

СТРУКТУРА СПЕЦПРОЦЕССОРА ДЛЯ ОПТИМИЗАЦИИ РАЗМЕЩЕНИЯ ФАЙЛОВ ПО УЗЛАМ ВЫЧИСЛИТЕЛЬНОЙ СЕТИ

Ладыженский Ю.В., Бельков Д.В.

Кафедра ПМИИ ДонНТУ

ladyzhen@pmi.donetsk.ua

Abstract

Ladyzhensky Y.V., Belkov D.V. Structure of a special processor for optimal allocating of files in computer network. A mathematical model and a heuristic algorithm to locate files in a network are proposed. Special processor to solve this problem with the algorithm is proposed.

Введение

При проектировании и эксплуатации вычислительных сетей возникает задача оптимального управления размещением файлов по узлам сети. Особую актуальность она приобретает в связи с развитием сети INTERNET, в государственных и корпоративных сетях управления, в локальных сетях для автоматизированного проектирования [1]. В задаче необходимо оптимизировать интенсивность информационного трафика для уменьшения времени ответа на запросы к файлам.

Обозначим: F_{ij} - количество запросов к файлу i из узла j в единицу времени; $X_{ij}=1$, если файл i расположен в узле j , иначе $X_{ij}=0$; V_i - объем файла i ; B_j - объем узла j ; T_j - время использования файлов узла j ; $i=1\dots m, j=1\dots n$.

Целевая функция

$$L = \sum_{i=1}^m \sum_{j=1}^n F_{i,j} X_{ij} T_j = \sum_{i=1}^m \sum_{j=1}^n L_{i,j} X_{i,j} \rightarrow \max \quad (1)$$

Ограничения:

$$X_{ij} \in \{0,1\}, \sum_{j=1}^n X_{ij} = 1, i=1\dots m \quad (2)$$

$$\sum_{i=1}^m V_i X_{ij} \leq B_j, j=1\dots n \quad (3)$$

В задаче (1)-(3) максимизируется суммарный поток локальных запросов. В ней необходимо найти матрицу размещений файлов X . Алгоритм решения задачи состоит из m этапов. На этапе i выполняется процедура размещения файла i в один из n узлов. Процедура состоит из 3 шагов. На первом шаге сравниваются свободный объем каждого узла и объем файла i чтобы найти те узлы, в которые файл помещается по размеру. На втором шаге, среди найденных узлов определяется узел с наибольшим значением $L_{ij} = F_{i,j} T_j, i=1, \dots, m; j=1, \dots, n$. На третьем шаге файл размещается в этот узел. Исходными данными для процедуры служат значения F_{ij}, T_j, V_i, B_j . Величина B_j первоначально совпадает со свободным объемом узла j - U_j . В результате работы процедуры формируются новые значения U_j и строка i матрицы X . Временная сложность алгоритма - $O(m(2n-1))$. Алгоритм размещает файлы по принципу "в первый подходящий узел в порядке убывания размеров файлов" (FFD). Перед

применением алгоритма файлы нужно упорядочить по убыванию их размеров, а узлы – по возрастанию объемов.

Задача управления размещением данных по узлам вычислительной сети должна решаться в режиме реального времени. Поэтому необходима аппаратная реализация алгоритма ее решения.

Структура системы

На рисунке 1 показана структура системы, которая состоит из спецпроцессора, буферной памяти и основного процессора. Буферная память имеет область входных регистров и выходной регистр. Система последовательно выполняет m этапов. Каждый этап имеет 5 шагов. На первом шаге этапа i основной процессор пересылает исходные данные для формирования строки i матрицы X во входные регистры. На втором шаге спецпроцессор считывает эти данные. На третьем шаге спецпроцессор структурно выполняет процедуру размещения файла i в один из n узлов. На четвертом шаге спецпроцессор передает результаты работы процедуры в выходной регистр. На пятом шаге и основной процессор считывает эту строку из буферной памяти.

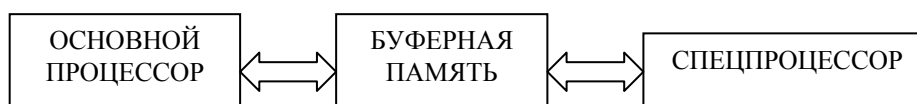


Рисунок 1 – Структура системы

Буферная память

Входные данные являются r – разрядными целыми числами без знака. Выходные данные – строка i матрицы X , $X_{i,j} \in \{0;1\}$. Размещение входных и выходных данных в буферной памяти показано на рисунке 2.

