ISSN 2007-9737

Modelado y reconstrucción de tiempos de respuesta en computadoras de placa reducida con RT-Linux

Diana Lizet González Baldovinos, Pedro Guevara López

Instituto Politécnico Nacional,
Escuela Superior de Ingeniería Mecánica y Eléctrica Culhuacán, CDMX,
México

glez_lizet@hotmail.com, pguevara@ipn.mx

Resumen. Los Sistemas de Tiempo Real no críticos pueden desarrollarse integrando tres componentes básicos: una computadora de placa reducida Raspberry Pi, un sistema operativo con una distribución GNU/Linux y la extensión de tiempo real PREEMPT_RT; para su desarrollo, dimensionamiento y correcto funcionamiento, es necesario conocer el comportamiento de los tiempos de respuesta de sus tareas para cumplir con la condición de predecibilidad y cumplimiento de plazos. En este sentido, es necesario definir ciertas condiciones de operación como: tarea en estudio, prioridad, complejidad del algoritmo en ejecución, esto con el objetivo de conocer la dinámica de la tarea en estudio y determinar su comportamiento.En este trabajo se presenta una memoria de Tesis Doctoral donde se desarrollaron dos modelos, el primero denominado ab initio y el segundo basado en el cociente de esparanzas matemáticas recursivas, a partir del cual se generaron dos algoritmos de estimación. Los tiempos de respuesta son generados por la tarea en estudio que consiste en la inversión de matrices de dimensiones 32×32, 64×64 y 128×128. La tarea está asociada a la política de planificación Round Robin con la mayor prioridad.

Palabras clave. Cociente de esperanzas matemáticas, dinámica de tiempos de respuesta, modelo de reconstrucción, RT-Linux.

Modeling and Reconstruction of Response Times on Single Board Computers with RT-Linux

Abstract. Non-critical real-time systems can be developed by integrating three essential components: a Raspberry Pi single board computer, an operating system with a GNU/Linux distribution, and the PREEMPT_RT real-time kernel extension; for its development, sizing, and correct operation, it is necessary to know the behavior of the response times of its tasks to meet the condition of predictability and compliance with deadlines. For this, it is necessary to define certain operating conditions such as the task to be studied, the priority, and complexity of the algorithm in execution, to know the dynamics of the task under study and determine its behavior. This paper presents a report on Ph.D. Thesis where two models were developed, the first one called ab initio and the second one based on the recursive mathematical expectation quotient, from which two estimation algorithms were generated. The response times under study were obtained from a task of the inversion of matrices with dimensions: 32×32, 64×64, and 128×128. The task under study is associated with the Round Robin scheduling policy with the highest priority.

Keywords. Mathematical expectation quotient, reconstruction model, response times dynamic, RT-Linux.

1. Introducción

Los Sistemas Operativos de Tiempo Real (SOTR) son un tipo especial de sistema operativo que debe contar con características adicionales a las ofrecidas por un sistema operativo de tiempo compartido, ya que en general, interactúan con procesos del mundo real y deben respetar sus restricciones temporales [5].

Este tipo de sistemas operativos, de acuerdo a su diseño, pueden ser propietarios o extensiones de tiempo real; éstas últimas pueden estar basadas en el kernel de Linux, y permiten tener

un SOTR a disposición, sin tener que pagar una licencia o utilizar hardware específico.

Por ello, en este trabajo se usa RT-Linux, que ofrece características de tiempo real a través del parche PREEMPT_RT, este parche proporciona la capacidad de ejecutar tareas de tiempo real y manejador de interrupciones en la misma máquina que Linux estándar [22, 13].

Todas las tareas computacionales generan tiempos de respuesta que dependen del hardware y el software del ordenador. Los tiempos de respuesta de las tareas ejecutadas en sistemas operativos en tiempo real como RT-Linux pueden variar a medida que evolucionan sus instancias aunque siempre ejecuten el mismo algoritmo.

El modelado de tiempos de respuesta, permite representar matemáticamente el comportamiento del sistema para conocer cómo podría comportarse en diferentes escenarios y conocer los peores.

En el caso de los sistemas de tiempo real, es bastante significativo hacerlos a medida para aplicaciones específicas y saber cómo se comportarían en situaciones de estrés con algoritmos que generen alta carga computacional y con prioridades del nivel más alto.

2. Trabajos relacionados

Los trabajos que a continuación se describen soportan y sugieren el uso de RT-Linux. Los autores del artículo [31] establecen que el parche PREEMPT_RT tiene el objetivo de aumentar la predictibilidad y reducir la latencia del kernel.

Además, afirmaron que Linux no puede ser considerado un sistema de tiempo real estrictamente, al menos no para escenarios de seguridad crítica.

En [24], los autores muestran que el parche RT es adec uado para las unidades que hacen hincapié en el procesamiento de datos, que dependen en gran medida de IPC (comunicación entre procesos).

En el artículo de [37] se muestran sorprendentes experimentos usando Linux y RT-Linux y los resultados destacan que el kernel optimizado por el parche RT-Linux generaba menos retraso que el kernel en la programación multihilo.

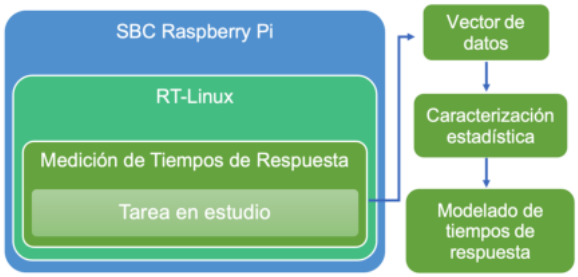

Fig. 1. Banco de pruebas para desarrollar este trabajo

```
Inicio
Definir la dimensión de la matriz M
Definir 1000 iteraciones
Declarar las variables k y para calcular tiempos
Asignar prioridad y planificador al proceso actual
Crear archivo de datos
Llamar a la función iniciar_matriz ()
Para todas las k <= iteraciones hacer
     Medir el tiempo de inicio
     Llamar a la función matriz_inversa ()
     Medir el tiempo final
     Convertir el tiempo inicial y final en milisegundos
     Tiempo de respuesta=Tiempo final-Tiempo inicial
     Imprimir el tiempo de respuesta
     Guardar el tiempo de respuesta
Fin
```

Fig. 2. Pseudocódigo del algoritmo de inversión de matrices para medición de tiempos de respuesta

En la tesis de maestría de [13], se probó y comparó el rendimiento de dos parches de tiempo real Xenomai y PREEMPT_RT.

Para el análisis de los tiempos medidos se utilizó el primer y segundo momento de probabilidad, que mostró el mejor rendimiento con el parche PREEMPT_RT y tuvo menores varianzas y redujo el tiempo de respuesta.

Todos estos trabajos muestran algunas ventajas del uso de RT-Linux frente a Linux estándar y otros parches de tiempo real.

Por lo tanto, los tiempos de respuesta de las tareas ejecutadas en sistemas operativos de tiempo real como RT-Linux pueden variar a medida que evolucionan sus instancias, aunque siempre ejecuten el mismo algoritmo; esto se debe a diversas causas como los tiempos de operación, los tiempos de espera, el ruido de las mediciones, el jitter, la programación, el paso de mensajes, las interrupciones de hardware y software, entre otras [34].

```
for (k=0; k<ITERACIONES; k++){
clock_gettime(CLOCK_REALTIME, &inicio);
matriz_inversa=inversa(matriz, M);
clock_gettime(CLOCK_REALTIME,&fin);
milisegundos_i=(inicio.tv_sec*1000.0)+((double)inicio.tv_nsec/1000000.0);
milisegundos_f= (fin.tv_sec*1000.0) + ((double)fin.tv_nsec/1000000.0);
ejecucion= milisegundos_f-milisegundos_i;
printf ("r(%d)= %f milisegundos\n", k, ejecucion);
sprintf(buffer, "\n%f\n", ejecucion);
write(des, buffer, sizeof(buffer));
}
```

Fig. 3. Fragmento de código, medición de los tiempos de respuesta para inversión de una matriz [13]

```
struct sched_param sched;
sched.sched_priority=sched_get_priority_max(SCHED_RR);
sched_setscheduler(0, SCHED_RR, &sched)
```

Fig. 4. Fragmento de código, asignación de planificador y alta prioridad

Esta variación disminuye a medida que aumenta la prioridad de las tareas; sin embargo, los tiempos de respuesta mínimos y máximos siguen estando presentes en la misma tarea y esto complica su seguimiento, disminuyendo su nivel de predictibilidad en caso de contingencia o sobrecarga, y además, dificultando el dimensionamiento de los recursos.

En el trabajo de los autores [23] se describen los tiempos de ejecución de las tareas periódicas; estos tiempos se utilizan para calcular la utilización alcanzable del procesador, y este concepto fue la base de futuras investigaciones.

[20] explican que los tiempos de respuesta de un sistema en tiempo real RT_i es la suma de los requisitos computacionales para todas las entradas de niveles superiores que se producen en un intervalo. Sjodin y Hansson en [33] presentaron un ejemplo de análisis de tiempo de respuesta.

Además, mostraron una ecuación del tiempo de respuesta mediante la suma del bloqueo máximo de los procesos de menor prioridad, el jitter máximo, el peor tiempo, el tiempo de computación, el periodo de las tareas y el plazo.

En este sentido, [4] presentaron una sencilla ecuación recursiva para determinar los tiempos de respuesta del mejor caso de las tareas periódicas bajo una programación preventiva de prioridad fija y un escalonamiento arbitrario. Los autores [30] siguieron el mismo sentido de los trabajos anteriores; presentaron una ecuación de recurrencia para calcular los tiempos de respuesta del mejor caso de un conjunto de tareas periódicas con prioridades fijas.

La solución se basa en la identificación de la fase del mejor caso de una tarea de baja prioridad en comparación con las tareas de mayor prioridad.

Este escalonamiento se produce cuando la tarea de baja prioridad se libera de forma que termina simultáneamente con las liberaciones de todas las tareas de mayor prioridad cuando éstas han experimentado su máxima fluctuación de liberación.

Hasta ahora, las cinco referencias revisadas tienen algo en común. Los autores proponen algún tipo de modelo teórico de tiempos de respuesta. Todos ellos se basan en un modelo de tiempos de respuesta en diferentes escenarios.

En [2], los autores presentan un modelo para estimar el tiempo de respuesta en el peor de los casos de tareas esporádicas con prioridades fijas en un uniprocesador preventivo, especifican que dependiendo del nivel de prioridad, las tareas pueden tener mayor o menor tiempo de respuesta, considerando que la respuesta para el peor caso no deba exceder el plazo máximo.

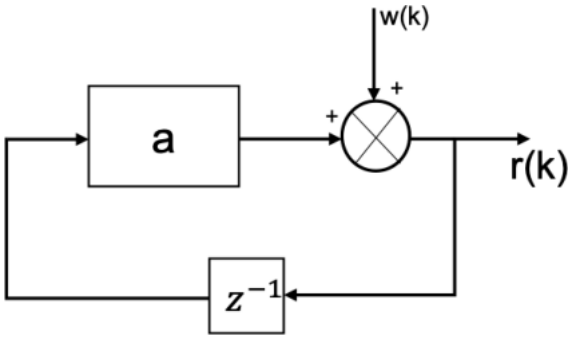

Fig. 5. Diagrama a bloques del modelo lineal de primer orden propuesto de acuerdo a la ecuación 5

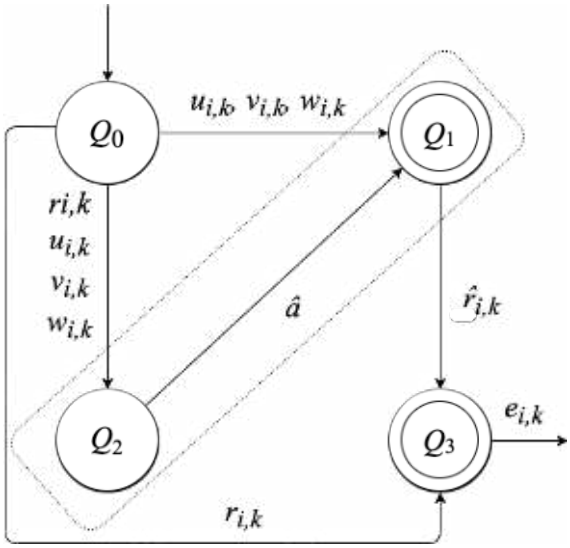

Fig. 6. Diagrama de estados R para reconstrucción de la dinámica de los tiempos de respuesta de una tarea con alta prioridad sobre RT-Linux [12]

De igual manera influye la complejidad computacional del algoritmo, otro factor es la continuidad en el algoritmo procesado, cuando el incremento de tiempo es muy pequeño se tendrán que hacer más operaciones por periodo; además, si se busca una buena aproximación a un sistema continuo se deben considerar tiempos de muestreo muy pequeños.

