Tạp chí Khoa học ĐHQGHN, Khoa học Tự nhiên và Công nghệ 27 (2011) 14-21

Ứng dụng cây QR tạo chỉ mục trong cơ sở dữ liệu không gian

Dư Phương Hạnh*

Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội, 144 Xuan Thủy, Hà Nội, Việt Nam

Nhận ngày 7 tháng 01 năm 2011

Tóm tắt. Bài báo này đề cập đến khái niệm và một số phương pháp đánh chỉ mục trong cơ sở dữ liệu không gian (spatial datadase – SDB). Là một trong những mô hình cơ sở dữ liệu được quan tâm hiện nay, SDB cho phép xử lý các đối tượng dữ liệu không gian, chẳng hạn dữ liệu bản đồ, dữ liệu multimedia... để từ đó có thể xây dựng nên những kho dữ liệu không gian. Một trong những bài toán cơ bản trong SDB chính là việc tối ưu hoá quá trình lưu trữ dữ liệu và truy vấn. Trong bài báo này, chúng tôi sẽ trình bày về hai phương pháp đánh chỉ mục điển hình liên quan đến vấn đề đánh chỉ mục giải bài toán trên, R-tree và Q-tree. Từ đó, ý tưởng kết hợp hai phương pháp này sẽ chính là định hướng chủ đạo cho việc tối ưu hoá lưu trữ dữ liệu cũng như truy vấn trên cơ sở dữ liệu không gian.

Từ khóa: Spatial database, spatial indexing, R-tree, Q-tree, QR-Tree.

1. Giới thiệu

Các nghiên cứu về công nghệ cũng như ứng dụng trong lĩnh vực cơ sở dữ liệu (CSDL) đang tăng trưởng với một sức mạnh đáng kinh ngạc. Cùng với sự tăng trưởng nhanh chóng của lượng thông tin cũng như sự đa dạng về thể loại thông tin cần lưu trữ và xử lý, chúng ta ngày càng nhận ra những hạn chế của các Hệ quản trị cơ sở dữ liệu quan hệ truyền thống, và nhu cầu cần phải có các hệ quản trị cơ sở dữ liệu với các dịch vụ phù hợp chính là yếu tố thúc đẩy những nghiên cứu mới trong lĩnh vực này. Một trong các mô hình cơ sở dữ liệu được quan tâm nhất hiện nay chính là mô hình cơ sở dữ liệu không gian - Spatial DataBase (SDB) xử lý các đối tượng dữ liệu không gian, chẳng hạn dữ liệu bản đồ, dữ liệu multimedia... và mở rộng hơn nữa là kho dữ liệu không gian - Spatial Data Warehouse (SDW). Các nghiên cứu trên lĩnh vực này đã thu được rất nhiều thành tựu, tuy nhiên cũng còn không ít khó khăn và thách thức đòi hỏi phải có các giải pháp mới.

Bài báo này trình bày một phương pháp đánh chỉ mục trên SDB, là sự kết hợp giữa hai phương pháp đánh chỉ mục phổ biến là Q-tree và R-tree, kết hợp các ưu điểm của cả hai phương pháp này cũng như giảm thiểu nhược điểm của chúng, nhằm tăng hiệu suất thực thi các phép toán.

2. Khái niệm cơ bản

Phần này sẽ được tập trung trình bày những khái niệm cơ bản liên quan đến mô hình SDB.

