УДК 681.142

ОБ ОПТИМАЛЬНОМ ПРИОРИТЕТНОМ ОБСЛУЖИВАНИИ В ПРОГРАММНОМ ОБЕСПЕЧЕНИИ ЭАТС

Б. С. ГОЛЬДШТЕЙН

Работа электронных управляющих машин (ЭУМ) узлов коммутации в реальном масштабе времени характеризуется целым рядом временных ограничений на выполнение различных программ, определяемых процессом обслуживания телефонной нагрузки и функционированием оборудования узла. Нарушение временных соотношений эквивалентно потерям в узле коммутации, так как теряется временная связь процесса управления с состоянием внешних источников и потребителей информации – телефонной периферией. Это вызывает необходимость организации в составе телефонной операционной системы ЭУМ алгоритмов приоритетного обслуживания (АПО).

АПО является той базой, на которой основывается работа других программных блоков операционной системы и с помощью которой определяется, в какой последовательности, когда и какая из активных программ вызывается на счет. Совокупность правил выбора, реализованных в АПО, состоит в следующем. Из списка активных программ выбирается программа с наибольшим значением приоритета. Запрос на включение программы приоритета, отличного от приоритета работающей программы, инициирует прерывание последней, причем в зависимости от принадлежности работающей программы определенному приоритетному уровню, такое прерывание может быть либо разрешено, либо заблокировано. Запрос на включение программы, приоритет которой принадлежит тому же или более низкому уровню, прерывания не вызывает. Таким образом реализуется приоритетная схема обслуживания с абсолютно-относительными приоритетами.

Опубликовано большое количество работ, в которых рассматриваются различные системы приоритетного обслуживания. Подробная библиография имеется, например, в [1]. Рассматриваемая в настоящей работе система обладает следующими свойствами.

Классификация потоков запросов, составляющих системную нагрузку операционной системы, строится по типам поступающей информации, допустимому времени на ее обработку и возможному штрафу на задержку в выполнении обслуживающей программы. Подчеркнем следующее обстоятельство: нагрузку операционной системы создают не только и не столько вызовы телефонных абонентов, но внутренние заявки, возникающие в самом процессе обслуживания вызовов, при управлении соединениями и разъединениями в коммутационной системе, обеспечении взаимодействия с АТС существующих систем, организации дополнительных видов обслуживания, решении задач контроля и диагностики, подготовке и вводе информации эксплуатационному персоналу узла, решении фоновых задач и т. д. Таким образом, источниками запросов на включение тех или иных программ является сама ЭУМ и, косвенно, телефонная периферия. Так как необходимо обеспечение обслуживания всех без исключения запросов на включение программ, АПО может рассматриваться как обслуживающая система, не допускающая явных потерь. При этом последовательность приоритетов выбирается с учетом допустимых времен на выполнение программ. Расчет АПО такого рода осуществляется для наименее благоприятной для узла коммутации ситуации – часа наибольшей нагрузки (ЧНН).

При наличии в ЭУМ любых запросов на включение программ центральный процессор начинает их обслуживание, а не ждет возникновения более важных запросов. Очереди запросов ограничены телефонной операционной системой и рассчитываются так, что

исключается возможность потери запроса из-за отсутствия мест на ожидание. Не ограничивая общности, можно считать, что номер очереди совпадает с номером приоритета. Заявки поступают на обслуживание в соответствии с системой абсолютно-относительных приоритетов, установленной заранее по алгоритму, приводимому ниже. Этой же системой определяются условия прерывания программы. В случае прерывания рабочей программы запросом более высокого приоритетного уровня возобновление работы этой программы продолжается с прерванного места. Запросы, имеющие одинаковый приоритет, обслуживаются по принципу «первый пришел – первый обслужен».

Для анализа такой системы с абсолютно-относительными приоритетами представляется целесообразным воспользоваться двумерным представлением приоритетов [2]. Пусть имеется N потоков заявок на включение L программ ($L \leq N$), которым поставлены в соответствие N приоритетов. Пусть эти приоритеты распределены каким-то образом по K уровням. Обслуживание запроса любого приоритета уровня k ($k = 2, 3, \ldots, K$) прерывается при появлении другого запроса, соответствующего меньшему значению k ($k = 1, 2, \ldots, K-1$). На каждом уровне располагается M_k приоритетов, запросы которых не превышают друг друга.

