Revista Iberoamericana de Automática e Informática Industrial
CEA IFAC comité español de automática
RIAI
ISSN: 1697-7912. Vol. 3, Núm. 2, Abril 2006, pp. 50-60
http://riai.isa.upv.es

ASIGNACIÓN DINÁMICA DE RECURSOS EN SISTEMAS DE CONTROL DE TIEMPO REAL [1]

Pau Martí * **Caixue Lin** * **Scott A. Brandt** *
Manel Velasco ** **Jordi Ayza** ** **Josep M. Fuertes** **

* *Computer Science Department*
University of California, Santa Cruz (UCSC)
1156 High Street, Santa Cruz, CA95064, USA
{*pmarti,lcx,scott*}*@cs.ucsc.edu*
** *Departament d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial (ESAII)*
Universitat Politècnica de Catalunya
Pau Gargallo, 5, 08028 Barcelona, Spain
{*manel.velasco,jordi.ayza,josep.m.fuertes*}*@upc.edu*

Resumen: Las técnicas tradicionales de tiempo real realizan la asignación de recursos a las tareas de control basándose en estrategias en "lazo abierto" que asignan (a priori) una porción de procesador a cada tarea. Esta asignación es estática desde un punto de vista de recursos, es decir, a lo largo de toda la vida de la aplicación se mantiene constante independientemente de la dinámica que cada lazo de control tenga en tiempo de ejecución. El uso de asignaciones estáticas proporciona un rendimiento de control medio, con mínimo coste computacional, pero no consiguen proporcionar el mejor rendimiento que puede obtenerse con los recursos de cómputo disponibles. En este trabajo mostramos cómo, asignando de forma dinámica los recursos a las tareas de control, el rendimiento proporcionado por el conjunto de la aplicación mejora significativamente. Además presentamos la política de asignación de recursos óptima respecto al rendimiento proporcionado por las tareas de control. Los resultados experimentales realizados sobre una plataforma de tiempo real muestran los beneficios de la políticas propuestas y avalan los resultados teóricos presentados.

Palabras Clave: Sistemas de tiempo real, sistemas controlados por computador, asignación de recursos, optimización

1. INTRODUCCIÓN

En los sistemas de control de tiempo real, la asignación de recursos a tareas de control determina el rendimiento de los lazos de control, a mayor asignación de recursos mayor frecuencia de ejecución del controlador y mejor rendimiento. Las técnicas tradicionales de tiempo real asignan los recursos a las tareas de control basándose en estrategias en "lazo abierto". Estas estrategias asignan estáticamente una porción de procesador a cada tarea de control con independencia de la dinámica que cada lazo de control tenga en

[1] Este trabajo ha sido financiado parcialmente por Intel Corporation, por Departament d'Universitats, Recerca i Societat de la Informació de la Generalitat de Catalunya, y por el proyecto CICYT ref. DPI2002-01621.

tiempo de ejecución (ver por ejemplo (Liu and Layland, 1973) o (Seto *et al.*, 1996)).

Las políticas de asignación de recursos en lazo abierto garantizan una proporción constante de procesador a cada controlador (que implica una frecuencia de ejecución fija). Ese hecho permite cumplir con las especificaciones tradicionales de rendimiento de control, pero si prestamos atención a la relación existente entre rendimiento de control y la frecuencia de ejecución del controlador observamos que estas técnicas parecen no ser óptimas. Un controlador puede no necesitar toda la frecuencia de ejecución asignada si el sistema está controlado y en equilibrio. Por contra, estando el sistema controlado, si sufre una perturbación (y su estado es desplazado lejos del punto de equilibrio), un incremento en la frecuencia de ejecución de la tarea de control puede acelerar su recuperación (Marti *et al.*, 2002).

Teniendo en cuenta esta observación, en este trabajo mostramos que, asignando dinámicamente recursos a las tareas de control según el *estado* de los sistemas controlados el rendimiento de control proporcionado por el conjunto de tareas de control se puede mejorar significativamente. A esta política de asignación la podemos considerar de lazo cerrado. Concretamente, presentamos una política de asignación de recursos óptima para tareas de control.

La correcta implementación de la política óptima implica considerar dos aspectos básicos. Por una parte debemos tener en cuenta que las tareas de control deben ser capaces de ejecutarse a distintas frecuencias (adaptando adecuadamente las ganancias de las leyes de control que implementan) dadas distintas asignaciones de recursos (como se explica en (Marti *et al.*, 2001*a*) y (Marti *et al.*, 2002))). Y, en segundo lugar, la implementación requiere el soporte de un sistema de tiempo real flexible, capaz de cambiar dinámicamente los recursos asignados (modificando los periodos de las tareas, por ejemplo), garantizando, así mismo, las restricciones temporales de las tareas. El sistema de tiempo real RBED (Rate-Based Earliest Deadline) (Brandt *et al.*, 2003) proporciona estas características, y facilita la implementación de las políticas de asignación de recursos. RBED está basado en la separación de la asignación de recursos del momento en que se despachan las tareas (modelo RAD - Resource Allocation/Dispatching, (Brandt and Nutt, 2002)), y permite asignar recursos de forma precisa según las necesidades de cada tarea de control.

Con el fin de poder efectuar una comparativa entre los diferentes resultados obtenidos utilizaremos tres políticas de asignación de recursos. En primer lugar generaremos una aplicación basada en las técnicas tradicionales (política que denominaremos *estática*), en la que cada tarea de control dispondrá de un porcentaje de recursos estático a lo largo del tiempo de ejecución del sistema. En segundo lugar implementaremos una política de asignación de recursos basada en el error total de cada sistema, de forma que a mayor error mayor proporción de recursos asignados (política que denominaremos *proporcional*). Y finalmente utilizaremos la política de asignación dinámica que se describe en este artículo.

Para ilustrar los beneficios de las técnicas de asignación de recursos en lazo cerrado presentamos unos resultados experimentales que muestran que, dada una misma secuencia de perturbaciones generadas de forma aleatoria sobre un conjunto de lazos de control, la política óptima mejora significativamente el rendimiento de control respecto a la implementación tradicional y la implementación proporcional.

2. ESTADO DEL ARTE

La mejora del rendimiento de control en plataformas con recursos limitados ha sido tratado en (Seto *et al.*, 1996), (Rehbinder and Sanfridson, 2000) o (Palopoli *et al.*, 2002). En estos trabajos la asignación de recursos es estática, y no se contempla la asignación de recursos de forma dinámica en tiempo de ejecución.

