行政院國家科學委員會補助專題研究計畫 ■ 成果報告
□期中進度報告

即時系統之可排程分析

計畫類別：■ 個別型計畫　□ 整合型計畫
計畫編號： NSC 91-2213-E-002 -070
執行期間： 91 年 08 月 01 日至 92 年 07 月 31 日

計畫主持人：郭大維教授
共同主持人：
計畫參與人員： 郭錦福、彭念劬、謝仁偉、魏仲佑、修丕承、劉毓華

成果報告類型(依經費核定清單規定繳交)：■精簡報告 □完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：

※ 中　華　民　國　92　年　8　月　20　日

行政院國家科學委員會補助專題研究計畫成果報告

計畫名稱：即時系統之可排程分析

計畫編號：NSC 91-2213-E-002 -070

執行期限：計畫自民國 91 年 08 月 01 日起至民國 92 年 07 月 31 日止

主持人：郭大維　　國立台灣大學資訊工程所

計畫參與人員：郭錦福、彭念劬、謝仁偉、魏仲佑、修丕承、劉毓華

國立台灣大學資訊工程所

一、摘要

【中文摘要】

本專題研究計畫著重在兩個部分：其一是探討週期性處理程序之間的週期關係（period pattern of periodic processes），藉以縮短 RMA 可排程性測試的執行時間 – 藉由減少處理程序的數目以及 RMA 測試的排程測試點（schedulability points）來改善可排程性測試。我們推導出一些輔助定理來幫助我們選定 RMA 最差結束時間測試（worst-case completion-time test）的初始值。我們完成了一系列的實驗，證明所提出的方法確實使整體效能獲得改善。另一個研究重點則是在 framebuffer 裝置的資源管理。我們提出一個藉由引入應用程式函式庫，即可在 framebuffer 裝置上進行資源管理的方法。不需對那些使用到 framebuffer 裝置的程式碼進行修改，我們建立一個新的 "虛擬" 裝置來維護那些負責 framebuffer 資源管理及存取控制的內部資料結構。我們在 Linux 上建立了一個系統原型來展示所提機制與方法的可行性。

【英文摘要】

The purpose of this work focus on two parts: One is to explore the period pattern of periodic processes to improve the runtime of the RMA schedulability test – The schedulability test is improved by reducing the number of processes and the number of testing points in RMA tests. We derive lemmas to help in selecting a proper initial value for the RMA worst-case completion-time test. A series of experiments is done to demonstrate the performance improvement of the proposed approach. The other part focuses on resource reservation and enforcement on framebuffer device. We propose an approach to reserve the usages of framebuffer devices through the inclusion of codes in application libraries. Without any modification of the existing source code to framebuffer devices, we create a new "virtual" device which maintains internal data structures for framebuffer resource management and access control. The feasibility of the proposed mechanism and methods is demonstrated with a system prototype over Linux.

【關鍵詞】

schedulability test, rate monotonic analysis, division graph, reduced set, QoS, resource reservation, real-time operating systems, framebuffer devices

二、前言

The real-time resource allocation problem has been an active research topic in recent decades. Recently, real-time scheduling problems have been analyzed from different architectural assumptions. While so many researchers keep exploring "more precise" polynomial-time schedulability tests, the needs of precise schedulability tests is growing. One of the purposes of this research is to explore the period pattern of periodic processes to improve runtime of the RMA schedulability tests, which can provide a sufficient and necessary condition to justify the scheduleability of a process set. We propose a number of theorems and show that the schedulability test could be improved by reducing the number of processes and the number of schedule-

ability points in RMA tests. We also derive theorems to help in selecting a proper initial value for the RMA worst-case completion-time test. Our simulation results show that the validation time for a RMA schedulability test can be much improved by exploring the period pattern of a process set.

Researchers in the area of real-time operating systems started exploring resource reservation and Quality-of-Service techniques over operating systems in the last decade. Researchers proposed their work over various operating systems, such as Windows, Mach, UNIX, Linux or other operating systems. The other purpose of this work is to propose a real-time resource reservation methodology and its implementation methods for the resource reservation and access control of framebuffer devices, where framebuffers usually refer to devices for screen displaying (and sometimes to the RAM buffers on display cards). A Quality-of-Service (QoS) reservation and enforcement mechanism is proposed to guarantee a proper on-line framebuffer usage for each individual application. Since a common way for a process to access a framebuffer device is to map the corresponding framebuffer to the user memory space of the process for direct access, a preload-library approach is presented to resolve the implementation issues. We propose not to modify the operating system or even any library functions in libc/glibc for the portability of the implementation. The implementation issues were also explored over various layers of the operating systems, and the overheads of the implementation were measured. The feasibility of the approach is demonstrated by the implementation of a system prototype over Linux.

