УДК 004.4
DOI: 10.17586/0021-3454-2016-59-11-964-970

ДОСТУП К РАЗДЕЛЯЕМЫМ РЕСУРСАМ В СИСТЕМАХ РЕАЛЬНОГО ВРЕМЕНИ С ПЕРЕМЕННЫМИ ПРИОРИТЕТАМИ ЗАДАЧ

В. В. НИКИФОРОВ[1], А. А. ТЮГАШЕВ[2]

[1] *Санкт-Петербургский институт информатики и автоматизации РАН, 199178, Санкт-Петербург, Россия*
E-mail: tau797@mail.ru
[2] *Университет ИТМО, 197101, Санкт-Петербург, Россия*

Рассматриваются вопросы обеспечения своевременности выполнения задач реального времени, разделяющих информационные ресурсы в однопроцессорных и многопроцессорных системах. Приведен критерий сравнительной эффективности применения различных сочетаний дисциплин планирования и протоколов доступа к ресурсам. Представлены результаты экспериментов, демонстрирующих преимущества использования дисциплин планирования с переменными приоритетами задач.

Ключевые слова: *системы реального времени, дисциплины планирования, протоколы доступа к ресурсам, выполнимость программных приложений реального времени*

Введение. Программное приложение для систем реального времени (СРВ) строится в виде комплекса кооперативных задач τ_1, τ_2, ..., τ_n. Каждая из задач τ_i является последовательной программой, замкнутой по передачам управления. Задача τ_i активизируется в моменты времени $t_{\text{arr}}\left(\tau_i^{(j)}\right)$, когда *j*-й раз возникает необходимость исполнения функций, возлагаемых на τ_i. Очередная *j*-я активизация задачи τ_i сопровождается порождением задания $\tau_i^{(j)}$ — динамического объекта, обеспечивающего исполнение кода задачи τ_i в сложившемся к моменту $t_{\text{arr}}\left(\tau_i^{(j)}\right)$ системном контексте.

Порождение задания $\tau_i^{(j)}$ означает увеличение числа претендентов на использование разделяемых (доступных различным задачам) системных ресурсов:

— исполнительных (процессоры, отдельные ядра процессоров);

— информационных (глобальные данные, коммуникационные ресурсы, периферийные устройства компьютерной системы).

Любое изменение условий доступа к системным ресурсам является *системным событием*. К разряду системных событий относится, например, событие, состоящее в порождении нового задания, поскольку увеличивается число претендентов на использование процессорного времени. Другим примером системных событий является завершение в момент $t_{\text{end}}\left(\tau_i^{(j)}\right)$ исполнения задания $\tau_i^{(j)}$, поскольку при этом уменьшается число претендентов на ресурс процессора.

Ключевые характеристики задачи τ_i — период T_i ее активизации и предельная продолжительность D_i ее исполнения, т.е. допустимая длина интервала существования любого из заданий типа τ_i (длина интервала времени $\left[t_{\text{arr}}\left(\tau_i^{(j)}\right), t_{\text{end}}\left(\tau_i^{(j)}\right)\right]$). Задача τ_i считается вы-

полнимой, если при любых сценариях системных событий длина интервала существования каждого из заданий типа τ_i не превосходит D_i.

Под проверкой *выполнимости* программного приложения понимается проверка гарантированной своевременности выполнения каждой из задач, входящих в его состав. Первые исследования, ориентированные на разработку методов проверки выполнимости для однопроцессорных и многопроцессорных СРВ, проводились уже более 40 лет назад [1, 2]. В последующем эти исследования интенсивно развивалась как в области однопроцессорных СРВ с классическими одноядерными процессорами [3—4], так и для СРВ с несколькими исполнительными ресурсами — многопроцессорных СРВ, СРВ на многоядерных процессорах [5, 6]. Эти исследования продолжаются и в настоящее время. В частности, совершенствуются методы оценки выполнимости приложений, исполняемых на многоядерных процессорах [7, 8].

