	r_{13} al p_{13}	1
T(A;-0.0392)	r_{31} al p_{13}	2
T(EA;0.0000)	r_{33} al p_{13}	3
T(M;-0.0133)	r_{11} al p_{13}	3
T(M;-0.0789)	r_{21} al p_{13}	3
T(B;0.0227)	r_{32} al p_{13}	3
T(MA;0.0255)	r_{12} al p_{13}	4
T(M;-0.0178)	r_{22} al p_{13}	4
T(EA;0.0000)	r_{33} al p_{23}	1
T(MA;0.0381)	r_{12} al p_{23}	3
T(EB;0.0000)	r_{24} al p_{23}	3
T(A;-0.0739)	r_{31} al p_{23}	4
T(B;-0.0762)	r_{32} al p_{23}	4
T(A;-0.0028)	r_{31} al p_{34}	1
T(M;0.0423)	r_{22} al p_{34}	1
T(M;-0.0334)	r_{32} al p_{34}	1
T(EB;0.0000)	r_{24} al p_{34}	1
T(EA;0.0000)	r_{33} al p_{34}	2
T(MA;0.0758)	r_{12} al p_{34}	2
T(M;-0.0347)	r_{13} al p_{34}	2
T(MB;0.0173)	r_{23} al p_{34}	2
T(M;0.0810)	r_{31} al p_{21}	3
T(EB;0.0668)	r_{23} al p_{21}	3
T(B;-0.0337)	r_{13} al p_{21}	4
T(B;0.0275)	r_{22} al p_{21}	6
T(EA;0.0000)	r_{33} al p_{21}	7
T(MA;-0.0348)	r_{12} al p_{21}	7
T(M;-0.0076)	r_{21} al p_{25}	2
T(B;-0.0591)	r_{23} al p_{11}	1
T(M;0.0813)	r_{11} al p_{11}	2
T(M;-0.0298)	r_{22} al p_{11}	2
T(EB;0.0000)	r_{24} al p_{11}	2
T(MA;0.0431)	r_{12} al p_{11}	5
T(B;-0.0106)	r_{21} al p_{11}	6
T(EB;0.0000)	r_{24} al p_{35}	4
T(MB;-0.0604)	r_{32} al p_{35}	6
T(MB;0.0742)	r_{13} al p_{35}	7
T(MB;0.0580)	r_{22} al p_{35}	7
T(B;-0.0534)	r_{31} al p_{35}	9
T(MA;0.0734)	r_{33} al p_{35}	10
T(EA;0.0000)	r_{12} al p_{35}	12
T(EB;0.0489)	r_{23} al p_{32}	4
T(B;-0.0824)	r_{13} al p_{32}	5
T(MB;0.0273)	r_{11} al p_{32}	5
T(EA;0.0000)	r_{12} al p_{32}	11
T(EB;0.0230)	r_{23} al p_{33}	5
T(MB;0.0682)	r_{21} al p_{33}	7
T(MB;0.0631)	r_{11} al p_{33}	7
T(MB;-0.0553)	r_{32} al p_{33}	7
T(EA;-0.0210)	r_{12} al p_{33}	8

T(MB;0.0110)	r_{13} al p_{33}	8
T(MB;0.0078)	r_{22} al p_{33}	8
T(MA;-0.0117)	r_{33} al p_{33}	9
T(MB;-0.0591)	r_{32} al p_{36}	5
T(EB;0.0000)	r_{24} al p_{36}	5
T(MB;0.0815)	r_{13} al p_{36}	6
T(MB;0.0443)	r_{11} al p_{36}	6
T(M;-0.0299)	r_{31} al p_{36}	8
T(MB;0.0191)	r_{21} al p_{36}	8
T(EA;0.0000)	r_{12} al p_{36}	9
T(EB;0.0000)	r_{22} al p_{36}	9
T(A;-0.0196)	r_{33} al p_{36}	11
T(EB;0.0032)	r_{23} al p_{24}	6
T(EB;0.0000)	r_{24} al p_{24}	6
T(MB;-0.0035)	r_{11} al p_{24}	8
T(EA;0.0000)	r_{12} al p_{24}	10

Fuente: Elaboración propia.

De esta manera, se respondió a todas las solicitudes de recursos de todos los procesos, considerando la exclusión mutua y las prioridades de los procesos, las prioridades nodales y las prioridades finales, teniendo en cuenta los estrictos requisitos de consenso establecidos para este escenario.

3.6. Evaluación

El modelo propuesto logra establecer un consenso que permite a los procesos acceder a todos sus recursos de manera secuencial sin que éstos puedan ser removidos hasta que el proceso que los mantiene los libere. El orden de otorgamiento de recursos será determinado por la prioridad promedio general de todas las asignaciones en el supuesto del escenario general. El sistema distribuido regula y actualiza constantemente el estado local de cada nodo, las decisiones de acceso a los recursos modifican estos estados por lo que debe ser reajustado repetidamente, garantizando la exclusión mutua y reordenando nuevas prioridades. El método debe repetirse siempre que haya procesos que requieran recursos compartidos.

En La Red Martínez (2017), en el escenario general, las asignaciones de recursos a los diferentes procesos se realizan en varias rondas de asignación, donde un proceso puede tener varias asignaciones de recursos en las diferentes rondas.

Este capítulo considera el promedio global de prioridades que cada proceso tiene sobre todos los recursos de todas sus asignaciones en las diferentes rondas, pero para la asignación global final de cada proceso, respeta el mismo orden de asignación de cada recurso en las diferentes rondas en las que fueron asignados en el escenario general.

Es decir, la elección del proceso al cual se le otorga los recursos se determina con el promedio global de prioridades en todas las asignaciones de todas las rondas ***PGFLPO***, pero el orden en el que se deben realizar esas asignaciones, respeta el de la tabla ***FASDLNC***, para cada proceso.

La representación de los r_{ij} indican los recursos (cuyo primer subíndice representa al nodo donde se encuentra y el segundo subíndice al propio número de recurso) que se le asignan al proceso p_{ek} (cuyo primer subíndice representa al nodo donde se encuentra y el segundo subíndice al propio número de proceso) en cada ronda, ver Fig. 12.

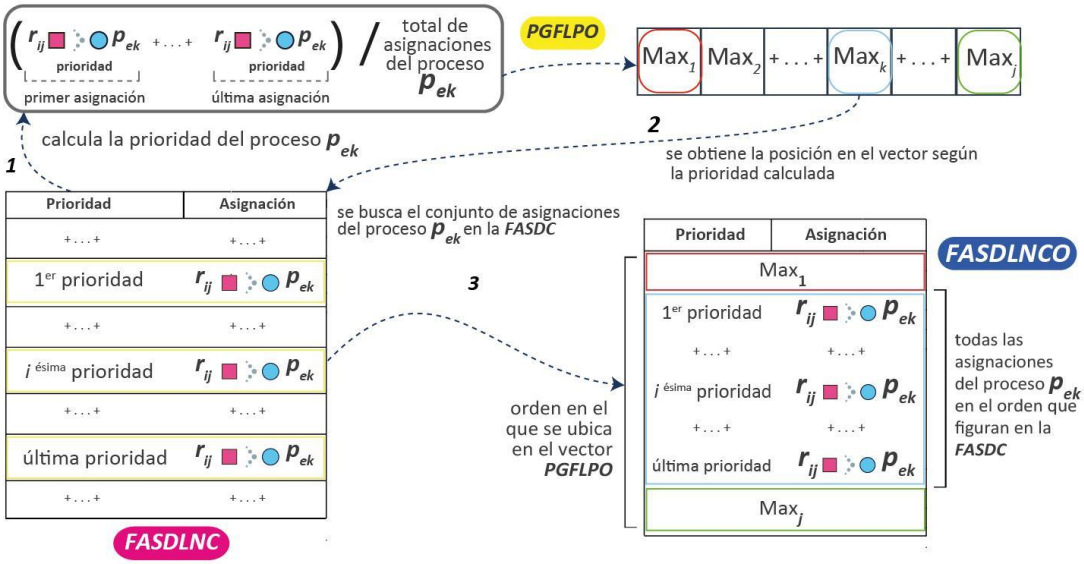


Fig. 12. Cálculo de prioridades para el proceso p_{ek} en ***PGFLPO.***

Fuente: Elaboración propia.

Aunque para los recursos los subíndices son distintos, no necesariamente son recursos

distintos, sino que pueden representar a un mismo recurso que se asigna en las distintas rondas, pero siempre al mismo proceso p_{ek}. La ubicación en la tabla ***FASDLNCO*** dependerá de la ubicación en el vector ***PGFLPO***.

El modelo propuesto incluye, como caso particular, un método que consiste en considerar la prioridad global de los procesos, en lugar de un grupo de variables de estado de cada nodo.

3.7. Discusiones y comentarios

Como los procesos se ejecutan en distintos procesadores y con todos sus recursos, no hay conflicto en cuanto a tener varios procesos en el mismo procesador.

En este escenario no se tiene en cuenta la cantidad de tiempo que cada proceso estará utilizando un nodo en particular. Tampoco se tiene en cuenta la cantidad de tiempo en que cada recurso estará asignado a un proceso en particular.

Se trata de un modelo general sin contemplar grupos de procesos. Se entiende por consenso que se priorice al proceso que resulte con mayor prioridad global promedio en sus asignaciones, según las tablas de asignaciones concatenada (***FASDLNC***).

Otra característica notable de la propuesta es su facilidad de implementación en el entorno de un administrador centralizado de recursos compartidos de un sistema distribuido.

Se buscó que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua pudiendo constituir grupos de procesos (los procesos independientes son considerados grupos unitarios); los procesos no requerían sincronización y tenían exigencias estrictas de consenso para lograr el acceso (se requería un consenso para asignar de manera consecutiva los recursos solicitados por un proceso o grupo de procesos, es decir, que iniciada la secuencia de

asignación de recursos a un proceso, la misma no pudo ser interrumpida para asignar recursos a otro proceso). Se ha presentado una variante de un método innovador para la gestión de recursos compartidos en sistemas distribuidos, basado en La Red Martínez (2017), Agostini et al. (2018) y Agostini et al. (2019a, 2019b, 2019c). Se ha establecido un modelo de consenso que favoreció el acceso secuencial de los procesos a todos los recursos solicitados. Se utilizó la tabla ***FASDLNC***, que es la concatenación de las tablas ***FASDLNO*** de todas las rondas del escenario general y consideró el promedio global de prioridades que cada proceso tuvo sobre todos los recursos de todas sus asignaciones en las diferentes rondas.

En el siguiente capítulo, se desarrolla una variante del método propuesto considerando otro operador de agregación para grupos de procesos que requieren requisitos de consenso estrictos.

El contenido de este capítulo sirvió de base para las siguientes publicaciones:

- Fornerón, J., Agostini, F. La Red Martínez, D. (2020). Gestión de procesos basado en lógica difusa con estrictos niveles de consenso. *International Journal of Information Systems and Software Engineering for Big Companies (IJISEBC)*. (ENVIADO)

Capítulo IV - Operador de agregación definido para el escenario 3 (E3)

4.1. Trabajos Previos

En La Red Martínez (2017) y Agostini et al. (2019a, 2019b) se desarrollan operadores de agregación para la asignación de recursos en sistemas distribuidos.

4.2. Premisas

Que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua **constituyendo grupos** de procesos (los procesos independientes son considerados grupos unitarios); los procesos no requieren sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y **con exigencias estrictas de consenso para lograr el acceso** (se requiere un consenso para asignar de manera consecutiva los recursos solicitados por un grupo de procesos, es decir, que iniciada la secuencia de asignación de recursos al grupo, la misma no puede ser interrumpida para asignar recursos a otro grupo).

4.3. Propuestas de solución

Se presentará una variante de un método innovador para la gestión de recursos compartidos en sistemas distribuidos, basado en La Red Martínez (2017) y La Red Martínez et al. (2018), donde se desarrolla un operador de agregación, para asignar recursos en sistemas distribuidos, pero en este caso, estableciendo un modelo de consenso que favorezca el acceso secuencial de los procesos de cada grupo a todos los recursos solicitados.

En los capítulos anteriores se han definido las siguientes prioridades globales: ***FASDLN (Función de Asignación para Sistemas Distribuidos Lingüística Normalizada)***, ***FASDLNO (Función de Asignación para Sistemas Distribuidos Lingüística Normalizada Ordenada)***,

FASDLNC (Función de Asignación para Sistemas Distribuidos Lingüística Normalizada Concatenada), y por último, la ***FASDLNCO (Función de Asignación para Sistemas Distribuidos Lingüística Normalizada Concatenada Ordenada)***.

Las premisas, las estructuras de datos y el operador mencionado en el tercer capítulo se utilizan como punto de partida para crear un nuevo operador en el escenario descrito a continuación, ver Fig. 13.

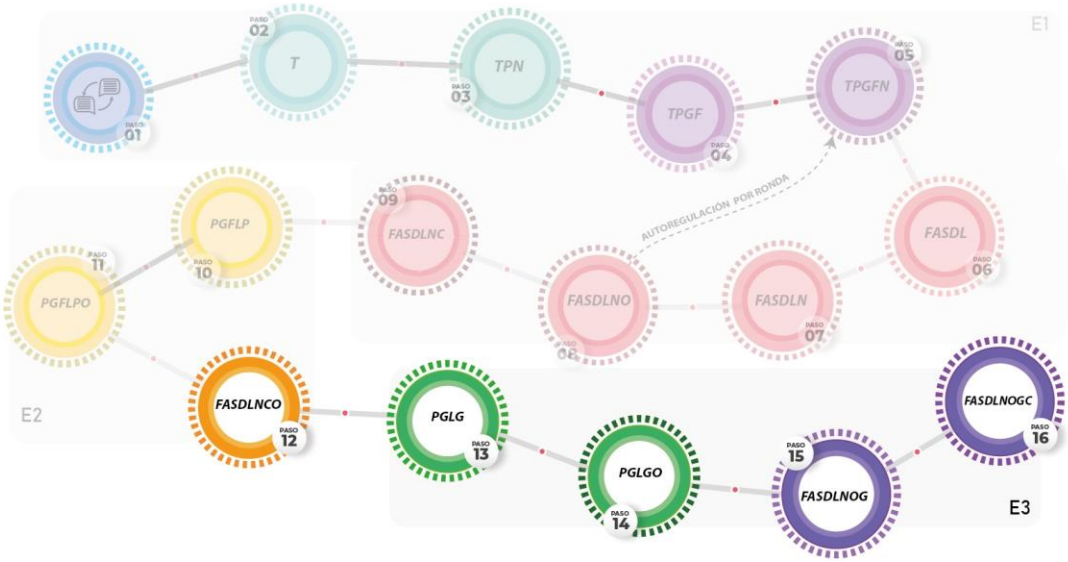


Fig. 13. Pasos para resolver los requerimientos de asignaciones en grupos de procesos.

Fuente: Elaboración propia.

El escenario propuesto considera las siguientes condiciones: en primer lugar, los procesos deben tener acceso a recursos compartidos en la modalidad de exclusión mutua, en segundo lugar, deben ser capaces de formar grupos de procesos (los procesos independientes se consideran grupos unitarios y con prioridad grupal cero), en tercer lugar, los procesos no deben requerir sincronización (es decir, estar activos en sus respectivos procesadores al mismo tiempo) y deben tener estrictos requisitos de consenso para acceder a los recursos (se requiere un acuerdo para asignar consecutivamente los recursos solicitados por todos los procesos de cada grupo, es decir,

una vez iniciada la secuencia de asignación de recursos al grupo, no se puede interrumpir para otorgársela a otro grupos de procesos, hasta que el proceso activo en cada grupo libere los recursos).

4.4. Descripción del operador de agregación

El operador propuesto consta de las siguientes etapas:

1. Expresión de los valores calculados en términos de 2-tupla utilizando un conjunto de etiquetas lingüísticas.

2. Cálculo de la carga computacional actual de los nodos.

3. Establecimiento de las categorías de carga computacional y de los vectores de pesos asociados a las mismas.

4. Cálculo de las prioridades o preferencias de los procesos teniendo en cuenta el estado del nodo (se las calcula en cada nodo para cada proceso).

5. Cálculo de las prioridades o preferencias de los procesos para acceder a los recursos compartidos disponibles (se las calcula en el administrador centralizado de recursos compartidos) y determinación del orden en que se asignarán los recursos y a qué proceso será asignado cada recurso.

6. Cálculo de las prioridades o preferencias de los procesos para acceder secuencialmente, a todos sus recursos compartidos.

7. Cálculo de las prioridades o preferencias de los grupos de procesos para acceder secuencialmente, a todos sus recursos compartidos mediante subrondas de asignación compatibles.

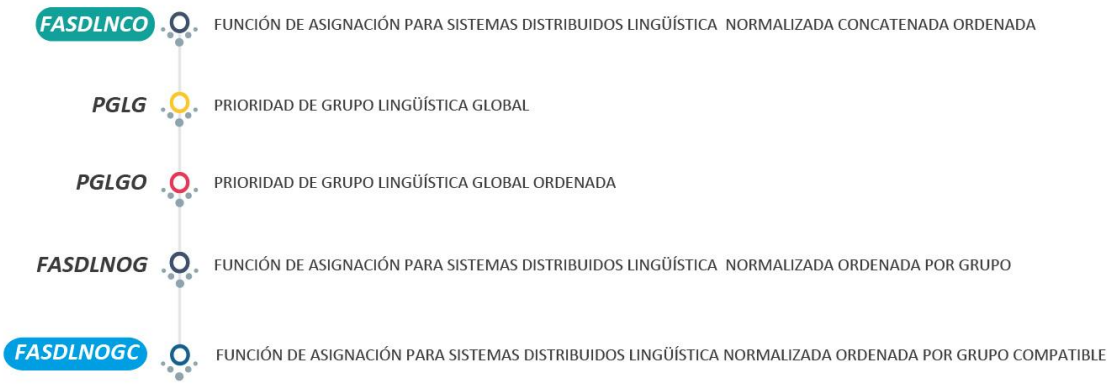


Fig. 14. Pasos para obtener desde la ***FASDLNCO*** a la ***FASDLNOGC.***

Fuente*: Elaboración propia.*

Este capítulo comienza con la tabla ***FASDLNCO*** (ver Fig. 14), donde se establecerá la **prioridad grupal** y el orden en el que se asignarán los recursos a procesos en los grupos. Para ellos se tendrá en cuenta todos los procesos y recursos vinculados en todas las instancias. La cardinalidad de grupo se medirá por la cantidad de procesos que hay en cada grupo, en todas las instancias.

Prioridad del grupo en la asignación

La sumatoria de todas las prioridades finales (***FASDLNCO***) de cada relación proceso y recurso por grupo, dividido la cardinalidad (números de asignaciones de cada grupo, en todas las instancias) del grupo, indicará la prioridad que tendrá dicho grupo para ser asignado, en relación a los demás grupos que también tendrán que ser asignados. Los procesos independientes, es decir, que no formen grupos, serán agrupados y considerados como el último grupo, su cardinalidad será el número de asignaciones que lo conformen y su prioridad cero. Esto constituye lo que se denominará Prioridad de Grupo Lingüística Global (***PGLG***), ver Fig. 15:

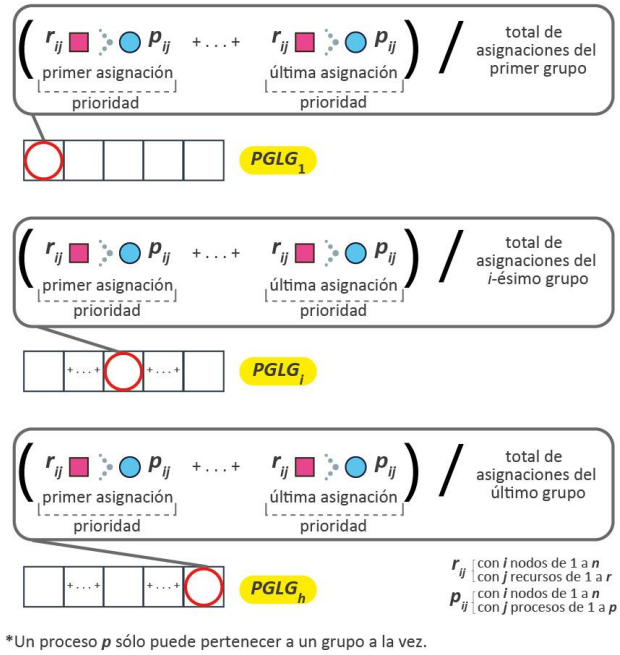


Fig. 15. Cálculo de la ***PGLG*** de cada grupo.

Fuente: Elaboración propia.

$$\boldsymbol{PGLG_i} = i = 1,\ldots, g = \frac{\sum \boldsymbol{FASDCO}\, j}{\text{Cantidad de asignaciones global del grupo } i}$$

g= total de grupos en el sistema (sumatoria de grupo de los nodos); j=n° de asignaciones del grupo i.

Del vector ***PGLG*** se debe ordenar sus elementos de mayor a menor para obtener el orden prioritario global de grupo, como se muestra en Fig. 16.

$\boldsymbol{PGLGO_i}$ = Prioridad Global Lingüística de Grupo Ordenada

g= cardinalidad de ***PGLG*** *(total de grupos en el sistema)*

$\boldsymbol{PGLGO_i}$ = Max ($\boldsymbol{PGLG_i}$ no ordenado) i= 1,…, g

No ordenado = $\boldsymbol{PGLG_i} \notin \boldsymbol{PGLGO}$

Orden de prioridad global de los grupos:

1ero: $\boldsymbol{PGLGO_1}$ = *Max* ($\boldsymbol{PGLG_i}$) i= 1,…, g

2do: ***PGLGO₂*** *= Max (****PGLGᵢ*** *no ordenado) i= 1,..., g*

...

Último: ***PGLGOⱼ*** *= Max (****PGLGᵢ*** *no ordenado) i= 1,..., g*

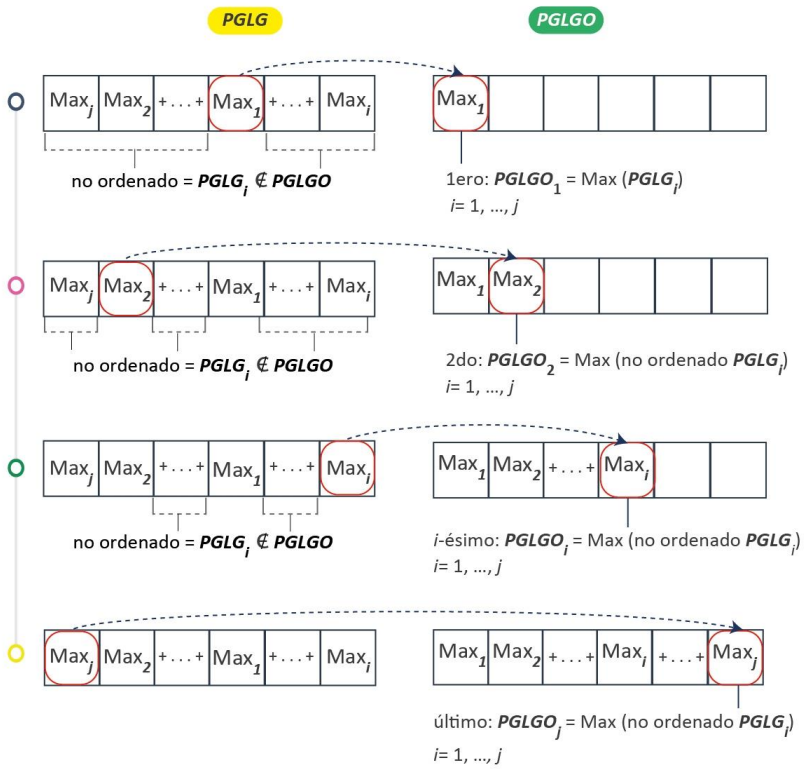


Fig. 16. Ejemplo para calcular la ***PGLGO*** de cada grupo de procesos.

Fuente: Elaboración propia.

*Función de Asignación para Sistemas Distribuidos Lingüística Normalizada Ordenada por Grupo (****FASDLNOG****)*

La ***FASDLNOG*** establecerá el **orden de asignación por prioridades de grupo** de los procesos para acceder a sus recursos y se establecerá el orden en el que se asignará cada uno. Para ello, se tendrá en cuenta la tabla ***FASDLNCO*** y la ***PGLGO***.

Se calcularán las cardinalidades (número de asignaciones de recursos a procesos) obtenidas de cada uno de los grupos del vector ***PGLGO*** en la tabla ***FASDLNCO***.

cg_i= cardinalidad grupo (***PGLGO***$_i$) en ***FASDLNCO***

Ejemplo:

cg_1= cardinalidad grupo (***PGLGO***$_1$) en ***FASDLNCO***

cg_2= cardinalidad grupo (***PGLGO***$_2$) en ***FASDLNCO***

cg_3= cardinalidad grupo (***PGLGO***$_3$) en ***FASDLNCO***

Luego habrá que obtener cada una de las asignaciones de recursos a procesos en la tabla ***FASDLNCO*** de cada uno de los grupos del vector ***PGLGO***. El total de elementos por cada grupo estará determinado por la cardinalidad calculada en el paso anterior.

