(12) BẢN MÔ TẢ SÁNG CHẾ THUỘC BẰNG ĐỘC QUYỀN SÁNG CHẾ

(19) **Cộng hòa xã hội chủ nghĩa Việt Nam (VN)**
CỤC SỞ HỮU TRÍ TUỆ

(11) **1-0051228**

(51)[2020.01] **G08G 1/00**

(13) **B**

(21) 1-2021-07244
(22) 12/04/2019
(86) PCT/CN2019/082349 12/04/2019
(87) WO2020/206665 15/10/2020
(45) 25/09/2025 450
(43) 25/01/2022 406A
(73) GRABTAXI HOLDINGS PTE. LTD. (SG)
3 Media Close, #01-03/06, Singapore 138498, Singapore
(72) ZHANG, Zhiyin (CN); HUANG, Xiaocheng (CN); SUN, Chaotang (CN); ZHENG, Shaolin (CN).
(74) Công ty TNHH Tầm nhìn và Liên danh (VISION & ASSOCIATES CO.LTD.)

(54) HỆ THỐNG CƠ SỞ DỮ LIỆU, PHƯƠNG PHÁP LƯU TRỮ DỮ LIỆU, KHO DỮ LIỆU KHÔNG GIAN TRONG BỘ NHỚ MỞ RỘNG ĐƯỢC CHO TÌM KIẾM K LÂN CẬN GẦN NHẤT VÀ PHƯƠNG PHÁP TĂNG TỐC TÌM KIẾM LÂN CẬN GẦN NHẤT

(21) 1-2021-07244

(57) Sáng chế đề cập đến hệ thống cơ sở dữ liệu được tạo cấu hình để cho phép tìm kiếm nhanh các lân cận gần nhất với đối tượng di động nằm trong không gian địa lý được tạo thành từ các không gian con khác biệt về không gian, mỗi không gian con này được tạo thành từ các ô. Hệ thống cơ sở dữ liệu có hệ điều hành điều khiển việc lưu trữ dữ liệu đối tượng trong số các nút lưu trữ, để biểu diễn một hoặc nhiều không gian con khác biệt về không gian, trong một nút lưu trữ tương ứng trong số các nút lưu trữ. Dữ liệu vị trí của mỗi đối tượng được sử dụng để lập chỉ mục rằng đối tượng đối với các ô tạo thành mỗi không gian con khác biệt về không gian trong mỗi nút.

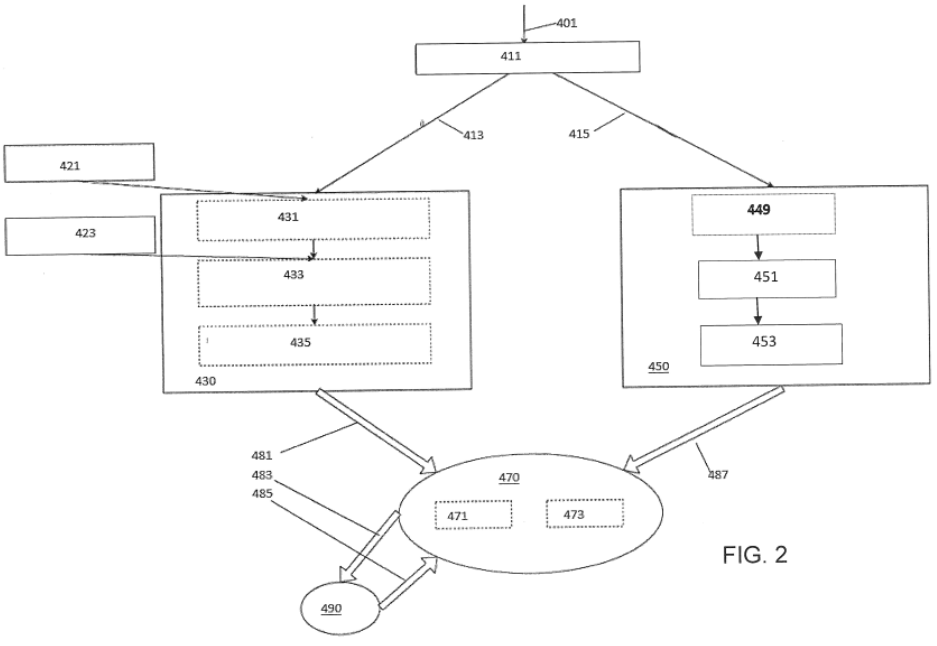


FIG. 2

Lĩnh vực kỹ thuật được đề cập

Nói chung sáng chế đề cập đến lưu trữ và truy xuất dữ liệu. Cụ thể hơn nhưng không giới hạn là sáng chế đề cập đến các hệ thống cơ sở dữ liệu để tạo điều kiện thuận lợi cho việc tìm kiếm K lân cận gần nhất. Một phương án làm ví dụ là trong lĩnh vực quản lý dịch vụ gọi xe.

Tình trạng kỹ thuật của sáng chế

Trong kịch bản gọi xe điển hình, người dùng tiềm năng đặt yêu cầu đặt xe thông qua ứng dụng điện thoại thông minh, mà sau đó được hoàn thành bởi máy chủ bằng cách điều phối người cung cấp dịch vụ ở gần, thích hợp nhất khả dụng để cấp dịch vụ được yêu cầu.

Việc định vị các đối tượng di chuyển gần nhất (ví dụ những người lái xe) ở thời gian thực là một trong số các vấn đề cơ bản mà dịch vụ gọi xe cần giải quyết. Máy chủ theo dõi các vị trí địa lý theo thời gian thực của các người cung cấp dịch vụ và tìm kiếm K người cung cấp dịch vụ khả dụng gần vị trí của người dùng cho mỗi yêu cầu đặt xe vì người cung cấp dịch vụ gần nhất có thể không phải luôn là lựa chọn tốt nhất. Để đơn giản hóa vấn đề, khoảng cách theo đường thẳng có thể được sử dụng thay vì khoảng cách định tuyến.

Không giống như nghiên cứu hiện có về các truy vấn K lân cận gần nhất (K-nearest neighbour, kNN) so với các đối tượng tĩnh chẳng hạn như định vị K nhà hàng gần nhất, hoặc các truy vấn K lân cận gần nhất liên tục so với các đối tượng di chuyển, chẳng hạn như tìm kiếm K trạm xăng gần nhất đến xe đang chạy, vấn đề liên quan đến các đối tượng di chuyển mà thực thi các truy vấn K gần nhất động, vấn đề này đặt ra nhiều thách thức.

Tình trạng kỹ thuật

Tìm kiếm K lân cận gần nhất đối với các đối tượng tĩnh, chẳng hạn như định vị các nhà hàng gần nhất, tập trung vào việc lập chỉ mục các đối tượng đúng. Có hai cách lập chỉ mục chính: dựa trên đối tượng và dựa trên nghiệm.

Việc lập chỉ mục dựa trên đối tượng nhắm vào các vị trí của các đối tượng. Cây R sử dụng các hình chữ nhật giới hạn tối thiểu để xây dựng chỉ mục phân cấp trong đó K lân cận gần nhất có thể được tính toán bởi kết nối không gian. Cách dựa trên nghiệm tập trung vào việc lập chỉ mục không gian nghiệm được tính toán trước, ví dụ phân chia không gian nghiệm dựa trên sơ đồ Voronoi và tính toán trước kết quả tìm kiếm lân cận gần nhất bất kỳ tương ứng với ô Voronoi. Các cách khác kết hợp hai cách trên và đề xuất chỉ mục phân chia dạng lưới để lưu trữ các đối tượng là các lân cận gần nhất tiềm năng của truy vấn bất kỳ nằm trong các ô Voronoi.