Кооперативные задачи τ_1, τ_2, ..., τ_n разделяют между собой исполнительные и информационные ресурсы. Порядок предоставления исполнительных ресурсов определяется принятой в системе дисциплиной планирования. Далее рассматриваются две дисциплины планирования: RM (Rate Monotonic) — дисциплина со статическими приоритетами задач (приоритеты задач снижаются с увеличением значений T_i) и EDF (Earliest Deadline First) — дисциплина с переменными приоритетами задач ($\tau_x^{(i)}$ является более приоритетным относительно $\tau_y^{(j)}$, если $t_{\text{end}}\left(\tau_x^{(i)}\right) < t_{\text{end}}\left(\tau_y^{(j)}\right)$) [9].

Порядок доступа к информационным ресурсам определяется принятым протоколом доступа. В случае применения примитивного протокола PP для допуска задания к использованию исполнительного ресурса достаточно, чтобы ресурс не был занят другим заданием (пригоден при использовании дисциплин планирования как со статическими, так и с переменными приоритетами задач). Более сложный протокол пороговых приоритетов (PCP — Priority Ceiling Protocol) защищает систему от возникновения взаимного блокирования задач (пригоден только при использовании дисциплин планирования со статическими приоритетами задач) [10]. Выполненные авторами настоящей статьи эксперименты по имитационному моделированию работы СРВ показывают, что применение протокола PCP в системах на многоядерных процессорах может привести к снижению эффективности использования процессора на десятки процентов. Подходы, позволяющие избежать такого снижения эффективности, предлагаются в данной статье.

Сегменты кода и параметры задач. В рассматриваемых ниже моделях программных приложений код каждой задачи представляется в виде последовательности сегментов. Каждый сегмент содержит фрагмент программного кода, замыкаемый оператором, который связан с системным событием. Используются три типа сегментов, отличающиеся разновидностью замыкающего сегмент системного события:

— финальный сегмент — его замыкающим оператором является оператор завершения исполняемого задания;

— сегмент с запросом ресурса — его замыкающим оператором является оператор запроса доступа задания к одному из системных информационных ресурсов;

— сегмент с освобождением ресурса — его замыкающим оператором является оператор завершения одного из занятых заданием исполнительных ресурсов (оператор освобождения ресурса).

Задачи, не содержащие операторов доступа к ресурсу, состоят из единственного (финального) сегмента.

Текстовое представление сегмента начинается со спецификатора типа сегмента и завершается указателем веса сегмента. Спецификаторы сегментов имеют следующий вид:

`"E_"` — спецификатор финального сегмента;

`"L_"` — спецификатор сегмента типа *lock* (запрос ресурса);

`"U_"` — спецификатор сегмента типа *unlock* (освобождение ресурса);

Вес сегмента представляется целым числом, отражающим объем вычислений, исполняемых в рамках сегмента. Условимся задавать объем вычислений в виде эквивалентного количества эталонных операций (эталоном такого рода может выступать, например, арифметическая операция с плавающей запятой).

Представление финального сегмента содержит два элемента: спецификатор и указатель веса — например, последовательность символов `"E_20"` представляет финальный сегмент с весом w=20 эталонных операций.

В представлении *lock*-сегмента между спецификатором и указателем веса помещается целое число, соответствующее номеру запрашиваемого ресурса — например, символы `"L_1_10"` представляют сегмент с весом w = 10, завершающийся оператором запроса доступа к ресурсу с номером 1. Аналогично символы `"U_1_40"` представляют *unlock*-сегмент с весом w = 40, завершающийся оператором освобождения ресурса с номером 1.