Входные			регистры
$F_{i,1}$	T_1	U_1	V_i
$F_{i,2}$	T_2	U_2	Выходной регистр
...	
$F_{i,j}$	T_j	U_j	X_i
...	
$F_{i,n}$	T_n	U_n	

Рисунок 2 – Размещение данных в буферной памяти

Входные данные размещены в $3n+1$ регистрах разрядностью r , выходные данные размещены в регистре разрядностью n . На первом шаге алгоритма $U_j = B_j$, $j=1,2,\dots,n$. На четвертом шаге алгоритма обновляются значения U_j во входных регистрах, и заполняется выходной регистр.

Структура спецпроцессора

Функции модулей

Спецпроцессор имеет 3 модуля (рис. 3). Модуль формирования признаков выполняет первый шаг процедуры и формирует значения признаков: $Z_j = 1$, если файл

i может поместиться в узел j , иначе $Z_j = 0$. Модуль выбора узла выполняет второй шаг процедуры и вычисляет значения X_{ij} : $X_{ij} = 1$, если файл i должен размещаться в узел j , иначе $X_{ij} = 0$. Модуль заполнения узла вычисляет новые значения U_j : $U_j := U_j - V_i X_{i,j}$. Каждый модуль имеет локальную входную и выходную память, операционное устройство и устройство управления.

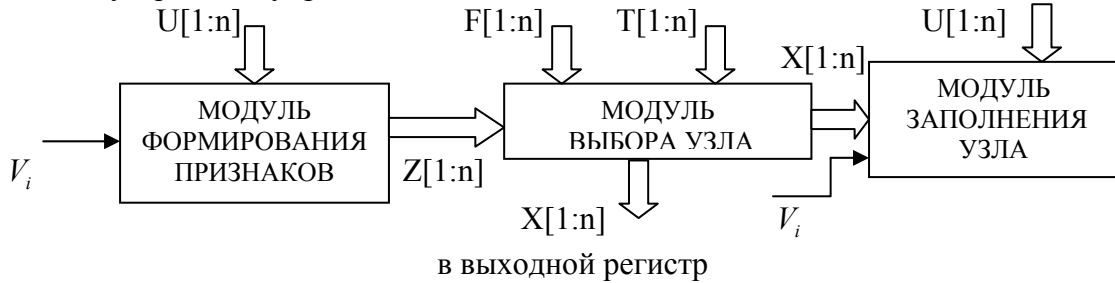


Рисунок 3 – Структура спецпроцессора

Модуль формирования признаков

Входная память модуля признаков состоит из r – разрядных регистров, хранящих числа U_j и V_i , $j=1,2,\dots,n$. Выходная память представляет собой n - разрядный регистр признаков Z_1, Z_2, \dots, Z_n . Операционное устройство (рис.4) состоит из n блоков, работающих параллельно.

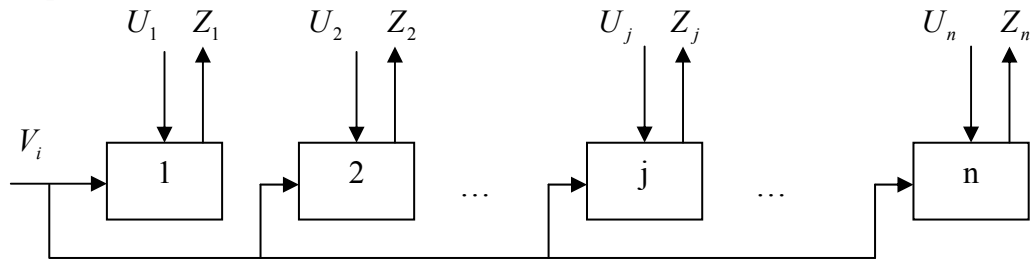


Рисунок 4 - Операционное устройство модуля формирования признаков

Блок j формирует признак Z_j : $Z_j = 1$, если $U_j \geq V_i$, иначе $Z_j = 0$. Он представляет собой итеративную сеть r ячеек [2] (рис. 5). На рисунке обозначено: U_j^k - разряд k числа U_j ; V_i^k - разряд k числа V_i ; P^{r-1}, R^{r-1} - входные граничные сигналы: $P^{r-1} = 1$, $R^{r-1} = 0$; P^0, R^0 - выходные граничные сигналы. Входные граничные сигналы вырабатывает локальное устройство управления. Ячейка k выполняет логические функции сравнений чисел U_j и V_i : $P^{k-1} = P^k \& (U_j^k \vee \bar{V}_i^k)$; $R^{k-1} = R^k \vee P^k \& U_j^k \& \bar{V}_i^k$. Если $P^0 \vee R^0 = 1$, то блок формирует признак $Z_j = 1$, иначе $Z_j = 0$.