Identifican tres propiedades deseables: continuidad, computabilidad eficiente y aproximabilidad. Otros trabajos que siguieron esta idea son los de [3, 9, 15, 1].

Los autores [26, 25] presentan un enfoque estadístico para el análisis del tiempo de respuesta de los sistemas embebidos en tiempo real, y su trabajo se basa en la teoría del valor extremo, las simulaciones de Monte Carlo y otros métodos estadísticos para obtener una estimación probabilística; los autores aseveran que el análisis de tiempos de respuesta debe hacerse con un enfoque estadístico (no es suficiente un enfoque determinístico) para calcular los peores tiempos de respuesta; una característica de este trabajo son sus resultados experimentales basados en el análisis de datos.

En [28], los autores proponen una estimación probabilística del tiempo de respuesta en el peor de los casos orientada a sistemas multinúcleo en tiempo real.

Su trabajo implica la generación de datos con clasificación de muestras e igualación del tamaño de las mismas, y la estimación se basa en un modelo de distribución de valores extremos y un método de ajuste del modelo de distribución de Pareto generalizado.

También se presenta la detección de umbrales y la estimación de parámetros.

Por último, [32] presentaron y demostraron un novedoso análisis del mejor caso exacto de tiempo de respuesta para tareas periódicas independientes en tiempo real con plazos arbitrarios programadas utilizando una programación de prioridad fija con umbrales de anticipación, los autores presentaron un desarrollo teórico completo y diferentes escenarios para probar el modelo.

3. Desarrollo

3.1. Banco de pruebas

El banco de pruebas se conforma por (figura 1) una Computadora de Placa Reducida (SBC) Raspberry Pi, un Sistema Operativo Real-Time Linux, como objeto de prueba se utiliza un proceso que genera alta carga computacional temporal basado en inversión de matrices por el método de Gauss-Jordan, se implementa mecanismo de planificación de tiempo real y se asigna la prioridad más alta al proceso; se miden los tiempos de

respuesta y se almacenan en un vector de datos para realizar la caracterización estadística; una vez analizado el comportamiento de los tiempos se propone un modelo matemático que permita representar y replicar su dinámica bajo diferentes condiciones de operación.

3.2. Algoritmo de inversión de matrices

Para fundamentar este trabajo experimental, se tomó como objeto de prueba el proceso de inversión de matrices apoyado en [16, 8, 13].

En la Figura 2 se muestra el pseudocódigo del algoritmo, puede observarse también que se realiza la medición del tiempo antes de llamar a la función `matriz_inversa()`, en esta sección del algoritmo se utiliza la función `clock_gettime()`, la cual tiene amplia resolución en el orden de nanosegundos y por ende entrega mediciones con alta precisión.

Las funciones que integra la biblioteca `matrices.h` son las siguientes: `iniciar_matriz()`, `llenar_matriz()`, `sumar_matriz()`, `producto_matriz()` e `inversa()`.

El algoritmo tiene el propósito de generar un proceso de alta carga computacional temporal en el sistema operativo, su complejidad es $O(n^3)$ y se emplea para hacer la medición de los tiempos de respuesta, la inversión de matrices se efectúa con las siguientes dimensiones: 32×32, 64×64 y 128×128, para cada dimensión se realizan 1000 inversiones, con el objetivo de observar el comportamiento de los tiempos de respuesta como métrica cuantitativa.

Es importante mencionar que la matriz se llena con valores pseudoaleatorios de punto flotante, se genera el llenado una sola vez y esa matriz se invierte 1000 veces.

La razón de utilizar este algoritmo, es debido a la complejidad computacional temporal y número de operaciones que involucra a medida que se incrementan las dimensiones de la matriz a invertir; Por otra parte, cabe señalar que las matrices tienen muchas aplicaciones en áreas como control, filtrado, imágenes, robótica, videojuegos, inteligencia artificial, etc.

Fig. 7. Diagrama de estados Q_0 para la medición experimental de la dinámica de los tiempos de respuesta [12]

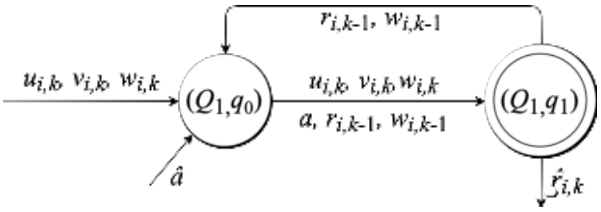

Fig. 8. Diagrama de estados Q_1 para el modelo de reconstrucción de la dinámica de los tiempos de respuesta [12]

La medición de los tiempos de respuesta se realiza dentro del algoritmo, para efectuar las mediciones, existe una serie de funciones dentro de la biblioteca `time.h`, la función que se utiliza para medir tiempos de ejecución es `clock_gettime()`, en la Figura 3 se muestra el fragmento de código donde se realiza la medición del tiempo de respuesta.

Los tiempos de respuesta medidos se guardan en un archivo de texto, para ser graficados fuera de línea en el software MATLAB.

3.3. Asignación de planificador y prioridad de tiempo real

De acuerdo con el trabajo de [13], para implementar mecanismo de planificación de tiempo real se utiliza `sched_setscheduler()` de la biblioteca `sched.h`, esta llamada establece tanto el mecanismo de planificación, como los parámetros para el hilo cuyo ID se especifica en el pid, este mecanismo de planificación requiere permisos de superusuario.

Pid es el ID de proceso al que se asignará la prioridad, puede establecerse con valor 0 para indicar que la prioridad se asignará al proceso actual.

Tabla 1. Lista de variables [12]

Variable	Descripción
J	Conjunto de tareas
J_i	Tarea
$j_{i,k}$	Instancia
$l_{i,k}$	Tiempo de arribo
$o_{i,k}$	Tiempo de operación
$s_{i,k}$	Tiempo de inicio
$c_{i,k}$	Tiempo de ejecución
$p_{i,k}$	Tiempo de desalojo
$f_{i,k}$	Tiempo de finalizado
$r_{i,k}$	Tiempo de respuesta
$u_{i,k}$	Entrada del sistema
$v_{i,k}$	Ruido interno
$w_{i,k}$	Ruido externo
$e_{i,k}$	Error de reconstrucción
$\hat{r}_{i,k}$	Tiempo de respuesta reconstruído
i	Índice de tarea
k	Índice de instancia
n	Número total de tareas
m	Número total de instancias
α	Número total de segmentos de tiempo
a	Parámetro del sistema
$\hat{a}$	Parámetro estimado del sistema
$\mu e^2_{i,k}$	Error cuadrático medio

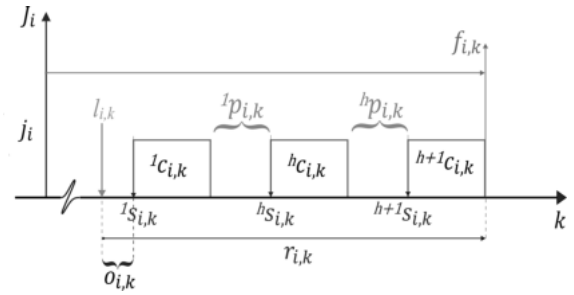

Fig. 9. Esquema de restricciones temporales para una instancia $j_{i,k}$ de una tarea J_i. Las restricciones de tiempo implican el tiempo de arribo $l_{i,k}$, el tiempo de inicio $s_{i,k}$, el tiempo de desalojo $p_{i,k}$, el tiempo de ejecución $c_{i,k}$ y el tiempo de finalizado $f_{i,k}$, h representa el número de segmento de cada restricción [12]

Policy es la política de planificación, con base en [29] `SCHED_RR`, es una política de planificación Round Robin similar a `SCHED_FIFO`, con la excepción de que los procesos que tienen el mismo nivel de prioridad se les asigna un timeslice cada 4 veces el periodo de reloj.

Finalmente, param es un puntero a la estructura `sched_param` el cual contiene la prioridad que se requiere asignar al proceso, en este argumento se especifica si se requiere la prioridad máxima o mínima del planificador en cuestión, esto se hace mediante las siguientes funciones las cuales retornan el valor entero de 1 y 99 respectivamente, [14].

- `sched_get_priority_min(policy).`
- `sched_get_priority_max(policy).`

En este trabajo se utiliza el planificador `SCHED_RR()` y la prioridad más alta, ya que en un estudio anterior [13], se observó en el análisis experimental, que los tiempos de respuesta tenían menores fluctuaciones y el proceso no era desalojado por otras tareas, razón por la cual se emplea sólo este planificador y la prioridad más alta 99 para el proceso de alta carga computacional temporal, es importante mencionar que cuando se abre el monitor de procesos esta prioridad se ve reflejada como rt, es decir, de tiempo real.

Por lo tanto, el recurso del procesador está enfocado en darle atención al proceso en ejecución, de tal manera que evita que sea desalojado por otras tareas hasta que termine su ejecución.

La asignación de prioridad y planificador se muestra en el fragmento de código de la figura 4.

3.4. Caracterización estadística de tiempos de respuesta

La caracterización estadística permite conocer la dinámica del sistema o fenómeno que se esté analizando, por ello el primer paso para caracterizar es calcular los momentos de probabilidad, que corresponden a la media y varianza, los cuales para este caso se calcularán de manera recursiva, pues es de interés conocer la dinámica de los tiempos de respuesta a medida que evoluciona el experimento, si se realizara el cálculo aplicando directamente las ecuaciones se obtendría un valor general, lo cual no permitiría indagar en el comportamiento de los tiempos de respuesta [14].

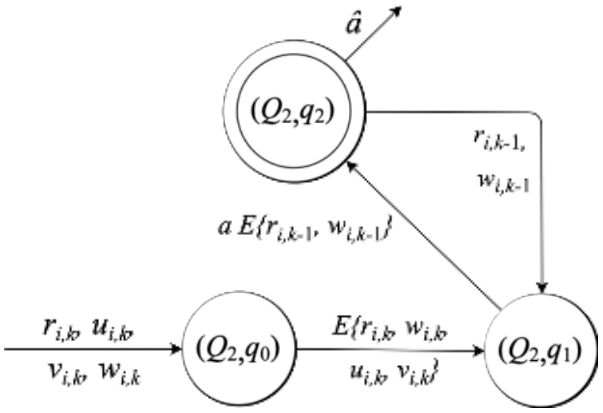

Fig. 10. Diagrama de estados Q_2 para estimación del parámetro $\hat{a}_{i,k}$ [12]

Los momentos de probabilidad indican la dinámica de los tiempos de respuesta conforme a la evolución del experimento y la dispersión que existe de los datos alrededor de la media recursiva.

También, proporciona información sobre la naturaleza de los tiempos, es decir, si son homogéneos (estacionarios) o heterogéneos (No estacionarios), dando pauta a su análisis con un mejor enfoque para elegir una técnica de modelado y reconstrucción adecuadas.

