ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И ОБРАБОТКА ИНФОРМАЦИИ

УДК 681.5 (519.95)

О ЗАДАЧЕ РАСПРЕДЕЛЕНИЯ ИНФОРМАЦИИ ПО УРОВНЯМ ХРАНЕНИЯ В ВЫЧИСЛИТЕЛЬНЫХ МОДУЛЯХ ТРЕНАЖЕРНОЙ СИСТЕМЫ

Е.В. Ларкин, А.Н. Привалов

На основе постулата о минимизации периода межмодульного взаимодействия программного обеспечения вычислительной среды тренажёрных систем сформулирована задача распределения информации по иерархическим уровням многоуровневой памяти в терминах теории сетей Петри – Маркова. Показано, что поставленная задача является типичной задачей линейного программирования, в которой решения, если они существуют, находятся на границе области допустимых решений. Предложены две методики распределения данных по уровням хранения, примененных в вычислительной среде тренажерной системы.

Ключевые слова: тренажёрная система, сеть Петри – Маркова, программный модуль.

Программное обеспечение современных тренажерных систем строится на основе модульного принципа и естественным образом разделяется на ряд программных модулей, реализующих алгоритмы, каждый из которых осуществляет свою функцию в общей задаче обработки данных [2, 3]. Это позволяет осуществить раздельную разработку, отладку и тестирование программных модулей, а также использовать в составе программного обеспечения ранее разработанные программы. Кроме того, в состав программного обеспечения входят так называемые элементарные программы: драйверы аналого-цифровых и цифроаналоговых преобразователей управляющих и сенсорных сигналов; программы формирования статических характеристик имитаторов органов управления подвижного наземного объекта, близких к реальным; стандартные программы интегрирования систем дифференциальных уравнений и т.п.

Известно, что алгоритмы вычислительной среды тренажёрных систем являются циклическими [1]. Большинству программных реализаций алгоритмов для функционирования требуются значительные объемы обрабатываемых данных, которые хранятся на различных вычислительных модулях. Каждый модуль содержит операторы обращения: к другим программным модулям, базам данных, датчикам сенсорной подсистемы и исполнительным устройствам тренажера. Существуют жесткие ограничения на периоды обращения к датчикам сенсорной подсистемы, определяемые условиями Найквиста теоремы Котельникова (теоремы об отсчетах). Другие ограничения связаны с периодом регенерации изображения на экране монитора. В связи с этим существенное значение приобретает задача минимизации указанных периодов.

Одним из существенных аспектов интерпретации алгоритмов программных модулей тренажерной системы является, помимо обычного информационного обмена между модулями, пересылки значительных массивов информации. При этом время доступа к информации напрямую влияет на величины временных интервалов, реализуемых в циклических алгоритмах. Таким образом, задача распределения информации по иерархическим уровням многоуровневой памяти является весьма существенной.

Сформулируем подходы к её постановке на основе теории сетей Петри –Маркова (СПМ) [4].

Пусть информация общим объемом V_{Σ}, используемая некоторым программным модулем, распределена на $J(a)$ уровнях хранения. Процессор ЭВМ фон-неймановского типа, интерпретируя некоторый алгоритм, генерирует поток заявок на обслуживание, которое заключается в том, что информация с соответствующего уровня хранения передается в ЭВМ для обработки. СПМ, моделирующая взаимодействие процессора и иерархической памяти, приведена на рис. 1.

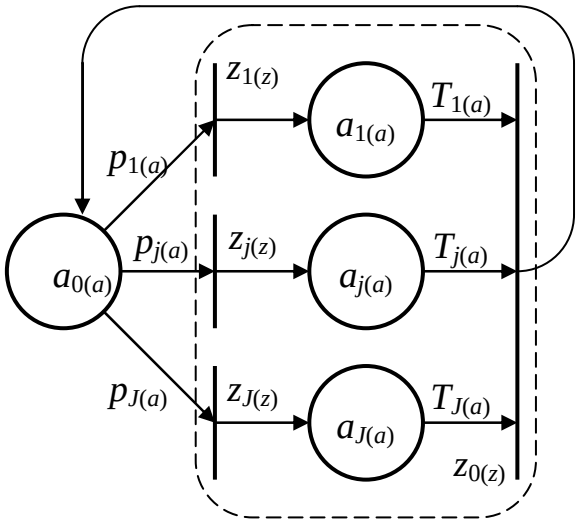

Рис. 1. Модель доступа к информации, распределенной в иерархической многоуровневой памяти