Таким образом, приоритеты можно описывать в виде пар чисел (k, m). На уровне k таких пар будет $(k, 1), (k, 2), \ldots, (k, M_k)$ Очевидно,

$$\sum_{k=1}^{k} M_k = N.$$

Разбиение $\mathcal{M}$ $\{M_1, M_2, \ldots, M_k\}$ полностью определяет АПО. Запросы приоритета (k, m) образуют пуассоновский поток интенсивности λ (k, m). Длительность выполнения программы, обслуживающей запрос из потока приоритета (k, m), является случайной величиной с функцией распределения $B_{k,\,m}(t)$, первым моментом $b(\kappa, m)$ и вторым начальным моментом $b^{(2)}(k, m)$.

В рассматриваемой системе с неограниченной очередью прерывание обслуживающей программы не приводит к потере затраченного времени, длительности периода занятости центрального процессора не зависят от порядка обслуживания, моменты окончания (или начала) периодов занятости являются точками регенерации для процесса, описывающего число заявок в ЭУМ в данный момент. Для такой системы стационарный режим имеет место при коэффициенте загрузки R, меньшем единицы [3]:

$$R = \sum_{i=1}^{K}\sum_{j=1}^{M_i} \lambda(i, j)\, b(i, j) = \sum_{i=1}^{K}\sum_{j=1}^{M_i} \rho(i, j) < 1,$$

где $\rho(i, j)$–коэффициент загрузки ЭУМ потоком запросов приоритета (i, j).

При помощи так называемых прямых методов, развитых в [1], вычислим средние значения параметров рассматриваемой системы. Обозначим через $U(k, m)$ среднее время пребывания в ЭУМ запроса приоритета (k, m), $W(k, m)$ – среднее время ожидания начала обслуживания, $V(k, m)$ – среднее время обслуживания, т.е. время от начала работы обслуживающей программы до завершения обслуживания запроса. Очевидно,

$$U(k, m) = W(k, m) + V(k, m). \qquad (1)$$

Рассмотрим среднее полное время обслуживания $V(k, m)$. Заметим, что на величину $V(k, m)$ не оказывают никакого влияния запросы из потоков с более низкими приоритетами $(k, m+1), \ldots, (k, M_k)$, поэтому их можно не учитывать при оценке $V(k, m)$. Таким образом, достаточно рассмотреть систему с потоками $(1, 1), (1, 2), \ldots, (k, m)$.

Для определения $V(k, m)$ воспользуемся следующим приемом. Потоки (1, 1), (1, 2), . . ., (k - 1, M_{k-1}) обладают абсолютным приоритетом по отношению к фиксированному запросу приоритета (k, m). Мысленно положим, что заявки из потоков (k, 1), (k, 2), . . ., (k, m - 1) обладают этим же свойством. В этом случае имеем систему с абсолютными приоритетами и для такой модифицированной системы согласно [1, 4]

$$V'(k,m)=\frac{b(k,m)}{1-\sum_{i=1}^{k}\sum_{j=1}^{\varphi(k,m)}\rho(i,j)},$$

где

$$\varphi(a,b)=\begin{cases}M_i & \text{ïðè } i<a\\ b & \text{ïðè } i=a\end{cases}.$$

В рассматриваемой системе с комбинированными приоритетами $V(k, m) = V'(k, m) - \Delta V(k, m)$, где величина $\Delta V(k, m)$ обусловлена следующими факторами. За время $\Theta(k, m)$, следующее за последним перерывом в работе программы, обслуживающей запрос (k, m), в ЭУМ могут скопиться запросы из потоков (k, 1), (k, 2), . . ., (k, m - 1), которые не прерывают работы программы. Суммарная интенсивность этих потоков $\sum_{j=1}^{m-1}\lambda(k,j)$ и среднее время обслуживания $\sum_{j=1}^{m-1}\rho(k,j)\Big/\sum_{j=1}^{m-1}\lambda(k,j)$. Таких запросов в ЭУМ скопится в $\Theta(k, m)$ $\sum_{j=1}^{m-1}\lambda(k,j)$.

