УДК 629.78.018

И.Б. ТУРКИН, Т.С. НИКИТИНА

Национальный аэрокосмический университет им. Н.Е. Жуковского "ХАИ", Украина

АНАЛИЗ АЛГОРИТМОВ ПЛАНИРОВАНИЯ ЗАДАЧ РЕАЛЬНОГО ВРЕМЕНИ ДЛЯ МНОГОПРОЦЕССОРНЫХ СИСТЕМ

Важным критерием для систем реального времени является выполнение задач к поставленному сроку. В данной статье приведены алгоритмы планирования задач реального времени для многопроцессорных систем и проведен их анализ. При разработке операционных систем полученные результаты могут помочь в выборе оптимального алгоритма. Также данные результаты будут полезными при разработке алгоритма планирования задач реального времени для многоядерной архитектуры процессоров, куда и будет направлено дальнейшее исследование.

системы реального времени, алгоритмы планирования, периодические задачи, многопроцессорные системы

Введение

С каждым годом системы реального времени становятся все более сложными и потребляют значительную вычислительную мощность. Многопроцессорные системы (МС), а теперь и многоядерные, предоставляют идеальную возможность повысить не только их производительность, но и их надежность.

Механизмы, обеспечивающие работу аппаратно-ориентированных приложений (взаимодействие задач друг с другом и с процессами во внешней среде), реализуются операционными системами реального времени, предоставляющими прикладным задачам необходимые сервисы. Планирование процессов в однопроцессорных системах осуществляется известными и хорошо проработанными алгоритмами, а для МС многие вопросы в теории планирования реального времени (ТПРВ) остаются открытыми для исследований [1 – 3].

Основная цель данного исследования – провести сравнительный анализ известных алгоритмов планирования задач реального времени (ЗРВ) для МС, определить их особенности, преимущества и недостатки.

1. Алгоритмы планирования для МС

Первоначально требуются алгоритмы, которые будут определять, в каком порядке и в зависимости от чего задачи будут назначаться процессорам. Планирование в МС осуществляется согласно двум схемам: глобальной (global) и раздельной (partitioning). Согласно схеме раздельного планирования задачи выполняются на одном из процессоров (статично) (рис. 1), а при втором подходе задачи могут перемещаться от одного процессора к другому (рис. 2).

В обеих схемах контролирующий механизм допуска (admission control mechanism) формирует подмножество задач, решаемых каждым процессором. Для схемы глобального и раздельного планирования применяются статические и динамические методы планирования. Важным критерием для контролирующего механизма допуска является вычислительная сложность алгоритма (computational complexity), которая должна быть минимальной.

В данной статье будет рассмотрена проблема планирования набора n периодических задач (ПЗ) реального времени $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$ на m процессорах.

Основные параметры ПЗ:

– время выполнения *Ci* (время ЦП, необходимое для завершения одного запуска задачи);

– период *Ti*;

– *крайний срок выполнения* D_i.

Каждая ПЗ τi характеризуется бесконечной последовательностью заданий (jobs).

Период T_i ПЗ τ_i является фиксированным интервалом между временем выполнения заданий.

Время выполнения C_i – максимальное время выполнения каждого задания (периодическая итерация). Период $T_i > 0$ и $0 < C_i \le T_i = D_i, (i = 1,..., n)$. $u_i = C_i / T_i$ – фактор загрузки ПЗ τ_i. Общий фактор загрузки всех ПЗ

$$U_{TOT} = \sum_{i=1}^{n} \frac{C_i}{T_i}.$$

Пусть α максимально возможная загрузка любой ПЗ в системе,

$$\alpha = \max_{i=1,...,n}(C_i / T_i).$$

Периодическая итерация задачи τ_i, которая выполнилась в момент времени t, должна быть завершена в интервал времени $[t, t + D_i]$.

Модель, которая будет использована в данной работе, накладывает следующие ограничения: время на переключение контекста задач не принимается во внимание; каждая периодическая итерация (job) может выполняться только на одном процессоре в один момент времени и должна быть завершена до наступления следующей периодической итерации.