La asignación y/o gestión de recursos en tiempo de ejecución sistemas de control se investiga en (Ramanathan, 1999), (Shin and Meissner, 1999), (Beccari *et al.*, 1999), (Caccamo *et al.*, 2000), (Cervin and Eker, 2000), (Marti *et al.*, 2001*b*), (Cervin *et al.*, 2002), (Buttazzo *et al.*, 2004) and (Balbastre *et al.*, 2004). Aunque la gestión de recursos sea dinámica, en estos trabajos no se investiga como reasignar recursos en función del estado de los sistemas controlados, en lazo cerrado.

Las técnicas que presentamos son similares a las arquitecturas en lazo cerrado presentadas en (Zhao and Zheng, 1999) y (Henriksson *et al.*, 2002). El objetivo del trabajo (Zhao and Zheng, 1999) es en el diseño de controladores para satisfacer criterios de estabilidad exponencial y asintótica con periodos constantes. Cabe destacar que nuestro objetivo no es el diseño de controladores, sino el estudio de cómo la adaptabilidad de los controladores a distintos periodos puede ser usado para mejorar el rendimiento de control en un sistema de control de tiempo real multitarea. La arquitectura presentada en (Henriksson *et al.*, 2002) se centra en controladores predictivos con modelo, donde la adaptación se consigue variando los tiempos de ejecución. Nuestra arquitectura apunta a una clase más amplia de lazos de control

con controladores diseñados con técnicas clásicas de control para sistemas lineales, y la variación de la frecuencia de ejecución de los controladores (variación que se consigue asignando en tiempo de ejecución distintas porciones de CPU a cada tarea de control).

Haceos hincapié en el hecho que la política de asignación de recursos óptima que presentamos soluciona el problema de planificación con calidad de control (QoC, Quality-of-Control) formulado en (Marti *et al.*, 2002), pero desde un punto de vista de asignación de recursos. Algunos resultados preliminares sobre asignación de recursos en lazo cerrado, que brevemente se incluyen en este trabajo, fueron presentadas en (Lin *et al.*, 2004).

3. DESCRIPCION DEL PROBLEMA

3.1 Arquitectura del sistema

El sistema de control de tiempo real que consideramos está formado por un conjunto de n tareas de control $\tau_1, \tau_2, \ldots, \tau_n$, cada una de ellas encargada de controlar una planta (entendemos por planta un sistema dinámico, además, los sistemas son independientes entre ellos).

Cada sistema puede modelarse con las ecuaciones en espacio de estados (1) y (2) que describen su dinámica lineal e invariante en el tiempo [2].

$$\dot{\boldsymbol{x}}_i(t) = A_i \boldsymbol{x}_i(t) + B_i \boldsymbol{u}_i(t) \quad t \in \mathbb{R}^+ \qquad (1)$$

$$\boldsymbol{y}_i(t) = C_i \boldsymbol{x}_i(t) + D_i \boldsymbol{u}_i(t) \qquad (2)$$

La Figura 1 ilustra la arquitectura que consideramos, donde el problema a resolver es como asignar recursos r_i (*factor parcial de utilización*) a las tareas de control teniendo en cuenta el estado de las plantas controladas.

3.2 Formalización del problema

Con el fin de dar forma a la política que pretendemos desarrollar utilizaremos la norma del vector de estado de cada uno de los sistemas (ecuaciones (1) y (2)) como distancia al punto de equilibrio. En todos los casos suponemos que deseamos conducir al sistema al estado $\overrightarrow{0}$, en caso contrario, un simple cambio de coordenadas sobre (1) y (2) nos llevará a la formulación en la que el $\overrightarrow{0}$ es el estado de equilibrio.

Observación 1. Esta medida (el *error*, e_i en (3)), que indica como es de *crítica* la situación de

[2] Los sistemas no lineales también pueden ser incluidos en la arquitectura si es posible linearizarlos entorno al punto de equilibrio.

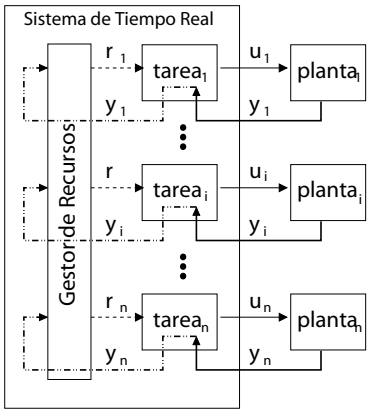

Figura 1. Arquitectura en lazo cerrado del sistema de tiempo real con multiples lazos de control

cada sistema controlado, es la información (de lazo cerrado) que cada tarea envía al gestor de recursos para que este re-asigne recursos en tiempo de ejecución.

La suposición de norma del vector de estado implica el conocimiento del valor de todos los estados del mismo, en caso de no disponer de ellos es posible extraer la información necesaria de las medidas disponibles utilizando el modelo del sistema (Åström and Wittenmark, 1997).

$$e_i = |x_i(t)| \qquad (3)$$

Cada tarea de control tiene un índice de rendimiento de control p_i que relaciona el rendimiento según la proporción de procesador (r_i en (4)) asignado para su ejecución.

$$p_i(r_i) = \alpha_i \frac{c_i}{h_i} + \beta_i = \alpha_i r_i + \beta_i \qquad (4)$$

Como se indica en (Cervin *et al.*, 2002), esta relación, usando índices estándares lineales o cuadráticos, se puede aproximar con una relación lineal, como se detalla en (4), donde c_i y h_i son, respectivamente, el peor tiempo de ejecución de la tarea i y su periodo. Los coeficientes α_i y β_i se calculan a priori e indican, para cada índice de rendimiento p_i, cual va a ser proporción entre el rendimiento y los recursos de cómputo r_i asignados a la tarea de control.

En la figura 2 se representa la curva real y la aproximación empleada para generar el índice de calidad de control. En las ordenadas aparece el valor del índice de rendimiento, $p_i(r_i)$, y en las abcisas el porcentaje de recursos entregados a la tarea de control.

En (4), teniendo en cuenta que el peor tiempo de ejecución de la tarea es constante, cualquier

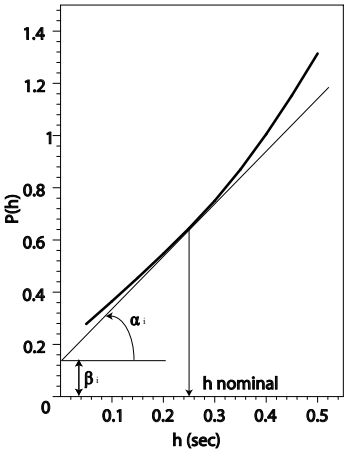

Figura 2. Ejemplo de relación entre la calidad de control de un sistema y los recursos asignados

variación en la asignación de recursos r_i implica variar el periodo h_i correspondiente. La aproximación lineal dada en (4) es una simplificación con el fin de facilitar el cálculo. La política óptima de asignación que se describe en la sección 4 también admite índices de rendimiento en forma polinomial de grado menor o igual que cinco, aunque para relaciones mas complejas es necesario implementar soluciones numéricas y resolverlas en tiempo de ejecución, lo que alteraría de forma drástica el tiempo de cómputo e invalidaría los resultados obtenidos.