La Función de Densidad de Probabilidad (FDP) junto con los momentos de probabilidad presentan información de la estacionariedad del proceso; se espera que sea gaussiana y simétrica para aplicar técnicas de reconstrucción como mínimos cuadrados o filtro de Kalman, de otra manera deberá normalizarse [14].

3.5. Primer momento de probabilidad o esperanza matemática

El primer momento de probabilidad es la media o valor esperado de una variable aleatoria x, denotada por μ. Por lo tanto, la media de una variable aleatoria se considera como una cantidad numérica alrededor de la cual los valores de la variable aleatoria tienden a agruparse [21, 13].

Para calcular la media recursiva se parte de la ecuación (1) y se aplica diferencias finitas para obtener la ecuación de la media recursiva, como se representa en (2):

$$\mu(r_{i,k}) = \frac{1}{k}\sum_{k=1}^{k}(r_{i,k}), \tag{1}$$

$$\mu(r_{i,k}) = \frac{(k-1)\mu(r_{i,k-1}+r_{i,k})}{k}. \tag{2}$$

3.6. Segundo momento de probabilidad o varianza

El segundo momento de probabilidad o varianza de una variable aleatoria x es una medida de la dispersión de sus valores alrededor de la media μ y se denota por σ^2 [21, 13].

El cálculo de la varianza recursiva se realiza a partir de la ecuación (3) y de igual forma se aplica diferencias finitas, quedando finalmente la ecuación (4):

$$\sigma^2(r_{i,k}) = \frac{1}{k}\sum_{k=1}^{k}(r_{i,k}-\mu(r_{i,k}))^2, \tag{3}$$

$$\mathrm{Var}(r_{i,k}) = \frac{(k-1)\,\mathrm{Var}(r_{i,k-1}) + (r_{i,k}-\mu(r_{i,k}))^2}{k}. \tag{4}$$

4. Propuesta de modelo ab initio para reconstrucción de tiempos de respuesta

De acuerdo con la caracterización de la dinámica de los tiempos de respuesta, se tiene que es un sistema lineal, estacionario e invariante en el tiempo. Por ello con base en el trabajo [7], se propone el siguiente modelo fundamental para tiempos de respuesta tipo SISO, lineal y de primer orden, presentado en la figura 5 y descrito en la ecuación (5):

$$r(k) = ar(k-1) + w(k), \tag{5}$$

donde:

$r(k)$ = Tiempo de respuesta.
a = Parámetro del sistema.
$w(k)$ = Ruido externo.

$$E\{r(k)\} = aE\{r(k-1) + w(k)\}. \tag{6}$$

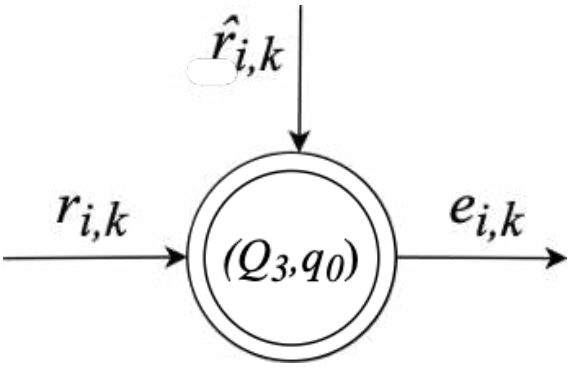

Fig. 11. Diagrama de estados Q_3 para cálculo del error de reconstrucción de tiempos de respuesta de una tarea con alta prioridad sobre RT-Linux [12]

Se procede a despejar el parámetro a de la ecuación (6):

$$\widehat{a} = \frac{E\{r(k) + w(k)\}}{E\{r(k-1)\}}. \tag{7}$$

Con el procedimiento anterior, se tiene el modelo fundamental para reconstrucción de tiempos de respuesta:

$$\widehat{r}(k) = \widehat{a}\widehat{r}(k-1) + w(k). \tag{8}$$

Para validar el modelo de reconstrucción, se calcula el error de reconstrucción mediante la siguiente ecuación:

$$e(k) = \widehat{r}(k) - r(k). \tag{9}$$

Finalmente, se calcula el error de convergencia del modelo propuesto, calculado a través de la ecuación (10):

$$J(k) = E\{e(k)^2\}. \tag{10}$$

5. Propuesta de modelo basado en CEM (cociente de esperanzas matemáticas)

En el modelo ab initio se tuvo un primer acercamiento en el cual se consideró un ruido externo $w_{i,k}$ y el tiempo de respuesta $r_{i,k}$ para generar el algoritmo de estimación y el modelo de reconstrucción.

Sin embargo, al ser una primera aproximación no se logra una adecuada convergencia a los valores reales.

Para mejorar el nivel y velocidad de convergencia, a continuación se presenta el desarrollo de una nueva versión del modelo de tiempos de respuesta, a través de un modelo autorregresivo de media móvil, considerando los resultados de la caracterización estadística de las mediciones experimentales de los tiempos de respuesta generados por una tarea de alta prioridad sobre RT-Linux.

El algoritmo reconstructor se basa en un modelo probabilístico basado en el primer momento de probabilidad para estimar el estado del parámetro.

Mediante el uso de un modelo de tiempos de respuesta y el algoritmo reconstructor es posible calcular el error de reconstrucción para determinar la calidad de la reconstrucción [12].

Para el procedimiento de reconstrucción, se propone un conjunto $R = \{Q_0, Q_1, Q_2, Q_3\}$, de la siguiente manera:

- R: Reconstrucción de la dinámica de los tiempos de respuesta.
- Q_0: Medición de los tiempos de respuesta.
- Q_1: Modelo de reconstrucción de los tiempos de respuesta.
- Q_2: Estimación de parámetro.
- Q_3: Error de reconstrucción de los tiempos de respuesta.

En términos formales, el conjunto R puede representarse como un diagrama de estados, como se muestra en la figura 6.

El primer estado es Q_0, en el que se necesitan varios experimentos para medir los tiempos de respuesta de las instancias para ejecutar el mismo algoritmo n veces.

El estado Q_1 es el modelo de reconstrucción de los tiempos de respuesta; en este estado, se propone un modelo estocástico recursivo para calcular los tiempos de respuesta $\hat{r}_{i,k}$ de las instancias $j_{i,k}$.

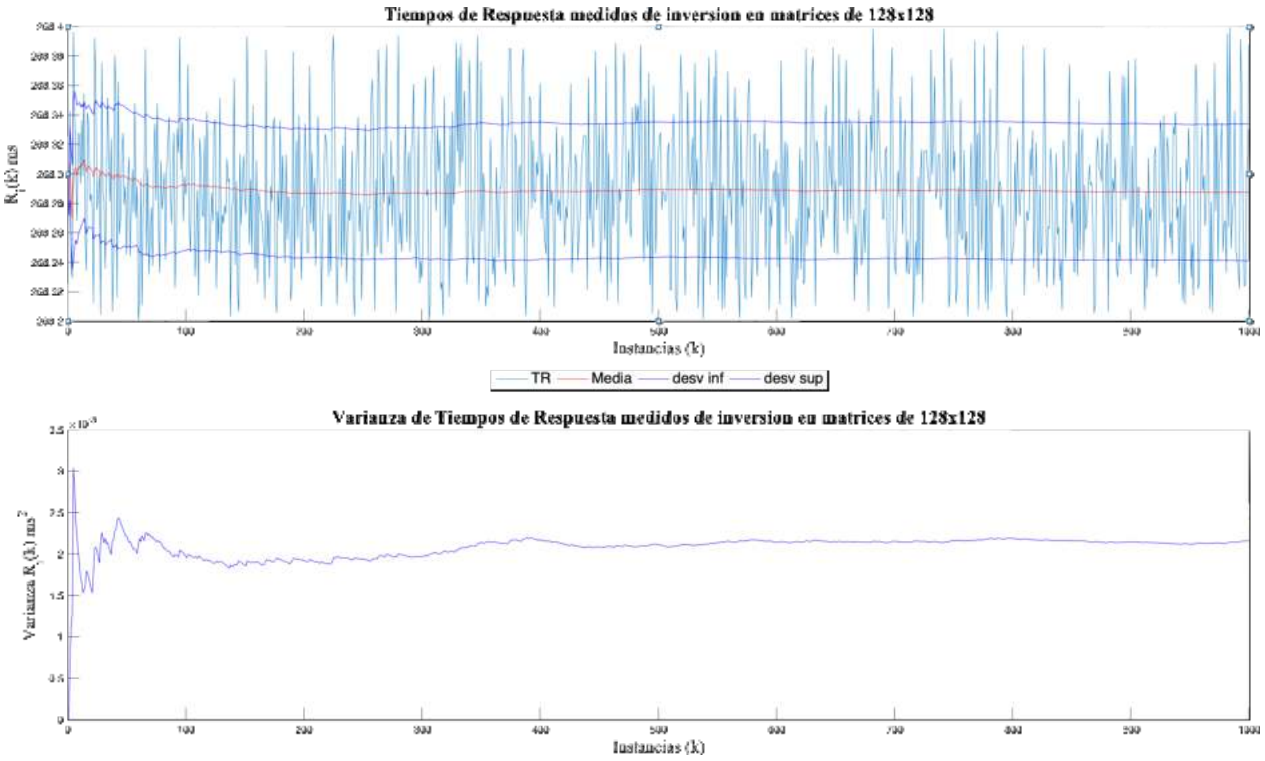

Fig. 12. Gráfica de primer y segundo momentos de probabilidad y desviación estándar de tiempos de respuesta, para experimento de inversión de matrices de 128×128 [14]

Tabla 2. Elementos del banco de pruebas

Elemento	Característica
SBC	Raspberry Pi 3 modelo B
Sistema operativo	RT-Linux con parche PREEMPT_RT
Planificador	Round Robin (SCHED_RR)
Prioridad	Alta prioridad (99)
Algoritmo	Inversión de matrices

El estado Q_2 es un estimador de parámetros basado en CEM, y el resultado de este estado es a, que es una entrada para el estado Q_1.

Por último, el estado Q_3 se utiliza para calcular el error de reconstrucción y validar la calidad de reconstrucción.

Obsérvese que los estados Q_1 y Q_2 están en un rectángulo dibujado con una línea discontinua en la Figura 6 para indicar que dos estados conforman el reconstructor de la dinámica de los tiempos de respuesta.

Siguiendo este contexto, para cada estado se propone un diagrama de estados de segundo nivel para representar los procedimientos internos a fin de obtener el respectivo resultado.

5.1. Q_0: Medición de tiempos de respuesta

En el estado Q_0, se miden los tiempos de respuesta generados por la ejecución de una tarea de inversión de matrices con alta prioridad en RT-Linux. El algoritmo de inversión matricial se programó previamente en lenguaje C, como muestra el pseudocódigo de la Figura 2 en la sección 3.3.

En el algoritmo, se utiliza la función `sched_setscheduler()`, donde se especifica la política de tiempo real Round Robin `SCHED_RR()` y la prioridad, donde se utiliza la función `sched_get_priority_max()` [29], para obtener la máxima prioridad del planificador, que devuelve un valor entero de 99.