Để tăng tốc các truy vấn kNN trên các đối tượng tĩnh dựa trên chỉ mục, đã đề xuất phát triển thuật toán nhánh và ràng buộc dựa trên cây R để thực hiện tìm kiếm đầu tiên tốt nhất trong khi duy trì danh sách ưu tiên của K đối tượng gần nhất.

Một cách khác nghiên cứu các truy vấn kNN di chuyển so với các đối tượng tĩnh, trong đó cách này trả về nhiều K mục kết quả hơn sao cho truy vấn K gần nhất ở vị trí mới cũng sẽ được bao gồm trong kết quả trước.

Tuy nhiên, việc duy trì các chỉ mục phức tạp này đối với các đối tượng di chuyển cần phải cập nhật vị trí thường xuyên.

Việc lập chỉ mục các đối tượng di chuyển/di động có thể được phân loại thành hai mục: (1) lập chỉ mục các vị trí hiện tại và dự đoán trong tương lai của các đối tượng di chuyển và (2) lập chỉ mục các quỹ đạo.

Một nghiên cứu trước đó tập trung vào việc lập chỉ mục các vị trí hiện tại và dự đoán trong tương lai của các đối tượng di chuyển và đề xuất chỉ mục cây R được tham số hóa theo thời gian (nghĩa là, cây TPR). Các hình chữ nhật giới hạn trong cây TPR là các hàm thời gian và liên tục theo các điểm dữ liệu được bao gồm hoặc các hình chữ nhật khác khi chúng di chuyển.

Các cách lập chỉ mục quỹ đạo đề xuất cây chùm quỹ đạo (Trajectory-Bundle, TB) để lưu giữ các quỹ đạo lịch sử và cho phép tìm kiếm phạm vi cây R điển hình. Lưu ý rằng trong thiết lập của sáng chế, các quỹ đạo quá khứ của các đối tượng không được quan tâm.

Tìm kiếm K lân cận gần nhất liên tục so với các đối tượng tĩnh cũng thu hút sự chú ý, ví dụ tìm ba trạm xăng gần nhất đối với xe đang chạy trên điểm bất kỳ dọc theo đường đi được định rõ trước.

Trái ngược với các cách truyền thống trong đó các đối tượng được lập chỉ mục, một cách khác xây dựng các chỉ mục trên cả hai truy vấn (nghĩa là, truy vấn Q) và các đối tượng (nghĩa là, lập chỉ mục giới hạn vận tốc (Velocity Constrained Indexing, VCI)). Một cách khác nữa giả định các đối tượng đang di chuyển không đổi với vận tốc hiện thời và do đó có thể suy ra K đối tượng gần nhất ở dấu thời gian tương lai. Nhiều nghiên cứu hơn nữa về giám sát truy vấn liên tục đều chú ý đến lập chỉ mục các truy vấn. Tuy nhiên, các phương pháp này hoặc là thực hiện các giả định về cách truy vấn di chuyển (ví dụ, theo quỹ đạo) hoặc giả định chỉ mục toàn cầu trong bộ nhớ.

Cần lưu ý rằng việc mở rộng các kỹ thuật lập chỉ mục phức tạp nêu trên là không đơn giản khi có khối lượng hoạt động ghi lớn. Cấu trúc chỉ mục đơn giản mà dễ dàng mở rộng cho cả hai hoạt động đọc và ghi phù hợp cho các ứng dụng trong đời thực.

Các cơ sở dữ liệu đối tượng di chuyển là rất thách thức. Một cách xem xét các cơ sở dữ liệu theo dõi và cập nhật các vị trí của các đối tượng di chuyển. Mặc dù trọng tâm là quyết định thời điểm vị trí của đối tượng di chuyển trong cơ sở dữ liệu cần được cập nhật. Các cơ sở dữ liệu không gian quản lý dữ liệu không gian và các truy vấn hệ thống thông tin địa lý (Geographic Information Systems, GIS) hỗ trợ chẳng hạn như việc liệu điểm truy vấn có nằm trong vùng đa giác hay không.

Vấn đề kỹ thuật là các cơ sở dữ liệu không thích hợp để xử lý các tải ghi nặng vì các chi phí I/O lớn.

Các kho dữ liệu giá trị khóa trong bộ nhớ mở rộng được mở rộng tốt khi ghi thường xuyên. Trong kho dữ liệu giá trị khóa, các đối tượng là các khóa còn các giá trị của chúng là các giá trị. Việc trả lời tìm kiếm K gần nhất do đó cần quét tất cả các khóa, độ trễ của nó là không thể chấp nhận.

Bản chất kỹ thuật của sáng chế

Theo khía cạnh thứ nhất, sáng chế đề xuất kho dữ liệu không gian trong bộ nhớ mở rộng được được điều chỉnh cho các tìm kiếm kNN, trong đó lưu trữ dữ liệu là phi tập trung.

Theo khía cạnh thứ hai, sáng chế đề xuất hệ thống và phương pháp để định vị các đối tượng di chuyển gần nhất (những người lái xe) ở thời gian thực.

Theo khía cạnh thứ ba, sáng chế đề xuất hệ thống cơ sở dữ liệu được tạo cấu hình để cho phép tìm kiếm nhanh các lân cận gần nhất với đối tượng di chuyển nằm trong không

gian địa lý được tạo thành từ các phân đoạn không gian phân biệt theo không gian, mỗi không gian con này được tạo thành từ các ô, được tạo cấu hình để điều khiển việc lưu trữ dữ liệu đối tượng trong số các nút lưu trữ, bằng cách này dữ liệu được lưu trữ theo cách phi tập trung, với dữ liệu vị trí của mỗi đối tượng di chuyển được sử dụng để lập chỉ mục rằng đối tượng đối với các ô tạo thành mỗi phân đoạn phân biệt theo không gian trong mỗi nút.

Theo khía cạnh thứ tư, sáng chế đề xuất hệ thống cơ sở dữ liệu được tạo cấu hình để cho phép tìm kiếm nhanh các lân cận gần nhất với đối tượng nằm trong không gian địa lý được tạo thành từ các không gian con khác biệt về không gian, mỗi không gian con này được tạo thành từ các ô, hệ thống cơ sở dữ liệu bao gồm các nút lưu trữ; và hệ điều hành, hệ điều hành này được tạo cấu hình để điều khiển việc lưu trữ dữ liệu đối tượng trong số các nút, trong đó hệ điều hành được tạo cấu hình để thực hiện việc lưu trữ dữ liệu biểu diễn một hoặc nhiều không gian con khác biệt về không gian trong một nút lưu trữ tương ứng trong số các nút lưu trữ, và trong đó dữ liệu vị trí của mỗi đối tượng được sử dụng để lập chỉ mục rằng đối tượng đối với các ô tạo thành mỗi không gian con khác biệt về không gian trong mỗi nút.