Observación 2. Dados los índices de rendimiento p_i, es fácil observar la norma básica de comportamiento. A más porcentaje de procesador,r_i, mejor es el rendimiento obtenido. Aunque es destacable que este hecho sólo se produce para aproximaciones lineales de la relación real realizadas en las zonas ascendentes de las curvas. Aun así el método que presentamos puede aplicarse a toda clase de aproximaciones.

Los algoritmos empleados para controlar los sistemas implementan leyes de control adaptables a distintos valores del periodo de muestreo (que coinciden con los periodos de las tareas), siguiendo las técnicas presentadas en (Wittenmark and Åström, 1980) y (Albertos and Salt, 1990). Para la clase de sistemas que se pueden describir con las ecuaciones (1) y (2), las señales de control $\boldsymbol{u}_i(t)$ vienen dadas por leyes de control discretas $L_i(h_i)$ que se diseñan con métodos estándares [3]. A cada ley de control, que depende paramétricamente del periodo de muestreo $L_i(h_i)$, se le especifica un rango de periodos de muestreo

[3] Con técnicas de control discreto o con técnicas de control continuo con posterior discretización.

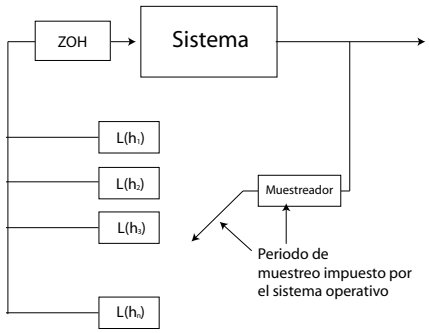

Figura 3. Modelo de control empleado

$h_i \in [h_i^{max} \dots h_i^{min}]$ de tal manera que cada uno de ellos cumple las especificaciones de diseño.

En tiempo de ejecución, cada tarea de control se ejecuta con un valor de periodo perteneciente al rango especificado, adaptando las ganancias, $L(h_i)$, apropiadamente (para mas detalles, ver (Marti *et al.*, 2001*a*)). Consiguientemente, la matriz del modelo de cada sistema de control en lazo cerrado tomará distintos valores, dependiendo de los valores que adopte el periodo de cada tarea de control. Cada una de estas matrices es estable de forma aislada, aunque para garantizar la estabilidad del conjunto es necesario recurrir a las técnicas descritas en (Marti *et al.*, 2001*c*), que se basan en los resultados sobre estabilidad de un conjunto de matrices, presentados en (Dogruel and Özgüner, 1995).

La estructura final del controlador es la que se presenta en la figura 3, en la que se observa un sistema de control conmutado, en el que la conmutación está ligada al periodo asignado en cada momento.

A nivel de sistema operativo, cada tarea de control τ_i se caracteriza por el factor parcial de utilización r_i (cuya variabilidad viene determinada por el peor tiempo de ejecución c_i y el rango de periodos $h_i \in [h_i^{max} \dots h_i^{min}]$), su índice de rendimiento p_i (que es una función que se calcula fuera a priori), y el error del sistema controlado e_i (que va variando en tiempo de ejecución según las perturbaciones que afectan al sistema controlado), tal y como se representa en (5).

$$\tau_i = \{r_i, p_i, e_i\} \tag{5}$$

3.3 Enunciado

Dado un sistema de tiempo real multitarea, para un conjunto de n tareas de control especificadas mediante (5), el problema es determinar (en tiempo de ejecución) los factores parciales de utilización r_i, $i = 1, \dots, n$ que deben asignarse a cada tarea de control, de tal manera que todas

las tareas sean planificables y el rendimiento de control instantáneo proporcionado por el conjunto de tares sea máximo en todo momento.

4. ASIGNACIÓN DE RECURSOS OPTIMA

4.1 Formulación genérica

El problema de asignación de recursos se puede formular como un problema genérico de optimización con restricciones, como se muestra a continuación [4],

$$\text{maximizar} \quad g(p_i(r_i), e_i) \tag{6}$$

$$\text{sujeto a} \quad \sum_{i=1}^{n} r_i \leq U_d \tag{7}$$

donde la solución es un vector $\boldsymbol{r} = [r_1, r_2, \ldots, r_n]$ que maximiza el rendimiento de control global proporcionado por el conjunto de tareas de control en un instante determinado, rendimiento representado por la función objetivo g en (6), teniendo en cuenta las restricciones de planificabilidad dadas por el factor de utilización deseado por el conjunto de tareas, U_d en (7). El máximo absoluto $\boldsymbol{r}$ se encontrará en el interior, en la frontera, o en los puntos extremos del conjunto de puntos definido por (7). Un algoritmo genérico para encontrar la solución se puede resumir en cuatro pasos (Chong and Zak, 1996):

Paso 1: Buscar máximos relativos en el interior del conjunto de puntos definido por (7), solucionando el sistema de ecuaciones especificado por (8),

$$\frac{\partial g}{\partial r_1} = 0, \ \frac{\partial g}{\partial r_2} = 0, \ \ldots, \ \frac{\partial g}{\partial r_n} = 0 \tag{8}$$

y guardar aquellos $\boldsymbol{r}$ que maximizan g.

Paso 2: Buscar máximos relativos en la frontera del conjunto de puntos definido por (7), solucionando el sistema de ecuaciones especificado por (9),

$$\begin{aligned}
&\frac{\partial g([U_d - r_2 - r_3 - \ldots - r_n, r_2, r_3, \ldots, r_n])}{\partial r_i} = 0, \ i \neq 1 \\
&\frac{\partial g([r_1, U_d - r_1 - r_3 - \ldots - r_n, r_3, \ldots, r_n])}{\partial r_i} = 0, \ i \neq 2 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
&\frac{\partial g([r_1, r_2, \ldots, U_d - r_1 - r_2 - \ldots - r_{n-1}])}{\partial r_i} = 0, \ i \neq n
\end{aligned} \tag{9}$$

y guardar aquellos $\boldsymbol{r}$ que maximizan g.

Paso 3: Buscar los valores de g para los extremos del conjunto de puntos definido por (7), como se especifica en (10),

$$\begin{aligned}
&g([U_d, 0, \ldots, 0]) \\
&g([0, U_d, \ldots, 0]) \\
&\qquad\vdots \\
&g([0, 0, \ldots, U_d])
\end{aligned} \tag{10}$$

y guardar aquellos $\boldsymbol{r}$ que maximizan g.

