УДК 004.7

Вычислительные структуры для размещения файлов по узлам компьютерной сети

Бельков Д.В.
Донецкий национальный технический университет
belkov65@list.ru

Бельков Д.В. Вычислительные структуры для размещения файлов по узлам компьютерной сети. *Статья предлагает метод и вычислительные структуры для оптимизации размещения файлов в компьютерных сетях. Для повышения эффективности компьютерных сетей за счет рационального размещения файлов сформулирована задача рационального размещения файлов. Для ускорения решения задачи предложены вычислительные структуры, ориентированные на ее решение.*

Ключевые слова: *Файлы, узлы компьютерной сети, специализированные процессоры для размещения файлов.*

Введение

Быстрое развитие информационных и коммуникационных технологий в современном мире привело к широкому распространению распределенных систем обработки данных на основе компьютерных сетей. Одним из способов повышения эффективности функционирования компьютерных сетей является оптимизация размещения файлов. Поэтому задача оптимального размещения файлов по узлам сети имеет важное практическое значение. Эта задача относится к классу NP - трудных. Точные методы можно применять только для решения задачи малой размерности. В случае большой размерности необходимо использовать приближенные методы.

В случае большого числа файлов программная реализация метода их распределения на однопроцессорной ЭВМ становится неэффективной, т.к. время распределения пропорционально числу файлов. По этой причине для оптимизации размещения большого числа файлов необходима разработка специализированных процессоров, позволяющих сети было минимальным.

В данной работе для уменьшения времени отклика оптимизируется интенсивность трафика. При функционировании сети в каждом узле образуются 2 типа запросов: сетевой запрос, для обработки которого необходим файл, не содержащийся в том узле, где возник запрос и локальный запрос, для обработки которого необходим файл, содержащийся в том узле, где возник запрос. Критерием оптимальности размещения файлов является суммарный поток локальных запросов, инициированных в узлах в ускорить распределение файлов за счет параллельных вычислений.

Таким образом, необходимость повышения эффективности функционирования компьютерных сетей делает актуальной разработку методов и вычислительных структур для размещения файлов в компьютерных сетях.

Цель статьи - повышение эффективности работы компьютерной сети за счет оптимизации размещения файлов с применением специализированных вычислительных структур. Задача работы - разработать вычислительные структуры для размещения файлов в компьютерной сети, позволяющие ускорить размещение большого количества файлов.

Задача размещения файлов

Для статического размещения файлов по узлам компьютерной сети необходимо при фиксированных значениях интенсивностей запросов к файлам так распределить файлы по узлам компьютерной сети, чтобы время отклика

единицу времени. Чем больше суммарный поток локальных запросов, тем меньше время отклика сети.

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; C_i - количество копий файла i; B_j - объем узла j, i=1...m, j=1…n.

Для рационального использования памяти узлов целесообразно минимизировать

объем их свободной памяти. Поэтому коэффициент заполнения узлов, равный отношению V_i / B_j необходимо максимизировать.

Задача размещения файлов по узлам компьютерной сети имеет вид:

Целевая функция

$$L = \sum_{i=1}^{m}\sum_{j=1}^{n} F_{ij}V_iX_{ij} / B_j = \sum_{i=1}^{m}\sum_{j=1}^{n} L_{ij}X_{ij} \to \max \quad (1)$$

Ограничения:

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

$$\sum_{i=1}^{m} V_iX_{ij} \le B_j \quad (3)$$

В задаче (1)-(3) необходимо найти матрицу размещений файлов X. Максимизируется суммарный поток локальных запросов. Это приводит к повышению эффективности функционирования сети за счет уменьшения среднего времени отклика на запросы к файлам.

Метод решения задачи (1)-(3), предложенный в [1], состоят из $C_i \cdot m$ этапов, C_i - количество копий файла i. 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 $L_{ij} = F_{ij}V_i / B_j$, j=1,…n. На третьем шаге файл размещается в этот узел. Исходными данными для процедуры служат значения F_{ij}, V_i, C_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 метода составляет $O(n\sum_{i=1}^{m} C_i)$, где m – количество файлов, n – число узлов, C_i - число копий файла i.