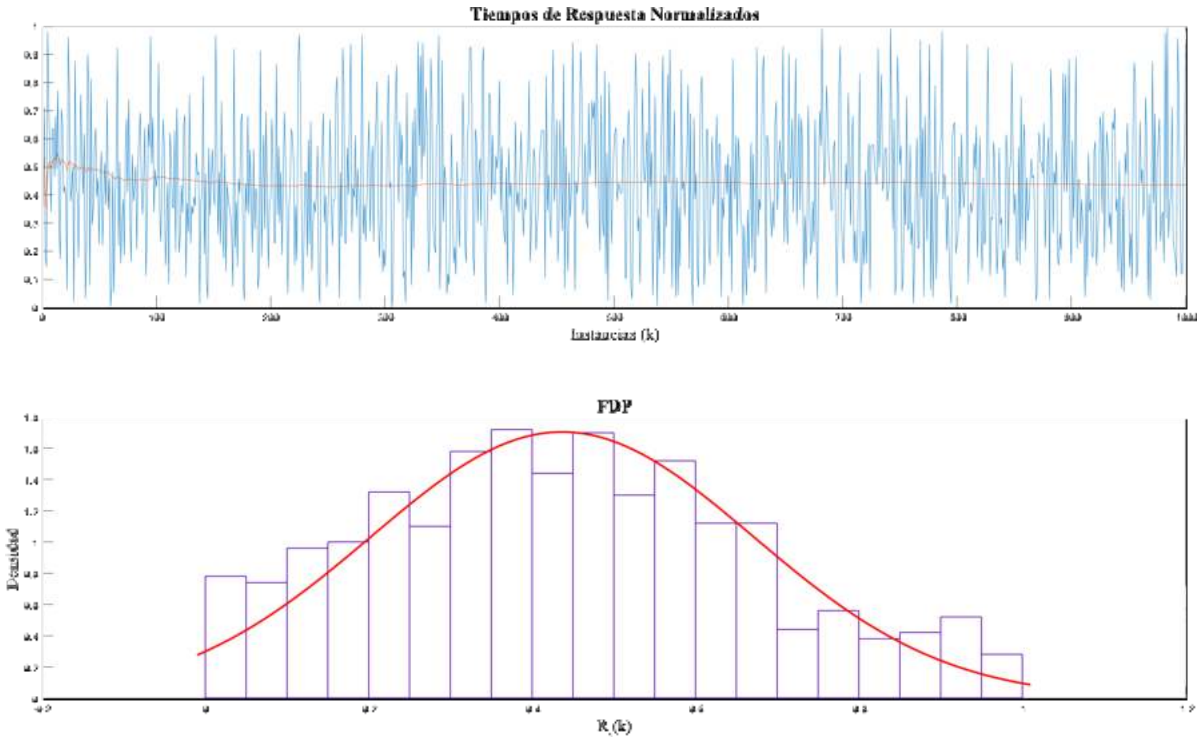

Fig. 13. Gráfica de tiempos de respuesta normalizados y función de densidad de probabilidad, para experimento de inversión de matrices de 128×128 [14]

Para la medición de los tiempos, se utiliza la función `clock_gettime()` ya que tiene una resolución en el orden de nanosegundos.

Utilizando estas funciones, se obtienen los tiempos de respuesta de cada instancia $j_{i,k}$. Este procedimiento se puede representar mediante el diagrama de estados de la Figura 7, y a su vez tiene dos sub estados:

- $Q_0 = \{(Q_0, q_0), (Q_0, q_1)\}$.
- (Q_0, q_0): Cálculo de la medición de tiempos de respuesta.
- (Q_0, q_1): Obtención de las variables a modelar.

Para cada instancia $j_{i,k}$, se miden las restricciones temporales $l_{i,k}$ y $f_{i,k}$, luego el estado q_0 es la diferencia entre $f_{i,k}$ y $l_{i,k}$, que se calculan y guardan como $r_{i,k}$. A continuación, se obtienen las variables de entrada $u_{i,k}, v_{i,k}, w_{i,k}$, para ser modeladas.

5.2. Q_1: Modelo de reconstrucción de tiempos de respuesta

En esta parte, se propone un conjunto de definiciones necesarias, un lema y un teorema para desarrollar el modelo de reconstrucción como se muestra en la Figura 8 con los siguientes estados:

- $Q_1 = \{(Q_1, q_0), (Q_1, q_1)\}$.
- (Q_1, q_0): Recepción de la entrada del sistema $u_{i,k}$ ruidos internos y externos $v_{i,k}$, $w_{i,k}$, respectivamente.
- (Q_1, q_1): Cálculo de la reconstrucción de los tiempos de respuesta.

Para obtener el diagrama de estados Q_1, es necesario formular las definiciones 1 a 4 y proponer un lema y un teorema; este diagrama incluye todo el desarrollo de esta sección.

ISSN 2007-9737

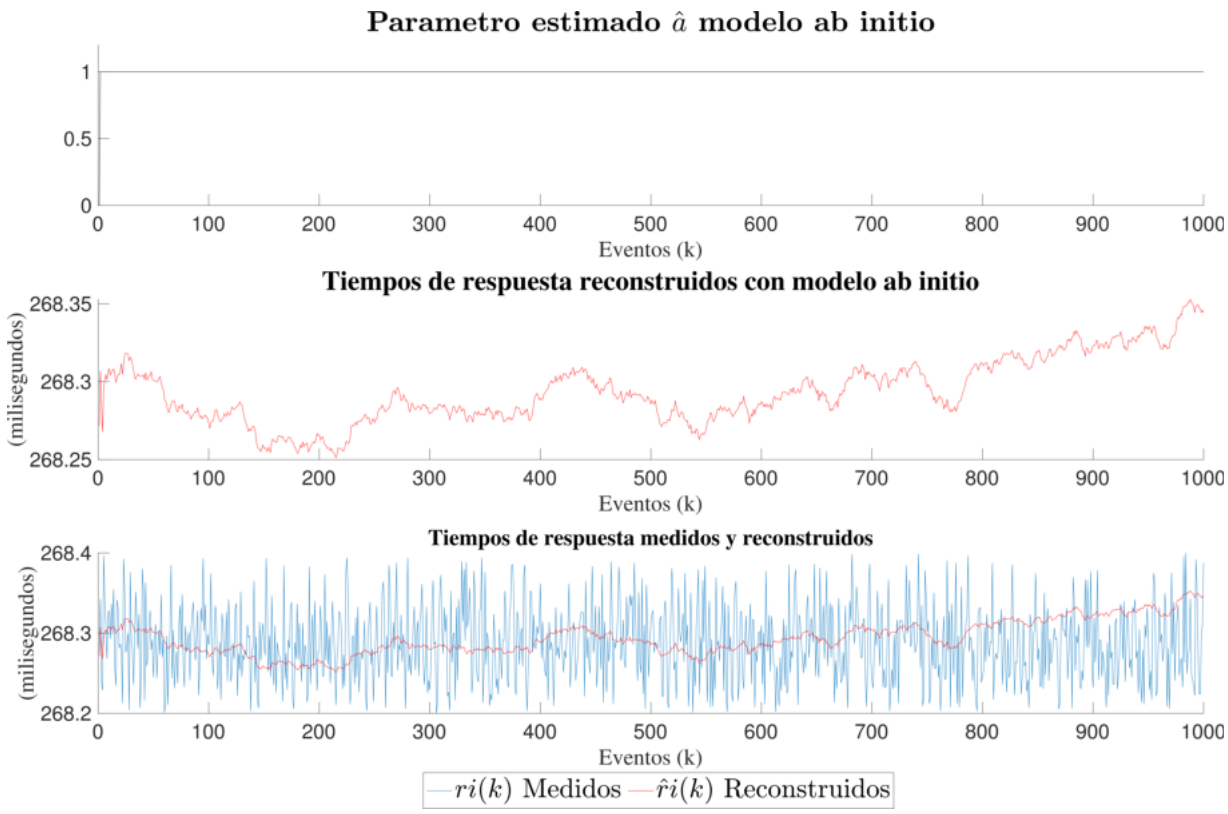

Fig. 14. Gráfica de parámetro estimado a, tiempos de respuesta reconstruidos $ri(k)$ y gráfica comparativa de tiempos de respuesta medidos vs tiempos de respuesta reconstruidos con modelo ab initio, para inversión de matrices de 128×128

Definición 1 (Tiempo de respuesta $r_{i,k}$). Cada instancia $j_{i,k}$ de la tarea J_i tiene un tiempo de respuesta, como:

$$r_{i,k} = f_{i,k} - l_{i,k}, \tag{11}$$

Con $r_{i,k}, f_{i,k}, l_{i,k} \in \mathbb{R}^+$ y $i, k \in \mathbb{N}$. Se considera que la dinámica de los tiempos de respuesta es la variación de los tiempos de respuesta de las instancias de una tarea, con respecto a la evolución temporal del sistema computacional; considerando que todas las instancias ejecutan el mismo algoritmo con prioridad alta fija.

La hipótesis de partida es la siguiente: la dinámica de los tiempos de respuesta de una tarea puede reconstruirse a partir de un modelo matemático, un parámetro estimado y la medición y caracterización de un conjunto de tiempos de respuesta de una tarea con prioridad alta fija [12].

Definición 2 (Tiempo de operación $o_{i,k}$).

El tiempo de operación $o_{i,k}$ de una instancia $j_{i,k}$ es el tiempo transcurrido que la instancia es atendida desde su hora de llegada, esto es: $o_{i,k} = s_{i,k} - l_{i,k}$, con $i, k \in \mathbb{N}$. $o_{i,k}$ es único e indivisible para cada instancia [12].

Definición 3 (Tiempo de ejecución $c_{i,k}$).

El tiempo de ejecución $c_{i,k}$ de una instancia $j_{i,k}$ es el tiempo que la instancia computa sus operaciones hasta que termina, con $i, k \in \mathbb{N}$ [12].

Definición 4 (Tiempo de desalojo $p_{i,k}$).

El tiempo de desalojo $p_{i,k}$ de una instancia $j_{i,k}$ es una interrupción temporal del tiempo de ejecución $c_{i,k}$ generada por una tarea de mayor prioridad, con $i, k \in \mathbb{N}$.

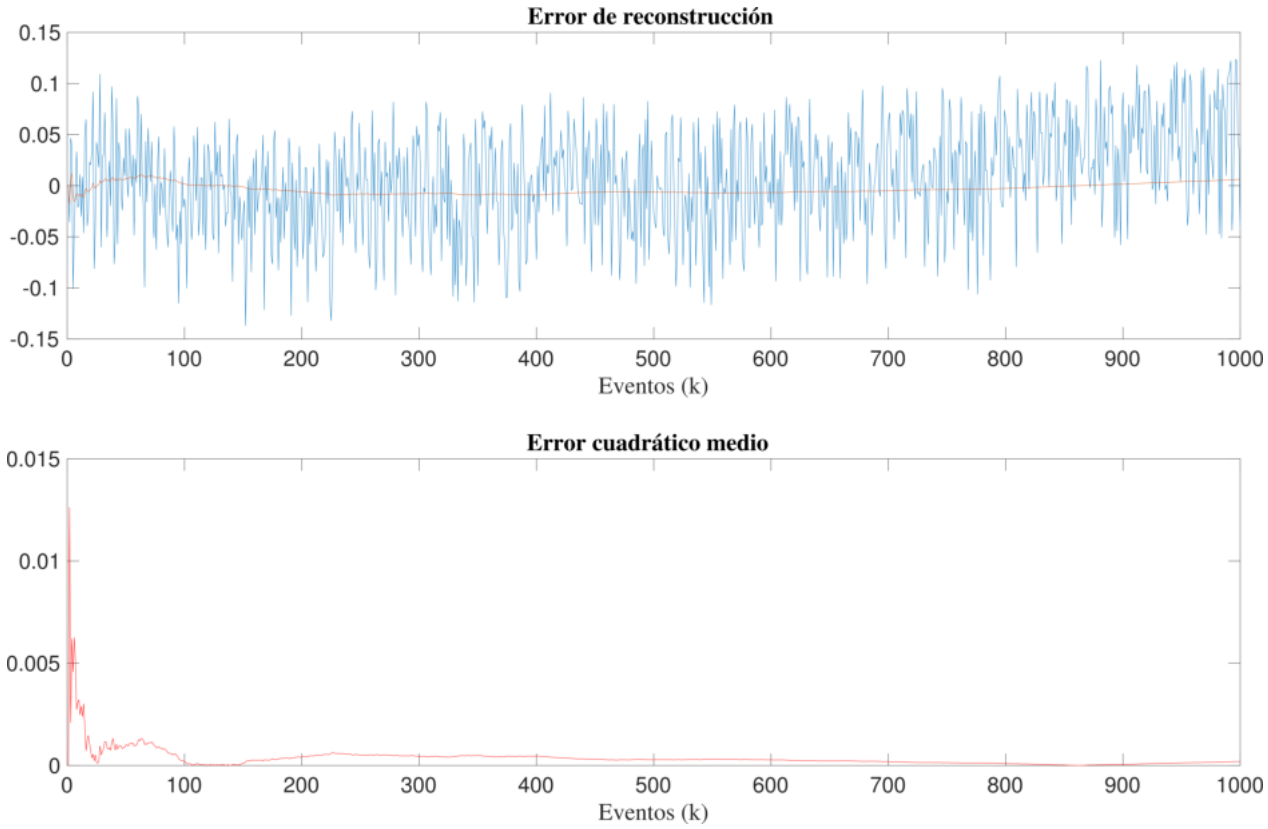

Fig. 15. Gráfica de error del modelo de reconstrucción ab initio y error cuadrático medio de los tiempos de respuesta, para el experimento de inversión de matrices de 128×128