三、RS-RMA 可排程測試

3.1 基本概念

The purpose of this section is to present a heuristic algorithm based on the RMA theorems. The main issue of this section is as follows: Given a simplified set of processes with periods no more than that of τ_i, we should determine whether τ_i is schedulable. If the process with the largest period in the simplified set is unschedulable, the simplified set is transformed into a new simplified set by splitting the offspring set of a selected RS-representative. Note that if such a test fails again, the split simplified set is split again in the same way by selecting a RS-representative for splitting. The algorithm repeats until no further splitting of any simplified set is possible. If no RMA test succeeds during the splitting, then τ_i is unschedulable. Note that a simplified set of T_i denotes an abstraction of its workload. The less number of processes in a simplified set, the more abstraction the simplified set is. When a simplified set is T_i, the simplified set is no longer an abstraction. In the worst case, Algorithm RS-RMA-test will need more time than an ordinary RMA schedulability-test algorithm. However, we shall show in the performance experiments that the validation time for a RMA schedulability test can be much improved by exploring the period pattern of a process set.

Theorem 6 Algorithm RS-RMA-test will terminate and determine the schedulability of τ_i correctly.

3.2 加速 RMA 執行的方法

3.2.1 最差結束時間測試的初始值設定

The purpose of this section is to present some interesting properties of the worst-case completion test such that engineers could use to further accelerate RMA tests based on their observations over systems.

Lemma 2 Let $W_i = W_i(m+1) = W_i(m) \neq W_i(m-1)$ be the worst-case completion time of process τ_i for some integer m when $W_i(0) = 0$. There exists an integer n such that $W_i' = W_i(n+1) = W_i(n) = W_i$ when $W_i(0) = w$ for

any $0 < w < W_i$.

Although Lemma 2 does not say anything when process τ_i is unschedulable, it is trivial to show that any initial value $W_i(0)$ (which is no more than the deadline of τ_i), will also end up growing over the deadline of τ_i during the worst-case completion-time test. In other words, what a user should do for the worst-case completion-time test is to have a good initial value $W_i(0)$ which is no larger than W_i.

Lemma 3 Let p be a schedulable schedule-ability point for process τ_i. When $W_i(0) = p$, the worst case-completion-time test will converge to a number no larger than p.

3.2.2 可排程測試點的選擇

The purpose of this section is to derive a proper candidate set of schedulability points so that a better initial value could be set for the worst-case completion-time test. We shall derive rules to remove schedulability points out of the potential candidate set of schedule-ability points based on the relationships of process periods and computation times. Note that users could also directly verify whether these schedulability points are schedulable.

Lemma 7 Let

$T = \{\tau_1 = (p_1, c_1), \tau_2 = (p_2, c_2), ..., \tau_n = (p_n, c_n)\}$

be a set of n processes. If $p_1 < p_2 < \ldots < p_n$, then the unschedulability of schedulability point $\lfloor \frac{p_2}{p_1} \rfloor p_1$ implies the unschedulability of the following schedulability points: $p_1, 2p_1, \ldots,$ and $\left(\lfloor \frac{p_2}{p_1} \rfloor - 1\right)p_1$.

3.2.3 虛擬 Root

The idea of RS-representative merges a certain specific processes to improve the runtime of the RMA schedulability test. However, if the difference in period between any two RS-representatives is large, numerous testing points might slow down RMA tests. To ease this circumstance, we introduce pseudo roots to further decrease the number of testing points. The idea of the pseudo roots works as follows: After a reduced set been split, we enlarge period (and its computation requirement as well) of some newly generated RS-representatives by a reasonable constant M. Note that the period of enlarged RS-representative is no larger than the period of its parent.

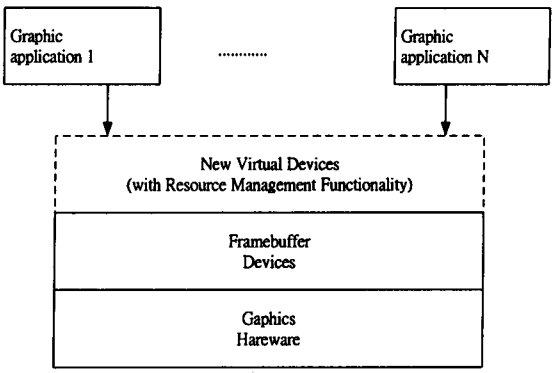
3.2.4 RS-RMA 可排程測試的精確性

Although RS-RMA-test tries to reduce the number of testing processes in evaluation progress, it introduces an inaccuracy factor as well. This inaccuracy comes from the over-estimate of computation requirements of those processes in reduced sets. It means the computation requirements of some jobs should not be taken into consideration since they do not arrive at that time! We modified our RS-RMA algorithm to eliminate this problem.

