УДК 681.5:519.8

Informatics and Mathematical Methods in Simulation
Vol. 4 (2014), No. 3, pp. 256-262

ОБРАБОТКА ИНФОРМАЦИОННЫХ ПОТОКОВ И СОСТАВЛЕНИЕ ДЛЯ НИХ РАСПИСАНИЙ В СИСТЕМАХ ЗАЩИТЫ ИНФОРМАЦИИ

Е. В. Иванченко, В. А. Хорошко, Ю. Е. Хохлачова

Национальный авиационный университет,
пр. Космонавта Комарова, 1, Киев, 03058, Украина; e-mail: manjelka@yandex.ru

Разработаны математические модели однородных систем обработки информации, которые позволяют анализировать и определять эффективность многомашинных (многопроцессорных) систем с числом вычислителей не больше 16. Рассмотрен алгоритм оперативного создания расписаний для многопроцессорных систем управления, которые используют эвристические средства.

Ключевые слова: математические модели, однородные системы обработки, многопроцессорные системы управления, эвристические средства.

Введение

Современный этап развития средств управления и защиты с использованием вычислительной техники характеризируется тенденцией к применению параллельной обработки информации и многопроцессорных систем, которые обладают повышенным быстродействием. Такое быстродействие было достигнуто за счет параллельной обработки задач. Опыт эксплуатации многомашинных (многопроцессорных) систем обработки информации (МСОИ), защиты показал, что большие возможности этих систем могут быть полностью раскрыты только при использовании эффективных методов и средств организации их работы, т.е. обработки информации и оптимальных расписаний информационных потоков.

Задачи управления можно подразделить [1, 2] на независимые и информационно-связанные. Последние можно представить в виде графа (обычно без циклов), вершины которого соответствуют задачам, а дуги определяют информационные связи, тем самым задавая отношение предшествования на множестве задач.

Существующие методы распределения групп задач можно разделить на два вида: допускающие прерывания и без прерываний. Методы распределения без прерываний могут быть как оптимальными, так и субоптимальными. Для получения оптимального расписания используются различные варианты дискретного программирования. Субоптимальные методы при малых затратах ресурсов и времени дают решения, близкие к оптимальным. При их использовании, как правило, осуществляется частичный перебор возможных решений в соответствие с эвристическими правилами.

С учетом того, что системы обработки, защиты и управления в настоящее время в основном имеют многомашинную (многопроцессорную) структуру, воспользуемся моделью многоканальной системы массового обслуживания. При этом в системах процессоры (машины) отожествляются с каналами обслуживания, а задачи – с заявками. Появление систем, адаптируемых к задачам, т.е. меняющих конфигурацию в зависимости от заявок, обусловило необходимость создания адекватных математических моделей систем с перестраиваемой структурой.

Основная часть

Рассмотрим модель [2] однородной системы при наличии приоритетного потока больших задач. При этом задача считается большой, если её ранг больше $N/2$, где N – число машин (процессоров) в системе. Предположим, что на однородную систему (ОС) из N машин (процессоров) поступает пуассоновский поток больших задач с интенсивностью λ. Вероятность поступления задачи с рангом n есть $\lambda_n(\sum_{n>N/2}\lambda_n = 1, \lambda_n > 0)$. Задачи обслуживаются в порядке поступления, причем закон обслуживания является произвольным и индивидуальным для каждого ранга. Если система находится в состоянии j, а i задач находятся в очереди, то вероятность $G_{ij}(t)$ перехода системы в другое состояние за время t определяется следующим образом:

$$G_{ij}(t) = \sum_{\nu>N/2} \lambda \int_0^t [1 - e^{\alpha(t-t_1)}] e^{-\alpha_{t1}} \frac{(\alpha_{t1})}{j!} dH_\nu(t_1);$$

$$G_{ij}(t) = \sum_{\nu>N/2} \lambda \int_0^t e^{\alpha_{t1}} \frac{(\alpha_{t1})^j}{j!} dH_\nu(t_1);$$

$$G_{ij}(t) = \sum_{\nu>N/2} \lambda \int_0^t e^{-\alpha_{t1}} \frac{(\alpha_{t1})^{j-i+1}}{(j-i+1)!} dH_\nu(t_1),$$

