АЛГОРИТМЫ РЕГИОНАЛЬНОГО ПОИСКА НА ОСНОВЕ R-ДЕРЕВЬЕВ

А.В. Скворцов, ТГУ
skv@sibgeoi.tomsk.su

A new approach to using of R-tree spatial index structure is concerned: primary global constructing of effective R-tree and subsequent its using with the aid of common dynamic algorithms. Suggested global strategy is reduced to the problem of separation of rectangular object set to K parts with minimal geometric intersection. For resulting separation improving and common dynamic algorithm efficiency raising the sharpening algorithm is suggested.

Задача регионального поиска является одной из базовых вычислительной геометрии. Так как форма различных объектов может значительно различаться, то для упрощения алгоритмов регионального поиска обычно для каждого из объектов вычисляется минимальный объемлющий прямоугольник со сторонами, параллельными осям координат. Тогда задача регионального поиска определяется как задача нахождения всех прямоугольных объектов, имеющим непустое перекрытие с прямоугольным регионом, являющимся запросом.

В настоящее время наиболее эффективной структурой, позволяющей обрабатывать неточечные объекты при эффективном размещении данных во вторичной памяти, считаются R-деревья [1] и ряд их модификаций, в частности, наиболее эффективные R*-деревья [2].

Структура R-дерева является сбалансированным по высоте деревом с индексными записями в листьях, содержащими указатели на объекты данных. Узлам дерева соответствуют дисковые страницы, если индекс располагается на диске. Структура является полностью динамической, вставка и удаление объектов может свободно перемежаться с запросами на поиск, при этом не требуется периодической реорганизации данных.

Данная индексная структура предназначена для эффективного выполнения регионального поиска, поэтому основная идея дерева заключается в разбиении пространства в каждом узле дерева на минимально пересекающиеся группы, что позволяет при выполнении поиска эффективно отсекать неверные ветви поиска. Задача разбиения в общем случае, видимо, по меньшей мере NP-полна, так как в настоящее время даже для двух групп не известно полиномиального алгоритма [1]. Поэтому во всех алгоритмах построения R-деревьев используются эвристические подходы, дающие на реальных данных далеко не оптимальные разбиения.

Высокая степень перекрытия входов в узлах дерева происходит из-за динамичности алгоритмов его построения, когда при вставке очередного элемента его необходимо за минимальное время разместить и при необходимости перестроить дерево. При этом глобальная оптимизация дерева на каждом шаге не производится.

На практике взаимодействие с индексными структурами состоит из двух основных этапов: начального построения дерева и последующей оперативной работы. При этом во многих случаях во время второго этапа производится только поиск объектов без вставки новых и удаления старых.

В описываемой работе предлагается учесть специфику начального построения структуры, построив вначале более эффективное R-дерево, а на последующих этапах использовать обычные алгоритмы для работы с R-деревьями. В частном случае, когда вначале ничего не строится, получается обычное R-дерево.

1. Глобальные алгоритмы

Во всех существующих структурах R-деревьев основным параметром, определяющим скорость поиска, считается степень взаимного пересечения потомков в узлах дерева [1,2]. Именно поэтому в алгоритмах построения R-деревьев одной из главных целей является минимизация такого пересечения. Это позволяет при выполнении регионального поиска на ранних стадиях эффективно отсекать тупиковые ветви работы алгоритма.

В связи с этим можно ввести следующее определение:

Определение. Построенная для заданного набора объектов структура R-дерева называется *оптимальной*, если сумма площадей попарных перекрытий входов во всех нелистовых узлах дерева является минимальной среди всех возможных структур R-деревьев:

$$\min_{R\text{-деревья}} \sum_{i \neq j} S(R_i \cap R_j),$$ где S – функция площади.

Автором предлагается следующая стратегия построения R-дерева, близкого к оптимальному.

Алгоритм Построить_Дерево. По заданному множеству из N объектов E_i строит R-дерево с параметрами m, M [1].

Шаг 1. Построить корень дерева и определить количество уровней дерева $L(N) = \lceil \log_M N \rceil$.

Шаг 2. Вызвать алгоритм **Построить_Узел**, передав ему в качестве параметра корень дерева и всё множество объектов.