Theo khía cạnh khác, sáng chế đề xuất phương pháp lưu trữ dữ liệu để cho phép tìm kiếm nhanh các lân cận gần nhất với đối tượng nằm trong không gian địa lý được tạo thành từ các không gian con khác biệt về không gian, mỗi không gian con này được tạo thành từ các ô, hệ thống cơ sở dữ liệu bao gồm các nút lưu trữ; phương pháp này bao gồm bước lưu trữ dữ liệu đối tượng trong số các nút lưu trữ, sao cho dữ liệu biểu diễn một hoặc nhiều không gian con khác biệt về không gian được lưu trữ trong một nút lưu trữ tương ứng trong số các nút lưu trữ, và sử dụng dữ liệu vị trí của mỗi đối tượng để lập chỉ mục rằng đối tượng đối với các ô tạo thành mỗi không gian con khác biệt về không gian trong mỗi nút lưu trữ.

Theo khía cạnh khác nữa, sáng chế đề xuất phương pháp tăng tốc tìm kiếm lân cận gần nhất bao gồm bước phân bố dữ liệu trong các nút lưu trữ theo mối quan hệ địa lý giữa dữ liệu, nhờ đó cho phép việc tìm kiếm dữ liệu được thực hiện nhờ sử dụng số lượng cuộc gọi từ xa được giảm bớt.

Theo khía cạnh khác, sáng chế đề xuất kho dữ liệu không gian trong bộ nhớ mở rộng được cho các tìm kiếm kNN bao gồm hệ thống cơ sở dữ liệu như được nêu trong khía

cạnh thứ tư.

Theo một phương án, dữ liệu của mỗi không gian con khác biệt về không gian được lưu trữ hoàn toàn trong nút lưu trữ duy nhất.

Theo các phương án, dữ liệu của mỗi không gian con khác biệt về không gian được sao chép đến các nút lưu trữ để tạo ra các bản sao dữ liệu.

Theo các phương án, các hoạt động ghi liên quan đến không gian con khác biệt về không gian được truyền đến tất cả các bản sao dữ liệu liên quan. Giao thức bỏ phiếu dựa trên túc số có thể được sử dụng.

Số lượng bản sao, theo một số phương án, có thể tạo cấu hình dựa trên các trường hợp sử dụng.

Theo một số phương án, thuật toán tìm kiếm theo chiều rộng trả lời các truy vấn K lân cận gần nhất

Theo một họ các phương án, dữ liệu được lưu trữ trong các nút lưu trữ nhờ sử dụng băm nhất quán nhờ đó chỉ định cho vòng băm trừu tượng.

Theo họ các phương án khác, dữ liệu được lưu trữ trong các nút lưu trữ nhờ sử dụng phép ánh xạ tạo cấu hình được bởi người dùng từ các không gian con đến các nút lưu trữ mà xác định rõ ràng không gian con nào thuộc về nút lưu trữ nào.

Theo họ khác nữa, cả phép ánh xạ tạo cấu hình được bởi người dùng từ các không gian con đến các nút lưu trữ mà xác định rõ ràng không gian con nào thuộc về nút nào và phép băm nhất quán được sử dụng cho dữ liệu khác nhau.

Dữ liệu trong cơ sở dữ liệu, theo tập hợp các phương án, được lưu trữ trong bộ nhớ.

Đối với dữ liệu không có trong phép ánh xạ, băm nhất quán có thể được sử dụng.

Một nút trong phép ánh xạ có thể được sử dụng làm bộ điều phối tĩnh để phát rộng các kết nối mới.

Truyền thông điệp kiểu tin đồn có thể được sử dụng để cho phép phát hiện nút.

Các đối tượng có thể đang di chuyển hoặc ít nhất là di động và có thể là các xe của người cung cấp dịch vụ của hệ thống gọi xe.

Hệ thống cơ sở dữ liệu này có thể được tạo cấu hình để giải quyết vấn đề về khối lượng hoạt động ghi bằng cách phân bố dữ liệu đến các nút khác nhau và lưu trữ trong bộ

nhớ.

Theo khía cạnh khác, sáng chế đề xuất hệ thống cơ sở dữ liệu trong đó hệ điều hành phân bố dữ liệu trong các nút lưu trữ theo mối quan hệ địa lý giữa dữ liệu, nhờ đó cho phép việc tìm kiếm dữ liệu được thực hiện nhờ sử dụng số lượng cuộc gọi từ xa được giảm bớt.

Mô tả vắn tắt các hình vẽ

Trên các hình vẽ khác nhau:

Fig.1 thể hiện sơ đồ khối cục bộ của hệ thống truyền thông làm ví dụ để sử dụng trong dịch vụ gọi xe;

Fig.2 thể hiện lưu đồ của kỹ thuật để tìm kiếm các lân cận gần nhất;

Fig.3 là sơ đồ của BFS cho việc tìm kiếm K gần nhất;

Fig.4 thể hiện thuật toán tìm kiếm K gần nhất thô;

Fig.5 thể hiện thuật toán tìm kiếm K gần nhất được tối ưu hóa;

Fig.6 thể hiện số lượng ô được ghé thăm trung bình trong phân đoạn được ghé thăm;

Fig.7 thể hiện các so sánh của băm với ánh xạ bảng phân đoạn (ShardTable);

Fig.8 thể hiện các kết quả khôi phục thất bại;

Fig.9 là bảng so sánh các phép tính cho các chỉ mục không gian địa lý khác nhau; và

Fig.10 thể hiện sơ đồ khối được đơn giản hóa cao của kiến trúc của cơ sở dữ liệu được phân bố.

Mô tả chi tiết sáng chế

Như được sử dụng trong tài liệu này, "cơ sở dữ liệu" là cấu trúc với hệ thống điều hành và quản lý, cấu trúc bao gồm bộ nhớ, và hệ thống điều hành và quản lý được tạo cấu hình để lưu trữ dữ liệu vào trong bộ nhớ theo cách để tạo điều kiện thuận lợi cho việc tìm kiếm dữ liệu được lưu trữ trong bộ nhớ.

Trong đó cơ sở dữ liệu có thể được coi là có các hàng logic mỗi biểu diễn đối tượng, và các cột logic, mỗi cột biểu diễn thuộc tính của các đối tượng, "bộ (tuple)" là hàng duy nhất biểu diễn tập hợp các thuộc tính của đối tượng cụ thể.

"Băm" là sự biến đổi của chuỗi ký tự thành mục dữ liệu được gọi là "khóa" biểu

diễn chuỗi ban đầu. Băm được sử dụng để lập chỉ mục và truy xuất các mục trong cơ sở dữ liệu vì việc tìm mục này nhờ sử dụng khóa băm ngắn hơn nhanh hơn so với việc tìm mục này nhờ sử dụng giá trị ban đầu.

"Băm nhất quán" là sơ đồ băm phân tán hoạt động độc lập với số lượng nút hoặc các đối tượng trong bảng băm phân tán bằng cách chỉ định cho chúng vị trí trên vòng trừu tượng hoặc vòng băm. Điều này cho phép các nút và các đối tượng được thêm vào hoặc loại bỏ mà không ảnh hưởng đến hệ thống tổng thể.

"Phân đoạn" đề cập đến việc phân chia cơ sở dữ liệu thành các bộ dữ liệu duy nhất, cho phép dữ liệu được phân bố đến nhiều máy chủ, nhờ đó tăng tốc tìm kiếm dữ liệu. Điển hình là, phân vùng ngang của cơ sở dữ liệu. Trong ngữ cảnh hiện tại, các bộ dữ liệu duy nhất, mỗi bộ dữ liệu này biểu diễn vùng phân biệt theo địa lý tương ứng, mỗi vùng này được gọi là phân đoạn.