Для периодической задачи τ_i значение разности $T_i = t_{\text{arr}}\left(\tau_i^{(j+1)}\right) - t_{\text{arr}}\left(\tau_i^{(j)}\right)$ между моментами порождения двух ее соседних экземпляров строго постоянно (не зависит от j). Апериодические задачи отвечают менее строгому ограничению: $T_i \geq t_{\text{arr}}\left(\tau_i^{(j+1)}\right) - t_{\text{arr}}\left(\tau_i^{(j)}\right)$. Для представления значений периодов задач в формальной модели приложения далее используются конструкции типа `"T=1000"`, означающего, что период задачи равен 1000 единиц физического времени (например, микросекунд). Значения D_i представляются конструкциями типа `"D=1000"`.

Текстовое представление модели задачи имеет вид строки, начинающейся описаниями ее параметров, за которыми следует описание последовательности сегментов ее кода. Описания отдельных параметров и сегментов разделяются пробелами. Например, для периодической задачи τ_i, представляемой строкой

```
T=1000 D=1000 L_1_10 U_1_40 E_20 ,
```

ее экземпляры $\tau_i^{(j)}$ порождаются каждую миллисекунду, допустимая продолжительность существования каждого из заданий типа τ_i не превышает 1 мс, общий вес W_i кода задачи τ_i составляет 70 эталонных операций.

Выраженный в единицах физического времени объем C_i ресурса процессора, требуемый для исполнения одного задания типа τ_i, зависит от производительности P процессора, выраженной числом эталонных операций, выполняемых процессором за единицу физического времени (например, микросекунду): $C_i = W_i / P$. Величина $u_i = C_i / T_i$ характеризует нагрузку — долю процессорного времени, расходуемого задачей τ_i. Сумма $U = \sum_{1 \leq i \leq n} u_i$ составляет общую нагрузку на процессор от многозадачного приложения. Описание

```
T=1000 D=1000 E_150
T=1150 D=1150 E_168
T=1330 D=1330 E_198
T=1520 D=1520 E_218
T=1750 D=1750 E_260
```
(1)

представляет модель приложения из пяти независимых задач τ_1, τ_2, τ_3, τ_4, τ_5 (задач без сегментов типа lock/unlock). Следующий пример представляет приложение с пятью задачами, разделяющими четыре ресурса g_1, g_2, g_3, g_4:

```
T=1000 D=1000 L_1_0001 U_1_0147 E_0002
T=1150 D=1150 L_2_0001 L_3_0004 U_2_0002 U_3_0002 E_0159
T=1330 D=1330 L_3_0001 L_4_0004 U_3_0002 U_4_0002 E_0189          (2)
T=1520 D=1520 L_2_0001 L_4_0004 U_2_0002 U_4_0002 E_0219
T=1750 D=1750 L_1_0001 U_1_0257 E_0002
```

Результаты моделирования исполнения задач. Выполнимость приложения с конкретной комбинацией дисциплины планирования и протокола доступа может быть обеспечена путем увеличения производительности P используемого процессора. Тогда роль критерия эффективности рассматриваемой комбинации может играть минимальная производительность $P_{\min}$ процессора, обеспечивающая выполнимость приложения. Значению $P_{\min}$ соответствует максимальная суммарная нагрузка $U_{\max}$ приложения. Использование $U_{\max}$ в качестве критерия эффективности более удобно, поскольку является безразмерной величиной. Величина $U_{\max}$ отражает максимально достижимую полезную нагрузку на процессор при использовании для данного приложения конкретной комбинации дисциплины планирования и протокола доступа. Назовем *плотностью* приложения безразмерную величину $\mathrm{Dens} = U_{\max} / m$ (m — число ядер процессора).

На рис. 1 приведены результаты измерений значения Dens для приложения со структурой (1) при различных значениях жесткости $H_i = T_i / D_i$ предельных сроков задач при их исполнении на одноядерном процессоре (*а*) и двуядерном процессоре (*б*).