2. Раздельная схема планирования задач в МС

В раздельной схеме планирования важно определить алгоритм планирования для каждого отдельного процессора и общий распределительный алгоритм назначения задач процессорам. Первым шагом является оценка планируемости набора задач. Процессор будет рассматриваться, как ящик (bin), вместимость, которого определяется его объемом. *Bin-packing проблема* возникает, когда необходимо разложить n объектов (задач) с массой w_k (загрузка $u_k = w_k$) в минимальное количество ящиков (процессоров), таким образом, чтобы общая масса объектов, в каждом ящике не превышала максимальный уровень вместимости

Эвристические алгоритмы распределения задач. Известны следующие эвристики, решающие проблему bin-packing [1]:

– *First-Fit (FF):* FF располагает новые объекты в не пустые ящики (bin) с наименьшим индексом, таким образом, чтобы масса новых объектов вместе с массой объектов уже расположенных объектов в корзине, не превышают ее вместимость. Если же объект превышает вместимость не пустых ящиков, то его располагают в пустой ящик.

– *Best-Fit (BF):* Если объект не может быть расположен в не пустой ящик, тогда он располагается в пустой ящик. Иначе, BF расположит объект в не пустой ящик с наименьшей доступной вместимостью. Также известны эвристики *Next-Fit (NF)*, *Worst-Fit (WF)*.

Эффективность алгоритмов для некоторого множества процессоров определено следующим образом.

Пусть N_{opt} – количество процессоров, используемых оптимальным алгоритмом, и пусть *N(A)* – количество процессоров, используемых исследуемым алгоритмом. Гарантируемая эффективность эвристического алгоритма *A* определена как

$$\Re_A = \lim_{N_{opt}\to\infty} \frac{Nopt}{N(A)}.$$

Необходимо отметить, что чем значение меньше (оптимальный показатель эффективности 1), тем решение алгоритма А близко к оптимальному значению.

При раздельной схеме планирования назначение задач процессорам зависит не только от применяемого алгоритма, но и от используемых условий планируемости набора ПЗ для МС.

Проверка условия планируемости набора задач для МС. При планировании задач распределенных по процессорам, в качестве алгоритма планирования может использоваться любой известный алгоритм планирования, применяемый для однопроцессорных систем. Для МС существуют различные модификации алгоритма RM, рассмотрим некоторые из них.

Известны следующие условия планируемости для алгоритма Rate Monotonic:

– *условие L&L (Liu&Layland)[2], с*огласно которому общая загрузка задач τ сравнивается с предельным коэффициентом загрузки, который зависит от количества задач в системе. Ни одна из задач не пропустит крайние сроки выполнения, если будут соблюдены следующие условия:

$$\sum_{i=1}^{n} \frac{C_i}{T_i} \quad \le \quad n(2^{1/n} - 1);$$

если $n \to \infty$, *тогда* $n(2^{1/n}-1) \to \ln 2$.

– *Условие IP (Increasing Period)*[3].

Необходимо, чтобы периоды задач располагались по возрастанию, согласно чему задачам назначается приоритет. Условие IP, предложено [liu leyland] и используется для представления алгоритмов RM Next Fit и RM First Fit для МС.

Теорема: Пусть $\tau = \{\tau1, \tau2, \ldots, \tau n\}$ набор ПЗ с периодами $T1 \le T2 \le \ldots \le Tn$ и пусть

$$u = \sum_{i=1}^{n} \frac{C_i}{T_i} \le (n-1)(2^{1/(n-1)} - 1).$$

Если следующее условие соблюдается

$$\frac{C_n}{T_n} \le 2(1 + \frac{u}{(n-1)})^{-(n-1)} - 1,$$

то набор задач будет корректно спланирован согласно алгоритму RM. Когда $n \to \infty$, то минимальная загрузка задачи τ_n приближается к $(2e^{-u} - 1)$.