2.1. Dữ liệu không gian

Thuật ngữ *dữ liệu không gian* (spatial data) được sử dụng theo nghĩa rộng, bao gồm các

* ĐT: 84-4-37547813.
E-mail: hanhdp@vnu.edu.vn

điểm đa chiều, các đường thẳng, hình khối... và các đối tượng hình học nói chung. Mỗi đối tượng dữ liệu này chiếm một *vùng không gian* (spatial extent) được đặc trưng bởi hai thuộc tính *vị trí* (location) và *biên* (boundary). Dưới góc nhìn từ một hệ quản trị cơ sở dữ liệu, có thể phân chia dữ liệu không gian thành hai kiểu: *dữ liệu điểm (*point data*)* và *dữ liệu vùng* (region data) [1]

Dữ liệu* điểm*; Với kiểu dữ liệu này, không gian ứng với một *điểm* được đăc trưng bởi tọa độ của nó; theo trực giác thì nó không chiếm một vùng không gian hay một đơn vị thể tích nào cả. *Dữ liệu điểm* là tập hợp các *điểm* trong không gian nhiều chiều, được lưu trữ trong CSDL dựa trên các tọa độ được tính toán trực tiếp, hoặc được sinh ra nhờ quá trình chuyển hóa dữ liệu thu được từ các phép đo khiến cho việc lưu trữ và thực hiện truy vấn trở nên dễ dàng hơn. Chẳng hạn *Raster data* là một ví dụ dữ liệu điểm được lưu trữ trực tiếp thông qua các bit maps hoặc pixel maps (chẳng hạn như ảnh vệ tinh, hoặc phim điện não đồ 3 chiều, …). Trong khi đó, *feature vectors data* được lưu trữ thông qua các dữ liệu được trích chọn, chuyển đổi từ một đối tượng dữ liệu điểm (thu được từ ảnh, văn bản...). Có thể thấy rằng, sử dụng các dữ liệu đã được biểu diễn để trả lời các truy vấn luôn dễ dàng hơn sử dụng ảnh hoặc ký hiệu nguyên bản.

Dữ liệu vùng: được xác định dựa trên tập các vùng không gian (spatial extents), trong đó mỗi vùng được đặc trưng bởi hai thuộc tính *vị trí* và *biên*. *Dữ liệu vùng* được lưu trữ trong CSDL như một đối tượng hình học đơn giản, xấp xỉ đúng với đối tượng dữ liệu thực sự. Việc mô tả các phép xấp xỉ đó được đặc tả thông qua vector dữ liệu, được xây dựng từ các điểm, các đoạn thẳng, các hình đa giác, hình cầu, hình ống... Rất nhiều ví dụ dữ liệu vùng được đưa ra trong các ứng dụng địa lý, chẳng hạn đường xá, sông ngòi có thể được biểu diễn dưới dạng tập hợp của các đoạn thẳng; quốc gia, thành phố có thể được biểu diễn dưới dạng các hình đa giác...

2.2. Các phương pháp truy vấn phổ biến trên dữ liệu không gian

a) Truy vấn theo phạm vi không gian (Spatial range queries):

Giả sử chúng ta có yêu cầu truy vấn "*Đưa ra tên tất cả các thành phố xuất hiện trong phạm vi 1000km quanh Hà Nội*" hoặc "*Đưa ra tên các con sông chảy qua khu vực Bắc Bộ*". Một truy vấn theo kiểu này sẽ chứa một vùng liên đới (với các thuộc tính vị trí và biên tương ứng), và kết quả trả về sẽ là một vùng bao trùm phạm vi không gian đã chỉ ra trong truy vấn hoặc là một tập hợp các vùng thuộc trong phạm vi không gian đã chỉ ra trong truy vấn. Kiểu truy vấn theo phạm vi được sử dụng trong các ứng dụng trên nhiều lĩnh vực đa dạng bao gồm truy vấn quan hệ, truy vấn GIS, truy vấn CAD/CAM [1]

b) Truy vấn dựa trên các láng giềng gần nhất (Nearest neighbor queries):

Với một yêu cầu chẳng hạn như "Đưa ra tên 19 thành phồ gần Hà Nội nhất", chúng ta thường muốn kết quả trả về được sắp xếp theo thứ tự nào đó về khoảng cách. Đây là dạng truy vấn được sử dụng nhiều nhất đối với dữ liệu multimedia. Trong trường hợp này, dữ liệu multimedia (chẳng hạn là ảnh) được biểu diễn dưới dạng một điểm, và dữ liệu tương tự cần tìm kiếm được tính toán theo khoảng cách gần nhất tới điểm biểu diễn đối tượng truy vấn. [1]

c) Truy vấn liên kết không gian (Spatial join queries):