Наличие существенного зазора между графиками плотности для EDF и RM дисциплин планирования (заштрихованная область) является следствием специфического соотношения параметров (значений периодов и нагрузок), составляющих приложение задач. При других соотношениях параметров величина зазора уменьшается. Для одноядерных процессоров при $H_i = 1$ это обстоятельство было обнаружено в работе [1]. Менее предсказуемые результаты получены авторами при имитационном моделировании приложений с взаимозависимыми задачами (задачами, разделяющими информационные ресурсы).

Графики плотности для приложений со структурой (2) приведены на рис. 2: *а* — на одноядерном процессоре, *б* — на двуядерном процессоре.

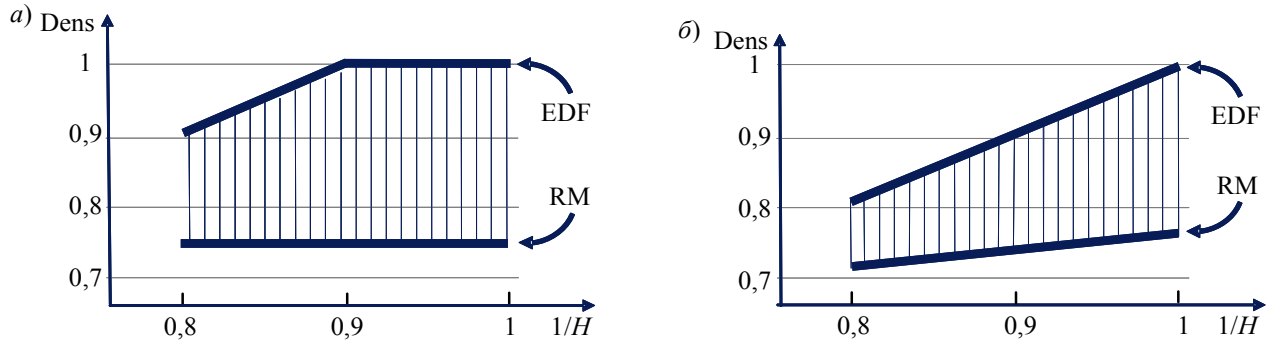

Рис. 1

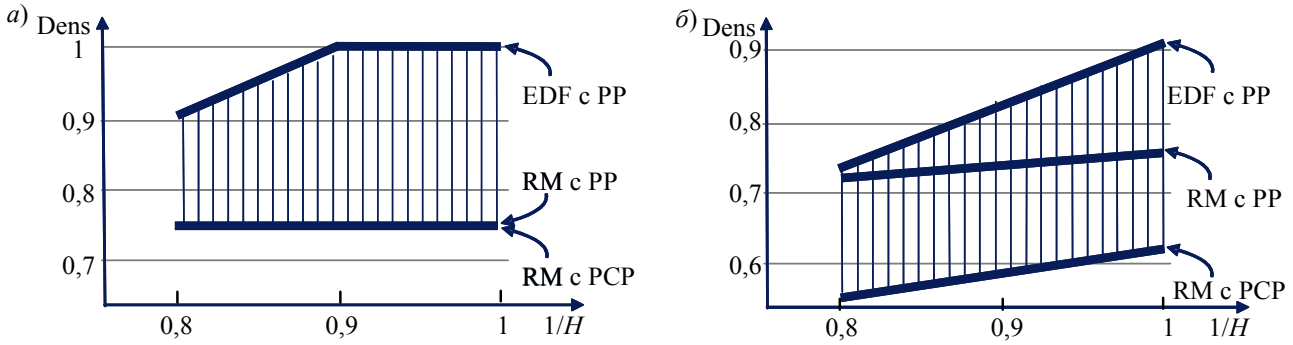

Рис. 2

Обращает на себя внимание совпадение графиков рис. 1, *а* и 2, *а*. Кроме того, значения плотности Dens, соответствующие RM дисциплине планирования с примитивным протоколом доступа к ресурсам (PP), совпадают со значениями, соответствующими RM дисциплине с протоколом пороговых приоритетов (PCP). На первый взгляд, это может показаться странным. Действительно, при наличии разделяемых ресурсов увеличивается значение времени отклика высокоприоритетных и среднеприоритетных задач, что, казалось бы, должно приводить к снижению значения Dens. Такого снижения в данном случае не происходит в силу той же специфичности соотношений параметров задач приложения (2), которая обусловливает максимальную ширину зазора между графиками плотности для EDF и RM.