Observe que una instancia $j_{i,k}$ puede tener un conjunto $p_{i,k}$ de h tiempos de espera durante su ejecución y tiene un $c_{i,k}$ dividido. Por lo tanto:

$$p_{i,k} = \sum_{h=1}^{\alpha} {}^{h} p_{i,k}. \tag{12}$$

Lo que implica que:

$$c_{i,k} = \sum_{h=1}^{\alpha+1} {}^{h} c_{i,k}. \tag{13}$$

Las restricciones temporales, la notación y sus relaciones se basan en [6]. En este sentido, se presenta la figura 9, la tabla 1 y todo el desarrollo matemático [12].

Lema 1. El tiempo de respuesta $r_{i,k}$ de una instancia $j_{i,k}$, es la suma aritmética del tiempo de operación $o_{i,k}$ más el tiempo de ejecución $c_{i,k}$ más el tiempo de desalojo $p_{i,k}$ [12]. Por lo tanto:

$$r_{i,k} = o_{i,k} + \sum_{h=1}^{\alpha+1} {}^{h} c_{i,k} + \sum_{h=1}^{\alpha} {}^{h} p_{i,k} \forall\, i,k,h,\alpha \in \mathbb{N}. \tag{14}$$

Prueba. Teniendo en cuenta (13) y sustituyendo en la ecuación (14):

$$r_{i,k} = o_{i,k} + c_{i,k} + p_{i,k}, \tag{15}$$
$$o_{i,k} = s_{i,k} - l_{i,k}, \tag{16}$$
$$r_{i,k} = (s_{i,k} - l_{i,k}) + c_{i,k} + p_{i,k}, \tag{17}$$
$$r_{i,k} = s_{i,k} + c_{i,k} + p_{i,k} - l_{i,k}. \tag{18}$$

Considerando $f_{i,k} = s_{i,k} + c_{i,k} + p_{i,k}$ y sustituyendo en la ecuación anterior de $r_{i,k}$, se obtiene la ecuación (11):

ISSN 2007-9737

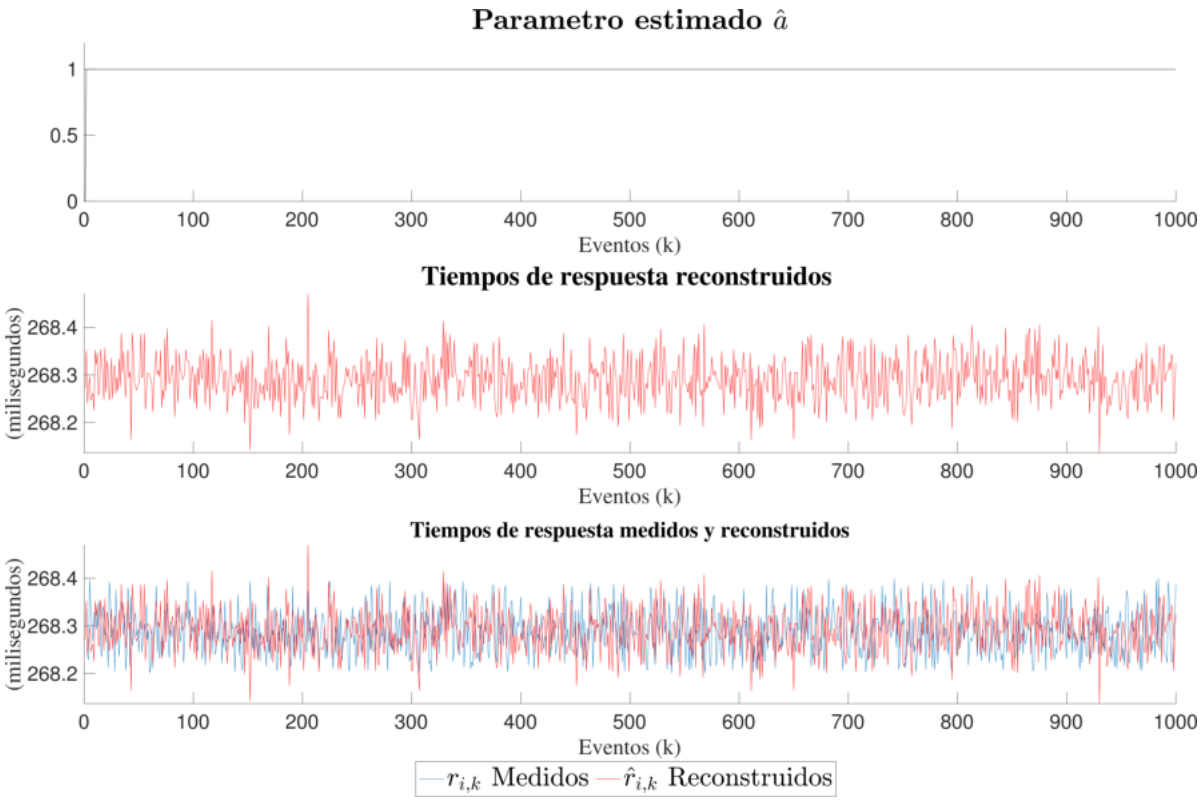

Fig. 16. Gráfica del parámetro estimado $\hat{a}$, tiempos de respuesta reconstruidos $\hat{r}_{i,k}$ y gráfica comparativa de tiempos de respuesta medidos $r_{i,k}$ vs. tiempos de respuesta reconstruidos $\hat{r}_{i,k}$ para el experimento de inversión de matrices de 128×128

$$r_{i,k} = f_{i,k} - l_{i,k} \blacksquare \tag{19}$$

Con las ecuaciones (14) y (11) es posible proponer un modelo dinámico recursivo [18, 17, 19]. Por consiguiente, se obtiene:

Teorema 1. (Dinámica de los tiempos de respuesta $r_{i,k}$). La dinámica de los tiempos de respuesta de una tarea con alta prioridad en un sistema estacionario se describe como un modelo lineal de primer orden con un parámetro invariante en el tiempo [12].

Considerando $u_{i,k}$ (entrada del sistema), a (parámetro del sistema), $w_{i,k}$ (ruido externo) y $v_{i,k}$ (ruido interno) la expresión del modelo se describe como sigue:

$$r_{i,k} = a[r_{i,k-1} - w_{i,k-1}] + u_{i,k} + v_{i,k} + w_{i,k}. \tag{20}$$

La ecuación (20) es un sistema lineal, invariante en el tiempo porque la caracterización dinámica de los tiempos de respuesta de una tarea en tiempo real con alta prioridad es un modelo recursivo, los niveles de ruido están acotados por la desviación estándar respecto a la esperanza matemática de los tiempos de respuesta, y a es un parámetro invariante porque considera que la esperanza matemática es paralela al eje horizontal.

Prueba. Considerando la ecuación (14) $r_{i,k} = o_{i,k} + c_{i,k} + p_{i,k}$, el comportamiento de $o_{i,k}$ y $p_{i,k}$ es aleatorio y sus magnitudes para cada instancia indexada por k son diferentes, esas dependen de otros procesos incluidos el núcleo del sistema operativo.

Entonces, esas son variables aleatorias de procesos estocásticos y se pueden sumar para obtener [12]:

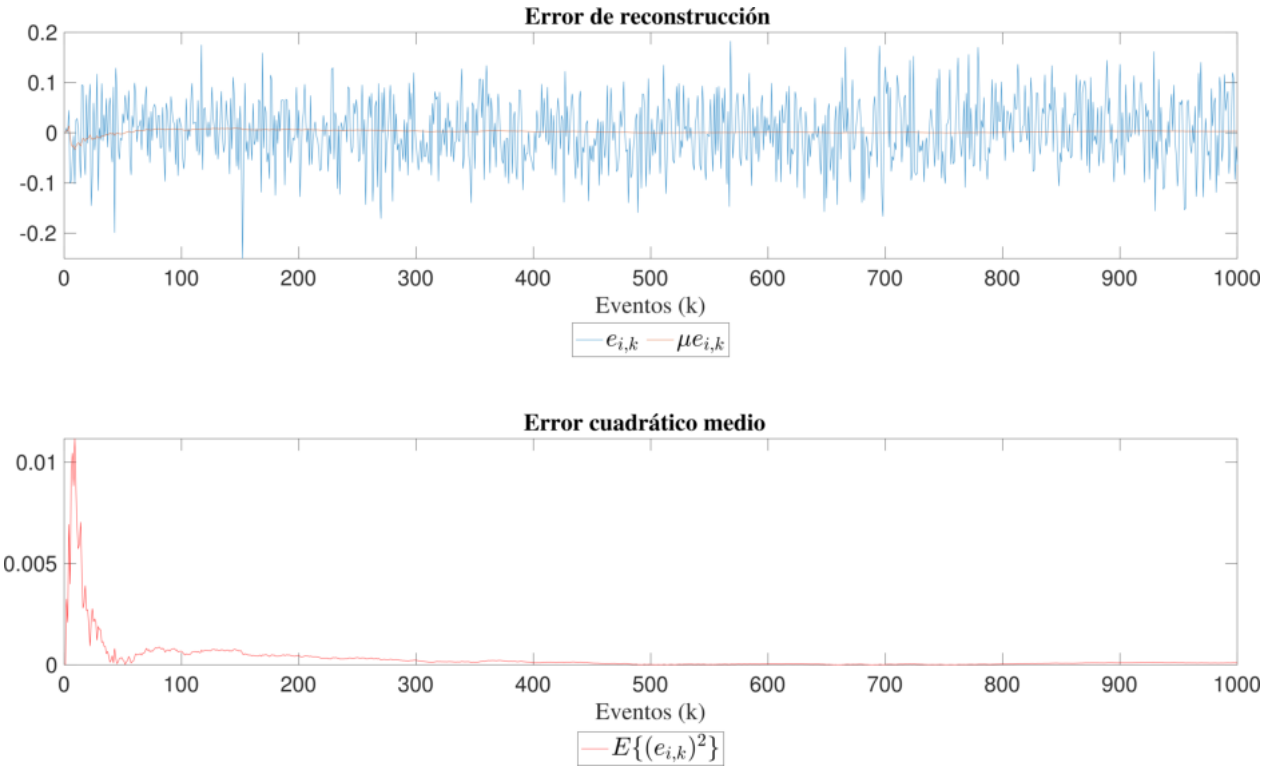

Fig. 17. Gráfica del error de reconstrucción y del error cuadrático medio del experimento de inversión de matrices de 128×128

$$w_{i,k} = o_{i,k} + p_{i,k}. \tag{21}$$

Tal que:

$$r_{i,k} = c_{i,k} + w_{i,k}. \tag{22}$$

Por otro lado, $c_{i,k}$ es un estado interno y no se puede medir directamente. Por esta razón, se propone una ecuación lineal de primer orden con un parámetro invariante en el tiempo, esto es:

$$c_{i,k} = ac_{i,k-1} + u_{i,k} + v_{i,k}. \tag{23}$$

$u_{i,k}$ es el valor de referencia.
$v_{i,k}$ es un ruido interno normalizado con media cero (Se podría considerar jitter).