Thuật ngữ "phân đoạn" còn được sử dụng ở đây để xác định nội dung dữ liệu của mỗi vùng, để tham chiếu đến phân đoạn x của dữ liệu tham chiếu đến tập hợp dữ liệu đối với phân đoạn địa lý x.

Tìm kiếm K lân cận gần nhất (tìm kiếm kNN) là tìm kiếm mà nhận dạng K lân cận gần nhất với đối tượng được xem xét.

"Redis" (Remote Dictionary Server - máy chủ từ điển từ xa) là kiểu máy chủ cấu trúc dữ liệu có thể sử dụng làm cơ sở dữ liệu với khả năng đọc-ghi rất cao.

"Cơ sở dữ liệu trong bộ nhớ" (in-memory database, IMDB) còn được gọi là hệ thống cơ sở dữ liệu bộ nhớ chính (main memory database system, MMDB) là hệ thống quản lý cơ sở dữ liệu mà chủ yếu dựa vào bộ nhớ chính để lưu trữ dữ liệu máy tính. Truy cập dữ liệu trong bộ nhớ làm giảm hoặc loại bỏ thời gian tìm kiếm khi truy vấn dữ liệu.

Thuật ngữ "các bộ bản sao" là để chỉ các phiên bản được lưu trữ riêng của cùng dữ liệu.

Đầu tiên dựa vào Fig.1, hệ thống truyền thông 100 dùng cho ứng dụng gọi xe được minh họa. Hệ thống truyền thông 100 bao gồm thiết bị máy chủ truyền thông 102, thiết bị truyền thông người cung cấp dịch vụ 104, còn được gọi ở đây là thiết bị người cung cấp dịch vụ và thiết bị truyền thông máy khách 106. Các thiết bị này được kết nối trong mạng truyền thông 108 (ví dụ mạng Internet) thông qua các liên kết truyền thông tương ứng 110,

112, 114 thực hiện, ví dụ, các giao thức truyền thông mạng internet. Các thiết bị truyền thông 104, 106 có thể có thể truyền thông thông qua các mạng truyền thông khác, chẳng hạn như các mạng điện thoại chuyển mạch công cộng (public switched telephone network, PSTN), bao gồm các mạng truyền thông tế bào di động, nhưng các mạng này được lược bỏ khỏi Fig.1 để rõ ràng.

Thiết bị máy chủ truyền thông 102 có thể là máy chủ duy nhất như được minh họa giản lược trên Fig.1, hoặc có chức năng được thực hiện bởi thiết bị máy chủ 102 được phân tán trên nhiều thành phần máy chủ. Trong ví dụ của Fig.1, thiết bị máy chủ truyền thông 102 có thể bao gồm nhiều thành phần riêng lẻ bao gồm, nhưng không bị giới hạn ở, một hoặc nhiều bộ xử lý 116, bộ nhớ 118 (ví dụ bộ nhớ khả biến chẳng hạn như RAM) để tải các lệnh có thể thực thi 120, các lệnh có thể thực thi xác định chức năng thiết bị máy chủ 102 thực hiện dưới sự điều khiển của bộ xử lý 116. Thiết bị máy chủ truyền thông 102 còn bao gồm môđun đầu vào/đầu ra 122 cho phép máy chủ truyền thông qua mạng truyền thông 108. Giao diện người dùng 124 được bố trí để người dùng điều khiển và có thể bao gồm, ví dụ, các thiết bị ngoại vi tính toán thông thường chẳng hạn như các màn hình hiển thị, bàn phím máy tính và thiết bị tương tự. Thiết bị máy chủ 102 còn bao gồm cơ sở dữ liệu 126, một mục đích của thiết bị này là để lưu trữ dữ liệu khi dữ liệu được xử lý, và làm cho dữ liệu này khả dụng dưới dạng dữ liệu lịch sử trong tương lai.

Thiết bị người cung cấp dịch vụ 104 có thể bao gồm nhiều thành phần riêng lẻ bao gồm, nhưng không bị giới hạn ở, một hoặc nhiều bộ vi xử lý 128, bộ nhớ 130 (ví dụ bộ nhớ khả biến chẳng hạn như RAM) để tải các lệnh có thể thực thi 132, các lệnh có thể thực thi xác định chức năng thiết bị người cung cấp dịch vụ 104 thực hiện dưới sự điều khiển của bộ xử lý 128. Thiết bị người cung cấp dịch vụ 104 còn bao gồm môđun đầu vào/đầu ra 134 cho phép thiết bị người cung cấp dịch vụ 104 truyền thông qua mạng truyền thông 108. Giao diện người dùng 136 được bố trí để người dùng điều khiển. Nếu thiết bị người cung cấp dịch vụ 104 là, chẳng hạn, điện thoại thông minh hoặc thiết bị dạng bảng, thì giao diện người dùng 136 sẽ có màn hiển thị cảm ứng như phổ biến trong nhiều điện thoại thông minh và các thiết bị cầm tay khác. Theo cách khác, nếu thiết bị truyền thông người cung cấp dịch vụ là, chẳng hạn, máy tính để bàn hoặc máy tính xách tay thông thường, thì giao diện người dùng có thể có, ví dụ, các thiết bị ngoại vi tính toán thông thường chẳng hạn như các màn hình hiển thị, bàn phím máy tính và thiết bị tương tự.

Thiết bị truyền thông máy khách 106 có thể là, ví dụ, điện thoại thông minh hoặc

thiết bị dạng bảng với kiến trúc giống hoặc tương tự với kiến trúc của thiết bị người cung cấp dịch vụ 104.

Khi sử dụng, theo một phương án, thiết bị người cung cấp dịch vụ 104 được lập trình để đẩy các gói dữ liệu đến thiết bị máy chủ truyền thông 102, ví dụ bằng cách gửi lệnh gọi API trực tiếp đến cơ sở dữ liệu. Các gói chứa thông tin, ví dụ thông tin biểu diễn ID của thiết bị người cung cấp dịch vụ 104, vị trí của thiết bị, dấu thời gian và dữ liệu khác biểu thị các khía cạnh khác chẳng hạn như ví dụ nếu người cung cấp dịch vụ bận hoặc nhàn rỗi.

Theo một số phương án, dữ liệu được đẩy được giữ trong hàng đợi để cho phép nó được truy cập bởi máy chủ 104 đồng bộ hóa với đồng hồ của máy chủ. Theo các phương án khác, dữ liệu được đẩy được truy cập ngay lập tức.

Theo các phương án khác nữa, thiết bị người cung cấp dịch vụ 104 đáp lại các yêu cầu thông tin từ máy chủ 102, thay vì đẩy dữ liệu đến máy chủ.

Theo các phương án khác nữa, dữ liệu được thu bằng cách lấy thông tin từ dòng dữ liệu được phát bởi thiết bị người cung cấp dịch vụ.

Theo các phương án trong đó dữ liệu được đẩy từ thiết bị người cung cấp dịch vụ, việc chuyển vào cơ sở dữ liệu của phương án có thể được thực hiện nhờ sử dụng các dòng Kafka. Trong đó không có phương tiện nào được sử dụng và số lượng nhỏ sự đẩy dữ liệu đồng thời xảy ra, cơ sở dữ liệu được tạo cấu hình để xử lý các dữ liệu này đồng thời. Trong đó số lượng lớn sự đẩy xảy ra, dữ liệu đến được giữ trong hàng đợi tin nhắn được thực hiện dưới dạng bộ nhớ FIFO.