Вид графиков на рис. 2, *б* существенно отличается от вида аналогичных графиков на рис. 1, *б*. Значения Dens для приложения с взаимозависимыми задачами, исполняемыми с использованием как EDF, так и RM дисциплин планирования, существенно ниже аналогичных значений для приложения с независимыми задачами.

Рис. 2, *б* отражает и существенное различие в эффективности применения протоколов PP или PCP при реализации программного приложения со структурой (2) на двуядерном процессоре. При исполнении с использованием дисциплины планирования RM и $H_i = 1$ применение PP обеспечивает полезную нагрузку 76 %, а применение PCP — лишь 62 %. Использование EDF обеспечивает полезную нагрузку до 91 %.

При создании программных приложений для систем реального времени протокол пороговых приоритетов применяется с целью гарантировать работу системы без попадания задач в состояния взаимных ожиданий. Поскольку PCP пригоден только при использовании дисциплин планирования со статическими приоритетами задач, его применение исключает использование более эффективных дисциплин планирования с переменными приоритетами. В системах на многоядерных процессорах, как было сказано выше, это может привести к потере эффективности использования процессорного времени в десятки процентов.

Для того чтобы избежать таких потерь эффективности, можно воспользоваться предложенным в работе [11] методом проверки структуры программного приложения на возможность возникновения ситуаций типа взаимного блокирования задач — методом анализа многодольного графа зависимостей связок критических интервалов. Если результат проверки отрицательный (показывает, что при любом допустимом сценарии системных событий взаимные блокирования исключаются), то нет смысла применять PCP — можно применять PP с использованием как дисциплины планирования RM, так и более эффективных дисциплин с переменными приоритетами. Если результат проверки показывает, что применение PP сопряжено с взаимным блокированием задач, то можно попытаться модифицировать структуру приложения так, чтобы эта опасность была устранена.

Другую возможность открывает предложенный в работе [12] протокол предотвращения взаимного блокирования (PPMB — Protocol for Prevention of Mutual Blocking), который допускает использование любой дисциплины планирования, в том числе дисциплин с переменными приоритетами задач (например, EDF), что позволяет обеспечить высокоэффективную реализацию приложений со сложной структурой взаимных зависимостей прикладных задач.

Заключение. Результаты имитационного моделирования исполнения программных приложений СРВ на многоядерных процессорах показывают, что применение протокола пороговых приоритетов для реализации приложений с взаимозависимыми задачами, исключающее возможность использования высокоэффективных процедур планирования с переменными приоритетами задач, приводит к снижению полезной нагрузки на десятки процентов. Вместе с тем применение протокола, исключающего возможность взаимного блокирования задач, необходимо лишь в том случае, если такая опасность взаимного блокирования действительно существует.

Наличие или отсутствие такой опасности может быть проверено методом анализа многодольного графа связок критических интервалов. Применение этого метода позволяет избежать снижения эффективности использования процессорного времени и подтверждает преимущества применения дисциплин планирования с переменными приоритетами задач.