СПМ описывается следующей системой математических выражений:

$$\Psi = \{П, M\}; \quad (1)$$

$$П = \{A, Z, \tilde{\boldsymbol{R}}, \hat{\boldsymbol{R}}\}; M = \{\boldsymbol{q}, \tilde{\boldsymbol{h}}(t), \boldsymbol{\Lambda}\}.$$

$$A = \{a_{0(a)}, a_{1(a)}, ..., a_{j(a)}, ..., a_{J(a)}\}; Z = \{z_{0(z)}, z_{1(z)}, ..., z_{j(z)}, ..., z_{J(z)}\};$$

$$\tilde{\boldsymbol{R}} = \begin{pmatrix} 0 & 1 & ... & 1 & ... & 1 \\ 1 & 0 & ... & 0 & ... & 0 \\ & & & ... & & \\ 1 & 0 & ... & 0 & ... & 0 \\ & & & ... & & \\ 1 & 0 & ... & 0 & ... & 0 \end{pmatrix} \begin{matrix} 0(a) \\ 1(a) \\ ... \\ j(a) \\ ... \\ J(a) \end{matrix}; \quad \hat{\boldsymbol{R}} = \begin{pmatrix} 1 & 0 & ... & 0 & ... & 0 \\ 0 & 1 & ... & 0 & ... & 0 \\ & & & ... & & \\ 0 & 0 & ... & 1 & ... & 0 \\ & & & ... & & \\ 0 & 0 & ... & 0 & ... & 1 \end{pmatrix},$$

где $a_{0(a)}$– позиция, моделирующая генератор потока заявок на обслуживание; $\{a_{1(a)}, ..., a_{j(a)}, ..., a_{J(a)}\}$ – подмножество позиций, каждая из которых моделирует доступ к информации на $j(a)$-м уровне хранения; $\{z_{1(z)}, ..., z_{j(z)}, ..., z_{J(z)}\}$ – подмножество примитивных переходов; $z_{0(z)}$–непримитивный переход, синхронизирующий работу процессора и иерархических уровней памяти.

Пусть на $j(a)$-м уровне хранится информация объемом $V_{j(a)}$ и

$$\sum_{j(a)=1(a)}^{J(a)} V_{j(a)} = V_{\Sigma}. \quad (2)$$

Будем в первом приближении считать, что вся информация является равномерно-релевантной. Это, в свою очередь, означает, что вероятности (частоты) заявок каждой единицы информации одинаковы, а вероятность обращения процессора на $j(a)$-й уровень хранения определяется по зависимости:

$$p_{j(a)} = \frac{V_{j(a)}}{V_{\Sigma}}; \quad \sum_{j(a)=1(a)}^{J(a)} p_{j(a)} = 1. \quad (3)$$

Если объем хранения на $j(a)$-м уровне ограничен величиной $V_{j(a)\max}$, то и вероятности обращения на указанный уровень ограничены величинами

$$p_{j(a)\max} = \frac{V_{j(a)\max}}{V_{\Sigma}}, \quad \sum_{j(a)=1(a)}^{J(a)} p_{j(a)\max} \geq 1. \quad (4)$$

Очевидно, что для равномерно-релевантной информации случай, когда $\sum_{j(a)=1(a)}^{J(a)} p_{j(a)\max} = 1$, не представляет интереса, поскольку является тривиальным.