Сетевые процессоры

Сетевые процессоры являются специализированными программируемыми устройствами для обработки сетевого трафика с высокой скоростью. Сетевые процессоры применяются в сетях ATM на канальном, сетевом и транспортном уровнях стека протоколов OSI.

На сетевом уровне используется протокол IP. Единица информации, передаваемая при помощи IP, называется IP-дейтаграммой. Важной функцией IP является маршрутизация дейтаграмм. Это протокол без установки соединения, который не гарантирует сквозную передачу данных, т.е. дейтаграммы могут теряться. Две дейтаграммы, последовательно отправленные в одно и то же место назначения, маршрутизируются независимо, могут следовать по разным маршрутам и прибыть в место назначения в порядке, противоположном отправке.

Протокол UDP является простым посредником между сетевым и прикладным уровнями. Он не выполняет никаких функций по обеспечению надежности передачи, не устанавливает логического соединения, не нумерует и не упорядочивает пакеты данных. Задача транспортного уровня заключается в передаче данных между любыми прикладными процессами, выполняющимися на любых узлах сети.

За годы использования стек TCP/IP накопил большое количество протоколов и сервисов прикладного уровня. К ним относятся протокол пересылки файлов FTP, протокол эмуляции терминала TELNET, гипертекстовые сервисы доступа к удаленной информации (HTTP), сетевая файловая система (NFS), протокол SNMP для организации сетевого управления. Структура сетевого процессора, предложенная в работе [2], показана на рисунке 1.

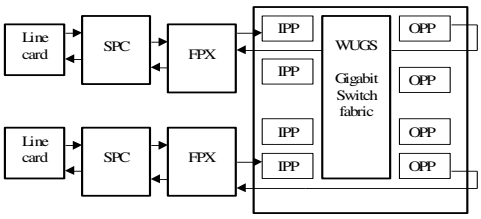

Рисунок 1 – Структура сетевого процессора

Центральным компонентом сетевого процессора является полнофункциональный ATM - переключатель (ATM-switch) WUGS. С ним соединены два дополнительных блока: smart port card (SPC) и field programmable port extender (FPX).

Блок SPC необходим для сложной обработки пакетов.

Блок FPX - это перепрограммируемая логика для пользовательских приложений. Структура блока показана на рисунке 2.

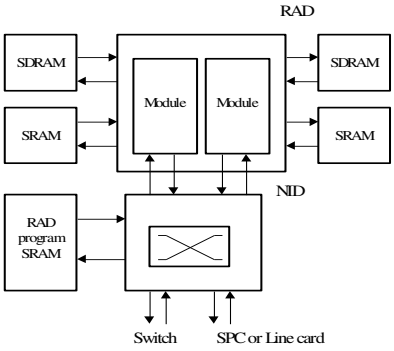

Рисунок 2 – Структура FPX

Блок содержит сетевой интерфейс (NID), реализованный на ПЛИС Xilinx XCV600E, и перепрограммируемое устройство для прикладных приложений (RAD), реализованное на ПЛИС Xilinx XCV2000E. ПЛИС NID служит для соединения RAD с WUGS, а также имеет логику для динамического перепрограммирования RAD. ПЛИС RAD содержит два программируемых пользователем модуля, которые соединены с блоками SRAM и SDRAM [4].

На рисунке 3 показана концепция сетевых процессоров (network wrapper concept), предложенная в работе [3].

Рисунок 3 - Концепция сетевых процессоров

Cell-процессор получает пакеты (cells), обрабатывает ATM – заголовки и распределяет пакеты по виртуальным каналам. В сетях ATM пакеты имеют фиксированный размер. Для использования данных произвольного размера фрейм – процессор организует пакеты в кадры [4].

IP–процессор необходим для поддержки протокола IP. Маршрутизатор, построенный на основе IP – процессора, поддерживает таблицу маршрутизации, которая применяется для нахождения следующего маршрутизатора или хоста. UDP–процессор необходим для поддержки протокола UDP.