Các yêu cầu truy vấn thông thường thuộc dạng này là "Đưa ra các thành phố cách nhau không quá 200km" hoặc "Đưa ra tên các con phố gần hồ". Các dạng truy vấn này thường rất mất thời gian để tính toán. Nếu chúng ta xem xét một quan hệ trong đó mỗi một phần tử là

một điểm biểu diễn một thành phố hoặc một cái hồ thì truy vấn trên có thể được thực hiện bằng phép nối quan hệ này với chính nó với điều kiện nối chỉ ra khoảng cách giữa hai phần tử tương ứng. Đương nhiên, nếu các thành phố và hồ được biểu diễn chi tiết hơn và có vùng không gian của chúng, ngữ nghĩa của truy vấn (chúng ta tìm kiếm hai thành phố mà trung tâm của chúng cách nhau 200km hay hai thành phố mà biên của chúng cách nhau 200km) và việc thực thi truy vấn đều trở nên phức tạp hơn nhiều. [1]

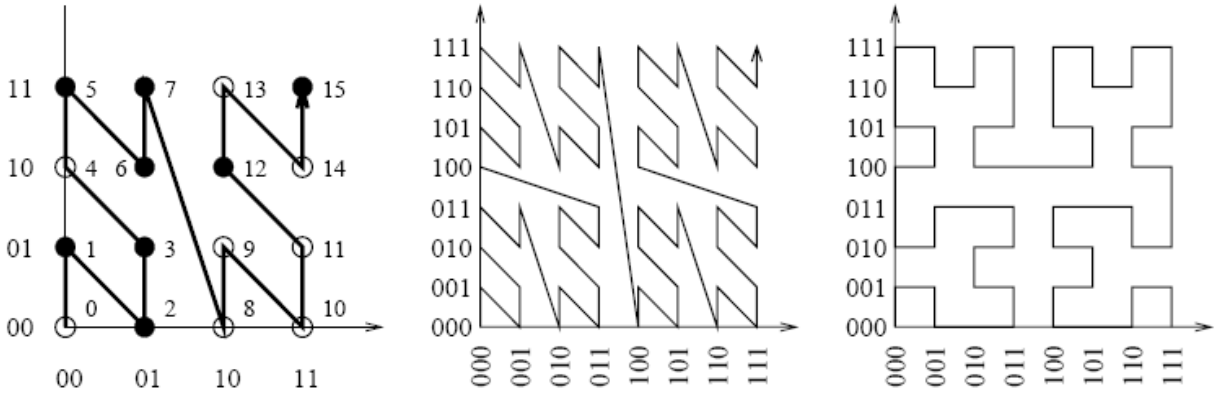
3. Q-Tree, R-Tree và QR-Tree

Rất nhiều cấu trúc đánh chỉ số trên CSDL không gian đã được đề xuất, một số được thiết kế chủ yếu dành cho tập dữ liệu điểm mặc dù chúng cũng có thể áp dụng cho kiểu dữ liệu vùng. Cấu trúc index dành cho dữ liệu điểm có thể kể tới Grid files, HB tree, KD tree, Point Quad tree và SR tree... Các kiến trúc khác như Region Quad tree, R tree và SKD tree áp dụng cho dữ liệu vùng, tuy nhiên chúng cũng có thể áp dụng cho dữ liệu điểm [2, 3].

Region Quad tree (Q-tree) và R-tree là hai hướng tiếp cận khác nhau và có rất nhiều biến thể. Hiện chưa có được sự nhất trí rằng cấu trúc đánh chỉ số nào là tốt nhất, tuy nhiên R tree là cấu trúc được sử dụng rộng rãi và đã xuất hiện trong các bản DBMS thương mại, do tính đơn giản và khả năng áp dụng cho cả hai dạng dữ liệu điểm và vùng.