Dữ liệu được đóng gói từ thiết bị người cung cấp dịch vụ 104 được sử dụng bởi máy chủ theo nhiều cách, ví dụ để so khớp các yêu cầu máy khách với các người cung cấp dịch vụ, để quản lý hệ thống gọi xe - ví dụ để tư vấn cho các người cung cấp dịch vụ về nơi trong đó công việc có khả năng hoặc trở nên khả dụng - và để lưu trữ dưới dạng cơ sở dữ liệu lịch sử 126.

Một số dữ liệu trong số dữ liệu được đóng gói được biến đổi thành các bộ dữ liệu để lưu trữ bởi cơ sở dữ liệu để thực hiện các tìm kiếm kNN.

Theo một phương án, bộ dữ liệu bao gồm bốn thuộc tính (id, loc, ts, siêu dữ liệu (metadata)) biểu diễn rằng đối tượng được nhận dạng duy nhất bởi id ở vị trí loc ở dấu thời

gian ts. Metadata định rõ trạng thái của đối tượng. Ví dụ, siêu dữ liệu của người cung cấp dịch vụ có thể biểu thị xem người cung cấp dịch vụ là người lái xe để đi chung xe hoặc người cung cấp dịch vụ xe máy để giao đồ ăn. Truy vấn tìm kiếm K gần nhất được biểu diễn dưới dạng (loc, ts, k) trong đó loc là tọa độ vị trí và ts là dấu thời gian. Cho truy vấn K gần nhất (loc, ts, k), cơ sở dữ liệu của một phương án trả về tối đa k bộ dữ liệu mà là gần nhất với vị trí truy vấn loc. Lưu ý rằng phương án này giả định khoảng cách theo đường thẳng.

Theo một họ các phương án, dấu thời gian truy vấn ts cũng được truy vấn để xác nhận tính hợp thời của các bộ dữ liệu vì tập trung vào các vị trí thời gian thực trong khoảng thời gian ngắn.

Cơ sở dữ liệu của phương án bao gồm kho dữ liệu phi tập trung trong đó dữ liệu được truyền qua các nút khác nhau để lưu trữ tại đó. Các bộ dữ liệu của các người cung cấp dịch vụ được định vị trong một hoặc nhiều phân đoạn địa lý được lưu trữ ở các nút tương ứng. Theo phương án hiện tại dữ liệu không bị lặp giữa các nút và chỉ một phiên bản duy nhất được ghi. Trong phạm vi có thể, phương án ghi các bộ dữ liệu biểu diễn các người cung cấp dịch vụ gần theo không gian với nhau để cho phép các tìm kiếm kNN nhanh chóng. Tuy nhiên cần lưu ý rằng trong đó người cung cấp dịch vụ thứ nhất quan tâm ở tại hoặc gần với biên của phân đoạn được lưu trữ ở một nút, có thể có các người cung cấp dịch vụ gần với người cung cấp dịch vụ thứ nhất nhưng được định vị thực tế trong phân đoạn lân cận mà có dữ liệu được lưu trữ ở nút khác.

Việc xác định nơi để lưu trữ dữ liệu đạt được bằng cách phân chia trước tiên các bộ dữ liệu thành các phân đoạn theo các vị trí địa lý của chúng. Thuật toán phân đoạn sau đó quyết định nút mà phân đoạn dữ liệu nằm trong đó.

Như được lưu ý ở trên, các bộ dữ liệu được phân chia thành các phân đoạn theo các vị trí địa lý của chúng. Theo phương án hiện tại, điều này đạt được bằng cách phân chia mặt phẳng WSG (World Geodetic System - hệ thống trắc địa thế giới) hai chiều thành các ô lưới (được gọi ở đây là các phân đoạn hoặc các phân đoạn địa lý).

Các giá trị vĩ độ và kinh độ lần lượt nằm trong khoảng từ -90 đến +90, và từ -180 đến +180. Để đơn giản hóa vấn đề, kích thước lưới được xác định là l x l, và do đó có tổng cộng $\left\lceil \frac{180}{l} \right\rceil * \left\lceil \frac{360}{l} \right\rceil$ ô lưới. Hàm lập chỉ mục thẳng *index (lat; lon)* được sử dụng để tính toán id lưới (nghĩa là, id phân đoạn) của vị trí đã cho bất kỳ (lat; lon):

$$shard\_id \quad = index\ (lat, lon) = \left(\left\lfloor \frac{lon + 180}{l} \right\rfloor, \left\lfloor \frac{lat + 90}{l} \right\rfloor\right)$$

Trong đó (-180, -90) là điểm gốc và phân đoạn là $\left\lfloor \frac{lon+180}{l} \right\rfloor th$ ô ở bên phải điểm gốc và $\left\lfloor \frac{lat+90}{l} \right\rfloor th$ ở trên cùng điểm gốc.

Để tăng tốc tìm kiếm K lân cận gần nhất, phương án duy trì phân cấp chỉ mục hai cấp. Bằng cách làm giảm kích thước lưới l, các phân đoạn địa lý được phân chia hơn nữa thành các ô nhỏ hơn (sau đây được gọi là các ô). Để đơn giản hóa vấn đề, theo một phương án kích thước ô được chọn sao cho mỗi ô thuộc về đúng một phân đoạn. Mỗi phân đoạn địa lý chứa tập hợp các ô. Lưu ý rằng kích thước vật lý của các phân đoạn có thể khác nhau; các phân đoạn gần đường xích đạo sẽ lớn hơn về mặt vật lý so với các phân đoạn gần các cực. Tuy nhiên, có thể giả sử rằng các phân đoạn ở gần có các kích thước vật lý tương tự, đặc biệt là trong đó quan tâm tập trung vào các đối tượng nằm trong bán kính nhỏ (< 10km). Theo một phương án, phân đoạn địa lý biểu diễn xấp xỉ 20km x 20km vuông ở đường xích đạo, trong khi ô biểu diễn xấp xỉ diện tích là 500 mét x 500 mét.

Phân đoạn địa lý là đơn vị phân đoạn nhỏ nhất. Như được lưu ý ở trên, dữ liệu thuộc về cùng phân đoạn địa lý được lưu trữ trong cùng bộ nhớ của nút. Phương án phân bố một hoặc nhiều phân đoạn địa lý cho nút dựa trên hàm phân đoạn, nghĩa là

node_id = sharding(index(lat; lon)).

Chi tiết của thuật toán phân đoạn được mô tả sau trong bản mô tả này. Tương tự, thuật toán phân đoạn sẽ ánh xạ ô đến id nút trong đó ô được lưu trữ.

node_id = sharding(cell_id)

Cho cơ sở dữ liệu có các nút, mỗi nút lưu trữ dữ liệu về các người cung cấp dịch vụ trong các phân đoạn tương ứng, tác vụ sau đó là tìm K lân cận gần nhất đến vị trí cụ thể bất kỳ, ví dụ vị trí trong đó khách hàng yêu cầu dịch vụ cần được cung cấp, ví dụ vị trí đón.

Tìm kiếm K lân cận gần nhất thô

Dựa vào Fig.4 (thuật toán 1), cho một vị trí, một phương án truy xuất K đối tượng gần nhất nhờ sử dụng tìm kiếm theo chiều rộng (breadth-first search, BFS).

