

Planificadores de tareas en tiempo real concurrentes: Una clasificación basada en funciones y teoría de conjuntos

Pedro Guevara López, Jorge Salvador Valdez Martínez y Gustavo Delgado Reyes

Sección de Estudios de Posgrado e Investigación,
Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Culhuacán,
Instituto Politécnico Nacional, D.F.,
México

pguevara@ipn.mx, {jsvaldezmtzgzdelgador1200}@alumno.ipn.mx

Resumen. Los Sistemas Operativos en Tiempo Real deben brindar soporte para concurrencia, para lograrlo se requieren de los Planificadores de tareas. Los planificadores operan sobre un Conjunto de Tareas en Tiempo Real Concurrentes donde sus instancias requieren ejecutarse hasta completarse dentro de sus plazos máximos; el planificador recibe un conjunto de Tiempos de Arribo y lo mapea hacia un Conjunto de Tiempos de Inicio para que las instancias empiecen su ejecución. En este contexto, un planificador se considera una función que mapea entre dos conjuntos que evolucionan en el tiempo, en este sentido, se presenta una clasificación de los planificadores basada en funciones y teoría de conjuntos siendo: críticos, no críticos, estáticos, adaptativos, predictivos, por desalojo de prioridades y óptimos. Esta propuesta de clasificación es novedosa ya que en el actual estado del arte solo se presentan clasificaciones verbales no formales y no aportan elementos que ayuden a su análisis, modelado y/o caracterización. Como resultado adicional, esta clasificación podrá ser utilizada para realizar futuros estudios cualitativos en optimalidad, estabilidad, controlabilidad, eficiencia, convergencia y predecibilidad desde el punto de vista computacional. Al final del documento se clasifican dos ejemplos de planificadores: RM (Rate Monotonic) y EDF (Earliest Deadline First).

Palabras clave. Tiempos de arribo, función, tiempos de inicio, planificador, clasificación, sistema en tiempo real.

Concurrent Real-Time Task Schedulers: A Classification Based on Functions and Set Theory

Abstract. Real-Time Operating Systems must provide support for concurrency; to achieve this, task schedulers are necessary. Schedulers operate on a set

of concurrent real-time tasks in which its instances are to complete their execution within their respective deadlines; a scheduler receives a set of arrival times and maps it to a set of start times for the instances to begin their execution. In this context, a scheduler is considered to be a function that performs mapping between two sets which evolve in time, with respect to this we present a classification of schedulers, based on functions and set theory, into such categories as critical, non-critical, static, adaptive, predictive, preemptive, and optimal. Our proposed classification is novel because the state of the art classifications are only verbal and non-formal and they do not support elements which assist in analysis, modeling and/or characterization of schedulers. As an additional result, this classification can be used for future qualitative studies of optimality, stability, controllability, efficiency, convergence, and predictability from the computational point of view. The paper concludes with two examples of schedulers: RM (Rate Monotonic) and EDF (Earliest Deadline First).

Keywords. Arrival time, function, starting time, scheduler, classification, real-time system.

1. Introducción

Según [1, 2, 3, 4, [5], una tarea es un conjunto $\{J_i, i=1, \dots, p\}$ de actividades básicas que se ejecutan en un sistema operativo que cumplen con ciertas características de acuerdo a [5, 6]. La unidad de trabajo de una tarea se llama instancia, de tal manera que una tarea en tiempo real está formada por un conjunto de instancias, es decir $J_i = \{j_{i,k}\}$, con $i = (1, \dots, p)$, $k = (1, \dots, n)$ para $i, k, n, p \in \mathbb{Z}^+$, donde i es el índice de la tarea y p es el número de tareas, t es el índice de la instancia y n es el número de instancias.

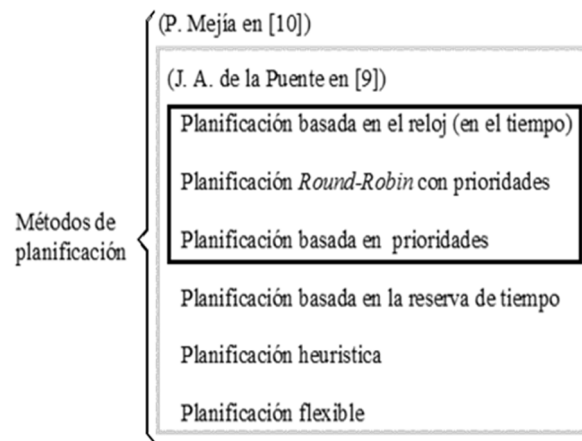


Fig. 1. Clasificación de los planificadores para tareas en tiempo real concurrentes por su método de planificación

Cuando un procesador tiene que ejecutar un conjunto de tareas concurrentes, estas pueden demandar al mismo tiempo los recursos de hardware y software del sistema operativo, ocasionando conflictos en su ejecución, por esta razón el CPU tiene que asignar los recursos a las tareas existentes de acuerdo a un criterio predefinido llamado política de planificación. De esta manera el conjunto de reglas que en cualquier tiempo determina el orden en el cual las tareas son ejecutadas son conocidas como algoritmo de planificación [1–5, 7].

En general, la investigación que gira en torno a los sistemas en tiempo real considera a los planificadores como algoritmos para asignación de tareas al procesador encontrándose varios ejemplos tal y como son: *Rate Monotonic* (RM), *FIFO*, *Deadline Monotonic* (DM), *Earliest Deadline First* (EDF), *Least Laxity First* (LLF), *Shortest Slack Time First* (SSTF). De los cuales, se pueden encontrar muchas definiciones y clasificaciones.

De acuerdo a en [8] un algoritmo de planificación es “un conjunto de reglas que determinan que tarea va a ser ejecutada en un tiempo determinado”. Otra definición de planificador de tareas en tiempo real concurrentes es la que proporcionó en [7]; en este trabajo se establece que “una planificación

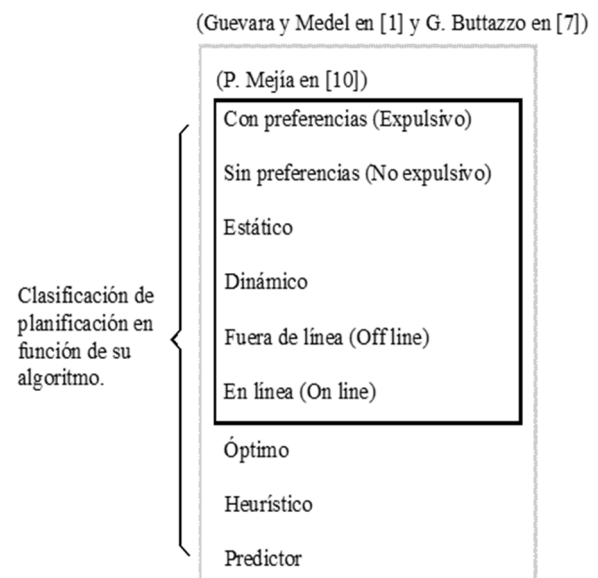


Fig. 2. Clasificación de los planificadores para tareas en tiempo real concurrentes de acuerdo a su algoritmo

