MỘT KHẢO SÁT VỀ CẤU TRÚC R-TREE CHO BÀI TOÁN TÌM KIẾM ẢNH

Lê Thị Vĩnh Thanh[1, 3], Lê Mạnh Thạnh[1], Văn Thế Thành[2,*]

[1]Khoa Công nghệ thông tin, Trường Đại học Khoa học, Đại học Huế

[2]Khoa Công nghệ thông tin, Trường Đại học Sư Phạm TP.HCM

[3]Trường Đại học Bà Rịa Vũng Tàu

Email: [1]lmthanh@hueuni.edu.vn, [2]thanhvt@hcmue.edu.vn, [3]thanhltv@bvu.edu.vn

Ngày nhận bài: 02/6/2022; ngày hoàn thành phản biện: 21/6/2022; ngày duyệt đăng: 22/6/2022

TÓM TẮT

Cùng với sự tiến bộ vượt bậc của công nghệ internet và các thiết bị kỹ thuật số, khối lượng dữ liệu hình ảnh đã gia tăng nhanh chóng. Do đó, việc sử dụng kỹ thuật lập chỉ mục lưu trữ dữ liệu ảnh lớn đã trở nên quan trọng nhằm nâng cao tốc độ tìm kiếm ảnh. Trong bài báo này, một khảo sát về các biến thể R-Tree để lập chỉ mục không gian đa chiều áp dụng cho bài toán tìm kiếm ảnh được thực hiện. Đầu tiên, các biến thể của R-Tree được trình bày bao gồm: R+Tree, R*-Tree, SS-Tree, SR-Tree. Thứ hai, chúng tôi giới thiệu các biến thể R-Tree ứng dụng cho bài toán tìm kiếm ảnh đã được công bố trong những thập niên gần đây. Trên cơ sở đó, một cấu trúc R^S-Tree được đề xuất để áp dụng cho bài toán tìm kiếm ảnh theo nội dung. Cuối cùng, chúng tôi tiến hành so sánh kết quả thực nghiệm của cấu trúc R^S-Tree và cấu trúc SR-Tree trên cùng bộ dữ liệu ảnh Wang.

Từ khóa: R^S-Tree; Biến thể R-Tree; Chỉ mục đa chiều; Truy vấn ảnh.

1. GIỚI THIỆU

Ảnh số ngày càng gia tăng theo thời gian và đóng vai trò quan trọng trong nhiều lĩnh vực như y tế, du lịch, giáo dục, nông nghiệp, giải trí v.v. Các ứng dụng sử dụng những loại dữ liệu đa phương tiện như hình ảnh, âm thanh, video… cần phải lưu trữ một khối lượng lớn dữ liệu [1]. Vì vậy, việc lập chỉ mục hiệu quả những dữ liệu không gian đa chiều là cần thiết giúp truy xuất dữ liệu nhanh chóng đáp ứng thời gian thực. Quản lý dữ liệu không gian là một lĩnh vực hoạt động nghiên cứu chuyên sâu trong hơn ba thập kỷ qua. Để hỗ trợ các đối tượng không gian đa chiều trong hệ thống cơ sở dữ liệu, cần xem xét các vấn đề bao gồm mô hình dữ liệu không gian, cơ chế lập chỉ mục, xử lý truy vấn hiệu quả và mô hình chi phí [2]. Một trong những phương pháp truy xuất có ảnh hưởng nhất trong lĩnh vực này là cấu trúc cây R-Tree [3] được

Guttman đề xuất vào năm 1984 là một giải pháp hiệu quả để lập chỉ mục các đối tượng không gian đa chiều. Kể từ đó, một số biến thể của cấu trúc R-Tree đã được đề xuất để truy xuất hiệu quả hơn và xử lý các đối tượng trong không gian đa chiều. Các biến thể của R-Tree được khảo sát trong bài báo này đáp ứng các nhiệm vụ khác nhau như nâng cao hiệu suất truy vấn; mở rộng với các ứng dụng khác nhau. Có nhiều loại biến thể được xây dựng để nâng cao hiệu suất truy vấn trong miền không gian bao gồm truy vấn phạm vi, truy vấn điểm, truy vấn láng giềng gần nhất, truy vấn vùng không gian... Một số biến thể của R-Tree được mở rộng để hỗ trợ các tính năng bổ sung. Bên cạnh đó, một số biến thể được phát triển để áp dụng trong lĩnh vực truy vấn ảnh.

Trong bài báo này, một khảo sát về các biến thể R-Tree được trình bày. Bên cạnh đó, các công trình nghiên cứu áp dụng các biến thể của R-Tree trong lĩnh vực truy vấn ảnh được giới thiệu. Từ đó, một cấu trúc R^S-Tree được đề xuất cho bài toán tìm kiếm ảnh tương tự nhằm nâng cao độ chính xác truy vấn và cải thiện thời gian tìm kiếm. Cuối cùng, chúng tôi tiến hành thực nghiệm tìm kiếm ảnh trên bộ ảnh Wang và so sánh hiệu suất truy vấn của cấu trúc R^S-Tree và biến thể SR-Tree.

2. CẤU TRÚC R-TREE VÀ CÁC BIẾN THỂ

2.1. R-Tree nguyên thủy

Trong cấu trúc R-Tree nguyên thủy, mỗi nút trong *Node* là một bộ $\langle MBR, p \rangle$, trong đó, MBR (Minimun Bounding Rectangle) là một vùng không gian hình chữ nhật chứa các vùng không gian nút con bên trong nó và p là con trỏ liên kết đến các nút con. Mỗi nút lá *Leaf* là một bộ $\langle MBR, oid \rangle$, trong đó, MBR vùng không gian hình chữ nhật chứa đối tượng dữ liệu và oid là định danh đối tượng. Mỗi nút lá *Leaf* trên cây có số phần tử tối thiểu là m và số phần tử tối đa là M. Mỗi nút lá phân chia dữ liệu thành một cụm trong không gian d-chiều. Việc loại bỏ một phần tử trên cây R-Tree có thể phải tái tạo cây lại từ đầu vì nếu một nút lá có số phần tử bằng m thì khi xóa phần tử đó nút lá không tồn tại và phải lấy phần tử còn lại của nút phân bố lại trên cây.