Пусть время доступа к информации определяется уровнем хранения. Если рассматривать вычислительный модуль соответствующего уровня хранения как обслуживающий прибор с экспоненциальным законом обслуживания $\mu_{j(a)}\exp[-\mu_{j(a)}t]$, то среднее время доступа к информации на n-м

уровне определяется в виде $T_{j(a)} = \dfrac{1}{\mu_{j(a)}}$. Если предположить, что процессор формирует в многоуровневую память равномерный поток заявок с плотностью $\lambda\exp[-\lambda t]$, то полумарковская матрица будет иметь вид

$$\tilde{\boldsymbol{h}}(t) = \begin{pmatrix} 0- & \dfrac{V_{1(a)}}{V_\Sigma} & \dots & \dfrac{V_{1(a)}}{V_\Sigma} & \dots & \dfrac{V_{1(a)}}{V_\Sigma} \\ 1 & 0 & \dots & 0 & \dots & 0 \\ \dots & & & & & \\ 1 & 0 & \dots & 0 & \dots & 0 \\ \dots & & & & & \\ 1 & 0 & \dots & 0 & \dots & 0 \end{pmatrix} \otimes$$

$$\otimes \begin{pmatrix} 0 & \mu_{1(a)}\exp\left[-\mu_{1(a)}t\right] & \dots & \mu_{J(a)}\exp\left[-\mu_{j(a)}t\right] \\ \lambda\exp(-\lambda t) & 0 & \dots & 0 \\ \dots & & & \\ \lambda\exp(-\lambda t) & 0 & \dots & 0 \\ \dots & & & \\ \lambda\exp(-\lambda t) & 0 & \dots & 0 \end{pmatrix}, \quad (5)$$

где $\otimes$– знак прямого произведения.

Матрица логических условий может быть представлена в виде

$$\boldsymbol{\Lambda} = \begin{pmatrix} \bigvee\limits_{j(a)=1(a)}^{J(a)} \left(a_{j(a)}, z_{0(z)}\right) & 0 & \dots & 0 & \dots & 0 \\ 0 & \left(a_{0(a)}, z_{1(z)}\right) & \dots & 0 & \dots & 0 \\ \dots & & & & & \\ 0 & 0 & \dots & \left(a_{0(a)}, z_{j(z)}\right) & \dots & 0 \\ \dots & & & & & \\ 0 & 0 & \dots & 0 & \dots & \left(a_{0(a)}, z_{J(z)}\right) \end{pmatrix}, \quad (6)$$

где $\left(a_{j(a)}, z_{k(z)}\right)$ означает, что соответствующий полушаг выполнен.

В оптимизационной задаче варьируемыми параметрами являются объемы хранения данных $V_{j(a)}$ на соответствующем уровне. Критерий качества имеет вид

$$\varsigma = \sum_{j(a)=1(a)}^{J(a)} \frac{1}{\mu_{j(a)} V_\Sigma} V_{j(a)} \to \min, \quad (7)$$

где $\dfrac{1}{\mu_{j(a)} V_\Sigma} > 0$ – постоянные коэффициенты.

На объемы хранения наложены ограничения

$$0 \le V_{j(a)} \le V_{j(a)\max}. \quad (8)$$

Кроме того, в оптимизационной задаче существует естественное ограничение вида (2).

Поставленная задача является типичной задачей линейного программирования, в которой решения, если они существуют, находятся на границе области допустимых решений. Существует ряд способов решения данной задачи, однако она может быть решена в аналитическом виде.

Без нарушения общности рассуждений будем считать, что каждый следующий уровень хранения имеет больший, или равный, по сравнению с предыдущим уровнем, объем и меньшую, или равную плотность потока обслуживаний, т.е.

$$V_{1(a)} \le \ldots \le V_{j(a)} \le \ldots \le V_{J(a)};\ \mu_{1(a)} \ge \ldots \ge \mu_{j(a)} \ge \ldots \ge \mu_{J(a)}. \quad (9)$$

Условие (9) выполняется, если, например информация распределена по уровням: кэш, оперативной памяти, жесткого диска модуля, решающего одну из конкретных задач в тренажере, жесткого диска файл-сервера тренажера, жесткого диска файл-сервера инструктора.

Предположим, что решением задачи является точка границы $V_{1(a)\max} \le \ldots \le V_{j(a)\max} \le \ldots \le V_{[J(a)-1]\max}$. Тогда критерий качества (7) может быть представлен в виде

$$\begin{aligned} \varsigma &= \sum_{j(a)=1(a)}^{J(a)-1} \frac{V_{j(a)\max}}{\mu_{j(a)} V_\Sigma} + \frac{1}{\mu_{J(a)}}\left(1 - \sum_{j(a)=1(a)}^{J(a)-1} \frac{V_{j(a)\max}}{V_\Sigma}\right) = \\ &= \sum_{j(a)=1(a)}^{J(a)-1} \frac{V_{j(a)\max}}{V_\Sigma}\left(\frac{1}{\mu_{j(a)}} - \frac{1}{\mu_{J(a)}}\right) + \frac{1}{\mu_{J(a)}}. \end{aligned} \quad (10)$$