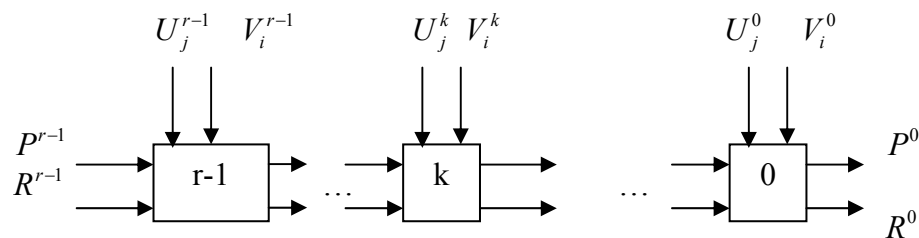


Рисунок 5 - Блок j модуля формирования признаков

Модуль выбора узла

Модуль выбора узла состоит из двух частей: модуля умножения и модуля поиска максимума. Входная память модуля умножения хранит числа $F_{i,j}$, V_i , T_j , $j=1,2,\dots,n$. Выходная память хранит числа $L_{i,j} = F_{i,j}T_j$. Операционное устройство (рис. 6) состоит из n блоков, работающих параллельно. Блок j выполняет умножение чисел $F_{i,j}$ и T_j : $L_{ij} = F_{ij} \cdot T_j$.

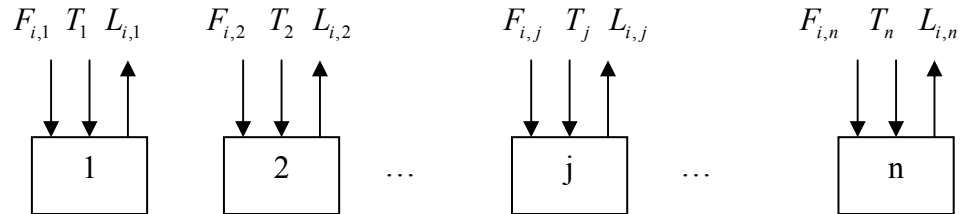


Рисунок 6 - Операционное устройство модуля умножения

Модуль поиска максимума формирует значения $X_{i,j}$: $X_{i,p}=1$, если $L_{i,p} = \max_j(L_{i,j})$ иначе $X_{i,j} = 0$. Входная память модуля поиска максимума хранит числа $L_{i,j}$. Выходной

регистр содержит вектор X_i . Операционное устройство модуля (рис. 7) содержит блок разрядной конъюнкции (БРК), блок буферной регистровой памяти (БРП), блок поиска максимума (БПМ). БРК выполняет логические функции $Z_j \& L_{i,j}^k$, $j=1,2,\dots,n$; $k=0,1,\dots,r-1$. Если $Z_j = 0$, то $L_{i,j}$ становится равным нулю. Результаты работы БРК поступают в регистры БРП и далее в БПМ. Этот блок является итеративной сетью $n \times r$ ячеек [2] (рис. 8). Ячейка на пересечении строки j и столбца k выполняет логические функции сравнения чисел $L_{i,j}$: $P_j^{k-1} = P_j^k \& (L_{i,j}^k \vee R_j^k)$; $Q_j^k = Q_j^{k-1} \vee L_{i,j}^k \& P_j^k$; $R_j^{k+1} = R_j^k$; $R_j^k = \overline{Q_j^{k+1}}$, $j=1,2,\dots,n$; $k=0,1,\dots,r-1$. Входные граничные сигналы вырабатывает локальное устройство управления: $P_1^{r-1} = 1$; $Q_1^k = 0$. Если $P_j^0 = 1$, то $X_{i,j} = 1$, иначе $X_{i,j} = 0$.

