УДК 004.421.6 DOI 10.18522/2311-3103-2024-4-50-63

И.И. Левин, К.Н. Алексеев, А.А. Гуленок

ПРЕОБРАЗОВАНИЕ ПРОСТЕЙШИХ СОРТИРУЮЩИХ СЕТЕЙ К НЕЧЕТНО-ЧЕТНОЙ СЕТИ БАТЧЕРА

Все алгоритмы сортировки являются информационно-эквивалентными, поэтому выбор наиболее эффективного алгоритма обычно зависит от скорости его работы и от количества используемой памяти. При параллельной, аппаратной реализации на эффективность алгоритмов сортировки также влияют: степень утилизации аппаратных ресурсов; латентность результирующей вычислительной структуры; количество и разрядность сортируемых элементов. Задача сортировки или упорядочивания данных не формализована в виде математических преобразований, поэтому каждый из известных алгоритмов ее решения принято рассматривать как атомарную, независимую единицу. Переход от одного алгоритма решения задачи к другому возможен при описании задачи в виде информационного графа, вершины которого отражают элементарные выполняемые операции, а дуги – информационные зависимости между ними. Имея набор элементарных преобразований, можно влиять на функциональную регулярность связей информационного графа, латентность вычислительной структуры, коэффициент параллелизма и т.д. К преимуществам работы с информационным графом задачи также можно отнести сравнительную простоту используемого понятийного аппарата. Информационный граф задачи сортировки «пузырьком» представляет собой простейшую сортирующую сеть, построенную по принципу объединения ступеней «голова-хвост». В данной работе показана и обоснована функциональная избыточность подобных сортирующих сетей; приведены способы оптимизации числа операций и изменения порядка их следования. Основным результатом работы является методика преобразования сортирующих сетей в нечетно-четную сеть слияния Батчера. Разработана программа, автоматически выполняющая преобразование сортирующих сетей и позволяющая подстраивать топологию информационного графа под наиболее эффективный вид в зависимости от результирующей степени параллелизма вычислительной структуры. Обобщая полученные результаты, можно отметить, что автоматизированное приведение известных алгоритмов к «быстрым» может обеспечить получение оптимальной параллельно-конвейерной программы при заданных ограничениях, что позволит значительно ускорить процесс их разработки.

Алгоритмы сортировки; сортирующие сети; сеть слияния Батчера; информационный граф; ярусно-параллельная форма; преобразование алгоритмов.

I.I. Levin, K.N. Alekseev, A.A. Gulenok

TRANSFORMATION OF THE SIMPLEST SORTING NETWORKS TO AN ODD-EVEN BUTCHER NETWORK

All sorting algorithms are information-equivalent. Therefore, the choice of the most effective algorithm usually depends on its operation velocity and the capacity of used memory. At parallel, hardware implementation, the efficiency of sorting algorithms is also affected by the degree of utilization of hardware resources; the latency of the resulting computing structure; the number and digit capacity of the sorted elements. The problem of sorting or ordering data is not formalized in the form of mathematical transformations. Therefore, each of the known algorithms for solving it is considered an atomic, independent unit. The transition from one algorithm for solving the problem to another is possible at describing the problem in the form of an information graph, the vertices of which represent the elementary performed operations, and the arcs – the information dependencies between them. Having a set of elementary transformations, it is possible to influence the functional regularity of the information graph connections, the latency of the computing structure, the coefficient of parallelism, etc. The information graph of the "bubble" sorting problem is a simple sorting network, developed on the "head-tail" principle of combining steps. In this paper, the functional redundancy of such sorting networks is shown and justified; the methods to optimize the number of operations and change the order of their sequence are given. The main result of the paper is the method for converting sorting networks into an odd-even Butcher mergesort. A program has been developed that automatically performs the transformation of sorting networks and allows to adjust the information graph topology to the most effective form, depending on the resulting degree of parallelism of the computing structure. Summarizing the

obtained results, note that the automated reduction of known algorithms to "fast" ones can ensure the optimal parallel pipeline program under specified constraints, which will significantly accelerate the process of their development.

Sorting algorithms; sorting networks; Batcher mergesort; information graph; tiered-parallel form; algorithm transformation.

Введение. Операция сортировки является составной частью анализа, фильтрации и преобразования данных, поэтому широко используется в множестве алгоритмов решения прикладных задач. Известно большое число различных алгоритмов сортировки данных [1–3], начиная от академической «сортировки пузырьком» или «сортировки слиянием» и заканчивая наиболее производительными алгоритмами, такими как «team sort» [4] и «q-sort» [5]. Многообразие алгоритмов сортировки обусловлено различием в числе и типе выполняемых операций: непосредственное сравнение данных или сравнение с константными значениями; сравнение с краями диапазона или нахождение количества однотипных элементов и т.д. [1–3].

При выполнении сортировки на вычислительных системах традиционной архитектуры наилучшим является алгоритм с меньшим числом операций, оптимально использующий процессорное время и позволяющий минимизировать простои при передаче информации между системой хранения данных, оперативной памятью и кэшем процессора. Для ускорения сортировки больших массивов данных широко используются параллельные аппаратные реализации алгоритмов сортировки, например, конвейерная сортировка пузырьком, сортирующие сети Батчера, поразрядная сортировка, а также иные гибридные методы сортировки [6–9]. В этом случае выбор алгоритма сортировки сводится решению задачи многокритериальной оптимизации, на сложность которой влияют: степень распараллеливания, определяющая коэффициент редукции и латентность вычислительной структуры; заданное время обработки данных; доступный аппаратный ресурс; регулярность информационного графа задачи; требуемый объем памяти для хранения результатов промежуточных вычислений; тип, количество и разрядность сортируемых элементов [6].

Следует отметить, что все существующие алгоритмы сортировки информационно-эквивалентны, поскольку все они выполняют функцию упорядочивания элементов множества. В тоже время между различными алгоритмами сортировки существует семантический разрыв вследствие того, что математический аппарат для описания различных алгоритмов сортировки не формализован и не существует методологических основ для осуществления перехода от одного алгоритма к другому. Каждый из алгоритмов сортировки рассматривается как отдельный и независимый подход к решению задачи упорядочивания данных, из-за чего невозможно автоматизировать процесс получения более эффективных алгоритмов из исходного для определенного аппаратного ресурса.

Иным подходом к синтезу параллельных алгоритмов является анализ и преобразование информационных графов задачи [10–11]. Например, в работах [12–18] показаны способы преобразования информационных графов с ассоциативными и дистрибутивными алгебраическими операциями, за счет чего были получены графы, соответствующие другим известным алгоритмам решаемой задачи. Преимуществом работы с информационными графами по сравнению с математическими формулами является сравнительная простота используемого понятийного аппарата. Исследования показывают, что задача преобразования информационного графа обладает меньшей алгоритмической сложностью по сравнению последовательным выполнением математических преобразований, что может обеспечить возможность ее дальнейшей автоматизации.

В рамках данной статьи описаны правила преобразования информационных графов задачи сортировки, представляющих собой сортирующие сети, на базе которых создана методика оптимизации сортирующих сетей. Работоспособность методики проиллюстрирована на примере преобразования информационного графа простейшей сортирующей сети, соответствующей последовательному алгоритму сортировки «пузырьком», к нечетно-четной сортирующей сети Батчера (Batcher odd-even merge sort) [19], содержащей наименьшее число сортирующих элементов.

