УДК 519.683.4

АНАЛИЗ ЭФФЕКТИВНОСТИ СБАЛАНСИРОВАННЫХ ДЕРЕВЬЕВ ПОИСКА В РЕШЕНИИ ЗАДАЧИ УПАКОВКИ В КОНТЕЙНЕРЫ

Д.Д. Стрыгин
Институт проблем управления им. В.А. Трапезникова РАН
Россия, 117997, Москва, Профсоюзная ул., 65
E-mail: denstrygin@gmail.com

А.В. Егоркин
Московский государственный университет имени М. В. Ломоносова
Россия, 199911, Москва, Ленинские горы, 1
E-mail: egorkinandrewv@mail.ru

Ключевые слова: задача упаковки в контейнеры, дискретная оптимизация, жадные алгоритмы, сбалансированные деревья поиска.

Аннотация: Данное исследование посвящено анализу применению сбалансированных деревьев поиска для оптимизации процессов упаковки в одномерные контейнеры. В работе рассматривается такая структура данных, как AVL-дерево и его модификации. Основное внимание уделяется анализу того, как структура и алгоритмы сбалансированного дерева способствует увеличению скорости поиска подходящего одномерного контейнера при упаковке объекта. Исследование включает сравнение времени, затрачиваемого различными модификациями. Алгоритмы сравниваются на четырех наборах данных по времени, необходимом для нахождения решения. Проведенное исследование показало, что эффективность второй предложенной в докладе модификации для FFD выше, чем у предшественника.

1. Введение

Задача упаковки в контейнеры (bin-packing problem, BBP) принадлежит области дискретной оптимизации. Она заключается в размещении объектов различных размеров в неограниченное конечное число контейнеров с заданной вместимостью таким образом, чтобы число используемых контейнеров было минимальным.

Классическая постановка задачи об упаковке в контейнеры впервые сформулирована в работе [1], где помимо постановки исследуются различные стратегии и методы, используемые в алгоритмах для выделения и освобождения памяти в компьютерных системах, а также оценивается их эффективность в самых неблагоприятных условиях.

Задачи онлайн упаковки в контейнеры (online bin packing problems) относятся к классу задач оптимизации, где решения должны быть приняты в режиме реального времени [4], без знания будущих входных данных. В отличие от традиционных задач упаковки, в онлайн-версии предметы для упаковки представляются последовательно, и каждое решение о распределении предмета по контейнерам должно быть принято без информации о последующих предметах.

BPP является NP-трудной задачей [5], потому нахождение точного решения за полиномиальное время в общем случае считается невозможным. В связи с этим в данной задаче применяются эвристические алгоритмы, которые находят приближенный оптимум. Наиболее используемыми на практике эвристиками являются жадные алгоритмы. Среди них можно выделить FFD (first-fit decreasing) и BFD (best-fit decreasing) [6]. Данные алгоритмы имеют асимптотическую сложность $O(n^2)$, однако могут быть ускорены с помощью сбалансированных деревьев поиска до временной оценки в $O(n \cdot \log_2 n)$.

С целью определения наиболее выигрышной структуры сбалансированного дерева поиска в докладе проведен эксперимент, заключающийся в сравнении затрачиваемого времени на разных наборах данных различных модифицированных деревьев. Оценка по времени работы алгоритма производиться от начала и вплоть до получения конечного результата.

2. Основная часть

2.1. Описание алгоритмов FFD и BFD

Исследование жадных эвристических алгоритмов [6] в решении BPP показало, что одними из вариантов получения наиболее приближенного к оптимуму результата являются алгоритмы первого подходящего контейнера (first-fit, FF) и наилучшего подходящего контейнера (best-fit) BF, применяемые к заранее отсортированному по не возрастанию ряду упаковываемых предметов (decreasing).

FFD работает по следующему алгоритму. Каждый предмет помещается в первый контейнер, в котором достаточно места для его размещения, начиная с начала списка задействованных контейнеров. Если ни в одном из задействованных контейнеров нет достаточного места, открывается новый контейнер, в который помещается предмет.

По алгоритму BFD каждый предмет помещается в контейнер таким образом, что оставшиеся пространство является минимально возможным среди всех задействованных контейнеров. В случае нехватки места, предмет располагается в новооткрытом контейнере.

2.2. Структуры используемых сбалансированных деревьев поиска

В качестве исследуемых структур данных была выбрана наиболее известная и применяемая структура сбалансированных деревьев поиска – AVL-дерево [7]. Для алгоритма BFD структура использовалась в классическом виде. Для FFD применяются две различные модификации.