Также известны условия планируемости *PO (Period Oriented), RBOUND, UO (Utilization Oriented), DU (Decreasing Utilization).*

В качестве примера рассмотрим алгоритм RMNF. Данный алгоритм сортирует задачи в порядке возрастания согласно их периода и использует условие планируемости IP.

На входе:

набор задач $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$, процессоры – $\{P_1, \ldots, P_m\}$, Ui – загрузка задачи τ_i.

На выходе:

j – требуемое количество процессоров.

1. Отсортировать задачи согласно периода
2. i := 1; j := 1; /* i = i-ая задача, j = j -й процессор*/
3. **while** (i ≤ n) **do**
4. **if** (ui ≤ Условие IP) **then**
5. Uj := Uj + ui ;
6. **else**
7. Uj+1 := Uj+1 + ui ;
8. j := j + 1;
9. i := i + 1;
10. **return (j);**
11. **end**

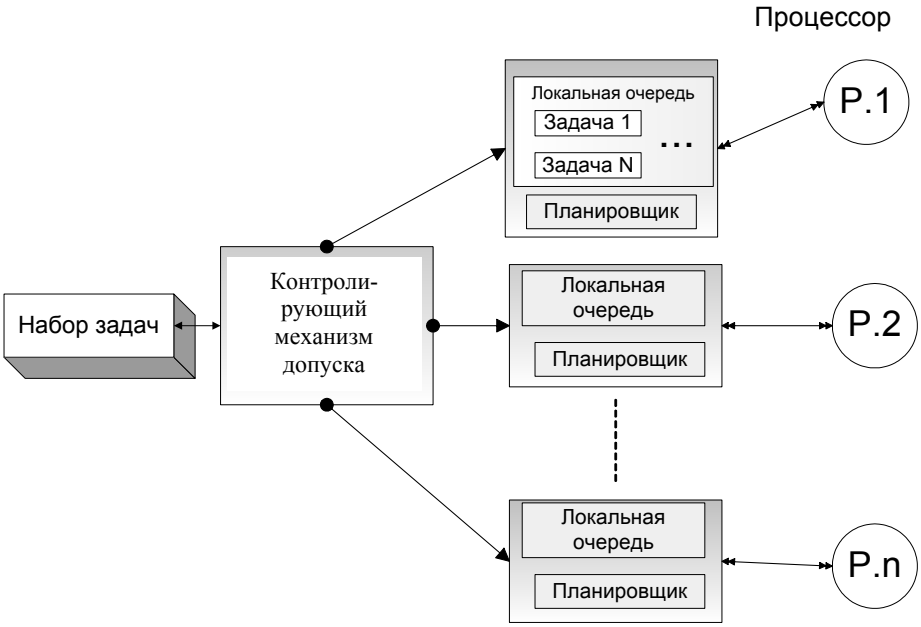

Рис. 1. Раздельная схема планирования

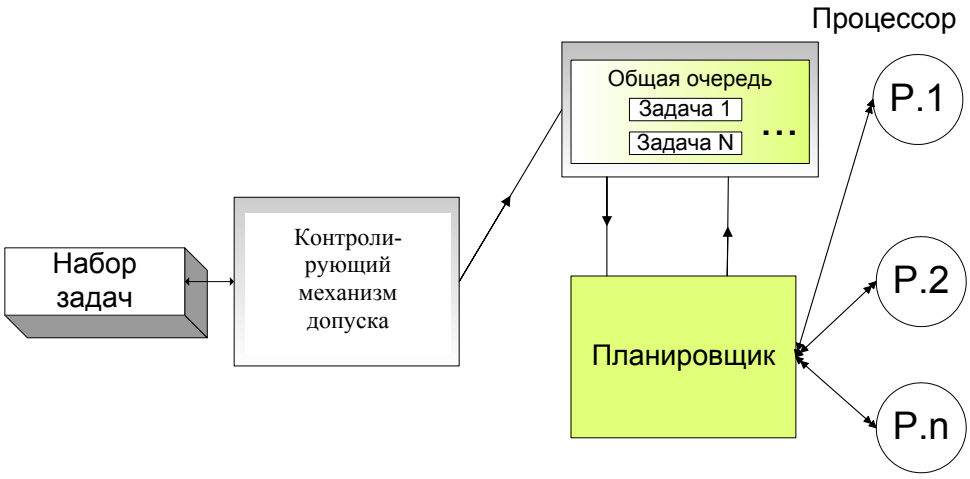

Рис. 2. Глобальная схема планирования

В данном алгоритме, задача τ_i назначается процессору P_j (шаг 5) только если, после использования условия планируемости IP, задачи являются планируемыми (шаг 4), иначе задача назначается процессору P_j+1.

После того, как задача была назначена процессору, следующий шаг увеличить i на единицу (шаг). Шаг 3 – 9 повторяется до тех пор, пока все задачи не будут распределены на процессоры.

Когда алгоритм завершится, то будет известно, сколько процессоров было задействовано (j). Вычислительная сложность данного алгоритма O(n log n) (где n – количество задач).

Как видно из табл. 1 наименьшую вычислительную сложность имеют алгоритмы RMNF-L&L, RMGT/M, а наибольшую имеет алгоритм *RBOUND-MP*.

У остальных алгоритмов вычислительная сложность – O(n log n). Алгоритмы RM-FFDU, RMST, обладают наилучшей эффективностью.

Таблица 1

Оценка сложности и эффективности алгоритма RM

	Алгоритм	**Условие**	**Сложность алгоритма**	**Эффективность алгоритма**
1	RMNF	IP	O(n log n)	0,37 [3]
2	RMFF	IP	O(n log n)	0,42 [4], 0,45 [5]