где $H_\nu(t_1), \nu > N/2$ – функция распределения времени решения задач ν-го ранга системой. С помощью производящих функций получены условия, при которых период занятости системы имеет конечную длительность – с учетом того, что под периодом занятости понимаем разность между t_k – моментом времени, когда система переходит из занятого состояния в свободное, и $t_k^{'}$ – наибольшим, не превосходящим его моментом времени перехода из свободного состояния в занятое. Помимо этого, с учетом полученных результатов определяется распределение вероятностей того, что в очереди находится некоторое число задач, ожидающих решения. Эта информация позволяет путем изменения числа машин (микропроцессоров) системы регулировать длину очереди и время обработки в необходимых пределах. Однако при обработке потоков сложных задач ОС можно предположить, что в системе из N машин с заданным объемом памяти время обслуживания программ описывается экспоненциальным законом. Предположим также, что в ОС могут одновременно существовать подсистемы всех рангов – от 1 до R, т.е. $N \geq \frac{R(R+1)}{2}$. Выделим в ОС множество из $N \in E(E = \{0,1,...,N\})$ элементарных машин и реструктуризируем его из подсистемы. Обозначим $l_{k,n}^{(t)}$ число подсистем ранга n, образованных в момент времени t посредством реструктуризации, а $l_{s,n}^{'(t)}$ – число p-программ ранга n, находящихся в очереди длиной S в момент t.

Рассмотрим, как воздействует на систему случайный процесс $\Omega(t) = \{l_{k,n}^{(t)}, l_{s,n}^{'(t)} / k \in E, n \neq \overline{1,k}, S = 0,\infty\}$ при условии, что он марковский.

Пусть $P_k^{l_1,...,l_R}$ – вероятность того, что в момент t $N_l\left(\sum_{n=1}^{R} nl_{N1}n(t)\right)$ машин занято обслуживанием p-программ и в очереди занято $N - N_1\left(\sum_{n=1}^{R} nl_{N-N1}n(t)\right)$ мест, при том, что в системе k ветвей. Требуется вычислить: P_{kN} – вероятность того, что в ОС

находится ровно k ветвей; $N_{cp} = \sum_{k=1}^{N} kP_k + N\sum_{S=1}^{\infty} P_{N+S}$ – среднее число занятых машин, $K_3(N) = N_{cp}/N$ – коэффициент занятости ОС, где $P_K \sum P_K^{l_1,\ldots,l_R} = \sum \lim_{t\to\infty} P^{l_1,\ldots,l_R}$ и суммирование ведется по всем возможным разбиениям на подсистемы в предположении, что выполняется условие нормирования $\sum_{k=0}^{\infty} P_k = 1$.

Пусть $K_{k,R}$ – число способов, с помощью которых можно разбить N машин p-программами, максимальный ранг которых равен R. Тогда, просуммировав $N_{k,R}$ и ряд $P_k^{l_1\ldots,l_r}$, получим искомое значение P_k.

Значение $N_{k,R}$ определяется [2,3] по рекуррентной формуле:

$$N_{k,R} = \sum_{j=1}^{R}\sum_{i=1}^{j} N_{k-j,i}$$

при начальных условиях: $N_{k,1} = 1;\ N_{k,k} = 1;\ N_{k,j} = 0;\ k < j.$

Вероятности $P^{l_1\ldots,l_R}$ того, что в ОС равно k ветвей, при определенном разбиении $l_n, l_n^{'}, n = 1, R$, находятся как решение системы линейных уравнений, составленной с использованием методов теории массового обслуживания. Вероятность занятости машин ОС в момент времени $t + \Delta t$ определяется как сумма вероятностей трех несовместимых событий: в момент t N машин производят операцию разбиения массива данных l на некоторые участки, и при этом за время Δt ни одна подсистема не заканчивает обслуживания вспомогательных программ; в момент t занято $N - N_1$ машин, но за время Δt поступила программа ранга $n, n = \overline{1,R}$; в момент t занято $N - N_1$ машин, но за время Δt закончилось обслуживание p-программ ранга $n, n = \overline{1,R}$.