FASDLNOG$_1$ es el primer elemento de la ***FASDLNCO*** para el grupo (***PGLGO***$_1$) y donde ***FASDLNOG***$_{cg1}$ es el último elemento de la ***FASDLNCO*** para el grupo (***PGLGO***$_1$)

FASDLNOG$_{cg1+1}$ es el primer elemento de la ***FASDLNCO*** para el grupo (***PGLGO***$_2$) y donde ***FASDLNOG***$_{cg1+cg2}$ es el último elemento de la ***FASDLNCO*** para el grupo (***PGLGO***$_2$)

FASDLNOG$_{cg1+cg2+\ldots+cgk-1+1}$ es el primer elemento de la ***FASDLNCO*** para el grupo (***PGLGO***$_k$) y donde ***FASDLNOG***$_{cg1+cg2+\ldots+cgk}$ es el último elemento de la ***FASDLNCO*** para el grupo (***PGLGO***$_k$)

FASDLNOG$_{cg1+cg2+\ldots+cgn-1+1}$ es el primer elemento de la ***FASDLNCO*** para el grupo (***PGLGO***$_n$) y donde ***FASDLNOG***$_{cg1+cg2+\ldots+cgn}$ es el último elemento de la ***FASDLNCO*** para el grupo (***PGLGO***$_n$)

Se designarán rondas de asignaciones a grupos, y para respetar la condición de exclusión mutua, habrá que establecer subrondas de asignaciones para aquellos procesos que no generen incompatiblidad en el acceso a los recursos, y otras subrondas para atender las asignaciones de

recursos a procesos que no han sido atendidos en la subronda anterior, por estar en situación de incompatibilidad.

*Función de Asignación para Sistemas Distribuidos Lingüística Normalizada Ordenada por Grupo Compatible (**FASDLNOGC**)*

La ***FASDLNOGC*** establecerá el orden de asignación por prioridades de grupo, de los procesos para acceder a sus recursos y se establecerá el orden en el que se asignará cada uno, considerando subrondas de asignación para procesos que no generen incompatibilidad en el acceso al recurso.

Dos asignaciones se consideran incompatibles, cuando involucran al mismo recurso y a procesos distintos, lo cual, violaría el principio de exclusión mutua.

Los procesos del mismo grupo que requieran un mismo recurso lo recibirán sucesivamente en las distintas subrondas, según la prioridad de la asignación.

Las rondas se darán a los grupos, las subrondas a las asignaciones de recursos a procesos que no generen incompatibilidad primero, y luego otras subrondas para aquellos que generaban incompatibilidad.

Se calcularán las cardinalidades de subgrupo (número de asignaciones de recursos a procesos por grupo compatible) obtenidas de cada uno de los grupos del vector ***PGLGO*** en la tabla ***FASDLNOG***.

$csub_i$= cardinalidad de subgrupo compatible (***PGLGO***$_i$) en ***FASDLNOG***

Para calcular la ***FASDLNOGC***, se tendrá en cuenta la tabla ***FASDLNOG*** y la ***PGLGO***:

FASDLNOGC$_i$ = Max (***FASDLNOG*** *i* no ordenado compatible) i= 1,…, j

No ordenado = $\{a_i\}$ de la ***FASDLNOG*** $\notin$ ***FASDLNOGC***

Compatible = que la asignación no involucre a un recurso que $\in$ ***FASDLNOGC***

La primera asignación corresponde al primer grupo de la ***PGLGO,*** y conforma la primera ronda, luego, cada iteracción de asignaciones de recursos a procesos que sean no ordenadas y compatibles, es decir, que no hayan sido cargadas (no ordenadas) en la matriz (***FASDLNOG***) y que sean compatibles, es decir, que no generen conflictos en el acceso a un recurso, conformaran cada subronda.

Primer subronda del primer grupo

$\boldsymbol{FASDLNOGC}_1$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_1$) y donde $\boldsymbol{FASDLNOGC}_{cg1sub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_1$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la primera subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la segunda subronda.

Segunda subronda del primer grupo

$\boldsymbol{FASDLNOGC}_{cg1sub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_1$) y donde $\boldsymbol{FASDLNOGC}_{cg1sub1+cg1sub2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_1$).

Se continúa este procedimiento para las ***k***-subrondas.

k subronda del primer grupo

FASDLNOGC$_{cg1subg1+cg1sub2+\ldots+cg1subk-1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_1$) y donde ***FASDLNOGC***$_{cg1sub1+cg1sub2+\ldots+cg1subk}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_1$).

Finalmente, se completa la ronda del primer grupo, con la última subronda correspondiente.

Última subronda del primer grupo

FASDLNOGC$_{cg1sub1+cg1sub2+\ldots+cg1subn-1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_1$) y donde ***FASDLNOGC***$_{cg1sub1+cg1sub2+\ldots+cg1subn}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_1$).

Una vez asignados todos los recursos no ordenados compatibles del primer grupo, se pasará a la siguiente ronda del siguiente grupo. La cantidad de subrondas lo determinará la cardinalidad por subrondas calculadas anteriormente. Se repite el mismo proceso del paso anterior para cada uno de los grupos.

Primer subronda del segundo grupo

FASDLNOGC$_{cg1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$) y donde ***FASDLNOGC***$_{cg1+cg2sub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$).

Segunda subronda del segundo grupo

FASDLNOGC$_{cg1+cg2sub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$) y donde ***FASDLNOGC***$_{cg1+cg2sub1+cg1sub2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$).

k subronda del segundo grupo

$\boldsymbol{FASDLNOGC}_{cg1+cg2subg1+cg2sub2+\ldots+cg2subk-1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_2$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2sub1+cg2sub2+\ldots+cg2subk}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_2$).

Última subronda del segundo grupo

$\boldsymbol{FASDLNOGC}_{cg1+cg2sub1+cg2sub2+\ldots+cg2subn-1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_2$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2sub1+cg2sub2+\ldots+cg2subn}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_2$).

Se repite el mismo proceso del paso anterior para cada uno de los ***k***-grupos.

Primer subronda del grupo k

$\boldsymbol{FASDLNOGC}_{cg1+cg2+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_k$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2+cgksub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_k$).

Segunda subronda del grupo k

$\boldsymbol{FASDLNOGC}_{cg1+cg2+cgksub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_k$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2+cgksub1+cgksub2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_k$).

i-ésima subronda del grupo k

$\boldsymbol{FASDLNOGC}_{cg1+cg2+cgksubg1+cgksub2+\ldots+cgksubi-1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_k$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2+cgksub1+cgksub2+\ldots+cgksubi}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_k$).

Última subronda del grupo k

$\boldsymbol{FASDLNOGC}_{cg1+cg2+cgksub1+cgksub2+\ldots+cgksubn-1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_k$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2+cgksub1+cgksub2+\ldots+cgksubn}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_k$).

Finalmente, se debe atender las solicitudes del último grupo y cada una de sus respectivas subrondas calculadas con la cardinalidad de subrondas de este grupo.

Primer subronda del grupo n

$\boldsymbol{FASDLNOGC}_{cg1+cg2+\ldots+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_n$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2+\ldots+cgnsub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_n$).

Segunda subronda del grupo n

$\boldsymbol{FASDLNOGC}_{cg1+cg2+\ldots+cgnsub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_n$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2+\ldots+cgnsub1+cgnsub2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_n$).

i-ésima subronda del grupo n

$\boldsymbol{FASDLNOGC}_{cg1+cg2+\ldots+cgnsubg1+cgnsub2+\ldots+cgnsubi-1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_n$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2+\ldots+cgnsub1+cgnsub2+\ldots+cgnsubi}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_n$).

Última subronda del grupo n

$\boldsymbol{FASDLNOGC}_{cg1+cg2+\ldots+cgnsub1+cgnsub2+\ldots+cgnsubn-1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_n$) y donde

FASDLNOGC$_{cg1+cg2+\ldots+cgnsub1+cgnsub2+\ldots+cgnsubn}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_n$).

Tabla 70. Esquema general de rondas y de subrondas de cada grupo

Ronda de asignación de recursos por grupo	*Subronda de asignación de recursos según compatibilidad*	*Proceso al que se asignará el recurso*
h-ésima ronda: ***PGLGO***$_h$ = *Max (****PGLG***$_h$*)* *h = 1, …, s*	***primer subronda*** *de asignación de recursos a procesos que no generan incompatibilidad*	***FASDLNOGC***$_1$ = Max (***FASDLNOG*** $_i$ no ordenado compatible) *i*= 1, …, *j* … ***FASDLNOGC***$_{cg1sub1}$ = Max (***FASDLNOG*** $_i$ no ordenado compatible) *i*= 1, …, *j*
	g-ésima subronda *de asignación de recursos a procesos que no generan incompatibilidad*	***FASDLNOGC***$_{cg1sub1+\ldots+1}$ = Max (***FASDLNOG*** $_i$ no ordenado compatible) *i*= 1, …, *j* … ***FASDLNOGC***$_{cg1sub1+\ldots+cg1subi}$ = Max (***FASDLNOG*** $_i$ no ordenado compatible) *i*= 1, …, *j*
	última subronda *de asignación de recursos a procesos que no generan incompatibilidad*	***FASDLNOGC***$_{cg1sub1+\ldots+cg1subn-1+1}$ = Max (***FASDLNOG*** $_i$ no ordenado compatible) *i*= 1, …, *j* … ***FASDLNOGC***$_{cg1sub1+\ldots+cg1subn}$ = Max (***FASDLNOG*** $_i$ no ordenado compatible) *i*= 1, …, *j*

Fuente: Elaboración propia.

De esta manera, se ha dado respuesta a todas las solicitudes de recursos de todos los procesos, respetando la exclusión mutua y las prioridades de los procesos, las prioridades nodales y las prioridades finales, teniendo en cuenta los estrictos requisitos de consenso establecidos para este escenario, ver Tabla 70.

4.5. Ejemplo

El escenario que se presenta a continuación, parte de la concatenación de la asignación ordenada de cada una de las iteraciones correspondientes al escenario antes mencionado. Todas las figuras y tablas que consignan fuente de elaboración propia, basado en La Red Martínez (2017), se transcriben en el ejemplo del escenario (E1).

La ***FASDLNCO*** se puede observar en la Tabla 71 detallada en el escenario (E2).

Tabla 71. Orden Final de asignación en la ***FASDLNCO***

Prioridad	Asignación	Rondas	Grupos
T(EA;-0.0326)	r_{12} al p_{37}	1	g_1
T(A;-0.0355)	r_{11} al p_{37}	1	g_1
T(M;0.0337)	r_{21} al p_{37}	1	g_1
T(M;-0.0676)	r_{32} al p_{37}	2	g_1
T(EA;0.0000)	r_{33} al p_{37}	4	g_1
T(B;0.0159)	r_{13} al p_{31}	3	g_3
T(A;-0.0777)	r_{31} al p_{31}	5	g_3
T(EA;0.0000)	r_{33} al p_{31}	6	g_3
T(M;-0.0721)	r_{21} al p_{12}	4	g_2
T(B;0.0082)	r_{11} al p_{12}	4	g_2
T(EA;0.0000)	r_{33} al p_{12}	5	g_2
T(M;-0.0467)	r_{22} al p_{12}	5	g_2
T(MA;-0.0133)	r_{12} al p_{12}	6	g_2
T(M;0.0638)	r_{31} al p_{12}	6	g_2
T(M;-0.0197)	r_{22} al p_{22}	3	g_3
T(B;0.0674)	r_{21} al p_{22}	5	g_3
T(M;0.0040)	r_{31} al p_{22}	7	g_3
T(EA;0.0000)	r_{33} al p_{22}	8	g_3
T(M;0.0274)	r_{13} al p_{13}	1	g_4
T(A;-0.0392)	r_{31} al p_{13}	2	g_4
T(EA;0.0000)	r_{33} al p_{13}	3	g_4
T(M;-0.0133)	r_{11} al p_{13}	3	g_4
T(M;-0.0789)	r_{21} al p_{13}	3	g_4
T(B;0.0227)	r_{32} al p_{13}	3	g_4
T(MA;0.0255)	r_{12} al p_{13}	4	g_4
T(M;-0.0178)	r_{22} al p_{13}	4	g_4
T(EA;0.0000)	r_{33} al p_{23}	1	g_4
T(MA;0.0381)	r_{12} al p_{23}	3	g_4
T(EB;0.0000)	r_{24} al p_{23}	3	g_4
T(A;-0.0739)	r_{31} al p_{23}	4	g_4
T(B;-0.0762)	r_{32} al p_{23}	4	g_4
T(A;-0.0028)	r_{31} al p_{34}	1	g_4
T(M;0.0423)	r_{22} al p_{34}	1	g_4
T(M;-0.0334)	r_{32} al p_{34}	1	g_4
T(EB;0.0000)	r_{24} al p_{34}	1	g_4
T(EA;0.0000)	r_{33} al p_{34}	2	g_4
T(MA;0.0758)	r_{12} al p_{34}	2	g_4
T(M;-0.0347)	r_{13} al p_{34}	2	g_4
T(MB;0.0173)	r_{23} al p_{34}	2	g_4
T(M;0.0810)	r_{31} al p_{21}	3	g_2
T(EB;0.0668)	r_{23} al p_{21}	3	g_2
T(B;-0.0337)	r_{13} al p_{21}	4	g_2
T(B;0.0275)	r_{22} al p_{21}	6	g_2
T(EA;0.0000)	r_{33} al p_{21}	7	g_2
T(MA;-0.0348)	r_{12} al p_{21}	7	g_2
T(M;-0.0076)	r_{21} al p_{25}	2	g_1
T(B;-0.0591)	r_{23} al p_{11}	1	g_1
T(M;0.0813)	r_{11} al p_{11}	2	g_1
T(M;-0.0298)	r_{22} al p_{11}	2	g_1
T(EB;0.0000)	r_{24} al p_{11}	2	g_1
T(MA;0.0431)	r_{12} al p_{11}	5	g_1
T(B;-0.0106)	r_{21} al p_{11}	6	g_1
T(EB;0.0000)	r_{24} al p_{35}	4	g_0
T(MB;-0.0604)	r_{32} al p_{35}	6	g_0
T(MB;0.0742)	r_{13} al p_{35}	7	g_0
T(MB;0.0580)	r_{22} al p_{35}	7	g_0
T(B;-0.0534)	r_{31} al p_{35}	9	g_0
T(MA;0.0734)	r_{33} al p_{35}	10	g_0
T(EA;0.0000)	r_{12} al p_{35}	12	g_0

T(EB;0.0489)	r_{23} al p_{32}	4	g_0
T(B;-0.0824)	r_{13} al p_{32}	5	g_0
T(MB;0.0273)	r_{11} al p_{32}	5	g_0
T(EA;0.0000)	r_{12} al p_{32}	11	g_0
T(EB;0.0230)	r_{23} al p_{33}	5	g_0
T(MB;0.0682)	r_{21} al p_{33}	7	g_0
T(MB;0.0631)	r_{11} al p_{33}	7	g_0
T(MB;-0.0553)	r_{32} al p_{33}	7	g_0
T(EA;-0.0210)	r_{12} al p_{33}	8	g_0
T(MB;0.0110)	r_{13} al p_{33}	8	g_0
T(MB;0.0078)	r_{22} al p_{33}	8	g_0
T(MA;-0.0117)	r_{33} al p_{33}	9	g_0
T(MB;-0.0591)	r_{32} al p_{36}	5	g_0
T(EB;0.0000)	r_{24} al p_{36}	5	g_0
T(MB;0.0815)	r_{13} al p_{36}	6	g_0
T(MB;0.0443)	r_{11} al p_{36}	6	g_0
T(M;-0.0299)	r_{31} al p_{36}	8	g_0
T(MB;0.0191)	r_{21} al p_{36}	8	g_0
T(EA;0.0000)	r_{12} al p_{36}	9	g_0
T(EB;0.0000)	r_{22} al p_{36}	9	g_0
T(A;-0.0196)	r_{33} al p_{36}	11	g_0
T(EB;0.0032)	r_{23} al p_{24}	6	g_0
T(EB;0.0000)	r_{24} al p_{24}	6	g_0
T(MB;-0.0035)	r_{11} al p_{24}	8	g_0
T(EA;0.0000)	r_{12} al p_{24}	10	g_0

Fuente: Elaboración propia.

La Tabla 71 muestra el orden de todas las asignaciones de recursos para cada proceso, que es el primer proceso con mayor prioridad global y al que se asignan en primer lugar los recursos.

Una vez que se haya completado la tabla de la ***FASDLNCO***, con la información de grupo de cada proceso, se calcularán las Prioridades Finales Globales Lingüística del Grupo (***PGLG***):

PGLG$_1$ = (T(EA;-0.0326) + T(A;-0.0355) + T(M;0.0337) + T(M;-0.0676) + T(EA;0.0000) + T(M;-0.0076) + T(B;-0.0591) + T(M;0.0813) + T(M;-0.0298) + T(EB;0.0000) + T(MA;0.0431 + T(B;-0.0106)) / 12

g_1= 0.5485 = T(M;0.0485)

PGLG$_2$ = (T(M;-0.0721) + T(B;0.0082) + T(EA;0.0000) + T(M;-0.0467) + T(MA;-0.0133) + T(M;0.0638) + T(M;0.0810) + T(EB;0.0668) + T(B;-0.0337) + T(B;0.0275) + T(EA;0.0000) + T(MA;-0.0348)) / 12

g_2 = 0.5594 = T(M;0.0594)

$\boldsymbol{PGLG}_3$ = (T(B;0.0159) + T(A;-0.0777) + T(EA;0.0000) + T(M;-0.0197) + T(B;0.0674) + T(M;0.0040) + T(EA;0.0000)) / 7

g_3 = 0.6176 = T(A;-0.0491)

$\boldsymbol{PGLG}_4$ = (T(M;0.0274) + T(A;-0.0392) + T(EA;0.0000) + T(M;-0.0133) + T(M;-0.0789) + T(B;0.0227) + T(MA;0.0255) + T(M;-0.0178) + T(EA;0.0000) + T(MA;0.0381) + T(EB;0.0000) + T(A;-0.0739) + T(B;-0.0762) + T(A;-0.0028) + T(M;0.0423) + T(M;-0.0334) + T(EB;0.0000) + T(EA;0.0000) + T(MA;0.0758) + T(M;-0.0347) + T(MB;0.0173)) / 21

g_4 = 0.2961 = T(B;-0.0372)

$\boldsymbol{PGLG}_0$ = la prioridad el g_0 es 0, porque esta conformado por procesos independientes, por lo tanto la 2-tupla correspondiente indicará la menor prioridad.

g_0 = 0.0000 = T(EB; 0.0000)

Al calcular el ***PGLG*** para todos los grupos, se obtendrá un vector como se muestra en la Tabla 72.

Tabla 72. Prioridad Global Lingüística del Grupo (***PGLG***)

Prioridad Global Final	Grupo
T(M;0.0485)	g_1
T(M;0.0594)	g_2
T(A;-0.0491)	g_3
T(B;-0.0372)	g_4
T(EB; 0.0000)	g_0

Fuente: Elaboración propia.

Del vector ***PGLG*** se debe ordenar sus elementos de mayor a menor para obtener el orden prioritario global de asignación a grupos, como se puede observar en la Tabla 73.

Tabla 73. Prioridad Global Lingüística del Grupo Ordenada (***PGLGO***)

Prioridad Global Final Ordenada	Grupo
T(A;-0.0491)	g_3
T(M;0.0594)	g_2
T(M;0.0485)	g_1
T(B;-0.0372)	g_4
T(EB; 0.0000)	g_0

Fuente: Elaboración propia.

Función de Asignación para Sistemas Distribuidos Ordenada por Grupo (FASDLNOG)

La ***FASDLNOG*** establecerá el **orden de asignación por prioridades de grupo** de los procesos para acceder a sus recursos y se establecerá el orden en el que se asignará cada uno. Para ello, se tendrá en cuenta la tabla ***FASDLNCO*** y la ***PGLGO***.

Se calcularán las cardinalidades (número de asignaciones de recursos a procesos) obtenidas de cada uno de los grupos del vector ***PGLGO*** en la tabla ***FASDLNCO***.

Luego habrá que obtener cada una de las asignaciones de recursos a procesos en la tabla ***FASDLNCO*** de cada uno de los grupos del vector ***PGLGO***. El total de elementos por cada grupo, estará determinado por la cardinalidad calculada en el paso anterior.

*Cálculo de las **FASDLNOG** para el primer grupo*

FASDLNOG $_{1}$ es el primer elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_1$) y donde ***FASDLNOG***$_7$ es el último elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_1$).

*Cálculo de las **FASDLNOG** para el segundo grupo*

FASDLNOG $_{7+1}$ es el primer elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_2$) y donde ***FASDLNOG***$_{7+12}$ es el último elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_2$).

*Cálculo de las **FASDLNOG** para el tercer grupo*

FASDLNOG $_{7+12+1}$ es el primer elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_3$) y donde ***FASDLNOG*** $_{7+12+12}$ es el último elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_3$).

*Cálculo de las **FASDLNOG** para el cuarto grupo*

FASDLNOG $_{7+12+12+1}$ es el primer elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_4$) y donde ***FASDLNOG*** $_{7+12+12+21}$ es el último elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_4$).

*Cálculo de las **FASDLNOG** para el quinto grupo*

FASDLNOG $_{7+12+12+21+1}$ es el primer elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_5$) y donde ***FASDLNOG*** $_{7+12+12+21+32}$ es el último elemento de la ***FASDLNCO*** para el grupo de procesos (***PGLGO***$_5$).

Se designarán rondas de asignaciones a grupos, y para respetar la condición de exclusión mutua, habrá que establecer subrondas de asignaciones para aquellos procesos que no generen incompatiblidad en el acceso a los recursos, y otras subrondas para atender las asignaciones de recursos a procesos que no han sido atendidos en la subronda anterior, por estar en situación de incompatibilidad.

Función de Asignación para Sistemas Distribuidos Ordenada por Grupo Compatible (FASDLNOGC)

La ***FASDLNOGC*** establecerá el orden de asignación por prioridades de grupo, de los procesos para acceder a sus recursos y se establecerá el orden en el que se asignará cada uno, considerando subrondas de asignación para procesos que no generen incompatibilidad en el acceso al recurso.

Dos asignaciones se consideran incompatibles, cuando involucran al mismo recurso y a procesos distintos, lo cual, violaría el principio de exclusión mutua.

Los procesos del mismo grupo que requieran un mismo recurso lo recibirán sucesivamente en las distintas subrondas, según la prioridad de la asignación.

Las rondas se darán a los grupos, las subrondas a las asignaciones de recursos a procesos que no generen incompatibilidad primero, y luego otras subrondas para aquellos que generaban incompatibilidad.

Se calcularán las cardinalidades de subgrupo (número de asignaciones de recursos a procesos por grupo compatible) obtenidas de cada uno de los grupos del vector ***PGLGO*** en la tabla ***FASDLNCO***.

$csub_i$= cardinalidad de subgrupo compatible (***PGLGO***$_i$) en ***FASDLNCO***

Para calcular la ***FASDLNOGC*** se tendrá en cuenta la tabla ***FASDLNOG*** y la ***PGLGO***:

FASDLNOGC$_i$ = Max (***FASDLNOG*** *i* no ordenado compatible) i= 1,…, j

No ordenado = {a_i} de la ***FASDLNOG*** $\notin$ ***FASDLNOGC***

Compatible = que la asignación no involucre a un recurso que $\in$ ***FASDLNOGC***

En la Fig. 17 se ven los grupos involucrados en el cálculo de la ***FASDLNOG*** compatibles.

Fig. 17. Grupos de procesos y grupos de procesos independientes.

Fuente: Elaboración propia.

La primera asignación corresponde al primer grupo de la ***PGLGO,*** y conforma la primera

ronda, luego, cada iteracción de asignaciones de recursos a procesos que sean no ordenadas y compatibles, es decir, que no hayan sido cargadas (no ordenadas) en la matriz (***FASDLNOG***) y que sean compatibles, es decir, que no generen conflictos en el acceso a un recurso, conformaran cada subronda.

Ronda de asignación del primer grupo (g_3)

El primer paso es atender las asignaciones de recursos a procesos que no generan incompatibilidad en el grupo g_3, ver Fig. 18.

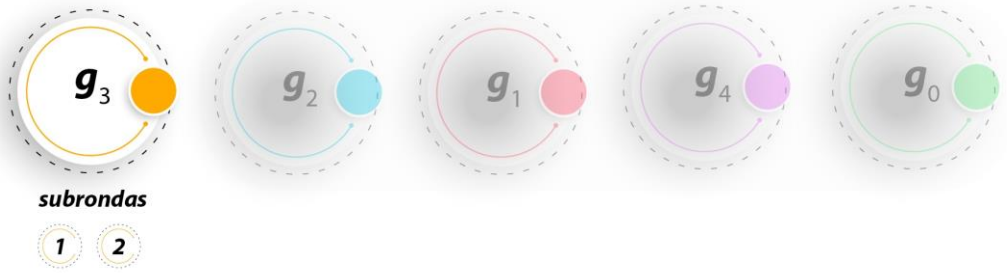


Fig. 18. Ronda correspondiente al grupo g_3.