В (10) $\frac{1}{\mu_{j(a)}} - \frac{1}{\mu_{J(a)}} \le 0$, а значение $V_{j(a)}$ может изменяться от $V_{j(a)\max}$ только в сторону уменьшения. Таким образом, любое изменение $V_{j(a)}$, $1(a) \le j(a) \le J(a) - 1$ может только увеличить среднее время доступа к программам, а следовательно, решение

$$V_{1(a)\max}, \ldots, V_{j(a)\max}, \ldots V_\Sigma - \sum_{j(a)=1(a)}^{J(a)-1} V_{j(a)\max} \quad (11)$$

является оптимальным.

Пусть информация, распределяемая по уровням памяти, не является равномерно-релевантной. В этом случае она может быть разделена на кластеры равного объема v, внутри которых вероятности (частоты) заявок каждой единицы информации приблизительно одинаковы. Предположим, что $V_{j(a)\max}$ и V_Σ делятся на v нацело, без остатков, и что

$$\frac{V_{j(a)\max}}{v} = K[j(a)]; \quad (12)$$

$$\frac{V_\Sigma}{v} = K. \qquad (13)$$

Считая, что релевантность всего массива оценивается в 1, релевантность k-го кластера будем оценивать величиной r_k. Для релевантностей выполняется соотношение

$$\sum_{k=1}^{K} r_k = 1. \qquad (14)$$

Тогда критерий качества будет иметь вид

$$\varsigma = \sum_{j(a)=1(a)}^{J(a)} \frac{v \sum_{k \in g[j(a)]} r_k}{\mu_{j(a)} V_\Sigma} \to \min_{g[j(a)]}, \qquad (15)$$

где $g[j(a)]$– выборка кластеров, помещенных на $j(a)$-й уровень хранения.

На объемы размещаемой информации накладываются ограничения (8, 12), а естественные ограничения принимают вид (13). Задача относится к области дискретного программирования и может быть решена с помощью метода ветвей и границ, однако может быть получено и аналитическое решение задачи.

Перенумеруем кластеры по возрастанию релевантности, введя номер кластера k, $1 \le k \le K$. В соответствии с уровнями может быть сформирована гистограмма релевантности, вид которой приведен на рис. 2. В гистограмме $r_1 \le ... \le r_k \le ... \le r_K$.

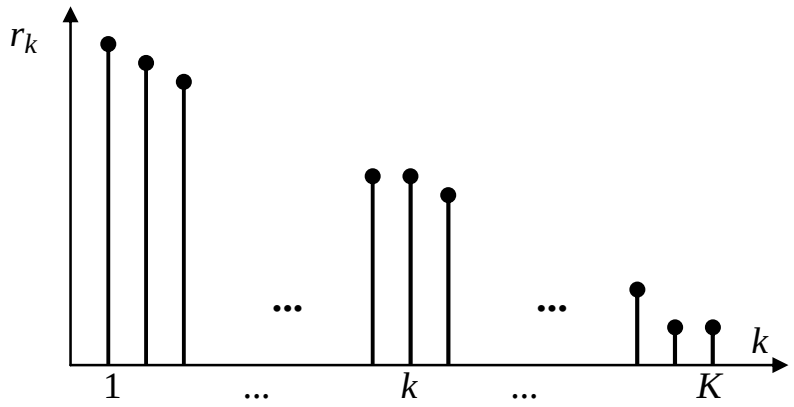

Рис. 2. Гистограмма релевантности

Определим кластеры с номерами $K[j(a) - 1] + 1 \le k \le K[j(a)]$ упорядоченной гистограммы на $j(a)$-й уровень хранения. Тогда критерий качества (14)

$$\varsigma = \sum_{j(a)=1(a)}^{J(a)-1} \frac{v \sum_{k=K[j(a)-1]+1}^{K[j(a)]} r_k}{\mu_{j(a)} V_\Sigma} + \frac{v \sum_{k=K[J(a)-1]+1}^{K} r_k}{\mu_{J(a)} V_\Sigma}. \qquad (16)$$

Физический смысл оптимального решения (7) заключается в том, что величины весов при значениях $V_{j(a)\max}$, $1(a) \le j(a) \le J(a) - 1$, не убывают при возрастании $j(a)$. Это правило выполняется и для (16).