Сортирующие сети. Известно большое число разнообразных сортирующих сетей, отличающихся количеством сортирующих элементов SE, латентностью вычислительной структуры, регулярностью связей меду сортирующими элементами [1–3, 19]. Из-за этого, разные сортирующие сети являются оптимальными при разной степени параллелизма [12]. Характерной чертой простейших сортирующих сетей (рис. 1,а,б), является объединение сортирующих элементов SE в «ступени», выполняющие следующие функции (на рис. 1,а,б выделены пунктиром):

1) нахождение максимума: $X = X_n > \{X_1, X_2, X_3, \ \ldots, X_{n-1}\}$, $X_n = max(A)$;
2) вставка элемента в отсортированное множество;
3) вставка элемента в частично упорядоченное множество;
4) перенос нескольких больших элементов ближе к старшим выходам графа.

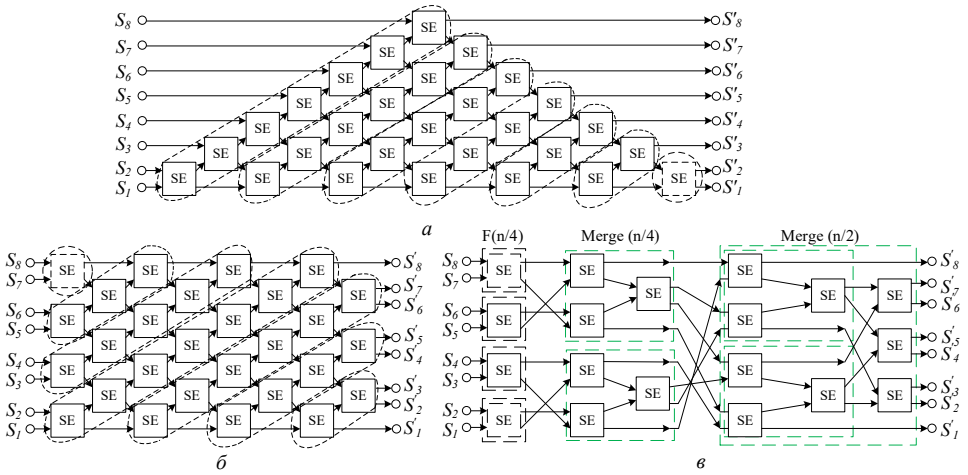


Рис. 1. Схемы простейших сортирующих сетей при n = 8, последовательная реализация которых соответствует последовательным алгоритмам: а – сортировки «пузырьком»; б – нечетно-четной сортировке; в – нечетно-четной сортирующей сети Батчера

Очевидно, чем меньше сортирующих элементов в схеме, тем большую степень масштабирования можно обеспечить в ограниченном аппаратном ресурсе. Кен Батчер создал две универсальные архитектуры сортирующих сетей с функционально-регулярными информационными графами и вычислительной сложностью около $O(n \cdot log(n))$, где n – мощность входного массива данных. Сеть, состоящая из наименьшего числа сортирующих элементов, называется «нечетно-четная сеть слияния Батчера» (рисунок 1-в). Основная идея при построении данной сети заключается в последовательном слиянии двух упорядоченных подмножеств по принципу: слияние нечетных элементов входных упорядоченных последовательностей; слияние четных элементов входных упорядоченных последовательностей; дальнейшее слияние промежуточных данных в одну общую упорядоченную последовательность с помощью $n/2 - 1$ операций.

Избыточность простейших сетей сортировки. Как отмечалось в работе [12], ступень сортирующей сети реализует сразу несколько различных функций, одной из которых является перенос нескольких больших элементов ближе к старшим выходам графа. Так как основным принципом работы ступени в простейших сортирующих сетях является именно нахождение максимума в неотсортированном множестве элементов, перемещение не максимальных элементов очевидно не несет никакого смысла. Также очевидно, что для сортировки элементов массива достаточно единожды переместить каждый элемент на новое место, для чего необходимо выполнить всего n операций перемещения элементов, что реализуется в последовательном алгоритме сортировки «выбором». Исходя из вышесказанного, можно сделать вывод, что простейшие сортирующие сети, количество сортирующих элементов в которых равно $n^2/2$ обладают избыточностью выполняемых операций сравнения и перемещения элементов.

Избавиться от подобной избыточности можно путем использования принципа «деления пополам», который соответствует принципу описания последовательных алгоритмов «разделяй и властвуй». Согласно данным принципам как раз и построена нечетно-четная сортирующая сеть Батчера, а также и некоторые «быстрые» алгоритмы иных задач: бинарный поиск, умножение Карацубы, быстрое дискретное преобразование Фурье и т.п. [20].

Для устранения избыточности выполняемых операций в первую очередь необходимо построить сортирующую сеть по принципу «деления пополам» (рис. 3), используя при этом принципы объединения ступеней, подробно рассмотренные в работе [12]. Заметим, что сортирующую сеть, соответствующую последовательному алгоритму «шейкерной» сортировки (рис. 2,а), можно логически разбить на подграфы: выделить два подграфа, которые представляют собой простейшие сортирующие сети меньшего размера *F(n/2)*, а также подграф сети слияния двух отсортированных последовательностей *Merge.* После аналогичного преобразования каждой из выделенных сортирующих сетей меньшего размера, была получена сортирующая сеть, состоящая из сетей слияния разного размера (рис. 2,б). Так как после перестановки ступеней в сортирующей сети число выполняемых операций не изменилось, можно сделать вывод о существующей избыточности выделенных сетей слияния.

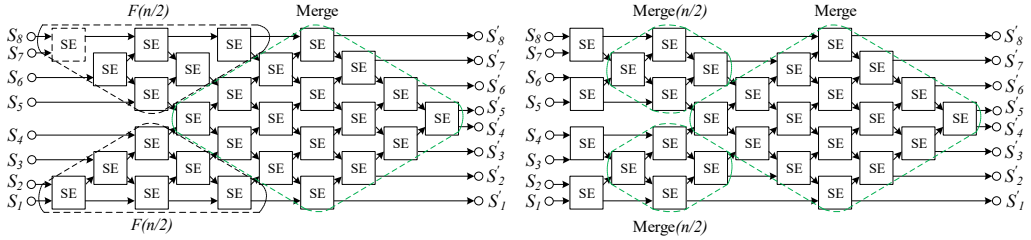


Рис. 2. Исходная (а) и преобразованная (б) схемы сортирующей сети, соответствующей последовательному алгоритму «шейкерной» сортировки

Было обнаружено, что сеть слияния можно функционально разделить на две составные части: сеть перестановки элементов, которая на рис. 3,а выделена пунктиром, и оставшаяся часть сортирующих элементов, отвечающая за непосредственное слияние последовательностей.

