Script nói Project CK DSA ngày 19/05/2023.

# Hải.

1. Giới thiệu:

Khi định chọn chủ đề để kết thúc môn học, nhóm chúng tôi đã mong muốn tạo ra một bản đồ (như một cách để mô tả thế giới mà bản thân đang sống). Có lẽ mặc định con người muốn tạo một bản đồ giống như khao khát làm chủ và không bị lạc lối trong thế giới muôn trùng cám giỗ và lựa chọn.

Mọi thứ dường như bế tắc khi rất khó để cố tạo 1 bản đồ mô phỏng hoàn hảo thực tại nên chúng tôi đã random danh sách chủ đề vốn có. May mắn thay chủ đề được random ra ngay lần đầu tiên là R Tree, 1 cấu trúc dữ liệu mà sau vài đường Google chúng tôi được biết nó được ứng dụng trong việc tạo bản đồ, cụ thể là lưu trữ và thực hiện các truy vấn với các loại dữ liệu đa chiều như các điểm tọa độ GIS địa lý. Nhưng càng tìm hiểu nhiều hơn, chúng tôi nhận ra nó còn có thể làm được nhiều điều hơn cả vậy.

Giới thiệu qua về nhóm chúng tôi. Chúng tôi là bộ ba thứ thiệt: Phạm Ngọc Hải, Lương Đức Anh và Nguyễn Văn Thắng. Chúng tôi sẽ trình bày 1 cách ngắn gọn mà đảm bảo tổng quan chủ đề qua 7 mục cốt lõi. Giờ chúng ta sẽ đến với phần đầu tiên. Tại sao là R Tree. Cụ thể phần này đặt ra 2 câu hỏi quan trọng bao quát cả bài luận này: Mục đích gì khiến R Tree được ra đời? và Tại sao nó được sử dụng phổ biến chứ không phải một cấu trúc dữ liệu nào khác cho việc giải quyết mục đích đã đặt ra đó.

1. Why is R Tree?

Trong quá khứ, người ta chỉ cần lưu trữ và truy vấn các dữ liệu đơn chiều vì các hệ thống và ứng dụng không đòi hỏi nhiều thông tin về vị trí và mối quan hệ giữa các đối tượng trong không gian đa chiều. Các dữ liệu đơn chiều là các dữ liệu có thể biểu diễn trên một trục đơn, ví dụ như các giá trị số nguyên, số thực (có thể chiếu lên 1 trục số), ký tự (có thể chiếu lên 1 trục các bit biểu diễn các ký tự unicode), boolean (với 0 quy định là false, các trường hợp khác là true thì dữ liệu cũng chỉ cần 1 trục - đơn chiều để biểu diễn), thời gian (1 đại lượng được biểu diễn tuyến tính tăng dần) hoặc địa chỉ (như địa chỉ bộ nhớ được đánh thứ tự trên 1 trục để truy cập).

Để lưu trữ và truy vấn các dữ liệu đơn chiều, các kiểu dữ liệu phổ biến được sử dụng là:

1. Mảng (Array): Lưu trữ các giá trị dữ liệu trên một trục đơn, và cho phép truy cập nhanh chóng đến các giá trị dữ liệu bằng cách sử dụng chỉ số.

2. Danh sách liên kết (Linked List): Lưu trữ các giá trị dữ liệu trên một trục đơn trong các nút của danh sách liên kết, và cho phép truy cập tuyến tính đến các giá trị dữ liệu.

3. Bảng băm (Hash Table): Lưu trữ các giá trị dữ liệu trên một trục đơn dựa trên quy tắc băm, và cho phép truy cập nhanh chóng đến các giá trị dữ liệu bằng cách sử dụng khóa.

4. Cây tìm kiếm nhị phân (Binary Search Tree): Lưu trữ các giá trị dữ liệu trên một trục đơn trong các nút của cây tìm kiếm nhị phân, và cho phép truy cập nhanh chóng đến các giá trị dữ liệu bằng cách sử dụng thuật toán tìm kiếm nhị phân.

5. Cây cân bằng (Balance Tree): lưu trữ và truy vấn dữ liệu đơn chiều theo thứ tự nhất định. Khi thêm phần tử mới, cây được cập nhật để đảm bảo cân bằng giữa chiều cao của các nhánh con bên trái và bên phải. Truy vấn phần tử trong cây cân bằng được thực hiện trong thời gian O(log n).