Fuente: Elaboración propia.

Primera subronda del primer grupo (g_3)

FASDLNOGC$_1$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_1$) y donde ***FASDLNOGC***$_{cg1sub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_1$).

La Fig. 19 corresponde a la subronda actual del grupo g_3.

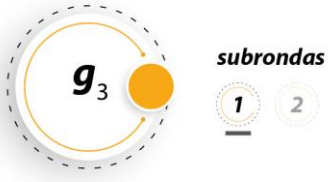


Fig. 19. Primera subronda para el grupo g_3.

Fuente: Elaboración propia.

La Tabla 74 contiene los datos de la primera subronda de asignación compatible, correspondiente al primer grupo (***g*₃**).

Tabla 74. Primera subronda de asignación compatible para el primer grupo (g_3)

Prioridad	Asignación	Rondas	Grupo
T(B;0.0159)	r_{13} al p_{31}	3	g_3
T(A;-0.0777)	r_{31} al p_{31}	5	g_3
T(EA;0.0000)	r_{33} al p_{31}	6	g_3
T(M;-0.0197)	r_{22} al p_{22}	3	g_3
T(B;0.0674)	r_{21} al p_{22}	5	g_3

Fuente: Elaboración propia.

FASDLNOGC*₁** es el primer elemento no ordenado compatible de la ***FASDLNOG para el grupo (***PGLGO*₁**) y donde ***FASDLNOGC*₅** es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO*₁**)

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la primer subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la segunda subronda.

Una vez asignados todos los recursos compatibles de la primer subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado incompatibilidad.

Segunda subronda del primer grupo (g_3)

FASDLNOGC*$_{cg1sub1+1}$** es el primer elemento no ordenado compatible de la ***FASDLNOG para el grupo (***PGLGO*₁**) y donde ***FASDLNOGC*$_{cg1sub1+cg1sub2}$** es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO*₁**).

La Fig. 20 corresponde a la subronda actual del grupo g_3.

Fig. 20. Segunda subronda para el grupo g_3.

Fuente: Elaboración propia.

La Tabla 75 contiene los datos de la segunda subronda de asignación compatible, correspondiente al primer grupo (***g*₃**).

Tabla 75. Segunda subronda de asignación compatible para el primer grupo (g_3)

Prioridad	Asignación	Rondas	Grupo
T(M;0.0040)	r_{31} al p_{22}	7	g_3
T(EA;0.0000)	r_{33} al p_{22}	8	g_3

Fuente: Elaboración propia.

FASDLNOGC$_{5+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_1$) y donde ***FASDLNOGC***$_{5+2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_1$).

Se continúa hasta completar todas las iteraciones de todos los grupos.

Se debe reiterar el procedimiento, pero para los siguientes grupos, atendiendo las asignaciones de recursos a procesos que no generan incompatibilidad, y luego retirando de las solicitudes de recursos las asignaciones ya hechas; para ser asignados a otros procesos que se encontraban en incompatiblidad.

La segunda asignación corresponde al segundo grupo de la ***PGLGO,*** y conforma la segunda ronda, luego, cada iteracción de asignaciones de recursos a procesos que sean no ordenadas y compatibles, es decir, que no hayan sido cargadas (no ordenadas) en la matriz (***FASDLNOG***) y que sean compatibles, es decir, que no generen conflictos en el acceso a un recurso, conformarán cada subronda.

Ronda de asignación del segundo grupo (g_2)

El primer paso es atender las asignaciones de recursos a procesos que no generan incompatibilidad en el grupo g_2, ver Fig. 21.

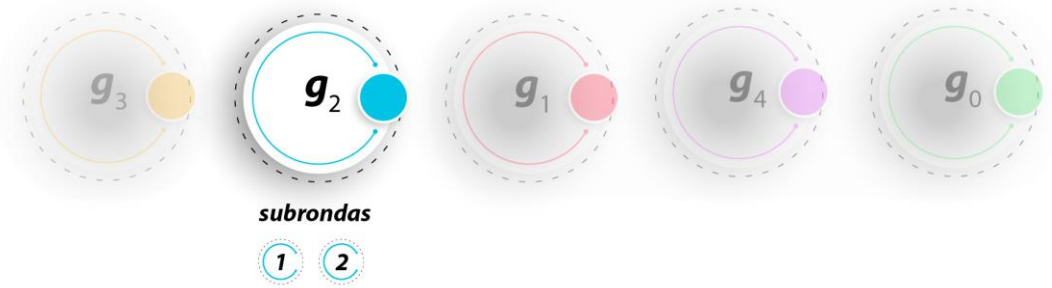


Fig. 21. Ronda correspondiente al grupo g_2.

Fuente: Elaboración propia.

Primer subronda del segundo grupo (g_2)

FASDLNOGC $_{cg1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$) y donde ***FASDLNOGC***$_{cg1+cg2sub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$).

La Fig. 22 corresponde a la subronda actual del grupo g_2.

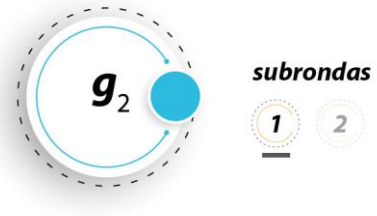


Fig. 22. Primera subronda para el grupo g_2.

Fuente: Elaboración propia.

La Tabla 76 contiene los datos de la primera subronda de asignación compatible, correspondiente al segundo grupo (***g***$_2$).

Tabla 76. Primera subronda de asignación compatible para el segundo grupo (g_2)

Prioridad	Asignación	Rondas	Grupo
T(M;-0.0721)	r_{21} al p_{12}	4	g_2
T(B;0.0082)	r_{11} al p_{12}	4	g_2
T(EA;0.0000)	r_{33} al p_{12}	5	g_2
T(M;-0.0467)	r_{22} al p_{12}	5	g_2
T(MA;-0.0133)	r_{12} al p_{12}	6	g_2
T(M;0.0638)	r_{31} al p_{12}	6	g_2
T(EB;0.0668)	r_{23} al p_{21}	3	g_2
T(B;-0.0337)	r_{13} al p_{21}	4	g_2

Fuente: Elaboración propia.

FASDLNOGC$_{7+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$) y donde ***FASDLNOGC***$_{7+8}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la primer subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la segunda subronda.

Una vez asignados todos los recursos compatibles de la primer subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado incompatibilidad.

Segunda subronda del segundo grupo (g_2)

FASDLNOGC$_{cg1+cg2sub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$) y donde ***FASDLNOGC***$_{cg1+cg2sub1+cg2sub2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$).

La Fig. 23 corresponde a la subronda actual del grupo g_2.

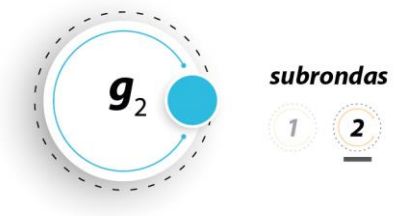


Fig. 23. Segunda subronda para el grupo g_2.

Fuente: Elaboración propia.

La Tabla 77 contiene los datos de la primera subronda de asignación compatible, correspondiente al segundo grupo (***g*₂**).

Tabla 77. Segunda subronda de asignación compatible para el primer grupo (g_2)

Prioridad	Asignación	Rondas	Grupos
T(M;0.0810)	r_{31} al p_{21}	3	g_2
T(B;0.0275)	r_{22} al p_{21}	6	g_2
T(EA;0.0000)	r_{33} al p_{21}	7	g_2
T(MA;-0.0348)	r_{12} al p_{21}	7	g_2

Fuente: Elaboración propia.

FASDLNOGC$_{7+8+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$) y donde ***FASDLNOGC***$_{7+8+4}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_2$).

Se continúa hasta completar todas las iteraciones de todos los grupos.

Se debe reiterar el procedimiento, pero para los siguientes grupos, atendiendo las asignaciones de recursos a procesos que no generan incompatibilidad, y luego retirando de las solicitudes de recursos las asignaciones ya hechas; para ser asignados a otros procesos que se encontraban en incompatiblidad.

La tercera asignación corresponde al tercer grupo de la ***PGLGO,*** y conforma la tercera ronda, luego, cada iteracción de asignaciones de recursos a procesos que sean no ordenadas y compatibles, es decir, que no hayan sido cargadas (no ordenadas) en la matriz (***FASDLNOG***) y que sean compatibles, es decir, que no generen conflictos en el acceso a un recurso, conformarán

cada subronda.

Ronda de asignación del tercer grupo (g_1)

El primer paso es atender las asignaciones de recursos a procesos que no generan incompatibilidad en el grupo g_1, ver Fig. 24.

Fig. 24. Ronda correspondiente al grupo g_1.

Fuente: Elaboración propia.

Primer subronda del tercer grupo (g_1)

FASDLNOGC $_{cg1+cg2+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$) y donde ***FASDLNOGC***$_{cg1+cg2+cg3sub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$).

La Tabla 78 contiene los datos de la primera subronda de asignación compatible, correspondiente al segundo grupo (g_1).

Tabla 78. Primera subronda de asignación compatible para el tercer grupo (g_1)

Prioridad	Asignación	Rondas	Grupos
T(EA;-0.0326)	r_{12} al p_{37}	1	g_1
T(A;-0.0355)	r_{11} al p_{37}	1	g_1
T(M;0.0337)	r_{21} al p_{37}	1	g_1
T(M;-0.0676)	r_{32} al p_{37}	2	g_1
T(EA;0.0000)	r_{33} al p_{37}	4	g_1
T(B;-0.0591)	r_{23} al p_{11}	1	g_1
T(M;-0.0298)	r_{22} al p_{11}	2	g_1
T(EB;0.0000)	r_{24} al p_{11}	2	g_1

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$) y donde ***FASDLNOGC***$_{7+12+8}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$).

La Fig. 25 corresponde a la subronda actual del grupo g_1.

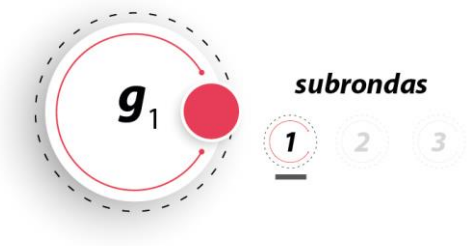


Fig. 25. Primera subronda para el grupo g_1.

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la primera subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la segunda subronda.

Una vez asignados todos los recursos compatibles de la primer subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado incompatibilidad.

Segunda subronda del tercer grupo (g$_1$)

FASDLNOGC$_{cg1+cg2+cg3sub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$) y donde ***FASDLNOGC***$_{cg1+cg2+cg3sub1+cg3sub2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$).

La Fig. 26 corresponde a la subronda actual del grupo g_1.

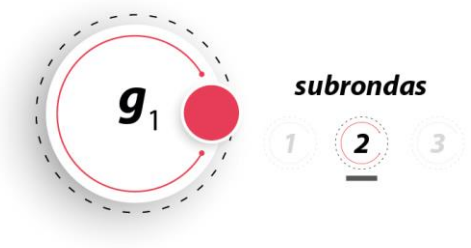


Fig. 26. Segunda subronda para el grupo g_1.

Fuente: Elaboración propia.

La Tabla 79 contiene los datos de la segunda subronda de asignación compatible, correspondiente al tercer grupo (g_1).

Tabla 79. Segunda subronda de asignación compatible para el tercer grupo (g_1)

Prioridad	Asignación	Rondas	Grupos
T(M;-0.0076)	r_{21} al p_{25}	2	g_1
T(M;0.0813)	r_{11} al p_{11}	2	g_1
T(MA;0.0431)	r_{12} al p_{11}	5	g_1

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+8+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$) y donde ***FASDLNOGC***$_{7+12+8+3}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la segunda subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la tercera subronda.

Una vez asignados todos los recursos compatibles de la segunda subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado incompatibilidad.

Tercera subronda del tercer grupo (g_1)

FASDLNOGC$_{cg1+cg2+cg3sub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$) y donde ***FASDLNOGC***$_{cg1+cg2+cg3sub1+cg3sub2+cg3sub3}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$).

La Fig. 27 corresponde a la subronda actual del grupo g_1.

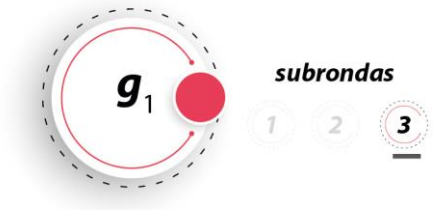


Fig. 27. Tercera subronda para el grupo g_1.

Fuente: Elaboración propia.

La Tabla 80 contiene los datos de la tercer subronda de asignación compatible, correspondiente al tercer grupo (***g*$_1$**).

Tabla 80. Tercera subronda de asignación compatible para el tercer grupo (g_1)

Prioridad	Asignación	Rondas	Grupos
T(B;-0.0106)	r_{21} al p_{11}	6	g_1

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+8+3+1}$ es el único elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_3$).

Se continúa hasta completar todas las iteraciones de todos los grupos.

Se debe reiterar el procedimiento, pero para los siguientes grupos, atendiendo las asignaciones de recursos a procesos que no generan incompatibilidad, y luego retirando de las solicitudes de recursos las asignaciones ya hechas; para ser asignados a otros procesos que se encontraban en incompatiblidad.

La cuarta asignación corresponde al cuarto grupo de la ***PGLGO,*** y conforma la cuarta ronda, luego, cada iteracción de asignaciones de recursos a procesos que sean no ordenadas y compatibles, es decir, que no hayan sido cargadas (no ordenadas) en la matriz (***FASDLNOG***) y que sean compatibles, es decir, que no generen conflictos en el acceso a un recurso, conformaran cada subronda.

<u>Ronda de asignación del cuarto grupo (g_4)</u>

El primer paso es atender las asignaciones de recursos a procesos que no generan incompatibilidad en el grupo g_4, ver Fig. 28.

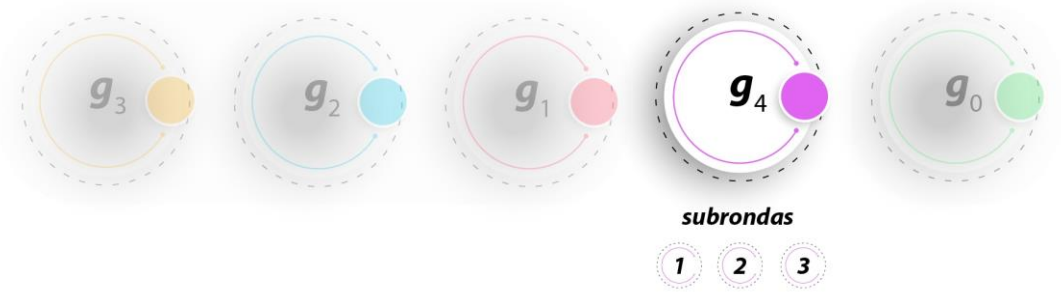


Fig. 28. Ronda correspondiente al grupo g_4.

Fuente: Elaboración propia.

Primera subronda del cuarto grupo (g_4)

FASDLNOGC$_{cg1+cg2+cg3+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$) y donde ***FASDLNOGC***$_{cg1+cg2+cg3+cg4sub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$).

La Fig. 29 corresponde a la subronda actual del grupo g_4.

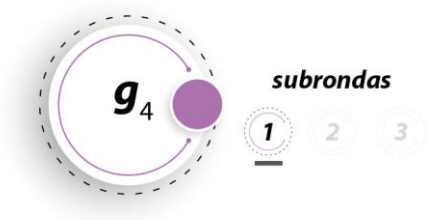


Fig. 29. Primera subronda para el grupo g_4.

Fuente: Elaboración propia.

La Tabla 81 contiene los datos de la primera subronda de asignación compatible, correspondiente al cuarto grupo (g_4).

Tabla 81. Primera subronda de asignación compatible para el cuarto grupo (g_4)

Prioridad	Asignación	Rondas	Grupos
T(M;0.0274)	r_{13} al p_{13}	1	g_4
T(A;-0.0392)	r_{31} al p_{13}	2	g_4
T(EA;0.0000)	r_{33} al p_{13}	3	g_4
T(M;-0.0133)	r_{11} al p_{13}	3	g_4
T(M;-0.0789)	r_{21} al p_{13}	3	g_4
T(B;0.0227)	r_{32} al p_{13}	3	g_4
T(MA;0.0255)	r_{12} al p_{13}	4	g_4
T(M;-0.0178)	r_{22} al p_{13}	4	g_4
T(EB;0.0000)	r_{24} al p_{23}	3	g_4
T(MB;0.0173)	r_{23} al p_{34}	2	g_4

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+12+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$) y donde ***FASDLNOGC***$_{7+12+12+10}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la primera subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la segunda subronda.

Una vez asignados todos los recursos compatibles de la primer subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado

incompatibilidad

Segunda subronda del cuarto grupo (g_4)

FASDLNOGC$_{cg1+cg2+cg3+cg4sub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$) y donde ***FASDLNOGC***$_{cg1+cg2+cg3+cg4sub1+cg4sub2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$).

La Fig. 30 corresponde a la subronda actual del grupo g_4.

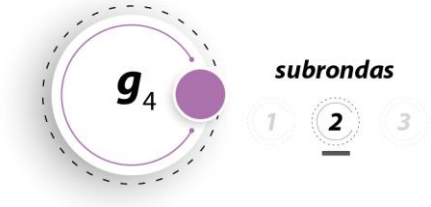


Fig. 30. Segunda subronda para el grupo g_4.

Fuente: Elaboración propia.

La Tabla 82 contiene los datos de la segunda subronda de asignación compatible, correspondiente al cuarto grupo (g_4).

Tabla 82. Segunda subronda de asignación compatible para el cuarto grupo (g_4)

Prioridad	Asignación	Rondas	Grupos
T(EA;0.0000)	r_{33} al p_{23}	1	g_4
T(MA;0.0381)	r_{12} al p_{23}	3	g_4
T(A;-0.0739)	r_{31} al p_{23}	4	g_4
T(B;-0.0762)	r_{32} al p_{23}	4	g_4
T(M;0.0423)	r_{22} al p_{34}	1	g_4
T(EB;0.0000)	r_{24} al p_{34}	1	g_4
T(M;-0.0347)	r_{13} al p_{34}	2	g_4

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+12+10+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$) y donde ***FASDLNOGC***$_{7+12+12+10+7}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de

asignaciones ya realizadas en la segunda subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la tercera subronda.

Una vez asignados todos los recursos compatibles de la segunda subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado incompatibilidad.

Tercera subronda del cuarto grupo (g4)

$\boldsymbol{FASDLNOGC}_{cg1+cg2+cg3+cg4sub1+cg4sub2+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_4$) y donde $\boldsymbol{FASDLNOGC}_{cg1+cg2+cg3+cg4sub1+cg4sub2+cg4sub3}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo ($\boldsymbol{PGLGO}_4$).

La Fig. 31 corresponde a la subronda actual del grupo g_4.

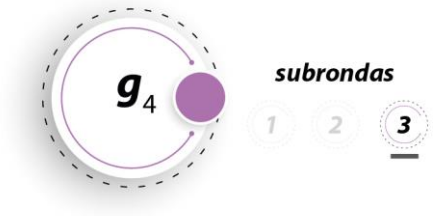


Fig. 31. Tercera subronda para el grupo g_4.

Fuente: Elaboración propia.

La Tabla 83 contiene los datos de la tercera subronda de asignación compatible, correspondiente al cuarto grupo ($\boldsymbol{g}_4$).

Tabla 83. Tercer subronda de asignación compatible para el cuarto grupo (g_4)

Prioridad	Asignación	Rondas	Grupos
T(A;-0.0028)	r_{31} al p_{34}	1	g_4
T(M;-0.0334)	r_{32} al p_{34}	1	g_4
T(EA;0.0000)	r_{33} al p_{34}	2	g_4
T(MA;0.0758)	r_{12} al p_{34}	2	g_4

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+12+10+7+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$) y donde ***FASDLNOGC***$_{7+12+12+10+7+4}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_4$).

Se continúa hasta completar todas las iteraciones del último grupo, conformado por los procesos que no pertenecían a ningún grupo, los cuales fueron asignados al **grupo g_0.**

Se debe reiterar el procedimiento, pero para el último grupo, atendiendo las asignaciones de recursos a procesos que no generan incompatibilidad, y luego retirando de las solicitudes de recursos las asignaciones ya hechas; para ser asignados a otros procesos que se encontraban en incompatiblidad.

La quinta asignación corresponde al quinto grupo de la ***PGLGO,*** y conforma la quinta ronda, luego, cada iteración de asignaciones de recursos a procesos que sean no ordenadas y compatibles, es decir, que no hayan sido cargadas (no ordenadas) en la matriz (***FASDLNOG***) y que sean compatibles, es decir, que no generen conflictos en el acceso a un recurso, conformaran cada subronda.

<u>Ronda de asignación del quinto grupo (g_0)</u>

El primer paso es atender las asignaciones de recursos a procesos que no generan incompatibilidad en el grupo g_0, ver Fig. 32.

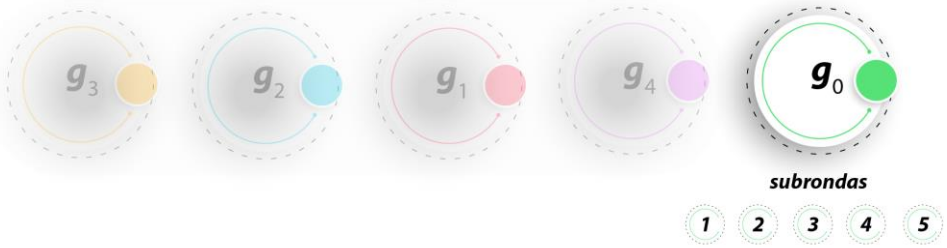


Fig. 32. Ronda correspondiente al grupo g_0.

Fuente: Elaboración propia.

Primera subronda del quinto grupo (g_0)

FASDLNOGC$_{cg1+cg2+cg3+cg4+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC***$_{cg1+cg2+cg3+cg4+cg5sub1}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

La Fig. 33 corresponde a la subronda actual del grupo g_0.

Fig. 33. Primera subronda para el grupo g_0.

Fuente: Elaboración propia.

La Tabla 84 contiene los datos de la primera subronda de asignación compatible, correspondiente al quinto grupo (g_0).

Tabla 84. Primera subronda de asignación compatible para el quinto grupo (g_0)

Prioridad	Asignación	Rondas	Grupos
T(EB;0.0000)	r_{24} al p_{35}	4	g_0
T(MB;-0.0604)	r_{32} al p_{35}	6	g_0
T(MB;0.0742)	r_{13} al p_{35}	7	g_0
T(MB;0.0580)	r_{22} al p_{35}	7	g_0
T(B;-0.0534)	r_{31} al p_{35}	9	g_0
T(MA;0.0734)	r_{33} al p_{35}	10	g_0
T(EA;0.0000)	r_{12} al p_{35}	12	g_0
T(EB;0.0489)	r_{23} al p_{32}	4	g_0
T(MB;0.0273)	r_{11} al p_{32}	5	g_0
T(MB;0.0682)	r_{21} al p_{33}	7	g_0

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+12+21+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC***$_{7+12+12+21+10}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la primer subronda; debe tenerse en cuenta que los recursos

asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la segunda subronda.

Una vez asignados todos los recursos compatibles de la primer subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado incompatibilidad.

Segunda subronda del quinto grupo (g_0)

FASDLNOGC $_{cg1+cg2+cg3+cg4+cg5sub1+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC*** $_{cg1+cg2+cg3+cg4+cg5sub1+cg5sub2}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

La
Tabla **34** corresponde a la subronda actual del grupo g_0.

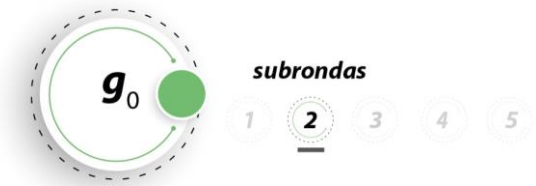


Fig. 34. Segunda subronda para el grupo g_0.

Fuente: Elaboración propia.

La Tabla 85 contiene los datos de la segunda subronda de asignación compatible, correspondiente al quinto grupo (g_0).