В концепции сетевых процессоров, предложенной в работе [4], прикладной уровень не рассматривается. Однако на прикладном уровне можно повысить эффективность функционирования сети за счет оптимизации времени отклика. Оптимизация должна осуществляться в режиме реального времени, поэтому требуется ее аппаратная поддержка. В статье предлагается расширить концепцию сетевых процессоров на прикладной уровень за счет использования спецпроцессора для оптимизации размещения файлов. Он должен быть ориентирован на решение задачи (1)-(3).

Спецпроцессор для размещения файлов

Задачу оптимизации размещения файлов по узлам вычислительной сети нужно решать в режиме реального времени. Поэтому необходима аппаратная реализация метода ее решения. На рисунке 4 показана структура вычислительной 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 узел. На четвертом шаге спецпроцессор передает результаты работы процедуры в выходной регистр. На пятом шаге основной процессор считывает эти результаты из буферной памяти.

Рисунок 4 – Структура вычислительной системы

Входные данные являются r – разрядными целыми числами без знака. Выходные данные - строка i матрицы X, $X_{ij} \in \{0;1\}$. Входные данные размещены в 2(n+1) регистрах разрядностью r, выходные данные размещены в регистре разрядностью n. На первом шаге метода $U_j = B_j$, j=1,2,…,n. На четвертом шаге метода обновляются значения

U_j во входных регистрах, и заполняется выходной регистр.

Структура спецпроцессора для размещения файлов показана на рисунке 5. В отличие от структуры, приведенной в работах [5,6], она позволяет решать задачу, в которой файлы могут иметь несколько копий. Описание модулей структуры дано в следующем разделе.

Каждый модуль построен на основе специализированного однородного процессора. Такой процессор является итеративной сетью, ориентированной на выполнение заданной базовой операции. Итеративная сеть есть логическая сеть, состоящая из одинаковых и одинаково связанных между собой элементов (ячеек). Особенностью итеративных сетей является то, что в них осуществляется прямое отображение алгоритмов в схемы. За счет прямого моделирования алгоритмов, итеративные сети позволяют синтезировать безызбыточные цифровые устройства высокой производительности. Специализированные однородные процессоры реализуют базовые операции структурно за один такт и относятся к компьютерам типа SIMD с крупноблочными базовыми операциями [7,8]. Процессоры обладают регулярной структурой и удобны для реализации в базисе ПЛИС

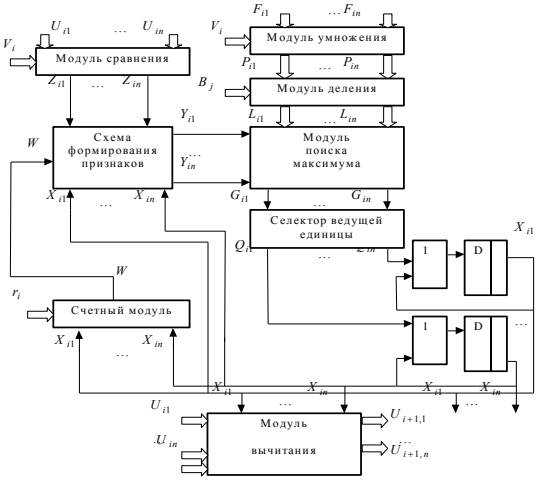

Рисунок 5 – Структура спецпроцессора

Процесс формирования элементов строки X_i происходит следующим образом. На первом шаге значения X_{ij} равны нулю, $Z_{ij}=1$, если $U_{ij}\geq V_i$, иначе $Z_{ij}=0$. На втором шаге осуществляется поиск максимального значения среди значений L_{ij}, таких, что $Z_{ij}=1$. В результате этого, одно из значений X_{ij} устанавливается в единицу.

Модули спецпроцессора

Модуль сравнения формирует элементы строки Z_i: $Z_{ij}=1$, если файл i может поместиться в узел j, иначе $Z_{ij}=0$. Модуль имеет локальную входную и выходную память, операционное устройство и устройство управления. Входная память модуля сравнения состоит из r – разрядных регистров, хранящих числа U_j и V_i, j=1,2,…,n. Выходная память представляет собой n-разрядный регистр элементов вектора Z_i: $Z_{i1},Z_{i2},\ldots,Z_{in}$. Операционное устройство, показанное на рисунке 6, состоит из n блоков, работающих параллельно. Блок j, показанный на рисунке 7, формирует элемент Z_{ij}: $Z_{ij}=1$, если $U_{ij}\geq V_i$, иначе $Z_{ij}=0$. Он представляет собой итеративную сеть r ячеек [8].