Алгоритм Построить_Узел. По заданному множеству из N объектов E_i и узлу T строит потомков узла, имеющего уровень L (L=1 для листьев).

Шаг 1. Если L=1, то заполнить узел объектами E_i и закончить.

Шаг 2. Вычислить количество потомков у данного узла $K = \left\lceil N^{1/L} \right\rceil$.

Шаг 3. Вызвать алгоритм **Разделить_На_Группы** для разбиения всего множества объектов на K групп, минимизируя сумму попарных пересечений групп. При этом в алгоритм необходимо передать номер уровня L для правильной оценки минимального и максимального допустимого количества объектов в группах. Для каждой полученной группы построить пустой узел - потомок текущего узла, и вызвать для него рекурсивно алго-

ритм **Построить_Узел** с этой группой в качестве множества.

Наиболее сложной частью данной стратегии является алгоритм **Разделить_На_Группы**, от трудоёмкости которого зависит общая сложность алгоритма. В работе предлагается 3 варианта данного алгоритма.

2. Алгоритмы разбиения множества объектов на минимально пересекающиеся группы

Во все существующие алгоритмы построения R-деревьев входят алгоритмы разбиения множества объектов на две части с минимальным пересечением. При этом выбирается два базовых объекта, последовательными присоединениями к которым всех остальных строятся необходимые группы.

Обобщая данный алгоритм на случай нескольких групп, можно выбрать по одному базовому объекту для каждой группы и, используя аналогичные итеративные алгоритмы добавления оставшихся объектов, произвести разбиение. На этом принципе построен *базовый* алгоритм разбиения.

Одним из недостатков его работы является недостаточно высокое качество разбиения на неравномерных распределениях, а также в случае, когда N незначительно превосходит K (при построении нижних уровней дерева). Это проявляется в ситуациях, когда наступает такой этап работы алгоритма, что некоторые группы уже заполнены, но при этом в соответствии с критерием выбора в них необходимо добавить очередной объект. Но так как это невозможно, то приходится добавлять объекты в некоторые другие группы, что в конечном итоге приводит к сильному перекрытию образовавшихся групп.

В качестве одного из вариантов решения данной проблемы автором предлагается временно удалять объекты, которые нельзя поместить в требуемые группы, из списка участвующих в глобальном алгоритме. При этом после окончания его работы все эти объекты необходимо поместить в дерево, используя уже обычные динамические алгоритмы построения R-дерева.

Как показало проведённое автором экспериментальное исследование работы такого алгоритма, количество объектов, которые нельзя разместить в требуемые группы, чаще всего незначительно, и они эффективно размещаются по окончании работы глобального алгоритма динамическим способом.

Трудоёмкость глобального алгоритма построения R-дерева, использующего базовый алгоритм разбиения, составляет $O(N \log N)$.

Как показало экспериментальное исследование, базовый алгоритм хорошо ведёт себя на равномерных распределениях непересекающихся объектов. Но на неравномерных распределениях часто происходит переполнение строящихся групп, что приводит к необходимости помещения очередных объектов в неоптимальные группы. В итоге получаются недопустимо большие перекрытия построенных групп. Поэтому автором предлагается другой алгоритм, в котором вначале на основе клеточного разбиения плоскости строятся группы объектов и только затем анализируются их количество и наполнение. В случае необходимости полученные группы делятся

пополам или объединяются с другими.

Одним из недостатков работы такого алгоритма является увеличение времени работы алгоритма на неравномерных распределениях, так как при этом часто происходит недополнение и переполнение групп, а также приходится выполнять перемещение объектов по группам. В худшем случае алгоритм может иметь квадратичную сложность в случае многократного перемещения объектов по группам.

Трудоёмкость глобального алгоритма построения R-дерева, использующего клеточный алгоритм разбиения, составляет $O(N^2)$.

Другой предлагаемый вариант алгоритма разбиения основан на стратегии «Разделяй и властвуй», в котором производится последовательное разбиение всего множества на две части до тех пор, пока всё множество не окажется разбито на требуемое число частей. Трудоёмкость глобального алгоритма построения R-дерева, использующего алгоритм разбиения «Разделяй и властвуй», составляет $O(N \cdot \log^2 N)$.