Переходя к пределу $\Delta t \to 0$, а затем при $\Delta t \to \infty$, получаем систему линейных алгебраических уравнений относительно вероятностей $P_k^{l_1\ldots,l_R}$. Решая эту систему, определяем P_k суммированием $P_k^{l_1\ldots,l_R}$ по различным комбинациям $l_1\ldots,l_R$. С помощью вероятностей $P_k, k = 0,\infty$, определяются вероятность отказа в обслуживании, вероятность того, что все машины свободны, среднее число занятых машин, а также другие характеристики системы. Однако, при анализе систем, в которых исполнение заявок основано на обработке информации от разнотипных источников (например, процессоры, устройства ввода-вывода и т.д.), рассмотренные методы исследования ОС не подходят. Для анализа таких систем предлагается модель многоканальной системы массового обслуживания [2]. Она основана на следующих положениях:

1. Узел обработки информации представляет собой ресурсы различных типов, в системе может быть произвольное, но фиксированное число устройств каждого типа.
2. В любой момент времени обработка осуществляется одним из нескольких устройств и имеет фиксированные требования на ресурсы. Множество состояния обработки задачи определяется классом ее принадлежности и фиксированными требованиями на ресурсы.
3. Заявки поступают на систему из одного или нескольких неограниченных источников; определена средняя интенсивность поступления заявок с различными начальными состояниями обработки.
4. Возможны три формы распределения ресурсов (без разделения или мультиплексирования; с разделением; стандартное мультиплексирование). Способ

распределения ресурсов влияет как на скорость обработки, так и на степень параллельности обработки задач системой.

Пусть J – число типов ресурсов; R_i – количество ресурсов i-го типа в системе, $i=\overline{1,J}$; $\overline{j}$ – число классов заявок; M – число различных векторов требований обработки системой допустимых ресурсов системой. Состояние обработки задачи определяется парой (j,k), что означает: обработка принадлежит классу j и требует множество ресурсов, вырабатываемых $\overline{V_k}$. Для удобства обозначений синтезируем изоморфное отображение множества пар $\{(j,k)\}$ на множество $\{l\}$ простых индексов; $L=J\times K$ – число состояний обработки задач; S_l – состояние l обработки задач, $1\le l\le L$ для завершения обработки ОС проходит через последовательность состояний, а ее завершение эквивалентно переходу в заключительное состояние «О», обозначаемое S_o. Время обслуживания требуемой заявки, перешедшей в состояние S_l – это время, в течение которого заявка использует ресурсы до перехода в очередное состояние. Пусть T_l - среднее время обслуживания, требуемое задачей в состоянии $S_l, l=\overline{1,L}$. Оказавшись в состоянии S_l, заявка остается в нем до истечения времени обслуживания. Следующее состояние определяется дискретным марковским процессом, соответствующим матрице $\overline{P}$; $P=(L+1)\times(L'+1)$ – стохастическая матрица переходных вероятностей. Заявки поступают в систему из одного или нескольких неограниченных источников, интенсивность их поступления в начальном состоянии S_l есть λ_l, общая интенсивность $\lambda=\sum_{l=1}^{L}\lambda_l$. Зная λ, можно определить вероятность нахождения заявки в начальном состоянии S_l: $f_l=f_{l/2}$. Распределение начальных состояний заявок определяется вектором $\overline{F}=[f_1,f_2...,f_L]$. Используя $\overline{F}$ и матрицу $\overline{P}$, можно определить $\overline{G}=[g_1,g_2,...,g_L]$, где g_L - среднее число прерываний обработки в состоянии S_l.