R-Tree là một cấu trúc dữ liệu cây được sử dụng để lưu trữ các chỉ mục dữ liệu không gian một cách hiệu quả. Các tính chất của cây R-Tree: (1) Bao gồm một gốc duy nhất, tập các nút trong và tập các nút lá; (2) Nút gốc chứa con trỏ đến vùng lớn nhất trong miền không gian; (3) Các nút cha chứa các con trỏ tới các nút con trong đó vùng của các nút con nằm bên trong vùng của các nút cha; (4) Các nút lá chứa dữ liệu các đối tượng; (5) Nút gốc phải có ít nhất hai nút con trừ khi nó là một nút lá; (6) Tất cả các lá nằm trên cùng một cấp.

R-Tree nguyên thủy có hai nhược điểm quan trọng: (1) Việc truy xuất một điểm trong cây R có thể dẫn đến việc xét nhiều đường dẫn từ gốc đến lá. Tính chất này có thể dẫn đến suy giảm hiệu suất, đặc biệt khi có sự chồng lấp của các vùng không gian;

(2) Các vùng không gian lưu trữ lớn có thể làm tăng mức độ chồng lấp, dẫn đến suy giảm hiệu suất khi thực hiện truy vấn vùng, do không gian trống.

Trên cơ sở các ưu nhược điểm của cấu trúc R-Tree nguyên thủy, nhiều biến thể khác nhau của R-Tree được xây dựng để nâng cao hiệu suất lưu trữ và tăng độ chính xác truy vấn. Các biến thể của R-Tree được sử dụng phổ biến trong lĩnh vực truy vấn dữ liệu đa phương tiện được trình bày chi tiết trong phần 2.2

2.2. Các biến thể của cấu trúc R-Tree

Các biến thể của R-Tree được phát triển đáp ứng các nhiệm vụ khác nhau như nâng cao hiệu suất truy vấn; mở rộng với các ứng dụng khác nhau. Tuy nhiên, trong bài báo này, chúng tôi chỉ thực hiện khảo sát các biến thể bao gồm: R+-Tree, R*-Tree, SS-Tree, SR-Tree ứng dụng cho lĩnh vực đa phương tiện.

2.2.1. Cấu trúc R+-Tree

Biến thể R+-Tree [4] được giới thiệu để tăng hiệu suất tìm kiếm, đặc biệt là các truy vấn điểm. Chi phí tìm kiếm thấp do giảm số lượng truy cập đĩa cần thiết để nhận được kết quả. Sự chồng lắp không gian và vùng bao phủ là hai yếu tố quan trọng đối với hiệu suất tìm kiếm. Khái niệm chính của biến thể này là không có sự chồng lắp. Để đạt được điều này, các đối tượng được chèn phải được chia thành hai hoặc nhiều MBR, điều đó có nghĩa là một mục đối tượng cụ thể có thể được sao chép và lưu trữ dự phòng trong một số nút. Nếu tồn tại một hình chữ nhật dữ liệu ở cấp thấp hơn chồng lắp với một hình chữ nhật khác, nó sẽ được phân tách thành một tập hợp các hình chữ nhật con không chồng lắp đồng thời liên kết chúng với hình chữ nhật ban đầu. Thuật toán chèn sẽ thêm hình chữ nhật vào nhiều nút bằng cách chia thành các hình chữ nhật con trong quá trình chồng lắp. Việc tìm kiếm được thực hiện dễ dàng vì ở đây tránh được việc tìm kiếm nhiều đường do chồng lắp.

2.2.2. Cấu trúc R*-Tree

R*-Tree [5] là biến thể của R-Tree, kết quả thực nghiệm cho thấy R*-Tree vượt trội hơn trong việc xử lý và hiệu suất truy vấn. Các thay đổi trong R*-Tree so với R-Tree bao gồm: (1) trong khi chọn cây con, nếu nút đang xét không phải nút lá và trỏ đến nút lá thì việc mở rộng chồng lắp tối thiểu được xem xét. Nếu nút hiện hành là nút trong không kế lá thì nút cần mở rộng diện tích ít nhất sẽ được chọn. Điều này làm tăng hiệu suất của các truy vấn cửa sổ nhỏ trên dữ liệu không đồng nhất; (2) Việc phân chia dựa trên giá trị bao gồm diện tích, đường biên và mức độ chồng lắp; (3) Vì tách nút là một quá trình tốn nhiều chi phí, nên để hạn chế việc tách nút phép toán chèn lại được yêu cầu. Điều này cải thiện việc sử dụng khả năng lưu trữ của các nút. Tuy nhiên, việc tái cấu trúc cây là một hoạt động tốn nhiều chi phí. Do đó, chỉ có một lần gọi lại phép toán chèn được phép cho mỗi cấp độ của cây. Khi tràn không thể được xử lý bằng cách chèn lại, việc tách nút được thực hiện.