Để bắt đầu, ô mà vị trí truy vấn thuộc về được nhận dạng (đường 1), nghĩa là, chấm trung tâm 320 trên Fig.3. Sau đó thuật toán tìm kiếm thực hiện tìm kiếm theo chiều rộng

đối với các ô liền kề (đường 11). Các số trên Fig.3 nêu số lần lặp. Khi ghé thăm ô, K đối tượng gần nhất nằm trong ô được trích xuất bởi thuật toán, nghĩa là, hàm *KNearest_InCell* (đường 9). Hàng đợi ưu tiên đối tượng toàn cầu có kích thước K (*kết quả* trong thuật toán) được duy trì dựa trên khoảng cách giữa đối tượng và vị trí tìm kiếm đã cho. Đường 10 so sánh K đối tượng gần nhất trong ô để hợp nhất với kết quả cuối cùng.

Lưu ý rằng đối tượng được tìm thấy trong lần lặp i + 1 (ví dụ chấm 323 trên Fig.3) có thể gần với các đối tượng được tìm thấy trong lần lặp trước i (ví dụ, chấm 325).

Cho ô truy vấn trong đó vị trí truy vấn nằm trong, khoảng cách giữa vị trí bất kỳ trong ô và các vị trí trong các ô được tìm thấy trong lần lặp *i* nằm trong khoảng từ *(i-1)*x*l* đến

$\sqrt{2}$x*(i+1)*x*l* trong đó *l* là chiều dài của ô.

Theo phương án này, khoảng cách Euclidean thay vì khoảng cách haversine được sử dụng, mà không mất tính tổng quát. Do đó, BFS kết thúc ở đầu của lần lặp *i* nếu K đối tượng gần nhất trong *kết quả* được tìm thấy nằm trong lần lặp *min_iter* trong đó

$$i=\left\lceil \sqrt{2} * (\text{min\_}iter + 1) + 1 \right\rceil$$

(đường 13). Lưu ý rằng *min_iter* được duy trì bởi hàm hợp nhất (đường 10).

Vấn đề của tìm kiếm K gần nhất thô là *KNearest_InCell* (đường 9) là lệnh gọi từ xa nếu *sharding(cell)* không phải là cục bộ. Trong trường hợp tệ nhất, sẽ có *O(n)* lệnh gọi từ xa trong đó *n* là số lượng ô được ghé thăm. Lưu ý rằng các ô thuộc về cùng phân đoạn được lưu trữ trong cùng nút, mà còn dẫn đến nhiều lệnh gọi đến cùng nút.

Tiếp theo, thuật toán tìm kiếm K gần nhất được tối ưu hóa (Fig.5, thuật toán 2) để giải quyết vấn đề được mô tả.

Nhớ rằng các ô nằm trong phân đoạn được lưu trữ cùng nhau. Ở đây thuật toán tổng hợp các lệnh gọi*KNearest_InCell* (*K, loc, ô*) từ xa với nhau nếu chúng nằm trong cùng phân đoạn để làm giảm số lượng lệnh gọi từ xa đến *O(m)* trong đó *m* là số lượng phân đoạn được ghé thăm. Trong thực tế, dịch vụ chỉ quan tâm đến các đối tượng gần nhất nằm trong bán kính *r* trong đó *r<< kích thước phân đoạn.* Do đó, số lượng *m* phân đoạn được ghé thăm gần như không đổi. Số lượng lệnh gọi từ xa do đó được làm giảm đến O(1). Thực tế, cho bán kính *r*, tổng số lượng lần lặp cần thiết trong thuật toán 1 có thể được tính toán

trước để thoát khỏi vòng lặp sớm. Hơn nữa, cũng có thể được xác minh xem ô có giao với với bán kính tròn *r* trước khi ghé thăm ô.

Thuật toán 2 thể hiện tìm kiếm K lân cận gần nhất được tối ưu hóa. Đầu tiên, thuật toán nhận dạng các phân đoạn cắt ở gần (đường 1), chi tiết về các phân đoạn này được lược bỏ. Naïve_BFS (*K, loc*) sau đó chạy trên nút cục bộ trong đó mỗi phân đoạn được lưu trữ (đường 3). Thuật toán sau đó hợp nhất các kết quả từ tất cả các phân đoạn (đường 4). Vì các phân đoạn độc lập nhau, các lệnh gọi từ xa được gửi song song. Các ô cũng độc lập, nằm trong Naïve_BFS (*K, loc*), để *KNearest_InCell* (*K, loc, ô*) cũng được chạy song song.

Khi đối tượng di chuyển, một phương án cập nhật vị trí của đối tượng. Sáng chế lưu trữ tất cả các bộ dữ liệu trong bộ nhớ để cập nhật nhanh. Nhớ rằng *index(loc)* nhận dạng duy nhất ô mà vị trí mới thuộc về. Nếu đối tượng đã tồn tại trong ô, vị trí của nó được cập nhật đơn giản. Ngược lại, bộ dữ liệu mới được đưa vào ô. Phương án hiện tại không hủy kích hoạt vị trí cũ của bộ ngay lập tức. Các bộ dữ liệu có TTL (Time To Live - thời gian sống). Khi đọc từ hoặc ghi vào phân đoạn, các bộ trong phân đoạn có các TTL đã hết hạn được hủy kích hoạt. Do đó có thể xảy ra là truy vấn K gần nhất không trả về vị trí mới nhất của người cung cấp dịch vụ. Tuy nhiên, tính hợp thời của các bộ được lưu giữ bởi các dấu thời gian. Phương án này nới lỏng định nghĩa của các truy vấn K gần nhất để trả về tối đa *k* bộ dữ liệu mà là gần nhất với vị trí truy vấn nằm trong khoảng thời gian. Điều này là đủ trong các ứng dụng trong đời thực.

Phương án này còn giải phóng các phân đoạn dữ liệu vô dụng một cách định kỳ. Các phân đoạn dữ liệu được tạo ra vào ban ngày khi hầu hết những người lái xe hoạt động được giải phóng vào ban tối khi những người lái xe nghỉ làm.

Về mặt hình thức, phân đoạn dữ liệu được giải phóng khỏi bộ nhớ nếu tất cả các vị trí của các người lái xe trong phân đoạn là lỗi thời (ví dụ, 10 phút trước). Trong thực tế, các phân đoạn dữ liệu được dọn dẹp mỗi 15 phút.

Có thể giả định rằng chỉ mục không gian địa lý sẽ phục vụ mục đích phân chia nếu các điều kiện sau được thỏa mãn:

- Phân chia Trái Đất thành các phần nhỏ;
- Ánh xạ duy nhất các tọa độ địa lý thành phần nhỏ (còn gọi là phân đoạn); và

- Truy xuất các phần nhỏ lân cận hữu hiệu.

Các chỉ mục không gian địa lý được phát triển gần đây chẳng hạn như S2 bởi Google, và H3 bởi Uber thực sự có tiềm năng đẩy nhanh giai đoạn truy vấn. Ví dụ, hình lục giác trong H3 có ít lân cận hơn so với hình vuông, mà làm giảm không gian tìm kiếm. Tuy nhiên, các chỉ mục đơn giản của phương án hiện tại có thể được tính toán nhanh hơn nhiều (Fig.9). Tính toán chỉ mục nhanh tăng tốc cả hai hoạt động ghi và đọc. Tuy nhiên, phương án hiện tại có thể là môđun và các chỉ mục nêu trên có thể được nạp vào hệ thống nếu chúng được cần đến.

