УДК 681.3:007

ПАРАЛЛЕЛЬНОЕ ПОСТРОЕНИЕ ДЕКАРТОВА ДЕРЕВА С ЛОГАРИФМИЧЕСКОЙ ОЦЕНКОЙ ВРЕМЕННОЙ СЛОЖНОСТИ

Ромм Я.Е., Чабанюк Д.А.

Таганрогский институт имени А.П. Чехова (филиал) РГЭУ (РИНХ), Таганрог, Россия (347936, Ростовская область, г. Таганрог, ул. Инициативная, 48), e-mail: romm@list.ru

В статье выполнен синтез параллельного алгоритма построения декартова дерева с применением максимально параллельной модификации сортировки подсчетом на основе матриц сравнений. При параллельном построении поддеревьев используются взаимно независимые сравнения элементов по типу отдельных строк таких матриц, что аналогично операциям сортировки Хоара в параллельной форме. Для множества N пар элементов (X_i, Y_i) временная сложность построения декартова дерева оценивается из соотношения $T(R) = O(\log_2 N)$, где число процессоров $R = (N^2 - N)/2$. Оценка получена на модели неветвящихся параллельных программ без учета операций обмена. В статье приводятся примеры использования параллельной сортировки, а также пошаговой интерпретации работы параллельного алгоритма построения декартова дерева в процессе определения корней и ветвей поддеревьев. Предложенный алгоритм распараллеливается на уровне алгоритмических и разрядных операций.

Ключевые слова: алгоритмы параллельных сортировок, структуры данных, декартово дерево, поразрядно-параллельное сравнение слов.

PARALLEL TREAP CONSTRUCTING WITH LOGARITHMIC ESTIMATE TIME COMPLEXITY

Romm Y.E., Chabanyuk D.A.

Taganrog Institute of Chekhov A.P. (branch) RGEU (RINH), Taganrog, Russia, (347936, Rostovskaya obl., Taganrog, Iniciativnaya str., 48), e-mail: romm@list.ru

In this article gives the synthesis of a parallel algorithm for constructing of treap with use the modification counting sorting in the maximum parallel form on the basis of the comparison matrix. At the parallel construction of subtrees of treap used mutually independent comparisons of the elements according to the type of individual strings of the matrix, which is similar to the Hoare sorting in parallel form. For a set of N pairs of elements of (X_i, Y_i) is the time complexity of construction of treap is estimated from the ratio of $T(R) = O(\log_2 N)$, where the number of processors $R = (N^2 - N)/2$. Estimate was obtained on the straight-line model of parallel programs without taking into account operations of the exchange. In this article gives examples of the use of parallel sorting, and made step by step interpretation of the parallel algorithm for constructing treap in the determination of the roots and branches of constructing subtrees of treap. The proposed algorithm is parallelized at the level of algorithmic operations and bit operations.

Keywords: parallel sorting algorithms, data structures, treap, bit by bit parallel comparison of words.

Постановка вопроса. Ускоряющаяся компьютеризация всех областей деятельности и информационных процессов, значительный рост объемов обрабатываемой информации приводит к трудностям при последовательном выполнении информационного поиска на основе традиционных алгоритмов [1, 3]. Древовидные структуры характеризуются эффективностью представления динамических данных для быстрого поиска информации. С целью дальнейшего увеличения эффективности целесообразна разработка и исследование алгоритмов параллельного построения древовидных структур данных. В таком аспекте ставится задача синтеза и оценки временной сложности параллельного алгоритма построения декартова дерева. Параллельные алгоритмы рассматриваются на модели

неветвящихся параллельных программ, согласно которой временная сложность $T(R)$ алгоритма (кратко – время выполнения) измеряется количеством последовательных шагов без учета обмена, R – число процессоров [9].