В МСОИ предполагается существование постоянных распределений времени обслуживания и интенсивности поступления для каждого класса заявок. Для этого определяется $P_0(\lambda)$ – вероятность того, что обслуживается комбинация $m, m=\overline{1,M}$. Если параметры $g_l, l=\overline{1,L}$, постоянны, распределение времени обслуживания также постоянно и общая интенсивность входного потока не меняется, то анализируемая система имеет фиксированные характеристики потока заявок. Для данного алгоритма распределений заданий и фиксированных характеристик потока заявок мощность системы определяется как интенсивность входного потока.

Граница мощности $\lambda_{\max}$ для МСОИ рассчитывается как интенсивность входных потоков, при которых гарантируется насыщенное независимое от используемого алгоритма распределение заявок.

Для определения $\lambda_{\max}$ МСОИ предлагается следующий метод. Предположим, что g_l – среднее время пребывания заявок в состоянии V_l; K_l – среднее число обслуживания заявок в состоянии S_l и T_l – среднее число пребывания заявок в состоянии S_l – положительные для каждого состояния обработки заданий $\lambda_{\max}$ константы; $\lambda_{\max}$ является решением задачи линейного программирования

$$\lambda_{\max}=\max_{p(\lambda)}\sum_{m=1}^{P}D_mP_m(\lambda).$$

Здесь $P_m(\lambda) \geq 0; D_m = \left(\sum_{l=1}^{l} V_{m,l} K_{l,m} \right) \Big/ \left(\sum_{l=1}^{l} g_l T_l \right), m = \overline{1, M}; l = \overline{1, L-1}$, где $K_{l,m}$ – число заявок из комбинации m в состоянии S_l; $V_{l,m}$ – средняя скорость обработки задачи из комбинации m в состоянии S_l.

Полученная аналитическая модель позволяет определить эффективность системы обработки информации с фиксированными характеристиками входного потока. Так, как задача линейного программирования содержит L ограничений, то не более L переменных $[P_m(\lambda)]$ должно быть ненулевым. Значение $P_m(\lambda) \geq 0$ определяет временной промежуток, в течение которого система должна обрабатывать комбинацию m, чтобы достичь границы M эффективности. Однозначного решения не гарантируется. Если есть несколько экстремальных точек, то любая выпуклая их комбинация также будет экстремальной.

Зная набор $\{P_m(\lambda)\}$, на котором достигается экстремум, можно определить «нежелательные» комбинации $P_m(\lambda) = 0$ и «желательные». Если стремится к достижению максимальной мощности, то следует отдать предпочтение последним. При этом необходимо учитывать расписание для МСОИ. Алгоритмы составления расписаний в многомашинных (многопроцессорных) системах используют задачи распределения информационно-связанных вершин, представленных в виде графа B без ветвлений и циклов. Этот алгоритм был предложен Ю. С. Шварцем.

Для графа B строится матрица A, которая содержит всю информацию, необходимую для составления расписания: условия очередности и время выполнения операций каждой вершиной (процессором). Определение подмножества распределяемых вершин осуществляется по следующим эвристическим правилам:

Шаг 1. Распределение вершин производится с минимизацией временных характеристик простоев.

Шаг 2. В первую очередь выделяются вершины с большим числом предшествий.

Шаг 3. Среди вершин с одинаковым числом предшествований приоритет имеют операторы с меньшим временем выполнения задачи.