四、Framebuffer 裝置上的管理與存取控制

4.1 相關演算法

In this section, we shall propose a simple approach for resource management on framebuffer devices, as shown in Figure.



4.1.1 資源保留與許可控制

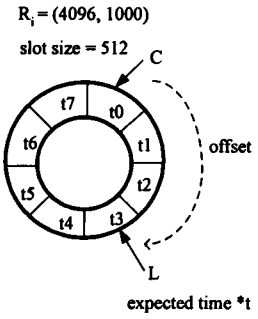
We are interested in processes which require a guarantee on the number of bytes which they can write to framebuffer devices within some specified time intervals. In order to prevent processes from overloading the system and interfering with each another, we assume that each process A_i could request a right to write B_i bytes of data to a specified framebuffer device within every W_i units of time. Once the

reservation is granted, the system should guarantee the reservation and, at the same time, prevent the process from using any of its reserved framebuffer devices for more than its reserved budget. The on-line admission control of resource reservations of processes could be done as follows: Given a collection of admitted reservations T = {(B_1, W_1), (B_2, W_2), ..., (B_n, W_n)} on a framebuffer device, where (B_i, W_i) means that B_i bytes might need to be written within every W_i units of time. Suppose that a new reservation (B_{n+1}, W_{n+1}) is made on the device, as long as the following formula is satisfied, the new reservation is granted; otherwise, it is rejected: $\sum \frac{C(B_i)}{W_i} \leq 1$

Let $C(B_i)$ be the time needed to write B_i bytes to a framebuffer device.

4.1.2 以環狀結構來管理存取控制

The purpose of this section is to propose a data structure and an algorithm to track the on-line usage of a framebuffer device with a linear time complexity O(n), where n is the maximum amount of data of a request.



Given a collection of admitted reservations T = {(B_1, W_1), (B_2, W_2), ..., (B_n, W_n)} on a framebuffer device, where (B_i, W_i) means that B_i bytes might need to be written by the corresponding process within every W_i units of time. For each reservation R_i = (B_i, W_i), a ring is allocated to track the on-line usage of a framebuffer device of that process, as shown in figure above. The ring size is the claimed budget divides by the chosen slot size, i.e. B_i/slot size.

五、結果與討論

This research explores the period pattern of periodic processes to improve the runtime of the RMA schedulability test. We propose the idea of reduced sets to analyze the schedulability of a process set in terms of transformed process sets which have a much less number of processes. We propose a number of theorems and show that the schedulability test could be improved by reducing the number of processes and the number of schedulability points in RMA tests. We also derive theorems to help in selecting a proper initial value for the RMA worst-case completion-time test. The performance of proposed methodology is verified by a series of simulation experiments, for which we have some encouraging results. The validation time for a RMA schedulability test can be much improved by exploring the period pattern of a process set. We must point out that the results of this paper are orthogonal to past research, e.g., [4, 6, 14], which did not utilize the knowledge of the period pattern.

With the advance of software and hardware technologies, there is an increasing demand to study the real-time resource allocation problems for different architectural assumptions and computation models. For future research, we shall extend the reduced-set methodology to analyze the schedulability of soft and firm real-time process sets, and even a process set mixed with hard, soft, and firm real-time processes. We will also exploit the idea behind the simplified sets in the reduced-set-based RMA schedulability test to further improve the performance of the RMA schedulability test.

六、參考文獻

[1] E. Bini and G.C. Buttazzo, "The Space of Rate Monotonic Schedulability," *IEEE 23rd Real-Time System Symposium*, December 2002, pp.169-178.

[2] C.-C. Han and H.-Y. Tyan, "A Better Polynomial-Time Schedulability Test for Real-Time Fixed Priority Scheduling Algorithms," *IEEE 18th Real-Time Systems Symposium*, December 1997, pp. 36-44.

[3] C.-C. Han, "A Better Polynomial-Time Schedulability Test for Multiframe Tasks," *IEEE 19th Real-Time Systems Symposium*, December 1998, pp. 104-113.