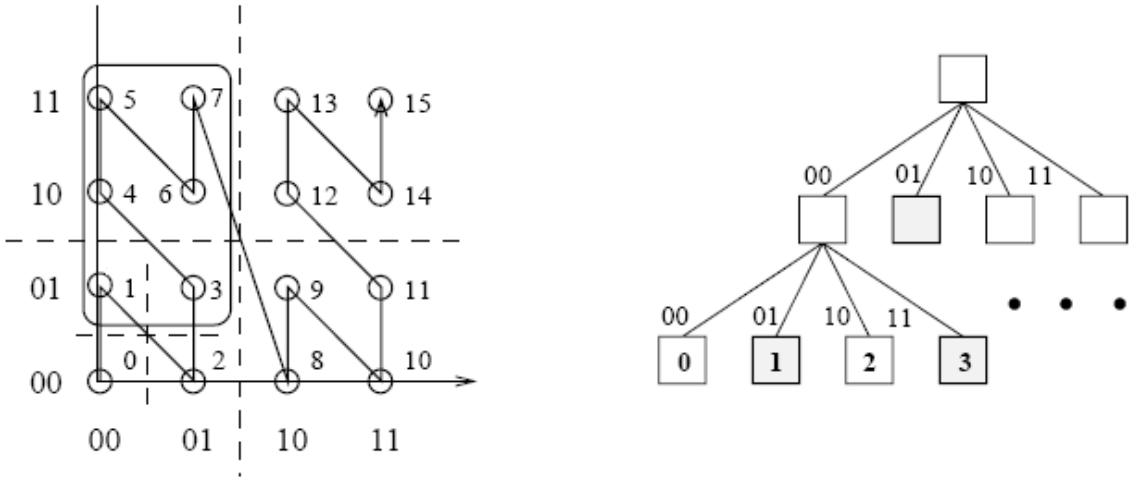
3.1. Q - tree

Q - tree [3] là phương pháp đánh chỉ số dựa trên đường cong *Space-Filling Curves* để sắp xếp các điểm dữ liệu. Việc đánh chỉ số được thực hiện dựa trên việc phân chia không gian dữ liệu một cách đệ quy, nhưng khác với R-tree, phương pháp này được thực hiện độc lập đối với tập dữ liệu thực sự. Space-Filling Curves được xây dựng dựa trên giả thiết rằng mọi giá trị thuộc tính nào đó đều có thể biểu diễn bởi một số bit xác định nào đó gọi là k bit, do đó số lượng các giá trị thuộc về cùng một chiều dữ liệu có thể đạt tới nhiều nhất là 2^k. Để đơn giản, hình vẽ dưới đây mô phỏng một tập dữ liệu 2-chiều mặc dù thực tế là phương pháp này có thể áp dụng với dữ liệu có số chiều bất kỳ. Hình vẽ thứ nhất sử dụng 2 bit để biểu diễn giá trị thuộc tính; hình thứ hai sử dụng 3 bit; và hình thứ ba là đường cong Hilbert với 3 bit.



Hình 1. Space-Filling Curves.

Trên ý tưởng này, Q-tree là phương pháp phân chia một cách đệ quy không gian dữ liệu thành các góc phần tư, được minh họa trong hình vẽ 3: Trong cấu trúc này, mỗi nút có 4 con lần lượt ứng với các góc phần tư 00 (góc phần tư bên trái phía dưới), 01 (góc phần tư bên trái phía trên), 10 (góc phần tư bên phải phía dưới) và 11 (góc phần tư bên phải phía trên). Trên hình vẽ, chúng ta có thể thấy rằng nếu không gian dữ liệu không được phân bố một cách đối xứng thì cây Q-tree sẽ bị lệch, bởi vì Q-tree không phải là một cấu trúc cây cân bằng, do đó trên những tập dữ liệu lớn, hiệu suất truy cập dữ liệu sẽ kém hiệu quả.