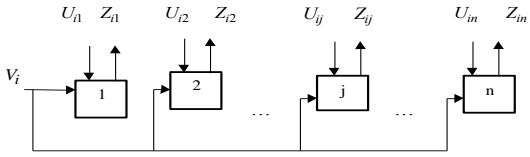

Рисунок 6 - Операционное устройство модуля сравнения

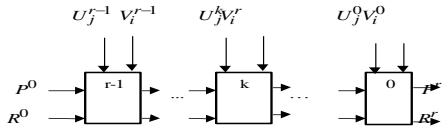

Рисунок 7 – Блок j модуля сравнения

На рисунке 8 обозначено: U_{ij}^k - разряд k числа U_{ij}; V_i^k - разряд k числа V_i; P^r, R^{r-1} - входные граничные сигналы: $P^r=1$, $R^r=0$; P^0, R^0 - выходные граничные сигналы.

Входные граничные сигналы вырабатывает локальное устройство управления. Ячейка k выполняет логические функции сравнения чисел U_{ij} и V_i: $P^{k-1}=P^k\,\&\,(U_{ij}^k\vee\overline{V}_i)$; $R^{k-1}=R^k\vee P^k\,\&\,U_{ij}^k\,\&\,\overline{V}_i^k$; если $P^0\vee R^0=1$, то блок формирует признак $Z_{ij}=1$, иначе $Z_{ij}=0$.

Модуль умножения формирует элементы вектора P_i: $P_{ij}=F_{ij}\cdot V_i$. Модуль имеет локальную входную и выходную память, операционное устройство и устройство управления. Входная память модуля умножения хранит числа F_{ij} и V_i, j=1,2,…,n. Выходная память хранит произведения $P_{ij}=F_{ij}\cdot V_i$.

Операционное устройство, показанное на рисунке 8, состоит из n блоков, работающих параллельно. Блок j является матричным однотактным умножителем [8]. Он выполняет умножение чисел F_{ij} и V_i по методу умножения, начиная со старших разрядов множителя, со сдвигом множимого вправо.

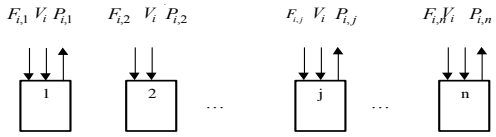

Рисунок 8 – Операционное устройство модуля умножения

Модуль деления формирует элементы вектора L_i: $L_{ij} = P_{ij} / B_j$. Модуль имеет локальную входную и выходную память, операционное устройство и устройство управления. Входная память модуля деления хранит числа P_{ij} и B_j, j=1,2,…,n. Выходная память хранит числа $L_{ij} = P_{ij} / B_j = F_{ij} \cdot V_i / B_j$. Операционное устройство модуля деления, показанное на рисунке 9, состоит из n блоков, работающих параллельно. Блок j является матричным однотактным устройством деления на основе итеративной сети [9]. Он выполняет целочисленное деление числа P_{ij} на число B_{ij} по методу деления с незавершением (nonperforming division).

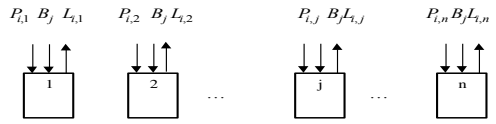

Рисунок 9 – Операционное устройство модуля деления

Счетный модуль используется для подсчета количества размещенных копий файла и формирования сигнала W, управляющего формированием признаков Y_{ij}. Входная память модуля хранит значение C_i. Структура операционного устройства счетного модуля показана на рисунке 10. Она имеет элемент ИЛИ, счетчик СТ, схему сравнения, триггер.