2.2.3. Cấu trúc SS-Tree

SS-Tree [6] là một cấu trúc chỉ mục được thiết kế để lập chỉ mục tương tự của dữ liệu điểm đa chiều. Đây là sự cải tiến của R*-Tree và cải thiện hiệu suất của các truy vấn láng giềng gần nhất bằng cách thay đổi các yếu tố sau: (1) sử dụng các hình cầu giới hạn thay vì các hình chữ nhật giới hạn; (2) SS-Tree sửa đổi cơ chế chèn lại của R*-Tree. Tâm của hình cầu là tâm trung bình của tất cả các phần tử. Khi chèn một phần tử mới, một nhánh con có tâm gần nhất với phần tử mới sẽ được chọn. Khi một nút hoặc một lá đã đầy, R*-Tree sẽ chèn lại một phần của các phần tử của nó thay vì tách nút, trừ khi việc chèn lại đã được thực hiện trên cùng một cấp cây. Trong cấu trúc SS-Tree, việc chèn lại các phần tử trừ khi việc chèn lại đã được thực hiện tại cùng một nút hoặc lá. Điều này làm gia tăng sự tái tổ chức cấu trúc cây. Vì khối cầu được xác định bởi tâm và bán kính của nó, nên nó yêu cầu ít dung lượng hơn so với hình chữ nhật có giới hạn với đường viền dưới và đường viền trên ở mọi chiều.

2.2.4. Cấu trúc SR-Tree

SR-Tree [7] là sự kết hợp R*-Tree và SS-Tree. Vùng giới hạn của SR-Tree được xác định bằng giao điểm của hình cầu giới hạn và hình chữ nhật giới hạn. Việc kết hợp các hình chữ nhật bao quanh cho phép các vùng lân cận được phân chia thành các vùng nhỏ hơn so với SS-Tree và giảm chồng lắp giữa các vùng. Điều này nâng cao hiệu suất trên các truy vấn láng giềng gần nhất, đặc biệt là đối với dữ liệu có chiều cao và không đồng nhất, có thể được ứng dụng trong việc lập chỉ mục độ tương tự hình ảnh/video. Khi chèn, SR-tree cần cập nhật cả hình cầu và hình chữ nhật có giới hạn. Cách cập nhật hình chữ nhật tương tự như trong R-Tree nguyên thủy, nhưng để xác định hình cầu giới hạn của nút cha, nó phải sử dụng cả hình cầu giới hạn và hình chữ nhật giới hạn của nút con. Do đó, việc tạo và cập nhật tương đối phức tạp và tốn kém. Bên cạnh đó, vì SR-tree chứa cả hình cầu và hình chữ nhật, nên kích thước của nó chắc chắn sẽ lớn hơn nhiều, dẫn đến vấn đề giảm fan-out mà có thể yêu cầu nhiều nút hơn để đọc trên các truy vấn, điều này rõ ràng ảnh hưởng đến hiệu suất.

2.2.5. So sánh ưu nhược điểm của các biến thể R-Tree

Bảng 1. So sánh giữa các cấu trúc chỉ mục R-Tree

Cấu trúc chỉ mục	**Loại truy vấn**	**Loại dữ liệu**	**Tính phức tạp**	**Ứng dụng**
R-Tree	Truy vấn vùng.	Dữ liệu đa chiều.	Sử dụng không gian không hiệu quả, không tính được độ phức tạp về thời gian trong trường hợp xấu nhất.	Ứng dụng thế giới thực như hệ thống định vị, v.v.

Cấu trúc chỉ mục	Loại truy vấn	Loại dữ liệu	Tính phức tạp	Ứng dụng
R+-tree	Truy vấn vùng.	Dữ liệu đa chiều.	Dữ liệu không chồng lắp sử dụng không gian hiệu quả hơn R-Tree.	Đối tượng dữ liệu đa chiều.
R*-tree	Truy vấn điểm, truy vấn vùng.	Dữ liệu đa chiều.	Chi phí triển khai cao hơn các biến thể cây R khác nhưng mạnh mẽ về phân phối dữ liệu.	Ứng dụng với dữ liệu ở dạng điểm và hình chữ nhật, lĩnh vực đa phương tiện.
SS-Tree	Truy vấn vùng, truy vấn KNN, Truy vấn điểm.	Dữ liệu đa chiều.	Tối ưu hóa không gian, tuy nhiên vẫn tốn chi phí cao do sử dụng phép toán chèn lại.	Ứng dụng sử dụng các phương pháp truy cập đa chiều, lĩnh vực đa phương tiện.
SR-tree	Truy vấn vùng, truy vấn KNN, Truy vấn điểm.	Dữ liệu đa chiều.	Độ phức tạp lớn hơn các cấu trúc khác do sử dụng cả khối cầu và khối hình chữ nhật.	Ứng dụng sử dụng các phương pháp truy cập đa chiều, lĩnh vực đa phương tiện.

3. ỨNG DỤNG R-TREE CHO LĨNH VỰC TRUY VẤN ẢNH

Trong những năm gần đây, các hệ thống tìm kiếm ảnh được thực hiện bởi nhiều phương pháp lập chỉ mục khác nhau và mang lại những kết quả tốt. Kỹ thuật lập chỉ mục dựa trên đặc trưng không gian và lưu trữ trên các cây KD-Tree, R-Tree , v.v là những cấu trúc cây ra đời khá sớm và thu được nhiều kết quả khả quan [1, 2]. Trên cơ sở nghiên cứu về các cấu trúc lập chỉ mục, nhiều công trình đã ứng dụng phương pháp này vào các bài toán truy vấn ảnh dựa trên nội dung được thực nghiệm trên các bộ dữ liệu khác nhau và thu được các kết quả khả quan, cụ thể là:

Abd Aziz và cộng sự (2018) đã giới thiệu một phương pháp giảm chiều véc-tơ đặc trưng hình ảnh sử dụng S-Map và thực hiện tìm kiếm tương tự dựa trên cấu trúc chỉ mục R-Tree. Nhóm tác giả sử dụng độ đo Euclid để đo độ tự giữa các đối tượng [8]. Shama, P. S. và cộng sự (2015) đã đề xuất một hệ thống truy vấn ảnh tương tự sử dụng cấu trúc R*-Tree cho bộ ảnh thực vật. Thực nghiệm được thực hiện trên 300 ảnh thực vật [9]. Alfarrarjeh và cộng sự (2020) đã đề xuất một lớp chỉ mục R*-Tree ứng dụng cho bài toán tìm kiếm ảnh tương tự với dữ liệu ảnh đường phố [10]. Tuy nhiên, việc truy vấn một đối tượng dựa trên cấu trúc R-Tree có thể dẫn đến việc xét nhiều đường dẫn từ gốc đến lá. Điều này có thể dẫn đến suy giảm hiệu suất, đặc biệt khi có sự chồng lấp của các vùng không gian. Trong cấu trúc R*-Tree thuật toán chèn lại phần tử khi gặp một nút tràn sẽ tổ chức lại cây dẫn đến tăng chi phí tạo cây.

Vanitha và cộng sự (2017) đã đề xuất một cấu trúc chỉ mục SR-Tree ứng dụng cho hệ thống tìm kiếm ảnh tương tự theo nội dung. Hệ thống thực hiện trích xuất đặc trưng màu sắc, đặc trưng không gian và lưu trữ véc-tơ đặc trưng trên cây SR-Tree. Sau đó, quá trình tìm kiếm ảnh được thực hiện dựa trên cây SR-Tree. Kết quả thực nghiệm trên bộ ảnh Wang cho thấy SR-Tree hoạt động hiệu quả hơn các cấu trúc lập chỉ mục tương tự khác [11]. Tuy nhiên, trong cấu trúc cây SR-tree khi chèn phần tử cần cập nhật cả hình cầu và hình chữ nhật có giới hạn. Do đó, việc tạo và cập nhật tương đối phức tạp và tốn kém. Bên cạnh đó, mỗi nút trên cây SR-Tree chứa cả hình cầu và hình chữ nhật, nên kích thước sẽ lớn hơn nhiều làm ảnh hưởng đến hiệu suất truy vấn.

Từ việc khảo sát các công trình trên cho thấy rằng cấu trúc lập chỉ mục R-Tree được vận dụng trong các bài toán truy vấn ảnh theo nội dung nhằm nâng cao hiệu quả và tốc độ truy vấn. Trên cơ sở đó một cấu trúc cải tiến cho bài toán tìm kiếm ảnh được đề xuất trong phần 4.

4. MỘT CẤU TRÚC CẢI TIẾN R-TREE CHO BÀI TOÁN TÌM KIẾM ẢNH

4.1. Mô tả cấu trúc R^S-Tree

R^S-Tree [12], một cấu trúc cải tiến của R-Tree, là cây đa nhánh cân bằng ứng dụng cho bài toán tìm kiếm ảnh tương tự. Việc gom nhóm dữ liệu được thực hiện trên từng nút của cây R^S-Tree dựa vào độ đo tương tự giữa các véc-tơ đặc trưng ảnh ngưỡng θ cho trước nhằm tạo ra một cây đa nhánh cân bằng để giảm thời gian tìm kiếm. Cây R^S-Tree là cây phân hoạch dữ liệu không gian bao gồm: một nút gốc, một tập nút trong và một tập nút lá. Mỗi nút trong S_{node} trên cây là một khối cầu MBS (Minimun Bounding Sphere) bao phủ tất cả khối cầu các nút thuộc nhánh cây con. Mỗi nút lá S_{leaf} trên cây gồm một tập các thực thể, mỗi thực thể *spED* là một khối cầu chứa không gian đối tượng dữ liệu và định danh đối tượng.

Trong phần này, R^S-Tree được đề xuất nhằm lưu trữ các phần tử hình ảnh trong không gian n chiều, mỗi phần tử $spED$ là một khối cầu có tâm $\vec{c}_{sp}$ và bán kính r_{sp} chứa véc-tơ đặc trưng ảnh $\vec{f} = (v_1, v_2, v_3, \dots, v_d)$, với v_i là đặc trưng cấp thấp thứ i của hình ảnh. Các nút trên R^S-Tree bao gồm:

+ Nút trong S_{node}: là một bộ $\langle MBS, p\rangle$, trong đó MBS là một khối cầu có tâm $\vec{c}_{node}$ và bán kính r_{node}, p là con trỏ liên kết đến các nút con. Khối cầu này bao phủ các khối cầu của các nút thuộc nhánh cây con. Mỗi nút trong S_{node} có số phần tử tối thiểu là 2 và tối đa là N.

+ Nút lá S_{leaf}: là bộ $\langle MBS, entity\rangle$, trong đó MBS là một khối cầu có tâm $\vec{c}_{leaf}$ và bán kính r_{leaf} chứa một tập thực thể $entity$ với mỗi thực thể $spED$ là một bộ $\langle MBS, oid\rangle$ trong đó MBS là khối cầu có tâm $\vec{c}_{sp}$ và bán kính r_{sp} chứa không gian đối tượng, oid là

định danh đối tượng $\vec{f} = (v_1, v_2, v_3, \dots, v_d)$. Mỗi nút lá S_{leaf} có số phần tử tối đa là M và số phần tử tối thiểu là $m(1 < m \leq {}^{M}/_{2})$.

R^S-Tree sử dụng khối cầu để lưu trữ dữ liệu vì các lý do sau: (1) Việc xác định một hình cầu chỉ phụ thuộc vào tâm và bán kính trong khi hình chữ nhật phụ thuộc vào trọng tâm, đường biên trên, đường biên dưới; (2) Khi không gian giản nỡ thì hình cầu sẽ tính toán ít biến hơn hình chữ nhật; (3) khi cập nhật tại một nút, nếu dùng MBR thì cần phải tìm tất cả các hình chữ nhật để xác định biên phải và biên trái bao phủ của một nút. Do đó, sẽ tốn chi phí tìm kiếm và sắp xếp. Trong khi dùng SBR thì không cần tốn kém các chi phí này; (4) Áp dụng được nhiều loại truy vấn bao gồm truy vấn vùng, truy vấn KNN, truy vấn điểm, truy vấn không gian.