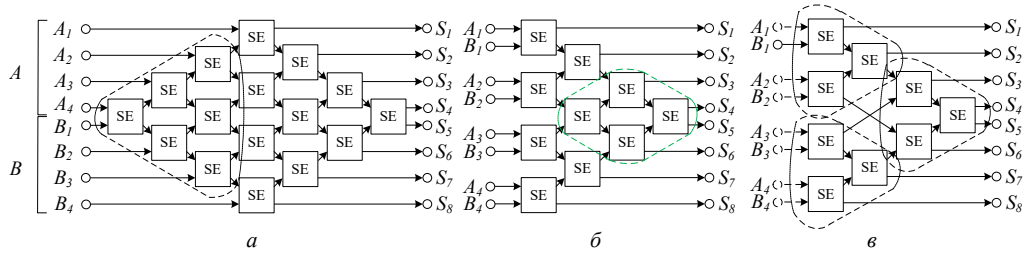


Рис. 3. а – сеть слияния, полученная из простейшей сортирующей сети при n = 8; б – сеть слияния после первого шага оптимизации; в – оптимизированная сеть слияния

Оптимизируем сеть слияния путем удаления функционально-избыточных операций. На первом шаге оптимизации удалим часть операций, отвечающую за перестановку данных (рис. 3,б), и изменим порядок поступления массивов A и B на функцию слияния. Заметим, что после этого часть сети слияния, выделенная пунктиром на рис. 3,б, представляет собой сеть слияния меньшего размера, которую можно оптимизировать тем же способом. На рис. 3,в приведена сеть слияния пирамидального вида, полученная после выполнения всех шагов оптимизации. Части сети слияния, выделенные пунктиром, являются простейшими сетями слияния Батчера.

На рис. 4 представлена сортирующая сеть для n = 8, полученная после оптимизации избыточных операций в сетях слияния. Хотя данная сеть отличается от сети Батчера, она имеет такое же число сортирующих элементов.

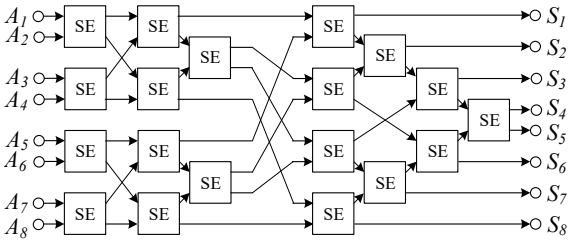


Рис. 4. Простейшая сортирующая сеть при n = 8 после оптимизации избыточных операций, отвечающих за перемещение элементов

На рис. 5 приведено сравнение структур сетей слияния, полученных после оптимизации простейшей сортирующей сети и из нечетно-четной сети Батчера, где видно, что при n = 2 и n = 4 графы сетей слияния изоморфны, а при n = 8 – состоят из одинакового числа операций. Число сортирующих элементов в сети слияния из нечетно-четной сортирующей сети Батчера определяется отношением: $N_{SE}(n) = n \cdot (\log_2 n - 1)/2 + 1$; а в пирамидальной сети слияния: $N_{SE}(n) = 3^{\log_2 n}$, в связи с чем можно говорить об имеющейся избыточности сетей слияния больших размерностей.

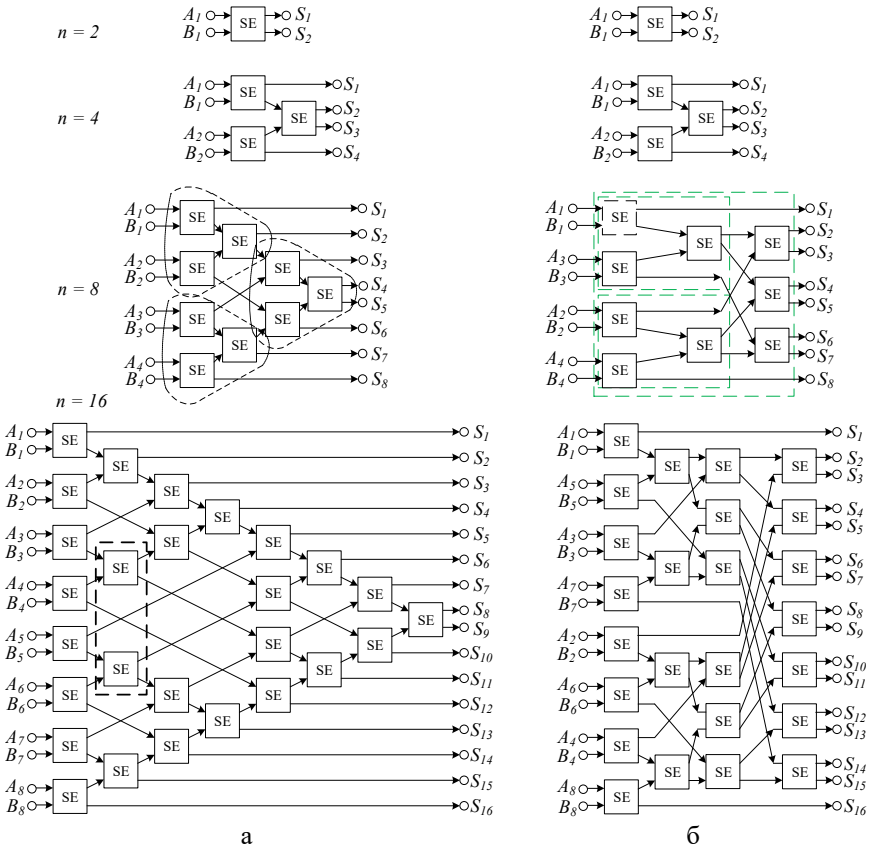


Рис. 5. Структуры сетей слияния при n=2, 4, 8 и 16, полученных: а – из простейшей сети сортировки; б – из нечетно-четной сети Батчера

Покажем, что при $n = 16$ два сортирующих элемента, выделенные пунктиром на рис. 5,а пунктиром, являются функционально-избыточными. Очевидно, что $max(A_1, B_1)$ будет первым, максимальным элементом результирующего упорядоченного множества, тогда как $max(A_2, B_2)$ может занять одну из позиций в результирующем множестве в диапазоне от 2 до $n/2 + 1$; следующий максимальный выход $max(A_3, B_3)$ может занять позицию в диапазоне от 3 до $n/2 + 3$ и т.д. При этом, $min(A_1, B_1)$ может занять одну из позиций в результирующем множестве в диапазоне от 2 до $n/2 + 1$; $min(A_2, B_2)$ может занять позицию в диапазоне от 3 до $n/2 + 2$; следующий минимальный выход $min(A_3, B_3)$ может занять позицию в диапазоне от 5 до $n/2 + 3$ и т.д. Таким образом, после первого слоя SE сортируемые данные разбиваются на некоторое количество пересекающихся диапазонов результирующего множества. Верхний элемент SE, выделенный на рис. 5,а пунктиром, принимает для сравнения $min(A_3, B_3)$ и $max(A_4, B_4)$, а выход данного элемента участвует в формировании результирующих элементов $S_4 - S_{10}$. При этом известно, что $min(A_3, B_3)$ никогда не сможет занять позицию S_4, в связи с чем данное сравнение может быть оптимизировано.

Анализируя диапазоны позиций элементов в результирующем множестве после первого слоя SE, можно определить функционально-избыточные элементы в сетях слияния любой мощности. Так на рис. 6 приведена сеть слияния при $n = 32$, в которой черным цветом выделены сортирующие элементы, которые могут быть удалены.