Поменяем в (16) любые два кластера местами, например кластер с номером $k[j(a)]$ с уровня хранения $j(a)$ переместим на место кластера $l[m(a)]$ уровня хранения $m(a)$, а кластер $l[m(a)]$ с уровня хранения $m(a)$ переместим на место кластера $k[j(a)]$ с уровнем хранения $j(a)$, $1(a) \le m(a) \le J(a)$, $m(a) > j(a)$, $\mu_{m(a)} \le \mu_{j(a)}$, $K[j(a) - 1] + 1 \le k[j(a)] \le K[j(a)]$, $K[m(a) - 1] + 1 \le k[m(a)] \le K[m(a)]$. Отметим, что $r_{m(a)} \le r_{j(a)}$. Поскольку все кластеры, кроме исследуемых, остаются на своих местах, изменение функции качества может быть определено по зависимости:

$$\tilde{\Delta}_\varsigma = \frac{v}{V_\Sigma}\left[\frac{1}{\mu_{j(a)}}\left(r_{m(a)} - r_{j(a)}\right) + \frac{1}{\mu_{m(a)}}\left(r_{j(a)} - r_{m(a)}\right)\right] =$$

$$= \frac{v}{V_\Sigma}\left[\frac{\Delta r}{\mu_{j(a)}} - \frac{\Delta r}{\mu_{m(a)}}\right] = \frac{v\Delta_r\Delta_\mu}{V_\Sigma\mu_{j(a)}\mu_{m(a)}} \ge 0, \qquad (17)$$

где $\Delta_r = r_{m(a)} - r_{j(a)} \le 0$; $\Delta_\mu = \mu_{m(a)} - \mu_{j(a)} \le 0$.

Таким образом, любая смена размещения кластеров по уровням памяти, по сравнению с размещением (16) приводит к увеличению среднего времени доступа к данным.

Отметим, что в решении (16) сдвиг кластеров в сторону нижних уровней иерархии памяти (на единицу влево) невозможен, так как все нижние уровни заполнены данными. Сместим кластеры, начиная с кластера с номером $k[j(a)]$, на одну позицию в сторону верхних уровней иерархии памяти (на единицу вправо). При этом с уровня $j(a)$ будет изъят кластер с номером $K[j(a)]$. На уровнях с $j(a) + 1$ по $J(a) - 1$ кластеры с $1[j(a) + 1]$ по $K[j(a) + 1] - 1$ останутся на месте, но будет добавлен кластер $K[j(a)]$ и изъят кластер $K[j(a) + 1]$. На уровень $K[J(a)]$ будет добавлен кластер $K[J(a) - 1]$. Изменение функции качества определится следующей зависимостью:

$$\hat{\Delta}_\varsigma = \frac{v}{V_\Sigma}\sum_{m(a)=j(a)}^{J(a)-1} r_{K[m(a)]}\left(\frac{1}{\mu_{m(a)+1}} - \frac{1}{\mu_{m(a)}}\right) =$$

$$= \frac{v}{V_\Sigma}\sum_{m(a)=j(a)}^{J(a)-1} \frac{r_{K[m(a)]}\delta_{\mu,m(a)}}{\mu_{m(a)}\mu_{m(a)+1}} \ge 0, \qquad (18)$$

где $\delta_{\mu,m(a)} = \mu_{m(a)} - \mu_{m(a)+1} \ge 0$.

Таким образом, сдвиг кластеров на единицу вправо также увеличивает функцию качества, что и доказывает оптимальность решения (16).

Приведенные рассуждения и доказательства позволяют сформулировать две простых методики распределения данных по уровням хранения, примененных в вычислительной среде тренажерной системе.

Методика 1 распределения равномерно-релевантных данных.

При распределении равномерно-релевантных данных по уровням хранения должны первоначально заполняться уровни с меньшим временем доступа.

Методика 2 распределения неравномерно-релевантных данных.