Этот алгоритм гарантирует допустимость решения, удовлетворяющего всем требованиям очередности. После каждого распределения производится переформирование матрицы в соответствии с эвристическими правилами [3]. Алгоритм Шварца используется для однородной двухпроцессорной (двухмашинной) системы в предложении, что каждая вершина может реализоваться на любом процессоре в единицу времени. Распределяемые операторы представляются в виде взвешенного орграфа без контуров, при этом ищутся операторы, не имеющие предшественников, и назначаются на определенные процессоры (машины); матрица распределения графа переформировывается с учетом того, что некоторые вершины уже назначены. Затем в матрице ведется поиск строк с нулевыми элементами и вершин, соответствующих этим строкам, причем назначение на процессоры (машины) эти строки получают сразу же или после простоя процессора (машины), чтобы время занятости процессоров (машин) было почти одинаковым, т.е. операторы пропускаются пакетом.

Более совершенным является алгоритм [4], в котором распределение вершин графа по процессорам (машинам) осуществляется с помощью ярусно-параллельной формы представления графа, т.е. разбиения множества X вершин графа $B: X = X_1 V, X_2 V, ..., X_n V$. Все вершины $x \in X_i$ находятся на расстоянии $i = 1$ от начала B. Применение ярусно-параллельной формы позволило разобрать эффективные методы получения расписаний, достаточно близких к оптимальным. Это достигается за счет того, что вместо критерия $\min\{\max T_j\}$, оптимизирующего характеристики, но

требующего большего времени обработки, определяется $\min \sum_{j=1}^{n} T_j^{\Theta}$, причем Θ выбирается из условия $\Theta > \frac{\lg r}{\left[\lg (\max t_{ij}) - tg\ a\right]}$, где $T_j = \sum_{i=1}^{m} t_{ij} x_{ij}; t_{ij}$ – время решения i-й вершины на j-й машине (процессоре), $i = \overline{1,m}$, $j = \overline{1,n}$; a – одно из ближайших решений и $\max(t_{ij})$ значений матрицы $\| t_{ij} \|$;

$$x_{ij} = \begin{cases} 1, \text{если } i - \text{й алгоритм распределен на } j - \text{ю машину,} \\ 0, \text{если } i - \text{й алгоритм не распределен на } j - \text{ю машину.} \end{cases}$$

При этом необходимо учитывать выполнение следующих условий:

$$\sum_{i=1}^{m} x_{ij} \le k, \sum_{j=1}^{n} x_{ij} = l, k = 1,2,...,m.$$

Развитием ранее рассмотренного алгоритма составления расписаний является алгоритм M независимых работ, выполняемых с помощью N идентичных машин (процессоров). Каждая работа характеризируется временем обработки t_i, штрафом за единицу неиспользованного времени Φ_i и директивным сроком $d_i = t_i$ для всех i. Потому целью алгоритма является минимизация общего штрафа

$$C = \sum_{i=1}^{m} p_i (T_i - t_i),$$

где T_i – действительное время выполнения i-й работы. Предлагаемый алгоритм развивает метод ветвей и границ Элмехреби и Пария, в котором отношения предшествования строятся на основе следующих результатов.

Лемма 1. Для $i < j$, если $t_i \le t_j$ и $P_i \ge P_j$, работа i предшествует работе $j (i < j)$ в оптимальном расписании.

Предполагаем, что работы заранее упорядочены по возрастанию $r_i = t_i / P_i$. Если $r_i = r_j$, то работе с меньшим временем присваивается меньший номер, если $P_i = P_j$ и $t_i = t_j$, то номера присваиваются произвольно.

Лемма 2. Существует оптимальное расписание, где работа 1 называется первой.

Правило развития структуры в предполагаемом алгоритме состоит в том, что назначаемая задача с минимальным номером из множества $A[P(k)]$ поступает на первую освободившуюся машину (процессор), причем конфликты разрешаются назначением задачи на машину с меньшим номером. Здесь под $A[P(k)]$ понимается множество задач, которые можно назначить на позицию $(k+1)$.

Для усиления этого алгоритма предлагается обоснование правил ветвления с помощью следующих положений [5].