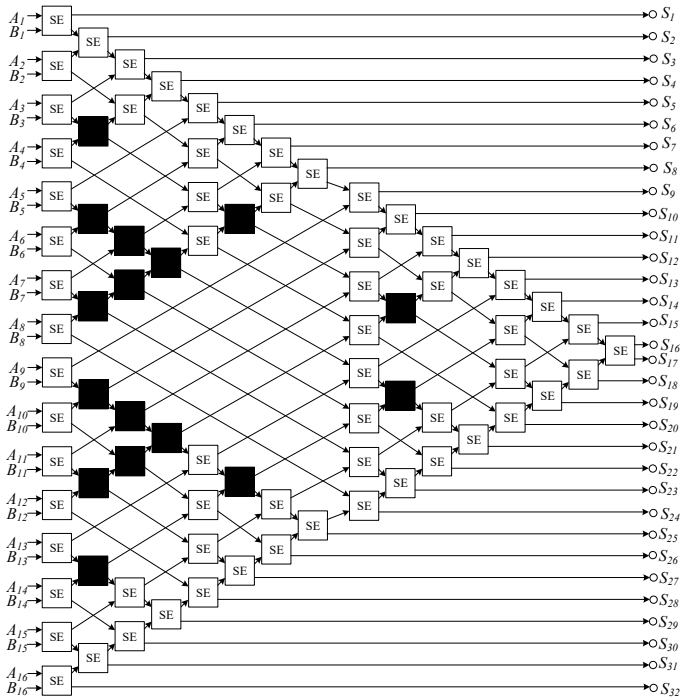


Рис. 6. Сеть слияния при $n = 32$ *с выделенными сортирующими элементами для дальнейшей оптимизации*

После удаления избыточных сортирующих элементов, их число будет соответствовать числу операций в нечетно-четной сети слияния Батчера. Однако, эксперименты показали, что сеть слияния после такого рода оптимизации перестает работать на всем множестве входных данных, из чего можно сделать вывод, что структура пирамидальной сети слияния не может быть оптимизирована без изменения порядка выполнения операций.

Изменение порядка ассоциативных операций. В сети слияния можно выделить подграфы, которые реализуют функцию вставки одного элемента в отсортированную последовательность (выделены пунктиром на рис. 7,а и рассмотрены отдельно на рис. 7,б,в).

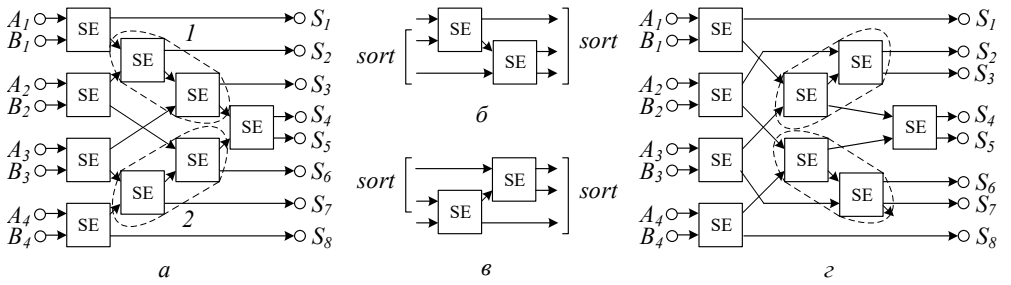


Рис. 7. а, б, в – выделение частей в сети слияния, реализующих операцию вставки элемента в отсортированную последовательность; г – сеть слияния после изменения порядка следования операций

Для подграфа №1 в качестве отсортированной части подаются элементы $max(A_2, B_2) > max(A_3, B_3)$, в качестве элемента для вставки – $min(A_1, B_1)$, а в результате будет получено упорядоченное множество элементов, содержащее два глобальных максимума. Аналогичная ситуация наблюдается при рассмотрении подграфа №2, с той разницей, что в результате будут получены два глобальных минимума. Так как результатом работы обеих рассмотренных структур будут являться упорядоченные множества элементов, можно утверждать, что они функционально-эквивалентны.

Наличие функциональной эквивалентности позволяет заменять одну структуру другой, фактически меняя структуру ступени: со структуры нахождения максимума на структуру нахождения минимума или наоборот. Так, изменив порядок следования операций в подграфах №1 и №2 на обратный, получим сеть слияния, представленную на рис. 7,г. Отметим, что граф полученной сети слияния имеет тот же порядок выполнения операций и идентичные связи между ними, что и сеть слияния нечетно-четной сортирующей сети Батчера, в связи с чем эти графы являются изоморфными.

Данное свойство можно обобщить для любой размерности ступеней сортирующей сети. Графы, приведенные на рис. 8,а,б, являются функционально-эквивалентными, поскольку выполняют одну и ту же функцию: вставку элемента в упорядоченное множество. Если изменить порядок следования операций, можно построить каскадную, пирамидальную схему, которая представлена на рис. 8,в. В данной схеме после сравнения A_2 и B, максимум перейдет для сравнения с A_1, а минимум – с A_3. Довольно очевидно, что данный граф также является функционально-эквивалентным графам на рис. 8,а,б. Рассмотренный принцип работает и для графов ступеней большей размерности (рис. 8,г,д,е,ж). При этом, если отдельно рассмотреть ступень, состоящую из трех сортирующих элементов и выделенную на рис. 8,д пунктиром, она сама по себе выполняет операцию вставки элемента в упорядоченное множество. Граф вставки элемента в отсортированный массив, приведенный на рис. 8,ж, реализует каскадную схему ставки при $n = 8$. Заметим, что данная схема построена по принципу бинарного поиска: первым выполняется сравнение со средним элементом упорядоченного подмножества, затем с серединами оставшихся частей и т.д.

Изменение порядка следования операций с последовательного «голова-хвост» на каскадный «деление пополам» при сохранении функциональной эквивалентности позволяет значительно уменьшить как число слоев в графе, так и латентность полученной на его основе вычислительной структуры.

Переход к нечетно-четной сети слияния Батчера. Рассмотрим простейшую сортирующую сеть на $n = 16$ элементов, соответствующую последовательному алгоритму сортировки «пузырьком». После перестроения сети согласно принципу «деления пополам» и оптимизации функционально-незначимых элементов получен граф, представленный на рис. 9,а. Для приведения данной сети к сети Батчера необходимо оптимизировать все выделенные пунктиром сети слияния.