4.2. Các nội dung cải tiến trên R^S-Tree

4.2.1. Cải tiến việc biểu diễn véc-tơ đa chiều thành khối cầu không gian

Cho hình ảnh I có véc-tơ đặc trưng $\vec{f_I} = (v_{I1}, v_{I2}, v_{I3}, \dots, v_{Id})$. Một khối cầu MBS của thực thể $spED$ được minh họa tại **Hình 3** là khối cầu chứa chứa đối tượng $\vec{f_I}$ gồm véc-tơ tâm $\vec{c}_{sp}$ và bán kính r_{sp} được mô tả như sau:

1) Tâm khối cầu thực thể:

$$\vec{c}_{sp} = (c_{I1}, c_{I2}, c_{I3}, \dots, c_{Id})$$

Với $c_{Ij} = \max(0, v_{Ij} - a_{Ij}), j = 1..d$

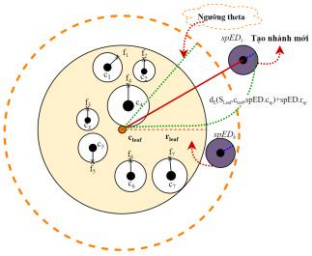
Trong đó, $a_{I1} = {}^{v_{I1}}/_{k}, a_{I2} = {}^{v_{I2}}/_{k}, \dots, a_{Id} = {}^{v_{Id}}/_{k}$, với k là tham số, $k \geq 2$.

2) Bán kính khối cầu thực thể:

$$r_{sp} = \frac{1}{d}\sqrt{\sum_{j=1}^{d}(c_{Ij} - v_{Ij})^2}$$

4.2.2. Cải tiến phép toán thêm phần tử

Quá trình thêm một phần từ $spED$ vào một nút lá hiện hành S_{leaf}, một ngưỡng θ được đề xuất nhằm thực hiện gom cụm các đối tượng tương đồng được minh họa như **hình 1**. Nếu $d_E(spED.\vec{c}_{sp}, S_{leaf}.\vec{c}_{leaf}) + spED.r_{sp} > \theta$ thì tạo một nhánh mới để lưu phần tử $spED$, ngược lại thêm phần tử $spED$ vào nút lá hiện hành. Điều này hạn chế việc đưa các ảnh có độ tương tự khác xa nhau vào trong một vùng không gian và hạn chế vấn đề chồng lấp không gian.

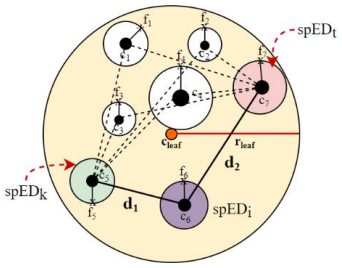


Hình 1. Mô tả thao tác thêm một phần tử vào nút lá

Việc biểu diễn véc-tơ đa chiều thành khối cầu không gian và thuật toán thêm phần tử vào cây được kế thừa từ công trình [12]. Trong bài báo này, thuật toán tách nút được cải tiến và trình bày trong phần 4.2.3

4.2.3. Cải tiến phép toán tách nút

Thuật toán tách nút trong công trình [12], việc phân hoạch tập các phần tử về hai nút con được thực hiện bằng cách lựa chọn tuần tự ngẫu nhiên trong tập *M-1* phần tử để phân phối lần lượt vào hai nút con. Trong bài báo này, một cải tiến được đề xuất nhằm nâng cao độ chính xác gom cụm dữ liệu. Thuật toán tách nút được cải tiến như sau: Tập *M-1* phần tử sẽ được tính độ đo sai biệt so với hai phần tử được chọn làm hai tâm của hai nút mới. Trong quá trình tách nút lá thành hai nút con, việc lựa chọn phần tử $spED$ phân phối vào hai nút lá mới cho phù hợp được thực hiện dựa theo độ sai biệt (ký hiệu là d_{Dif}) được minh họa như trong **hình 2**, phần tử có độ đo sai biệt lớn sẽ được chọn để phân phối trước.



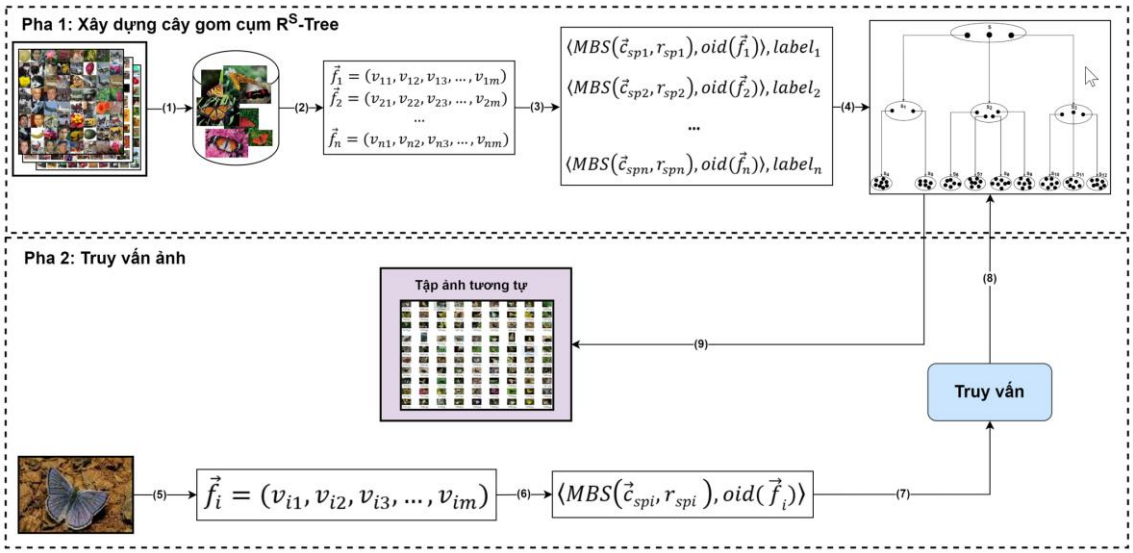
Hình 2. *Minh họa độ đo sai biệt của phần tử* $spED_i$