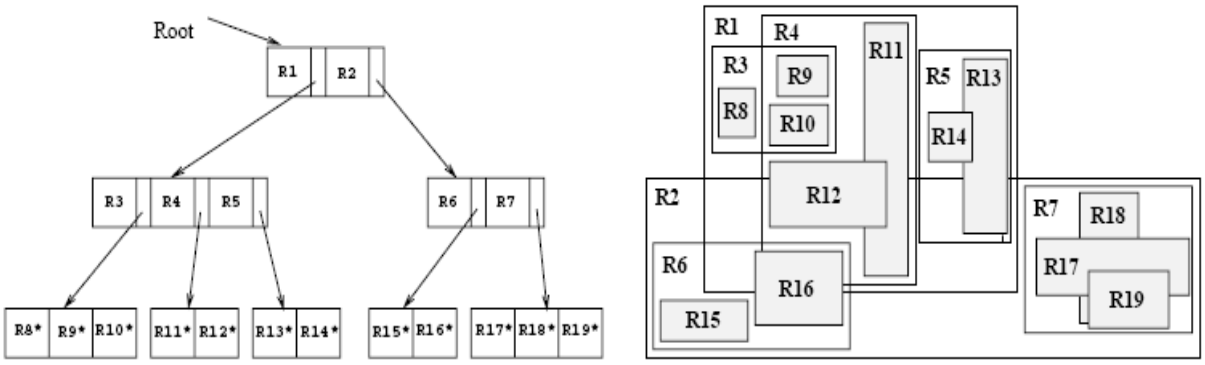


Hình 2. Cấu trúc đánh chỉ mục Q-tree.

Một mặt khác, trong các ứng dụng đòi hỏi việc lưu trữ dữ liệu có tính chất liên tục (chẳng hạn dữ liệu về một đối tượng chuyển động) thay vì các dữ liệu xác định, chúng ta gặp phải một vấn đề rất khó để cân nhắc bởi vì: việc sử dụng cây Q-tree có độ sâu càng lớn thì độ chính xác biểu diễn dữ liệu càng tốt, tuy nhiên nó lại khiến cho việc xây dựng cấu trúc này trở nên kém hiệu quả trên cả phương diện không gian lưu trữ và thời gian xử lý các thao tác.

3.2. R-tree

R-tree là phương pháp phân chia không gian dữ liệu thành các khối có thể lồng nhau hoặc chồng chéo lên nhau, được minh họa trong hình 4. Đơn giản nhất, hình khối thường được sử dụng là hình chữ nhật nhỏ nhất chứa dữ liệu (Minimum Bounding Rectangle – MBR). Như vậy, chính các MBR được lưu trữ trên cấu trúc cây chứ không phải bản thân dữ liệu. CÁc nút không phải lá được biểu diễn bởi cặp (R, child-pointer) trong đó R là MBR của đối tượng và child-pointer là con trỏ trỏ tới nút con; các nút là được biểu diễn bởi cặp (R, obj-pointer) trong đó R là MBR của đối tượng và obj-pointer là con trỏ trỏ tới mô tả chi tiết của đối tượng. Mỗi nút trong cây tương ứng với một trang bộ nhớ. Và mặc dù các MBR có thể chồng chéo lên nhau, tức là các nút có thể chứa dữ liệu giống nhau, nhưng mỗi đối tượng dữ liệu phải được lưu trữ trọn vẹn trên một nút lá.



Hình 3. Cấu trúc đánh chỉ mục R-tree.

Chúng ta có thể thấy R-tree là một biến thể của B+ tree và nó là một cây cân bằng. Truy nhiên, do các MBR có thể chồng chéo lên nhau và sự chồng chéo này gia tăng khi lượng dữ liệu gia tăng nên cấu trúc này có yếu điểm là kéo theo sự gia tăng các truy cập tìm kiếm không cần thiết. Thêm nữa, chúng ta bắt buộc phải tiến hành tìm kiếm tại mọi mức của cây, ngay cả trong các trường hợp không có (hoặc có rất ít) đối tượng dữ liệu thỏa mãn yêu cầu.