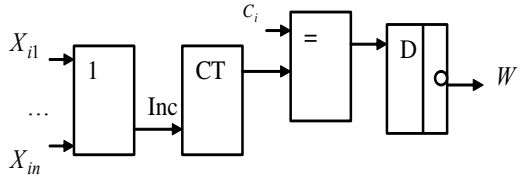

Рисунок 10 - Операционное устройство счетного модуля

Схема формирования признаков показана на рисунке 11. Она вырабатывает сигналы Y_{ij}: $Y_{ij} = \neg(W \,\&\, X_{ij}) \,\&\, Z_{ij} \,\&\, RD$, j=1,2,…,n. Значение $Y_{ij} = 0$ есть признак того, что узел j запрещен для размещения файла i.

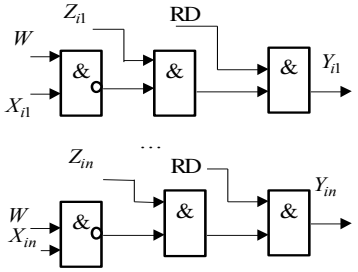

Рисунок 11 – Схема формирования признаков

Модуль поиска максимума (МПМ) используется для определения узлов с наибольшим значением L_{ij}. Модуль имеет локальную входную и выходную память, операционное устройство и устройство управления. Входная память модуля хранит числа L_{ij}. Операционное устройство модуля поиска максимума является итеративной сетью $n \times r$ ячеек [8], показанной на рисунке 12. Ячейка на пересечении строки j и столбца k выполняет логические функции сравнения L_{ij}: $P_j^{k-1} = P_j^k \,\&\, (L_{i,j}^k \vee R^k{}_j)$; $Q^k{}_{j+1} = Q^k{}_j \vee L_{i,j}^k \,\&\, P_j^k$ $R^k{}_{j+1} = R^k{}_j$; $R^k{}_1 = \overline{Q}^k{}_{n+1}$, j=1,2,…,n; k=0,1,…,r-1. Входные граничные сигналы вырабатывает локальное устройство управления: $P_j^r = 1$; $Q_1^k = 0$. Если $P_j^0 = 1$, то $X_{ij} = 1$, иначе $X_{ij} = 0$.

Для выделения первого среди наибольших значений L_{ij} используется селектор ведущей единицы (СВЕ). Он является итеративной сетью из n ячеек [10], показанной на рисунке 13. Ячейка j выполняет функции: $Y_j = P_j^0$, $X_{ij} = P_j^0 \,\&\, \overline{Y}_{j+1}$.

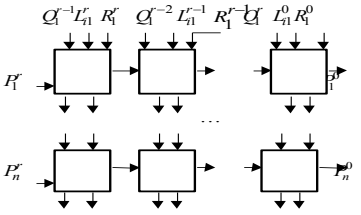

Рисунок 12 – Операционное устройство модуля поиска максимума

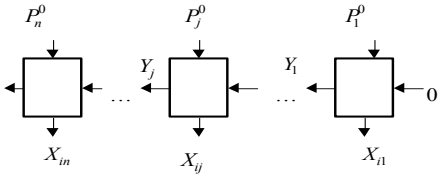

Рисунок 13 – Структура селектора ведущей единицы

Счетный модуль, схема формирования признаков, модуль поиска максимума и селектор ведущей единицы предназначены для выполнения процедуры формирования строки X_i: $X_{ij}=1$, если файл i должен размещаться в узле j, иначе $X_{ij}=0$.

Процесс формирования элементов вектора X_i происходит следующим образом. На первом шаге значения X_{ij} равны нулю, $Y_{ij}=1$ и счетчик СТ установлен в нуль. На втором шаге содержимое счетчика сравнивается с C_i. Если они не равны между собой, вырабатывается сигнал *W*. С его помощью на третьем шаге формируются новые значения Y_{ij}: $Y_{ij}=\neg(W \,\&\, X_{ij}) \,\&\, Z_{ij} \,\&\, RD$. На четвертом шаге осуществляется поиск первого максимального значения среди значений L_{ij}, таких, что $Y_{ij}=1$. В результате этого, одно из значений X_{ij} устанавливается в единицу. На пятом шаге содержимое счетчика увеличивается на единицу, и процесс повторяется с шага 2 до тех пор, пока $W=1$. Для того чтобы определенные на данном шаге значения X_{ij} не потерялись на следующем шаге, при их сохранении используется блок элементов ИЛИ. Если все значения X_{ij} сформированы, то сигнал *W* устанавливается в ноль. Структура для формирования строки X_i показана на рисунке 14.