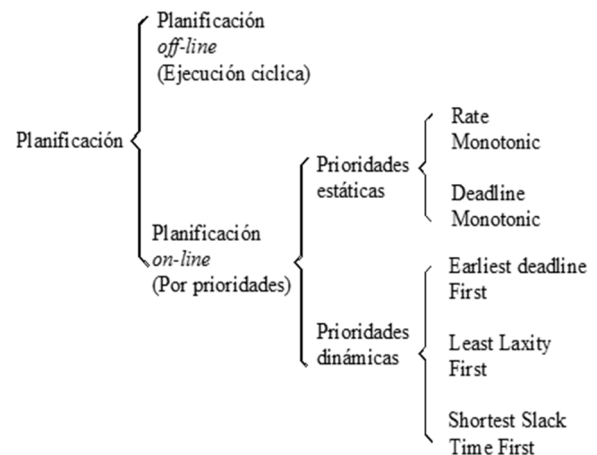


Fig. 3. Clasificación general de los planificadores para tareas en tiempo real concurrentes

es una asignación de tareas al procesador donde cada tarea es ejecutada hasta completarse”. De manera más formal se dice que:

$$\sigma = \mathbf{R}^+ \rightarrow \mathbf{N} \text{ tal que } \forall t \in \mathbf{R}^+ \exists t_1, t_2 \text{ donde } t \in [t_1, t_2) \text{ y } \forall t' \in [t_1, t_2) \sigma(t) = \sigma(t'), \quad (1)$$

donde $\alpha(t)$ es definida como una función escalón y $\alpha(t)=k$, con $k>0$, significa que la tarea J_i es ejecutada en t , cuando $\alpha(t)=0$ significa que el CPU está inactivo.

De forma general, todo planificador de tareas en tiempo real concurrentes se puede definir como “un algoritmo que asigna los recursos del procesador a diferentes tareas y en diferentes momentos”.

En cuanto a las clasificaciones de planificadores para tareas en tiempo real concurrentes halladas en el estado del arte del presente trabajo, algunas se relacionan con su método de planificación, y otras más en función de su algoritmo. La propuesta dada por J. A. de la Puente en [9], es en función de su método de planificación, la cual es extendida por P. Mejía en [10] y se puede apreciar en la figura 1.

Mientras que [1, 7] coinciden en clasificar a los planificadores en función de su algoritmo. P. Mejía sub-clasifica a los planificadores de tareas en tiempo real concurrentes de acuerdo al algoritmo [1, 10], aunque coincide con ellos solo en una parte, como se muestra en la figura 2.

Pero la mayoría de los autores consultados (Mejía en [10], Crespo y Alonso en [11], González en [12], Moncusi en [13], Pastor en [14], Vedia en [15] y Yepez en [16]) los clasifican por planificación en línea (online) y fuera de línea (offline). Esto se puede apreciar en la figura 3.

Se observa en el estado de arte que las clasificaciones mostradas del planificador de tareas en tiempo real no son formales y no aportan elementos que ayuden a su análisis, modelado y/o caracterización, en este contexto, la principal aportación en este trabajo es presentar una clasificación original basada en funciones y teoría de conjuntos a partir de la definición del planificador en tiempo real mostrada en [19] que es una publicación previa de los autores. La propuesta de clasificación presentada en este trabajo podrá ser utilizada para realizar estudios cualitativos en optimalidad, estabilidad, controlabilidad, eficiencia, convergencia y predecibilidad desde el punto de vista computacional.

2. Desarrollo

2.1. Tareas en tiempo real (TTR)

Formalmente se define a una tarea en tiempo real (TTR) como una entidad ejecutable de trabajo J_i que al menos es caracterizada por un tiempo de arribo y una restricción temporal. La instancia $j_{i,k}$ de una tarea en tiempo real está definida mínimamente con $j_{i,k} = (l_{i,k}, c_{i,k}, d_{i,k})$ con $i, k \in \mathbb{Z}^+$, donde el tiempo de arribo absoluto $l_{i,k} \in \mathbb{R}^+$ de una instancia $j_{i,k}$ de una TTR J_i está definido como el tiempo en que la instancia pide atención al procesador en relación con el origen temporal de referencia. El tiempo de ejecución $c_{i,k} \in \mathbb{R}^+$, es el tiempo en que la instancia $j_{i,k}$ de una TTR J_i concluye sus operaciones, sin considerar los desalojos de ésta en el procesador. El plazo máximo $d_{i,k} \in \mathbb{R}^+$, es el límite temporal superior en relación con el origen temporal de referencia, antes del cual la instancia $j_{i,k}$ debe finalizar; cuando se trata de tareas periódicas, $d_{i,k}$ crece de manera monótonica tal que $d_{i,k} = d_{i,k-1} + T_i$ donde T_i es el periodo de activación de la instancia $j_{i,k}$ [17]. Donde $l_{i,k}$, $c_{i,k}$, $d_{i,k}$ y T_i están dados en unidades temporales [UT].

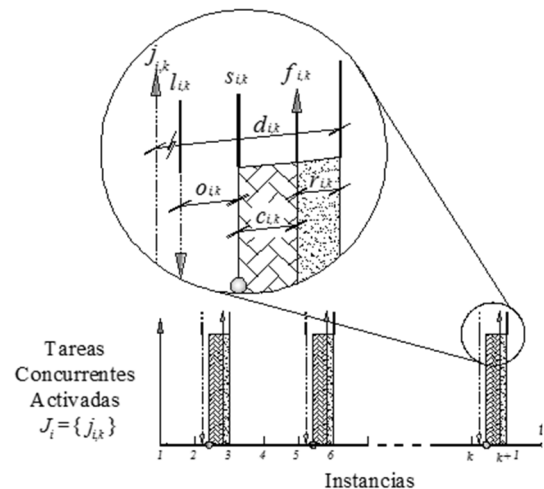


Fig. 4. Restricciones temporales de la instancia $j_{i,k}$ de la tarea en tiempo real J_i : tiempo de arribo $l_{i,k}$, tiempo de inicio $s_{i,k}$, tiempo de finalización $f_{i,k}$, tiempo de ejecución $c_{i,k}$ (región ashurada a la derecha), tiempo de operación $o_{i,k}$ y plazo $d_{i,k}$ (región ashurada a la izquierda), tiempo excedente $r_{i,k}$ (región ashurada a la derecha), tiempo de operación $o_{i,k}$ y plazo $d_{i,k}$.

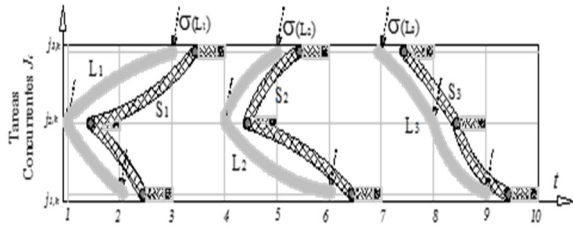


Fig. 5. Conjuntos de tiempos de arribo L_k (región en color gris), el conjunto de tiempos de inicio S_k (región ashurada) y el planificador de tareas en tiempo real concurrentes (L_k)

De acuerdo a [18], la instancia $j_{i,k}$ de una tarea en tiempo real se define por la quintupla $j_{i,k} = (l_{i,k}, s_{i,k}, c_{i,k}, f_{i,k}, d_{i,k})$ con $i, k \in \mathbf{Z}^+$, siendo $s_{i,k} \in \mathbf{R}^+$ el tiempo de inicio de una instancia $j_{i,k}$ de una TTR J_i definido como el instante en que la instancia es atendida por el procesador con relación al origen temporal de referencia. Y $f_{i,k} \in \mathbf{R}^+$ es el tiempo de finalizado de una instancia $j_{i,k}$ de una TTR J_i definido como el instante en que la instancia es terminada por el procesador con relación al origen temporal de referencia, donde $s_{i,k}$ y $f_{i,k}$ están dados en unidades temporales [UT].