Siguiendo los párrafos anteriores, se tiene que representar $r_{i,k}$ en función de $r_{i,k-1}$ y no en función de $c_{i,k}$ porque no se conoce el estado interno del sistema. Así que, se despeja $c_{i,k}$ de la ecuación (22) por lo que, se tiene:

$$c_{i,k} = r_{i,k} - w_{i,k}. \tag{24}$$

Aplicando un retardo a la ecuación (24) se obtiene:

$$c_{i,k-1} = r_{i,k-1} - w_{i,k-1}. \tag{25}$$

Sustituyendo la igualdad de las ecuaciones (24) y (25) en la ecuación (23), se obtiene:

$$r_{i,k} - w_{i,k} = a[r_{i,k-1} - w_{i,k-1}] + u_{i,k} + v_{i,k}. \tag{26}$$

Despejando $r_{i,k}$ de la ecuación (26) se obtiene el modelo representado por la ecuación (20):

$r_{i,k} = a[r_{i,k-1} - w_{i,k-1}] + u_{i,k} + v_{i,k} + w_{i,k}$ ■

5.3. Q_2: Estimador de parámetro

Una cuestión importante sobre la estimación de parámetros es cómo calcular el parámetro a.

Entonces, para este estado se necesita estimar un parámetro $\hat{a}$ como se explica en [35], de manera que $\hat{a} \to a$.

ISSN 2007-9737

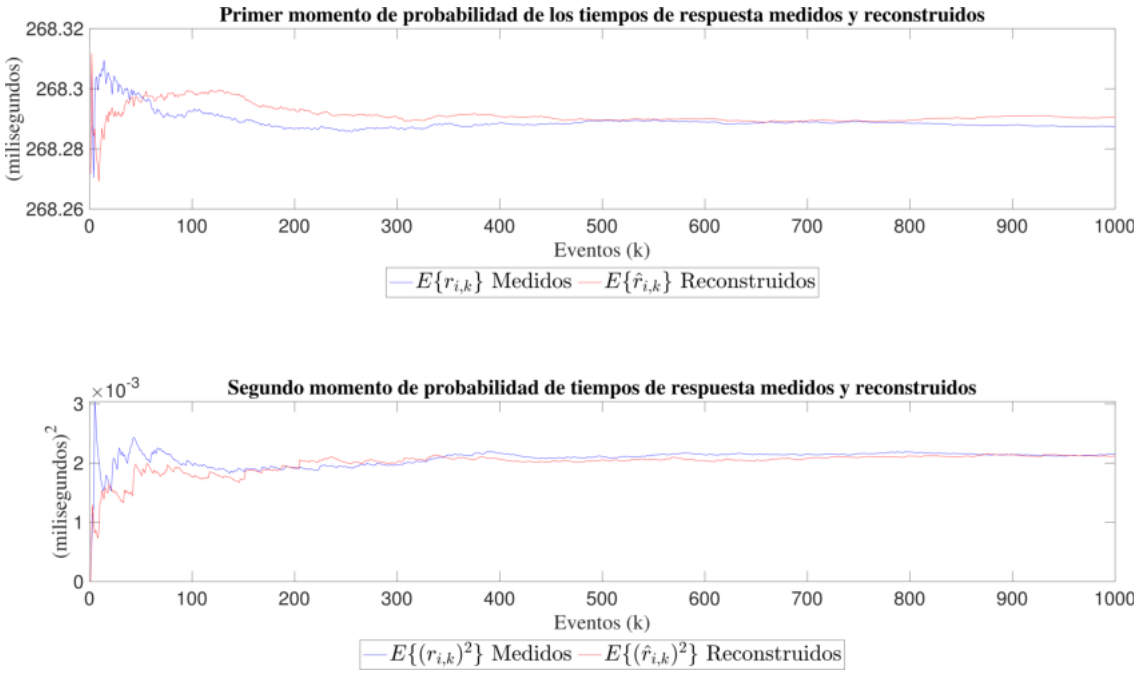

Fig. 18. Primer y segundo momentos de probabilidad de tiempos de respuesta reconstruidos $\hat{r}_{i,k}$ y tiempos de respuesta medidos $r_{i,k}$ para el experimento de inversión de matrices de 128×128

Nótese que la estimación de $\hat{a}$ es el último valor del cálculo recursivo de $\hat{a}_{i,k}$, tal que $\hat{a} = \hat{a}_{i,m}$ para m instancias. Para calcular $\hat{a}$, es necesario tener en cuenta las siguientes condiciones:

- $E\{w_{i,k} r_{i,k-1}\} = 0$,
- $E\{w_{i,k} r_{i,k}\} = \sigma^2_{w_k}$,
- $E\{v_{i,k} r_{i,k}\} = \sigma^2_{v_k}$,
- $E\{v_{i,k} v_{i,k}\} = \sigma^2_{v_{i,k}}$,
- $E\{w_{i,k} w_{i,k}\} = \sigma^2_{w_{i,k}}$,
- $E\{v_{i,k} w_{i,k}\} = 0$.

Con las condiciones anteriores se propone el siguiente lema.

Lema 2. (Estimador de Parámetro $\hat{a}$ para los tiempos de respuesta $r_{i,k}$). El estimador de parámetro para el modelo presentado en el teorema 1, se describe por:

$$\hat{a} = \frac{E\{r_{i,k} - w_{i,k} - u_{i,k} - v_{i,k}\}}{E\{r_{i,k-1} - w_{i,k-1}\}}. \tag{27}$$

Prueba. Debido a que el sistema es lineal, estacionario y de primer orden, se propone construir un estimador basado en CEM, partiendo de la ecuación(26), se aplica esperanza matemática en ambos lados de la igualdad [12]:

$$\begin{aligned} E\{r_{i,k} - w_{i,k} - u_{i,k} - v_{i,k}\} = \\ aE\{r_{i,k-1} - w_{i,k-1}\}. \end{aligned} \tag{28}$$

Y despejando el parámetro a se obtiene $\hat{a}$ como en la ecuación (27). ■

En la figura 10, se presentan los estados Q_1 y Q_2 que forman parte del procedimiento de reconstrucción de tiempos. Entonces, sustituyendo la ecuación (27) en la ecuación (20) se obtienen los tiempos de respuesta reconstruidos $\hat{r}_{i,k}$, esto es:

$$\hat{r}_{i,k} = \hat{a}[\hat{r}_{i,k-1} - w_{i,k-1}] + u_{i,k} + v_{i,k} + w_{i,k}. \tag{29}$$

La ecuación (29) está calculada de forma recursiva y permite observar la dinámica de $\hat{r}_{i,k}$ a través de la evolución del sistema. Del mismo modo, se calcula $\hat{a}$, la reconstrucción de $\hat{r}_{i,k}$ implica que $\hat{r}_{i,k} \to r_{i,k}$ [12].

Tabla 3. Últimos valores del primer y segundo momento de probabilidad, error de reconstrucción y error cuadrático medio de los tiempos de respuesta medidos y generados por los modelos ab initio y CEM, para cada experimento

	Últimos valores	**32×32**	**64×64**	**128×128**
Mediciones	$E\{r_{i,k}\}$	$4.3151\,\mathrm{ms}$	$34.2510\,\mathrm{ms}$	$268.2875\,\mathrm{ms}$
	$E\{(r_{i,k})^2\}$	$8.1334 \times 10^{-6}\,\mathrm{ms}^2$	$1.3903 \times 10^{-4}\,\mathrm{ms}^2$	$0.0022\,\mathrm{ms}^2$
Modelo ab initio	$E\{\hat{r}i(k)\}$	$4.3143\,\mathrm{ms}$	$34.2507\,\mathrm{ms}$	$268.2937\,\mathrm{ms}$
	$E\{(\hat{r}i(k))^2\}$	$1.0916 \times 10^{-6}\,\mathrm{ms}^2$	$2.4606 \times 10^{-5}\,\mathrm{ms}^2$	$4.2910 \times 10^{-4}\,\mathrm{ms}^2$
	$\hat{a}$	1	1	1
	$E\{ei(k)\}$	-7.9364×10^{-4}	-3.5152×10^{-4}	0.0061
	$E\{(ei(k))^2\}$	1.7816×10^{-4}	1.1116×10^{-5}	1.9402×10^{-4}
Modelo de CEM	$E\{\hat{r}_{i,k}\}$	$4.3148\,\mathrm{ms}$	$34.2514\,\mathrm{ms}$	$268.2906\,\mathrm{ms}$
	$E\{(\hat{r}_{i,k})^2\}$	$1.2186 \times 10^{-6}\,\mathrm{ms}^2$	$1.3903 \times 10^{-4}\,\mathrm{ms}^2$	0.0021
	$\hat{a}$	0.9990	0.9990	0.9990
	$E\{e_{i,k}\}$	-2.6673×10^{-4}	3.7965×10^{-4}	0.0030
	$E\{(e_{i,k})^2\}$	1.7454×10^{-6}	1.2006×10^{-5}	9.5556×10^{-5}

5.4. Q_3: Error de reconstrucción de los tiempos de respuesta

Esta etapa es muy importante, se valida la calidad del modelo de reconstrucción propuesto mediante el error de reconstrucción y el error cuadrático medio. Para ello, se presenta el tercer estado como se muestra en la figura 11:

$$Q_3 = \{(Q_3, q_0)\}, \tag{30}$$

donde: (Q_3, q_0) es el cálculo del error de reconstrucción $e_{i,k}$de tiempos de respuesta. El error de reconstrucción de los tiempos de respuesta, consiste en calcular la diferencia entre el tiempo de respuesta estimado y el tiempo de respuesta medido (ver [27]):

$$e_{i,k} = \hat{r}_{i,k} - r_{i,k}. \tag{31}$$

De acuerdo con [11], el error cuadrático medio mide la amplitud de la variación del estimador con respecto al propio estimado, se esperan estimadores con menor error cuadrático medio y más cercanos al valor medido que se busca estimar:

$$\mu e^2_{i,k} = E\{(e_{i,k})^2\}. \tag{32}$$

6. Resultados

Esta sección presenta un compendio de gráficas que ilustran los tiempos de respuesta medidos y reconstruidos a través de los modelos propuestos en la sección de desarrollo.

Los experimentos se realizaron sobre un banco de trabajo configurado por una computadora de placa reducida con sistema operativo RT-Linux y una tarea que ejecuta el algoritmo de inversión matricial programado en lenguaje C.

Es importante resaltar que se llevaron a cabo experimentos para inversión de matrices de dimensión 32×32, 64×64 y 128×128. Sin embargo, para este apartado, se presentan las figuras del experimento de inversión de matrices de 128 × 128.

6.1. Especificación de elementos del banco de pruebas

La tabla 2 especifica los elementos del banco de pruebas y sus características para realizar estos experimentos. Es muy importante mencionar que se considera la ejecución de una tarea en los experimentos realizados.