Структура используемого дерева для алгоритма BFD имеют следующие общие свойства:

- Каждый узел дерева для алгоритма BFD является контейнером
- Каждый узел дерева хранит число, обозначающее оставшееся свободное место в контейнере
- После полного заполнения узла в дереве (т.е. свободное место узла равно 0), он удаляется
- После обновления значения оставшегося места в узле, он удаляется и вставляется в дерево заново при наличии свободного места с перебалансировкой дерева при нарушении условий баланса после каждой из операций

Алгоритм BFD с применением дерева включает в себя следующие шаги:

- Для каждого предмета из очереди в сбалансированном дереве ищется узел, обладающий достаточным свободным местом. Если оставшееся после вставки

пространство равно нулю, узел удаляется. При нарушении условий баланса дерево перебалансируется, а счетчик используемых контейнеров увеличивается на единицу

- В случае нахождения узла, удовлетворяющего объему предмета, но заполняемого им не полностью, ищется узел, пространство в котором после вставки будет минимальным. Поиск происходит в левом поддереве найденного в первом пункте узла и продолжается до тех пор, пока свободное пространство узла после вставки не будет равно 0 или не будет достигнут листовой узел дерева.
- В случае достижения листового узла после поиска минимального остающегося пространства, узел выбирается среди пройденных на пути. Значение свободного пространства в нем обновляется. Выбранный узел удаляется. При нарушении условий баланса дерево перебалансируется
- В случае достижения листа дерева без найденного подходящего узла для упаковки, в дерево вставляется новый узел (контейнер) с соответствующим свободным местом. При нарушении баланса дерево перебалансируется

Для Алгоритма FFD используются две различные модифицированные структуры деревьев. В первой модификации [7] в узле хранится следующая информация:

- Номер используемого контейнера
- Оставшиеся место в используемом контейнере
- Наибольшее оставшиеся место среди самого узла и его дочерних узлов (MAX)

Данная структура является сбалансированным деревом поиска по номерам используемых контейнеров «рис. 1». Контейнер для вставки определяется по следующему алгоритму:

- Поиск подходящего контейнера начинается с корневого узла.
- Если в узле MAX меньше необходимого, в дерево вставляется новый узел с соответствующим свободным местом. Вставка происходит по правилам бинарного дерева поиска. При переходе на новый узел в поисках подходящего места для вставки, происходит сравнение MAX, записанного в узле. В случае меньшего значения, оно перезаписывается на значение вставляемого. После вставки при нарушении баланса дерево перебалансируется
- Если свободное место в текущем узле меньше или равно MAX, а также оно позволяет произвести вставку в себя, выполняется сравнение MAX левого дочернего узла. В случае MAX, позволяющего вставку в левое поддерево, алгоритм переходит к нему и повторяет пройденные шаги
- Если после сравнения MAX левого дочернего узла, оно оказалось меньше необходимого, то вставка производиться в текущий узел. После вставки значения MAX изменяются при необходимости во всех пройденных на пути узлах, а также при нарушении баланса дерево перебалансируется
- Если свободное место в текущем узле меньше или равно MAX и оно не позволяет произвести вставку в себя, выполняется сравнение MAX и левого, и правого дочернего узла. В случае MAX, позволяющего вставку в левый узел, алгоритм переходит к нему и повторяет пройденные шаги. Иначе, в случае MAX, позволяющего вставку в правый узел, для перехода выбирается он.

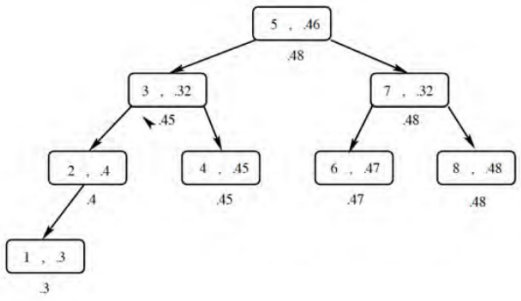

Рис. 1. 1-ая модификация для AVL-дерева в алгоритме FFD.

Таким образом выполняется условие выбора первого подходящего используемого контейнера для вставки, а в худшем случае поиск нужного контейнера происходит за высоту дерева, что в сбалансированных деревьях является $O(\log_2 n)$.

Во второй модификации в узле хранится следующая информация:

- Номер используемого контейнера
- Оставшиеся место в используемом контейнере
- Ссылка на наименьший по номеру контейнер правого поддерева
- Наибольшее свободное место среди узлов правого поддерева

Данная структура является сбалансированным деревом поиска по оставшемуся свободному месту используемых контейнеров «рис.2». Контейнер для вставки определяется по следующему алгоритму:

- Поиск подходящего контейнера начинается с корневого узла.
- Если текущий узел позволяет вставку, то ссылка на него сохраняется в отдельную переменную. Далее спуск в левый дочерний узел.
- Если вставка в текущий узел невозможна при позволяющем максимальном свободном месте, спуск в правое поддерево с повторением предыдущих шагов, иначе вставка происходит в контейнер по ссылке. Если ссылка в случае вставки в неё не ведет в узел – это означает, что подходящего для вставки открытого контейнера нет.
- Если подходящий контейнер не был найден, то новый узел (контейнер) вставляется в дерево с оставшимся местом после вычета вставляемого элемента по правилам бинарного дерева поиска с обновлением наименьшей заполненности, наименьшего номера и ссылки при необходимости. При нарушении баланса после каждой из операций производится перебалансировка.