Trong **Hình 2**, độ đo sai biệt (d_{Dif}) của phần tử $spED_i$ so với hai phần tử $spED_k$ và $spED_t$ là hai phần tử được chọn làm tâm của hai nút lá mới S_{L1} và S_{L2} được tính như sau: $d_{Dif} = |d_1 - d_2|$, trong đó $d_1 = d_E(spED_i.\vec{c}_{spi}, spED_k.\vec{c}_{spk})$, $d_2 = d_E(spED_i.\vec{c}_{spi}, spED_t.\vec{c}_{spt})$. Độ sai biệt càng lớn thì khả năng xác định phần tử thuộc về một trong hai nút càng cao.

4.3. Mô hình truy vấn ảnh dựa trên cấu trúc RS-Tree.

Một ảnh cần truy vấn *I* được trích xuất véc-tơ đặc trưng và thực hiện truy vấn trên cấu trúc cây RS-Tree. Quá trình truy vấn trên cây cho đến khi gặp được nút lá thì tập hợp tất cả các phần tử dữ liệu trong nút lá đó được gọi là một tập ảnh tương tự của ảnh truy vấn. Sau đó, tập ảnh này được sắp xếp theo độ đo tương tự để tìm ra các ảnh

tương tự gần nhất. Mô hình truy vấn ảnh tương tự theo nội dung với một ảnh truy vấn đầu vào cho trước dựa trên cây RS-Tree được minh họa như **Hình 3**.



Hình 3. Mô hình tìm kiếm ảnh theo nội dung CBIR-RST dựa trên cây RS-Tree

Quá trình tìm kiếm ảnh được thực hiện gồm hai pha, pha thứ nhất thực hiện gom cụm và lưu trữ dữ liệu hình ảnh trên cây RS-Tree, pha thứ hai thực hiện tìm kiếm các hình ảnh tương tự và phân lớp ngữ nghĩa cho ảnh đầu vào. Quá trình thực hiện được mô tả như sau:

Xây dựng cây phân cụm. Quá trình xây dựng cây RS- Tree gồm 2 bước như sau:

Bước 1. Trích xuất tập véc-tơ đặc trưng f_i của tập dữ liệu ảnh.

Bước 2. Biểu diễn véc-tơ đặc trưng của tập dữ liệu ảnh thành khối cầu không gian.

Bước 3. Cấu trúc RS-Tree được tạo ra chứa các véc-tơ f_i đặc trưng của hình ảnh.

Tìm kiếm ảnh. Quá trình tìm kiếm ảnh được thực hiện theo các bước như sau:

Bước 1. Trích xuất véc-tơ đặc trưng của ảnh cần truy vấn.

Bước 2. Thực hiện truy vấn ảnh tương tự dựa trên cấu trúc không gian RS-Tree.

Bước 3. Tra cứu tập ảnh tương tự dựa trên tập chỉ mục đã được truy vấn.

5. THỰC NGHIỆM

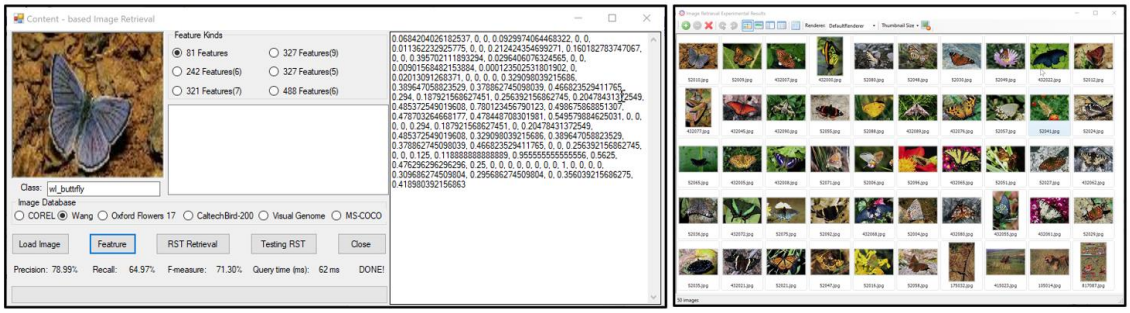
5.1. Môi trường thực nghiệm

Pha tiền xử lý được thự hiện trên máy PC CPU 2.3GHz 8-core 9th-generation Intel Core i9, 16GB 2666MHz memory, 1TB flash storage. Pha tìm kiếm được thực

nghiệm trên máy PC CPU Intel Core i7-6500U CPU @ 2.50GHz, 8.0GB RAM, hệ điều hành Windows 10 Pro 64 bit.