ISSN 2007-9737

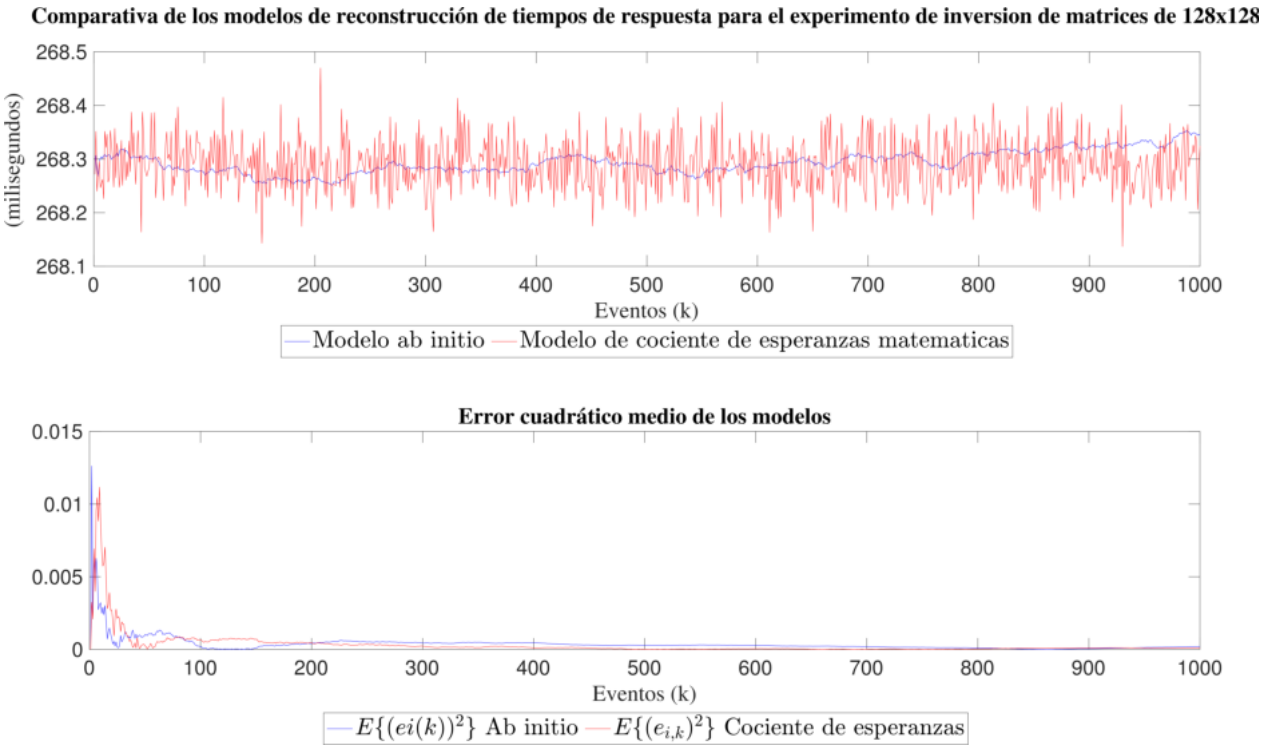

Fig. 19. Comparativa de error de reconstrucción y error cuadrático medio con los modelos ab initio y CEM para el experimento de inversión de matrices de 128×128

Es importante mencionar que previamente en el estudio de [13], se probaron los planificadores FIFO y Round-Robin, el mecanismo de planificación se implementó con la función `sched_setscheduler()`.

Se observó que con el planificador Round-Robin hubo variaciones menores en comparación con FIFO, debido a que Round-Robin asigna timeslices a cada tarea garantizando que los tiempos de respuesta se comporten de manera uniforme en RT-Linux.

6.2. Resultados de caracterización de tiempos de respuesta

A continuación se muestran las gráficas obtenidas del experimento de inversión de matrices de 128×128 al aplicar las ecuaciones (2 y 4) presentadas en la sección de desarrollo, correspondientes al primer y segundo momentos de probabilidad, se grafica también su desviación estándar superior e inferior lo cual es de importancia ya que permite tener una región acotada en donde se tiene mayor concentración de tiempos de respuesta, lo que se encuentra fuera de esa región podría considerarse jitter.

Aunado a esto se grafica su función de densidad de probabilidad, estas herramientas permiten conocer el comportamiento de los tiempos de respuesta, visualizar si son estacionarios o no, y tomarlo como punto de partida para proponer un modelo.

En la Figura 13 se muestran las gráficas de los tiempos de respuesta normalizados, considerando su medida máxima 1 para posteriormente definir un espacio de probabilidad con el objetivo de reconstruir la dinámica de tiempos de respuesta y elegir la mejor técnica a través de estimadores e identificadores estocásticos.

En la Figura 12 se observan los tiempos de respuesta en un espacio estadístico para inversión de matrices de 128×128, se observa

que los tiempos oscilan entre $268.2\,\mathrm{ms}$ y $268.4\,\mathrm{ms}$, su media recursiva tiene un comportamiento fluctuante durante las primeras 100 instancias, pero en el progreso del experimento tiende a mantenerse constante.

En la misma gráfica se muestra la desviación estándar superior e inferior, donde se representa una región acotada de la mayor concentración de tiempos de respuesta, en la cota superior el tiempo es alrededor de $268.34\,\mathrm{ms}$ y la cota inferior aproximadamente $268.24\,\mathrm{ms}$.

En la varianza recursiva se observan fluctuaciones durante las primeras 400 instancias, pero en el progreso del experimento se mantiene casi constante manteniendo un valor aproximado de $2.4 \times 10^{-3}\,\mathrm{ms}^2$.

En los tres experimentos realizados, la media recursiva tuvo un comportamiento similar durante las primeras 100 instancias y después los tiempos se mantuvieron constantes.

De acuerdo a las gráficas de varianza recursiva se observa de manera general que difícilmente llega a converger a un valor, los datos son fluctuantes parcialmente en la evolución de los experimentos.

En suma, el comportamiento de los tiempos de respuesta se describe como un sistema estocástico estacionario e invariante en el tiempo con estabilidad en media [36], este es un dato muy importante ya que con este antecedente se pasa a la siguiente etapa que consiste en la propuesta de un modelo ARMA y un estimador de parámetros basado en el CEM.

6.3. Resultados del modelo ab initio para reconstrucción de tiempos de respuesta

A partir del análisis estadístico presentado en la sección anterior y con base en la dinámica de los tiempos de respuesta observada, se determinó que el sistema es lineal, estacionario e invariante en el tiempo.

En este apartado, se presentan los resultados del modelo presentado en la sección de desarrollo.

En la figura 14, se muestra un conjunto de gráficas que presentan el parámetro estimado a, el tiempo de respuesta reconstruido empleando el modelo y una superposición entre los tiempos de respuesta medidos y los tiempos de respuesta reconstruidos. Se observa que en las tres figuras, el parámetro estimado converge a 1, lo cual indica que la reconstrucción es inexacta y que el sistema podría considerarse como marginalmente estable.

Esto se confirma al observar en la tercera gráfica que no se tiene un adecuado seguimiento de los tiempos reconstruidos respecto a la dinámica de los tiempos de respuesta medidos, si bien es un comportamiento esperado ya que al ser un modelo ab initio que solo considera un ruido externo, se necesitan más elementos a considerar para que la reconstrucción tenga un mejor seguimiento.

6.4. Validación de modelo ab initio

De acuerdo con los resultados del modelo de reconstrucción, se calcula el error de reconstrucción, que es la diferencia entre el tiempo de respuesta reconstruido y el tiempo de respuesta medido; además se realiza el cálculo del error cuadrático medio y se presentan las gráficas resultantes.

En la figura 15, se observa que el error de reconstrucción tiene un comportamiento fluctuante; si bien tiene valores cercanos a cero, la reconstrucción inexacta causa estas fluctuaciones; el error cuadrático medio tiene una respuesta aceptable, se observa que en el transcurso de los eventos se aproxima asintóticamente a cero en los tres experimentos.

6.5. Resultados del modelo de tiempos de respuesta basado en CEM

Esta etapa de resultados se apoya en la caracterización estadística previamente realizada.

De acuerdo con el Teorema 1 en la sección de desarrollo, el comportamiento de los tiempos de respuesta se describe como un sistema estocástico estacionario e invariante en el tiempo.

Se considera un modelo invariante en el tiempo porque de acuerdo con la caracterización de los tiempos de respuesta medidos, no se observan grandes variaciones en sus magnitudes, por lo que se estima un parámetro constante $\hat{a}$.

Por lo tanto, se presenta el modelo de reconstrucción de los tiempos de respuesta basado en las medidas y la caracterización del sistema. Se considera que el sistema es lineal ya que el comportamiento de la dinámica de los tiempos de respuesta permanece constante dentro de un rango acotado; si el tamaño de la matriz aumenta, entonces la magnitud de los tiempos de respuesta crece exponencialmente, por su complejidad $O(n^3)$ [38].

A continuación, se presentan las gráficas resultantes del modelo de reconstrucción para el experimento de 128×128: En la Figura 16 se observan tres gráficas. La primera representa el parámetro estimado $\hat{a}$. La segunda muestra el resultado de los tiempos de respuesta reconstruidos. Finalmente, en la tercera, se observa una superposición entre las gráficas de los tiempos de respuesta medidos (azul) y los reconstruidos (rojo).

Posteriormente, en la figura 18 se muestran el primer y segundo momentos de probabilidad. Estas figuras son muy importantes porque se puede ver que tiene convergencia en casi todos los puntos, lo que valida el modelo propuesto basado en CEM.

6.6. Validación del modelo basado en CEM

Para validar el modelo de reconstrucción propuesto, la figura 17 muestra el error de reconstrucción y el error cuadrático medio, donde se observa en todos los experimentos que los errores son cercanos a cero.

Con respecto a esto, se confirma que el modelo propuesto tiene una buena convergencia. Para explicar detalladamente los resultados, se presenta una breve discusión de los valores de los últimos datos de cada experimento en la Tabla 3.

6.7. Discusión resultados de los modelos respecto a los valores reales

Para mostrar con mejor detalle los resultados del modelo ab initio y el modelo basado en CEM, este trabajo se sintetiza en la tabla 3, donde se presentan los valores representativos que indican: primer y segundo momentos de probabilidad, valor del parámetro estimado, error de reconstrucción, y error cuadrático medio para cada experimento.

A continuación se muestra una colección de figuras que representan gráficas de error de reconstrucción y error cuadrático medio, de los modelos ab initio (color azul) y CEM (color rojo).

Por último, para el experimento de 128×128, se presenta la figura 19, donde el error de reconstrucción para el modelo de cociente de esperanzas llega al valor 0.0030 y su error cuadrático medio de 9.5556×10^{-5}; en comparación con el modelo ab initio, su último valor en error de reconstrucción es 0.0061, y su error cuadrático medio de 1.9402×10^{-4}.

Obsérvese que el valor del parámetro estimado a; en el caso del modelo ab initio fué de 1 en los tres experimentos, esto significa que el modelo aplicable es marginalmente estable, lo cual justifica la reconstrucción inexacta de los tiempos de respuesta, debido a que en el modelo se considera solo un ruido externo.

En comparación con el modelo de CEM, el parámetro estimado para los tres experimentos fué de 0.9990, pues de acuerdo con la dinámica de la reconstrucción de los tiempos, hay un seguimiento adecuado respecto a los valores medidos, se tiene una mejor aproximación ya que en el modelo se consideran la entrada del sistema, ruido interno y ruido externo.

7. Conclusiones y trabajo futuro

Se construyó un banco de pruebas en una computadora de placa reducida Raspberry Pi, el kernel Linux PREEMPT_RT que conforma al sistema operativo RT-Linux, y una tarea de inversión de matrices programada en lenguaje C. Se destacan las siguientes contribuciones:

- Se desarrolló la teoría necesaria para presentar la dinámica de los tiempos de respuesta.
- Se llevó a cabo el desarrollo de dos modelos para la reconstrucción de tiempos de respuesta de una tarea de alta prioridad sobre RT-Linux.
- Se desarrollaron estimadores de parámetros para cada modelo propuesto.