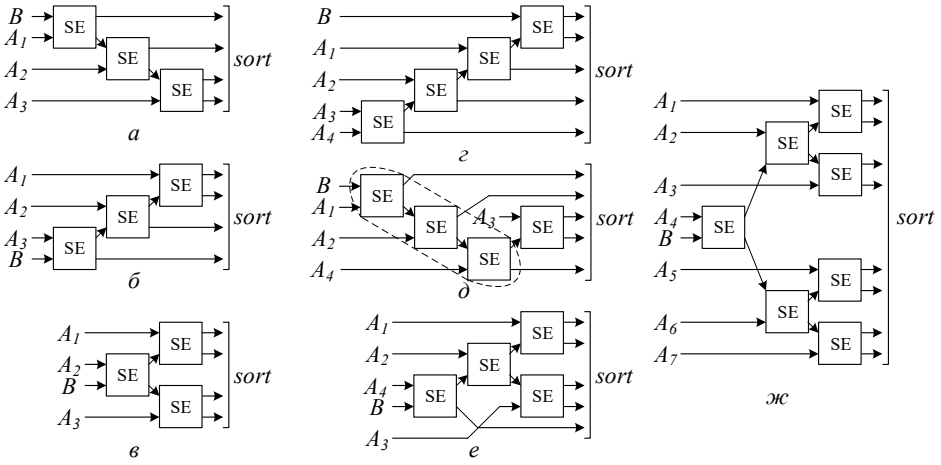
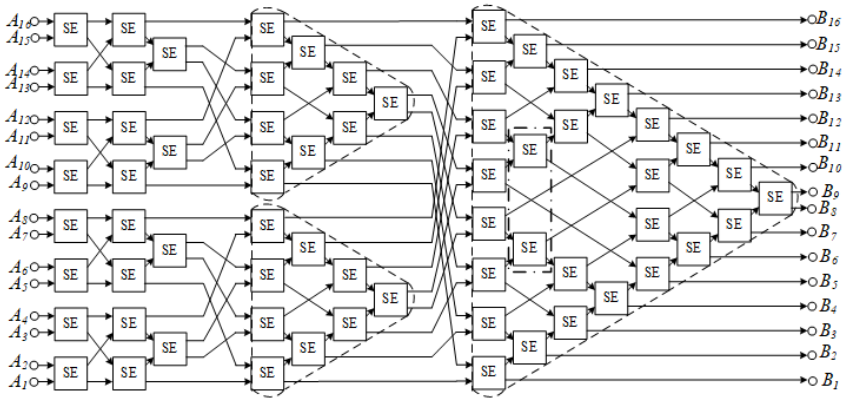
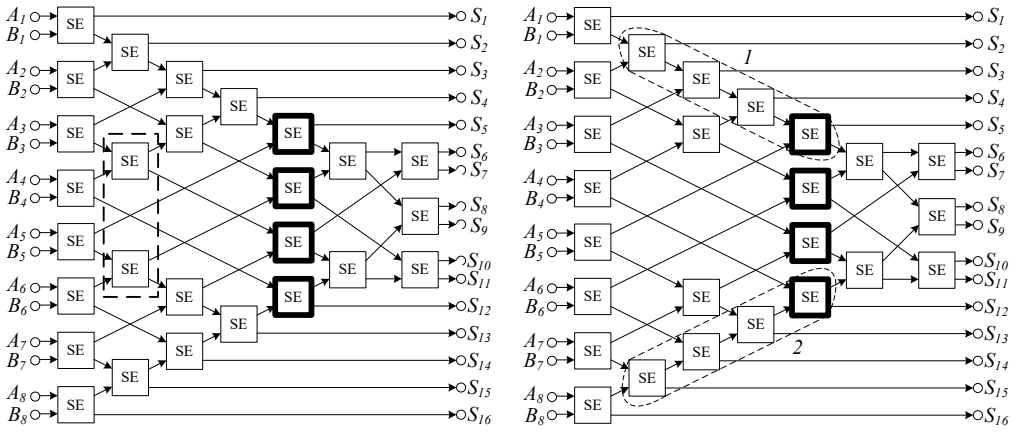


Рис. 8. Варианты графов вставки элемента в отсортированный массив при: а-в – $n = 4$; г-е – $n = 5$; ж – $n = 8$



а



б в

Рис. 9. а – сортирующая сеть при $n = 16$ после перестроения согласно принципу «деления пополам» и оптимизации функционально-незначимых элементов; б – структура сети слияния после преобразования подсети слияния средних элементов; в – после оптимизации числа сравнений

Преобразования стоит начинать с последнего, самого большого подграфа, описывающего сеть слияния двух последовательностей в отсортированное множество. Заметим, что конец данной сети тоже является сетью слияния меньшего размера и выполняет слияние центральных элементов последовательности. Выполним приведение данного подграфа к сети слияния Батчера путем изменения последовательности операций, как это было показано на рисунке 8, после чего оптимизируем функционально-избыточные сортирующие элементы, выделенные пунктиром на рис. 9,б. В данном случае после их удаления, сеть, структура которой приведена на рис, 9,в, продолжает выполнять функцию слияния. Таким образом, все наибольшие сети слияния, приведенные на рис. 9, являются функционально-эквивалентными.

Однако, для сетей слияния большей размерности это не является правилом, и сеть перестает выполнять заданную функцию, поэтому для сохранения функциональной эквивалентности требуется выполнить дальнейшие преобразования, а именно, изменить порядок ассоциативных операций таким образом, чтоб сортирующие элементы, выделенные жирной линией на рис. 9 переместились на место второго слоя сети слияния.

Изменим порядок ассоциативных операций, выделенных пунктиром на рис. 9,в. Для части сети слияния под номером 1, в качестве отсортированной части подаются элементы $max(A_2, B_2) > max(A_3, B_3) > max(A_5, B_5)$ и элемент $max(min(A_3, B_3), max(A_4, B_4), min(A_2, B_2))$. В качестве элемента для вставки поступает элемент $min(A_1, B_1)$. Последовательность преобразований для изменения порядка выполнения операций приведена на рис. 10. Аналогичные преобразования выполним для части сети слияния, представленной на рис. 9,б под номером 2. После выполнения всех преобразований, структура оптимизированной сети слияния, полученной из квадратичной сортирующей сети, станет изоморфна структуре сети слияния, выделенной из нечетно-четной сортирующей сети Батчера.

Для завершения преобразования сортирующей сети, представленной на рис. 9,в сеть Батчера необходимо последовательно изменить порядок ассоциативных операций во всех сетях слияния, выделенных пунктиром, после чего данная сеть станет изоморфна нечетно-четной сети слияния Батчера.

Анализ показал, что выполнение представленной последовательности действий позволяет привести любую простейшую сортирующую сеть к сети Батчера при любой мощности сортируемого массива данных.

Была разработана методика преобразования сортирующих сетей в нечетно-четную сеть слияния Батчера, согласно которой необходимо:

1. С помощью правил перестановки ступеней [12], привести сортирующую сеть к виду, из которого можно однозначно выделить сортирующие сети меньшей мощности и сеть слияния.

2. Повторить пункт 1 для всех полученных сортирующих сетей меньшей мощности. Рекурсивно исполнять данный пункт, пока не выполнится условие: все сортирующие сети выполняют сортировку 2 элементов исходной последовательности. Данные преобразования соответствуют построению сортирующей сети согласно принципу «деление пополам».

3. Выделить самую крупную сеть слияния.

4. Оптимизировать все выделенные сети слияния путем удаления избыточных операций, отвечающих за перемещение элементов.

5. Если размерность выделенной сети слияния больше четырех, выделить все сети слияния вдвое меньшей мощности и перейти к пункту 4. Иначе, перейти к пункту 6.

6. В полученной сети выделить и удалить функционально-избыточные сортирующие элементы, которые не участвуют в формировании результата.

7. Выделить последнюю и предпоследнюю сети слияния, не являющуюся сетью Батчера.

8. Выполнить изменение порядка следования операций начиная с краев сети слияния таким образом, чтоб первый слой SE сети слияния центральных элементов множества стал вторым слоем сети слияния центральных элементов большего размера.

9. Выполнять пункты методики 7–8 до тех пор, пока граф сортирующей сети не станет изоморфным графу сети нечетно-четного слияния Батчера.