Tabla 85. Segunda subronda de asignación compatible para el quinto grupo (g_0)

Prioridad	Asignación	Rondas	Grupos
T(B;-0.0824)	r_{13} al p_{32}	5	g_0
T(EA;0.0000)	r_{12} al p_{32}	11	g_0
T(EB;0.0230)	r_{23} al p_{33}	5	g_0
T(MB;0.0631)	r_{11} al p_{33}	7	g_0
T(MB;-0.0553)	r_{32} al p_{33}	7	g_0
T(MB;0.0078)	r_{22} al p_{33}	8	g_0
T(MA;-0.0117)	r_{33} al p_{33}	9	g_0
T(EB;0.0000)	r_{24} al p_{36}	5	g_0
T(M;-0.0299)	r_{31} al p_{36}	8	g_0
T(MB;0.0191)	r_{21} al p_{36}	8	g_0

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+12+21+10+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC***$_{7+12+12+21+10+10}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la segunda subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la tercera subronda.

Una vez asignados todos los recursos compatibles de la segunda subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado incompatibilidad.

Tercera subronda del quinto grupo (g_0)

FASDLNOGC $_{cg1+cg2+cg3+cg4+cg5sub1+cg5sub2+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC*** $_{cg1+cg2+cg3+cg4+cg5sub1+cg5sub2+cg5sub3}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

La Tabla 35 corresponde a la subronda actual del grupo g_0.

Fig. 35. Tercera subronda para el grupo g_0.

Fuente: Elaboración propia.

La Tabla 86 contiene los datos de la tercera subronda de asignación compatible, correspondiente al quinto grupo (g_0).

Tabla 86. Tercer subronda de asignación compatible para el quinto grupo (g_0)

Prioridad	Asignación	Rondas	Grupos
T(EA;-0.0210)	r_{12} al p_{33}	8	g_0
T(MB;0.0110)	r_{13} al p_{33}	8	g_0
T(MB;-0.0591)	r_{32} al p_{36}	5	g_0
T(MB;0.0443)	r_{11} al p_{36}	6	g_0
T(EB;0.0000)	r_{22} al p_{36}	9	g_0
T(A;-0.0196)	r_{33} al p_{36}	11	g_0
T(EB;0.0032)	r_{23} al p_{24}	6	g_0
T(EB;0.0000)	r_{24} al p_{24}	6	g_0

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+12+21+10+10+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC***$_{7+12+12+21+10+10+8}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la tercera subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la **cuarta** subronda.

Una vez asignados todos los recursos compatibles de la tercer subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado incompatibilidad.

Cuarta subronda del quinto grupo (g_0)

FASDLNOGC$_{cg1+cg2+cg3+cg4+cg5sub1+cg5sub2+cg5sub3+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC***$_{cg1+cg2+cg3+cg4+cg5sub1+cg5sub2+cg5sub3+cg5sub4}$ es el último elemento no ordenado compatible de la

FASDLNOG para el grupo (***PGLGO***$_5$).

La Tabla 36 corresponde a la subronda actual del grupo g_0.

Fig. 36. Cuarta subronda para el grupo g_0.

Fuente: Elaboración propia.

La Tabla 87 contiene los datos de la cuarta subronda de asignación compatible, correspondiente al quinto grupo (g_0).

Tabla 87. Cuarta subronda de asignación compatible para el quinto grupo (g_0)

Prioridad	Asignación	Rondas	Grupos
T(MB;0.0815)	r_{13} al p_{36}	6	g_0
T(EA;0.0000)	r_{12} al p_{36}	9	g_0
T(MB;-0.0035)	r_{11} al p_{24}	8	g_0

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+12+21+10+10+8+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC***$_{7+12+12+21+10+10+8+3}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas en la cuarta subronda; debe tenerse en cuenta que los recursos asignados estarán disponibles una vez liberados por los procesos y, por lo tanto, pueden asignarse a otros procesos en la **quinta** subronda.

Una vez asignados todos los recursos compatibles de la cuarta subronda, se pasará a la siguiente subronda, dentro de la misma ronda (**grupo**), contemplando aquellas asignaciones de recursos a procesos que no fueron atendidos en subrondas anteriores, por haber generado

incompatibilidad.

Quinta subronda del quinto grupo (g_0)

FASDLNOGC $_{cg1+cg2+cg3+cg4+cg5sub1+cg5sub2+cg5sub3+cg5sub4+1}$ es el primer elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$) y donde ***FASDLNOGC*** $_{cg1+cg2+cg3+cg4+cg5sub1+cg5sub2+cg5sub3+cg5sub4+cg5sub5}$ es el último elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

La Tabla 37 corresponde a la subronda actual del grupo g_0.

Fig. 37. Quinta subronda para el grupo g_0.

Fuente: Elaboración propia.

La Tabla 88 contiene los datos de la quinta subronda de asignación compatible correspondiente al quinto grupo (g_0).

Tabla 88. Quinta subronda de asignación compatible para el quinto grupo (g_0)

Prioridad	Asignación	Rondas	Grupos
T(EA;0.0000)	r_{12} al p_{24}	10	g_0

Fuente: Elaboración propia.

FASDLNOGC$_{7+12+12+21+10+10+8+3+1}$ es el único elemento no ordenado compatible de la ***FASDLNOG*** para el grupo (***PGLGO***$_5$).

La Tabla 89 muestra el orden final de todas las asignaciones de recursos para cada proceso ordenada por grupo compatible (***FASDLNOGC***).

Tabla 89. Orden Final de asignación en la ***FASDLNOGC***

Prioridad	Recursos	Rondas	Grupo	Subronda
T(B;0.0159)	r_{13} al p_{31}	3	g_3	1
T(A;-0.0777)	r_{31} al p_{31}	5	g_3	1
T(EA;0.0000)	r_{33} al p_{31}	6	g_3	1
T(M;-0.0197)	r_{22} al p_{22}	3	g_3	1
T(B;0.0674)	r_{21} al p_{22}	5	g_3	1
T(M;0.0040)	r_{31} al p_{22}	7	g_3	2
T(EA;0.0000)	r_{33} al p_{22}	8	g_3	2
T(M;-0.0721)	r_{21} al p_{12}	4	g_2	1
T(B;0.0082)	r_{11} al p_{12}	4	g_2	1
T(EA;0.0000)	r_{33} al p_{12}	5	g_2	1
T(M;-0.0467)	r_{22} al p_{12}	5	g_2	1
T(MA;-0.0133)	r_{12} al p_{12}	6	g_2	1
T(M;0.0638)	r_{31} al p_{12}	6	g_2	1
T(EB;0.0668)	r_{23} al p_{21}	3	g_2	1
T(B;-0.0337)	r_{13} al p_{21}	4	g_2	1
T(M;0.0810)	r_{31} al p_{21}	3	g_2	2
T(B;0.0275)	r_{22} al p_{21}	6	g_2	2
T(EA;0.0000)	r_{33} al p_{21}	7	g_2	2
T(MA;-0.0348)	r_{12} al p_{21}	7	g_2	2
T(EA;-0.0326)	r_{12} al p_{37}	1	g_1	1
T(A;-0.0355)	r_{11} al p_{37}	1	g_1	1
T(M;0.0337)	r_{21} al p_{37}	1	g_1	1
T(M;-0.0676)	r_{32} al p_{37}	2	g_1	1
T(EA;0.0000)	r_{33} al p_{37}	4	g_1	1
T(B;-0.0591)	r_{23} al p_{11}	1	g_1	1
T(M;-0.0298)	r_{22} al p_{11}	2	g_1	1
T(EB;0.0000)	r_{24} al p_{11}	2	g_1	1
T(M;-0.0076)	r_{21} al p_{25}	2	g_1	2
T(M;0.0813)	r_{11} al p_{11}	2	g_1	2
T(MA;0.0431)	r_{12} al p_{11}	5	g_1	2
T(B;-0.0106)	r_{21} al p_{11}	6	g_1	3
T(M;0.0274)	r_{13} al p_{13}	1	g_4	1
T(A;-0.0392)	r_{31} al p_{13}	2	g_4	1
T(EA;0.0000)	r_{33} al p_{13}	3	g_4	1
T(M;-0.0133)	r_{11} al p_{13}	3	g_4	1
T(M;-0.0789)	r_{21} al p_{13}	3	g_4	1
T(B;0.0227)	r_{32} al p_{13}	3	g_4	1
T(MA;0.0255)	r_{12} al p_{13}	4	g_4	1
T(M;-0.0178)	r_{22} al p_{13}	4	g_4	1
T(EB;0.0000)	r_{24} al p_{23}	3	g_4	1
T(MB;0.0173)	r_{23} al p_{34}	2	g_4	1
T(EA;0.0000)	r_{33} al p_{23}	1	g_4	2
T(MA;0.0381)	r_{12} al p_{23}	3	g_4	2
T(A;-0.0739)	r_{31} al p_{23}	4	g_4	2
T(B;-0.0762)	r_{32} al p_{23}	4	g_4	2
T(M;0.0423)	r_{22} al p_{34}	1	g_4	2
T(EB;0.0000)	r_{24} al p_{34}	1	g_4	2
T(M;-0.0347)	r_{13} al p_{34}	2	g_4	2
T(A;-0.0028)	r_{31} al p_{34}	1	g_4	3
T(M;-0.0334)	r_{32} al p_{34}	1	g_4	3
T(EA;0.0000)	r_{33} al p_{34}	2	g_4	3
T(MA;0.0758)	r_{12} al p_{34}	2	g_4	3
T(EB;0.0000)	r_{24} al p_{35}	4	g_0	1
T(MB;-0.0604)	r_{32} al p_{35}	6	g_0	1
T(MB;0.0742)	r_{13} al p_{35}	7	g_0	1
T(MB;0.0580)	r_{22} al p_{35}	7	g_0	1
T(B;-0.0534)	r_{31} al p_{35}	9	g_0	1
T(MA;0.0734)	r_{33} al p_{35}	10	g_0	1
T(EA;0.0000)	r_{12} al p_{35}	12	g_0	1

Prioridad	Recursos	Rondas	Grupo	Subronda
T(EB;0.0489)	r_{23} al p_{32}	4	g_0	1
T(MB;0.0273)	r_{11} al p_{32}	5	g_0	1
T(MB;0.0682)	r_{21} al p_{33}	7	g_0	1
T(B;-0.0824)	r_{13} al p_{32}	5	g_0	2
T(EA;0.0000)	r_{12} al p_{32}	11	g_0	2
T(EB;0.0230)	r_{23} al p_{33}	5	g_0	2
T(MB;0.0631)	r_{11} al p_{33}	7	g_0	2
T(MB;-0.0553)	r_{32} al p_{33}	7	g_0	2
T(MB;0.0078)	r_{22} al p_{33}	8	g_0	2
T(MA;-0.0117)	r_{33} al p_{33}	9	g_0	2
T(EB;0.0000)	r_{24} al p_{36}	5	g_0	2
T(M;-0.0299)	r_{31} al p_{36}	8	g_0	2
T(MB;0.0191)	r_{21} al p_{36}	8	g_0	2
T(EA;-0.0210)	r_{12} al p_{33}	8	g_0	3
T(MB;0.0110)	r_{13} al p_{33}	8	g_0	3
T(MB;-0.0591)	r_{32} al p_{36}	5	g_0	3
T(MB;0.0443)	r_{11} al p_{36}	6	g_0	3
T(EB;0.0000)	r_{22} al p_{36}	9	g_0	3
T(A;-0.0196)	r_{33} al p_{36}	11	g_0	3
T(EB;0.0032)	r_{23} al p_{24}	6	g_0	3
T(EB;0.0000)	r_{24} al p_{24}	6	g_0	3
T(MB;0.0815)	r_{13} al p_{36}	6	g_0	4
T(EA;0.0000)	r_{12} al p_{36}	9	g_0	4
T(MB;-0.0035)	r_{11} al p_{24}	8	g_0	4
T(EA;0.0000)	r_{12} al p_{24}	10	g_0	5

Fuente: Elaboración propia.

De esta manera, se respondió a todas las solicitudes de recursos de todos los procesos de cada grupo, considerando la exclusión mutua y las prioridades de los procesos, las prioridades nodales y las prioridades finales, teniendo en cuenta los estrictos requisitos de consenso establecidos para este escenario.

4.6. Evaluación

El modelo propuesto logra establecer un consenso que permite a los grupos de procesos acceder a todos sus recursos de manera secuencial y que éstos no pueden ser removidos hasta que el mismo grupo de procesos que los mantiene los libera. El orden de asignación será determinado por la prioridad promedio general de todas las asignaciones de cada grupo. El sistema distribuido regula y actualiza constantemente el estado local de cada nodo, las decisiones de acceso a los recursos modifican estos estados por lo que debe ser reajustado repetidamente, garantizando la exclusión mutua y reordenando nuevas prioridades. El método debe repetirse siempre que haya

grupos de procesos que requieran recursos compartidos.

En La Red Martínez (2017), en el escenario general, las asignaciones de recursos a los diferentes procesos se realizan en varias rondas de asignación, donde un proceso puede tener varias asignaciones de recursos en las diferentes rondas.

Este capítulo considera el promedio global de prioridades que cada grupo de procesos tiene sobre todos los recursos de todas sus asignaciones en las diferentes rondas ***PGLGO***, pero para la asignación global final del grupo, establece subrondas compatibles de asignación, dado que en un mismo grupo puede haber procesos que requieran un mismo recurso, por lo tanto, los procesos de mayor prioridad serán asignados en una primer subronda y los que generan incompatiblidad se asignarán en una siguiente subronda.

Es decir, la elección del grupo de procesos al que se le otorgarán recursos, se establece con el promedio global de prioridades en todas las asignaciones, pero el orden en el que se deben realizar esas asignaciones, se establece por medio de subrondas compatibles de asignación.

La representación de los $\boldsymbol{r}_{ij}$, indican los recursos (cuyo primer subíndice representa al nodo donde se encuentra y el segundo subíndice al propio número de recurso) que se les asignan a los procesos $\boldsymbol{p}_{ij}$ (cuyo primer subíndice representa al nodo donde se encuentra y el segundo subíndice al propio número de proceso) en cada ronda. Aunque para los recursos/procesos los subíndices son iguales, no necesariamente sean los mismos recursos/procesos, sino que pueden representar a distintos recursos/procesos que se asigna varias veces en las distintas rondas, pero siempre al mismo grupo $\boldsymbol{g}_{\boldsymbol{h}}$.

Para las cardinalidades, "*cg*" corresponde a la identificación del grupo, y "*sub*" a la subronda de asignación que pertenece. Se debe continuar con el mismo procedimiento para los

siguientes grupos en la ***PGLGO***, ver Fig. 38.

El modelo propuesto incluye, como caso particular, un método que consiste en considerar la prioridad global de los grupos de procesos, en lugar de un grupo de variables de estado de cada nodo.

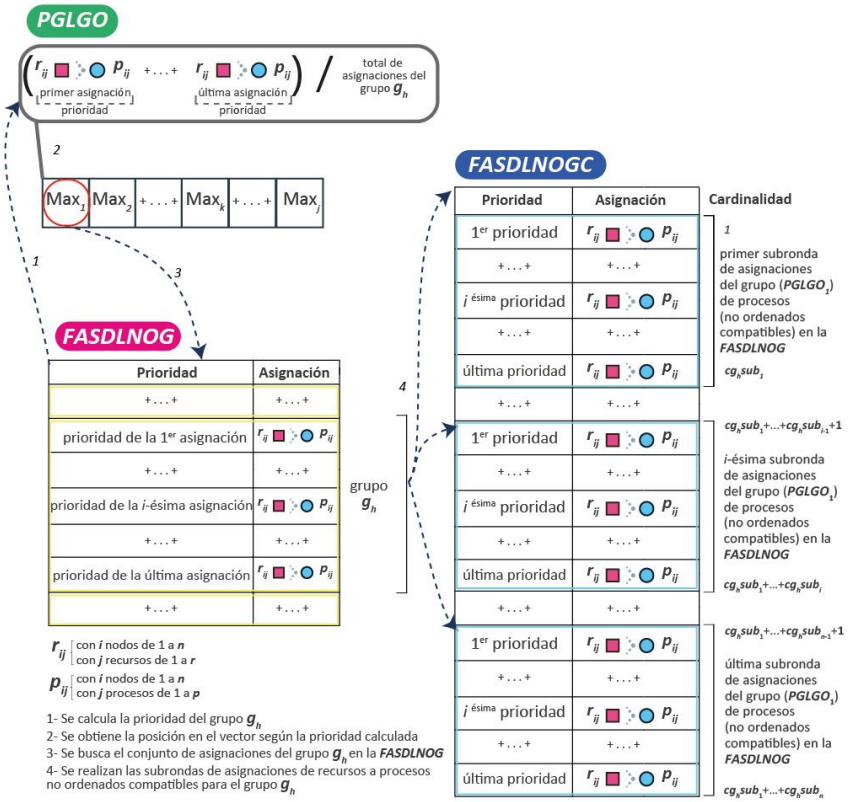


Fig. 38. Secuencia para obtener la ***FASDLNOGC***.

Fuente: Elaboración propia.

4.7. Discusiones y comentarios

No hay conflicto en cuanto a que varios procesos requieran el mismo recurso, dado que los recursos son asignados por subrondas de asignaciones compatibles.

Como la ejecución de los procesos se realiza en diferentes procesadores utilizando todos sus recursos, no hay conflicto en la ejecución de varios procesos en el mismo procesador.

No se tiene en cuenta la cantidad de tiempo que cada proceso utilizará en un procesador de un nodo en particular, ni la cantidad de tiempo en que cada recurso será asignado a un proceso de cada grupo en particular.

Los procesos independientes, es decir, que no forman grupos, son agrupados y considerados como el último grupo, su cardinalidad es el número de procesos que lo conforman y su prioridad cero, favoreciendo a los procesos que pertenecen a grupos.

El contenido de este capítulo sirvió de base para la siguiente publicación:

- Agostini F., La Red Martínez D. L., Fornerón Martínez J. T. (2019). Nuevo Operador de Agregación para Grupos de Procesos. Conferencias IADIS (International Association for Development of the Information Society). *Ibero-Americanas WWW/Internet y Computación Aplicada*. ISBN N° 978-989-8533-96-8, pp. 223-230. Lisboa, Portugal. (PUBLICADO)

Capítulo V - Modelos de decisión y operadores de agregación

5.1. Introducción a los modelos de decisión

El Modelo de Decisión permite definir objetivos claros, se basa en la recolección de datos y su relevancia, estudia las posibles alternativas y evalúa sus consecuencias. La tarea consiste en seleccionar la mejor alternativa para lograr la mejor solución.

El objetivo principal de las redes de computadoras es compartir recursos, de tal manera que todos los programas, equipos y datos se encuentren disponibles para quien lo solicite sin importar su ubicación. Para ello se tendrá en cuenta el acceso a recursos compartidos en la modalidad de exclusión mutua distribuida y mediante requisitos de sincronización. Se debe decidir, en función a la demanda de recursos por parte de los procesos, cuáles serán las prioridades para asignar cada uno. Para satisfacer esto, el permiso de acceso a los recursos compartidos de un nodo no sólo dependerá de si los nodos los utilizan o no, sino también del valor de agregación de las preferencias (prioridades) de los distintos nodos en cuanto a la concesión de acceso a los recursos compartidos (alternativas). Los nodos deberán expresar sus prioridades para la asignación de los diferentes recursos compartidos de acuerdo con las necesidades de recursos de cada proceso.

Las premisas, estructuras de datos y el operador mencionado en La Red Martínez (2017) y en Agostini et al. (2019a, 2019b y 2019c) son aplicadas para resolver el escenario que se describe a continuación. Se opera con lógica difusa, etiquetas lingüísticas y estructuras del tipo 2-tupla, basado en Dutta et al. (2019).

5.2. Escenario propuesto

En la Fig. 39 se muestra una macroimagen del sistema, en la cual, también se consideran transiciones de estados de acuerdo a la actualización iterativa de los requerimientos del sistema,

en donde se encuentran los distintos nodos, cuyo estado puede ser activo, cuando interactúa con los demás nodos aunque no envíe información ni participe del uso de recursos o la ejecución de procesos con recursos externos, activo compartido, cuando interactúa con los demás nodos, enviando y recibiendo información, creando procesos que utilizan recursos de otros nodos y compartiendo recursos propios con nodos externos, inactivo, cuando no se tiene información del estado real del nodo (puede estar sobrecargado y no puede realizar el intercambio de información, o bien, pudo haberse caído el enlace que lo conecta con los demás nodos), por último, existe un nodo central, que a través de un *Runtime* gestiona la información de todos los nodos que participen en las asignaciones de recursos.

Fig. 39. Macroimagen del estado de los nodos en un Sistema Distribuido.

Fuente: Elaboración propia.

El nodo central, es el encargado de recopilar la información de todos los nodos, aplicar el proceso de agregación y obtener la lista de asignaciones de recursos a procesos. Un macrociclo contempla la iteración de este proceso, en el cual el modelo de decisión es una etapa que se repite n veces. En este macrociclo hay un paso previo de recopilación de información del estado de red, nodos activos y conectados, cuántos están entregando información, un sondeo previo. Luego de la recepción se continúa con el proceso de evaluación y la emisión de los resultados. Se vuelve a aplicar el loop y el universo va cambiando en cada iteración.

5.3. Operador de agregación utilizado por el Modelo de Decisión

En La Red Martínez (2017), las características de los operadores de agregación descritos permiten considerar que el método propuesto pertenece a la familia de operadores de agregación *Neat-OWA*. La propuesta desarrollada en este trabajo consiste en generar un modelo de decisión y su correspondiente operador de agregación para la gestión de grupos de procesos que comparten recursos, realizando modificaciones de los operadores de la familia *OWA* mencionados.

Este método es una manera rápida y sencilla para realizar una elección en un problema de decisión; el proceso para realizar una decisión Multicriterio con este operador es el siguiente:

a) Identificar el propósito del problema;

b) Identificar las alternativas;

c) Listar los criterios que van a ser tenidos en cuenta a elegir la mejor alternativa;

d) Asignar una ponderación para cada uno de los criterios;

e) Cálculo de las prioridades o preferencias de los procesos teniendo en cuenta el estado del nodo;

f) Cálculo de las prioridades o preferencias de los procesos para acceder a los recursos compartidos disponibles y determinación del orden en que se asignarán los recursos y a qué proceso será asignado cada recurso.

a **Identificar el problema**

Este paso consiste en identificar qué es lo que se desea alcanzar. Para este trabajo lo que se intenta resolver son las solicitudes de recursos por parte de los procesos que se encuentran en nodos distribuidos. Para ello se utilizará la expresión de los valores calculados en términos de 2-

tupla utilizando un conjunto de etiquetas lingüísticas. El conjunto de etiquetas utilizado puede variar en cada nodo, es decir, que el conjunto es independiente del nodo central, y es éste el encargado de realizar la traslación simbólica, proceso que consiste en convertir los conjuntos de etiquetas recibidos en el nodo central, en el conjunto que este último trabaja, el cual debe ser el de mayor número de etiquetas. De igual manera, esa traslación simbólica se realiza cuando el nodo central devuelve la información final de prioridades a cada nodo.

b **Identificar las alternativas**

Son las opciones que podrían ayudar a la consecución del propósito que se trazó en el primer paso. En cada nodo se define una interfaz entre las aplicaciones y el sistema operativo, que a través de un *Runtime* (software en tiempo de ejecución complementario al sistema operativo) incluido en esa interfaz, gestiona los procesos y recursos compartidos y define el escenario correspondiente. Además, los *Runtime* interactúan entre sí para intercambiar información y en uno de los nodos hay un *Runtime* coordinador global que evalúa y ejecuta el modelo de decisión y el operador de agregación correspondiente.

c **Listar los criterios que van a ser tenidos en cuenta para elegir la mejor alternativa**

Se contemplará el cálculo de la carga actual de los nodos. Para obtener un indicador de la carga computacional actual de cada nodo, se pueden adoptar diferentes criterios; en esta propuesta los criterios serán el porcentaje de uso de la CPU, el porcentaje de uso de la memoria y el porcentaje de uso de la operación de entrada/salida. La carga computacional de cada nodo, el número de criterios para determinar la carga de los nodos, los criterios que se aplican y el cálculo de la carga computacional de cada nodo, son los mencionados en La Red Martínez (2017).

d **Asignar una ponderación para cada uno de los criterios**

Consiste en indicar qué tan importante es para el decisor el criterio en cuestión. Una representación lingüística podría constituirse mediante la siguiente Tabla 90:

Tabla 90. Escala de ponderación de criterios

Escala	Significado
1	Muy poco importante
2	Poco importante
3	Importancia media
4	Algo importante
5	Muy importante

Fuente: Elaboración propia.

Las categorías de carga computacional actuales de cada nodo, el número de categorías para determinar la carga de los nodos, las categorías que se aplican, los vectores de pesos asociados a las categorías de carga computacional actuales de cada nodo pueden ser representadas con la escala anteriormente mencionada.

Para cada alternativa podría darse con el siguiente cálculo:

$$S_j = \sum_i w_i r_{ij}$$

Donde: r_{ij} indica el valor de la alternativa j para el criterio i; w_i indica la ponderación de cada criterio; S_j indica la valoración para cada alternativa j.

e **Cálculo de las prioridades o preferencias de los procesos teniendo en cuenta el estado del nodo**

Estas prioridades se calculan en cada nodo para cada solicitud de recursos originada en cada proceso; el cálculo considera el vector de pesos correspondiente según la carga actual del nodo y el vector de los valores concedidos por el nodo según los criterios de evaluación de la solicitud.