СПИСОК ЛИТЕРАТУРЫ

1. *Liu C., Layland J.* Scheduling algorithms for multiprocessing in a hard real-time environment // J. of the ACM. 1973. Vol. 20, N 1. P 46—61.
2. *Dhall S. K., Liu C. L.* On a real-time scheduling problem // Operating Research. 1978. Vol. 26, N 1. P. 127—140.
3. *Ferrari A. D.* Real-time scheduling algorithms // Dr.Dobb's Journal. 1994. N 12. P. 60—66.
4. *Никифоров В. В., Павлов В. А.* Операционные системы для встроенных приложений // Программные продукты и системы. 1999. № 4. С 24—30.
5. *Baruah S. K.* Fairness in periodic real-time scheduling algorithms // Proc. of the 16th IEEE Real-Time Systems Symp. 1995. P. 200—209.
6. *Baker T.* Multiprocessors EDF and deadline monotonic schedulability analysis // Proc. of the 24th IEEE Real-Time Systems Symp. 2003. P. 120—129.
7. *Baruah S., Bonifaci V., Marchetti-Spaccamela A.* The global EDF scheduling of systems of conditional sporadic DAG tasks // Proc. of the 27th Euromicro Conf. on Real-Time Systems (ECRTS). 2015. P. 222—231.
8. *Sun Y., Lipari G., Guan N., Yi W.* Improving the response time analysis of global fixed-priority multiprocessor scheduling // Proc. of the 20th IEEE Intern. Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA). 2014. P. 1—9.
9. *Liu J. W. S.* Real-Time Systems. NJ: Prentice Hall, 2000. 590 p.
10. *Sha L., Rajkumar R., Lehoczky J. P.* Priority inheritance protocols: An approach to real-time synchronization // IEEE Transact. on Computers. 1990. Vol. 39(9). P. 1175—1185.
11. *Никифоров В. В., Павлов В. А.* Структурные модели для анализа многозадачных программных систем // Информационно-измерительные и управляющие системы. 2011. № 9. С. 19—29.
12. *Никифоров В. В.* Протокол предотвращения взаимного блокирования задач в системах реального времени // Изв. вузов. Приборостроение. 2014. Т. 57, № 12. С. 21—27.

Сведения об авторах

Виктор Викентьевич Никифоров — д-р техн. наук, профессор; СПИИРАН, лаборатория информационно-вычислительных систем и технологий программирования; E-mail: nik@iias.spb.su

Андрей Александрович Тюгашев — д-р техн. наук; Университет ИТМО, кафедра компьютерных технологий в образовании; E-mail: tau797@mail.ru

Рекомендована СПИИРАН

Поступила в редакцию
01.06.16 г.

Ссылка для цитирования: *Никифоров В. В., Тюгашев А. А.* Доступ к разделяемым ресурсам в системах реального времени с переменными приоритетами задач // Изв. вузов. Приборостроение. 2016. Т. 59, № 11. С. 964—970.

ACCESS TO SHARED RESOURCES IN REAL-TIME SYSTEMS WITH VARYING TASKS PRIORITIES

V. V. Nikiforov [1], A. A. Tyugashev [2]

[1] *St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences, 199178, St. Petersburg, Russia*
E-mail: tau797@mail.ru

[2] *ITMO University, 197101, St. Petersburg, Russia*

The problems of ensuring timeliness of real-time tasks, sharing information resources in uniprocessor and multiprocessor systems are considered. A criterion of comparative efficiency of application of various combinations of scheduling disciplines and protocols of access to resources is formulated. Results of experiments demonstrating the benefits of the use of planning disciplines with varying task priorities are presented.

Keywords: real-time systems, scheduling modes, protocols for access to resources, feasibility of applications for real-time systems

Data on authors

Viktor V. Nikiforov — Dr. Sci., Professor; SPIIRAS, Laboratory of Informational-Computing Systems and Software Technologies; E-mail: nik@iias.spb.su

Andrey A. Tyugashev — Dr. Sci.; ITMO University, Department of Computer Educational Technologies; E-mail: tau797@mail.ru

For citation: *Nikiforov V. V., Tyugashev A. A.* Access to shared resources in real-time systems with varying tasks priorities // Izv. vuzov. Priborostroenie. 2016. Vol. 59, N 11. P. 964—970 (in Russian).

DOI: 10.17586/0021-3454-2016-59-11-964-970