Paso 4: Escoger el $\boldsymbol{r}$ entre los obtenidos en *Paso 1, 2* y *3* que maximiza g.

Según como sea la función objetivo g en (6), solucionar el problema de optimización en tiempo de ejecución puede no ser viable debido al coste computacional. Pero como explicamos a continuación, para la arquitectura que el problema se puede simplificar de tal manera que 1) tiene solución y 2) es viable encontrarla en tiempo de ejecución.

4.2 Simplificación

La simplificación del problema de optimización está basada en las siguientes obsevaciones:

Observación 3. Cada tarea de control se considera independiente, en el sentido de estar encargada de controlar un sistema físico (sección 3).

Considerando la independencia entre sistemas, la función objetivo $g(\cdot)$ en (6) que enlaza los rendimientos p_i de cada tarea de control se define como la suma (ponderada) de los rendimientos proporcionados por cada tarea de control (también se definió en los procedimientos de optimización presentados en (Seto *et al.*, 1996) o (Cervin *et al.*, 2002) de esta forma). Cada índice de rendimiento puede ser ponderado con un peso w_i, proporcionando así un mecanismo que permita comparar los distintos lazos de control del sistema. Esta comparación es importante, los sistemas son intrínsecamente diferentes, y es necesario definir constantes que normalicen los índices de forma que podamos establecer comparaciones coherentes entre ellos.

Observación 4. La solución $\boldsymbol{r}$ debe tener en cuenta que, para cualquier lazo de control, a mayor error e_i y dada una asignación r_i, mayor debe ser el rendimiento proporcionado por la tarea de control.

Esta lógica se consigue escalando el rendimiento p_i con el error e_i del sistema controlado, $e_i p_i$. En resumen, el problema de optimización se puede re-escribir como:

[4] Cabe destacar que las restricciones del problema de optimización se tendrían que completar con la restricción $\forall i, \ r_i \geq 0$. Sin embargo esta omisión no altera el resultado final y simplifica de forma significativa la explicación del algoritmo solución del problema.

$$\text{maximizar} \quad \sum_{i=1}^{n} w_i e_i p_i(r_i) \tag{11}$$

$$\text{sujeto a} \quad \sum_{i=1}^{n} r_i \leq U_d \tag{12}$$

La complejidad de la solución del problema planteado en (11) y (12) depende de cada función p_i por el hecho que las ecuaciones (8) y (9) se han simplificado respectivamente al conjunto de ecuaciones especificados en (13)

$$\frac{\partial b_1}{\partial r_1} = 0, \; \frac{\partial b_2}{\partial r_2} = 0, \; \ldots, \; \frac{\partial b_n}{\partial r_n} = 0 \tag{13}$$

y (14), donde $b_i = w_i e_i p_i(r_i)$.

$$\begin{aligned} \frac{\partial b_1(U_d - r_2 - r_3 - \ldots - r_n)}{\partial r_i} &= 0, \; i \neq 1 \\ \frac{\partial b_2(U_d - r_1 - r_3 - \ldots - r_n)}{\partial r_i} &= 0, \; i \neq 2 \\ &\vdots \\ \frac{\partial b_n(U_d - r_1 - r_2 - \ldots - r_{n-1})}{\partial r_i} &= 0, \; i \neq n \end{aligned} \tag{14}$$

Si los índices de rendimiento p_i son lineales (como se definió definió en (4)), el problema de optimización es solución $\boldsymbol{r} = [r_1, r_2, \ldots, r_n]$ se encuentra haciendo una simple búsqueda (correspondiente al *Paso 3*) ya que las ecuaciones (13) y (14) correspondientes a los *Pasos 1* y *2* quedan indeterminadas. Por ejemplo, si p_i son lineales ($p_i = \alpha_i r_i$) en (13), obtenemos el siguiente conjunto de ecuaciones (15)

$$\begin{aligned} \frac{\partial b_1}{\partial r_1} &= \frac{\partial w_1 e_1 \alpha_1 r_1}{\partial r_1} = w_1 e_1 \alpha_1 = 0 \\ \frac{\partial b_2}{\partial r_2} &= \frac{\partial w_2 e_2 \alpha_2 r_2}{\partial r_2} = w_2 e_2 \alpha_2 = 0 \\ &\vdots \\ \frac{\partial b_n}{\partial r_n} &= \frac{\partial w_n e_n \alpha_n r_n}{\partial r_n} = w_n e_n \alpha_n = 0 \end{aligned} \tag{15}$$

Y lo mismo ocurre con el conjunto de ecuaciones especificado en (14).

En consecuencia, la solución se reduce en realizar la búsqueda del *Paso 3*, ajustada al problema planteado en (11) y (12). Así, evaluando

$$\begin{aligned} g([U_d, 0, \ldots, 0]) &= b_1(U_d) + b_2(0) + \ldots + b_n(0) \\ &= w_1 e_1 \alpha_1 U_d \\ g([0, U_d, \ldots, 0]) &= b_1(o) + b_2(U_d) + \ldots + b_n(0) \\ &= w_2 e_2 \alpha_2 U_d \\ &\vdots \\ g([0, 0, \ldots, U_d]) &= b_1(0) + b_2(0) + \ldots + b_n(U_d) \\ &= w_n e_n \alpha_n U_d \end{aligned} \tag{16}$$

encontraremos la asignación óptima de recursos. Cabe destacar que (16) equivale a buscar el máximo $w_i e_i \alpha_i$, $\forall i = 1 \ldots n$.

Teorema 1. La solución óptima $\boldsymbol{r} = [r_1, r_2, \ldots, r_n]$ del problema de optimización especificado en (11) y (12) es $\boldsymbol{r} = [0, 0, \ldots, 0, r_i = U_d, 0, \ldots, 0]$, $i \in [1, \ldots, n]$ tal que $w_i e_i \alpha_i$ es máximo $\forall i \in [1, \ldots, n]$, si el conjunto de tareas de control se especifican como en (5).

Demostración.
Se deduce de la explicación previa al teorema. □

Observación 5. En términos de asignación de recursos, el teorema concluye que todo el procesador disponible (U_d) debe de asignarse a la tarea de control que tenga mayor $w_i e_i p_i$. En el caso particular en el que todos los índices p_i son iguales y todos los w_i son iguales (plantas idénticas), todo el procesador disponible debe ser asignado a la tarea de control cuyo sistema controlado tenga el mayor error e_i. En la practica, a cada tarea de control τ_i, se le asigna un factor de utilización mínimo dado por su h_i^{max} para garantizar una frecuencia mínima de ejecución, sin embargo, U_d no contempla los valores asignados de forma fija a las tareas. Consecuentemente, este resultado indica que a la tarea de control con mayor error se le debe asignar todo el procesador disponible, dado que el resto de tareas ya tienen asignado un mínimo para el control de las plantas.