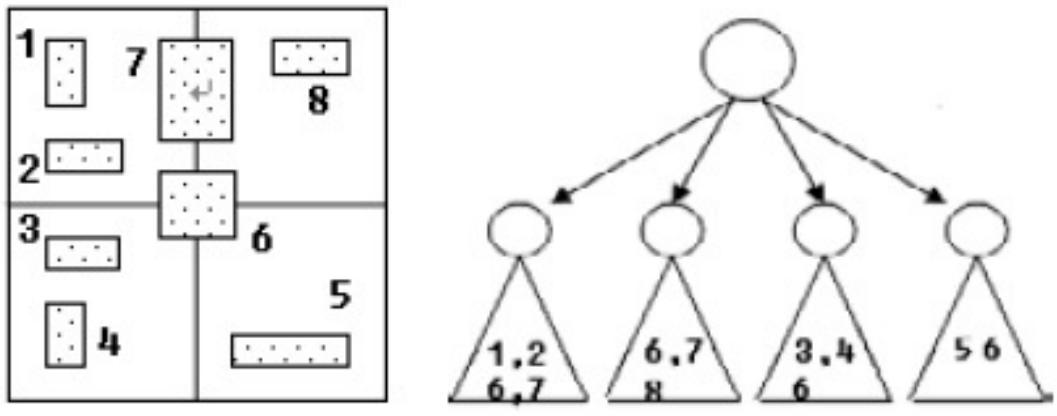
3.3. Kết hợp R-tree và Q-tree

Q-tree và R-tree đều có các ưu điểm và nhược điểm riêng, phụ thuộc cả vào các tình huống và các thao tác khác nhau.

1) Tốc độ thực hiện xây dựng cây Q-tree nhỏ hơn nhiều so với R-tree bởi vì việc phân chia, rồi lựa chọn MBR, sau đó chèn lần lượt từng nút vào R-tree là rất tốn kém thời gian

2) Tuy nhiên việc đánh chỉ số theo Q-tree không phù hợp với các tập dữ liệu lớn do tính không cân bằng của nó.

Cả hai cấu trúc này đều có các biến thể với rất nhiều cải tiến, tuy nhiên, chúng vẫn không thể độc lập đáp ứng các đòi hỏi về tốc độ thực thi của các ứng dụng thời gian thực. Như vậy, giải pháp kết hợp hai phương pháp này với nhau (hybrid) để tận dụng ưu điểm của cả hai phương pháp, bổ trợ cho nhau dường như là một giải pháp hợp lý. Hình vẽ 5 minh họa việc sử dụng QR-tree.



Hình 4. Cấu trúc đánh chỉ mục sử dụng QR-tree.

4. Tối ưu hoá quá trình đánh chỉ mục

4.1. Các công trình liên quan

Rất nhiều các cải tiến về kỹ thuật đánh chỉ mục đã được công bố nhằm tăng hiệu quả thực thi truy vấn.

D. Pfoser[4] đã đưa ra STR-tree (Spatio-Temporal R-tree) và TB-tree (Trajectory-Bandle tree) và chỉ ra rằng hai cấu trúc này hiệu quả hơn hẳn so với các cấu trúc trước đó trong lĩnh vực lưu trữ các đối tượng chuyển động. Tao và Papadias [5] đề xuất MV3R-tree (Multi Version 3D R-tree), là sự kết hợp giữa B-tree và 3D-tree.