Các kiểu dữ liệu này thường được sử dụng để lưu trữ và truy vấn dữ liệu đơn chiều. Tuy nhiên, cùng với sự phát triển và nhu cầu để lưu trữ và truy vấn về vị trí và mối quan hệ giữa các đối tượng trong không gian đa chiều tăng lên, tuy nhiên hiệu quả của chúng sẽ bị giảm xuống nghiêm trọng.

1. Array: Mảng có thể được sử dụng để lưu trữ các giá trị dữ liệu trong một không gian đa chiều bằng cách **sử dụng một mảng đa chiều**. O(n^m) vs m là số chiều

2. Linked list: Mỗi node có thêm thuộc tính chiều và truy vấn cũng sẽ duyệt tuyến tính từng chiều như dùng Array. O(n^m) vs m là số chiều

3. Hash table: Hash table được thiết kế để lưu trữ và truy vấn dữ liệu đơn chiều dưới dạng cặp key-value. Tuy nhiên, nếu muốn lưu trữ và truy vấn dữ liệu đa chiều, bạn có thể sử dụng một số kỹ thuật như kết hợp các hash table với nhau hoặc sử dụng các cấu trúc dữ liệu khác như cây bảo tồn (trie).

4. Binary tree: Cây tìm kiếm nhị phân có thể được sử dụng để lưu trữ và truy vấn dữ liệu đa chiều bằng cách sử dụng các nút trong cây để lưu trữ các giá trị dữ liệu. Mỗi nút trong cây có thể chứa một giá trị dữ liệu và hai con trỏ đến các cây con trái và phải. Sử dụng các kỹ thuật như kết hợp các binary tree với nhau hoặc sử dụng các cấu trúc dữ liệu khác như quad tree hoặc kd-tree. Bởi vì Binary Tree chỉ sắp xếp sẵn 1 chiều dữ liệu (các chiều còn lại thì không được nên truy vấn cũng như lưu trữ đều không hiệu quả.

5. Balace Tree: giống như Binary Tree khiến chúng đều trở nên kém hiệu quả.

Do vậy chúng ta có:

Quadtree: được sử dụng để lưu trữ và truy vấn dữ liệu trong không gian hai chiều. Quadtree chia không gian hai chiều thành các ô vuông con và mỗi ô vuông được đại diện bởi một nút trong cây. Các nút con của một nút đại diện cho các ô vuông con nhỏ hơn và các phần tử được lưu trữ trong các nút lá của cây.

Octree: được sử dụng để lưu trữ và truy vấn dữ liệu trong không gian ba chiều. Octree chia không gian ba chiều thành các ô lập phương con và mỗi ô lập phương được đại diện bởi một nút trong cây. Các nút con của một nút đại diện cho các ô lập phương con nhỏ hơn và các phần tử được lưu trữ trong các nút lá của cây.

Nhưng vì không gian bị chia đều thành các phần bằng nhau như vậy nên cũng vẫn chưa hiệu quả.

Cho đến 1984, Antonin Guttman đã đề xuất ra R Tree để giải quyết nỗ lực lưu trữ và truy vấn các dữ liệu đa chiều hiệu quả dù vẫn có 1 số nhược điểm nhất định. Sau đó, hàng loạt phiên bản biến thể của R Tree ra đời để khắc phục nhược điểm cụ thể của nó trong từng lĩnh vực.

Đây là 1 cái nhìn thoáng qua về R Tree (cụ thể là biến thể R Star Tree với các điểm dữ liệu trên khắp Trái Đất). Ý tưởng của R Tree khác Quadtree hay Octree ở chỗ thay vì chia đều không gian thành các phần bằng nhau thì các điểm dữ liệu hay đối tượng đa chiều sẽ được giới hạn trong 1 vùng không gian (tạm gọi là hình chữ nhật đa chiều) mà đặc điểm để tạo ra giới hạn vùng không gian này phụ thuộc vào nhiều quy tắc phân chia - sự khác nhau của các quy tắc cho ta các phiên bản khác so với R Origin.

Cụ thể chúng ta cần quan tâm 2 khái niệm quan trọng là Đối tượng đa chiều và Minimum Bounding Rectangle (MBR) tức vùng không gian hình chữ nhật đa chiều nhỏ nhất có thể chứa Đối tượng đa chiều đó.