En este trabajo, se agregan tres restricciones temporales más para describir a la instancia de la tarea en tiempo real mediante la óctupla $j_{i,k} = (l_{i,k}, o_{i,k}, s_{i,k}, c_{i,k}, f_{i,k}, r_{i,k}, h_{i,k}, d_{i,k})$ con $i, k \in \mathbf{Z}^+$. Esta óctupla se puede apreciar en la figura 4.

Siendo $o_{i,k} \in \mathbf{R}^+$ el tiempo de la instancia $j_{i,k}$ de la TTR J_i con $o_{i,k} = s_{i,k} - l_{i,k} \forall i, k, p \in \mathbf{Z}^+$. El tiempo excedente es $r_{i,k} \in \mathbf{R}^+$ y se considera como la diferencia existente entre el tiempo de finalizado $f_{i,k}$ y el plazo máximo absoluto $d_{i,k}$ de la instancia $j_{i,k}$ de una TTR J_i , esto es $r_{i,k} = f_{i,k} - d_{i,k}$, para que la instancia no pierda su plazo deberá cumplirse $r_{i,k} \leq 0$, con $i, k, n, p \in \mathbf{Z}^+$. Además se define a $h_{i,k} \in \mathbf{R}^+$ es el tiempo de holgura y se define como el valor negativo de $h_{i,k} = -r_{i,k}$. Donde $o_{i,k}$, $r_{i,k}$ y $h_{i,k}$ están dados en unidades temporales [UT].

2.2. El Concepto de planificador de tareas en tiempo real concurrentes

Considerando la figura 4, J es el Conjunto de Tareas en Tiempo Real Concurrentes $J = \{J_1, \dots, J_i, \dots, J_p\}$, formado por al menos p Tareas en

Tiempo Real que compiten por q recursos o procesadores con la restricción de que $p > q$. L_k es el Conjunto de Tiempos de Arribo en el índice k de las $j_{i,k}$ instancias: $L_k = \{l_{1,k}, \dots, l_{i,k}, \dots, l_{p,k}\}$. S_k es el conjunto de Tiempos de Inicio en el índice k de las $j_{i,k}$ instancias: $S_k = \{s_{1,k}, \dots, s_{i,k}, \dots, s_{p,k}\}$. O_k es el conjunto de Tiempos de Operación en el índice k de las $j_{i,k}$ instancias: $O_k = \{o_{1,k}, \dots, o_{i,k}, \dots, o_{p,k}\}$. C_k es el conjunto de Tiempos de Ejecución en el índice k de las $j_{i,k}$ instancias: $C_k = \{c_{1,k}, \dots, c_{i,k}, \dots, c_{p,k}\}$. F_k es el Conjunto de Tiempos de Finalizado en el índice k de las $j_{i,k}$ instancias: $F_k = \{f_{1,k}, \dots, f_{i,k}, \dots, f_{p,k}\}$. D_k es el Conjunto de Plazos Máximos en el índice k de las $j_{i,k}$ instancias: $D_k = \{d_{1,k}, \dots, d_{i,k}, \dots, d_{p,k}\}$. R_k es el conjunto de Tiempos Excedentes en el índice k de las $j_{i,k}$ instancias: $R_k = \{r_{1,k}, \dots, r_{i,k}, \dots, r_{p,k}\}$. H_k es el conjunto de Tiempos de Holgura en el índice k de las $j_{i,k}$ instancias: $H_k = \{h_{1,k}, \dots, h_{i,k}, \dots, h_{p,k}\}$. Entonces con base en [1]-[5], [17] y [19] se presenta la siguiente descripción de lo que es un planificador: “Un Planificador de Tareas en Tiempo real Concurrentes está descrito como una función $\sigma(L_k)$ que mapea del conjunto de tiempos de arribo L_k al conjunto de tiempos de inicio S_k ”. Esto es:

$$\sigma(L_k): L_k \rightarrow S_k, \text{ con } k \in \mathbf{Z}^+ \quad (2)$$

De acuerdo con esto, cualquier Planificador de Tareas en Tiempo Real Concurrentes debe cumplir con (2). La figura 5., presenta un esquema representativo de un planificador de tiempo real; las tres regiones en color gris representan los conjuntos de tiempos de arribo L_k , mientras que las tres regiones ashuradas son los conjuntos de tiempos de inicio S_k .

2.3. Clasificación de los planificadores de tareas en tiempo real concurrentes

Los planificadores deben buscar cumplir con la restricción $f_{i,k} \leq d_{i,k} \forall f_{i,k} \in F_k$ y $d_{i,k} \in D_k$. No se considera como una condición suficiente mapear del conjunto L_k al conjunto S_k , además es necesario buscar el cumplimiento de los plazos de cada instancia $j_{i,k}$ sin embargo esto es difícil, considerando que el problema de la planificación de tareas tiene complejidad computacional temporal del tipo NP completo [20]. Como resultado de las definiciones descritas en [19] se

proponen las siguientes definiciones de planificadores para tareas en tiempo real concurrentes:

- Planificador Crítico para Tareas en Tiempo Real Concurrentes. Es aquel planificador con la función $\sigma(L_k)$ tal que $\#(L_k) = \#(S_k)$ y $\#(F_k) = \#(D_k) \forall k \in \mathbb{Z}^+$.
- Planificador No Crítico para Tareas en Tiempo Real Concurrentes. Es aquel planificador con la función $\sigma(L_k)$ que tal que $\text{card}(L_k) \geq \text{card}(S_k)$ y $\exists f_{i,k} > d_{i,k} \forall k \in \mathbb{Z}^+$.
- Planificador Estático para Tareas en Tiempo Real Concurrentes. Es aquel planificador con la función $\sigma(L_k)$ que nunca modifica su algoritmo de planificación sin importar la dinámica cambiante del conjunto L_k .
- Planificador Adaptativo para Tareas en Tiempo Real Concurrentes. Es aquel planificador con la función $\sigma(L_k)$ que modifica su algoritmo de planificación de acuerdo a la dinámica cambiante del conjunto L_k .
- Planificador Predictivo para Tareas en Tiempo Real Concurrentes. Es aquel planificador tal que la función $\sigma(L_k)$ mapea del conjunto de tiempos de arribo L_k al conjunto de tiempos de inicio S_{k+h} con $h, k \in \mathbb{Z}^+$; esto es $\sigma(L_k): L_k \rightarrow S_{k+h} \forall k$; i.e., el planificador en el índice k , $\sigma(L_k)$ es capaz de planificar para el índice $k+h$.
- Prioridad de ejecución $\alpha_{i,k}$ de toda instancia $j_{i,k} \in J_i$. Es el nivel de preferencia para ejecutarse respecto a la instancia $j_{g,h} \in J_u$ con prioridad $\alpha_{g,h}$ tal que $\alpha_{i,k} \neq \alpha_{g,h}$ y $g, h, i, k, \alpha_{i,k}, \alpha_{g,h} \in \mathbb{Z}^+$ siempre que $j_{i,k}$ y $j_{g,h}$ se ejecuten en el intervalo $(l_{i,k}, d_{i,k}]$.
- Planificador $\sigma(L_k)$ por Desalojo de Prioridades (preemptive). Es aquel planificador donde una instancia $j_{i,k}$ y prioridad $\alpha_{i,k}$ cumple con las siguientes consideraciones:

- a. $\exists l_{i,k} \geq s_{i,k}$ donde $s_{i,k} \in S_k$ y $l_{i,k} \in L_k$
 $\exists s_{i,k} + c_{i,k} \geq f_{i,k}$ donde $s_{i,k} \in S_k$, $c_{i,k} \in C_k$ y $f_{i,k} \in F_k$
- b. debido a que una instancia $j_{g,h}$ con prioridad $\alpha_{g,h}$ y $\alpha_{g,h} > \alpha_{i,k}$ y $g \neq i$ es ejecutada en el intervalo $(l_{i,k}, d_{i,k}]$