Phần mô tả sau về cách mà phương án hiện tại quản lý các nút theo thiết lập phân tán để đạt được độ trễ thấp, độ tin cậy và khả dụng cao. Đề xuất thứ nhất là ShardTable dưới dạng bổ sung để băm nhất quán để phân bố các phân đoạn dữ liệu cho các nút để đạt được cân bằng tải. Giao thức tin đồn nổi tiếng SWIM sau đó được sử dụng để phát hiện nút và phát hiện lỗi. Cuối cùng, được thể hiện là cách mà một phương án khôi phục từ các lỗi khu vực nhanh chóng.

Thuật toán phân đoạn

Phần này mô tả cách một phương án phân bố các phân đoạn dữ liệu cho các nút khác nhau.

Phép băm nhất quán được sử dụng rộng rãi để phân bố số lượng phân đoạn dữ liệu bằng nhau cho các nút khác nhau, với lợi ích là lượng dữ liệu tối thiểu cần được di chuyển khi các nút mới được thêm vào. Tuy nhiên, cách này dẫn đến các vấn đề hiệu suất lớn trong thực tế vì các kích thước phân đoạn và các nhu cầu truy vấn không cân bằng. Các phân đoạn nhất định chứa nhiều đối tượng hơn các phân đoạn khác. Ví dụ, phân đoạn trong thành phố lớn hơn (ví dụ, Singapore) có số lượng người lái xe lớn hơn năm lần so với thành phố nhỏ hơn (ví dụ, Bali). Thứ hai là, các phân đoạn trong các vùng nhu cầu cao (ví dụ, vùng trung tâm thành phố) được truy vấn thường xuyên hơn nhiều so với các vùng nông thôn. Khi phân bố các phân đoạn đồng đều cho các nút, thấy được rằng các nút nhất định trở thành các điểm nóng với sử dụng CPU vượt quá 80% trong khi một số nút khác là nhàn rỗi.

Hơn nữa, việc thêm các máy mới dưới băm nhất quán có thể khiến mọi thứ tồi tệ hơn. Ví dụ, trong các dịch vụ mạng của Amazon (Amazon Web Services, AWS), mở rộng quy mô thường được kích hoạt bởi sử dụng CPU cao của nút, nghĩa là, điểm nóng. Khi nút

mới được thêm vào, băm nhất quán chọn ngẫu nhiên một hoặc một vài nút và chuyển các phân đoạn dữ liệu của chúng (và do đó tải truy vấn) cho nút mới. Không may là, nút điểm nóng không được đảm bảo là sẽ được chọn, mà dẫn đến việc bổ sung các nút nhàn rỗi mới trong khi điểm nóng hầu như không được giảm thiểu.

Do đó, một phương án, cân bằng giữa thời gian di chuyển dữ liệu và thời gian truy vấn nhanh bằng cách sử dụng *ShardTable*. ShardTable là ánh xạ tạo cấu hình được bởi người dùng từ các phân đoạn đến các nút mà xác định rõ ràng phân đoạn nào thuộc về nút nào. Theo một phương án nút được dành riêng cho mỗi vùng có nhu cầu cao trong thành phố. Trong một số trường hợp, nút có thể phục vụ nhiều thành phố nhỏ. Đối với các phân đoạn mà không ở trong bảng phân đoạn, thì sử dụng băm nhất quán.

ShardTable là bán tự động. Khi nút điểm nóng được quan sát, phương án hiện tại tính toán các phân đoạn mà cần được di chuyển dựa trên tải đọc/ghi đối với các phân đoạn. Quản trị viên sau đó di chuyển các phân đoạn đến nút nhàn rỗi hiện có hoặc nút mới.

Cấu trúc bán tự động hoạt động tốt cho người yêu cầu. Sự can thiệp của con người hiếm khi được cần đến khi ShardTable được tạo cấu hình đúng ngay từ đầu.

Phát hiện nút và khôi phục lỗi

Một phương án áp dụng truyền thông điệp kiểu tin đồn để phát hiện nút. Mỗi nút tán gẫu xung quanh về sự hiểu biết của nó trên cấu trúc mạng. Cụ thể là, Serf được chọn vì nó thực hiện SWIM với sự tăng cường Lifeguard. Một vấn đề với SWIM là khi nút mới kết nối, thì bộ điều phối tĩnh được cần đến để xử lý yêu cầu kết nối để tránh được nhiều thành viên trả lời.

Một phương án sử dụng lại một cách tinh vi một nút trong ShardTable làm bộ điều phối tĩnh để phát rộng các kết nối mới. Điều đáng nói là SWIM cung cấp tính đầy đủ giới hạn thời gian, nghĩa là, thời gian phát hiện trường hợp xấu nhất về lỗi của thành viên bất kỳ được liên kết. Để đạt được điều này, SWIM áp dụng lựa chọn mục tiêu thăm dò vòng tròn (*Round-Robin Probe Target Selection*). Mỗi nút duy trì danh sách thành viên hiện tại, và chọn các đích ping theo cách theo thứ tự vòng tròn thay vì ngẫu nhiên. Các nút mới được đưa vào danh sách ở các vị trí ngẫu nhiên thay vì được nối vào đầu để tránh được bị bỏ ưu tiên. Thứ tự của danh sách được xáo trộn mọi thứ sau khi một lần quét được hoàn thành. Ngoài ra, SWIM làm giảm các khẳng định sai của các lỗi bằng cách cho phép các thành viên nghi ngờ nút trước khi khai báo nút này bị lỗi.

Lưu ý rằng việc sử dụng các dịch vụ phát hiện nút của bên thứ ba được cố gắng tránh, để giảm thiểu sự phụ thuộc dịch vụ càng nhiều càng tốt.

Một phương án lưu nhanh dữ liệu một cách định kỳ để phát hiện lỗi. Bản lưu nhanh được lưu trữ trong Redis kho dữ liệu giá trị khóa bên ngoài. Trong trường hợp mất điện trong đó tất cả chu kỳ nguồn của các nút và do đó tất cả trong bộ nhớ dữ liệu bị mất, một phương án có thể bắt đầu lại bằng cách quét dữ liệu bản lưu nhanh trong Redis. Các thí nghiệm thể hiện rằng một phương án có thể khôi phục từ các lỗi trong một phút.

Bộ bản sao và chuyển tiếp truy vấn

Độ tin cậy và độ bền cao cần có bản sao dữ liệu. Một phương án áp dụng các bộ bản sao để sao chép dữ liệu. Mỗi phân đoạn dữ liệu được sao chép đến nhiều nút trong đó mỗi nút được xử lý như nhau. Các hoạt động ghi trên phân đoạn được truyền đến tất cả các nút bản sao. Tùy thuộc vào cấu hình nhất quán, giao thức bỏ phiếu dựa trên túc số có thể hoặc có thể không được áp dụng. Nếu tính khả dụng được ưu tiên hơn tính nhất quán, và vì tính hợp thời của dữ liệu vị trí, tính nhất quán có thể được nới lỏng. Số lượng bản sao có thể tạo cấu hình dựa trên các trường hợp sử dụng.