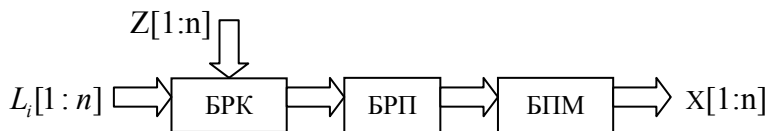


Рисунок 7 - Операционное устройство модуля поиска максимума

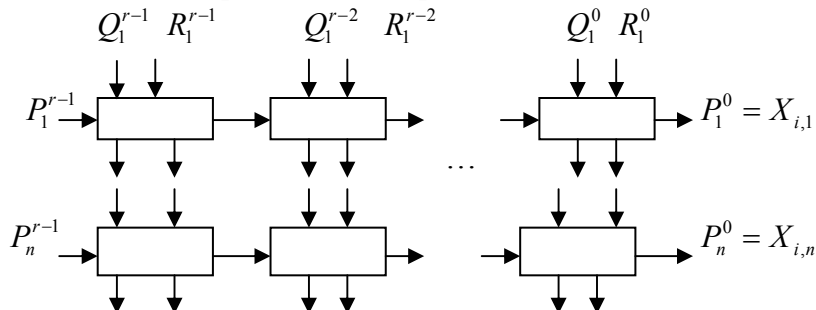


Рисунок 8 – Блок поиска максимума

Модуль заполнения узла

Входная память модуля признаков состоит из n – разрядного регистра значений $X_{i,j}$, а также r – разрядных регистров, хранящих числа U_j и V_i , $j=1,2,\dots,n$. Выходная память состоит из r – разрядных регистров, хранящих новые значения U_j . Операционное устройство (рис. 10) состоит из n блоков, работающих параллельно.

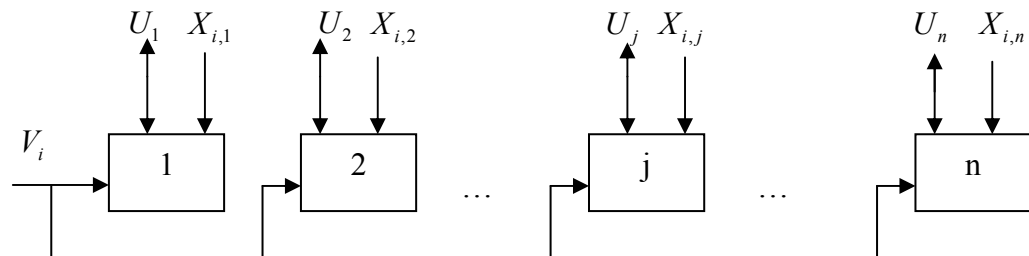


Рисунок 9 - Операционное устройство модуля заполнения узла

Блок j (рис. 10) содержит схему разрядной конъюнкции (СРК), r - разрядный буферный регистр (БР) и вычитатель r - разрядных чисел. СРК выполняет логическую функцию $V_i^k \& X_{i,j}$, где V_i^k - разряд k числа V_i . Если $X_{i,j} = 1$, то в БР и далее к вычитателю подается значение V_i . На выходе вычитателя появляется новое значение U_j : $U_j := U_j - V_i X_{i,j}$. Если $X_{i,j} = 0$, то в БР и далее к вычитателю подается нуль и значение U_j не изменяется.