Observación 6. Si los índices p_i no son lineales pero son funciones polinomiales en r_i de grado menor que cinco (Gellert *et al.*, 1988), se puede encontrar una solución analítica siguiendo los *Pasos 1*, *2*, *3* y *4* (solucionando las ecuaciones (13), (14) y (10)), lo cual implica que en tiempo de ejecución, la solución es computable sin causar un *overhead excesivo.*

Observación 7. Para el caso de índices p_i lineales, la explicación geométrica de la solución óptima al problema formulado en (11) y (12) es como sigue. La solución óptima $\boldsymbol{r}$ es uno de los puntos extremos (vértices) en la proyección de la hiperecta definida por las restricciones (11) sobre el hiperplano definido por la función objetivo (12). En la figura 4 se observa el caso particular de dos tareas de control donde $U_d = 0.8$, y las tareas tienen el mismo índice de rendimiento y pesos ($\alpha_1 = \alpha_2 = 1$, implicando que $p_1(r_1) = r_1$ y $p_2(r_2) = r_2$, y $w_1 = w_2 = 1$), pero una planta en el instante t tiene un error mas grande ($e_1 = |x_1(t)| = 4$) que la otra ($e_2 = |x_2(t)| = 1$). Como se puede ver en la figura, el máximo de la función objetivo g ($g = \sum_{i=1}^{n} w_i e_i p_i(r_i)$, ver (6) y (11)) considerando las restricciones de planificabilidad

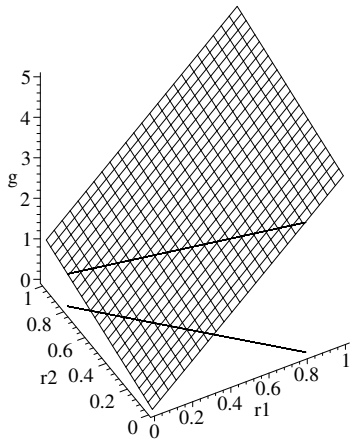

Figura 4. Solución óptima para dos tareas de control

se encuentra en $\boldsymbol{r} = [r_1, r_2] = [0.8, 0.0]$ (punto extremo).

5. IMPLEMENTACIÓN DE LAS POLÍTICAS DE ASIGNACIÓN DE RECURSOS

Hemos implementado y evaluado la arquitectura en lazo cerrado para descrita en la sección 3 en el sistema de tiempo real integrado RBED.

RBED asigna recursos a tareas (en porcentajes de CPU) de tal de tal manera que la asignación total es igual o menor al 100%, y posteriormente planifica todas la tareas con el algoritmo Earliest Deadline First (EDF) (Liu and Layland, 1973).

EDF garantiza que todas las instancias de las tareas de control cumplen sus plazos si $\sum_{i=1}^{n} r_i \leq 1$ (donde los plazos se consideran igual a los periodos). En cualquier instante de tiempo t, debido a una re-asignación dinámica de recursos, los factores parciales de utilización de las tareas (y por consiguiente sus periodos) se actualizan $r'_i = \frac{c_i}{h'_i}$ de tal manera que $\sum_{i=1}^{n} r'_i \leq 1$. Cabe destacar que cualquier cambio en el periodo de las tareas está sujeto a las restricciones descritas en (Brandt *et al.*, 2003). En consecuencia, RBED garantiza todos los nuevos plazos de las tareas cuando se da la asignación de nuevos periodos.

Aparte de implementar en RBED la política óptima (obtenida en la Sección 4), y para demostrar sus beneficios y evaluar su rendimiento, también hemos implementado una política de asignación de recursos en lazo cerrado, llamada *proporcional*, que asigna recursos a las tareas en proporción al error. Además, para poder comparar directamente el rendimiento con aquel que se obtiene con implementaciones tradicionales, también hemos implementado una política de base, llamada *estática*, en la cual las tareas de control siempre comparten los recursos disponibles de forma fija (sin usar asignación dinámica de recursos). La política estática implementa leyes de control *tradicionales* (no adaptativas).

Política óptima: La política óptima se puede resumir en:

$$r_i = \begin{cases} U_d - \sum_{1 \leq j \leq n, j \neq i} r_j^{min}, & \textit{if } w_i e_i p_i(r_i) \text{ is max} \\ r_i^{min}, & \textit{otherwise} \end{cases}$$

donde $\frac{c_i}{r_i} \in [h_i^{min} \ldots h_i^{max}]$, n es el número de tareas de control, y r_j^{min} (que corresponde a h_j^{max}) es el factor parcial de utilización garantizado a cada tarea de control j (cuyo $w_j e_j p_j$ no es máximo).

Política proporcional: La política proporcional, que asigna recursos a cada tarea de control en proporción a $w_i e_i p_i$, se puede resumir en:

$$r_i = \frac{w_i e_i p_i(r_i)}{\sum_{1 \leq j \leq n} w_j e_j p_j(r_j)} \cdot U_d \ , \quad \frac{c_i}{r_i} \in [h_{min} \ldots h_{max}]$$

Esta política es la que podemos denominar *justa*, en el sentido que a cualquier lazo de control se le asigna un porcentaje de procesador proporcional al error que está cometiendo en relación a los demás lazos de control. Si el factor parcial de utilización asignado a cualquier tarea τ_i resulta en un periodo más largo que h_i^{max}, entonces el controlador se ejecutara con h_i^{max}. Este mecanismo garantiza una frecuencia mínima de ejecución a cada tarea.

Política estática: La política estática, que asigna recursos a cada tarea de control de forma fija, se puede resumir en:

$$r_i = \frac{w_i}{\sum_{1 \leq j \leq n} w_j} \cdot U_d$$

La implementación de las políticas dinámicas requiere 1) que las tareas de control puedan enviar al gestor de recursos del sistema operativo, en cada ejecución, el error de su sistema controlado y 2) que conozcan el periodo que se les va aplicar para poder ajustar las ganancias de forma adecuada. Esto se consigue con una llamada al sistema con el error como parámetro de entrada, y el periodo (calculado por la política dinámica) como parámetro de salida.