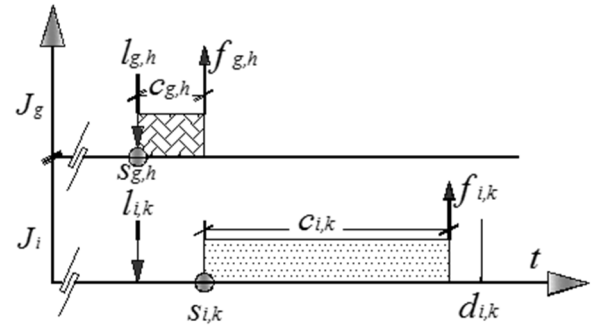


Fig. 6. Gráfica donde se muestra que la instancia $j_{i,k}$ desplazada desde $l_{i,k}$ hasta $s_{i,k}$ debido a que se ejecuta la instancia $j_{g,h}$ con mayor prioridad, cumpliendo con la primera parte de la descripción del planificador con prioridad de ejecución

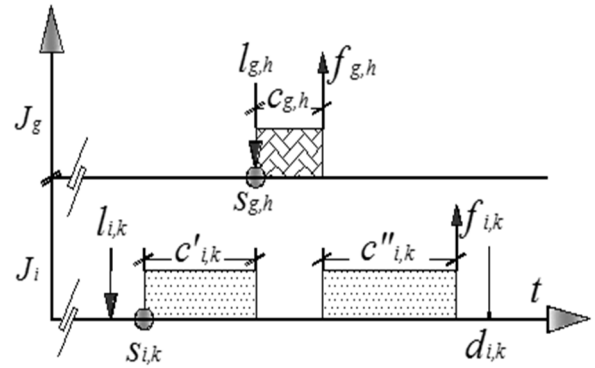


Fig. 7. Gráfica donde se muestra que la instancia $j_{i,k}$ es segmentada en dos partes: $c'_{i,k}$ y $c''_{i,k}$ durante su ejecución debido a que se ejecuta la instancia $j_{g,h}$ con mayor prioridad, cumpliendo con la descripción del planificador por desalojo de prioridades

desplazando o segmentando la ejecución de $j_{i,k}$ donde $\forall \alpha, i, k, g, h \in \mathbb{Z}^+$.

- c. $f_{i,k} \leq 1/2f_{\max}$, donde f_{\max} es la frecuencia máxima de operación que solicita la atención del sistema real.
- d. Entonces, suponiendo que $j_{i,k}$ y $j_{g,h}$ arriban al mismo tiempo $l_{i,k} = l_{g,h}$ con $\alpha_{g,h} > \alpha_{i,k}$, se debe mostrar que el inicio de la instancia $j_{i,k}$ se desplaza de su arribo: $s_{i,k} > l_{i,k}$. Como la instancia $j_{g,h}$ tiene mayor prioridad que la instancia $j_{i,k}$, al arribar $j_{g,h}$ comienza a ejecutarse,

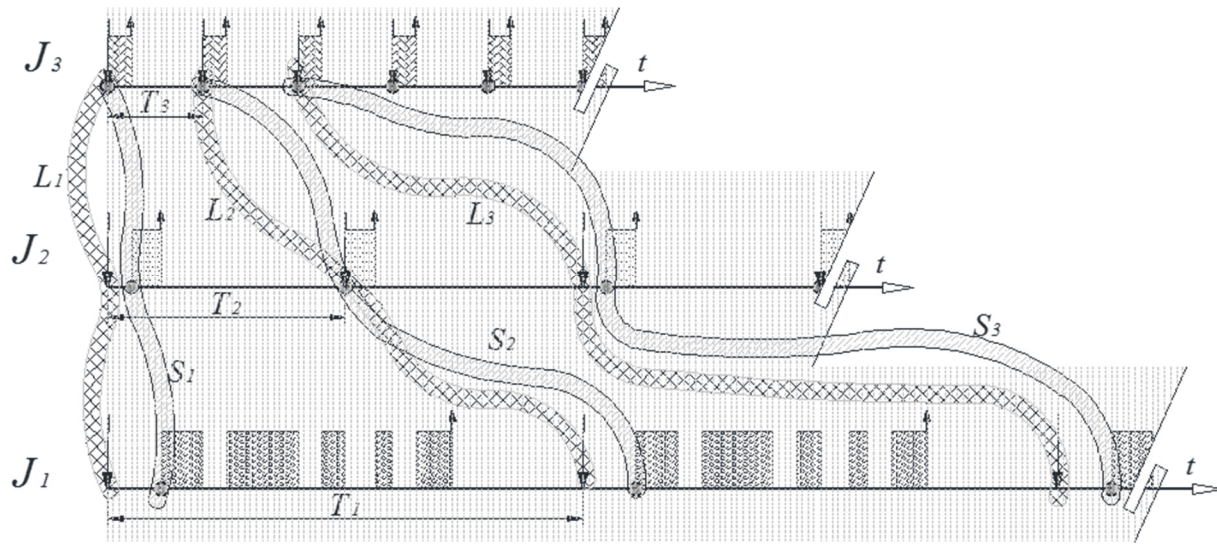


Fig. 8. Gráfica donde se observa la evolución temporal de J_1 , J_2 y J_3 a lo igual que sus respectivas instancias por el efecto del planificador Rate Monotonic. Los rectángulos ashurados representan las instancias (o segmentos de éstas), las regiones irregulares ashuradas con líneas gruesas son los conjuntos de tiempos de arribo L_1 , L_2 y L_3 , de manera similar las regiones irregulares ashuradas con líneas delgadas son los conjuntos de tiempos de inicio S_1 , S_2 y S_3 .

obteniendo $s_{g,h}=l_{g,h}$ y haciendo que la ejecución de $j_{i,k}$ se desplace una distancia de $c_{g,h}$ tal que $s_{i,k}=l_{i,k}+c_{g,h}$. Por lo tanto $s_{i,k} \geq l_{i,k}$. Esto se puede observar en la figura 6.

Suponiendo que $s_{g,h} \in (l_{i,k}, d_{i,k}]$ y $\alpha_{g,h} > \alpha_{i,k}$ se debe mostrar que $f_{i,k} \geq s_{i,k}+c_{i,k}$. Como la instancia $j_{g,h}$ tiene mayor prioridad que la instancia $j_{i,k}$, entonces $s_{g,h}=l_{g,h}$ y el tiempo de finalizado se desplaza $f_{i,k}=s_{i,k}+c_{i,k}+c_{g,h}$. Por lo tanto $f_{i,k} \geq s_{i,k}+c_{i,k}$. Esto se observa en la figura 7.

Cabe mencionar que esta clasificación es estrictamente cualitativa y descriptiva; las características cuantitativas dependerán del tipo de algoritmo de planificación que se esté analizando, en específico para los casos adaptativo y predictivo.