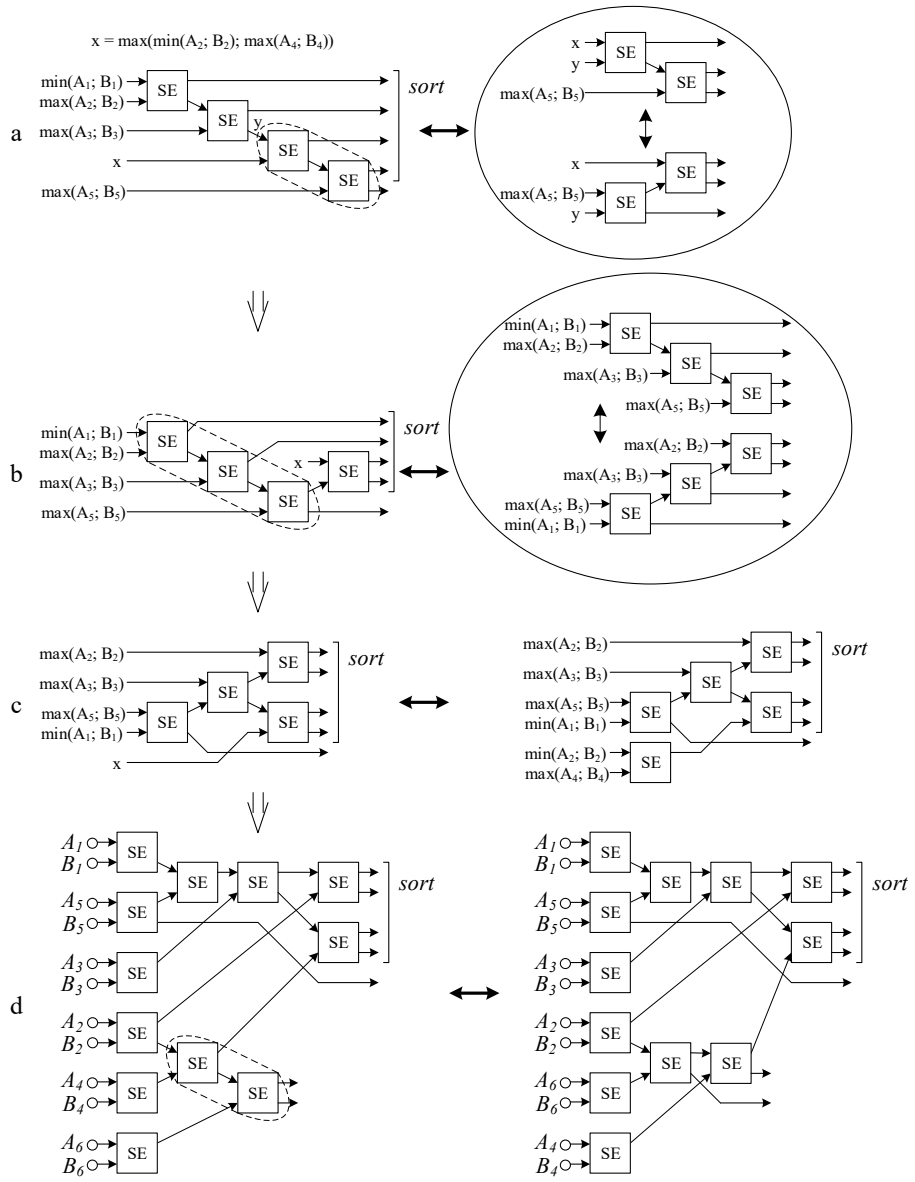


Рис. 10. Последовательность преобразований частей сети слияния, которая реализует вставку элемента в отсортированную последовательность

Апробация. Представленная методика преобразования сортирующих сетей в нечетно-четную сеть слияния Батчера была реализована в виде программы, написанной на языке высокого уровня C#. На вход программы поступает информационный граф сортирующей сети, соответствующей последовательному алгоритму сортировки «пузырьком». На выходе программа формирует информационный граф, изоморфный сети нечетно-четного слияния Батчера. Результат работы программы для $n = 32$ приведен на рис. 11.

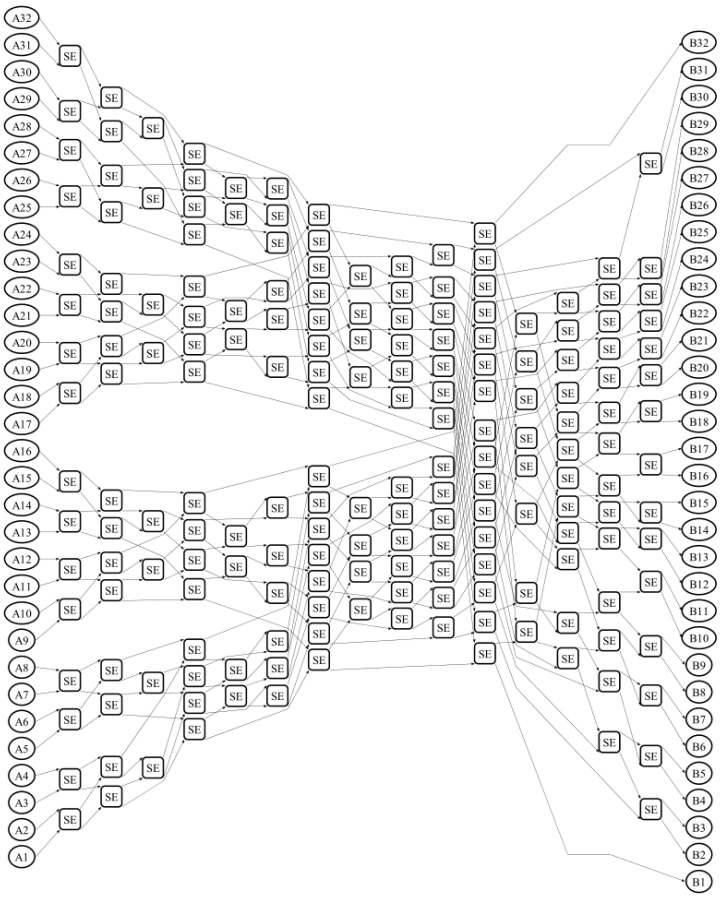


Рис. 11. Результат работы программы приведения информационных графов сортирующих сетей к сети нечетно-четного слияния Батчера

Разработанная программа была протестирована на сортирующих сетях большей размерности: для $n = 64, 128, 256$. Стоит отдельно отметить, что все операции выполняются над графом автоматически, без участия человека.

Заключение. В рамках данной работы рассмотрены способы оптимизации функционально-избыточных операций в сортирующих сетях, а также способы изменения порядка следования ассоциативных операций. Основным результатом является методика преобразования сортирующих сетей в нечетно-четную сеть слияния Батчера. Реализация разработанной методики в виде программного продукта позволяет подстраивать топологию информационного графа под наиболее эффективный вид в зависимости от результирующей степени параллелизма вычислительной структуры. Автоматизация процесса преобразования сортирующих сетей позволяет увеличить эффективность используемого оборудования без особых временных затрат на разработку и отладку кода параллельной программы при изменении темпа поступления и обработки данных.

Стоит отметить, что оптимизация функционально-избыточных элементов стала возможна только после построения сортирующей сети по принципу «деления пополам». Обобщая данный подход, можно утверждать, что алгоритмы, построенные согласно принципу «деления пополам» способны значительно уменьшить общее число выполняемых операций, что наблюдается при переходе от сортирующей сети, соответствующей последовательному алгоритму сортировки «пузырьком», к нечетно-четной сети слияния Батчера.