При распределении неравномерно-релевантных данных по уровням хранения должны первоначально заполняться уровни с меньшим временем доступа данными, обладающими наибольшей релевантностью.

Таким образом, на основе предлагаемого подхода возможно проектирование оптимизационных процедур при проектировании программного обеспечения вычислительной среды тренажёрных систем.

Список литературы

1. Ларкин Е.В., Привалов А.Н. Проектирование программного обеспечения вычислительной среды тренажерных систем. Тула: ТулГУ, 2010. 259 с.

2. Привалов А.Н. Применение унифицированных программных модулей при разработке тренажёрных систем// Программные продукты и системы. Тверь.2009. № 4. С. 92 – 98.

3. Привалов А.Н., ЛаркинЕ.В. Создание программного обеспечения тренажерных систем на основе унифицированных программных модулей // Вестник компьютерных и информационных технологий.2010.№ 4. С. 75 – 82.

4. Питерсон Дж. Теория сетей Петри и моделирование систем. М.: Мир, 1984. 264 с.

Ларкин Евгений Васильевич, д-р техн. наук, проф., зав. кафедрой, elarkin@mail.ru, Россия, Тула, Тульский государственный университет,

Привалов Александр Николаевич, д-р техн. наук, проф., privalov.61@mail.ru, Россия, Тула, Тульский государственный педагогический университет им. Л.Н. Толстого

ON THE PROBLEM OF DISTRIBUTION INFORMATION ON STORAGE LEVELS IN COMPUTING MODULE TRAINING SYSTEMS

E.V.Larkin, A.N.Privalov

The problem of sharing information on the multi-level memory hierarchies is formulated on the basis of the postulate of minimizing the period of inter-module interaction software computing environment simulation systems in terms of the theory of Petri – Markov nets. It is shown that the problem is a typical linear programming problem in which solutions, if they exist, are at the boundary of the domain of feasible solutions. Formulated two methods for distributing data storage levels applied in the computing environment, the simulator system.

Key words: simulation system, Petri – Markov nets software module.

Larkin EvgeniyVasilevich, doctor of technical sciences, professor, manager of department, elarkin@mail.ru, Russia, Tula, Tula State University,

Privalov Aleksandr Nicolaevich, doctor of technical sciences, professor, privalov61 @mail.ru, Russia, Tula, Tula State Pedagogical University named after L.N. Tolstoy

УДК 681.5 (519.95)

МОДЕЛИРОВАНИЕ ВЗАИМОДЕЙСТВИЯ В ВЫЧИСЛИТЕЛЬНОЙ СРЕДЕ ТРЕНАЖЁРНОЙ СИСТЕМЫ КАК СИСТЕМЕ МАССОВОГО ОБСЛУЖИВАНИЯ

А.Н. Привалов

Рассмотрен подход к моделированию информационного взаимодействия между субъектами вычислительной среды тренажёрной системы как системы массового обслуживания. Предложена модель автономного генератора, формирующего поток заявок на обслуживание, как элементарная подсеть Петри – Маркова. Показано преобразование её в полумарковский процесс, приведены выражения для расчёта количественных характеристик. Разработана методика ликвидации петель в графе полумарковского процесса.

Ключевые слова: тренажёрная система, элементарная подсеть Петри-Маркова, система массового обслуживания, автономный генератор заявок, полумарковский процесс, граф состояний.

Современные тренажёрные системы и комплексы являются сложными системами, функционирующими под управлением вычислительной среды. Вычислительная среда тренажерной системы включает значительное количество субъектов (центральная и периферийные ЭВМ, файл-серверы, контроллеры сенсоров и исполнительных механизмов и т.п.), взаимодействующих между собой. При этом, как правило, субъекты действуют автономно, по известному алгоритму, а взаимодействие осуществляется через формирование потоков запросов одного из субъектов на взаимодействие с другими субъектами. Поэтому каждый из взаимодействующих субъектов вычислительной среды может рассматриваться, с одной стороны, как программный генератор потока запросов на обслуживание, а с другой стороны, как программный обслуживающий прибор системы массового обслуживания. Это создаёт предпосылки для того, что к исследованию процессов, протекающих в вычислительной среде, может быть применен аппарат теории массового обслуживания (СМО).