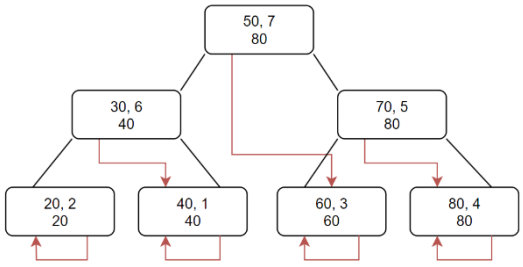

Рис. 2. 2-ая модификация AVL-дерева в алгоритме FFD.

В худшем случае поиск нужного контейнера происходит также за высоту дерева, то есть за $O(\log_2 n)$.

2.3. Результаты исследования

Исследуемые структуры данных и эвристические алгоритмы написаны на языке программирования Python (версия 3.9), для оценки времени использовалась библиотека timeit. Набор данных для исследования подробно описан в [8]. На графиках «рис. 3» представлены результаты временной оценки выполнения программы для BFD и FFD на AVL-деревьях.

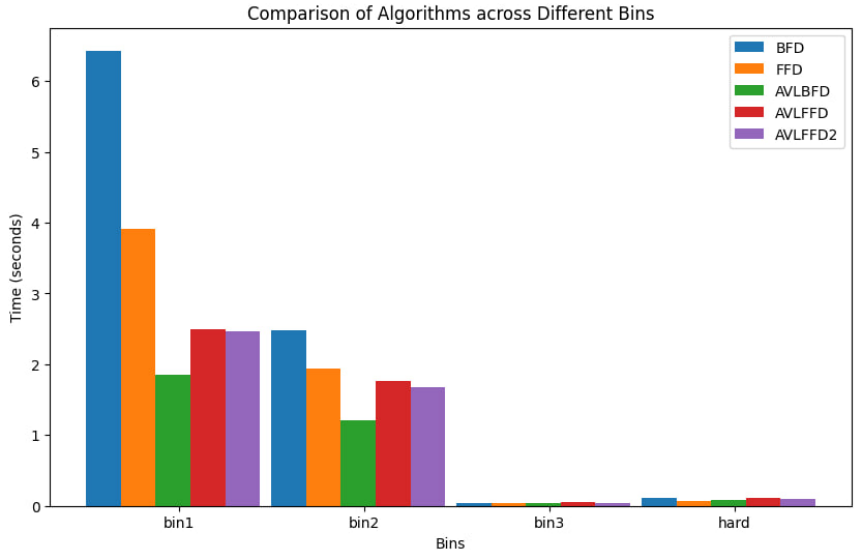

Рис. 3. Столбчатая диаграмма затрачиваемого времени.

Из полученных результатов можно сделать вывод, что наиболее быстрыми структурами данных для исследуемых алгоритмов на классических наборах данных являются AVL для BFD и AVLFFD2 (2-ая модификация) для FFD.

3. Заключение

В данной работе исследовалось ускорение жадных эвристических алгоритмов при применении сбалансированных деревьев поиска. Представлена 2-ая модификация сбалансированного бинарного дерева для алгоритма FFD. Проведено сравнение среднего времени, затрачиваемого на поиск приближенного решения различных структур деревьев и классических алгоритмов без их использования. Из полученных данных можно с уверенностью сделать вывод, что AVL-деревья значительно ускоряют классические алгоритмы и что среди модификаций наиболее быстрой является AVLFFD2.

Исследование выполнено при частичной финансовой поддержке РНФ в рамках научного проекта 22-71-10131.

Список литературы

1. Garey M.R., Graham R.L., Ullman J.D. Worst-case analysis of memory allocation algorithms // Proceedings of the fourth annual ACM symposium on Theory of computing. New York: Association for Computing Machinery, 1972. P. 143-150. doi: 10.1145/800152.804907.
2. Ram B. The pallet loading problem: A survey // International Journal of Production Economics, 1992. Vol. 28, No. 2. P. 217=225. doi: 10.1016/0925-5273(92)90034-5.
3. Christensen H.I., Khan A., Pokutta S., Tetali P. Multidimensional bin packing and other related problems: A survey // Computer Science Review. 2016. Vol. 24. 34 p. doi: 10.1016/j.cosrev.2016.12.001.
4. Christensen H.I., Khan A., Pokutta S., Tetali P. Multidimensional bin packing and other related problems: A survey // Computer Science Review. 2016. Vol. 24. 34 p. doi: 10.1016/j.cosrev.2016.12.001.

5. Seiden S.S. On the online bin packing problem // Journal of the ACM (JACM). 2002. Vol. 49, No. 5. P. 640-671.
6. Фуремс Е.М. Обратная задача об упаковке в контейнеры при наличии качественных критериев // Системы поддержания принятия решений. 2016. С. 35-37.
7. https://ics.uci.edu/~goodrich/teach/cs165/notes/BinPacking.pdf (дата обращения 18.01.2024).
8. Барашов Е.Б., Егоркин А.В., Лемтюжникова Д.В., Посыпкин М.А. Анализ эффективности алгоритма редукции в решении задачи об упаковке в контейнеры // Системы и средства информатики. 2023. Т. 33, Вып. 3. С. 61-75.