– Se realizó la validación experimental de los modelos propuestos a través de una tarea real ejecutada con alta prioridad sobre RT-Linux.
– El modelo basado en CEM, tiene una mejor aproximación a los valores reales de acuerdo con la respuesta del error de reconstrucción.

Se realizaron dos propuestas de modelo, la primera a través de un modelo ab initio, donde se consideró al sistema lineal, de primer orden, con una entrada y una salida, y un parámetro a invariante en el tiempo.

La reconstrucción se llevó a cabo a partir de la estimación del parámetro a, basándose en el cociente de esperanzas matemáticas recursivas considerando únicamente el ruido externo al sistema. Sin embargo, al haber sido un primer acercamiento, su nivel y velocidad de convergencia no eran tan buenos.

Para mejorar el modelo ab initio, se consideraron dos elementos más: una entrada $u_{i,k}$ y un ruido interno $v_{i,k}$ normalizado con media cero, que podría considerarse jitter; Con este modelo se tuvo una mejor convergencia a los tiempos de respuesta medidos, y el error de reconstrucción fué muy cercano a cero.

El escenario propuesto para el modelado es estable computacionalmente, aplicable a tareas de alta prioridad en un sistema operativo de tiempo real no propietario como RT-Linux.

El uso del modelo y la reconstrucción se puede centrar en aplicaciones que requieran implementar tareas periódicas, algoritmos iterativos, o modelos que utilicen operaciones matriciales de forma exhaustiva, donde exista la necesidad de mejorar el rendimiento computacional y la eficiencia algorítmica con RT-Linux en una SBC como Raspberry Pi; por ejemplo, en [10].

Los modelos que son referencia en la sección de trabajos relacionados son un precedente para el desarrollo de este trabajo, sin embargo, no realizan análisis experimental. Es muy importante enfatizar el desarrollo de modelos dinámicos, capaces de describir el comportamiento de los tiempos de respuesta del conjunto de instancias de una tarea con alta prioridad en un sistema RT-Linux, totalmente diferente a una descripción cualitativa a través de un análisis estadístico, ya que el objetivo de esos trabajos no era reconstruir el sistema, sino describirlo estadísticamente según sus cualidades. El modelo para reconstruir la dinámica de los tiempos de respuesta tiene dos usos principales para futuras aplicaciones:

En primer lugar, dimensionar el sistema computacional para hacer uso adecuado de sus recursos de memoria y del uso de la CPU, y en segundo lugar, proponer esquemas de tolerancia a fallos, ya que cuando la magnitud de los tiempos de respuesta comienza a aumentar o se observa una alta variación del primer y segundo momento de probabilidad el sistema computacional podría fallar.

Los resultados obtenidos son satisfactorios; sin embargo, se puede proponer otra técnica para estimación de la reconstrucción que podría tener una mejor aproximación. Para futuros trabajos, sería interesante proponer un modelo multivariable que implique la ejecución simultánea de más de una tarea con diferentes niveles de prioridad. Además, proponer otras técnicas de estimación y modelado, basadas en: filtro de Kalman, variable instrumental, lógica difusa, factor de olvido exponencial, aprendizaje automático, entre otras.

Agradecimientos

Los autores agradecemos el apoyo al Consejo Nacional de Ciencia y Tecnología y a la Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Culhuacán del Instituto Politécnico Nacional.

Referencias

1. **Baruah, S. K., Burns, A., Davis, R. I. (2011).** Response-time analysis for mixed criticality systems. 2011 IEEE 32nd Real-Time Systems Symposium, pp. 34–43. DOI: 10.1109/RTSS.2011.12.
2. **Bini, E., Chau Nguyen, T. H., Richard, P., Baruah, S. (2008).** A response-time bound in fixed-priority scheduling with arbitrary deadlines. IEEE Transactions on Computers, Vol. 58, No. 2, pp. 279–286. DOI: 10.1109/TC.2008.167.

3. **Bril, R. J., Lukkien, J. J., Verhaegh, W. F. (2009).** Worst-case response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption. Real-Time Systems, Vol. 42, No. 1-3, pp. 63–119. DOI: 10.1007/s11241-009-9071-z.

4. **Bril, R. J., Steffens, L., Verhaegh, W. F. (2001).** Best-case response times of real-time tasks. Philips Workshop on Scheduling and Resource Management, pp. 19–27.

5. **Burns, A., Wellings, A. (1996).** Real-time systems and programming languages addwesley.

6. **Buttazzo, G. C. (2011).** Hard real-time computing systems: Predictable scheduling algorithms and applications. Vol. 24. DOI: 10.1007/978-1-4614-0676-1.

7. **Cano, J. L., González, D. L., Guevara, P., Hernández, L. (2020).** Modelo ab initio para reconstrucción de tiempos de respuesta en procesos con alta prioridad en una SBC con RT-Linux. Jornada de Ciencia y Tecnología Aplicada, Vol. 3, No. 1, pp. 103–108.

8. **Cano, L. (2015).** Efecto del Overclocking sobre los tiempos de ejecución generados por inversión de matrices en una computadora embebida. Master's thesis, Sección de Estudios de Posgrado e Investigación. ESIME Culhuacán. Instituto Politécnico Nacional, México City.

9. **Davis, R. I., Burns, A. (2008).** Response time upper bounds for fixed priority real-time systems. Symposium on Real-Time Systems (RTSS'08). DOI: 10.1109/RTSS.2008.18.

10. **Delgado-Reyes, G., Guevara-Lopez, P., Loboda, I., Hernandez-Gonzalez, L., Ramirez-Hernandez, J., Valdez-Martinez, J. S., Lopez-Chau, A. (2020).** State vector identification of hybrid model of a gas turbine by real-time kalman filter. Mathematics, Vol. 8, No. 5, pp. 659. DOI: 10.3390/math8050659.

11. **Edge, M. D. (2019).** Statistical thinking from scratch: A primer for scientists. DOI: 10.1093/oso/9780198827627.001.0001.

12. **Gonzalez-Baldovinos, D. L., Guevara-López, P., Cano-Rosas, J. L., Valdez-Martínez, J. S., Lopez-Chau, A. (2022).** Response times reconstructor based on mathematical expectation quotient for a high priority task over RT-Linux. Mathematics, Vol. 10, No. 1, pp. 134.

13. **González, D. (2018).** Análisis Experimental de los Tiempos de Respuesta en RT- LINUX para una SBC. Master's thesis. DOI: 10.1088/1742-6596/1237/5/052017.

14. **González, D., Cano, L., Esparza, C., Guevara, P. (2019).** Caracterización estadística de los tiempos de respuesta de un algoritmo complejo en RT-Linux. IX Congreso Internacional de Computación, pp. 72–78.

15. **Guan, N., Stigge, M., Yi, W., Yu, G. (2009).** New response time bounds for fixed priority multiprocessor scheduling. 2009 30th IEEE Real-Time Systems Symposium, pp. 387–397. DOI: 10.1109/RTSS.2009.11.

16. **Guevara, P. (2004).** Filtrado Digital en Tiempo Real: Análisis Computacional para Estimación de Parámetros en Sistemas Estocásticos Lineales Estacionarios. Ph.D. thesis, Centro de Investigación en Computación y Sistemas IPN.

17. **Guevara, P., Medel, J., Cruz, D. (2004).** Modelo dinámico para una tarea en tiempo real. Computación y Sistemas, Vol. 8, No. 1.

18. **Gustafsson, F. (2000).** Adaptive filtering and change detection, Vol. 1.

19. **Haykin, S. S. (2008).** Adaptive filter theory.

20. **Joseph, M., Pandya, P. (1986).** Finding response times in a real-time system. The Computer Journal, Vol. 29, No. 5, pp. 390–395. DOI: 10.1093/comjnl/29.5.390.

21. **Kreyszig, E. (2005).** Spline interpolation. Advanced Engineering Mathematics. Wiley: Hoboken, NJ, USA, pp. 810–816.

22. **Kuppens, H. (2011).** The basics of real-time linux. Radboud University Nijmegen.

23. **Liu, C. L., Layland, J. W. (1973).** Scheduling algorithms for multiprogramming in a hard-real-time environment. Journal of the Association for Computing Machinery, Vol. 20, No. 1, pp. 46–61. DOI: 10.1145/321738.321743.

24. **Liu, Y., Su, Y., Ma, Y., Wang, J. (2018).** Real time optimization of linux system in aerospace. International Journal of Performability Engineering, Vol. 14, No. 12, pp. 3257–3264. DOI: 10.23940/ijpe.18.12.p35.32573264.

25. **Lu, Y., Nolte, T., Bate, I., Cucu-Grosjean, L. (2012).** A statistical response-time analysis of real-time embedded systems. 2012 IEEE 33rd Real-Time Systems Symposium, pp. 351–362. DOI: 10.1109/RTSS.2012.85.

26. **Lu, Y., Nolte, T., Kraft, J., Norstrom, C. (2010).** A statistical approach to response-time analysis of complex embedded real-time systems. 2010 IEEE 16th international conference on embedded and real-time computing systems and applications, pp. 153–160. DOI: 10.1109/RTCSA.2010.13.

27. **Medel Juárez, J. J., Guevara López, P. (2002).** Comparación de la dinámica en tiempo real de los métodos mínimos cuadrados y variable instrumental para estimación de parámetros. VIII Congreso Argentino de Ciencias de la Computación, pp. 687–697.

28. **Mucha, M., Mottok, J., Deubzer, M. (2015).** Probabilistic worst case response time estimation for multi-core real-time systems. 4th Mediterranean Conference on Embedded Computing, pp. 31–36. DOI: 10.1109/MECO.2015.7181918.

29. **QNX (2007).** Qnx neutrino real-time operating system.library reference.

30. **Redell, O., Sanfridson, M. (2002).** Exact best-case response time analysis of fixed priority scheduled tasks. Proceedings 14th Euromicro Conference on Real-Time Systems. Euromicro RTS 2002, pp. 165–172. DOI: 10.1109/EMRTS.2002.1019196.

31. **Reghenzani, F., Massari, G., Fornaciari, W. (2019).** The Real-Time Linux Kernel: A survey on preemptrt. ACM Computing Surveys, Vol. 52, No. 1, pp. 1–36. DOI: 10.1145/3297714.

32. **Rivera-Verduzco, H. J., Bril, R. J. (2017).** Best-case response times of real-time tasks under fixed-priority scheduling with preemption thresholds. Proceedings of the 25th International Conference on Real-Time Networks and Systems, pp. 307–346. DOI: 10.1145/3139258.3139260.

33. **Sjodin, M., Hansson, H. (1998).** Improved response-time analysis calculations. Proceedings 19th IEEE Real-Time Systems Symposium, pp. 399–408. DOI: 10.1109/REAL.1998.739773.

34. **Stappert, F., Altenbernd, P. (2000).** Complete worst-case execution time analysis of straight-line hard real-time programs. Journal of Systems Architecture, Vol. 46, No. 4, pp. 339–355. DOI: 10.1016/S1383-7621(99)00010-7.

35. **Valdez Martínez, J. S., Reyes, G. D., Lopez, P. G., García Infante, J. C. (2014).** Reconstrucción de la dinámica de los tiempos de ejecución de tareas en tiempo real empleando filtrado digital difuso. Revista Facultad de Ingenieria Universidad de Antioquia, Vol. 1, No. 70.

36. **Villavicencio, J. (2010).** Introducción a series de tiempo. Puerto Rico.

37. **Wang, C., Yang, F., Wang, H., Guo, P., Hou, J. (2019).** Improving real time performance of Linux System using RT-Linux. Journal of Physics: Conference Series, Vol. 1237, pp. 052017. DOI: 10.1088/1742-6596/1237/5/052017.

38. **Wilf, H. S. (2002).** Algorithms and complexity. DOI: 10.1201/9780429294921/.

Article received on 01/12/2022; accepted on 20/03/2023.
Corresponding author is Pedro Guevara López.