Сортировка подсчетом. На этапе построения декартова дерева ко всем компонентам будет применяться максимально параллельная сортировка подсчетом по матрице сравнений [5]. Модификация известного алгоритма заключается в следующем. Для одномерного массива $a=(a_1, a_2, \ldots, a_n)$ сортируемых элементов составляется матрица сравнений, элемент $\alpha_{i,j}$ которой определяется из соотношений:

$$\alpha_{i,j} = sign\,(a_j - a_i) = \begin{cases} 1(+), & a_j > a_i; \\ 0, & a_j = a_i; \\ -1(-), & a_j < a_i; \quad i=\overline{1,n};\ j=\overline{1,n}. \end{cases}$$

Входной элемент с номером j получает номер k в отсортированном массиве по правилу: в j-м столбце матрицы подсчитывается количество нулей и плюсов сверху вниз до главной диагонали включительно и складывается с количеством только плюсов ниже диагонали в этом же столбце. Суммарное количество составит значение выходного номера k. Пусть, например, требуется отсортировать по неубыванию массив $a=(1, -1, 0, 3, -3, 5)$.

Матрица сравнений примет вид:

		a_1	a_2	a_3	a_4	a_5	a_6
		1	–1	0	3	–3	5
a_1	1	0	–	–	+	–	+
a_2	–	+	0	+	+	–	+
a_3	0	+	–	0	+	–	+
a_4	3	–	–	–	+	–	+
a_5	–	+	+	+	0	0	+
a_6	5	–	–	–	–	–	0

Согласно правилу вставки получится: $a_1=1 \to c_{(+++0)} \to c_4$; $a_2=-1 \to c_{(0+)} \to c_2$; $a_3=0 \to c_{(+0+)} \to c_3$; $a_4=3 \to c_{(+++0+)} \to c_5$; $a_5=-3 \to c_{(0)} \to c_1$; $a_6=5 \to c_{(+++++0)} \to c_6$. Отсортированный массив примет вид: $c=(-3, -1, 0, 1, 3, 5)$. Процедура сортировки в консольном приложении Delphi [3, 5] при соответственном задании типа элементов и их индексов запишется в виде:

```
procedure sort (var n: integer; var a,c: vekt00; var e: vekt);
var i,j,k: integer;
begin
for j:= 1 to n do
    begin
    k:= 0;
```

```
    for i:= 1 to j do
    if a[j] >= a[i] then k:= k+1;
    for i := j+1 to n do
    if a[j] > a[i] then k := k+1;
    c[k] := a[j]; e[k] := j;
    end;
end;
```

Сортировка сохраняет порядок равных элементов, операторы c[k]:=a[j]; e[k]:=j; в явном виде задают взаимно однозначное соответствие входных и выходных индексов сортируемых элементов. Сортировка обладает максимальным параллелизмом: все сравнения в матрице взаимно независимы и выполнимы параллельно за время $T\left(\left(N^2 - N\right)/2\right) = O(1)$ [5].

Описание метода. Декартово дерево – это структура данных, хранящая в узлах пары элементов (X, Y) в виде двоичного дерева таким образом, что она является двоичным деревом по X и двоичной пирамидой по Y [1]. Двоичное дерево – это структура данных, для которой у левого потомка произвольной вершины значение элемента X меньше либо равно значения X его вершины, а у правого потомка произвольной вершины значение X строго больше, чем значение X его вершины. Двоичная пирамида – двоичное дерево, в каждой вершине которой хранится ключ, и для каждой вершины дерева с ключом Y у всех непосредственных потомков ключ не больше значения Y предка. Пусть задано некоторое множество пар элементов (X_i, Y_i). Корнем декартова дерева является пара элементов, которая имеет максимальный Y (по условию двоичной пирамиды). Пусть (X_i, Y_i) – пара с максимальным Y. Тогда все пары с меньшими X будут в левом поддереве, остальные – в правом поддереве. Каждое из поддеревьев строится аналогично. На рис. 1 изображен пример декартова дерева для массива пар чисел:

(1; 4), (1; 12), (4; 13), (9; 14), (4; 7), (6; 8), (7; 6), (12; 5), (13; 11), (14; 12), (18; 0). (1)