Với ý tưởng cơ bản nêu trên, ta có thể hình dung 1 cây với các node lá đều cân bằng cùng 1 độ cao là gồm các đối tượng đa chiều. Khác với Balace Tree truyền thống, các node trong của R Tree không chứa đối tượng đa chiều mà sẽ chứa vùng không gian giới hạn (mà từ nay sẽ được gọi là MBR) chứa đối tượng đa chiều đó.

Vậy câu hỏi đặt ra là điều gì giúp ta định nghĩa lên các MBR (vùng không gian hình chữ nhật đa chiều nhỏ nhất chứa đối tượng). R Tree bởi vậy phải có thêm 2 thuộc tính đặc biệt là giới hạn tối thiểu và tối đa mà mỗi node trong cây có thể chứa các MBR hay các Object. Thông thường để đảm bảo cây ổn định thì m = M/2. Với giới hạn tối thiểu và tối đa này thì lại nảy sinh vấn đề khi chèn hoặc xóa đó là ta có thể phải gộp node hoặc tách node. Lúc này ta lại cần định nghĩa quy luật gộp tách node này.

Như vậy, các phiên bản của R Tree từ sự khác biệt ở cách xác định (phân chia) MBR nay đã quy thành sự khác biệt của cách gộp tách 1 node trong cây. Đối tượng NodeSplitter ở đây chính là điểm khác biệt cho các phiên bản của cây R và như vậy tùy từng trường hợp sử dụng trong các lĩnh vực cụ thể mà ta có thể tùy chỉnh cho NodeSplitter phù hợp và tạo ra phiên bản R Tree của riêng mình.

Từ ý tưởng cơ bản và cách triển khai cấu trúc R Tree nêu trên, ta có thể tính ra số node tối đa trên mỗi mức độ (lá là mức 1 và root là mức cao nhất) từ thông tin chiều cao, số lượng đối tượng, và giới hạn tối thiểu có trong node. Chiều cao tối đa lại có thể tính nhờ giới hạn tối thiểu và số đối tượng có trong cây (chú ý đối tượng đa chiều sẽ chỉ ở node lá).

Sau khi đã hiểu sâu sắc về cấu trúc của R Tree chúng ta sẽ cùng tìm hiểu về 1 số thuật toán cập nhật và truy vấn có trên R Tree.

# Đức Anh.

1. Thuật toán chèn:
2. Thuật toán xóa:

# Thắng.

1. Thuật toán range search:
2. Thuật toán knn:
3. Thuật toán exact search (cái tội lười không làm):

# Hải.

1. Demo project:

Thử với 10 điểm data 2D cho thêm, xóa, truy vấn phạm vi và truy vấn lân cận gần nhất.

Tuy nhiên với lượng dữ liệu data lớn hơn như 36800 điểm data dữ liệu địa lý của 4 quận nội thành Hà Nội thì nó có phần chậm hơn với chương trình đồ họa này.

Bởi vậy chúng tôi đã thử cách mô phỏng chúng qua 1 ảnh có độ phân giải cao và sử dụng trình xem ảnh mặc định để mượt mà hơn. Sau đây với lượng dữ liệu lớn này thì chúng tôi sẽ thực hiện nhiều loại truy vấn hơn và kết quả được thể hiện qua terminal.

1. Các ứng dụng của R Tree:

Các video đây là 1 cái nhìn thoáng qua về cách R Tree được sử dụng trong giai đoạn đầu. Nó được sử dụng trong xây dựng và truy vấn các dữ liệu địa lý đa chiều như ản đồ đa chiều. R-tree giúp tìm kiếm các chip điện tử nhanh chóng và cho phép thực hiện các truy vấn phức tạp như tìm kiếm các chip điện tử nằm trong một vùng xác định hoặc tìm kiếm các chip điện tử gần nhau trong thiết kế VLSI (*Very large-scale integration)* - liên quan đến thiết kế và sản xuất các mạch tích hợp có kích thước rất nhỏ trên một mảng bán dẫn. Hay như sử dụng với cơ sở dữ liệu thiên văn học, như sao, hành tinh và thiên thạch trong không gian ba chiều. R-tree giúp tìm kiếm các đối tượng thiên văn nhanh chóng và cho phép thực hiện các truy vấn phức tạp như tìm kiếm các đối tượng nằm trong một vùng xác định hoặc tìm kiếm các đối tượng gần nhau.