3. Уточнение разбиения

Повышение качества получаемого разбиения множества объектов на группы является главным направлением совершенствования алгоритмов работы с R-деревьями. Поэтому для повышения качества разбиения множества объектов на группы предлагается алгоритм уточнения полученных групп, позволяющий в ряде случаев значительно улучшить качество получаемого разбиения. Такой подход может использоваться как в обычных динамических алгоритмах манипуляции с R-деревьями (в алгоритме Расщепить_Узел при вставке объектов в дерево), так и в алгоритмах разбиения, применяемых в глобальных алгоритмах. Для реализации данного подхода предлагается следующий алгоритм уточнения разбиения…

Теорема 4 (трудоёмкость алгоритма уточнения). Трудоёмкость алгоритма уточнения разбиения составляет $O(N)$. *Без доказательства.*

Следствием данной теоремы является то, что порядок трудоёмкости алгоритмов построения R-деревьев, использующих алгоритм уточнения, не изменяется по сравнению с алгоритмами без уточнения.

4. Практический анализ глобальных алгоритмов

Для количественного анализа построенных алгоритмов автором было проведено моделирование работы различных алгоритмов построения R-деревьев на различных распределениях объектов и поисковых регионах. При этом анализировались такие параметры, как время построения R-дерева, процент попадания при поиске (отношение количества узлов, в которых найден хотя бы один объект, к их общему количеству, проверенному при выполнении регионального поиска) и процент перекрытия потомков (отношение суммы площадей попарных пересечений входов в узлах к общей площади всех входов в процентах) [1].

Для сравнения алгоритмов в различных условиях в экспериментах строились наборы данных с 10 000 объектами, располагавшимися в единичном квадрате $[0,1)^2$, с различными характеристиками (базовые типы распределений взяты из работы [2]).

Сравнение экспериментальных данных подтвердило правильность выбранной стратегии на минимизацию взаимных перекрытий потомков узлов в R-дереве, так как практически везде меньшему проценту перекрытия потомков соответствует больший процент попадания в нужные узлы при поиске.

Предложенные в работе базовый и клеточный алгоритмы наиболее хорошо работают на равномерных распределениях, когда объекты мало пересекаются между собой. Но на неравномерных распределениях заметно ухудшение работы алгоритмов по сравнению с существующими вариантами R-деревьев. В то же время алгоритм «Разделяй и властвуй» одинаково хорош как на равномерных, так и на неравномерных распределениях. Практически во всех тестах он обходит остальные алгоритмы по качеству получающегося построения. Только при построении R-дерева на наборе данных с большими объектами этот алгоритм строит дерево, уступающее по проценту перекрытия классическому алгоритму построения R-дерева.

Из результатов экспериментального моделирования можно сделать вывод, что предложенный алгоритм уточнения разбиения позволяет несколько повысить качество получаемой структуры R-дерева. При использовании данного подхода в применении к обычному R*-дереву достигается увеличение попадания на 5-10% и сокращение перекрытия потомков на 7-15%. Применение же локального уточнения в составе глобальных алгоритмов практически не даёт никакого эффекта, за исключением уменьшения перекрытия потомков на регулярном наборе данных.

Литература

Guttmann A. R-trees: A Dynamic Index Structure For Spatial Searching // Proc. ACM SIGMOD Int. Conf. on Management of Data, 1984, p. 47-57.

Beckmann N., Kriegel H., Schneider R., Seeger B. The R* tree: An Efficient and Robust Access Method for Points and Rectangles // ACM, 1990, p. 322-331.

Kriegel H., Schiwietz M., Schneider R. Seeger B. Performance comparison of point and spatial access methods // Proc. Symp. On the Design and Implementation of Large Spatial Databases, Santa Barbara, 1989, Lecture Notes in Computer Science.

ПРОБЛЕМЫ ИНТЕГРАЦИИ ГИС И САПР

А.В. Скворцов, ТГУ
skv@sibgeoi.tomsk.su

The analysis of two branches of graphical systems that deal with territorially distributed information (GIS and CAD) is performed. A brief review of data structures, attributive information processing principles, 3D object modeling