6. RESULTADOS EXPERIMENTALES

La políticas de asignación de recueros para tareas de control descritas en la sección 5 se han integrado en RBED y se han implementado, por razones de sencillez, en Linux (kernel 2.4.20). Se ha ejecutado el sistema durante largos periodos de tiempo y se ha realizado un gran número de experimentos con cargas generadas aleatoriamente,

de tal manera que los experimentos se pueden considerar generales, y no limitados a escenarios especiales. Todos los experimentos se realizaron en un PC estándar (Pentium III, 1Ghz, 512MB RAM y disco duro de 40GB).

En los experimentos se ejecuta un conjunto de tres tareas de control, cada una de ellas controlando un proceso *Ball and Beam* simulado. El modelo lineal e invariante en el tiempo, en espacio de estados, usado en nuestros experimentos es

$$\frac{dx(t)}{dt} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \boldsymbol{x}(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

Cada tarea de control implementa la misma ley de control paramétrica, obtenida usando técnicas estándares de asignación de polos. Se definen todos los sistemas idénticos con el fin de facilitar el análisis de rendimiento. Es interesante destacar que en este caso, w_i y p_i son iguales para todas las tareas, lo que significa que el error e_i sera el factor principal a tener en cuenta en cada política [5] .

El factor de utilización deseado para las tres tareas de control es $U_d = 97\%$, dejando un 3% de CPU a tareas de propósito general. El peor tiempo de ejecución de cada tarea es $0.135s$. Para las políticas dinámicas (óptima y tarea puede ejecutarse con un periodo dentro del rango $[0.3s \ldots 0.5s]$. Para la política estática, cada tarea tiene un periodo fijo proporcional a un tercio de U_d, esto es, un period de 0.42s.

Para cada política, ejecutamos las tres tareas durante una hora y generamos perturbaciones aleatorias (para cada *Ball and Beam*) con diferentes intervalos. De esta manera, la distancia entre perturbaciones varia de tal manera que un sistema puede ser permanentemente perturbado o casi nunca perturbado (capturando así todos los posibles escenarios).

6.1 Análisis de la asignación de recursos

Figura 5 muestra la variación en los periodos de las tres tareas de control, según las dos políticas dinámicas cuando las perturbaciones afectan a los *ball and beam*. Sólo se muestran los 20 primeros segundos de una simulación de una hora, para fijar la atención en como se asignan los recursos (como

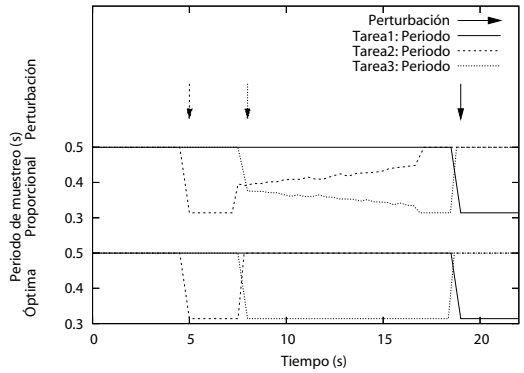

Figura 5. Variación de los periodos de las tareas según las políticas de asignación óptima y proporcional

varían los periodos). El eje horizontal representa el tiempo y el eje vertical representa la variación de los periodos entre 0.3s y 0.5s para la política proporcional (arriba) y la política óptima (abajo), y las flechas simbolizan el momento en que los sistemas son perturbados.

En la política proporcional, cualquier cambio en el error de cada *ball and beam* produce un cambio en el periodo de la tarea de control, lo que deriva en una adaptación constante (ya que el error cambia constantemente después de la llegada de cada perturbación). En la política óptima, la adaptación de periodos sólo se lleva a cabo si cambia la tarea que tiene un error mayor (echo que normalmente ocurre - de forma puntual - a la llegada de cada perturbación).

Por ejemplo, en la Figura 5, cuando $t = 0$s todas las tareas se ejecutan con el periodo más largo, 0.5s, suponiendo que los sistemas están en el punto de equilibrio. A los 5s aproximadamente una perturbación afecta al sistema controlado por la tarea 2, con lo cual, al ser el único lazo de control que sufre error, se le asigna un periodo más pequeño (cercano a 0.3s), tanto en la política proporcional como en la óptima. Pero a los 7s, llega una perturbación afectando al sistema controlado por la tarea 3. En la política óptima, al ser el sistema afectado la nueva perturbación el que tiene un mayor error, se le asigna toda la CPU disponible dando un periodo de 0.3s, y la tarea 2 vuelve a tener el periodo máximo, 0.5s. Por contra, en la política proporcional, a partir de los 7s, la tarea 3 va decrementando gradualmente su periodo mientras que la tarea 2 lo va incrementado (debido a que el error en del sistema controlado por la tarea 3 es mayor que el de la tarea 2).

Es interesante destacar que las políticas dinámicas, en contraposición a la tradicional, ofrecen un ahorro de CPU. En una implementación estática, el 97% de la CPU (o cualquiera que sea el factor de utilización deseado U_d para el conjunto de tareas

[5] Si se utilizan distintos procesos caracterizados con distintos índices de rendimiento p_i y pesos w_i, el tipo de resultados que obtendremos serán cualitativamente idénticos a los que presentamos. El uso de sistemas diferentes para cada lazo de control incurriría en una comparación asimétrica de resultados, que puede llevar a confusión, especialmente en la parte que comprende la definición de los escalados de las funciones de rendimiento por medio de w_i, en un experimento como este es difícil definir cual de todos los procesos es más importante.

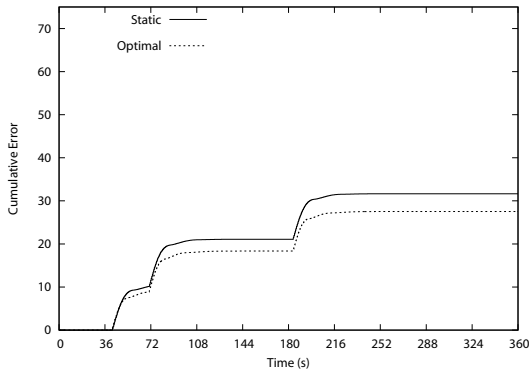

Figura 6. Visión detallada del análisis comparativo del rendimiento de control de las políticas óptima y estática (6min).

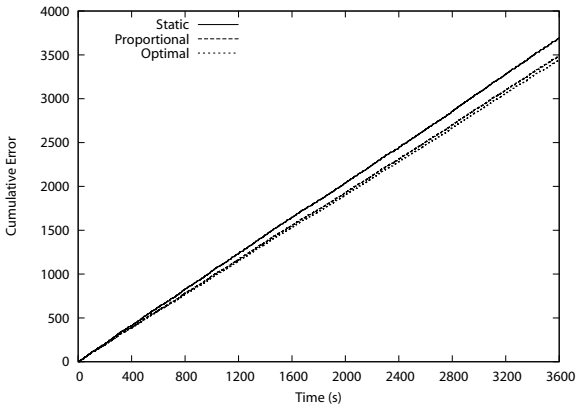

Figura 7. Visión general del análisis comparativo del rendimiento de control de las políticas óptima, proporcional y estática

de control) siempre está ocupado por las tareas de control. En las políticas dinámicas, cuando el intervalo de perturbaciones incrementa, cuando no hay presencia de error en los sistemas controlados, las tareas de control se ejecutan con su periodo máximo, proporcionando un ahorro de CPU que puede ser usado por otras tareas.