Los vectores de valoración que se aplicarán para cada solicitud de un recurso por un proceso, de acuerdo con los criterios establecidos para la determinación de la prioridad que en cada caso y momento fijará el nodo en el que se produce la solicitud. Los criterios podrían determinarse de acuerdo con la importancia de cada uno, por ejemplo, Tabla 91:

Tabla 91. Significado de cada criterio en una escala de 1 a 9

Escala	Significado
1	Extra bajo
2	Muy bajo
3	Bajo
4	Poco bajo
5	Medio
6	Medio alto
7	Alto
8	Muy alto
9	Extra alto

Fuente: Elaboración propia.

Los valores son representados en el formato de 2-tupla, considerando las etiquetas lingüísticas propuestas anteriormente. Por lo tanto, el valor de cada criterio deberá compararse con el valor medio que tendrá cada etiqueta, la diferencia mínima de esa comparación será la etiqueta apropiada, algunos de estos conceptos pueden verse en Dutta et al. (2019).

El primer elemento de la 2-tupla será el valor lingüístico de esa etiqueta. El segundo elemento será la diferencia entre el valor de los criterios buscados y el valor medio de la etiqueta seleccionada.

dm = la diferencia mínima entre las diferencias de cp_m (criterios) y el valor más representativo de cada etiqueta lingüística.

En resumen, la prioridad nodal (que debe calcularse en el nodo en el que se produce la solicitud) de un proceso de acceso a un recurso determinado (que puede ser en cualquier nodo) se calcula mediante el producto escalar de los vectores mencionados: prioridad nodal $(r_{ij}\ p_{kl}) = \Sigma\ w_{om}$ * T(etiqueta$_m$; dm_m) = T(etiqueta$_n$; dm_n) = $\boldsymbol{TPN_{ijkl}}$ (Tupla de Prioridad Nodal) con la o indicando

el vector de pesos según la carga del nodo, manteniendo todos los demás subíndices los significados explicados anteriormente. Con *m* y *n* indicando la etiqueta lingüística correspondiente dentro del conjunto adoptado definido anteriormente. Esta prioridad nodal debe transformarse en el formato de 2-tuplas, teniendo en cuenta las etiquetas lingüísticas ya mencionadas. Por lo tanto, será necesario comparar cada valor de prioridad nodal con el valor medio de cada etiqueta, la diferencia mínima de estas comparaciones indicará la etiqueta correspondiente.

- **Cálculo de las prioridades o preferencias de los procesos para acceder a los recursos compartidos disponibles y determinación del orden en que se asignarán los recursos y a qué proceso será asignado cada recurso**

Para calcular las prioridades finales, se colocan las prioridades o preferencias nodales calculadas en la etapa anterior; en esta tabla cada fila contiene la información de las prioridades nodales de los diferentes procesos para acceder a un determinado recurso.

A continuación, es necesario calcular el vector de pesos finales que se utilizará en el proceso de agregación para determinar el orden o la prioridad de acceso a los recursos.

El siguiente paso es normalizar los pesos recién obtenidos dividiendo cada uno de ellos por la suma de todos. Así se obtiene un vector de peso normalizado (en el rango de 0 a 1 inclusive) y con la restricción de que la suma de los elementos del vector debe dar 1.

Las prioridades nodales tomadas fila por fila para cada recurso serán multiplicadas escalarmente por el vector de peso final normalizado. De esta manera es posible obtener las prioridades globales finales de acceso de cada proceso a cada recurso. El orden o la prioridad con que se asignarán los recursos y a qué proceso se asignará cada uno, estará determinado por el cálculo de la Prioridad global final ($r_{ij}\ p_{kl}$) = $\boldsymbol{TPN_{ijkl}}$ = $\boldsymbol{TPGF_{ijkl}}$ (Tupla de prioridad global final)

con el r_{ij} indicando el recurso *j* del nodo *i*, $\boldsymbol{TPN_{ijkl}}$ es el formato de 2-tuplas, *ij* indicando el recurso *j* del nodo *i*, *kl* el proceso *l* del nodo *k* y el producto de la prioridad global final del proceso para acceder a dicho recurso.

El mayor de estos productos realizados para los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que se le asignará a ese recurso, en relación con los demás recursos que también deberán asignarse. Esto es lo que se denominará Función de Asignación de Sistemas Distribuidos Lingüística (***FASDL***):

$\boldsymbol{FASDL}$ $(r_{ij}) = \Sigma\ \boldsymbol{TPGFN_{ijkl}}$ = prioridad de asignación de recursos r_{ij}.

Calculando el ***FASDL*** para todos los recursos se obtendrá un vector 2-tupla, y ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos.

Además, como ya se ha indicado, el mayor de los productos $\boldsymbol{TPGFN_{ijkl}}$ para cada recurso indicará el proceso al que se asignará el recurso.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

5.4. Modelo de decisión propuesto

El objetivo principal del modelo propuesto es considerar el entorno de ejecución distribuida de procesos, que podrán estar agrupados, con distintas exigencias de consenso respecto del acceso a recursos compartidos, para seleccionar el método de agregación más adecuado a cada escenario

posible, para generar la secuencia de asignación de los recursos a los procesos que los solicitan, respetando la exclusión mutua en el acceso a dichos recursos.

El modelo considera la información proveniente de la caracterización de los procesos (si los procesos pertenecen a grupos o son independientes, y si la aplicación a la que pertenecen requiere algún tipo de consenso respecto del acceso a los recursos), las prioridades nodales, estado de los nodos, recursos, procesos y del sistema para producir una solución global satisfactoria. Este método evalúa esta información, para determinar cuál es el escenario según la carga de trabajo proporcionada, y en base a ello elije el operador de agregación correspondiente (ver Fig. 40).

En el modelo de decisión propuesto se ejecuta el primer operador de agregación, que corresponde al escenario general (Función de Asignación en Sistemas Distribuidos Linguística, *FASDL*). Ver capítulo 2, correspondiente al escenario 1.

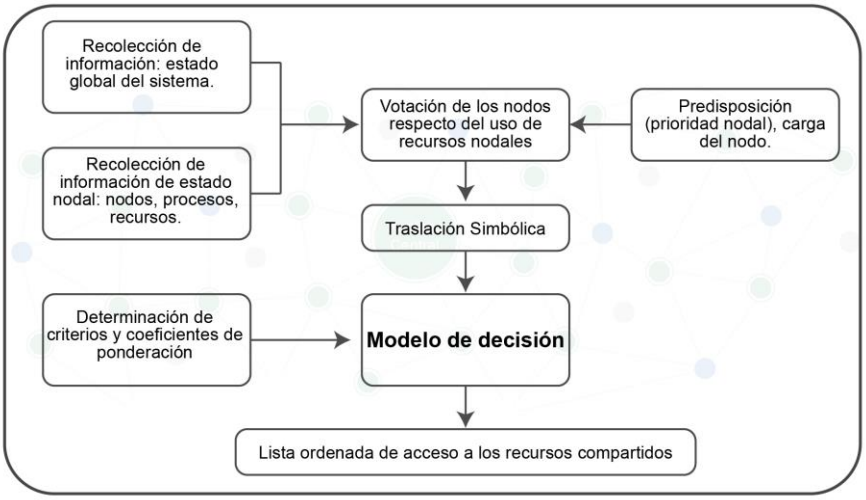


Fig. 40. Modelo global de decisión para acceso a recursos compartidos.

Fuente: Elaboración propia.

De acuerdo a la información ingresada, se podrá seguir con el escenario general o hacer uso de uno de los otros operadores para los siguientes escenarios, la Función de Asignación en Sistemas Distribuidos Lingüística Concatenada Ordenada, ***FASDLCO*** (ver capítulo 3,

correspondiente al escenario 2), la Función de Asignación para Sistemas Distribuidos Lingüística Ordenada por Grupo Compatible, ***FASDLOGC*** (ver capítulo 4, correspondiente al escenario 3). Seguidamente se deberá aplicar el proceso de resolución. Ver Fig. 41.

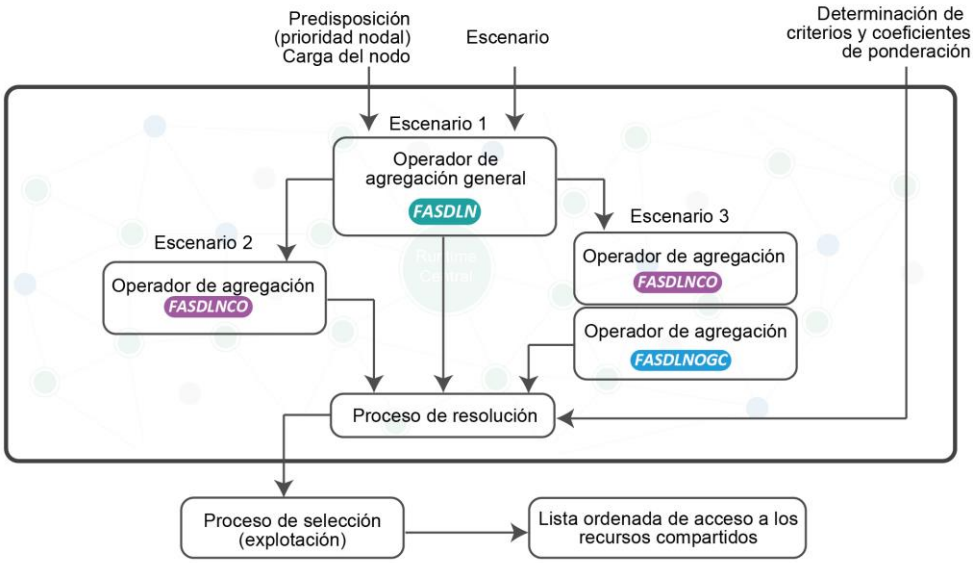


Fig. 41. Modelo de decisión propuesto.

Fuente: Elaboración propia.

5.5. Consideraciones del modelo de decisión propuesto

En el modelo propuesto, en cada nodo se define una interfaz entre las aplicaciones y el sistema operativo, que a través de un *Runtime* (software en tiempo de ejecución complementario del sistema operativo) incluído en esa interfaz, gestiona los procesos y recursos compartidos y define el escenario correspondiente. Además, los *Runtime* interactúan entre sí para intercambiar información y existe un *Runtime* coordinador global en uno de los nodos que evalúa y ejecuta el modelo de decisión y el operador de agregación correspondiente, ver Fig. 42.

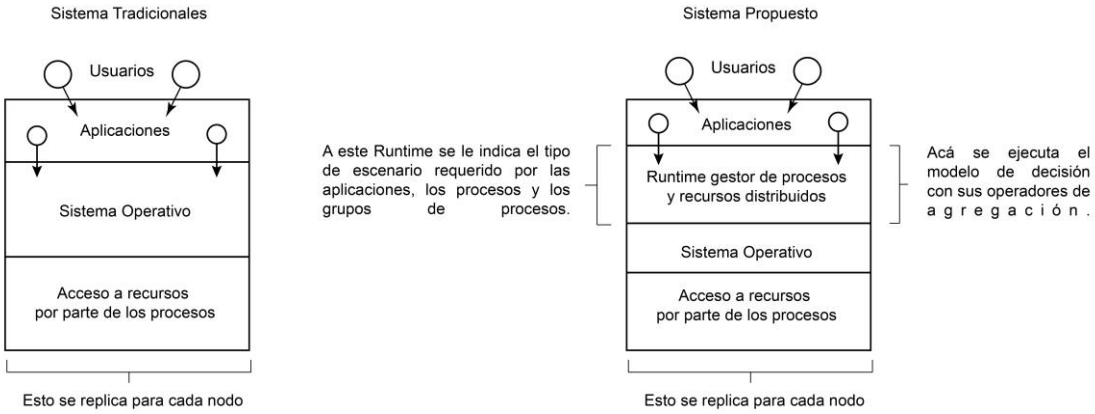


Fig. 42. Comparación del Modelo propuesto y los modelos tradicionales.

Fuente: Elaboración propia.

Comparación de las diferencias más significativas del método propuesto con respecto a los métodos tradicionales, se considera el algoritmo centralizado, el Algoritmo Distribuido de Lamport, Ricart y Agrawala y el Algoritmo de *Token Ring* mencionados en Colouris et al. (1988) y en Tanenbaum y Van Steen (2001): El algoritmo propuesto establece el orden inicial de asignación por prioridades, mientras que los demás lo hacen teniendo en cuenta la *timestamp*. Los requisitos de acceso a recursos en el algoritmo propuesto se establecen calculando prioridades, mientras que en el algoritmo centralizado es por mensajes al coordinador, el Algoritmo Distribuido de Lamport, Ricart y Agrawala a través de mensajes a procesos *n*-1 y el Algoritmo *Token Ring* a través de mensajes a sus vecinos. El método de asignación en el algoritmo propuesto contempla la mayor prioridad, mientras que en los demás se tiene en cuenta la *timestamp* más antigua. Sólo el método propuesto tiene en cuenta los grupos de procesos, mientras que los demás sólo tienen en cuenta procesos independientes. La predisposición (prioridad nodal), carga de nodos, estado nodal (nodos, procesos, grupos, recursos) y estado global del sistema, sólo se tiene en cuenta en el algoritmo propuesto. En los métodos tradicionales no se utilizan etiquetas lingüísticas ni 2-tuplas para la representación de los datos, además, si lo hicieran, en el proceso de traslación simbólica,

sólamente se trabajaría con la prioridad inicial o la *timestamp*, a diferencia del método propuesto que considera muchas variantes, evaluaciones e información adicional de los nodos, recursos y procesos que intervienen.

5.6. Prioridad nodal colaborativa

En los sistemas distribuidos, se utilizan distintos protocolos de enrutamiento, que permiten la incorporación de nodos a la red (por ejemplo, en una red *Ad Hoc*). No existe un registro en el sistema sobre la incorporación de un nodo ni de los recursos que este puede ofrecerle a la red, cada nodo puede salir perjudicado en cuanto a cantidad de recursos si ingresa a una red que no llene sus expectativas y que por el contrario sólo le solicite recursos. La red puede estar aceptando un nodo que es egoísta y que solo va a explotar recursos de la red y en realidad no va a aportar nada, según Toh et al. (2010) y Loo et al. (2012).

En este ámbito, se podría tener un escenario muy acorde al incluir la capacidad de toma de decisiones en los nodos, de manera que autónomamente indiquen la prioridad colaborativa que tiene cada uno, dependiendo de la cantidad y el orden en el que ofrezcan sus recursos.

Partiendo de la idea aplicada a los sistemas de *grid*, pero adecuándola a un concepto totalmente colaborativo, esta solución es superadora del problema mencionado anteriormente, dado que el algoritmo proporciona el orden de asignaciones, pero no hay restricciones en el acceso a los recursos, es un esquema colaborativo.

Ningún nodo restringe el acceso a sus recursos, solamente se ordena el acceso en base a las pautas de prioridad, consumo de recursos, etc. El modelo opera en base a distintos ciclos, recoge información y resuelve, posterior a cada finalización de ciclo, se calcula una prioridad colaborativa, donde se agrega un ítem más de prioridad, que va a entrar en el cálculo de las

prioridades nodales, donde en función de la cantidad de recursos y la prioridad de otorgar recursos de un nodo, obtenga más prioridad para acceder a sus propios requerimientos. En este ciclo de realimentación, se genera una nueva variable que marca la prioridad colaborativa, y en donde se da más prioridad al nodo que más compartió recursos.

Se implementa un indicador de prioridad por nodo, que depende de la cantidad de recursos y del orden en cual los ofreció. Para el cálculo se utiliza un vector de pesos asociado a orden y cantidad, y es un vector variable, dado que se podría evaluar que se va a ponderar más, si la cantidad de recursos ofrecidos por el nodo, o el orden en el cual los ofrece. Por ejemplo, dos nodos que compartieron la misma cantidad de recursos cada uno, pero uno los compartió de entrada y el otro lo hizo al final, se debería poder balancear ambos criterios, mediante un vector de ponderación en función de la cantidad de recursos y el orden, para obtener un valor de prioridad. Una ponderación sería priorizar de igual manera los dos criterios, el coeficiente sería igual para ambos, también se podría priorizar el orden, por sobre el de la cantidad, o viceversa, por lo cual los valores de coeficientes serían distintos.

La **autorregulación nodal colaborativa** que se muestra en la Fig. 43, corresponde a los valores calculados por el *Runtime* Central para cada nodo, al finalizar el cálculo global final de asignación de recursos a procesos para cada Macrociclo, entendiendose por Macrociclo, a la recopilación de información de requerimientos de asignaciones de recursos a procesos y de estado de los distintos nodos que se habrá resuelto con los operadores de agregación pertinente a los escenarios correspondientes (la tabla ***FASDLNC*** para el escenario E1, la tabla ***FASDLNCO*** para el escenario E2 y la tabla ***FASDLNOGC*** para el escenario E3).

El valor nodal colaborativo calculado por el *Runtime* Central, será un criterio más utilizado en el cálculo de las prioridades nodales para cada nodo (**paso 02**), a partir del segundo Macrociclo,

estando estos valores en 0 para el primer Macrociclo.

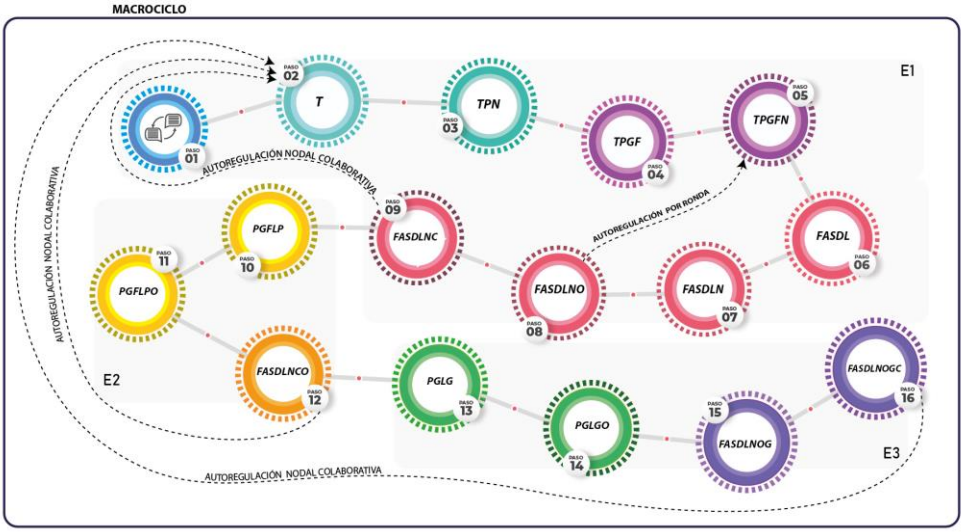


Fig. 43. Representación de un macrociclo, autoregulación por ronda y autoregulación nodal colaborativa por finalización de cada escenario.

Fuente: Elaboración propia.

Como resultado del ciclo de realimentación, se tendría una variable más, actualizada y ponderada, que indicara qué tan colaborativo fue el nodo con los demás, el que fue poco colaborativo va a tener un valor de prioridad colaborativa muy baja, porque habrá tenido muy pocos recursos que ofreció, o porque los ofreció con un orden muy bajo, y otro nodo muy colaborativo va a tener mayor prioridad. Esto contribuye al ***balanceo dinámico*** y a la ***autorregulación del sistema***.

5.7. Traslación Simbólica Descendente

Consiste en brindar información a los distintos nodos que han participado en el proceso de evaluación, del estado global del sistema, de las asignaciones de recursos a procesos, para saber cuál es el orden de asignación global de solicitudes, con el objetivo de que puedan tomar decisiones, medidas, criterios, acciones, para mejorar el desempeño de su propio rendimiento y así aumentar las posibilidades de obtener mejores resultados a la hora de la evaluación de prioridades. Esta traslación la realiza el *Runtime* Central del nodo coordinador, a través de su modelo de

decisión, ver Fig. 44.

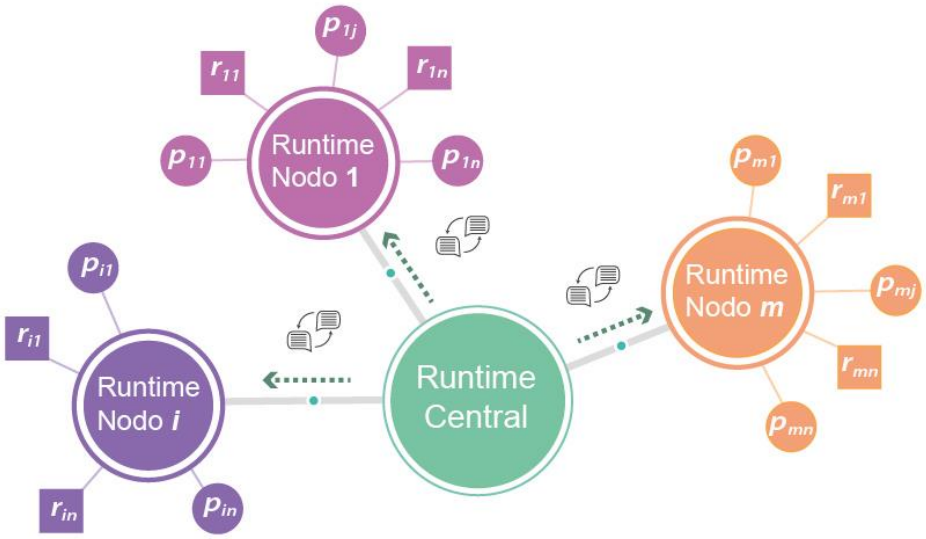


Fig. 44. Traslación Simbólica Descendente.

Fuente: Elaboración propia.

Este proceso consiste en transformar el conjunto de etiquetas utilizado por el *Runtime* Central, a cada uno de los conjuntos propios de cada Nodo, con el propósito de que cada *Runtime* local de cada nodo pueda interpretar de manera correcta el listado final de asignaciones de acuerdo a su propio conjunto de etiquetas.

5.8. Ejemplo de modelo de decisión aplicado a alguno de los algoritmos tradicionales

Un caso particular del modelo de decisión propuesto consiste en visualizar cómo alguno de los métodos considerados tradicionales en este trabajo, son un caso particular de este método.

Tabla 92. Pesos asignados a los procesos para el cálculo de prioridades

Procesos	Pesos Finales	
	Si integra un grupo de procesos	Si es independiente
p_{11}	-	$wf_{11}=1/np$
….	-	….
p_{kl}	-	$wf_{kl}=1/np$
….	-	
p_{np}	-	$wf_{np}=1/np$

Fuente: Elaboración propia. Basado en La Red Martínez (2017).

Como los métodos tradicionales no consideran los grupos de procesos, el cálculo sólo

realizará con procesos independientes y se deberá inhabilitar la columna que considera si un proceso integra un grupo de procesos, ver Tabla 92.

Los métodos considerados tradicionales no consideran la mayoría de los criterios contemplados en el modelo propuesto (además de no considerar la representación mediante etiquetas lingüísticas ni 2-tuplas), solo se basan en el cálculo del criterio "Prioridad Proceso", por lo tanto, los pesos asignados a los criterios de la Tabla 93, para calcular la prioridad global, solo se tendrá en cuenta el criterio "Prioridad Proceso", inhabilitando los demás criterios.

Para cada requerimiento de un recurso hecho por un proceso, se aplican los vectores de valoraciones según los criterios establecidos para la determinación de la prioridad, en el nodo en el cual se produce el requerimiento; se debe multiplicar escalarmente cada vector de valoraciones de cada requerimiento por el vector de pesos correspondiente a la categoría de carga actual del nodo para obtener la prioridad nodal.

Tabla 93. Pesos asignados a los criterios para calcular la prioridad

Categorías	Pesos							
	N° Proc.	% CPU	% Mem.	% MV	Prioridad Proc.	Sobrec. Mem.	Sobrec. Proc.	Sobrec. E/S
Alta	-	-	-	-	0.1000	-	-	-
Media	-	-	-	-	0.2000	-	-	-
Baja	-	-	-	-	0.1000	-	-	-

Fuente: Elaboración propia. Basado en La Red Martínez (2017).

Si bien el modelo de decisión obtiene la información de todos los criterios, cabe destacar que para los métodos tradicionales, del vector de pesos (Tabla 93), sólo va a influir en la prioridad, el criterio "Prioridad Proceso".

Las valoraciones asignadas a los criterios para calcular la prioridad o preferencia que cada nodo otorgará a cada requerimiento de cada proceso según la carga del nodo, serán las utilizadas en las Tabla 11 y Tabla 12. Para el Cálculo de las prioridades o preferencias de los procesos

teniendo en cuenta el estado del nodo, se utilizán las Tabla 13 y Tabla 14 pero inhabilitando todos los criterios, excepto el de la "Prioridad Proceso", esto se muestra en las Tabla 94.