2.4 Planificación óptima para tareas en tiempo real concurrentes

El problema de la planificación óptima para un conjunto de planificadores de tareas en tiempo real concurrentes $\Sigma_k=\{\sigma(L_k), \sigma'(L_k), \sigma''(L_k)\}$

pretende la minimización del valor máximo del conjunto de tiempos de operación y la maximización del valor mínimo del conjunto de tiempos de holgura en el índice k para un planificador $\sigma(L_k)$; esto es: $\min(\max(O_k))$ y $\max(\min(H_k)) \forall k \in \mathbb{Z}^+$. En este contexto se propone el concepto de Planificador Óptimo:

- Planificador Óptimo para Tareas en Tiempo Real Concurrentes. En un planificador cuya única función $\sigma(L_k)$ que comparada con cualquier $\sigma'(L_k)$ cumple: $\max(O_k) < \max(O'_k)$ y $\min(H_k) > \min(H'_k)$ con $\forall k \in \mathbb{Z}^+$. Este es el único planificador capaz de mandar a ejecutar todas las instancias del conjunto de tareas en tiempo real concurrentes J , minimizando la diferencia entre tiempos de inicio $l_{i,k}$ y tiempos de arribo $s_{i,k}$ y maximizando la diferencia entre plazos máximos $d_{i,k}$ y tiempos de finalizado $f_{i,k}$ durante toda la evolución del sistema.

Todo este estudio de clasificación podrá servir de base para futuros análisis de:

Optimalidad. De acuerdo al Principio de Optimalidad de Bellman, que dice: "Cualquier

Tabla 1. Valores del conjunto de tareas concurrentes $J = \{J_1, J_2, J_3\}$ para el planificador RM

Tarea	K	$l_{1,k}$	$s_{1,k}$	$c_{1,k}$	T_1	$f_{1,k}$	$\alpha_{1,k}$	$o_{1,k}$	$r_{1,k}$	$U_{1,k}$
Tarea J_1	1	0	9	32	80	58	1	9	-22	0.400
	2	80	89	32	80	138	1	9	-22	0.400
	3	160	169	32	80	218	1	9	-22	0.400
Tarea	k	$l_{2,k}$	$s_{2,k}$	$c_{2,k}$	T_2	$f_{2,k}$	$\alpha_{2,k}$	$o_{2,k}$	$r_{2,k}$	$U_{2,k}$
Tarea J_2	1	0	4	5	40	9	2	4	-31	0.125
	2	40	40	5	40	45	2	0	-35	0.125
	3	80	84	5	40	89	2	4	-31	0.125
Tarea	k	$l_{3,k}$	$s_{3,k}$	$c_{3,k}$	T_3	$f_{3,k}$	$\alpha_{3,k}$	$o_{3,k}$	$r_{3,k}$	$U_{3,k}$
Tarea J_3	1	0	0	4	16	4	3	0	-12	0.250
	2	16	16	4	16	20	3	0	-12	0.250
	3	32	32	4	16	36	3	0	-12	0.250

Tabla 2. Conjuntos de restricciones temporales observados para el planificador RM

L_k	$\sigma(L_k)$	S_k
$L_1 = \{l_{1,1}, l_{2,1}, l_{3,1}\} = \{0, 0, 0\}$	$\sigma(L_1): L_1 \rightarrow S_1;$	$S_1 = \{s_{1,1}, s_{2,1}, s_{3,1}\} = \{9, 4, 0\}$
$L_2 = \{l_{1,2}, l_{2,2}, l_{3,2}\} = \{80, 40, 16\}$	$\sigma(L_2): L_2 \rightarrow S_2;$	$S_2 = \{s_{1,2}, s_{2,2}, s_{3,2}\} = \{89, 40, 16\}$
$L_3 = \{l_{1,3}, l_{2,3}, l_{3,3}\} = \{160, 80, 32\}$	$\sigma(L_3): L_3 \rightarrow S_3;$	$S_3 = \{s_{1,3}, s_{2,3}, s_{3,3}\} = \{169, 84, 32\}$
C_k	F_k	D_k
$C_1 = \{c_{1,1}, c_{2,1}, c_{3,1}\} = \{32, 5, 4\}$	$F_1 = \{f_{1,1}, f_{2,1}, f_{3,1}\} = \{58, 9, 4\}$	$D_1 = \{d_{1,1}, d_{2,1}, d_{3,1}\} = \{80, 40, 16\}$
$C_2 = \{c_{1,2}, c_{2,2}, c_{3,2}\} = \{32, 5, 4\}$	$F_2 = \{f_{1,2}, f_{2,2}, f_{3,2}\} = \{138, 45, 20\}$	$D_2 = \{d_{1,2}, d_{2,2}, d_{3,2}\} = \{160, 80, 32\}$
$C_3 = \{c_{1,3}, c_{2,3}, c_{3,3}\} = \{32, 5, 4\}$	$F_3 = \{f_{1,3}, f_{2,3}, f_{3,3}\} = \{218, 89, 36\}$	$D_3 = \{d_{1,3}, d_{2,3}, d_{3,3}\} = \{240, 120, 48\}$
O_k	R_k	H_k
$O_1 = \{o_{1,1}, o_{2,1}, o_{3,1}\} = \{9, 4, 0\}$	$R_1 = \{r_{1,1}, r_{2,1}, r_{3,1}\} = \{-22, -31, -12\}$	$H_1 = \{h_{1,1}, h_{2,1}, h_{3,1}\} = \{22, 31, 12\}$
$O_2 = \{o_{1,2}, o_{2,2}, o_{3,2}\} = \{9, 0, 0\}$	$R_2 = \{r_{1,2}, r_{2,2}, r_{3,2}\} = \{-22, -35, -12\}$	$H_2 = \{h_{1,2}, h_{2,2}, h_{3,2}\} = \{22, 35, 12\}$
$O_3 = \{o_{1,3}, o_{2,3}, o_{3,3}\} = \{9, 4, 0\}$	$R_3 = \{r_{1,3}, r_{2,3}, r_{3,3}\} = \{-22, -31, -12\}$	$H_3 = \{h_{1,3}, h_{2,3}, h_{3,3}\} = \{22, 31, 12\}$

subsecuencia de decisiones de una secuencia óptima de decisiones que resuelve un problema también debe ser óptima respecto al subproblema que resuelve” [22].

Esto es: Si un algoritmo de planificación es óptimo, sus subalgoritmos de planificación serán óptimos también. En otras palabras, si $\sigma(L_k)$ es óptimo para un conjunto de tareas concurrentes J , entonces será óptimo para cualquier subconjunto $J' \in J$.

Estabilidad. De acuerdo con [23], se refiere a la vaciación continua de los algoritmos de planificación durante la evolución del sistema computacional. Esta condición involucra a los planificadores estáticos y adaptativos, como ejemplo del primer caso puede mencionarse el algoritmo RM, mientras que para el segundo puede ser el algoritmo EDF.

Controlabilidad. En [24] se describen las condiciones de controlabilidad de un planificador, se explica que la meta del control de desempeño

Tabla 3. Valores del conjunto de tareas concurrentes $J = \{J_1, J_2, J_3\}$ para el planificador EDF

Tarea J_1		k	$h_{1,k}$	$s_{1,k}$	$c_{1,k}$	T_1	$f_{1,k}$	$\alpha_{1,k}$	$o_{1,k}$	$r_{1,k}$	$U_{1,k}$
		1	0	9	32	80	53	1	9	-27	0.400
		2	80	89	32	80	133	1	9	-27	0.400
		3	160	169	32	80	213	1	9	-27	0.400
Tarea J_2		k	$h_{2,k}$	$s_{2,k}$	$c_{2,k}$	T_2	$f_{2,k}$	$\alpha_{2,k}$	$o_{2,k}$	$r_{2,k}$	$U_{2,k}$
		1	0	4	5	40	9	2	4	-31	0.125
		2	40	53	5	40	58	1	13	-22	0.125
		3	80	84	5	40	89	2	4	-31	0.125
Tarea J_3		k	$h_{3,k}$	$s_{3,k}$	$c_{3,k}$	T_3	$f_{3,k}$	$\alpha_{3,k}$	$o_{3,k}$	$r_{3,k}$	$U_{3,k}$
		1	0	0	4	16	4	3	0	-12	0.250
		2	16	16	4	16	20	3	0	-12	0.250
		3	32	32	4	16	36	3	0	-12	0.250