QR-Tree được đề xuất bởi Manolopoulos, Y. năm 1996 là cấu trúc gồm hai tầng: áp dụng Q-tree ở tầng thứ nhất để phân chia không gian dữ liệu, sau đó tầng thứ hai áp dụng R-tree trên các vùng dữ liệu đã được chia nhỏ bởi Q-tree. Cũng với phương pháp kết hợp R-tree và Q-tree, K. Chakarabarti và S.Mehrotra [6] đã đưa ra một cấu trúc cây *lai* được sử dụng cho việc đánh chỉ mục với dữ liệu có số chiều lớn. Yuni Xia và Sunil Prabhakar [7] đã đề xuất Q+Rtree áp dụng trong các bài toán đối tượng chuyển động, cải tiến hiệu suất thực thi trong cả hai thao tác cập nhật và truy vấn.

4.2. Phương pháp QR-Tree cải tiến

QR-Tree mặc dù có ưu điểm rõ ràng nhưng nó vẫn tồn tại điểm yếu. Nhìn vào hình vẽ 1.5, có thể thấy rõ ràng rằng hai đối tượng 6 và 7 xuất hiện tại cả hai nút. Như vậy mỗi khi cần cập nhật nội dung, xóa hoặc truy vấn dữ liệu, chúng ta vẫn phải thực hiện lặp lại công việc ở tất cả hai nhánh chứa 6 và 7, gây ảnh hưởng tới tốc độ thực thi của các.phép toán. Phương pháp QR cải tiến được xây dựng dựa trên nền tảng là phương pháp QR-Tree để giải quyết vấn đề trên.

Trong phương pháp này, R-Tree được áp dụng không chỉ ở mức lá của Q-Tree mà còn kết hợp với cả các nút không phải là lá của Q-Tree. Điều này có nghĩa là nếu một đối tượng thuộc về nhiều vùng dữ liệu khác nhau (như trường hợp 6 và 7 trong hình 5) thì mức cha của nó sẽ được xem xét liệu nó có thể chứa toàn bộ đối tượng dữ liệu này hay không. Việc kiểm tra này cứ tiếp tục cho đến gốc (root). Một đối tượng O được định nghĩa là thuộc về vùng không gian con S nếu O hoàn toàn nằm trong S và S là vùng không gian con nhỏ nhất chứa O. Như vậy, các đối tượng nằm tương đối xa nhau sẽ được lưu trữ trên các nhánh khác nhau, nhờ đó giảm thiểu sự chồng chéo giữa các MBR. Lúc này, một đối tượng cụ thể được gắn một chỉ số duy nhất nên hiệu suất của quá trình chèn dữ liệu vào cây sẽ tăng lên (do việc thời gian xây dựng lại cây được rút ngắn); mỗi phép toán sẽ được thực hiện trên một tập các vùng dữ liệu tối thiểu (do không có chứa các dữ liệu lặp sinh ra do sự chồng chéo các vùng không gian) nên việc truy cập dữ liệu sẽ nhanh hơn, thời gian đáp ứng yêu cầu truy vấn được rút ngắn.

Cụ thể hơn, Q-tree được sử dụng để phân chia thô toàn bộ dữ liệu và lưu trữ trong bộ nhớ chính. R-tree sẽ được sử dụng để duy trì cấu trúc logic của cây, được thể hiện dưới dạng một bảng chỉ số mà mỗi dòng trong đó tương ứng với một nút của cây R-tree. Mọi cây R-tree tương ứng với các nút của Q-tree được lưu trữ trong cùng một bảng chỉ số. Mỗi dòng trong bảng này có chứa một thuộc tính có tên ‘Partition’ để chỉ ra vùng không gian con có chứa nút đó. Bằng cách tổ chức như vậy, một cây R-tree tương ứng với một vùng không gian con Q-tree có thể được tham chiếu tới nhờ vào

giá trị của thuộc tính 'Partition' của các nút của cây Q-tree. Để chèn một đối tượng với MBR của nó, trước tiên ta thêm vào bảng dữ liệu, lấy ra ID của đối tượng này rồi gọi một hàm thực hiện việc định vị vị trí của nó trên Q-tree; vị trí tìm được có thể là nút gốc, nút lá hoặc một nút cha trong cây. Dựa vào vị trí này, kết hợp với bảng chỉ số ta có thể truy cập tới cây R-tree và xác định được root của cây R-tree đó. Cứ như vậy, quá trình lặp lại trên các nhánh con của cây tới khi gặp nút lá có triển vọng nhất thì tiến hành chèn MBR và ID của đối tượng, và cuối cùng là cây R-tree nếu cần thiết.