Tabla 94. Las valoraciones asignadas a los criterios para calcular la prioridad o preferencia nodal que cada nodo concederá a cada requisito de cada proceso según la carga del nodo

Recurso	N°Proc	%CPU	%Mem	%MV	**Priorid Proc**	Sobrec Mem	Sobrec Proc	Sobre E/S
$Pri(p_{11}r_{11})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{11}r_{12})$	0	0	0	0	T(EB;0.0300)	0	0	0
$Pri(p_{11}r_{21})$	0	0	0	0	T(MB;-0.0767)	0	0	0
$Pri(p_{11}r_{22})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{11}r_{23})$	0	0	0	0	T(MB;-0.0717)	0	0	0
$Pri(p_{11}r_{24})$	0	0	0	0	T(EB;0.0600)	0	0	0
$Pri(p_{12}r_{11})$	0	0	0	0	T(MB;-0.0667)	0	0	0
$Pri(p_{12}r_{12})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{12}r_{21})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{12}r_{22})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{12}r_{31})$	0	0	0	0	T(EB;0.0300)	0	0	0
$Pri(p_{12}r_{33})$	0	0	0	0	T(EB;0.0300)	0	0	0
$Pri(p_{13}r_{11})$	0	0	0	0	T(EB;0.0600)	0	0	0
$Pri(p_{13}r_{12})$	0	0	0	0	T(MB;-0.0767)	0	0	0
$Pri(p_{13}r_{13})$	0	0	0	0	T(MB;-0.0767)	0	0	0
$Pri(p_{13}r_{21})$	0	0	0	0	T(EB;0.0500)	0	0	0
$Pri(p_{13}r_{22})$	0	0	0	0	T(EB;0.0500)	0	0	0
$Pri(p_{13}r_{31})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{13}r_{32})$	0	0	0	0	T(EB;0.0400)	0	0	0
$Pri(p_{13}r_{33})$	0	0	0	0	T(MB;-0.0767)	0	0	0
$Pri(p_{21}r_{12})$	0	0	0	0	T(MB;-0.0267)	0	0	0
$Pri(p_{21}r_{13})$	0	0	0	0	T(MB;-0.0267)	0	0	0
$Pri(p_{21}r_{22})$	0	0	0	0	T(MB;-0.0667)	0	0	0
$Pri(p_{21}r_{23})$	0	0	0	0	T(MB;-0.0667)	0	0	0
$Pri(p_{21}r_{31})$	0	0	0	0	T(MB;0.0133)	0	0	0
$Pri(p_{21}r_{33})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{22}r_{21})$	0	0	0	0	T(MB;-0.0467)	0	0	0
$Pri(p_{22}r_{22})$	0	0	0	0	T(MB;0.0133)	0	0	0
$Pri(p_{22}r_{31})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{22}r_{33})$	0	0	0	0	T(MB;-0.0067)	0	0	0
$Pri(p_{23}r_{12})$	0	0	0	0	T(MB;-0.0667)	0	0	0
$Pri(p_{23}r_{24})$	0	0	0	0	T(EB;0.0600)	0	0	0
$Pri(p_{23}r_{31})$	0	0	0	0	T(MB;-0.0267)	0	0	0
$Pri(p_{23}r_{32})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{23}r_{33})$	0	0	0	0	T(MB;0.0133)	0	0	0
$Pri(p_{24}r_{11})$	0	0	0	0	T(MB;-0.0667)	0	0	0
$Pri(p_{24}r_{12})$	0	0	0	0	T(MB;0.0133)	0	0	0
$Pri(p_{24}r_{23})$	0	0	0	0	T(MB;-0.0067)	0	0	0
$Pri(p_{24}r_{24})$	0	0	0	0	T(EB;0.0600)	0	0	0
$Pri(p_{25}r_{21})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{31}r_{13})$	0	0	0	0	T(EB;0.0700)	0	0	0
$Pri(p_{31}r_{31})$	0	0	0	0	T(EB;0.0700)	0	0	0
$Pri(p_{31}r_{33})$	0	0	0	0	T(MB;-0.0767)	0	0	0
$Pri(p_{32}r_{11})$	0	0	0	0	T(MB;-0.0767)	0	0	0
$Pri(p_{32}r_{12})$	0	0	0	0	T(EB;0.0800)	0	0	0
$Pri(p_{32}r_{13})$	0	0	0	0	T(MB;-0.0767)	0	0	0

Pri($p_{32}r_{23}$)	0	0	0	0	T(EB;0.0600)	0	0	0
Pri($p_{33}r_{11}$)	0	0	0	0	T(EB;0.0600)	0	0	0
Pri($p_{33}r_{12}$)	0	0	0	0	T(EB;0.0300)	0	0	0
Pri($p_{33}r_{13}$)	0	0	0	0	T(EB;0.0300)	0	0	0
Pri($p_{33}r_{21}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{33}r_{22}$)	0	0	0	0	T(EB;0.0700)	0	0	0
Pri($p_{33}r_{23}$)	0	0	0	0	T(EB;0.0600)	0	0	0
Pri($p_{33}r_{32}$)	0	0	0	0	T(EB;0.0400)	0	0	0
Pri($p_{33}r_{33}$)	0	0	0	0	T(EB;0.0600)	0	0	0
Pri($p_{34}r_{12}$)	0	0	0	0	T(EB;0.0700)	0	0	0
Pri($p_{34}r_{13}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{34}r_{22}$)	0	0	0	0	T(MB;-0.0767)	0	0	0
Pri($p_{34}r_{23}$)	0	0	0	0	T(MB;-0.0767)	0	0	0
Pri($p_{34}r_{24}$)	0	0	0	0	T(EB;0.0700)	0	0	0
Pri($p_{34}r_{31}$)	0	0	0	0	T(EB;0.0700)	0	0	0
Pri($p_{34}r_{32}$)	0	0	0	0	T(EB;0.0600)	0	0	0
Pri($p_{34}r_{33}$)	0	0	0	0	T(MB;-0.0767)	0	0	0
Pri($p_{35}r_{12}$)	0	0	0	0	T(MB;-0.0767)	0	0	0
Pri($p_{35}r_{13}$)	0	0	0	0	T(MB;-0.0767)	0	0	0
Pri($p_{35}r_{22}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{35}r_{24}$)	0	0	0	0	T(EB;0.0600)	0	0	0
Pri($p_{35}r_{31}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{35}r_{32}$)	0	0	0	0	T(EB;0.0400)	0	0	0
Pri($p_{35}r_{33}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{36}r_{11}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{36}r_{12}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{36}r_{13}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{36}r_{21}$)	0	0	0	0	T(EB;0.0700)	0	0	0
Pri($p_{36}r_{22}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{36}r_{24}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{36}r_{31}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{36}r_{32}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{36}r_{33}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{37}r_{11}$)	0	0	0	0	T(MB;-0.0767)	0	0	0
Pri($p_{37}r_{12}$)	0	0	0	0	T(EB;0.0500)	0	0	0
Pri($p_{37}r_{21}$)	0	0	0	0	T(EB;0.0600)	0	0	0
Pri($p_{37}r_{32}$)	0	0	0	0	T(EB;0.0800)	0	0	0
Pri($p_{37}r_{33}$)	0	0	0	0	T(EB;0.0800)	0	0	0

Fuente: Elaboración propia.

Se debe calcular las prioridades nodales, pesos finales y prioridades globales de los procesos para acceder a los recursos.

La Tabla 95, Tabla 96 y Tabla 97 se construye a partir de los valores de prioridades nodales calculadas de la Tabla 94, que para este ejemplo coincide con el criterio “Prioridad Proceso”.

Tabla 95. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla (p_{11}, p_{12}, p_{13}, p_{21}, p_{22})

	p_{11}	p_{12}	p_{13}	p_{21}	p_{22}
r_{11}	TPN(EB;0.0800)	TPN(MB;-0.0667)	TPN(EB;0.0600)	-	-
r_{12}	TPN(EB;0.0300)	TPN(EB;0.0800)	TPN(MB;-0.0767)	TPN(MB;-0.0267)	-
r_{13}	-	-	TPN(MB;-0.0767)	TPN(MB;-0.0267)	-
r_{21}	TPN(MB;-0.0767)	TPN(EB;0.0800)	TPN(EB;0.0500)	-	TPN(MB;-0.0467)
r_{22}	TPN(EB;0.0800)	TPN(EB;0.0800)	TPN(EB;0.0500)	TPN(MB;-0.0667)	TPN(MB;0.0133)
r_{23}	TPN(MB;-0.0717)	-	-	TPN(MB;-0.0667)	-
r_{24}	TPN(EB;0.0600)	-	-	-	-
r_{31}	-	TPN(EB;0.0300)	TPN(EB;0.0800)	TPN(MB;0.0133)	TPN(EB;0.0800)
r_{32}	-	-	TPN(EB;0.0400)	-	-
r_{33}	-	TPN(EB;0.0300)	TPN(MB;-0.0767)	TPN(EB;0.0800)	TPN(MB;-0.0067)

Fuente: Elaboración propia.

Tabla 96. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla (p_{23}, p_{24}, p_{25}, p_{31}, p_{32})

	p_{23}	p_{24}	p_{25}	p_{31}	p_{32}
r_{11}	-	TPN(MB;-0.0667)	-	-	TPN(MB;-0.0767)
r_{12}	TPN(MB;-0.0667)	TPN(MB;0.0133)	-	-	TPN(EB;0.0800)
r_{13}	-	-	-	TPN(EB;0.0700)	TPN(MB;-0.0767)
r_{21}	-	-	TPN(EB;0.0800)	-	-
r_{22}	-	-	-	-	-
r_{23}	-	TPN(MB;-0.0067)	-	-	TPN(EB;0.0600)
r_{24}	TPN(EB;0.0600)	TPN(EB;0.0600)	-	-	-
r_{31}	TPN(MB;-0.0267)	-	-	TPN(EB;0.0700)	-
r_{32}	TPN(EB;0.0800)	-	-	-	-
r_{33}	TPN(MB;0.0133)	-	-	TPN(MB;-0.0767)	-

Fuente: Elaboración propia.

Tabla 97. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla (p_{33}, p_{34}, p_{35}, p_{36}, p_{37})

	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}
r_{11}	TPN(EB;0.0600)	-	-	TPN(EB;0.0800)	TPN(MB;-0.0767)
r_{12}	TPN(EB;0.0300)	TPN(EB;0.0700)	TPN(MB;-0.0767)	TPN(EB;0.0800)	TPN(EB;0.0500)
r_{13}	TPN(EB;0.0300)	TPN(EB;0.0800)	TPN(MB;-0.0767)	TPN(EB;0.0800)	-
r_{21}	TPN(EB;0.0800)	-	-	TPN(EB;0.0700)	TPN(EB;0.0600)
r_{22}	TPN(EB;0.0700)	TPN(MB;-0.0767)	TPN(EB;0.0800)	TPN(EB;0.0800)	-
r_{23}	TPN(EB;0.0600)	TPN(MB;-0.0767)	-	-	-
r_{24}	-	TPN(EB;0.0700)	TPN(EB;0.0600)	TPN(EB;0.0800)	-
r_{31}	-	TPN(EB;0.0700)	TPN(EB;0.0800)	TPN(EB;0.0800)	-
r_{32}	TPN(EB;0.0400)	TPN(EB;0.0600)	TPN(EB;0.0400)	TPN(EB;0.0800)	TPN(EB;0.0800)
r_{33}	TPN(EB;0.0600)	TPN(MB;-0.0767)	TPN(EB;0.0800)	TPN(EB;0.0800)	TPN(EB;0.0800)

Fuente: Elaboración propia.

Como se había mencionado, los métodos tradicionales no consideran los grupos de procesos, el cálculo sólo tendrá en cuenta que los procesos son independientes. Dado que en el ejemplo existen 15 procesos, y que el cálculo de los pesos es wf_{ij} igual a 1/np para procesos independientes, donde np es el número de procesos en el sistema (15), el cálculo para los pesos de cada proceso (wp_{ij}) es igual a 1/15. Para el cálculo de los pesos normalizados (nwp_{ij}) se divide

cada valor de wp_{ij} por la sumatoria de todos los wp_{ij}, esto se puede observar en la Tabla 95, Tabla 96 y Tabla 97. Se debe calcular el vector de peso final que se utilizará en el proceso de agregación final para determinar el orden o la prioridad de acceso a los recursos. Además, los pesos recientemente obtenidos deberán normalizarse dividiendo cada uno de ellos por la suma de todos ellos, esto se puede observar en la Tabla 98 y Tabla 99.

Tabla 98. Pesos finales y pesos finales normalizados (p_{11}, p_{12}, p_{13}, p_{21}, p_{22}, p_{23}, p_{24}, p_{25})

	p11	**p12**	**p13**	**p21**	**p22**	**p23**	**p24**	**p25**
Pesos Finales (wp_{ij})	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667
Pesos Finales Normalizados (nwp_{ij})	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667

Fuente: Elaboración propia.

Tabla 99. Pesos finales y pesos finales normalizados (p_{31}, p_{32}, p_{33}, p_{34}, p_{35}, p_{36}, p_{37})

	p31	**p32**	**p33**	**p34**	**p35**	**p36**	**p37**	**Total**
Pesos Finales (wp_{ij})	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667	1
Pesos Finales Normalizados (nwp_{ij})	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667	0.0667	1

Fuente: Elaboración propia.

Las prioridades nodales indicadas en la Tabla 95, Tabla 96 y Tabla 97 tomadas fila por fila, es decir, para cada recurso, se multiplicarán por el vector de peso normalizado (nwp_{ij}) de la Tabla 98 y Tabla 99. Esto se puede ver en la Tabla 100, Tabla 101 y Tabla 110.

Tabla 100. Prioridades globales finales de los procesos para acceder a cada recurso (p_{11}, p_{12}, p_{13}, p_{21}, p_{22})

Recursos	Prioridades Nodales de los Procesos				
	p_{11}	p_{12}	p_{13}	p_{21}	p_{22}
r_{11}	TPN(EB;0.005)	TPN(EB;0.007)	TPN(EB;0.004)	-	-
r_{12}	TPN(EB;0.002)	TPN(EB;0.005)	TPN(EB;0.006)	TPN(EB;0.009)	-
r_{13}	-	-	TPN(EB;0.006)	TPN(EB;0.009)	-
r_{21}	TPN(EB;0.006)	TPN(EB;0.005)	TPN(EB;0.003)	-	TPN(EB;0.008)
r_{22}	TPN(EB;0.005)	TPN(EB;0.005)	TPN(EB;0.003)	TPN(EB;0.007)	TPN(EB;0.012)
r_{23}	TPN(EB;0.006)	-	-	TPN(EB;0.007)	-
r_{24}	TPN(EB;0.004)	-	-	-	-
r_{31}	-	TPN(EB;0.002)	TPN(EB;0.005)	TPN(EB;0.012)	TPN(EB;0.005)
r_{32}	-	-	TPN(EB;0.003)	-	-
r_{33}	-	TPN(EB;0.002)	TPN(EB;0.006)	TPN(EB;0.005)	TPN(EB;0.011)

Fuente: Elaboración propia.

Tabla 101. Prioridades globales finales de los procesos para acceder a cada recurso (p_{23}, p_{24}, p_{25}, p_{31}, p_{32})

Recursos	Prioridades Nodales de los Procesos				
	p_{23}	p_{24}	p_{25}	p_{31}	p_{32}
r_{11}	-	TPN(EB;0.007)	-	-	TPN(EB;0.006)
r_{12}	TPN(EB;0.007)	TPN(EB;0.012)	-	-	TPN(EB;0.005)
r_{13}	-	-	-	TPN(EB;0.005)	TPN(EB;0.006)
r_{21}	-	-	TPN(EB;0.005)	-	-
r_{22}	-	-	-	-	-
r_{23}	-	TPN(EB;0.011)	-	-	TPN(EB;0.004)
r_{24}	TPN(EB;0.004)	TPN(EB;0.004)	-	-	-
r_{31}	TPN(EB;0.009)	-	-	TPN(EB;0.005)	-
r_{32}	TPN(EB;0.005)	-	-	-	-
r_{33}	TPN(EB;0.012)	-	-	TPN(EB;0.006)	-

Fuente: Elaboración propia.

Tabla 102. Prioridades globales finales de los procesos para acceder a cada recurso (p_{33}, p_{34}, p_{35}, p_{36}, p_{37})

Recursos	Prioridades Nodales de los Procesos				
	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}
r_{11}	TPN(EB;0.004)	-	-	TPN(EB;0.005)	TPN(EB;0.006)
r_{12}	TPN(EB;0.002)	TPN(EB;0.005)	TPN(EB;0.006)	TPN(EB;0.005)	TPN(EB;0.003)
r_{13}	TPN(EB;0.002)	TPN(EB;0.005)	TPN(EB;0.006)	TPN(EB;0.005)	-
r_{21}	TPN(EB;0.005)	-	-	TPN(EB;0.005)	TPN(EB;0.004)
r_{22}	TPN(EB;0.005)	TPN(EB;0.006)	TPN(EB;0.005)	TPN(EB;0.005)	-
r_{23}	TPN(EB;0.004)	TPN(EB;0.006)	-	-	-
r_{24}	-	TPN(EB;0.005)	TPN(EB;0.004)	TPN(EB;0.005)	-
r_{31}	-	TPN(EB;0.005)	TPN(EB;0.005)	TPN(EB;0.005)	-
r_{32}	TPN(EB;0.003)	TPN(EB;0.004)	TPN(EB;0.003)	TPN(EB;0.005)	TPN(EB;0.005)
r_{33}	TPN(EB;0.004)	TPN(EB;0.006)	TPN(EB;0.005)	TPN(EB;0.005)	TPN(EB;0.005)

Fuente: Elaboración propia.

El siguiente paso es normalizar la Tabla 100, Tabla 101 y Tabla 110 entre los valores extremos. Para ello, se debe restar el valor numérico de la 2-tupla por el valor mínimo de todas ellas y dividirlo por el rango, que es la diferencia entre el valor máximo y el valor mínimo de las tablas ya mencionadas. Esto se puede ver en la Tabla 103.

Tabla 103. Valoraciones para normalizar la ***FASDL***

Etiqueta	Valor
Valor Máximo	0.012
Valor Mínimo	0.002
Rango	0.010

Fuente: Elaboración propia.

El resultado de esta normalización se puede ver en la Tabla 104, Tabla 105 y Tabla 106. El mayor de estos productos realizados para los diferentes procesos en relación con el mismo recurso indicará cuál de los procesos tendrá acceso al recurso.

Tabla 104. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla normalizada (p_{11}, p_{12}, p_{13}, p_{21}, p_{22})

Recursos	Prioridades Nodales de los Procesos				
	p_{11}	p_{12}	p_{13}	p_{21}	p_{22}
r_{11}	TPGFN(B;0.000)	TPGFN(M;-0.033)	TPGFN(MB;0.033)	-	-
r_{12}	TPGFN(EB;0.000)	TPGFN(B;0.000)	TPGFN(B;0.067)	TPGFN(A;0.067)	-
r_{13}	-	-	TPGFN(B;0.067)	TPGFN(A;0.067)	-
r_{21}	TPGFN(B;0.067)	TPGFN(B;0.000)	TPGFN(MB;-0.033)	-	TPGFN(A;-0.067)
r_{22}	TPGFN(B;0.000)	TPGFN(B;0.000)	TPGFN(MB;-0.033)	TPGFN(M;-0.033)	TPGFN(EA;0.000)
r_{23}	TPGFN(M;-0.067)	-	-	TPGFN(M;-0.033)	-
r_{24}	TPGFN(MB;0.033)	-	-	-	-
r_{31}	-	TPGFN(EB;0.000)	TPGFN(B;0.000)	TPGFN(EA;0.000)	TPGFN(B;0.000)
r_{32}	-	-	TPGFN(EB;0.067)	-	-
r_{33}	-	TPGFN(EB;0.000)	TPGFN(B;0.067)	TPGFN(B;0.000)	TPGFN(MA;0.033)

Fuente: Elaboración propia.

Tabla 105. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla normalizada (p_{23}, p_{24}, p_{25}, p_{31}, p_{32})

Recursos	Prioridades Nodales de los Procesos				
	p_{23}	p_{24}	p_{25}	p_{31}	p_{32}
r_{11}	-	TPGFN(M;-0.033)	-	-	TPGFN(B;0.067)
r_{12}	TPGFN(M;-0.033)	TPGFN(EA;0.000)	-	-	TPGFN(B;0.000)
r_{13}	-	-	-	TPGFN(B;-0.067)	TPGFN(B;0.067)
r_{21}	-	-	TPGFN(B;0.000)	-	-
r_{22}	-	-	-	-	-
r_{23}	-	TPGFN(MA;0.033)	-	-	TPGFN(MB;0.033)
r_{24}	TPGFN(MB;0.033)	TPGFN(MB;0.033)	-	-	-
r_{31}	TPGFN(A;0.067)	-	-	TPGFN(B;-0.067)	-
r_{32}	TPGFN(B;0.000)	-	-	-	-
r_{33}	TPGFN(EA;0.000)	-	-	TPGFN(B;0.067)	-

Fuente: Elaboración propia.

Tabla 106. Prioridades nodales de los procesos para acceder a cada recurso en 2-tupla normalizada (p_{33}, p_{34}, p_{35}, p_{36}, p_{37})

Recursos	Prioridades Nodales de los Procesos				
	p_{33}	p_{34}	p_{35}	p_{36}	p_{37}
r_{11}	TPGFN(MB;0.033)	-	-	TPGFN(B;0.000)	TPGFN(B;0.067)
r_{12}	TPGFN(EB;0.000)	TPGFN(B;-0.067)	TPGFN(B;0.067)	TPGFN(B;0.000)	TPGFN(MB;-0.033)
r_{13}	TPGFN(EB;0.000)	TPGFN(B;0.000)	TPGFN(B;0.067)	TPGFN(B;0.000)	-
r_{21}	TPGFN(B;0.000)	-	-	TPGFN(B;-0.067)	TPGFN(MB;0.033)
r_{22}	TPGFN(B;-0.067)	TPGFN(B;0.067)	TPGFN(B;0.0000)	TPGFN(B;0.000)	-
r_{23}	TPGFN(MB;0.033)	TPGFN(B;0.067)	-	-	-
r_{24}	-	TPGFN(B;-0.067)	TPGFN(MB;0.033)	TPGFN(B;0.000)	-
r_{31}	-	TPGFN(B;-0.067)	TPGFN(B;0.000)	TPGFN(B;0.000)	-
r_{32}	TPGFN(EB;0.067)	TPGFN(MB;0.033)	TPGFN(EB;0.067)	TPGFN(B;0.000)	TPGFN(B;0.000)
r_{33}	TPGFN(MB;0.033)	TPGFN(B;0.067)	TPGFN(B;0.000)	TPGFN(B;0.000)	TPGFN(B;0.000)

Fuente: Elaboración propia.

La suma de todos estos productos en relación con el mismo recurso indicará la prioridad que deberá asignarse a este recurso, en relación con los demás recursos que también deberán

asignarse. Esto constituye la Función de Asignación del Sistema Distribuido Lingüística (***FASDL***):

FASDL(r_{ij}) = Σ **TPGFN**$_{ijkl}$ = r_{ij} prioridad de asignación de recursos.

Al calcular la ***FASDL*** para todos los recursos, se obtendrá un vector de 2-tuplas y, ordenando sus elementos de mayor a menor, se obtendrá el orden de prioridad de asignación de recursos, que deberá ser normalizado asegurando que las 2-tuplas obtenidas estén en el intervalo [0, 1]. Esto se puede observar en la Tabla 107.

Tabla 107. Valoraciones para normalizar la ***FASDL***

Etiqueta	Valor
Valor Máximo	4.5999
Valor Mínimo	1.4003
Rango	3.1996

Fuente: Elaboración propia.

Además, como se ha indicado anteriormente, el mayor de los TPGFN$_{ijkl}$ de cada recurso indicará el proceso al que se asignará el recurso.

El resultado de la normalización de las 2-tuplas constituye lo que se denominará Función de Asignación del Sistema Distribuido Lingüística Normalizada a (***FASDLN***):

FASDLN(r_{ij}) = Σ (***TPGFN***$_{ijkl}$ / (Máximo (***TPGFN***$_{ijkl}$) - Mínimo (***TPGFN***$_{ijkl}$)) = r_{ij} prioridad de asignación de recursos normalizada entre valores extremos, como se puede ver en la Tabla 108.

Tabla 108. Orden o prioridad final de asignación de los recursos y proceso al cual se asigna cada recurso en una iteración

Orden de asignación de los recursos	Prioridad de la asignación	Proceso al que se asignará el recurso
r_{33}	T(EA;0.0000)	p_{23}
r_{12}	T(EA;-0.0625)	p_{24}
r_{31}	T(A;0.0209)	p_{21}
r_{22}	T(A;0.0209)	p_{22}
r_{13}	T(M;-0.0418)	p_{21}
r_{11}	T(M;-0.0626)	p_{12}
r_{21}	T(B;0.0417)	p_{22}
r_{23}	T(B;0.0311)	p_{24}
r_{24}	T(EB;0.0000)	p_{36}
r_{32}	T(EB;0.0000)	p_{23}

Fuente: Elaboración propia.

