Тема 4. Методы поиска

4.3. Логарифмический поиск в динамических таблицах

4.3.4. Цифровой поиск

Рассмотренные ранее методы поиска были основаны на сравнении имен, т. е. на предположении о том, что на пространстве имен S определен линейный порядок и что время, требующееся для сравнения двух имен, не зависит от мощности пространства имен S. Методы *цифрового поиска* используют тот факт, что имя или его представление можно всегда интерпретировать как последовательность символов над некоторым конечным алфавитом A (в крайнем случае, над двоичным алфавитом $\{0, 1\}$). Тогда естественным порядком на S является лексикографический порядок, индуцированный линейным порядком на A. Предполагается, что исход сравнения двух символов (не имен) получается за время, не зависящее от мощности пространства имен и от мощности n исследуемой таблицы T.

В качестве примера цифрового поиска рассмотрим построение таблицы для таких последовательностей десятичных цифр: 2448, 28, 31415, 64, 672, 67430, которые можно представить в виде *дерева цифрового поиска* (рис. 4.14). В дереве цифрового поиска последовательность цифр для каждого имени получается при прохождении некоторого пути от корня дерева к листу; идентичные префиксы (начальные подпоследовательности) различных имен объединяются до тех пор, пока это возможно.

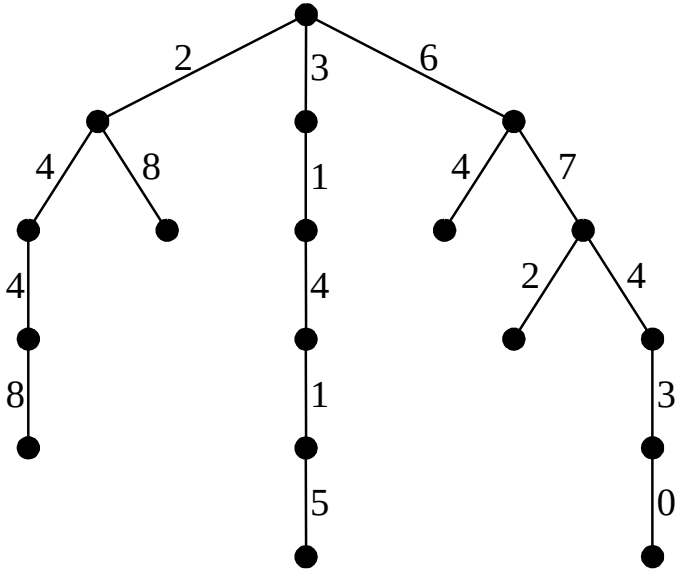

Рис. 4.14. Дерево цифрового поиска
для таблицы T = {2448, 28, 31415, 64, 672, 67430}

Поиск имени z с цифрами $d_1, d_2, ..., d_l$ начинается в корне дерева и продолжается сравнением самой левой еще не сравнивавшейся цифры d_j с метками всех ветвей, выходящих из текущей вершины. Если d_j совпадает с меткой некоторой ветви, то продолжается движение по этой ветви до следующей вершины. Если ни одна метка не совпадает с d_j, то поиск завершается безуспешно. Поиск завершается успешно, если все цифры имени z найдены.

Особенностью рассмотренного примера является то, что ни одно из имен в таблице не является префиксом другого имени. Для многих приложений это нехарактерно. Можно предложить два метода, позволяющих строить деревья цифрового поиска для имен, которые являются префиксами других имен в таблице. Первый метод заключается в том, что к каждой вершине дерева добавляется специальный разряд со значением «конец имени». Таким способом отмечается конец слова в этой вершине.

Если в рассмотренном примере необходимо, чтобы последовательности 314 и 31415 были именами (314 является префиксом имени 31415), то в вершинах, к которым ведут ребра, помеченные цифрами 4 и 5, надо установить этот разряд равным 1, а во всех других вершинах вдоль пути 31415 надо установить этот разряд равным 0. Второй метод заключается в том, что имена модифицируются таким образом, чтобы ни одно имя не могло быть префиксом другого, например, к концу каждого имени присоединить пробел или другой символ, не принадлежащий алфавиту. Очевидно, что второй метод может увеличить число вершин в дереве цифрового поиска по сравнению с первым методом.

Дерево цифрового поиска можно представить в памяти различными способами. Для эффективной реализации представление должно позволять одновременное сравнение символа в имени с метками всех ветвей, выходящих из вершины, поскольку поочередный просмотр всех выходящих из вершины ветвей приведет к существенному увеличению времени поиска, и такой метод в большинстве случаев будет медленнее бинарного поиска. Один из способов превращения многократного просмотра ветвей в элементарную операцию, которая может быть выполнена за время, не зависящее от числа ветвей, заключается в следующем. Каждой вершине дерева сопоставляется вектор с компонентами для каждого символа алфавита независимо от того, имеется ли в таблице имя, использующее эту компоненту. Каждая цифра (символ) в подлежащем поиску имени используется как индекс или указатель соответствующей компоненты вектора, которая определяет следующий шаг поиска. Представленное таким способом дерево цифрового поиска называется *бором*. Бор, соответствующий дереву цифрового поиска (см. рис. 4.14), представлен на рис. 4.15. Для экономии памяти последовательность вершин вдоль пути, где не происходит ветвление (например, путь 1415), сжата в один узел. Когда такой узел достигается, оставшиеся символы искомого имени должны сравниваться с символами, содержащимися в узле.