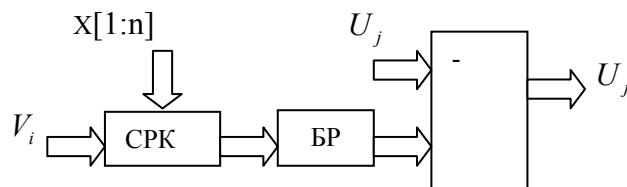
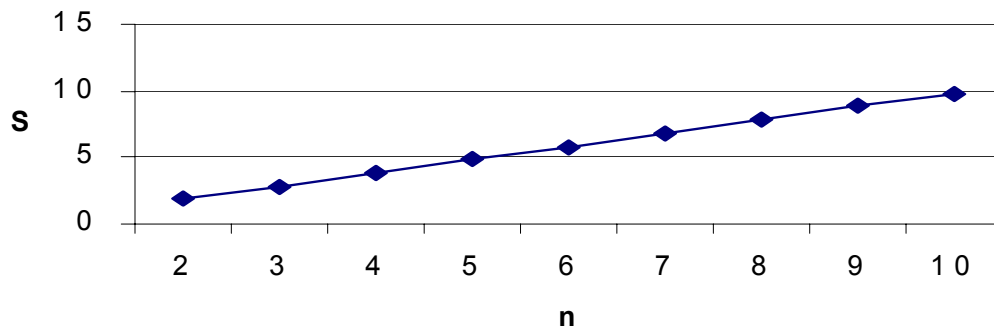
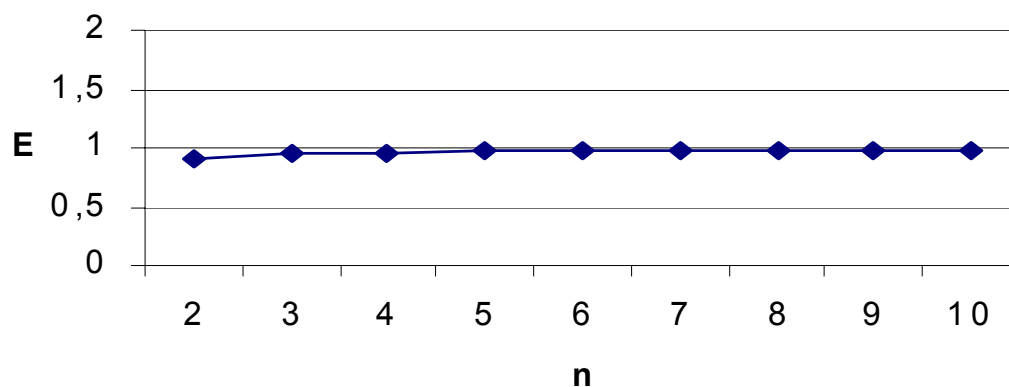


Рисунок 10 - Блок j модуля заполнения узла

Оценка эффективности спецпроцессора

Для размещения одного файла на однопроцессорной ЭВМ требуется $2n-1$ операций сравнения, $2n$ разрядных конъюнкций, n умножений и n вычитаний. Всего необходимо $6n-1$ операций. Наиболее продолжительной является операция умножения. Поэтому примем, что время выполнения операции t_{on} равно времени выполнения операции умножения. В таком случае время размещения одного файла на однопроцессорной ЭВМ: $T_1 = (6n-1)t_{on}$. Для размещения одного файла с использованием спецпроцессора требуется 2 операции сравнения, 2 разрядных конъюнкции, 1 операция умножения, 1 операция вычитания. Всего необходимо 6 операций. Поэтому время размещения одного файла с использованием спецпроцессора: $T_2 = 6t_{on}$. Обозначим: S - ускорение, получаемое при использовании спецпроцессора, E – эффективность применения спецпроцессора: $S = T_1/T_2 = (6n-1)t_{on}/6t_{on} = n-1/6$; $E = S/n = (n-1/6)/n = 1-1/6n$. Зависимости величин S и E от числа узлов сети показаны на рисунках 11, 12.

Рисунок 11 – Залежність прискорення S від числа вузлів n Рисунок 12 – Залежність ефективності E від числа вузлів n

Заключення

Предлагаемый спецпроцессор предназначен для оптимизации размещения файлов по узлам вычислительной сети в режиме реального времени. Он структурно ориентирован на процедуру размещения файла в один из узлов. Каждый модуль спецпроцессора является однородной системой блоков. Блоки работают параллельно. Каждый блок представляет собой итеративную сеть одинаковых ячеек или несколько разных итеративных сетей, связанных через буферную память. Каждый блок относится к типу SIMD и является регулярной структурой с распределенной обработкой. Работоспособность спецпроцессора проверена его функциональным моделированием в среде SIMULINK. Ускорение, получаемое при использовании спецпроцессора для решения задачи размещения файлов по узлам вычислительной сети пропорционально количеству узлов.

Литература

1. Ладыженский Ю.В., Бельков Д.В. Рациональное размещение файлов распределенной базы данных в вычислительной сети с произвольной топологией. В кн. Информатика, кибернетика и вычислительная техника (ИКВТ-99). Сборник трудов ДонГТУ, Выпуск 10. Донецк: ДонГТУ, 1999, С. 44-49.
2. Фет Я.И. Параллельные процессоры для управляющих систем. Москва: 1981–157 с.

Поступила в редакційну колегію 20.12.2002