Модуль вычитания формирует элементы вектора U_{i+1}: $U_{i+1,j}=U_{ij}-V_iX_{ij}$. Модуль имеет локальную входную и выходную память, операционное устройство и устройство управления. Операционное устройство модуля вычитания показано на рисунке 15. Оно состоит из n блоков, работающих параллельно. Входная память модуля состоит из n – разрядного регистра значений X_{ij}, а также r – разрядных регистров, хранящих числа U_{ij} и V_i, j=1,2,…,n. Выходная память состоит из r – разрядных регистров, хранящих значения $U_{i+1,j}$.

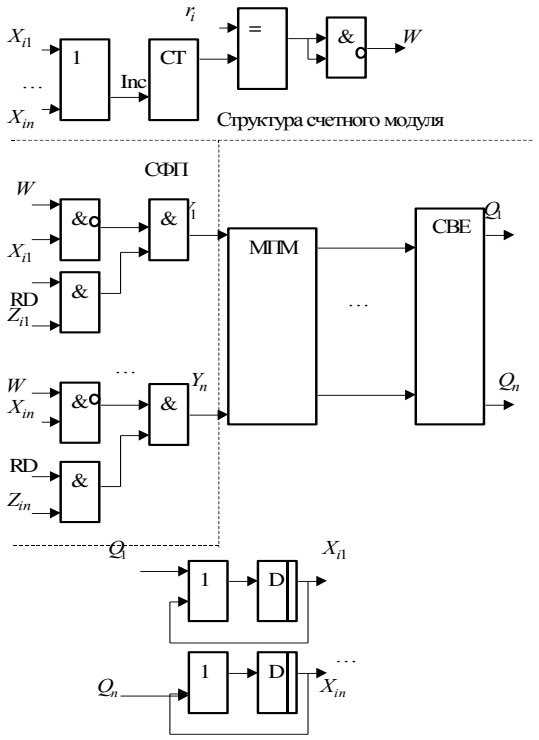

Рисунок 14 - Структура для формирования строки X_i

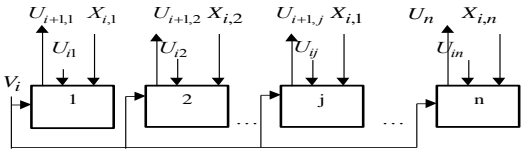

Рисунок 15 - Операционное устройство модуля вычитания

Блок j, показанный на рисунке 17, содержит схему поразрядной конъюнкции (СРК) и вычитатель r-разрядных чисел. СРК выполняет логическую функцию $V_i^k \,\&\, X_{i,j}$, где V_i^k - разряд k числа V_i. Блок работает следующим образом. Если $X_{ij}=1$, то к вычитателю подается значение V_i. На выходе вычитателя формируется значение $U_{i+1,j}$: $U_{i+1,j}=U_{ij}-V_iX_{ij}$. Если $X_{ij}=0$, то вместо значения V_i к вычитателю подается ноль и значение U_{ij} не изменяется.

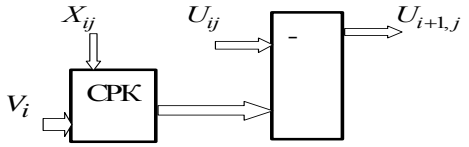

Рисунок 17 - Блок j модуля вычитания

Оценка эффективности спецпроцессора

Величина коэффициента эффективности спецпроцессора зависит от конкретной задачи размещения файлов. Обозначим: t_{cp} - время необходимое на одну операцию сравнения. Для размещения одного файла на однопроцессорном компьютере требуется в лучшем случае *n+1* операций сравнения. Время необходимое на все операции сравнения в лучшем случае: $T_1^{л} = (n+1)t_{cp}$. В худшем случае для размещения одного файла требуется *2n-1* операций сравнения. Время необходимое на все операции сравнения в худшем случае: $T_1^{x} = (2n-1)t_{cp}$. Для размещения одного файла с помощью разработанной структуры спецпроцессора требуется 2 операции сравнения в любом случае: $T_2 = 2t_{cp}$. Минимальное значение коэффициента ускорения от использования спецпроцессора: $S_{\min} = T_1^{л} / T_2 = (n+1)t_{cp} / 2t_{cp} = (n+1)/2$.