Вместе с этим, существуют иные подходы, иначе реализующие парадигму «разделяй и властвуй»: предобработка входных данных с последующим выполнением основных функций. По таким принципам построена битоническая сортировка Батчера, «q-sort», «ведерная» сортировка и поразрядная сортировка. Теоретически, автоматизация перехода от сортирующих сетей к данным алгоритмам сортировки позволит выбирать оптимальный алгоритм сортировки для имеющегося аппаратного ресурса и заданного времени решения задачи без участия разработчика, что позволит значительно ускорить процесс разработки параллельно-конвейерных программ.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Кнут Д.Э.* Искусство программирования. Т. 3. Сортировка и поиск. – 2-е изд.: пер. с англ. – М.: Изд. дом «Вильямс», 2001.
2. *Скиена С.С.* Алгоритмы. Руководство по разработке. – 3-е изд.: пер. с англ. – СПб.: БХВ-Петербург, 2022. – 848 с.
3. Sorting algorithm. – Режим доступа: https://en.wikipedia.org/wiki/Sorting_algorithm (дата обращения: 31.07.2024).
4. Timsort. – Режим доступа: https://en.wikipedia.org/wiki/Timsort (дата обращения: 31.07.2024).
5. Quicksort. – Режим доступа: https://en.wikipedia.org/wiki/Quicksort (дата обращения: 31.07.2024).
6. *Алексеев К.Н.* Методы и средства создания параллельно-конвейерных программ для решения задач реального времени на реконфигурируемых вычислительных системах: дисс. … канд. техн. наук, по специальности 05.13.11 “Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей” / научный руководитель: д.т.н., проф. Левин И.И. – Таганрог, 2020. – 213 с.
7. *Алексеев К.Н., Левин И.И., Сорокин Д.А.* Структурно-процедурная реализация на реконфигурируемых вычислительных системах кодирования Хаффмана в реальном масштабе времени // Вестник компьютерных и информационных технологий. – 2018. – № 9. – С. 3-10. – ISSN 1810-7206. – DOI: 10.14489/vkit.2018.09.pp.003-010. – http://www.vkit.ru/index.php//current-issue-rus/751-003-010.
8. *Алексеев К.Н., Сорокин Д.А., Семерникова Е.Е.* Структурно-процедурная реализация кодирования алгоритмом Хаффмана в задачах управления // Матер. 10-й Всероссийской мультиконференции по проблемам управления (МКПУ-2017), 11-16 сентября 2017 г. – Ростов-на-Дону: Изд-во ЮФУ, 2017. – Т. 3. – С. 126-127.
9. *Dudnikov E.A., Alekseev K.N., Sorokin D.A.* High-Speed Huffman encoding of dense data flows on FPGA // German International Journal of Modern Science / Deutsche internationale Zeitschrift für zeitgenössische Wissenschaft. – 2021. – No. 9, Vol. 1. – P. 36-40. – https://dizzw.com/wp-content/uploads/2021/05/Deutsche-internationale-Zeitschrift-für-zeitgenössische-Wissenschaft-№9-part-1-2021-37-41.pdf.
10. *Каляев А.В., Левин И.И.* Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений. – М.: Янус-К, 2003. – 380 с.
11. *Каляев И.А., Левин И.И., Семерников Е.А., Шмойлов В.И.* Реконфигурируемые мультиконвейерные вычислительные структуры. – 2-е изд., перераб. и доп. / под общ. ред. И.А. Каляева. – Ростов-на-Дону: Изд-во ЮНЦ РАН, 2009. – 344 с. – ISBN 978-5-902982-61-6.
12. *Левин И.И., Алексеев К.Н.* Преобразование сортирующих сетей для разной степени параллелизма // Известия ЮФУ. Технические науки. – 2023. – № 5 (235). – С. 104-118. – DOI: 10.18522/2311-3103-2023-5-104-118.
13. *Писаренко И.В., Алексеев К.Н., Мельников А.К.* Ресурсонезависимое представление сортирующих сетей на языке программирования Set@l // Вестник компьютерных и информационных технологий. – 2019. – № 11 (185). – С. 53-60. – DOI: 10.14489/vkit.2019.11.pp.053-060. http://vkit.ru/index. php/ current-issue-rus/857-053-060.
14. *Писаренко И.В., Касаркин А.В., Алексеев К.Н.* Рекурсивное описание графов с ассоциативными операциями на языке программирования Set@l // Всероссийская научная конференция "Суперкомпьютерные технологии (СКТ) 2020", 05-10 октября 2020 г. Крым, Алушта, Россия. – Ростов-на-Дону; Таганрог: Изд-во ЮФУ, 2020. – С. 79-83.
15. *Михайлов Д.В.* Методы создания параллельно-конвейерных программ для задач с последовательным информационным графом: дисс. ... канд. техн. наук по специальности 05.13.11 “Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей” / научный руководитель: д.т.н., проф. Левин И.И. – Таганрог, 2022. – 213 с.
16. *Михайлов Д.В., Левин И.И., Дордопуло А.И., Писаренко И.В.* Представление графов с ассоциативными операциями на языке программирования SET@L // Известия ЮФУ. Технические науки. – 2020. – № 3 (213). – С. 98-109. – DOI: 10.18522/2311-3103-2020-3-98-111.

17. *Михайлов Д.В.* Преобразование некоторых видов последовательных информационных графов в параллельно-конвейерную форму // Известия ЮФУ. Технические науки. – 2020. – №7 (217). – С. 78-93. – DOI: 10.18522/2311-3103-2020-7-78-93.
18. *Михайлов Д.В.* Преобразование последовательного информационного графа метода прогонки в параллельную форму // Известия ЮФУ. Технические науки. – 2021. – № 7 (224). – С. 177-188. – DOI: 10.18522/2311-3103-2021-7-177-188.
19. Batcher odd–even mergesort. – Режим доступа: https://en.wikipedia.org/wiki/Batcher_odd%E2%80%93even_mergesort (дата обращения: 31.07.2024)
20. *Richard E.* Blahut Fast Algorithms for Digital Signal Processing. Cambridge University Press, 05.2011. – DOI: https://doi.org/10.1017/CBO9780511760921.