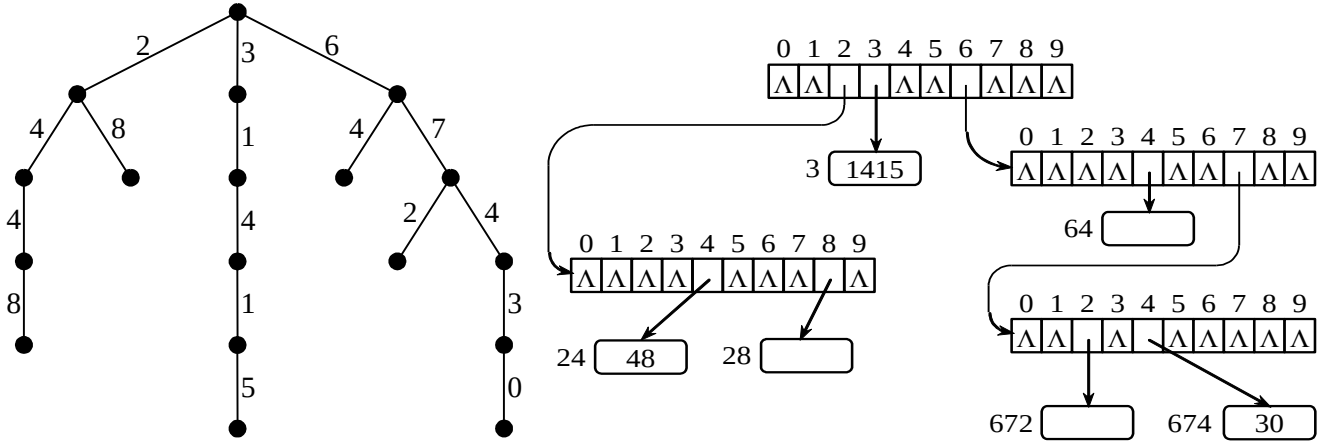

Рис.4.15. Представление дерева цифрового поиска в виде бора

В качестве примера рассмотрим поиск по бору имени 2448. Цифра 2 используется как индекс в векторе, сопоставленном корню (остальные компоненты вектора не рассматриваются и время выполнения этой операции не зависит от размера алфавита). Найденный элемент вектора является указателем другого узла, в котором в качестве индекса используется цифра 4. Найденная компонента является указателем специального узла, содержащего суффикс 48 искомого имени. Соединение префикса 24, соответствующего пути в этот узел, и суффикса 48, который хранится явно в специальном узле, дает полное искомое имя 2448. Таким образом, имя 2448 найдено за два шага, так как исследовано минимальное число цифр, которые могут выделить это имя просмотром слева направо из всех остальных имен в таблице. При поиске имени 746 пустой указатель для седьмой компоненты вектора, сопоставленного корню, определяет поиск как безуспешный, т. е. за один шаг дается ответ, что этого имени нет в таблице.

Рассмотренный пример наглядно показывает два характерных свойства боров: они допускают быстрый поиск, особенно в случае безуспешных поисков, и имеют тенденцию к излишнему расходу памяти. Причиной неэффективного использования памяти является то, что схема размещения поля для каждого символа алфавита в каждом узле приспособлена для представления всего пространства имен, а не содержимого конкретной таблицы, т. е. бор обычно содержит место для многих имен, не принадлежащих таблице. В общем случае излишний объем памяти возникает в узлах вблизи нижней части бора, а узлы около корня имеют тенденцию к полноте. Поэтому на практике с целью экономии памяти часто используется комбинированный вариант, заключающийся в объединении методов поиска по борам для отдельных префиксов имен с другими методами поиска для суффиксов.

Поскольку методы цифрового поиска основаны на сравнении между символами в представлении имен, а на каждый символ представления требуется только по одному такому сравнению, время поиска пропорционально средней длине имен. При алфавите из c символов средняя длина n разных имен не может быть меньше, чем $\log_c n$, т. е. цифровой поиск может выполняться за время, логарифмическое относительно размера таблицы, хотя и не гарантирует этого в худшем случае. Коэффициент пропорциональности зависит от многих факторов, но на практике в большинстве случаев цифровой поиск обычно выполняется быстрее бинарного, так как многопутевое разветвление от одного символа (для алфавита с c символами это обычно c-путевое разветвление) в основном не длиннее, чем двухпутевое разветвление при сравнении двух имен в бинарном поиске.

Цифровой поиск отличается от других методов также тем, что для данного множества имен существует только одно дерево цифрового поиска, т. е. не возникает проблемы построения оптимального дерева. Различные множества имен приводят к борам разного вида.

Операции включения и исключения в бору прямые, поэтому можно считать, что цифровой поиск является достаточно подходящим методом для динамических таблиц.