5. Kết luận

SDB đã và đang thu hút được nhiều nghiên cứu trong thời gian gần đây, nhất là khi những dịch vụ trong lĩnh vực GIS hay multimedia ngày càng phát triển. Với những dữ liệu có yêu cầu lưu trữ lớn như vậy, bài toán tối ưu hoá quá trình đánh chỉ mục cho những dữ liệu đó là một bài toán thời sự và liên quan mật thiết đến hiệu năng của những truy vấn trong SDB. Dựa trên hai phương pháp đánh chỉ mục R-Tree, Q-Tree và phương pháp *lai* QR-Tree kết hợp những ưu điểm từ hai phương pháp trên, chúng tôi đã đề xuất cải tiến phương pháp đánh chỉ mục QR-Tree để giảm thiểu hơn nữa sự chồng chéo trong lưu trữ dữ liệu nhằm nâng cao hiệu năng thực thi truy vấn và các phép toán khác. Những kết quả thực nghiệm trong thời gian tới của nhóm tác giả sẽ cho phép kiểm chứng những ưu điểm thu được từ những đề xuất lý thuyết của phương pháp này.

Lời cảm ơn

Công trình này được tài trợ một phần từ đề tài mang mã số: QC.08.03, Đại học Quốc gia Hà Nội.

Tài liệu tham khảo

[1] Raghu Ramakrishnan/Johannes Gehrke. *Database Management Systems,* McGraw Hill, 2sd edition.

[2] Manolopoulos, Y. (1996). QR-tree-a hybrid spatial data structure, *Proceedings of the 1st International Conference on Geographic Information Systems in Urban, Regional and Environmental Planning,* Samos Island, Greece, pp. 3–7.

[3] Rauber A., Tomish P., Riedel H., and Kouba Z. *Integrating Geo-Spatial Data into OLAP Systems Using a Set-based Quad-Tree Representation.* In Proc. of the 4th Int. Conf. onInformation technology for Balanced Automation Systems in Production and Transportation, BASYS, 2000.

[4] D. Pfoser, C. S .Jensen, and Y. Theodoridis. Novel approaches in query processing for moving objects. *Proceedings of the 26th International Conference on Very Large Databases* (VLDB), September 2000.

[5] Papdias D., Kalnis P., Zhang J., and Tao Y. *Efficient OLAP Operations in Spatial Data Warehouse.* In Proc. of the 6th International Symposium on Spatial and Temporal Databases, SSTD, 2001.

[6] K. Chakarabarti and S.Mehrotra. The hybrid tree: An index structure for high dimensional feature spaces. *Proceedings of he Fourteenth International Conference on data engineering* (ICDE'99), 1999.

[7] Yuni Xia, Sunil Prabhakar. Q+Rtree: *Efficient Indexing for Moving Object Databases.* In Proc. of the 8th International Symposium on Spatial and Temporal Databases, SSTD, 2004.

Using QR-Tree for Spatial Database Indexing

Du Phuong Hanh

University of Engineering and Technology, VNU, 144 Xuan Thuy, Hanoi, Vietnam

This paper presents several indexing methods for the spatial datawarehouse (SDW). Actually, SDW is considered as one of most interresting models for manipulating the sptial entities like digital maps, multimedia,… For the SDW, the query optimization is very important due of the mass of spatial data.Thus, this paper investigates the two modern techniques, Q-Tree and R-Tree, for indexing the spatial data in order to improve the performance of the query optimizer. Then, the hybrid approach using QR-Tree will be mostly considered for optimizing the data storage query optimization for spatial database.