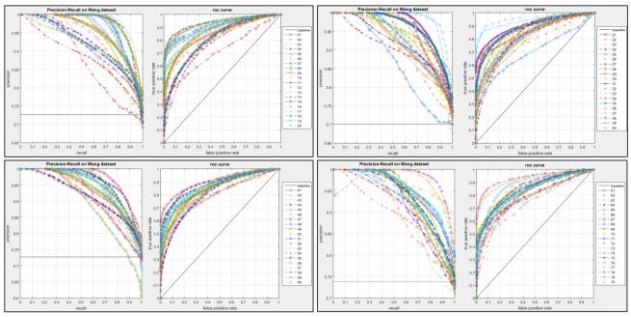
5.2. Ứng dụng và kết quả thực nghiệm

Trong bài báo này, chúng tôi tiến hành thực nghiệm trên bộ ảnh Wang gồm 10.800 ảnh được chia thành 80 phân lớp. Ứng dụng thực nghiệm được minh họa trong **hình 4**.



Hình 4. Giao diện truy vấn ảnh trên cấu trúc R^S-Tree

Để đánh giá hiệu quả của phương pháp tìm kiếm ảnh, phần thực nghiệm được đánh giá các giá trị gồm: độ chính xác (precision), độ phủ (recall) và độ đo dung hòa F-measure. Kết quả thực nghiệm được thể hiện như trong **hình 5**. Mỗi đường cong trên đồ thị mô tả kết quả truy vấn từ một chủ đề ảnh trong bộ dữ liệu Wang. Đồng thời, đường cong tương ứng trong đồ thị ROC cho biết tỷ lệ kết quả truy vấn đúng và sai, nghĩa là diện tích dưới đường cong đánh giá tính đúng đắn của các kết quả truy vấn.



Hình 5. Precision-Recall và đường cong ROC của bộ dữ liệu Wang

Hiệu suất truy vấn của thuật toán tìm kiếm ảnh tương tự dựa trên cấu trúc R^S-Tree được biểu diễn trong **Bảng 2**

Bảng 2. Hiệu suất tìm kiếm của hệ truy vấn CBIR-RST trên bộ dữ liệu Wang

Bộ dữ liệu thực nghiệm	**Avg. precision**	**Avg. recall**	**Avg. F-measure**	**Avg. query time (ms)**
0-20	70.98%	64.55%	67.61%	57.28

21-40	72.94%	65.82%	69.20%	46.69
41-60	74.98%	66.21%	70.32%	67.91
61-80	76.94%	68.18%	72.30%	70.28
Kết quả	**73.96%**	**66.19%**	**69.86%**	**60.54**

Bảng 3. So sánh độ chính xác giữa các phương pháp trên bộ dữ liệu WANG

Phương pháp	**Avg. precision**	**Avg. recall**	**Avg. F-measure**
J. Vanitha, 2017 [9]	75.80%	0.16%	26.42%
Phương pháp đề xuất	**73.96%**	**66.19%**	**69.86%**

Bảng 3, trong công trình [9], nhóm tác đã sử dụng SR-Tree để lưu trữ dữ liệu hình ảnh. Tuy nhiên, quá trình chèn trên SR-tree cần cập nhật cả giới hạn hình cầu và hình chữ nhật. Do đó, việc tạo và cập nhật tương đối phức tạp và tốn kém. Độ chính xác là 75,8% nhưng độ phủ không cao (16%) dẫn đến độ dung hòa thấp. Trong cấu trúc R^S-Tree, một hướng cải tiến là đưa ra một ngưỡng $\theta \in (0,1)$ để làm phép lọc nhằm gom cụm các phần tử tương tự nhau, đồng thời cấu trúc R^S-Tree lưu trữ các phần tử ảnh bằng các khối cầu phù hợp với việc truy vấn điểm và truy vấn vùng không gian nhằm tối ưu hóa kết quả truy vấn. Do đó, kết quả truy vấn của phương pháp đề xuất tối ưu hơn về độ chính xác và thời gian truy vấn nhanh hơn.

6. KẾT LUẬN

Trong bài báo này, chúng tôi thực hiện một khảo sát nhằm đánh giá hiệu suất tìm kiếm trên cây R-Tree và các biến thể của R-Tree được áp dụng trong bài toán truy vấn ảnh. Mỗi biến thể của R-Tree đều có những ưu và nhược điểm riêng để thích ứng với từng loại dữ liệu tiếp cận và hướng đến tập dữ liệu khác nhau. Đồng thời các biến thể cây R-Tree được ứng dụng cho từng bài toán cùng với các mô hình tìm kiếm ảnh được khảo sát. Từ đó, một cấu trúc R^S-Tree được cải tiến dựa trên R-Tree để nâng cao hiệu suất truy vấn. Bên cạnh đó, một mô hình tìm kiếm ảnh sử dụng cấu trúc R^S-Tree được đề xuất để làm cơ sở thực hiện ứng dụng thực nghiệm trên các bộ ảnh khác nhau. Kết quả thực nghiệm trên bộ ảnh Wang có độ chính xác là 73.86%, độ phủ 66,19%, độ đo dung hòa 69,86%. Theo kết quả thực nghiệm cho thấy tính hiệu quả so với các công trình khác trên cùng một tập dữ liệu ảnh. Hướng phát triển tiếp theo của chúng tôi là áp dụng các phương pháp học máy trên cấu trúc R^S-Tree, đồng thời kết hợp các cấu trúc khác để nâng cao độ chính xác cho bài toán tìm kiếm ảnh.

LỜI CẢM ƠN

Chúng tôi xin trân trọng cám ơn Khoa Công nghệ thông tin – Đại học Khoa học- Đại học Huế, nhóm nghiên cứu SBIR-HCM đã góp ý chuyên môn cho nghiên cứu này. Chúng tôi xin trân trọng cảm ơn Trường Đại học Bà Rịa Vũng Tàu, Trường ĐH Sư phạm Tp.HCM đã tạo điều kiện về cơ sở vật chất giúp chúng tôi hoàn thành bài nghiên cứu này.