6.2 Análisis de rendimiento de control

El análisis de rendimiento se explica a partir de las Figuras 6, 7 y 8.

Figura 6 detalla el análisis de rendimiento comparativo entre la política óptima y estática, durante un periodo de 6min. En el eje vertical se mide el error acumulado total de los tres *ball and beams* (sujetos a un secuencias de perturbaciones generadas de forma aleatoria) que se ha calculado como sigue

$$\int_0^{t_e} \sum_{i=1}^{3} |\boldsymbol{x}_i(t)| dt$$

donde t_e es el tiempo que duró cada experimento y $\boldsymbol{x}_i(t)$ es el vector d'estado de cada *ball and beam* (que coincide con el error si se tiene en cuenta que la referencia y el punto de equilibrio de cada proceso es el cero).

En la Figura 6 se observa como el error acumulado aumenta de forma diferente para cada una de las políticas en los instantes $t = 36$s, $t = 72$s y $t = 180$s, que son los instantes en que alguna de las plantas se ve afectada por una perturbación. Se puede observar que la política óptima, para cada perturbación, minimiza el error acumulado, ofreciendo un mejor rendimiento de control.

En la Figura 7 se hace el mismo estudio comparativo que en la Figura 6, pero para un tiempo de ejecución mas largo, e incluyendo también la política proporcional. De esta figura se concluye que:

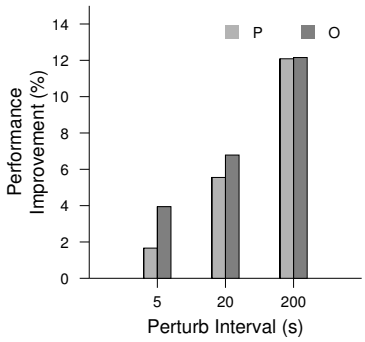

Figura 8. Incremento del rendimiento de control de la políticas dinámicas (O - óptima, P - proporcional) con respecto a la estática para distintos intervalos de perturbaciones

- la política óptima es la que proporciona mejor rendimiento comparado con la proporcional y la estática, ya que es la que presenta un menor error acumulado
- las políticas dinámicas (óptima y proporcional) dan un mejor rendimiento que la estática, hecho que también justifica asignar recursos dinámicamente en función de los estados de los sistemas controlados es una buena estrategia.

Se observa en la Figura 7 que la diferencia entre la política óptima y la proporcional no es tan significativa. Una posible explicación tendría su origen en la secuencia aleatoria de perturbaciones. Si el intervalo entre perturbaciones es corto, la probabilidad de que todas las tareas tengan error es alta, y por tanto, la política óptima y la proporcional asignaran recursos de forma muy diferente. Pero si el intervalo entre perturbaciones se incrementa, la probabilidad de que todas las tareas tengan error disminuye, con lo cual, la política proporcional asignaría de la misma forma que la óptima, proporcionando entonces un rendimiento de control muy parecido. Esto echo se ilustra a continuación.

Figura 8 resume el análisis de rendimiento (en porcentaje) de las políticas óptima (O) y proporcional (P) con respecto a la estática para distintos intervalos de perturbaciones (5s, 20s y 200s). Un intervalo de perturbaciones define el tiempo medio que transcurre entre perturbaciones que afectan una misma planta. El porcentaje de incremento de rendimiento está calculado con respecto a la política estática. De la Figura 8 obtenemos las siguientes observaciones:

- Primero, las conclusiones obtenidas anteriormente a partir de la Figura 7 se mantienen para los diferentes intervalos de perturbaciones: la política óptima es la que proporciona mejor rendimiento de control.
- Segundo, se puede observar la siguiente tendencia. Cuanto más largo es el intervalo de perturbaciones, mejor se comportan las políticas dinámicas. Cuando el intervalo es suficientemente corto (5s) como para que siempre haya presencia de error en los sistemas controlados, la mejora del rendimiento no puede ser tan buena como cuando el intervalo es suficientemente largo ($200s$) de forma que las perturbaciones no estén solapadas.
- finalmente, apuntar que cuando el intervalo de perturbaciones es largo, la política proporcional se comporta como la óptima, por las razones explicadas anteriormente.

En resumen, el análisis del rendimiento de control muestra los beneficios de las políticas dinámicas y avala los resultados teóricos presentados.

6.3 Análisis de sobrecarga

El análisis de sobrecarga se ha centrado en estudiar de forma comparativa la utilitzación de CPU y la sobrecarga de las políticas dinámicas frente a la estática.

La consumo de CPU de las tareas de control en las políticas dinámicas permite un ahorro de procesador, cuya magnitud depende de los distintos intervalos entre perturbaciones. Para intervalos entre perturbaciones pequeños, no hay ahorro de procesador. Pero conforme el intervalo crece, aparece un ahorro de CPU, que crece al incrementar el intervalo.

Cuando el intervalo entre perturbaciones es largo, la probabilidad de que todos los sistemas de control estén en equilibrio aumenta. Y en estos casos las tareas de control se ejecutan con su h_i^{max}, lo que permite un ahorro de procesador. Por ejemplo, para el intervalo entre perturbaciones de 200s, el ahorro the CPU de las políticas dinámicas alcanza casi un 10%. Los consumos de CPU de las tareas de control en la política óptima y en la proporcional son muy parecidos, independientemente del intevalo entre perturbaciones.

La sobrecarga debida a la reasignación de recursos en las políticas dinámicas es negligible comparada con el consumo de CPU de las tareas de control y las sobrecargas de los cambios de contexto y del planificador. La sobrecarga de los cambios de contexto, del planificador y de la reasignación de recursos se situa entre un 0.3% y 0.4% de la utilización total de CPU de las tareas de control. La evaluación realizada muestra que los cambios de contexto son los responsables de la mayor parte de sobrecarga, 75% aproximadamente, y la planificación es responsable del 25% restante.

Comparando las políticas dinámicas, aun siendo la sobrecarga negligible (cercana al 0%), la política proporcional introduce más sobrecarga ya que implica un mayor número de adaptaciones de periodos y por tanto, una mayor frecuencia de canvios en la asignación de recursos (tal y como se comentó en la sección 6.1).