[4] D.-I. Kang, R. Gerber, and M. Saksena, "Performance-Based Design of Distributed Real-Time Systems," *IEEE 1997 Real-Time Technology and Applications Symposium*, June 1997, pp. 2-13.

[5] T.-W. Kuo and A.K. Mok, "Load Adjustment in Adaptive Real-Time Systems," *IEEE 12th Real-Time Systems Symposium*, December 1991, IEEE Transaction on Computers, Vol. 16, Number 12, December 1997.

[6] T.-W. Kuo, Y.-H. Liu, and K.J. Lin, "Efficient On-Line Schedulability Tests for Priority Driven Real-Time Systems," *IEEE Real-Time Technology and Applications Symposium*, May 31 – June 2, 2000, pp. 4-13.

[7] G. Koren and D. Shasha, "Skip-Over: Algorithms and Complexity for Overloaded Systems that Allow Skips," *IEEE 16th Real-Time Systems Symposium*, December 1995, pp. 110-117.

[8] J.P. Lehoczky, L. Sha, and Y. Ding, "The Rate Monotonic Scheduling Algorithms: - Exact Characterization and Average Behavior," *IEEE 10th Real-Time Systems Symposium*, December 1989.

[9] C.L. Liu and J.W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment," *JACM*, Vol. 20, No. 1, January 1973, pp. 46-61.

[10] J. W.S. Liu, K.J. Lin, and S. Natarajan, "Scheduling Real-Time Periodic Jobs Using Imprecise Results," *IEEE 8th Real-Time Systems Symposium*, December 1987.

[11] M.D. Natale and J.A. Stankovic, "Dynamic End-to-End Guarantees in Distributed Real-Time Systems," *IEEE 15th Real-Time Systems Symposium*, December 1994, pp. 216-227.

[12] W.-K. Shih and J. W.S. Liu, "On-Line Scheduling of Imprecise Computations to Minimize Error," *IEEE 13th Real-Time Systems Symposium*, December 1992, pp. 280-289.

[13] B. Adelberg, H.Garcia-Molina, and B.Kao, "Emulating Soft Real-Time Scheduling Using Traditional Operating Systems Schedulers," *IEEE 15th Real-Time Systems Symposium*, December 1994, pp.292-298.

[14] S. Childs and D. Ingram, "The Linux-SRT Integrated Multimedia Operating Systems: Bring QoS to the Desktop," *IEEE 2001 Real-Time Technology and Applications Symposium*, Taipei, Taiwan, ROC, pp. 135-140.

[15] Z. Deng and J. W.-S. Liu, "Scheduling Real-Time Applications in an Open Environment," *IEEE 18th Real-Time Systems Symposium*, December 1997.

[16] Mei-Ling Hsu, Wang-Ru Yang, Yuan-Ting Kao, Giun-Haur Huang, and Tei-Wei Kuo, "Providing Real-Time Access Control to Remote Resources," *The Third Workshop on Real-Time and Media Systems (RAMS'97)*, 1997, Taipei, Taiwan, ROC, pp. 137-143

[17] Giun-Haur Huang, Shie-Kai Ni, and Tei-Wei Kuo, 1996, "The Design and Implementation of the CPU Power Regulator for Multimedia Operating Systems," *IEEE 17th Real-Time Systems Symposium (RTSS'96), Work-In-Progress Session Proceeding*, Washington D.C., USA, pp. 27-30.

[18] Tei-Wei Kuo and Ching-Hui Li, 1999,"A Fixed-Priority-Driven Open Environment for Real-Time Applications," *the IEEE 20th Real-Time Systems Symposium*, Phoenix, USA, December, 1999.

[19] Clifford W. Mercer, S. Savage, and H. Tokuda, "Processor Capacity Reserves for Multimedia Operating Systems," *Technical Report CMU-CS-93-157*, School of Computer Science, Carnegie Mellon University, May 1993.

[20] Clifford W. Mercer, S. Savage, and H. Tokuda, "Processor Capacity Reserves for Multimedia Applications," *In Proceedings of the IEEE International Conference on Multi-media Computing and Systems (ICMCS)*, May 1994, pp. 90-99.

[21] Clifford W. Mercer and Ragunathan Rajkumar, "An Interactive Interface and RT-Mach Support for Monitoring and Controlling Resource Management," *IEEE Real-Time Technology and Applications Symposium*, May 1995.

[22] Y.-C. Wang and K.J. Lin, "Implementing a General Purpose Real-Time Scheduling Framework in the RED-Linux Real-Time Kernel," *IEEE Real-Time Systems Symposium*, Arizona, USA, 1999, pp. 246-255.