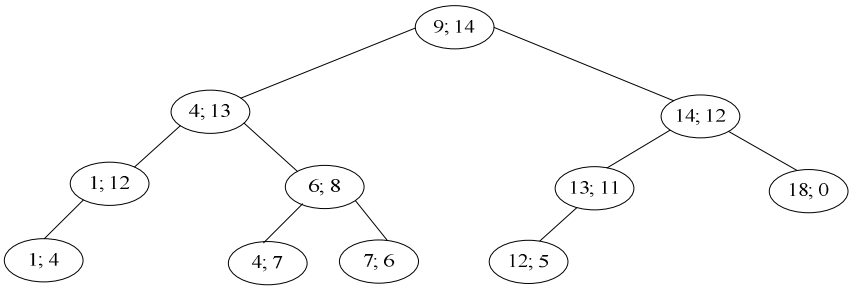

Рис. 1. Пример декартова дерева

Построение декартова дерева с применением представленной максимально-параллельной сортировки (ниже МП-сортировки) выполняется следующим образом. Ко всем компонентам Y_i – элементов данного множества применяется МП-сортировка по неубыванию. При этом пары уже отсортированных по Y_i элементов (X_i, Y_i) располагаются с сохранением входного индекса без изменения единичной оценки времени. Наибольший среди Y_i элемент

окажется в правом конце отсортированного массива вместе с входным индексом пары (X_i, Y_i). Пусть эта пара обозначена $(X_i, Y_i)_N$, а ее входной индекс – E_N, $E_N = i$, таким образом, определен корень декартова дерева $(X_i, Y_i)_N$. Далее, без изменения его местоположения во входном массиве, которое идентифицируется по индексу E_N, этот элемент объявляется серединным в данном входном массиве. Относительно середины выполняется упорядочение по X_i: все элементы в левой части массива, которые меньше X_i из $(X_i, Y_i)_N$, остаются в левом подмассиве с сохранением исходного взаимного порядка; все элементы в левой части массива, которые больше X_i из $(X_i, Y_i)_N$, переносятся из левой части в правый подмассив, также с сохранением исходного взаимного порядка. Все элементы правой части массива, которые больше X_i из $(X_i, Y_i)_N$, остаются в правом подмассиве с сохранением исходного взаимного порядка; все элементы правой части массива, которые меньше X_i из $(X_i, Y_i)_N$, переносятся из правой части в левый подмассив, также с сохранением исходного взаимного порядка.

Излагаемый ниже способ упорядочения относительно середины аналогичен тому, который применяется для параллельного варианта сортировки Хоара [4].

Для примера (1) середина – пара $(X_i, Y_i)_N = (9; 14)$, $i = 4$. Упорядочение относительно $X_i = 9$ выполняется параллельно по всем элементам по схеме МП-сортировки в одной строке:

		a_1	a_2	a_3	$\boldsymbol{a_4}$	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}
		1	1	4	***9***	4	6	7	12	13	14	18
$\boldsymbol{a_4}$	***9***	–	–	–	0	–	–	–	+	+	+	+

Все элементы со знаком сравнения "–" из правого подмассива с сохранением индексов переносятся в левый подмассив. В результате получится:

		a_1	a_2	a_3	a_5	a_6	a_7	$\boldsymbol{a_4}$	a_8	a_9	a_{10}	a_{11}
		1	1	4	4	6	7	***9***	12	13	14	18
$\boldsymbol{a_4}$	***9***	–	–	–	–	–	–	0	+	+	+	+

Для левого подмассива элементы декартова дерева расположатся в виде:

a_1	a_2	a_3	a_5	a_6	a_7
(1; 4)	(1;12)	(4; 13)	(4; 7)	(6; 8)	(7; 6)

В сформированном таким способом левом подмассиве повторяется МП-сортировка по Y_i. В результате определяется корень поддерева в этом подмассиве: (4; 13) с исходным индексом $i = 3$. Относительно его положения в подмассиве выполняется упорядочение по изложенной схеме по X_i:

		a_1	a_2	$\boldsymbol{a_3}$	a_5	a_6	a_7
		1	1	***4***	4	6	7
$\boldsymbol{a_3}$	***4***	–	–	0	0	+	+

В результате определились элементы нового поддерева искомого декартова дерева с корнем (4; 13):

a_1	a_2	$\boldsymbol{a_3}$	a_5	a_6	a_7
(1; 4)	(1;12)	**(4; 13)**	(4; 7)	(6; 8)	(7; 6)

Сравнение с серединным элементом выполняется параллельно по всем элементам исходного подмассива аналогично одной строке МП-сортировки за единичное время на процессорах, число которых равно числу N_i элементов подмассива. Число процессоров для определения корня поддерева равно $R_i = \frac{1}{2}(N_i)^2(N_i - 1)$.