El siguiente paso es repetir el procedimiento, pero eliminando las solicitudes de

asignaciones ya realizadas; cabe señalar que los recursos asignados estarán disponibles una vez que los procesos los liberen y, por lo tanto, podrán asignarse a otros procesos.

Dado que el sistema se autorregula liberando los recursos ya asignados a los procesos en el paso anterior, y dado que hay solicitudes de recursos de los procesos que aún no han sido satisfechas, los cálculos de las Tabla 107 y Tabla 108 se repiten con sus respectivos valores, omitiendo los procesos ya completados.

El resultado final de la concatenación de todas las rondas de asignaciones para este ejemplo se puede ver en la Tabla 109.

Tabla 109. Concatenación de todas las rondas de asignaciones (***FASDLNC***) para los métodos tradicionales

Recurso	2-tuplas	*P*roceso	Ronda
r_{33}	T(EA;0.0000)	p_{23}	1
r_{12}	T(EA;-0.0625)	p_{24}	1
r_{31}	T(A;0.0209)	p_{21}	1
r_{22}	T(A;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(B;0.0417)	p_{22}	1
r_{23}	T(B;0.0311)	p_{24}	1
r_{24}	T(EB;0.0000)	p_{36}	1
r_{32}	T(EB;0.0000)	p_{23}	1
r_{33}	T(EA;0.0000)	p_{22}	2
r_{12}	T(EA;-0.079)	p_{21}	2
r_{31}	T(A;-0.0613)	p_{23}	2
r_{22}	T(A;-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(B;0.0351)	p_{11}	2
r_{23}	T(MB;0.0832)	p_{21}	2
r_{24}	T(EB;0.0000)	p_{34}	2
r_{32}	T(EB;0.0000)	p_{36}	2
r_{33}	T(EA;0.0000)	p_{13}	3
r_{12}	T(EA;-0.0333)	p_{23}	3
r_{22}	T(A;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(MB;0.0832)	p_{11}	3
r_{24}	T(EB;0.0333)	p_{11}	3
r_{32}	T(EB;0.0000)	p_{37}	3
r_{33}	T(EA;0.0000)	p_{31}	4
r_{12}	T(EA;-0.069)	p_{13}	4
r_{22}	T(A;0.0231)	p_{11}	4
r_{31}	T(A;-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(MB;0.0402)	p_{34}	4
r_{24}	T(MB;-0.0632)	p_{23}	4
r_{32}	T(EB;0.0000)	p_{34}	4
r_{33}	T(EA;0.0000)	p_{34}	5
r_{12}	T(EA;-0.0769)	p_{35}	5
r_{22}	T(A;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(MB;-0.0513)	p_{32}	5
r_{24}	T(MB;-0.0513)	p_{24}	5
r_{32}	T(EB;0.0000)	p_{13}	5
r_{33}	T(EA;0.0000)	p_{21}	6
r_{12}	T(MA;0.0714)	p_{12}	6
r_{22}	T(A;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(B;0.0000)	p_{36}	6
r_{21}	T(B;0.0000)	p_{36}	6
r_{23}	T(EB;0.0476)	p_{33}	6
r_{24}	T(EB;0.0476)	p_{35}	6
r_{32}	T(EB;0.0000)	p_{33}	6
r_{33}	T(EA;0.0000)	p_{35}	7
r_{12}	T(MA;0.0556)	p_{32}	7
r_{22}	T(A;-0.0556)	p_{36}	7
r_{31}	T(M;-0.0556)	p_{31}	7
r_{11}	T(B;0.0000)	p_{13}	7
r_{21}	T(B;-0.0556)	p_{37}	7
r_{13}	T(MB;0.0556)	p_{31}	7
r_{32}	T(EB;0.0556)	p_{35}	7
r_{33}	T(EA;0.0000)	p_{36}	8
r_{12}	T(MA;0.0128)	p_{36}	8

r_{22}	T(M;-0.0385)	p_{33}	8
r_{31}	T(B;-0.0256)	p_{34}	8
r_{11}	T(MB;0.0641)	p_{33}	8
r_{21}	T(MB;-0.0128)	p_{13}	8
r_{33}	T(EA;0.0000)	p_{37}	9
r_{12}	T(MA;-0.0833)	p_{34}	9
r_{22}	T(B;-0.0833)	p_{13}	9
r_{33}	T(EA;0.0000)	p_{33}	10
r_{12}	T(A;0.0000)	p_{37}	10
r_{12}	T(EA;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(EA;0.0000)	p_{33}	12

Fuente: Elaboración propia.

5.9. Comparación de los resultados obtenidos con los métodos tradicionales y el método propuesto

Se compara la Tabla 109 de este escenario, con respecto a la ***FASDLNC*** de la primera ronda (Tabla 110) del escenario E1 y se busca cada asignación de la primera ronda de esta última en la Tabla 109.

En la Tabla 110 se pueden observar el orden de asignaciones para la primer ronda del escenario E1, mientras que la Tabla 111 representa el orden en el que aparecen las mismas asignaciones pero para el método tradicional, en que ronda aparecen y en que posición respecto de cada ronda.

Tabla 110. Valores correspondientes a la primera ronda de iteración del escenario E1 (***FASDLNO***)

Pos.	Recurso	2-tuplas	Proceso	Ronda
1	r_{33}	T(EA;0.0000)	p_{23}	1
2	r_{12}	T(EA;-0.0326)	p_{37}	1
3	r_{31}	T(A;-0.0028)	p_{34}	1
4	r_{11}	T(A;-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(B;-0.0591)	p_{11}	1
10	r_{24}	T(EB;0.0000)	p_{34}	1

Fuente: Elaboración propia.

Tabla 111. Valores correspondientes a las mismas asignaciones de recursos a procesos de la tabla ***FASDLNO*** de la primera iteración del E1 encontradas en la tabla ***FASDLNC*** de los métodos tradicionales

Pos.	Recurso	2-tuplas	Proceso	Ronda
1	r_{33}	T(EA;0.0000)	p_{23}	1
2	r_{12}	T(A;0.0000)	p_{37}	10
4	r_{31}	T(B;-0.0256)	p_{34}	8
5	r_{11}	T(M;0.0517)	p_{37}	4
3	r_{22}	T(A;0.0335)	p_{34}	3
6	r_{21}	T(B;-0.0556)	p_{37}	7
6	r_{13}	T(M;-0.0791)	p_{13}	2
10	r_{32}	T(EB;0.0000)	p_{34}	4
8	r_{23}	T(MB;0.0832)	p_{11}	3
9	r_{24}	T(EB;0.0000)	p_{34}	2

Fuente: Elaboración propia.

El primer elemento de la Tabla 110, asignación del r_{33} al p_{23}, es el único que ocurre en la misma iteración (primera), todas las demás asignaciones en el ejemplo de los métodos tradicionales ocurren en diferentes rondas y en diferentes posiciones. Se ha visto que en esta comparación, los resultados de asignaciones en los métodos tradicionales no son iguales que los del modelo propuesto, y esto se debe a que los métodos tradicionales contemplan un solo tipo de criterio (prioridad de proceso), y no tienen en cuenta el número de procesos, %CPU, %Mem, %MV, etc., es decir, la carga de cada nodo y el estado global del sistema.

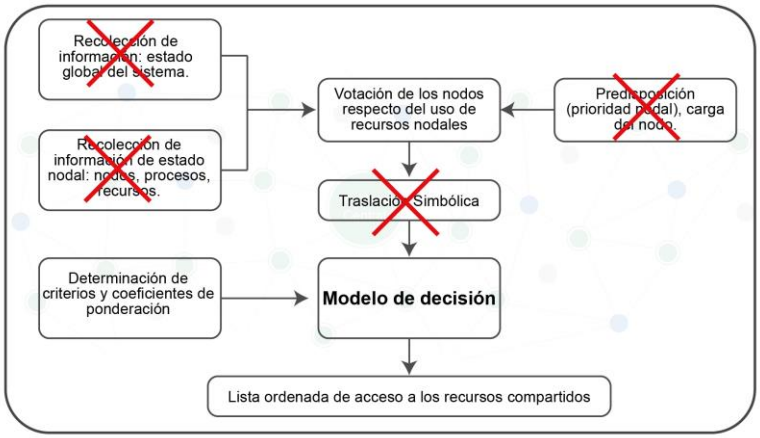


Fig. 45. Modelo global de decisión para accesos a recursos compartidos.

Fuente: Elaboración propia.

En este sentido, se puede decir que además de ser los métodos tradicionales, un caso particular del método propuesto, este nuevo modelo permite una evaluación más aproximada al

estado real del sistema, lo que permitiría obtener mejores resultados en las asignaciones.

En la Fig. 45 se puede observar que en el modelo global, para el ejemplo de los métodos tradicionales, no se tiene en cuenta la recolección de información del estado global del sistema, ni la predisposición (prioridad nodal), ni la carga del nodo, sólo se contempla la prioridad de proceso.

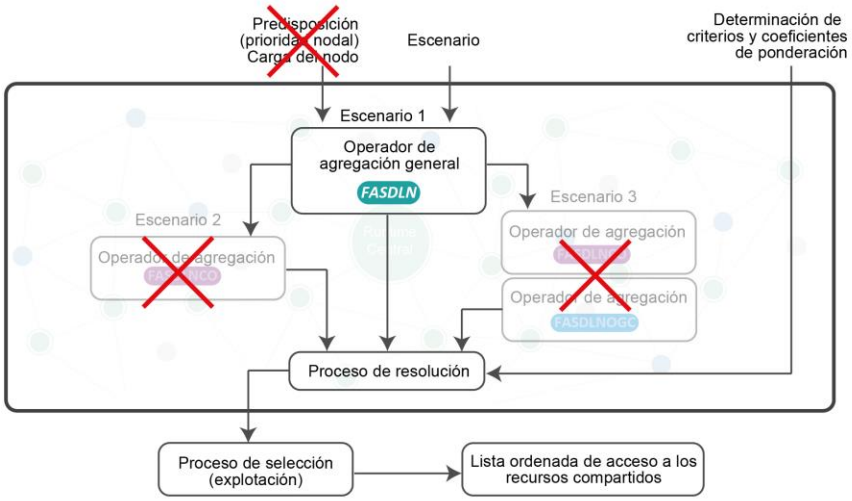


Fig. 46. Modelo de decisión incluyendo en el proceso de resolución los métodos de agregación específicos para al acceso a recursos compartidos.

Fuente: Elaboración propia.

Para la Fig. 46 el modelo de decisión, en el caso del ejemplo de los métodos tradicionales, no se tiene en cuenta la predisposición (prioridad nodal), ni la carga del nodo. La determinación de los criterios y coeficientes de ponderación se basan sólo en la prioridad de proceso. Además, no se contemplan los demás escenarios.

Cabe destacar que los resultados finales obtenidos se ajustan a cada escenario en particular. Es decir, al cambiar las condiciones del escenario, los resultados obtenidos en la aplicación del Modelo de Decisión pueden ser diferentes.

Algunas de las diferencias más significativas con respectos a los métodos considerados tradicionaels se pueden observar en la Tabla 112.

Tabla 112. Cuadro comparativo del modelo propuesto y los tradicionales

Características	Modelo propuesto	Algoritmo centralizado	Algoritmo Distribuido de Lamport, Ricart y Agrawala	Algoritmo de anillo de fichas
Orden inicial	por prioridades	marca de tiempo	marca de tiempo	marca de tiempo
Solicitudes	cálculo de prioridad	mensajes al coordinador	mensajes a *n*-1 procesos	mensajes al vecino
Método de asignación	mayor prioridad calculada	menor marca de tiempo o prioridad inicial	menor marca de tiempo o prioridad inicial	menor marca de tiempo o prioridad inicial
Grupos de procesos	agrupación por grupos	procesos independientes	procesos independientes	procesos independientes
Predisposición (prioridad nodal), carga del nodo.	✓	✗	✗	✗
Estado nodal (nodos, procesos, grupos, recursos.)	✓	✗	✗	✗
Estado global del sistema.	✓	✗	✗	✗

Fuente: Elaboración propia.

5.10. Consideraciones acerca de las operaciones de agregación

En La Red Martínez (2017) y La Red Martínez et al. (2018) las características de las operaciones de agregación descriptas permiten considerar que el método propuesto pertenece a la familia de operadores de agregación *Neat-OWA*, operadores que se caracterizan por lo siguiente:

La definición de los operadores *OWA* indica que $f(a_1, a_2, \ldots, a_n) = \sum_{j=1}^{n} w_j \cdot b_j$, donde b_j es el *j*-ésimo valor mayor de las a_n, con la restricción para los pesos de satisfacer (1) $w_i \in [0,1]$ y (2) $\sum_{i=1}^{n} w_i = 1$.

Para la familia de operadores *Neat OWA* los pesos serán calculados en función de los elementos que se agregan, o más exactamente de los valores a agregar ordenados, los b_j, manteniéndose las condiciones (1) y (2). Es este caso los pesos son: $w_i = f_i(b_1, ..., b_n)$, definiéndose

el operador

$$F(a_1,...a_n) = \sum_i f_i(b_1,...,b_n)\cdot b_i$$

Para esta familia, donde los pesos dependen de la agregación, no se exige la satisfacción de todas las propiedades de los operadores *OWA*.

Además, para poder afirmar que un operador de agregación es *neat*, es necesario que el valor final de agregación sea independiente del orden de los valores. Sea $A = (a_1,...,a_n)$ las estradas a agregar, sea $B = (b_1,...,b_n)$ las entradas ordenadas y $C = (c_1,...,c_n) = Perm(a_1,...,a_n)$ una permutación de las entradas. Formalmente se define un operador *OWA* como *neat* si

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

Produce el mismo resultado para cualquier asignación $C = B$.

5.11. Discusiones y comentarios

Se resalta el dinamismo, la magnitud de los nodos, procesos y la cantidad de requerimientos que pueden ser aplicables a grandes sistemas, mediante una solución global, pero también aplicables a sistemas de menor cantidad de nodos y requerimientos. La carga de tráfico, de procesos y de requerimientos varía, por lo que sistemas que estaban operando dejan de operar porque terminan y aparecen otros, o porque se activan nodos con recursos y procesos que hacen falta, además, los nodos pueden estar activos, pero son incorporados al algoritmo cuando algún proceso solicita algún recurso, o algún recurso es solicitado por otro proceso de otro nodo. Un nodo puede estar activo, pero no formar parte de las evaluaciones de asignaciones.

El modelo propuesto logra establecer un consenso que permite a los grupos de procesos

acceder a todos sus recursos de manera secuencial y que éstos no puedan ser removidos hasta que el mismo grupo de procesos que los mantiene, los libera. El orden de asignación será determinado por la prioridad promedio general de todas las asignaciones de cada grupo. El sistema distribuido regula y actualiza constantemente el estado local de cada nodo, las decisiones de acceso a los recursos modifican estos estados por lo que debe ser reajustado repetidamente, garantizando la exclusión mutua y reordenando nuevas prioridades. El método debe repetirse siempre que haya grupos de procesos que requieran recursos compartidos.

Las características del Modelo de Decisión permiten evaluar las posibles alternativas y consecuencias y así poder definir los objetivos claramente. La mejor optimización ha sido lograda por seleccionar la mejor alternativa posible en casa caso particular. Como objetivo principal del modelo propuesto se consideró el entorno de ejecución distribuida de procesos, que podrían estar agrupados, el acceso a recursos compartidos se estableció de acuerdo a distintas exigencias de consenso, para generar la secuencia de asignación de los recursos a los procesos que los solicitan mediante el uso del método de agregación más adecuado a cada escenario posible, respetando la exclusión mutua en el acceso a dichos recursos. Se ha explicado que el modelo de decisión utiliza un *Runtime* que gestiona los procesos y recursos compartidos y define el escenario correspondiente. Los modelos tradicionales se han comparado mediante un cuadro comparativo y un ejemplo de aplicación con respecto al modelo propuesto y se ha comentado las consideraciones de los operadores de agregación desarrollados.

El método de agregación utilizado y la estructura de datos mencionada en este trabajo no están totalmente cubiertos por los métodos tradicionales, por ejemplo, no contemplan la predisposición (prioridad nodal), carga del nodo, el estado nodal (nodos, procesos, grupos, recursos.) ni el estado global del sistema, para el cálculo de prioridades en las asignaciones de

recursos a procesos.

Una importante característica por señalar de los operadores *Neat OWA* utilizados es que los valores a agregar no necesitan ser ordenados para su proceso. Esto implica que la formulación de un operador *neat* puede ser definida usando directamente los argumentos en lugar de los elementos ordenados.

En el operador de agregación propuesto los pesos se calculan en función de valores del contexto del cual surgen los valores a agregar.

El contenido de este capítulo sirvió de base para la siguiente publicación:

- Fornerón, J., Agostini, F. La Red Martínez, D. (2020). Modelo de decisión para gestión de recursos y procesos en sistemas distribuidos con balanceo dinámico de carga de trabajo. *Conferencia Ibero Americana WWW/Internet (CIAWI 2020)*. Lisboa, Portugal. (ENVIADO)

Capítulo VI - Conclusiones y futuras líneas de investigación

6.1. Conclusiones

Capítulo I

En el capítulo I se ha hecho la descripción de antecedentes, el marco teórico, se han planteado las hipótesis, se han contemplado situaciones en las que los procesos accedieron a recursos compartidos en la modalidad de exclusión mutua con o sin constituir grupos de procesos, que podían requerir o no sincronización y que podían tener o no exigencias estrictas de consenso para lograr el acceso.

Se han descripto las estructuras de datos, los elementos que intervienen dentro del sistema, los nodos, recursos, procesos, que tienen prioridades y cargas de trabajo y que en diferentes situaciones permitieran que el sistema se auto regule reiteradamente en función del estado local de los *n* nodos y del estado global del sistema.

Se ha podido demostrar que las estructuras de datos propuestas son apropiadas y suficientes para la descripción de los escenarios mencionados, y la aplicación de las soluciones presentadas, para el cumplimiento de los objetivos indicados.

Capítulo II

En el capítulo II se ha definido la premisa para el escenario general en el que los procesos accedieran a recursos compartidos en la modalidad de exclusión mutua pudiendo constituir grupos de procesos, los procesos no requerían sincronización (estar activos en sus respectivos procesadores en un mismo lapso de tiempo) y no tenían exigencias estrictas de consenso para lograr el acceso a los recursos.

Se ha explicado el operador de agregación para este escenario, que parte de La Red Martínez (2017), y se ha desarrollado un ejemplo con los datos provenientes del mismo, hasta obtener la tabla ***FASDLN*** final de asignaciones de recursos a procesos, la cual es utilizada como punto de partida para los siguientes capítulos.

Se ha demostrado que el operador de agregación propuesto es adecuado para el logro de los objetivos propuestos, permitiendo adecuar la solución propuesta en La Red Martínez (2017) a estructuras de datos difusas, generalizando así dicha solución.

Capítulo III

En el capítulo III se ha definido la premisa para el escenario 2, se buscó que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua pudiendo constituir grupos de procesos (los procesos independientes son considerados grupos unitarios); los procesos no requerían sincronización y tenían exigencias estrictas de consenso para lograr el acceso (se requería un consenso para asignar de manera consecutiva los recursos solicitados por un proceso o grupo de procesos, es decir, que iniciada la secuencia de asignación de recursos a un proceso, la misma no pudo ser interrumpida para asignar recursos a otro proceso).

Se ha presentado una variante de un método innovador para la gestión de recursos compartidos en sistemas distribuidos, basado en La Red Martínez (2017), La Red Martínez et al. (2018) y Agostini et al. (2019c).

Se ha establecido un modelo de consenso que favoreció el acceso secuencial de los procesos a todos los recursos solicitados. Las premisas, las estructuras de datos y el operador mencionado en el capítulo anterior se utilizaron como punto de partida para crear un nuevo operador en el escenario descrito en este capítulo. Utilizó la tabla ***FASDLNC***, que es la concatenación de las

tablas ***FASDLN*** ordenadas de todas las rondas del escenario general y consideró el promedio global de prioridades que cada proceso tuvo sobre todos los recursos de todas sus asignaciones en las diferentes rondas, para obtener la ***FASDLNCO***.

Se ha demostrado que la operatoria propuesta basada en etiquetas lingüísticas y 2-tuplas permite efectuar con precisión los cálculos requeridos para el logor de los objetivos planteados.

Capítulo IV

En el capítulo IV se han definido las premisas para el escenario 3, se buscó que los procesos accedan a recursos compartidos en la modalidad de exclusión mutua constituyendo grupos de procesos (los procesos independientes son considerados grupos unitarios); los procesos no requerían sincronización y tenían exigencias estrictas de consenso para lograr el acceso (se requirió un consenso para asignar de manera consecutiva los recursos solicitados por un grupo de procesos, es decir, que iniciada la secuencia de asignación de recursos al grupo, la misma no pudo ser interrumpida para asignar recursos a otro grupo).

Se ha presentado una variante de un método innovador para la gestión de recursos compartidos en sistemas distribuidos, basado en La Red Martínez (2017), La Red Martínez et al. (2018) y Agostini et al. (2019a y 2019b), donde se desarrolló un operador de agregación, para asignar recursos en sistemas distribuidos, pero en este caso, se ha establecido un modelo de consenso que favoreció el acceso secuencial de los procesos de cada grupo a todos los recursos solicitados.

Las premisas, las estructuras de datos y el operador mencionado en el segundo capítulo se utilizan como punto de partida para crear un nuevo operador. Utilizó la tabla ***FASDLNCO***, que es la concatenación de las tablas ***FASDLN*** ordenadas por procesos de todas las rondas del escenario

general y consideró el promedio global de prioridades que cada grupo de procesos tiene sobre todos los recursos de todas sus asignaciones en las diferentes rondas, pero para la asignación global final del grupo, estableció subrondas compatibles de asignación, dado que en un mismo grupo pudo haber procesos que requieran un mismo recurso, por lo tanto, los procesos de mayor prioridad fueron asignados en una primer subronda y los que generaban incompatibilidad se asignaron en una siguiente subronda y así obtener la ***FASDLNOGC.***

Se ha demostrado la viabilidad de la solución propueta, generalizando soluciones anteriores basadas en estructuras de datos numéricas, ampliándolas a estructuras difusas con el uso de conjuntos de etiquetas lingüística y 2-tuplas. Asimismo, se hace notar que la cardinalidad de los conjuntos de etiquetas lingüísticas utilizados en los distintos nodos puede ser diferente, ganando en generalidad y flexibilidad.

Capítulo V

Se han definido las características del Modelo de Decisión para permitir definir los objetivos claramente, para estudiar las posibles alternativas y evaluar sus consecuencias. Se ha explicado que el modelo de decisión utiliza un *Runtime* que gestiona los procesos y recursos compartidos y define el escenario correspondiente.

Además, se ha realizado un cuadro comparativo del modelo propuesto con respecto a los modelos tradicionales y se ha comentado las consideraciones de los operadores de agregación desarrollados, basándose en las características de los operadores *Neat OWA*.

Se han comparado las características y diferencias con respecto a los modelos tradicionales mediante un cuadro comparativo y un ejemplo de aplicación con respecto al modelo propuesto y se ha comentado las consideraciones de los operadores de agregación desarrollados.

Se ha demostrado la viabilidad del modelo de decisión propuesto y su mayor amplitud de criterio respecto de las solucione tradicionalmente aplicadas.

6.2. Futuras líneas de investigación

Se considera desarrollar modelos de decisión desde la óptica cognitiva para la toma de decisiones en grupos de procesos, contemplando los principios de la cibernética de segundo orden, en el contexto de sistemas complejos de auto-regulación, que trasciendan el enfoque tradicional de las ciencias de la computación considerando la posibilidad de imputación de datos faltantes, por ejemplo, como consecuencia de problemas en las comunicaciones entre los procesos. En la línea de investigación que involucra a imputación de datos, también se considerará lógica difusa, en base a la experiencia lograda en esta tesis.

Simulador

Se prevé desarrollar un simulador en el que se consideren los distintos escenarios posibles para permitir al sistema predecir, comparar y optimizar el comportamiento de sus procesos simulados en un tiempo muy breve sin el coste ni el riesgo de llevarlos a cabo, haciendo posible la representación de los procesos, recursos y nodos en un modelo dinámico. Con la ayuda del correspondiente soporte informático, el modelo de simulación permitirá la capacidad de considerar complejas tareas interrelacionadas y proyectarlas mediante la realización de muchas combinaciones alternativas en cuestión de segundos. Además, la interacción de los recursos con los procesos se traducirá en un gran número de escenarios y de posibles resultados imposibles de abarcar y valorar sin la ayuda de un modelo de simulación computarizado.