Максимальное значение коэффициента ускорения от использования спецпроцессора: $S_{\max} = T_1^{x} / T_2 = (2n-1)t_{cp} / 2t_{cp} = n-1/2$.

Ускорение решения задачи размещения файлов при использовании спецпроцессора пропорционально количеству узлов компьютерной сети.

Выводы

Статья посвящена повышению эффективности работы компьютерной сети за счет оптимизации размещения файлов с применением специализированных структур. Для ускорения размещения файлов по узлам компьютерной сети предложена структура спецпроцессора, ориентированная на решение этой задачи.

Литература

1. Бельков Д.В. Методы и вычислительные структуры для оптимизации размещения файлов в компьютерных сетях. Автореферат диссертации. Донецк: 2004. – 20 с.
2. F. Braun, J. Lockwood, M. Waldvogel. Protocol wrappers for layered network packet processing in reconfigurable hardware. //IEEE Micro - 2002. - № 1. - P. 12 – 19.
3. Braun F., Lockwood J., Waldvogel M. Layered protocol wrappers for Internet packet processing in reconfigurable hardware. //www.arl.wustl.edu/arl/projects/fpx/
4. Lockwood J., Naufel N., Turner J. S., Taylor D. E. Programmable network packet processing on the Field Programmable Port Extender (FPX). //www.arl.wustl.edu/arl/projects/fpx/
5. Ладыженский Ю.В., Бельков Д.В. Структура спецпроцессора для оптимизации размещения файлов по узлам вычислительной сети. // Зб. Наукових праць ДонНТУ. Серія “Проблеми моделювання та автоматизації проектування динамічних систем”. Вип. 52: - Донецьк: ДонНТУ, 2002. - С. 119-124.
6. Деклараційний патент на винахід № 59736А, G 06 F 9/46 /Бєльков Д.В., Ладиженський Ю.В., Пристрій для розподілу файлів серед комп’ютерів. Заявл. 29.11.2002. Опубл. 15.09.2003, Бюл. № 9.
7. Огнев И.В., Борисов В.В. Ассоциативные среды. М.: Радио и связь, 2000.– 309 с.
8. Фет Я.И. Параллельные процессоры для управляющих систем. М.: Радио и связь, 1981.–157 с.
9. Карцев М.А., Брик В.А. Вычислительные системы и синхронная арифметика. М.: Радио и связь, 1981.- 358 с.
10. Бандман О.Л. Специализированные процессоры для высокопроизводительной обработки данных. Новосибирск: Сибирское отделение АН СССР, 1988. – 289 с.

Бельков Д.В. Вычислительные структуры для размещения файлов по узлам компьютерной сети. *Статья предлагает метод и вычислительные структуры для оптимизации размещения файлов в компьютерных сетях. Для повышения эффективности компьютерных сетей за счет рационального размещения файлов сформулирована задача рационального размещения файлов. Для ускорения решения задачи предложены вычислительные структуры, ориентированные на ее решение.*
Ключевые слова: *Файлы, узлы компьютерной сети, специализированные процессоры для размещения файлов.*

Belkov D.V. Methods and computer structures for optimization of file allocation in the computer networks. *The article presents the method and computer structures for optimization of file allocations in computer networks. For a rise of efficiency of computer network due to the rational file allocations, the task of the rational file allocations on nodes of computer network is formulated. Heuristic method for solving of this task is proposed. For acceleration of solving of task of file allocations on nodes of computer network, structures of special processors oriented on the solving of this task are proposed.*
Keywords: *Files, nodes of computer network, special processors for file allocation.*

Статья поступила в редакцию 20.09.2017
Рекомендована к публикации д-ром техн. наук В.Н. Павлышом