Параллельно с левым подмассивом исходного массива аналогично обрабатывается правый подмассив:

a_8	a_9	a_{10}	a_{11}
(12; 5)	(13; 11)	(14; 12)	(18; 0)

Повторяется МП-сортировка по Y_i: в результате определяется корень поддерева: (14; 12) с исходным индексом $i = 10$. Относительно его исходного положения в данном правом подмассиве выполняется упорядочение по изложенной схеме относительно X_i:

		a_8	a_9	$\boldsymbol{a_{10}}$	a_{11}
		12	13	***14***	18
$\boldsymbol{a_{10}}$	***14***	–	–	0	+

Тем самым, оказалось, что окончательно сформировано поддерево правого подмассива:

a_8	a_9	$\boldsymbol{a_{10}}$	a_{11}
(12; 5)	(13; 11)	**(14; 12)**	(18; 0)

На этом завершено полное построение декартова дерева, оно совпадает с деревом, изображенным на рис. 1.

В общем случае описанные действия параллельно повторяются отдельно в каждом подмассиве, при этом наибольший среди Y_i элемент левого подмассива окажется левым потомком корня декартова дерева $(X_i, Y_i)_N$, наибольший среди Y_i элемент правого подмассива окажется правым его потомком.

Действия воспроизводятся в каждом новом подмассиве, образуемом потомком как корнем поддерева, причем параллельно по всем подмассивам, соответственным корням текущего уровня.

Шаги описанного параллельного алгоритма продолжаются до полного построения декартова дерева, их количество, очевидно, не превосходит $\log_2 N$. Число процессоров в каждом поддереве для МП-сортировки по уровням корней располагается в последовательность:

$$\frac{N^2-N}{2},\ \frac{N_1^2-N_1}{2}+\frac{N_2^2-N_2}{2},\ \frac{N_3^2-N_3}{2}+\frac{N_4^2-N_4}{2}+\frac{N_5^2-N_5}{2}+\frac{N_6^2-N_6}{2},\ldots \qquad (2)$$

В (2) нечетные индексы соответствуют поддеревьям левых подмассивов, четные – правых. С учетом $N_2 = N - N_1$ получится, что

$$\frac{N_1^2-N_1}{2}+\frac{N_2^2-N_2}{2}=\frac{N_1^2-N_1}{2}+\frac{(N-N_1)^2-(N-N_1)}{2}\le\frac{N^2-N}{2}\ .$$

В общем случае, соответственно уровню корней поддеревьев с номером $\log_2 k$, $k \le N$, количество сортируемых по всем подмассивам элементов удовлетворяет равенству $\sum_{i=1}^{\log_2 k} N_i = N$. Отсюда количество процессоров при параллельном по всем i выполнении МП-сортировки каждого подмассива из N_i элементов на уровне с номером $\log_2 k$ удовлетворяет соотношениям:

$$\sum_{i=1}^{\log_2 k} R_i = \frac{1}{2}\sum_{i=1}^{\log_2 k} N_i\left(N_i-1\right)\le\frac{1}{2}N\sum_{i=1}^{\log_2 k}\left(N_i-1\right),$$

поэтому

$$\sum_{i=1}^{\log_2 k} R_i \le \frac{1}{2}N\left(\sum_{i=1}^{\log_2 k} N_i - 1\right)=\frac{N^2-N}{2}.$$

Таким образом, с учетом единичного времени каждого последовательного шага и числа шагов $\log_2 N$ получается оценка временной сложности предложенного алгоритма:

$$T\left(\left(N^2-N\right)/2\right)=O(\log_2 N)\,. \qquad (3)$$

Из изложенного вытекает

Теорема 1. Построение декартова дерева может быть выполнено с помощью детерминированного параллельного алгоритма за логарифмическое число шагов с временной сложностью (3), где N – число элементов входного множества пар (X_i, Y_i).

Очевидной является возможность перемещения входных индексов пар (X_i, Y_i) по уровням и подмассивам в соответствии шагам изложенного построения декартова дерева, что дает возможность адресоваться от элементов окончательно построенного декартова дерева непосредственно к элементам входного массива и обратно.