Tabla 4. Conjuntos de restricciones temporales observados para el planificador EDF

L_k	$\sigma(L_k)$	S_k
$L_1 = \{l_{1,1}, l_{2,1}, l_{3,1}\} = \{0, 0, 0\}$	$\sigma(L_1): L_1 \rightarrow S_1;$	$S_1 = \{s_{1,1}, s_{2,1}, s_{3,1}\} = \{9, 4, 0\}$
$L_2 = \{l_{1,2}, l_{2,2}, l_{3,2}\} = \{80, 40, 16\}$	$\sigma(L_2): L_2 \rightarrow S_2;$	$S_2 = \{s_{1,2}, s_{2,2}, s_{3,2}\} = \{89, 53, 16\}$
$L_3 = \{l_{1,3}, l_{2,3}, l_{3,3}\} = \{160, 80, 32\}$	$\sigma(L_3): L_3 \rightarrow S_3;$	$S_3 = \{s_{1,3}, s_{2,3}, s_{3,3}\} = \{169, 84, 32\}$
C_k	F_k	D_k
$C_1 = \{c_{1,1}, c_{2,1}, c_{3,1}\} = \{32, 5, 4\}$	$F_1 = \{f_{1,1}, f_{2,1}, f_{3,1}\} = \{53, 9, 4\}$	$D_1 = \{d_{1,1}, d_{2,1}, d_{3,1}\} = \{80, 40, 16\}$
$C_2 = \{c_{1,2}, c_{2,2}, c_{3,2}\} = \{32, 5, 4\}$	$F_2 = \{f_{1,2}, f_{2,2}, f_{3,2}\} = \{133, 58, 20\}$	$D_2 = \{d_{1,2}, d_{2,2}, d_{3,2}\} = \{160, 80, 32\}$
$C_3 = \{c_{1,3}, c_{2,3}, c_{3,3}\} = \{32, 5, 4\}$	$F_3 = \{f_{1,3}, f_{2,3}, f_{3,3}\} = \{213, 89, 36\}$	$D_3 = \{d_{1,3}, d_{2,3}, d_{3,3}\} = \{240, 120, 48\}$
O_k	R_k	H_k
$O_1 = \{o_{1,1}, o_{2,1}, o_{3,1}\} = \{9, 4, 0\}$	$R_1 = \{r_{1,1}, r_{2,1}, r_{3,1}\} = \{-27, -31, -12\}$	$H_1 = \{h_{1,1}, h_{2,1}, h_{3,1}\} = \{27, 31, 12\}$
$O_2 = \{o_{1,2}, o_{2,2}, o_{3,2}\} = \{9, 13, 0\}$	$R_2 = \{r_{1,2}, r_{2,2}, r_{3,2}\} = \{-27, -22, -12\}$	$H_2 = \{h_{1,2}, h_{2,2}, h_{3,2}\} = \{27, 22, 12\}$
$O_3 = \{o_{1,3}, o_{2,3}, o_{3,3}\} = \{9, 4, 0\}$	$R_3 = \{r_{1,3}, r_{2,3}, r_{3,3}\} = \{-27, -31, -12\}$	$H_3 = \{h_{1,3}, h_{2,3}, h_{3,3}\} = \{27, 31, 12\}$

de los planificadores es definir un conjunto de rangos entre diferentes tipos de cargas computacionales o usuarios; en algunas ocasiones el control deseado puede alcanzarse y en otras no es posible. A las condiciones para alcanzar el control deseado se le llaman condiciones de controlabilidad del desempeño del planificador. Esta característica va ligada al desempeño del algoritmo de planificación y en general involucra a los planificadores adaptativos y por desalojo de prioridades.

Eficiencia. Con base en [7, 21], la eficiencia se mide a través del factor de utilización U , se define como la capacidad de un algoritmo de planificación para ejecutar un conjunto de tareas

en tiempo real concurrentes sin que éstas pierdan su plazo máximo de ejecución. Esta condición va dirigida a la obtención de la máxima eficiencia, en su caso a la obtención de un planificador óptimo.

Convergencia. En [25] se hace referencia a la velocidad del algoritmo de planificación, ésta afecta directamente a su implementación, es decir, afecta la velocidad en que se obtienen sus resultados y esto es esencial en los sistemas en tiempo real. La planificación de tareas es un problema NP completo, por ello encontrar la mejor solución implica una gran complejidad computacional temporal, por ello en ocasiones se busca la velocidad de respuesta del algoritmo

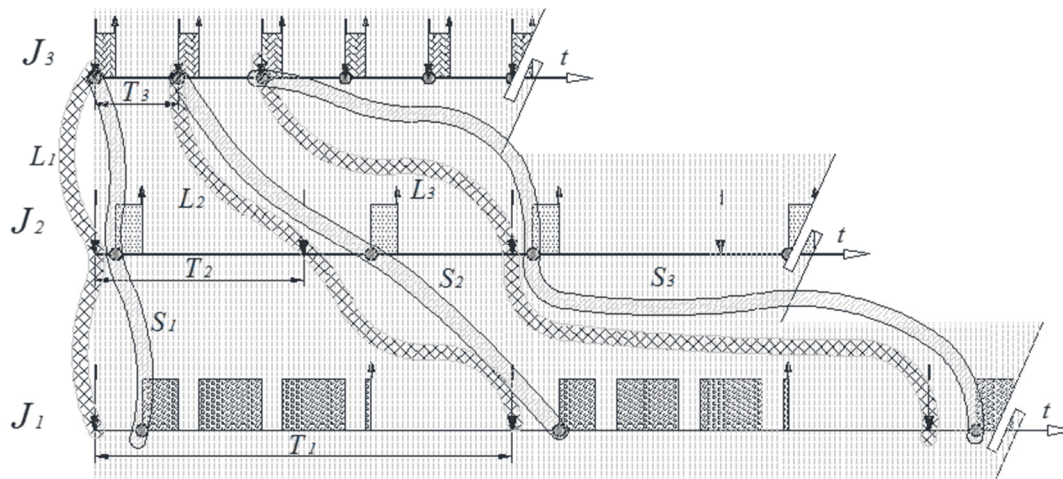


Fig. 9 Gráfico donde se observa la evolución temporal de J_1 , J_2 y J_3 a lo igual que sus respectivas instancias por el efecto del planificador Earliest Deadline First. Los rectángulos ashurados representan las instancias (o segmentos de éstas), las regiones irregulares ashuradas con líneas gruesas son los conjuntos de tiempos de arribo L_1 , L_2 y L_3 , de manera similar las regiones irregulares ashuradas con líneas delgadas son los conjuntos de tiempos de inicio S_1 , S_2 y S_3 .

para cumplir con los plazos máximos de ejecución.

Predecibilidad. La predecibilidad, respecto a [26] es una propiedad del sistema en tiempo real para predecir la puntualidad de la ejecución de las instancias dentro de sus plazos máximos de ejecución antes de que el sistema inicie su operación; para ello se requieren pruebas de planificabilidad y análisis de las restricciones temporales. En este contexto, los planificadores predictivos tratan de dar una respuesta a los posibles arribos de instancias futuras, esto dependerá del algoritmo de predicción que se utilice, que en ocasiones pudiera obtenerse a través de un filtro digital [4].