TÀI LIỆU THAM KHẢO

[1]. Shaik Abdul Nusrath Begum, K. P. Supreethi, (2018). "A Survey on Spatial Indexing". Journal of Web Development and Web Designing Volume 3 Issue 1, pp. 1-25.

[2]. Y. Manolopoulos, A. Nanopoulos, A.N. Papadopoulos and Y. Theodoridis (2010). "R-Trees: Theory and Applications", Springer, London, pp. 117–125.

[3]. A. Guttman, (1984). "R-trees: A dynamic index structure for spatial searching", Proc. ACM SIGMOD Int. Conf. on Management of Data, Boston, MA, pp.47-57.

[4]. Timos K. Sellis, Nick Roussopoulos and Christos Faloutsos, (1987). The R+-Tree: A Dynamic Index for Multi-Dimensional Objects, Proceedings of 13th International Conference on Very Large Data Bases, pp.507-518.

[5]. N. Beckmann, H.-P. Kriegel, R. Schneider and B. Seeger (1990). The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles, Proceedings of the ACM SIGMOD International Conference on Management of Data, pp.322-331.

[6]. White, D. A., & Jain, R. (1996). Similarity indexing with the SS-tree. In Proceedings of the Twelfth International Conference on Data Engineering . IEEE, pp. 516-523.

[7]. Katayama, N., & Satoh, S. I. (1997). The SR-tree: An index structure for high-dimensional nearest neighbor queries. ACM Sigmod Record, 26(2), pp.369-380.

[8]. Abd Aziz, Dewi Noorsyitta Azida Binti, Naoya Higuchi, and Takeshi Shinohara. "A Study on Optimization of Similarity Search by R-tree using Dimension Reduction." Information Processing Society of Japan, (2018).

[9]. Vanitha, J., and M. SenthilMurugan. "An efficient content based image retrieval using block color histogram and color co-occurrence matrix." International Journal of Applied Engineering Research (2017), pp. 15966-15971.

[10]. Shama, P. S., K. Badrinath, and Anand Tilugul. "An Efficient Indexing Approach for Content based Image Retrieval". International Journal of Computer Applications Vol.117, No.15 (2015), pp. 11-18.

[11]. Alfarrarjeh, Abdullah, et al. "A Class of R*-tree Indexes for Spatial-Visual Search of Geo-tagged Street Images". IEEE 36th International Conference on Data Engineering (ICDE). IEEE, (2020), pp. 1990-1993.

[12]. Lê Thị Vĩnh Thanh, Văn Thế Thành, Lê Mạnh Thạnh (2021). "Một phương pháp tìm kiếm ảnh hiệu quả dựa trên cấu trúc R-Tree", Kỷ yếu Hội thảo Quốc gia về Công nghệ thông tin

và ứng dụng trong các lĩnh vực (CITA2021), Đại học Đà Nẵng, Nhà xuất bản Đà Nẵng, ISBN: 978-604-84-5998-7, trang 259-271.

A SURVEY ON R-TREE STRUCTURE FOR IMAGE RETRIEVAL PROBLEM

Le Thi Vinh Thanh[1, 3], Van The Thanh[2], Le Manh Thanh[1,*]

[1]Faculty of Information Technology, University of Sciences, Hue University

[2]Faculty of Information Technology, University of Education HoChiMinh city

[3]University of Ba Ria Vung Tau

[1]lmthanh@hueuni.edu.vn, [2]thanhvt@hcmue.edu.vn, [3]thanhltv@bvu.edu.vn

ABSTRACT

With the great advancement of the Internet and digital devices, the volume of image data has been increased rapidly. Therefore, it has become important to use indexing techniques to store and retrieve large image data to improve image query speed. In this paper, a survey of R-Tree variants for multidimensional spatial indexing applied to the image retrieval problem is performed. First, variants of R-Tree are presented including R+Tree, R*-Tree, SS-Tree, SR-Tree. Secondly, other R-Tree variants for image search problems published in recent decades will be introduced. On that basis, an R^S-Tree structure is proposed to use for the content-based image retrieval problem. Finally, we compare the experimental results of the R^S-Tree and the SR-Tree on the Wang image dataset.

Keywords: Image retrieval, Multi-demensional indexes;R^S-Tree; R-Tree variants.

Lê Thị Vĩnh Thanh sinh năm 1983. Bà tốt nghiệp ngành Sư phạm tin học Trường Đại học Sư phạm TP.HCM vào năm 2006, nhận bằng Thạc sĩ ngành Hệ thống thông tin tại Trường Đại học Khoa học Tự Nhiên TP.HCM vào năm 2014. Hiện là nghiên cứu sinh ngành Khoa học máy tính Trường Đại học khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: xử lý ảnh, tìm kiếm ảnh và cơ sở dữ liệu.

Lê Mạnh Thạnh sinh năm 1953. Ông nhận bằng Tiến sĩ ngành khoa học máy tính tại Đại học Budapest (ELTE), Hungary vào năm 1994. Nhận học hàm Phó giáo sư tại trường Đại học Khoa học, Đại học Huế, Việt Nam vào năm 2004.

Lĩnh vực nghiên cứu: cơ sở dữ liệu, cơ sở tri thức và lập trình logic.

Văn Thế Thành sinh năm 1979. Ông tốt nghiệp chuyên ngành Toán tin tại Đại học Khoa học Tự nhiên - Đại học Quốc gia TP.HCM vào năm 2001, nhận bằng Thạc sĩ Khoa học Máy tính tại Đại học Quốc gia TP.HCM vào năm 2008. Năm 2016, nhận bằng Tiến sĩ Khoa học Máy tính tại trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: xử lý ảnh, khai thác dữ liệu ảnh và tìm kiếm ảnh.