Замечание 1. В [6-8] рассматривается построение декартова дерева с учетом поразрядно-параллельных сравнений элементов без вычисления переноса на основе аналога арифметической обработки [2]. Учитывая время МП-сортировки $T\left(\left(N^2-N\right)/2\right)=O(1)$ и оценку (3), полное построение декартова дерева в максимально параллельной форме на данной основе можно выполнить с временной сложностью $T(2N^2n)=O(\log_2 N)\tau_0$, где N – число элементов входного множества пар (X_i,Y_i), n – число разрядов наибольшего по числу символов элемента в наборе, τ_0 – время одного сравнения двух бит.

В [6] даны результаты программного эксперимента с последовательным моделированием параллельного алгоритма построения декартова дерева, на выходе которого для произвольного множества пар входных данных всегда получается правильный результат.

Заключение. В статье изложен синтез алгоритма параллельного построения декартова дерева с применением максимально-параллельной сортировки подсчетом и аналога сортировки Хоара в параллельной форме. Временная сложность построения декартова дерева по данному алгоритму имеет оценку вида $T\left(\left(N^2-N\right)/2\right)=O(\log_2 N)$. Предложенный алгоритм распараллеливается на уровне алгоритмических и разрядных операций, при этом достигает снижения оценки временной сложности известного аналога [10] построения декартова дерева. Приведены примеры использования сортировки и построения декартова дерева с помощью представленного алгоритма, правильность работы которого подтверждается программными экспериментами [6]. Представленный в статье подход может рассматриваться в качестве основы для увеличения эффективности параллельного построения древовидных структур данных.

Список литературы

1. Кнут Д. Искусство программирования для ЭВМ. Т.3. Сортировка и поиск. – М.: Вильямс, 2007. – 832 с.

2. Ромм Я.Е. Метод вертикальной обработки потока целочисленных групповых данных. II. Приложение к бинарным арифметическим операциям // Кибернетика и системный анализ. 1998. - № 6. – С. 146-162.

3. Ромм Я.Е., Белоконова С.С. Детерминированный информационный поиск на основе сортировки с распараллеливанием базовых операций. – М.: Научный мир, 2014. – 198 с.

4. Ромм Я.Е., Виноградский В.В. Преобразование сортировки Хоара в параллельную форму на основе матриц сравнений // Проблемы программирования. – 2008. - № 2-3. – С. 332-340.

5. Ромм Я.Е., Заика И.В. Численная оптимизация на основе алгоритмов сортировки с приложением к дифференциальным и нелинейным уравнениям общего вида // Кибернетика и системный анализ. – 2011. - № 2. – С. 165-180.
6. Ромм Я.Е., Чабанюк Д.А. Параллельные алгоритмы обработки структур данных и последовательное моделирование параллельного построения декартова дерева / Таганрог. ин-т им. А.П. Чехова (филиал) ФГБОУ ВПО РГЭУ (РИНХ). – Таганрог, 2015. – 53 с. Деп. в ВИНИТИ 16.01.15, № 9 – В2015.
7. Ромм Я.Е., Чабанюк Д.А. Поразрядно-параллельное сравнение ключей в некоторых древовидных структурах данных / Таганрог. ин-т им. А.П. Чехова (филиал) ФГБОУ ВПО РГЭУ (РИНХ). – Таганрог, 2014. – 41 с. Деп. в ВИНИТИ 04.09.14, № 244 – В2014.
8. Ромм Я.Е., Чабанюк Д.А. Сравнение слов с единичной временной сложностью // Известия ЮФУ. Технические науки. – 2014. - № 7(156). – С. 230-238.
9. Солодовников В.И. Верхние оценки сложности решения систем линейных уравнений // В кн.: Теория сложности вычислений. I: Записки научных семинаров ЛОМИ АН СССР. – Л., 1982. –Т. 118. – С. 159-187.
10. Blelloch G., Reid-Miller M. Fast set operations using treaps // Proc. 10th ACM Symp. Parallel Algorithms and Architectures, New York: ACM. – 1998. – P. 16-26.

Рецензенты:

Веселов Г.Е., д.т.н., директор Института компьютерных технологий и информационной безопасности, Инженерно-технологическая академия Южного федерального университета, г. Таганрог;

Турулин И.И., д.т.н., профессор, профессор кафедры информационных измерительных технологий и систем, Институт нанотехнологий, электроники и приборостроения Южного федерального университета, г. Таганрог.