3. Clasificación de dos planificadores para tareas en tiempo real concurrentes

Para demostrar la utilidad de la clasificación de los planificadores de tareas en tiempo real concurrentes desarrollada en este trabajo, se propone analizar dos planificadores: el planificador de tasa monotonica (RM por sus

siglas en ingles de *Rate Monotonic*) y el planificador de primer plazo cumplido (EDF por sus siglas en ingles de *Earliest Deadline First*) los cuales planificarán un conjunto de tareas $J = \{J_1, J_2, J_3\}$ periódicas concurrentes definidas por :

$$\begin{aligned} J_1 &\text{ con } c_{1,k} = 32, T_1 = 80; \\ J_2 &\text{ con } c_{2,k} = 5, T_2 = 40; \\ J_3 &\text{ con } c_{3,k} = 4, T_3 = 16. \end{aligned} \quad (3)$$

El planificador RM fue desarrollado en [8], cuya máxima prioridad de ejecución $\alpha_{i,k}$ será estática de inicio para las tareas que tengan el periodo de activación T_i de la instancia $j_{i,k}$ más pequeño; y de acuerdo a la clasificación mostrada en este documento (figura 3), los autores lo clasifican como planificador *on-line* con prioridades estáticas. El planificador EDF también es expuesto en [8] aunque con un nombre diferente: Algoritmo de planificación de manejo de plazos (*Deadline Driven Scheduling Algorithm*), cuya máxima prioridad de ejecución $\alpha_{i,k}$ será dinámica a lo largo de la evolución del sistema para las tareas que tengan los plazos relativos $d_{i,k}$ más cortos. Y es clasificado de acuerdo a lo

revisado en el estado de arte (figura 3), como planificador *on-line* con prioridades dinámicas.

En [21], establece que la real ventaja del planificador RM con respecto al planificador EDF es su simpleza de implementación en *kernels* de sistemas operativos comerciales que no provean de soporte explícito para restricciones de tiempo. Otras propiedades del este planificador es su predictibilidad en condiciones de sobrecarga o mejor control de *jitter* exclusivamente para tareas con mayor prioridad. Mientras que el planificador EDF permite el uso del tiempo del procesador $U_{i,k}$ en la ejecución del conjunto de tareas, el cual implica una mejor explotación de los recursos computacionales y una mejor respuesta para tareas aperiódicas. Estas propiedades llegan a ser muy importantes para sistemas embebidos que trabajan con recursos computacionales limitados y para sistemas multimedia. Por último, ambos planificadores no son apropiados para trabajar en condiciones de sobrecarga y para lograr control de *jitter*.

3.1. Caracterización y clasificación del planificador de tareas concurrente RM

A partir de los datos mostrados en (3), se obtienen los valores de las restricciones temporales de las instancias $j_{i,k}$ de las tareas en tiempo real concurrentes J_i a partir de la planificación hecha mediante el algoritmo RM los cuales son mostrados en la tabla 1.

Donde $l_{i,k}$, $s_{i,k}$, $c_{i,k}$, T_i , $f_{i,k}$, $o_{i,k}$ y $r_{i,k}$ están dados en unidades temporales [UT], la asignación de prioridades $\alpha_{i,k}$ no tiene unidades y $U_{i,k}$ es la fracción de uso del tiempo del procesador en la ejecución del conjunto de tareas, el cual se obtiene de [1-5 y 9] y se muestra en (4):

$$U_k = c_i^k T_k^i \quad (4)$$

Con $T_k^i := (T_i^k)^{-1}$ Cuya evolución del conjunto de restricciones temporales de las instancias del conjunto de tareas en tiempo real concurrentes J usando el planificador *Rate Monotonic* se puede observar en la figura 8, que junto con los valores de la tabla 1, se obtienen los conjuntos de restricciones temporales mostrados en la tabla 2.

De acuerdo a la obtención de los conjuntos anteriores y a las definiciones presentadas en este trabajo, el algoritmo *Rate Monotonic* denotado por $\sigma(L_k)$ se le considerara de acuerdo a la definición 2 como Planificador Crítico para Tareas en Tiempo Real Concurrentes ya que $\#(L_k) = \#(S_k)$ y $\#(F_k) = \#(D_k) \forall k \in \mathbf{Z}^+$; se le considerará de acuerdo a la definición como Planificador Estático para Tareas en Tiempo Real Concurrentes ya que $\sigma(L_k)$ nunca cambia durante su evolución y por consecuencia no se le considera como Planificador Adaptativo para Tareas en Tiempo Real Concurrentes; por último cabe mencionar que de acuerdo es un Planificador por Desalojo de Prioridades, debido a que una instancia con alta prioridad se ejecuta desalojando a una instancia de menor prioridad. No es predictivo ni tampoco óptimo debido a que no cumplen con las definiciones.

3.2. Caracterización y clasificación del planificador de tareas concurrente EDF

Considerando los datos mostrados en (3), se obtienen los valores de las restricciones temporales de las instancias $j_{i,k}$ de las tareas en tiempo real concurrentes J_i mediante el uso del algoritmo de planificación EDF, los cuales son mostrados en la tabla 3. Los valores de las restricciones temporales se pueden observar gráficamente en la figura 9.

Mediante los valores de la tabla 3 así como del gráfico mostrado en la figura 9, se obtienen los conjuntos de restricciones temporales que son mostrados en la tabla 4.

Entonces teniendo en consideración la propuesta de clasificación de este trabajo, el planificador EDF se le considerará por la definición 2 como Planificador Crítico para Tareas en Tiempo real Concurrentes pues es una función $\sigma(L_k)$ tal que $\#(L_k) = \#(S_k)$ y $\#(F_k) = \#(D_k) \forall k \in \mathbf{Z}^+$. También se le considerará por la definición 5 como Planificador Adaptativo para Tareas en Tiempo Real Concurrentes, pues es una función $\sigma(L_k)$ que modifica su algoritmo de planificación y lo adapta a la dinámica cambiante del conjunto L_k . Y considerando la prioridad de ejecución, se afirma que el planificador EDF es un Planificador por Desalojo de Prioridades ya que

en una instancia $j_{i,k}$ y prioridad $\alpha_{i,k}$ cumple con las siguientes consideraciones: $\exists l_{i,k} \geq s_{i,k}$ donde $s_{i,k} \in S_k$ y $l_{i,k} \in L_k$ o $\exists s_{i,k} + c_{i,k} \geq f_{i,k}$ donde $s_{i,k} \in S_k$, $c_{i,k} \in C_k$ y $f_{i,k} \in F_k$, debido a que una instancia $j_{g,h}$ con prioridad $\alpha_{g,h}$ y $\alpha_{g,h} > \alpha_{i,k}$ y $g \neq i$ es ejecutada en el intervalo $(l_{i,k}, d_{i,k}]$ desplazando o segmentando la ejecución de $j_{i,k}$ donde $\forall \alpha, i, k, g, h \in \mathbb{Z}^+$. No se clasifica como predictivo y para poder clasificarlo como óptimo se necesita comparar con otros planificadores existentes.

4. Conclusiones

Todo planificador para tareas en tiempo real concurrentes se define informalmente como un conjunto de reglas que determinan que tarea va a ser ejecutada en un tiempo determinado, sin embargo, esta descripción no permite hacer un estudio de sus cualidades; esto sirvió de motivación para que de acuerdo a [19] se considere la definición formal de todo planificador como una función σ que mapea del conjunto de tiempos de arribo L_k al conjunto de tiempos de inicio S_k en un conjunto de tareas en tiempo real concurrentes J . Partiendo de esto, se describe el concepto de prioridad $\alpha_{i,k}$, y se plantean los conjuntos de restricciones temporales de toda tarea en tiempo real: $L_k, O_k, S_k, C_k, R_k, H_k$ y D_k .