3	RMBF	IP	O(n log n)	0,42
4	RM-FFDU	UO	O(n log n)	0,6
5	RMNF-L&L	L&L	O(n)	0,34
6	RBOUND-MP	RBOUND	O(n(m+log n))	0,22
7	RM Small Task (RMST)	PO	O (n log n)	0,57
8	RM General Task M(RMGT/M)	PO	O(n)	0,42

Выводы

Хотя теория планирования для МС развивается не так давно, уже существует множество алгоритмов и каждый из авторов представляет свои результаты в различном виде и стиле, поэтому необходимо было целенаправленное обобщение известных уже результатов и проведение анализа данных алгоритмов. В данной работе были рассмотрены алгоритмы планирования периодических задач для МС на основе схемы с фиксированными приоритетами, проведен анализ схемы планирования. Для анализа алгоритмов использовались такие показатели, как эффективность и вычислительная сложность.

При разработке систем СРВ полученные результаты могут помочь в выборе оптимального алгоритма планирования.

Интересным направлением в ТПРВ является разработка алгоритмов для систем на основе многоядерной архитектуры процессоров, куда и будет направлено дальнейшее исследование.

Литература

1. Andersson B Static Priority Scheduling in Multiprocessors. PhD Thesis, Department of Comp.Eng., Chalmers University, 2003. 18 p.

2. Baruah S. Robustness Results Concerning EDF Scheduling upon Uniform Multiprocessor. Euromicro Conf. on Real-Time Systems, 2002. – P. 48-51.

3. Baruah S. Optimal Utilization Bounds for the Fixed-Priority Scheduling of Periodic Tasks Systems on Identical Multiprocessors // IEEE Transactions on Computers, 2004. – P. 781-784.

4. Baruah S., Scheduling Periodic Tasks on Uniform Multiprocessor, Euromicro Conference on Real-Time Systems, June 2000. P. 334-337.

5. Baruah S. Deadline-based Scheduling of Periodic Task Systems on Multiprocessor // Information Processing Letters. – 84 (2). November 2002. – P. 93-98.

Поступила в редакцию 1.02.2008

Рецензент: д-р техн. наук, проф. И.В. Чумаченко, Национальный аэрокосмический университет им. Н.Е. Жуковского “ХАИ”, Харьков.