Кроме того, в модифицированной системе с абсолютными приоритетами за время $\Theta(k, m)$ пошли бы новые запросы из потоков (1,1), (1,2), . . ., (k, m - 1), которые также были бы обслужены. Этот факт [3] учитывается множителем $\left[1-\sum_{i=1}^{k}\sum_{j=1}^{\varphi(k,m-1)}\rho(k,j)\right]^{-1}$. После несложных преобразований получаем

$$V(k,m)=\frac{b(k,m)-\Theta(k,m)\sum_{j=1}^{m-1}\rho(k,j)}{1-\sum_{i=1}^{k}\sum_{j=1}^{\varphi(k,m-1)}\rho(k,j)}. \qquad (2)$$

Определим величину $\Theta(k, m)$. Пусть время работы программы, обслуживающей запросы (k, m) равно t. С вероятностью $\exp\left\{-t\sum_{i=1}^{k-1}\sum_{j=1}^{M_i}\lambda(i,j)\right\}$ это обслуживание не будет прервано и тогда $\Theta(k, m) = b(k, m)$. Для потоков (1,1), (1,2), . . ., (1, M_1) эта вероятность равна единице и $\Theta(1, m) \equiv b(k, m)$. В случае прерываний с вероятностью $1-\exp\left\{-x\sum_{i=1}^{k-1}\sum_{j=1}^{M_i}\lambda(i,j)\right\}$ время $\Theta(k, m)$ будет меньше некоторого значения x.

Отсюда

$$\Theta(k,m)=\int_0^{\infty}\left[t\exp\left\{-t\sum_{i=1}^{k-1}\sum_{j=1}^{M_i}\lambda(i,j)\right\}+\int_0^{t}x\,d\left(1-\exp\left\{-x\sum_{i=1}^{k-1}\sum_{j=1}^{M_i}\lambda(i,j)\right\}\right)\right]dB_{k,m}(t).$$

Произведя интегрирование, получим

$$\Theta(k,m)=\begin{cases} b(k,m) & \text{ïðè} \quad k=1 \\ \dfrac{1}{\sum\limits_{i=1}^{k-1}\sum\limits_{j=1}^{M_i}\lambda(i,j)}\int\limits_0^{\infty}\left[1-\exp\left\{-t\sum\limits_{i=1}^{k-1}\sum\limits_{j=1}^{M_i}\lambda(i,j)\right\}\right]dB_{k,m}(t) & \text{ïðè} \quad k\neq 1. \end{cases} \qquad (3)$$

Среднее время ожидания начала обслуживания для произвольного запроса из потока (*k, m*) определяется аналогичным методом. Представим модифицированную систему с потоками (1,1), (1,2), . . ., (*k, M_k*) *и с* абсолютными приоритетами, тогда, согласно [1],

$$W'(k,m)=\frac{\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m)}\rho(i,j)\dfrac{b^{(2)}(i,j)}{2b(i,j)}}{\left[1-\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m)}\rho(i,j)\right]\left[1-\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m-1)}\rho(i,j)\right]}.$$

В исследуемой системе *W* (*k, m*) = *W'* (*k, m*) + ΔW (*k, m),* где последнее слагаемое определяется из следующих соображений. В момент инициирования запроса (*k, m*) может выполняться программа, обслуживающая запрос из потоков (*k, m* + 1), (*k, m* + 2), . . ., (*k, M_k*), работу которой запрос (*k, m*) не прерывает. Среднее время работы такой программы в присутствии запроса (*k, m*) обозначим через *ν* (*k, j*). В системе за это дополнительное время могут быть ииициированы заявки из потоков (1,1), (1,2), . . ., (*к, m* - 1), что учитывается величиной

$$\frac{\sum\limits_{j=m+1}^{M_k}\rho(k,j)\,\nu(k,j)}{\left[1-\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m-1)}\rho(i,j)\right]\left[1-\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m)}\rho(i,j)\right]}$$

Однако запросы из потоков (1,1), (1,2), . . ., (*k*– 1, M_{k-1}) прерывают работу программы *(k, j),* следовательно,

$$\Delta W(k,m)=\frac{\left[1-\sum\limits_{i=1}^{k-1}\sum\limits_{j=1}^{M_i}\rho(i,j)\right]\sum\limits_{j=m+1}^{M_k}\rho(k,j)\,\nu(k,j)}{\left[1-\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m-1)}\rho(i,j)\right]\left[1-\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m)}\rho(i,j)\right]}.$$