REFERENCES

1. *Knut D.E.* Iskusstvo programmirovaniya. T. 3. Sortirovka i poisk [The Art of Computer Programming. Vol. 3. Sorting and Search]. 2nd ed.: transl. from engl. Moscow: Izd. dom «Vil'yams», 2001.
2. *Skiena S.S.* Algoritmy. Rukovodstvo po razrabotke [Algorithms. Development Guide]. 3rd ed.: transl. from engl. Saint Petersburg: BKhV-Peterburg, 2022, 848 p.
3. Sorting algorithm. Available at: https://en.wikipedia.org/wiki/Sorting_algorithm (accessed 31 July 2024).
4. Timsort. Available at: https://en.wikipedia.org/wiki/Timsort (accessed 31 July 2024).
5. Quicksort. Available at: https://en.wikipedia.org/wiki/Quicksort (accessed 31 July 2024).
6. *Alekseev K.N.* Metody i sredstva sozdaniya parallel'no-konveyernykh programm dlya resheniya zadach real'nogo vremeni na rekonfiguriruemykh vychislitel'nykh sistemakh: diss. ... kand. tekhn. nauk, po spetsial'nosti 05.13.11 “Matematicheskoe i programmnoe obespechenie vychislitel'nykh mashin, kompleksov i komp'yuternykh setey” [Methods and tools for creating parallel-pipeline programs for solving real-time problems on reconfigurable computing systems: cand. of eng. sc. diss., in specialty 05.13.11 “Mathematical and software support for computing machines, complexes and computer networks”], scientific supervisor: dr. of eng. sc., professor Levin I.I. Taganrog, 2020, 213 p.
7. *Alekseev K.N., Levin I.I., Sorokin D.A.* Strukturno-protsedurnaya realizatsiya na rekonfiguriruemykh vychislitel'nykh sistemakh kodirovaniya Khaffmana v real'nom masshtabe vremeni [Structural-procedural implementation on reconfigurable computing systems of Huffman coding in real time], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Bulletin of computer and information technologies], 2018, No. 9, pp. 3-10. ISSN 1810-7206. DOI: 10.14489/vkit.2018.09.pp.003-010. Available at: http://www.vkit.ru/index.php//current-issue-rus/751-003-010.
8. *Alekseev K.N., Sorokin D.A., Semernikova E.E.* Strukturno-protsedurnaya realizatsiya kodirovaniya algoritmom KHaffmana v zadachakh upravleniya [Structural and procedural implementation of coding by the Huffman algorithm in control problems], *Mater. 10-y Vserossiyskoy mul'tikonferentsii po problemam upravleniya (MKPU-2017), 11-16 sentyabrya 2017 g.* [Proceedings of the 10th All-Russian Multiconference on Control Problems (MCCP-2017), September 11-16, 2017]. Rostov-on-Don: Izd-vo YuFU, 2017, Vol. 3, pp. 126-127.
9. *Dudnikov E.A., Alekseev K.N., Sorokin D.A.* High-Speed Huffman encoding of dense data flows on FPGA, *German International Journal of Modern Science / Deutsche internationale Zeitschrift für zeitgenössische Wissenschaft*, 2021, No. 9, Vol. 1, pp. 36-40. Available at: https://dizzw.com/wp-content/uploads/2021/05/Deutsche-internationale-Zeitschrift-für-zeitgenössische-Wissenschaft-№9-part-1-2021-37-41.pdf.
10. *Kalyaev A.V., Levin I.I.* Modul'no-narashchivaemye mnogoprotsessornye sistemy so strukturno-protsedurnoy organizatsiey vychisleniy [Modularly scalable multiprocessor systems with structural-procedural organization of computations]. Moscow: Yanus-K, 2003, 380 p.
11. *Kalyaev I.A., Levin I.I., Semernikov E.A., Shmoylov V.I.* Rekonfiguriruemye mul'tikonveyernye vychislitel'nye struktury [Reconfigurable multipipeline computing structures]. 2nd ed., under the general ed. of I.A. Kalyaeva. Rostov-on-Don: Izd-vo YuNTS RAN, 2009, 344 p. ISBN 978-5-902982-61-6.
12. *Levin I.I., Alekseev K.N.* Preobrazovanie sortiruyushchikh setey dlya raznoy stepeni parallelizma [Transformation of sorting networks for different degrees of parallelism], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2023, No.№ 5 (235), pp. 104-118. DOI: 10.18522/2311-3103-2023-5-104-118.
13. *Pisarenko I.V., Alekseev K.N., Mel'nikov A.K.* Resursonezavisimoe predstavlenie sortiruyushchikh setey na yazyke programmirovaniya Set@l [Resource-independent representation of sorting networks in the Set@l programming language], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Bulletin of Computer and Information Technologies], 2019, No. 11 (185), pp. 53-60. DOI: 10.14489/vkit.2019.11. pp.053-060. Available at: http://vkit.ru/index. php/ current-issue-rus/857-053-060.

14. *Pisarenko I.V., Kasarkin A.V., Alekseev K.N.* Rekursivnoe opisanie grafov s assotsiativnymi operatsiyami na yazyke programmirovaniya Set@l [Recursive Description of Graphs with Associative Operations in the Set@l Programming Language], *Vserossiyskaya nauchnaya konferentsiya "Superkomp'yuternye tekhnologii (SKT) 2020", 05-10 oktyabrya 2020 g. Krym, Alushta, Rossiya* [All-Russian Scientific Conference "Supercomputer Technologies (SCT) 2020", October 5-10, 2020. Crimea, Alushta, Russia]. Rostov-on-Don; Taganrog: Izd-vo YuFU, 2020, pp. 79-83.
15. *Mikhaylov D.V.* Metody sozdaniya parallel'no-konveyernykh programm dlya zadach s posledovatel'nym informatsionnym grafom: diss. ... kand. tekhn. nauk po spetsial'nosti 05.13.11 “Matematicheskoe i programmnoe obespechenie vychislitel'nykh mashin, kompleksov i komp'yuternykh setey” [Methods for creating parallel-pipeline programs for problems with a sequential information graph:... cand. of eng. sc. diss. in specialty 05.13.11 “Mathematical and software support for computing machines, complexes and computer networks”], scientific supervisor: dr. of eng. sc., professor Levin I.I. Taganrog, 2022, 213 p.
16. *Mikhaylov D.V.* Metody sozdaniya parallel'no-konveyernykh programm dlya zadach s posledovatel'nym informatsionnym grafom: diss. ... kand. tekhn. nauk po spetsial'nosti 05.13.11 “Matematicheskoe i programmnoe obespechenie vychislitel'nykh mashin, kompleksov i komp'yuternykh setey” [Methods for creating parallel-pipeline programs for problems with a sequential information graph:... cand. of eng. sc. diss. in specialty 05.13.11 “Mathematical and software support for computing machines, complexes and computer networks”], scientific supervisor: dr. of eng. sc., professor Levin I.I. Taganrog, 2022, 213 p.
17. *Mikhaylov D.V.* Preobrazovanie nekotorykh vidov posledovatel'nykh informatsionnykh grafov v parallel'no-konveyernuyu formu [Transformation of Some Types of Sequential Information Graphs into a Parallel-Pipeline Form], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2020, No. 7 (217), pp. 78-93. DOI: 10.18522/2311-3103-2020-7-78-93.
18. *Mikhaylov D.V.* Preobrazovanie posledovatel'nogo informatsionnogo grafa metoda progonki v parallel'nuyu formu [Transformation of the sequential information graph of the sweep method into a parallel form] *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2021, No. 7 (224), pp. 177-188. DOI: 10.18522/2311-3103-2021-7-177-188.
21. Batcher odd–even mergesort. Available at: https://en.wikipedia.org/wiki/Batcher_odd%E2%80%93even_mergesort (accessed 31 July 2024).
19. *Richard E.* Blahut Fast Algorithms for Digital Signal Processing. Cambridge University Press, 05.2011. DOI: https://doi.org/10.1017/CBO9780511760921.

Статью рекомендовал к опубликованию д.т.н., профессор А.В. Боженюк.

Левин Илья Израилевич – Южный федеральный университет; e-mail: iilevin@sfedu.ru; г. Таганрог, Россия; тел.: 88634612111; зав. кафедрой; д.т.н.; профессор.

Алексеев Кирилл Николаевич – e-mail: alexseev91@mail.ru; тел.: 89283536268; к.т.н.; доцент.

Гуленок Андрей Александрович – Научно-исследовательский центр суперЭВМ и нейрокомпьютеров; e-mail: gulenok@superevm.ru; г. Таганрог, Россия; тел.: 88634612111; начальник сектора; к.т.н.

Levin Ilya Izrailevich – Southern Federal University; e -mail: iilevin@sfedu.ru; Taganrog, Russia; phone: +78634612111; head of department; dr. of eng. sc.; professor.

Alekseev Kirill Nikolayevich – e-mail: alexseev91@mail.ru; phone: +79283536268; cand. of eng. sc.; associate professor.

Gulenok Andrei Aleksandrovich – Supercomputers and Neurocomputers Research Center; e-mail: gulenok@superevm.ru; Taganrog, Russia; phone: +78634612111; head of sector; cand. of eng. sc.