Giờ đây, với sự đa dạng của dữ liệu với ngày càng nhiều kiểu dữ liệu với nhiều chiều thông tin thì R Tree càng được ứng dụng rộng rãi. Như để quản lý tài sản, hệ thống trò chơi điện tử, y tế, giao thông hay phân tích dữ liệu lớn.

1. Ưu - Nhược điểm của R Tree:

Quay lại câu hỏi đặt ra từ đầu bài luận, đã từng có nhiều nỗ lực để giải quyết vấn đề lưu trữ và truy vấn dữ liệu đa chiều. Câu hỏi tại sao R Tree lại được sử dụng để giải quyết vấn đề này 1 cách phổ biến hơn cả.

Từ thông tin về cấu trúc triển khai và các thuật toán có trên nó, có thể thấy rằng R Tree giúp truy vấn dữ liệu đa chiều 1 cách nhanh chóng.

Nó dễ dàng mở rộng, cập nhật khi thêm hay xóa. Khi số lượng đối tượng tăng lên, R-Tree có thể được mở rộng dễ dàng. Nó cho phép thêm hoặc xoá các đối tượng một cách linh hoạt mà không cần tái tạo toàn bộ cây.

R Tree đã là giải pháp tối ưu không gian lưu trữ mã vẫn đạt được hiệu suất cập nhật và truy vấn nêu trên. Phương pháp tối ưu hóa không gian để giảm thiểu số lượng không gian lưu trữ được sử dụng để lưu trữ các đối tượng trong không gian đa chiều (Điểm khác biệt lớn so với Quadtree hay Occtree). Điều này giúp giảm thiểu tải cho hệ thống lưu trữ.

R Tree ngoài ra còn hỗ trợ nhiều loại truy vấn phức tạp bên cạnh 2 loại truy vấn (range search và knn search), có thể kể đến truy vấn chính xác điểm dữ liệu (exact search), truy vấn giao hay kết hợp nhiều truy vấn.

Bên cạnh đó vẫn tồn tại 1 số nhược điểm của cây R.

Không phù hợp với các truy vấn liên quan đến thời gian. Ví dụ như truy vấn các đối tượng trong khoảng thời gian nhất định. Điều này yêu cầu sự hỗ trợ của một cấu trúc dữ liệu khác như R\*-Tree, …

Không phù hợp cho các đối tượng lớn. Vì nó có thể tốn nhiều bộ nhớ và thời gian để tìm kiếm các đối tượng trong không gian đa chiều.

Không phù hợp với trường hợp động. Vì nó yêu cầu một quá trình tái xây dựng cây để cập nhật các đối tượng mới.

Không hoạt động tốt với các kiểu dữ liệu rời rạc. R-tree được thiết kế để làm việc với các đối tượng liên tục trong không gian, ví dụ như các hình chữ nhật, vòng tròn, hay các đa giác. Khi sử dụng R-tree với các kiểu dữ liệu rời rạc (các lưới (grid), matrix, mạng lưới (networks) thậm chí dùng graph hoặc tree hiệu quả hơn, ...), các đối tượng đó sẽ được chuyển đổi thành các đối tượng liên tục trong không gian, dẫn đến mất mát thông tin và làm giảm hiệu suất của R-tree.

1. Tối ưu:

Với các trường hợp nhược điểm nêu trên, tùy từng lĩnh vực ứng dụng mà đã có nhiều biến thể ra đời (thay đổi cách phân tách, gộp node nói riêng và tiêu chí xác định 1 MBR nói chung) để khắc phục các nhược điểm trên.

1. Kết:

Hành trình phát triển của R Tree vẫn sẽ tiếp tục và đồng hành cùng sự tiến bộ của loài người. Từ dự án tạo lập R Tree cho 4 quận ở Hà Nội biết đâu 1 ngày nó sẽ còn được dùng cho những mục tiêu lớn hơn như xây dựng bản đồ 3D cho hành trình con người chinh phục sao Hỏa hay tạo 1 bản đồ ảo mô tả trực tiếp thế giới thực theo thời gian thực để từ đó các thiết bị và nhiều phương tiện kết nối truy vấn trực tiếp và tạo nên 1 đô thị thông minh hơn. Sự phát triển vẫn sẽ tiếp diễn.