Таким образом,

$$W(k,m)=\frac{\frac{1}{2}\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m)}\lambda(i,j)\,b^{(2)}(i,j)+\left[1-\sum\limits_{i=1}^{k-1}\sum\limits_{j=1}^{M_i}\rho(i,j)\right]\sum\limits_{j=m+1}^{M_k}\rho(k,j)\,\nu(k,j)}{\left[1-\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m-1)}\rho(i,j)\right]\left[1-\sum\limits_{i=1}^{k}\sum\limits_{j=1}^{\varphi(k,m)}\rho(i,j)\right]}, \qquad (4)$$

где ν(*k, j*) по аналогии с [3] может быть выражена через Θ(*k, j*) следующим образом. Пусть в момент инициирования запроса (*k, m)* время, оставшееся до завершения работы программы (*k, j*), равно *t.* Эта программа будет выполнена либо полностью, либо будет прервана запросом из потоков (1,1), (1,2), . . ., *(k* - 1, M_{k-1}*).* Учитывая, что плотность вероятности времени *t* равна [1 - $B_{k,j}$ (*t*)]/*b* (*k, j*) [4], имеем

$$\nu(k,j)=\int_0^{\infty}\frac{1-\exp\left\{-t\sum_{i=1}^{k-1}\sum_{j=1}^{M_i}\lambda(i,j)\right\}}{\sum_{i=1}^{k-1}\sum_{j=1}^{M_i}\lambda(i,j)}\quad\frac{1-B_{k,j}(t)}{b(k,j)}=\frac{b(k,j)-\Theta(k,j)}{b(k,j)\sum_{i=1}^{k-1}\sum_{j=1}^{M_i}\lambda(i,j)}. \qquad (5)$$

Введенная в рассмотрение дисциплина обслуживании с комбинированными приоритетами представляет известный практический интерес, поскольку такая дисциплина при оптимальном выборе $\mathfrak{M} = \{M_1, M_2, \ldots, M_k\}$ в принципе обеспечивает не худшее обслуживание по сравнению с «чистыми» дисциплинами (с абсолютными и относительными приоритетами). Так, при $K = N$, $M_i = 1$, $\forall i = 1, 2, \ldots, K$ мы получаем обслуживание с абсолютным приоритетом, а при $K = 1$, $M_1 = N$ – с относительными приоритетами.

Анализ параметров АПО проводился при упрощающем предположении об отсутствии потерь на прерывания программ. Для анализа и поиска оптимального АПО, более адекватных реальной ситуации, следует предположить наличие некоторых непроизводительных затрат времени процессора на переключение программ. Впрочем, при аппаратной реализации системы прерывания эти затраты времени достаточно малы. Однако объем оперативной памяти ЭУМ, который связан с защитой прерываемой программы при переключении $(i, j) \rightarrow (k, m)$, с запоминанием содержимого базовых и операционных регистров, регистра адреса команд, соответствующих признаков и т. д. для последующего восстановления прерванной программы при окончании прерывания $(k, m) \rightarrow (i, j)$ существен для эффективной работы ЭУМ. Эти затраты оперативной памяти и объем аппаратных средств на реализацию системы прерывания пропорциональны числу уровней абсолютного приоритета.

В связи с этим определим оптимальность АПО следующим образом: набор приоритетных индексов, соответствующих разбиению $\mathfrak{M} = \{M_1, M_2, \ldots, M_k\}$, обеспечивающий выполнение всех временных ограничений

$$\begin{aligned} &U(1,1) < \tau(1,1),\\ &U(1,2) < \tau(1,2),\\ &\ldots\ldots\ldots\ldots\ldots\\ &U(K, M_k) < \tau(K, M_k), \end{aligned} \qquad (6)$$

при оптимальном значении K оптимален.

Так как число потоков N конечно, задача поиска оптимального АПО может быть решена простым перебором и выбором лучшего из всех возможных вариантов разбиения $\mathfrak{M} = (M_1, M_2, \ldots, M_k)$ при условии выполнения (6) хотя бы одного такого варианта.