La principal aportación en este trabajo es presentar una clasificación original basada en funciones y teoría de conjuntos basándose en la definición de planificador antes mencionada. Con base en esta propuesta es posible clasificar a los planificadores en: crítico, no crítico, estático, adaptativo, predictivo, por desalojo de prioridades y óptimo; estas características podrán ser usadas para realizar estudios de optimalidad, estabilidad, controlabilidad, eficiencia, convergencia y predecibilidad desde el punto de vista computacional.

En el análisis de los dos casos de planificadores: Rate Monotonic y Earliest Deadline First se hace una clasificación utilizando con los conceptos de este trabajo, la clasificación impide ambigüedades y permite dar información cualitativa adicional para cada algoritmo.

Referencias

1. Cruz, P.D., Medel, J.J., & Guevara, P. (2009). Modelo para Tiempos de Arribo de Tareas en Tiempo Real Concurrentes. *Computación y Sistemas*, Vol.12, No. 4, pp. 460–474.
2. Guevara, P. & Medel, J.J. (2003). *Introducción a los Sistemas de Tiempo Real*. Primera edición, Editorial Politécnico, México, D.F., pp. 12–13.
3. Guevara, P. & Medel, J.J. (2005). Filtrado Digital en Tiempo Real: Análisis Computacional para Estimación de Parámetros en Sistemas Estocásticos Lineales estacionarios. *Computación y Sistemas*, Vol. 8, No. 3, pp. 252–264.
4. Medel, J.J., Guevara, P., & Flores, R.A. (2004). Caracterización de Filtros Digitales en Tiempo Real para Computadoras Digitales. *Computación y Sistemas*, Vol. 7, No. 3, pp. 190–209.
5. Medel J.J., Guevara P., & Cruz, P.D. (2007). *Temas selectos de Sistemas en Tiempo Real*. Primera edición, Editorial Politécnico, México D.F., pp. 30–60.
6. Márquez, F. (2004). *UNIX Programación avanzada*. 3ª edición, Editorial Ra-Ma, España, Madrid, pp. 42–52.
7. Buttazzo, G. (2011). *Hard real-time computing systems: Predictable scheduling, algorithms and applications*. 3ª edición, Springer Science+Business Media, Vol. 24, New York, USA.
8. Liu, C. & Layland, J. (1973). Scheduling algorithms for multiprogramming in hard-real-time environment. *Journal of the ACM*, Vol. 20, No. 1, pp. 46–61.
9. de la Puente, J.A. (2007). *Tema 5: Planificación de tareas en tiempo real*. Departamento de Arquitectura y Tecnología de Sistemas Informáticos, Universidad Politécnica de Madrid.
10. Mejía, P. (2008). *Curso de Sistemas En Tiempo Real: Métodos de planificación. (Sistemas Operativos)*. CINVESTAV-IPN, Sección de Computación. pp. 8–15.
11. Crespo, A. & Alonso, A. (2006). Una panorámica de los Sistemas de Tiempo Real. *Revista Iberoamericana de Automática e Informática Industrial*, Vol. 3, No. 2. pp. 7–18.
12. González, J.L. (2010). *Curso sistemas informáticos en tiempo real: Capítulo 4, 5 y 6*. Universidad de Valladolid, pp. 13–20.
13. Moncusi, M.A. (2005). *Ahorro energético en la planificación de sistemas en tiempo real*. Tesis de Doctorado en Arquitectura y Tecnología de

- Computadoras, Universidad Politécnica de Cataluña, España, Cataluña, pp. 26–30.
14. **Pastor, F. (2005).** *Curso de Sistemas Informáticos en Tiempo Real: Planificación en tiempo real.* Universidad de Valencia, Departamento de Electrónica e Informática, pp. 13–16.
15. **Vedia, J. (2009).** *Diseño y desarrollo de aplicaciones distribuidas en tiempo real basadas en servicios mediante modelado UML.* Tesis de Ingeniería Técnica de Telecomunicación especialidad Telemática, Universidad Carlos III de Madrid, España, Madrid, pp. 25–28.
16. **Yepez, J. (2007).** *Diseño e implementación de una herramienta para la planificación de ejecutivos cíclicos.* Tesis de Ingeniería en Sistemas, Automática e Informática industrial, Universidad Politécnica de Cataluña, España, Cataluña, pp. 10–11.
17. **Medel, J.J., Guevara, P., & Cruz, D. (2007).** *Temas Selectos de Sistemas en Tiempo real.* Primera edición, Editorial Politécnico, México D.F., pp. 50–54.
18. **Guevara, P., Medel, J.J., & Delgado, G. (2011).** Description of execution time dynamics for a set of concurrent real-time tasks. *Revista Facultad de Ingeniería: Universidad de Antioquia*, Vol. 61, pp. 123–31.
19. **Guevara, P., Morales, O., & Falcón, J. (2012).** Concurrent real-time schedulers, a classification based on functions. *Studies in Informatics and Control Journal*, Vol. 21, No. 1, pp. 27–32.
20. **Baruah, S. & Goossens, J. (2004).** *Handbook of Scheduling: Algorithms, Models, and Performance Analysis.* Chapman Hall/CRC Press, Bruselas, Bélgica, pp. 2–4.
21. **Buttazzo, G. (2005).** Rate Monotonic vs. EDF: Judgment Day. *Real-Time Systems*, Vol. 29, No. 1, pp. 5–26.
22. **Bellman, R. (1957).** The theory of dynamic programming. *Bull. Amer. Math. Soc.*, Vol. 60, pp. 503–515.
23. **Schedule Stability (2014).** *Schedule Stability, Encyclopedia of Production and Manufacturing Management.* Springer.
24. **Woodside, C. (1986).** Controllability of Computer Performance Tradeoffs Obtained Using Controlled-Share Queue Schedulers. *IEEE Transactions on Software Engineering*, Vol. 12, No. 10, pp. 1041–1048.
25. **Kai, L., Yong, W., & Meilin, L. (2014).** A Non-Cooperative Game Model for Reliability-Based Task Scheduling in Cloud Computing. *Proc. of CoRR 2014*, China.
26. **Campoy, M., Sáez, S., Perles, A., & Busquets, J.V. (2004).** Schedulability Analysis in the EDF Scheduler with Cache Memories. *Lecture Notes in Computer Science*. Vol. 2968, pp. 328–341.

Pedro Guevara López obtuvo su grado de Doctor y Maestro en Ciencias de la Computación e Ingeniero Electricista, todos del Instituto Politécnico Nacional. Es Profesor Investigador de la Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Culhuacán del Instituto Politécnico Nacional. Sus áreas de investigación son sistemas en tiempo real y modelado de sistemas dinámicos

Jorge Salvador Valdéz Martínez es ingeniero en Comunicaciones y Electrónica egresado de la Escuela Superior de Ingeniería Mecánica y Eléctrica, Maestro en Tecnología Avanzada por el Centro de Investigación en Ciencia Aplicada y Tecnología Avanzada y Candidato a Doctor en Ciencias en Comunicaciones y Electrónica por la Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad “Culhuacán”, todos del Instituto Politécnico Nacional. Sus áreas de interés son sistemas en tiempo real y sistemas de control.

Gustavo Delgado Reyes es ingeniero en Comunicaciones y Electrónica y Maestro en Ciencias de Ingeniería en Microelectrónica egresado de la Escuela Superior de Ingeniería Mecánica y Eléctrica del IPN. Actualmente es Candidato a Doctor en Comunicaciones y Electrónica en el Instituto Politécnico Nacional, Sus áreas de interés son sistemas en tiempo real, sistemas embebidos y teoría de control.

Article received on 20/08/2013, accepted on 14/10/2014.