Теорема. Если существует работа $j \in A[P(k)]$ такая, что $j < \lambda$ является неподходящим первым получателем информации $P(k)$, где $\lambda : T_\lambda = \min T_{md}; m_d \in D$, то $P(k) = \left\{ g^{(\lambda}, ..., g^{(k)}, g^{(i)} \right\}$ – работа, назначенная на позицию i.

Следствие. Если существует работа $j \in A[P(k)]$ такая, что $j < m_d$ для всех $m_d < D$, ветви, ведущие от работы $g(k-1)$ в $P(k-1)$, должны быть вычеркнуты. Этот

процесс называется «ловушкой». Включение в предлагаемый алгоритм «ловушки» обеспечивает следующие преимущества: экономия времени при поиске решения; возможность определения неоптимальных ветвей даже при неэффективности нижней оценки; возможность обходить подвести всех уровней основной ветви.

Выводы

Техника определения границы эффективности «желательных» и «нежелательных» комбинаций, информацию о которых полезно использовать при распределении заявок, на практике может применяться только при исследовании систем процессоров (машин) до 16, для которых число возможных комбинаций весьма невелико.

Предложенные алгоритмы позволяют решать как плановые, так и оперативные задачи. Основное достоинство разработанных алгоритмов – минимальные затраты машинного времени при составлении расписаний для многопроцессорных систем управления и их оптимизации для систем защиты управления.

Список литературы

1. Ленков, С. В. Методы и средства защиты информации в 2-х томах. / Ленков С. В., Перегудов Д. А., Хорошко В. А. – К.:Арий, 2008.
2. Евреинов, Э. В. – Однородные вычислительные системы / Евреинов Э. В., Хорошевский В. Г. – Новосибирск: Наука, 1978. – 318 с.
3. Скорик, В. Н. Мультимикропроцессорные системы / Скорик В. Н., Степанов А. Е., Хорошко В. А. – К.: Техника, 1989. – 192 с.
4. Иванченко, И. С. Оценка эффективности распараллеливания программ для информационных ресурсов / Иванченко И. С., Хорошко В. А. // Сучасний захист інформації, Спец. вип, 2013. – С. 73-81.
5. Хорошко, В. А. Методы составления расписаний для многомашинных систем управления / Хорошко В. А., Моржов С. В. // Проблемы управления и информации, № 3, 2000. – С. 126-129.

ОБРОБКА ІНФОРМАЦІЙНИХ ПОТОКІВ І СКЛАДАННЯ ДЛЯ НИХ РОЗКЛАДІВ В СИСТЕМАХ ЗАХИСТУ ІНФОРМАЦІЇ

Є. В. Іванченко, В. О. Хорошко, Ю. Є. Хохлачова

Національний авіаційний университет,
пр. Космонавта Комарова, 1, Київ, 03058, Україна; e-mail: manjelka@yandex.ru

Розроблені математичні моделі однорідних систем обробки інформації, які дозволяють аналізувати і визначати ефективність багатомашинних (багатопроцесорних) систем з числом обчислювачів не більше 16. Розглянутий алгоритм оперативного створення розкладів для багатопроцесорних систем управління, які використовують евристичні засоби .

Ключові слова: математичні моделі, однорідні системи обробки, багатопроцесорні системи управління, евристичні засоби.

DATAFLOW PROCESSING AND SCHEDULING IN DATA SECURITY SYSTEMS

Ye.V. Ivanchenko, V.O. Khoroshko, Yu.Ye. Hochlacheva

National Aircraft University,
1, Cosmonaut Komarov Ave., Kyiv, 03058, Ukraine; e-mail: manjelka@yandex.ru

Mathematical models of homogeneous data processing systems were developed to analyze and determine the efficiency of multicomputer (multiprocessor) systems incorporating not more than 16 computers. An on-the-fly scheduling algorithm was examined for multiprocessor control systems using heuristic means.

Keywords: mathematical models, homogeneous data processing systems, multiprocessor control systems, heuristic means.