Однако нетрудно показать, что общее число возможных комбинаций равно Z^{N-1}, а при больших N метод перебора нереален. Преодоление этих трудностей возможно на путях разработки таких методов, которые позволяют исключить из рассмотрения большее число этих комбинаций. В качестве одного из таких методов предлагается алгоритм направленного поиска оптимального АПО, позволяющий получить решение, рассматривая ничтожно малый процент возможных комбинаций.

Для рассматриваемой системы с комбинированными приоритетами алгоритм направленного поиска оптимального АПО имеет следующий вид.

1. Первому потоку из совокупности 1, 2, . . . , N присвоить приоритет (1, 1), т. е. k: =

1, *m:* = 1.

2. Положить значение $M_k = N - \sum_{i=1}^{k-1} M_i$.

3. Проверить выполнение условия $M_k > 0$. Если условие выполнено, вычислить величину $U(k, m)$ по формулам (1), (2) и (4) с учетом (3) и (5). В противном случае сформировать и вывести на печать сообщение о несовместимости системы ограничений (6) и завершить выполнение алгоритма.

4. Проверить выполнение соответствующего неравенства из (6) для $U(k, m)$. Если (6) для $U(k, m)$ не выполняется, т.е. временные ограничения для потока заявок (k, m) не соблюдаются при данной комбинации приоритетов, положить M_k: = M_k - 1 и вернуться к п.3. Если (6) для $U(k, m)$ выполняется, перейти к п. 5.

5. Следующему потоку из совокупности 1, 2, . . ., N присвоить приоритет $(k, m + 1)$, т.е. $m: = m + 1$.

6. Проверить выполнение равенства $m + 1 = M_k$. Если равенство не выполняется, вернуться к п. 3. В противном случае перейти к п. 7.

7. Проверить выполнение равенства $\sum_{i=1}^{k} M_i = N$. Если равенство выполняется, принять $K: = k$, вывести на печать значение $M_1\ M_2$, . . ., M_k и завершить выполнение алгоритма. В противном случае перейти к п. 8.

8. Следующему потоку из совокупности $1, 2, \ldots,$ присвоить приоритет $(к + 1,1)$, т.е. $k:= k + 1$, $m:= 1$ и возвратиться к п. 2.

Найденному алгоритму можно дать ясную физическую интерпретацию: процессор обслуживает запросы на включение программ таким образом, чтобы с минимальным запасом обеспечить соблюдение всех временных соотношений, стараясь при этом ограничиться наименьшим возможным числом уровней абсолютного приоритета K, а следовательно, наименьшим объемом оперативной памяти под запоминание прерванных задач и временем процессора на обработку сигналов прерывания.

При работе алгоритма полагается, что потоки запросов занумерованы в порядке убывания их важности и срочности и с учетом того, что запросы, выполнение которых логически взаимосвязано, должны обслуживаться в строго определенном порядке. В связи с этим перенумерация потоков в процессе работы алгоритма недопустима.

Общее число циклов C, необходимое для получения оптимального АПО по алгоритму направленного поиска, можно определить по формуле

$$C = K(N+1) - \sum_{k=1}^{K} \sum_{i=1}^{k} M_i.$$

Для системы с N = 9 оптимальный АПО, определяемый разбиением $\mathfrak{M}$ = {3, 2, 4}, был найден через 13 циклов алгоритма направленного потока. Методом полного перебора потребовалось бы рассмотреть около 256 вариантов. Это подтверждает высокую эффективность предлагаемого алгоритма, которая возрастает с ростом N.

По описанному алгоритму были составлены программы для ЭУМ «Нева». В заключении отметим, что описанный метод успешно прошел апробирование практикой при разработке конкретного АПО для импульсно-временного транзитного узла коммутации.

Литература

1. *Бронштейн О. И., Духовный И. М.* Модели приоритетного обслуживания в информационно-вычислительных системах. М.: Наука, 1976.
2. *Herzog U.* Prioriti models for communication processors including overhead. – In: International Teletraffic Congress. Melburne, Nov., 10–17,1976.
3. *Бронштейн О. И., Розенталъ Г. О.* О двух алгоритмах обслуживания в системе со смешанными приоритетами.– В кн.: Вопросы промышленной кибернетики. М., 1969, с. 48-50 (Труды / ЦНИИКА; Вып. 6).
4. *Климов Г. П.* Стохастические системы обслуживания. М.: Наука, 1966.