7. CONCLUSIONES

En este trabajo se han presentado las políticas de asignación de recursos para tareas de control de tiempo real basadas en la dinámica de los sistemas controlados. Hemos planeado el problema de asignación de recursos como un problema de optimización lineal con restricciones, cuya solución ofrece la política óptima en términos de rendimiento de control.

Los resultados experimentales efectuados sobre un sistema de tiempo real confirman los resultados teóricos, y demuestran que la asignación dinámica de recursos en lazo cerrado ofrece un rendimiento de control superior a la política estática de asignación de recursos que tradicionalmente se aplica a las tareas de control los sistemas de tiempo real.

Además las políticas dinámicas ofrecen la posibilidad de ahorrar CPU sin degradar el rendimiento de control. Esta propiedad puede ser utilizada para gestionar situaciones de sobrecarga en el sistema de tiempo real.

Finalmente, la evaluación de sobrecarga demuestra que la sobrecarga debida a la reasignación de recursos de las políticas dinámicas es negligible comparada con la utilización de CPU de las tareas de control y las sobrecargas debidass a los canvios de contexto y a la planificación.

El trabajo futuro se centrará en estudiar la viabilidad de aplicar las políticas dinámicas de asignación de recursos a tareas de control presentadas en este trabajo a sistemas de control de tiempo real distribuidos.

REFERENCIAS

Albertos, P. and J. Salt (1990). Digital regulators redesign with irregular sampling. In: *11th IFAC World Congress (Preprints)*. Vol. 8. pp. 157–161.

Åström, K. J. and B. Wittenmark (1997). *Computer-Controlled Systems. Third Edition*. Prentice–Hall.

Balbastre, P., I. Ripoll, J. Vidal and A. Crespo (2004). A task model to reduce control delays. *Journal of Real-Time Systems* **27**(3), 215–236.

Beccari, G., S. Caselli, M. Reggiani and F. Zanichelli (1999). Rate modulation of soft real-time tasks in autonomous robot control systems. In: *Proceedings of the 11th Euromicro Conference on Real-Time Systems*.

Brandt, S. A. and G. Nutt (2002). Flexible soft real-time processing in middleware. *Real-Time Systems* **22**, 77–118.

Brandt, S. A., S. Banachowski, C. Lin and T. Bisson (2003). Dynamic integrated scheduling of hard real-time, soft real-time and non-real-time processes. In: *Proceedings of the 24th IEEE Real-Time Systems Symposium*. pp. 396–407.

Buttazzo, G., M. Velasco, P. Marti and G. Fohler (2004). Managing quality-of-control performance under overload conditions. In: *Proceedings of the 16th Euromicro Conference on Real-Time Systems*.

Caccamo, M., G. Buttazzo and L. Sha (2000). Elastic feedback control. In: *Proceedings of the 12th Euromicro Conference on Real-Time Systems*. pp. 121–128.

Cervin, A. and J. Eker (2000). Feedback scheduling of control tasks. In: *Proceedings of the 39th IEEE Conference on Decision and Control*. Sydney, Australia.

Cervin, A., J. Eker, B. Bernhardsson and K-E. Årzén (2002). Feedback-feedforward scheduling of control tasks. *Real-Time Systems* **23**, 25–53.

Chong, E.K.P. and S.H. Zak (1996). *An Introduction to Optimization*. John Villey and Sons, Inc.

Dogruel, M. and U. Özgüner (1995). Stability of a set of matrices: A control theoretic approach. In: *Proceedings of the 34th IEEE Conference on Decision and Control*.

Gellert, W., S. Gottwald and M. Hellwich (1988). *The VNR Concise Encyclopedia of Mathematics*. Van Nostrand Reinhold Company.

Henriksson, D., A. Cervin, J. Åkesson and K-E. Årzén (2002). On dynamic real-time scheduling of model predictive controllers. In: *Proceedings of the 41th IEEE Conference on Decision and Control*. Las Vegas, NV.

Lin, Caixue, Pau Marti, Scott A. Brandt, Scott Banachowski, Manel Velasco and Josep M. Fuertes (2004). Improving control performance using adaptive quality of service in a real-time system.

Liu, C. L. and J. W. Layland (1973). Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the Association for Computing Machinery* **20**(1), 46–61.

Marti, P., G. Fohler, K. Ramamritham and J. M. Fuertes (2002). Improving quality-of-control uisng flexible time constraints: Metric and scheduling issues. In: *Proceedings of the 23nd IEEE Real-Time Systems Symposium*.

Marti, P., J. M. Fuertes, G. Fohler and K. Ramamritham (2001*a*). Jitter compensation for real-time control systems. In: *Proceedings of the 22nd IEEE Real-Time Systems Symposium*.

Marti, P., R. Villà, J. M. Fuertes and G. Fohler (2001*b*). On real-time control tasks schedulability. In: *Proceedings of the European Control Conference*. Porto, Portugal.

Marti, P., R. Villà, J. M. Fuertes and G. Fohler (2001*c*). Stability of on-line compensated real-time scheduled control tasks. In: *IFAC Conference on New Technologies for Computer Control*. Hong Kong, P.R. China.

Palopoli, L., C. Pinello, A. L. Sangiovanni-Vincentelli, L. El-Ghaoui and A. Bicchi (2002). Synthesis of robust control systems under resource constraints. **LNCS 2289**, 337–350.

Ramanathan, P. (1999). Overload management in real-time control applications using (m, k)-firm guarantee. *IEEE Transactions on Parallel and Distributed Systems* **10**(6), 549–559.

Rehbinder, H. and M. Sanfridson (2000). Integration of off-line scheduling and optimal control. In: *Proceedings of the 12th Euromicro Conference on Real-Time Systems*. Stockholm, Sweden. pp. 137–143.

Seto, D., J.P. Lehoczky, L. Sha and K.G. Shin (1996). On task schedulability in real-time control systems. In: *Proceedings of the 17th IEEE Real-Time Systems Symposium*.

Shin, K. G. and C. Meissner (1999). Adaptation adn graceful degradation of control system performance by task reallocation and period adjustment. In: *Proceedings of the 11th Euromicro Conference on Real-Time Systems*. pp. 29–37.

Wittenmark, B. and K. J. Åström (1980). Simple self-tuning controllers. *Unbehauen, Ed. Methods and Applications in Adaptive Control, Lecture Notes in Control and Information Sciences* **24**, 21–29.

Zhao, Q. and D-Z. Zheng (1999). Stable and real-time scheduling on a class of perturbed hybrid dynamic systems. In: *IFAC World Congress*. pp. 91–96.