Seguridad

Considerar aspectos vinculados a la seguridad en la ejecución de los procesos, en el acceso

a los recursos y en la comunicación entre los nodos.

Migración controlada de procesos

Se consideraría la transferencia de procesos entre nodos de forma automática para hacer mejor uso de los recursos, característica que también está ligada a la distribución y el equilibrio de la carga de trabajo entre los nodos, contribuyendo a aumentar la tolerancia a fallos y así mejorar el rendimiento global del sistema distribuido, aportando a su autoregulación.

Referencias

Agostini F., Fornerón Martínez J. T., La Red Martínez D. L. (2019a). Nuevo operador de agregación para grupos de procesos. *16ª Conferencia Ibero Americana WWW/INTERNET 2019 (CIAWI)*. Lisboa, Portugal.

Agostini F., La Red Martínez D. L. (2019b). Allocation of shared resources. *14th Iberian Conference on Information Systems and Technologies - CISTI 2019*. ISBN N° 978-989-98434-9-3, pp. 1-6. Universidad de Coimbra; Coimbra, Portugal.

Agostini F., La Red Martínez D. L. y Acosta J. C. (2018). Modeling of the consensus in the allocation of resources in distributed systems. *International Journal of Advanced Computer Science and Applications (IJACSA), vol.* 9, n° 12. The Science and Information (SAI) Organization, England, U.K. http://dx.doi.org/10.14569/IJACSA.2018.091204.

Agostini F., La Red Martínez D. L. y Acosta J. C. (2019c). Assignment of Resources in Distributed Systems With Strict Consensus Requirements. *10th International Multi-Conference on Complexity, Informatics and Cybernetics (IMCIC 2019), vol.* 1, Orlando, Florida, USA.

Agrawal, D. y El Abbadi, A. (1991). An Efficient and Fault-Tolerant Solution of Distributed Mutual Exclusion. *ACM Trans. on Computer Systems, vol.* 9, pp. 1-20, USA.

Alagarsamy, K. (2003). Some Myths About Famous Mutual Exclusion Algorithms. *ACM SIGACT News 34 (3)*: 94-103.

Andrews, G. (2000). *Foundation of Multithreaded, Parallel, and Distributed Programming*. Reading, MA: Addison Wesley. USA.

Attiya, H. y Welch, J. (2004). *Distributed Computing Fundamentals, Simulations, and Advanced Topics*, John Wiley, 2nd ed., New York, USA.

Bagrodia, R. (1989). Process synchronization: design and performance evaluation of distributed

algorithms. *IEEE Transactions on Software Engineering, vol.* 15, issue 9, pp. 1053 - 1065.

Bateson, G. (1991). *Pasos Hacia Una Ecología de la Mente*. Planeta - Carlos Lohlé. Argentina.

Birman, K. P., Schiper, A. y Stephenson, P. (1991). Lightweight Causal and Atomic Group Multicast. *ACM Trans. on Computer Systems, vol.* 9, pp. 272-314. USA.

Birman, K. (2005). *Reliable Distributed Systems: Technologies. Web Services and Applications*. New York: Springer-Verlag.

Bouyssou D., Marchant T., Pirlot M., Tsoukias A. and Vincke P. (2000). Evaluation and decision models: a critical perspective. *Kluwer Academic Publishers International series in Operations Research & Management Sciences Massachusetts*. United States of America, *vol.* 32, pp. 262. ISBN 978-0-7923-7250-9.

Brix, V. H. (1970). *You are a Computer: Cybernetics in Everyday Life*. Emerson books.

Cao, G. y Singhal, M. (2001). A Delay-Optimal Quorum-Based Mutual Exclusion Algorithm for Distributed Systems. *IEEE Transactions on Parallel and Distributed Systems, vol.* 12, n°. 12, pp. 1256-1268. USA.

Cao, J., Zhou, J., Chen, D., Wu, J. (2004). An Efficient Distributed Mutual Exclusion Algorithm Based on Relative Consensus Voting. *Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04). IEEE*.

Carvalho de Barros, L., Bassanezi, R. C., Lodwick, W.A., (Ed.). (2017) A First Course in Fuzzy Logic, Fuzzy Dynamical Systems, and Biomathematics. Theory and Applications. *Studies in Fuzziness and Soft Computing*. Springer International Publishing. Berlín. ISSN 1860-0808 (electrónico).

Chao, X., Kou, G., Peng, Y. (2016). An optimization model integrating different preference formats. *6th International Conference on Computers Communications and Control*

(ICCCC), pp. 228 - 231.

Chen, C. T. (2001). Applying linguistic decision-making method to deal with service quality evaluation problems. *International Journal of Uncertainty, Fuzzyness and Knowledge-Based Systems*, 9 (Suppl.): 103-114.

Chiclana, F., Herrera, F., Herrera-Viedma, E. (2000). The Ordered Weighted Geometric Operator: Properties And Application. *Proc. of 8th International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems*, pp. 985-991. España.

Chiclana, F., Herrera, F., Herrera-Viedma, E. (2001). Integrating Multiplicative Preference Relations In A Multipurpose Decision Making Model Based On Fuzzy Preference Relations. Fuzzy Sets and Systems. 112: 277-291.

Chiclana, F., Herrera-Viedma, E., Herrera, F., Alonso, S. (2004). Induced Ordered Weighted Geometric Operators and Their Use in the Aggregation of Multiplicative Preferences Relations. *International Journal of Intelligent Systems. 19*: 233-255.

Cicirelli, F., & Nigro, L. (2016). Modelling and Verification of Mutual Exclusion Algorithms. *IEEE/ACM 20th International Symposium on Distributed Simulation and Real Time Applications*, pp. 136-144.

Colouris, G., Dollimore, J. y Kindberg, T. (1988). *Distributed Systems. Concepts and Design*. Pearson.

Collan, M., Fedrizzi, M. y Kacprzyk, J., (Ed.). (2016) Fuzzy Technology. Present Applications and Future Challenges. *Studies in Fuzziness and Soft Computing*. Springer International Publishing. Suiza. ISSN 1860-0808 (electrónico).

Dong, Y., Zhang, H. y Herrera-Viedma, E. (2016). Consensus reaching model in the complex and

dynamic MAGDM problema. *Knowledge-Based Systems, vol.* 106, pp. 206-219. Elsevier.

Doña, J. M., Gil, A. M., Peláez, J. I. y La Red Martínez, D. L. (2011). A System Based on the Concept of Linguistic Majority for the Companies Valuation. *Revista Econo Quantum, vol.* 8, n° 2, pp. 121-142. México.

Dutta, B., Labella, A., Rodríguez, R.M., Martínez, L. (2019). Aggregating Interrelated Attributes in Multi-Attribute Decision-Making With ELICIT Information Based on Bonferroni Mean and Its Variants. *International Journal of Computational Intelligence Systems, vol.* 12, n° 2, pp. 1179-1196. DOI: https://doi.org/10.2991/ijcis.d.190930.002.

Fitzek F. y Katz M. (2014). *MOBILE CLOUDS: Exploiting Distributed Resources in Wireless, Mobile and Social Networks*. L. John Wiley & Sons, ed., New Delhi, India. ISBN: 978-0-470-97389-9.

Fodor, J. C., Roubens, M. (1994). *Fuzzy Preference Modelling and Multicriteria Decision Support*. Kluwer, Dordrecht.

Fullér, R. (1996). OWA Operators in Decision Making. En *Carlsson, C. ed. Exploring the Limits of Support Systems, TUCS General Publications*, n° 3, Turku Centre for Computer Science, pp. 85-104.

Gaertner W. (2009). *A Primer in Social Choice Theory Revised*, Oxford: Oxford University Press. ISBN: 9780199565306.

Gómez-Cruz N. y Maldonado C. (2011). Sistemas bio-inspirados: Un marco teórico para la ingeniería de sistemas complejos. *Documentos de Investigación, Escuela de Administración. Bogotá,* n° 112, pp. 1-24. ISSN: 0124-8219.

Greco, S.; Matarazzo, B., Slowinski, R. (2002). Rough sets methodology for sorting problems in presence of multiple attributes and criteria. *European Journal of Operational Research,*

vol. 138, pp. 247-259.

Guerraoui, R. y Rodrigues, L. (2006). *Introduction to Reliable Distributed Programming. Berlin*, Springer-Verlag.

Herrera, F. y Martinez. L. (2000). An Approach For Combining Linguistic And Numerical Information Based On The 2-Tupla Fuzzy Linguistic Representation Model In Decision-Making. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol.* 8, n° 5, pp. 539-562.

Herrera, F. y Martínez, L. (2000). A 2-Tuple Fuzzy Linguistic Representation Model for Computing with Words. *IEEE Transactions On Fuzzy Systems, vol.* 8, n° 6, pp. 746-752.

Herrera, F., Herrera-Viedma, E., Alonso, S., Chiclana, F. (2009). 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

Jiménez, G.E. y Zulueta, Y. (2017). A 2-tupla linguistic multi-period decision making approach for dynamic green supplier selection. *DYNA, 84*(202), pp. 199-206.

Joseph, T.A. y Birman, K.P. (1989). Reliable Broadcast Protocols, en *Distributed Systems. Mullender*, S. (Ed). ACM Press. USA.

Joshi, R., Holzmann, G. J. (2007). A Mini-Challenge: Build a Verifiable Filesystem, *Formal Aspects of Computing, vol.* 19.

Kaashoek, M. K. (1992). Group Communication In Distributed Computer Systems, *Centrale Huisdrukkerij Vrije Universiteit*, Amsterdam.

Kacprzyk, J., Filev, D., Beliakov, G., (Ed.). (2017). Granular, Soft and Fuzzy Approaches for Intelligent Systems. *Studies in Fuzziness and Soft Computing*. Springer International Publishing, Suiza. ISSN 1860-0808 (electrónico).

Kamei, S., Kakugawa, H. (2017). An asynchronous message-passing distributed algorithm for the global critical section problem. *Fifth International Symposium on Computing and Networking*, pp. 414-419.

Lamport, L. (1978). Time, Clocks, and the Ordering of Events in a Distributed System, *Communications of the ACM, vol.* 21, n° 7, pp. 558-565.

La Red Martínez, D. L. (2004). *Sistemas Operativos*. EUDENE. Argentina.

La Red Martínez, D. L., Doña, J. M., Peláez, J. I., Fernández, E. B. (2011). WKC-OWA, a New Neat-OWA Operator to Aggregate Information in Democratic Decision Problems. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, World Scientific Publishing Company, págs. 759-779. Francia.

La Red Martínez, D. L. & Acosta, J. C. (2014). Perspectives of New Decision Making Models of Processes Synchronization in Distributed Systems, *vol.* 09, n° 1, *International Journal of Management and Information Technologies*, pp. 1504-1512, ISSN N° 2278-5612, U.S.A.

La Red Martínez, D. L., Acosta, J. C. (2015a). Review of Modeling Preferences for Decision Models, Volume 11 - N° 36, *European Scientific Journal (ESJ)*, pp. 1-18, ISSN N° 1857-7881, Macedonia.

La Red Martínez, D. L., Acosta, C. (2015b). Aggregation Operators Review - *Mathematical Properties and Behavioral Measures; vol.* 7, n° 10; *International Journal of Intelligent Systems and Applications (IJISA)*; pp. 63-76; Hong Kong.

La Red Martínez, David L. (2017). Aggregation Operator for Assignment of Resources in Distributed Systems, *(IJACSA) International Journal of Advanced Computer Science and Applications, vol.* 8, n° 10.

La Red Martínez, D. L., Agostini, F., Primorac, C. (2018). Nueva propuesta para la administración

de recursos y procesos en sistemas distribuidos. *Workshop de Investigadores en Ciencias de la Computación, WICC.* ISBN 978-987-3619-27-4. Corrientes, Argentina.

Lejeune, J., Arantes, L., Sopena, J. y Sens, P. (2013). A prioritized distributed mutual exclusion algorithm balancing priority inversions and response time. *42nd International Conference on Parallel Processing*, pp. 290-299.

Lejeune, J., Arantes, L., Sopena, J. y Sens, P. (2012). Service Level Agreement for Distributed Mutual Exclusion in Cloud Computing. *12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pp. 180-187. IEEE.

Lejeune, J., Arantes, L., Sopena, J., y Sens, P. (2015). A fair starvation-free prioritized mutual exclusion algorithm for distributed systems. *J. Parallel Distrib. Comput.*, *vol.* 83, pp. 13-29. Elsevier.

Lin, S.-D., Lian, Q., Chen, M. y Zhang, Z., (2004). 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 Computer Sc., (La Jolla, CA). Springer-Verlag, Berlin.

Lodha, S. y Kshemkalyani, A. (2000). A Fair Distributed Mutual Exclusion Algorithm. *IEEE Trans. Parallel and Distributed Systems, vol.* 11, n° 6, pp. 537-549, USA.

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

Loo J., Lloret Mauri, J., Ortiz J. (2011). *Mobile Ad Hoc Networks*. Boca Raton: CRC Press. https://doi.org/10.1201/b11447.

Lu, J.; Zhang, G.; Ruan, D., Wu, F. (2007). *Multi-objective Group Decision Making: Methods,*

Software and Applications with Fuzzy Set Technology. London: Imperial College Press.

Lynch, N. (1996). *Distributed Algorithms*. Morgan Kaufmann, San Mateo, CA, USA.

Macedonia, M. R., Zyda, M. J., Pratt, D. R., Brutzman, D. P., Barham, P. T. (1995). Exploiting Reality with Multicast Groups: A Network Architecture for Large-scale Virtual Environments. *Proc. of IEEE VRAIS* (RTP, NC, Mar., 1995), pp. 2-10.

Maher A. y Maysam F. (2016). Classifiers consensus system approach for credit scoring. *Knowledge Based Systems, vol.* 104, pp. 89-105. ISSN: 0950- 7051. DOI:10.1016/j.knosys.2016.04.013.

Marakas, G. (2002). *Decision Support Systems*. New Jersey, USA. Prentice Hall.

Martínez, L., Liu, J., Yang, J. B. (2006). A fuzzy model for design evaluation based on multiple criteria analysis in engineering systems. *International Journal of Uncertainty. Fuzziness and Knowledge Based Systems. 14(3)*: 317-336.

Martínez, L., Liu, J., Ruan, D., Yang, J. B. (2007). Dealing with heterogeneous information in engineering evaluation processes. *Information Sciences. 177(7)*: 15331542.

Meier, A., Pedrycz, W., Portmann, E., (2017). The Application of Fuzzy Logic for Managerial Decision Making Processes. *Fuzzy Management Methods*. Springer International Publishing, Suiza. ISSN 2196-4149

Mendel, Jerry M. (2007). Computing with Words: Zadeh, Turing, Popper and Occam, *IEEE Computational Intelligence Magazine*, pp. 10-17.

Naimi, M., Thiare, O. (2013). A Distributed Deadlock-Free Quorum-Based Algorithm for Mutual Exclusion. *(IJCSIS) International Journal of Computer Science and Information Security, vol.* 11, n° 8, pp. 7-13.

Neamatollahi, P., Taheri, H., Naghibzadeh, M. (2012). Info-based approach in distributed mutual

exclusion algorithms. *J. Parallel Distrib. Comput., vol.* 72, pp. 650-665. Elsevier.

Parsegian V. L. (1973). *This Cybernetic World of Men, Machines, and Earth Systems*. Anchor Books: Garden City, New York.

Peláez, J.I., Doña, J.M. (2003). Majority Additive-Ordered Weighting Averaging: A New Neat Ordered Weighting Averaging Operators Based on the Majority Process. *International Journal of Intelligent Systems, vol* 18, n° 4, pp. 469-481.

Peláez, J.I., Doña, J.M., La Red Martínez, D.L., Mesas, A. (2004). Majority Opinion in Group Decision Making Using the QMA-OWA Operator. *Proceeding of ESTYLF*, pp. 449-454.

Peláez, J.I., Doña, J.M., Gómez-Ruiz, J.A. (2007). Analysis of OWA Operators in Decision Making for Modelling the Majority Concept. *Applied Mathematics and Computation, vol.* 186, pp. 1263-1275.

Peláez, J.I., Doña, J.M., La Red Martínez, D.L. (2009). A Mix Model of Discounted Cash-Flow and OWA Operators for Strategic Valuation. *Revista Interactive Multimedia and Artificial Intelligence - Special Issue On Business Intelligence And Semantic Web*, *vol.* 1, n° 2, pp. 20-25. España.

Petrosino, A., Loia, V., Pedrycz, W., (Ed.). (2016) Fuzzy Logic and Soft Computing Applications. *The 11th. International Workshop on Fuzzy Logic and Applications, WILF 2016*. Springer International Publishing. Suiza. ISSN 1611-3349 (electrónico).

Pissinou, N., Makki, K., Park, E. K., Hu, Z., Wong, W. (1996). An Efficient Distributed Mutual Exclusion Algorithm. *1996 International Conference on Parallel Processing*, pp. 196-203.

Pitt, J., Busquets, D. and Riveret, R. (2015). The pursuit of computational justice in open systems. *AI & Soc., vol.* 30, n° 3, pp. 359-378. https://doi.org/10.1007/s00146-013-0531-6.

Raymond, K. (1989). A tree-based algorithm for distributed mutual exclusions. *ACM Transactions*

on Computer Systems, vol. 7, n° 1, pp. 61-77.

Razzaque, A. y Hong, C. S. (2008). Multi-Token Distributed Mutual Exclusion Algorithm. *22nd. International Conference on Advanced Information Networking and Applications*, pp. 963-970. IEEE.

Ricart, G. y Agrawala, A. K. (1981). An Optimal Algorithm for Mutual Exclusion in Computer Networks. *Commun. of the ACM, vol.* 24, pp. 9-17.

Ríos, D. A. y La Red Martínez, D. L. (2019). Nuevo Modelo de Decisión Para Gestión de Tráfico en Redes. *International Journal Of Information Systems And software Engineering For Big Companies (IJISEBC), vol.* 6, n° 2, pp. 99-122.

Rodrigues, L. A., Cohen, J., Arantes, L. y Duarte Jr., E. P. (2013). Robust Permission-Based Hierarchical Distributed k-Mutual Exclusion Algorithm. *IEEE 12th International Symposium on Parallel and Distributed Computing*, pp. 151-158. IEEE.

Sakamoto, S. y Ogata, K. (2018). Model Checking of the Suzuki-Kasami Distributed Mutual Exclusion Algorithm with SPIN. *5th International Conference on Dependable Systems and Their Applications (DSA)*, pp. 136-141.

Saxena, P. y Rai, J. (2003). A Survey of Permission-based Distributed Mutual Exclusion Algorithms. *Computer Standards and Interfaces, vol.* 25, *n° 2*, pp 159-181.

Sha, L., Rajkumar, R. y Lehoczky, J. P. (1990). Priority inheritance protocols: An approach to real-time synchronization. Computers, *IEEE Transactions on, vol.* 39, n° 9, pp. 1175-1185.

Singh, H., Gupta, M., Meitzler, T., Hou, Z., Garg, K., Solo, A., y Zadeh, L. A. (2013). Real-Life Applications of Fuzzy Logic. *Hindawi Publishing Corporation. Advances in Fuzzy Systems*, Volume 2013, Article ID 581879. doi:10.1155/2013/581879

Silberschatz, A., Galvin, P.B. y Gagne, G. (2006). *Fundamentos de Sistemas Operativos*. 7ma.

Edición. McGraw-Hill / Interamericana de España. S.A.U. España.

Stallings, W. (2005). *Sistemas Operativos*. 5ta. Edición. Pearson Educación S.A., España.

Taheri, H, Neamatollahi, P. Naghibzadeh, M. (2011). A hybrid token-based distributed mutual exclusion algorithm using wraparound two-dimensional array logical topology. *Information Processing Letters 111*, pp. 841-847. Elsevier.

Tanenbaum, A. S. (1996). *Sistemas Operativos Distribuidos*. Prentice - Hall Hispanoamericana S.A., México.

Tanenbaum, A. S. (2009). *Sistemas Operativos Modernos*. 3ra. Edición. Pearson Educación S. A., México.

Tanenbaum, A. S., Van Steen, M. (2008). *Sistemas Distribuidos - Principios y Paradigmas*. 2da. Edición. Pearson Educación S. A. México.

Tao, Z.F., Chen, H.Y., Zhou, L.G., Liu, J.P. (2015). 2-Tuple linguistic soft set and its application to group decision making. *Soft Comput., vol.* 19, n° 5, pp. 1201-1213.

Tel G. (2000). *Introduction to Distributed Algorithms*. Cambridge University Press, 2nd ed. Cambridge, UK.

Toh C. K., Kim D., Oh S. and Yoo H. (2010). The controversy of Selfish nodes in ad hoc networks. *The 12th International Conference on Advanced Communication Technology (ICACT)*, Phoenix Park, pp. 1087-1092.

Varela, F. J. (1990). *Conocer - Las Ciencias Cognitivas: Tendencias y Perspectivas - Cartografía de las Ideas Actuales*. Gedisa Editorial. España.

Velázquez, M. (1993). A Survey of Distributed Mutual Exclusion Algorithms. *Technical Report* CS-93-116, University of Colorado at Boulder.

Wen, T., Chang, K. y Lai, H. (2020). Integrating the 2-tupla linguistic representation and soft set

to solve supplier selection problems with incomplete information, *Engineering Applications of Artificial Intelligence 87*.

Wiener, N. (1961). *Cybernetics,* The M.I.T Press: Cambridge, Massachusetts, 2nd ed.

Wiener, N. (1985). *Cibernética o el Control y la Comunicación en Animales y Máquinas*. 2a Edición. Tusquets. España.

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

Yager, R. (1993). Families Of OWA Operators. *Fuzzy Sets and Systems, vol.* 59, pp. 125148.

Yager, R., Kacprzyk, J. (1997). *The Ordered Weighted Averaging Operators. Theory And Applications*. Kluwer Academic Publishers. USA.

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

Ying, M. (2002). A Formal Model of Computing With Words. *IEEE Transactions On Fuzzy Systems, vol.* 10, n° 5.

Yuan He, K. Gopalakrishnan, Eli Gafni. (2018). Group mutual exclusion in linear time and space. *Theoretical Computer Science, vol.* 709, pp. 31-47. Elsevier.

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

Zadeh, L. A. (1988). Fuzzy logic. *Computer*, 21(4), 83-93

Zadeh, L. A. (1996). Fuzzy Logic = Computing with Words, *IEEE Transactions On Fuzzy*

Systems, vol. 4, n° 2, pp. 103-111.

Zadeh, L.A. (1999). From Computing with Numbers to Computing with Words—From Manipulation of Measurements to Manipulation of Perceptions. *IEEE Transactions on Circuits and Systems - I: Fundamental Theory and Applications*, Vol. 45, no. 1

Zadeh, L. A. (2008). Is there a need for fuzzy logic?. *Information sciences*, 178(13), 2751-2779. doi:10.1016/j.ins.2008.02.012

Zapata, S., Fuentealba, D. y Valenzuela, G. (2015). Aplicación del modelo de representación de información lingüística 2-tuplas con información multigranular. *Revista Trilogía: Ciencia, Tecnología y Sociedad, vol.* 27, nº 37, pp. 110-127. Facultad de Ingeniería UTEM.

Zhang, H.D., Xiong, L.L., Ma, W.Y. (2015). On interval-valued hesitant fuzzy soft sets. *Math. Probl. Eng.* 254764.

Zhang, X., Xu, Z. (2017). Hesitant Fuzzy Methods for Multiple Criteria Decision Analysis. *Studies in Fuzziness and Soft Computing*. Springer International Publishing, Suiza. ISSN 1860-0808

Apéndice de publicaciones

1. Agostini F., La Red Martínez D. L., Fornerón Martínez J. T. (2019). Nuevo Operador de Agregación para Grupos de Procesos. Conferencias IADIS (International Association for Development of the Information Society). *Ibero-Americanas WWW/Internet y Computación Aplicada*. ISBN N° 978-989-8533-96-8, pp. 223-230. Lisboa, Portugal. (PUBLICADO)

2. Fornerón, J., Agostini, F., La Red Martínez, D. (2020). Resource and Process Management with a Decision Model based on Fuzzy Logic. *World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI 2020)*. (ACEPTADO)

3. Fornerón, J., Agostini, F., La Red Martínez, D. (2020). Load Balancing in Distributed Systems with Fuzzy Logic. *International Journal of Interactive Multimedia and Artificial Intelligence (IJIMAI)*. (ENVIADO)

4. Fornerón, J., Agostini, F. La Red Martínez, D. (2020). Gestión de procesos basado en lógica difusa con estrictos niveles de consenso. *International Journal of Information Systems and Software Engineering for Big Companies (IJISEBC).* (ENVIADO)

5. Fornerón, J., Agostini, F. La Red Martínez, D. (2020). Modelo de decisión para gestión de recursos y procesos en sistemas distribuidos con balanceo dinámico de carga de trabajo. *Conferencia Ibero Americana WWW/Internet (CIAWI 2020)*. (ENVIADO)