Một phương án ưu tiên các bộ bản sao hơn thiết kế chủ-tớ. Việc duy trì thành viên chủ hoặc bầu cử lại chủ phát sinh thêm chi phí. Ngược lại, bộ bản sao là linh hoạt hơn. Nó đổi tính nhất quán để lấy tính khả dụng. Đối với các phân đoạn được phân bố cho các nút bằng cách băm nhất quán, phương án thực hiện cổ điển được sử dụng, cụ thể là lưu trữ các bản sao của nó trên các nút tiếp theo trong vòng. Đối với phân đoạn trong ShardTable, ánh xạ duy trì trong đó các bản sao của phân đoạn được lưu trữ.

Khi trả lời các truy vấn K lân cận gần nhất, phương án này coi mỗi nút bản sao tương đương nhau. Khi nút nhận yêu cầu tìm kiếm K lân cận gần nhất đối với vị trí, nó gọi ra thuật toán 2.

Đối với các lệnh gọi từ xa (đường 3 trong thuật toán 2), vì có các bản sao đối với mỗi phân đoạn, có hai chiến thuật để cân bằng các truy vấn trên các bản sao, phân nhánh ra hoặc theo thứ tự vòng tròn. Trong thiết lập phân nhánh ra, nút gửi các lệnh gọi từ xa đến các bản sao song song và lấy kết quả được trả về nhanh nhất. Trong thiết lập theo thứ tự vòng tròn, mỗi bản sao lần lượt thực hiện các lệnh gọi từ xa.

Các truy vấn K gần nhất

Trong phần này, sẽ so sánh hiệu suất của truy vấn K lân cận gần nhất thuật toán 1 và thuật toán 2, nhờ sử dụng các truy vấn K lân cận gần nhất trong đời thực của người yêu cầu. Người đăng ký hỗ trợ gần 6 triệu lượt đi hàng ngày, đạt một tỷ truy vấn K lân cận gần nhất mỗi ngày. Nhớ rằng độ phức tạp thời gian của thuật toán 1 bị chi phối bởi số lượng lệnh gọi từ xa, mà là tuyến tính theo số lượng ô được ghé thăm. Thuật toán 2 tuyến tính theo số lượng phân đoạn được ghé thăm. Do đó, số lượng ô được ghé thăm trung bình trong phân đoạn được ghé thăm được sử dụng để thể hiện sự cải thiện của thuật toán 2 so với thuật toán 1.

Fig.6 thể hiện số lượng ô được ghé thăm trung bình trong phân đoạn được ghé thăm. Lưu ý rằng khi thời gian thay đổi (trục x), thì số lượng ô được ghé thăm trung bình trong phân đoạn được ghé thăm hơi thay đổi. Trung bình, việc ghé thăm phân đoạn quét 27:3 ô, với trường hợp xấu nhất là 120 ô. Do đó, thuật toán 2 nhanh hơn 27:3 lần so với thuật toán 1 về trung bình. Ngoài ra, số lượng phân đoạn trung bình được ghé thăm trong thuật toán 2 là 1:27, mà xác nhận độ phức tạp thời gian không đổi.

Cân bằng tải

Trong phần này, băm nhất quán được so sánh với ShardTable về hiệu suất cân bằng tải của chúng. Các thí nghiệm được chạy trên 10 nút. Trong một thiết lập, phép băm nhất quán được sử dụng để phân bố phân đoạn, trong khi theo một phương án khác được sử dụng với cả lập chỉ mục ShardTable và băm nhất quán. Cả ghi và các tải truy vấn K gần nhất được so sánh theo môi trường đời thực. Một số mức độ chi tiết không được thể hiện để chỉ thể hiện các biện pháp thứ cấp vì lý do thương mại.

Fig.7a thể hiện ghi và phân bố tải truy vấn trên 10 nút theo băm nhất quán. Nhớ rằng mặc dù các phân đoạn bằng nhau trong thế giới thực, các quốc gia nhất định có nhiều người lái xe hơn trong một phân đoạn so với các quốc gia khác và các hoạt động ghi tuyến tính theo số lượng người lái xe. Như Fig.7a thể hiện, nút cực hạn nhất chứa 32:9% của toàn bộ người lái xe trong khi nút khác chiếm ít nhất 0:37% người lái xe. Phương sai mẫu tối đa cao 103. Tương tự, tải truy vấn K lân cận gần nhất cũng không cân bằng nằm trong khoảng từ 0:72% đến 39:84%.

Fig.7b thể hiện phân bố tải truy vấn ghi trong số 10 nút nhờ sử dụng phương án. Rõ ràng là tải ghi rất cân bằng, nằm trong khoảng từ 8:71% đến 13:92%. Phương sai mẫu thấp nhất là 3:64. Điều đáng chú ý là hiện tại

một phương án ưu tiên cân bằng tải ghi hơn là tải truy vấn K lân cận gần nhất. Fig.7c thể hiện sự phân bố tải truy vấn của phương án này. Có thể thấy rằng với tải ghi cân bằng, nghĩa là, mỗi nút chứa gần như cùng số lượng người lái xe, tải truy vấn vẫn thay đổi, từ 1:93% đến 35:49%. Tuy nhiên, nó vẫn tốt hơn băm nhất quán.

Khôi phục lỗi

Trong phần con này, hiệu suất của một phương án được đánh giá đối với khôi phục lỗi. Thí nghiệm được chạy trên Mac Pro với lõi Intel i7 2,7 GHz và bộ nhớ 16 GB. Fig.8 thể hiện các kết quả.

Thời gian khôi phục được đánh giá khi số lượng người lái xe tăng lên. Như được thể hiện trên Fig.8 (lưu ý rằng thang đo logarit của số lượng người lái xe), khi số lượng người lái xe tăng từ 1K đến 5 triệu, thời gian khôi phục tăng lên một cách tuyến tính. Phương án có thể khôi phục trong ít hơn 25 giây, thậm chí với 5 triệu người lái xe.

Lưu đồ

Dựa vào Fig.2, lưu đồ thể hiện hai quy trình 430 và 450, mỗi quy trình chạy bên trong các nút, trong khi khối 470 biểu diễn các bộ bản sao. Quy trình lưu nhanh dữ liệu 490 cũng chạy bên trong mỗi nút.

Như được thể hiện, dữ liệu yêu cầu và ghi 401 được nhập vào thiết bị cân bằng tải 411 mà vận hành để phân bố các yêu cầu và ghi trong số các nút khác nhau, nhờ đó đảm bảo tải đồng đều và khả năng xử lý đọc và ghi nhiều. Thiết bị cân bằng tải 411 sắp xếp dữ liệu theo kiểu thành ghi dữ liệu vị trí theo thời gian thực 413, bao gồm các bộ dữ liệu ghi và các yêu cầu truy vấn K gần nhất 415.

Các bộ dữ liệu ghi bao gồm vị trí địa lý của các đối tượng đang được xem xét ví dụ các xe ở trạng thái gọi xe, ID của mỗi đối tượng, thông tin dấu thời gian và siêu dữ liệu, như được mô tả ở các phần khác trong bản mô tả này.

Các yêu cầu K gần nhất bao gồm dữ liệu, ví dụ trong các gói chứa dữ liệu vị trí, dấu thời gian, K và bán kính tìm kiếm.

Dữ liệu vị trí thời gian thực 413 được chuyển đến bộ phận lưu trữ 430, mà thực hiện hai quyết định. Bộ phận quyết định thứ nhất 431 được cấp dữ liệu biểu thị sự phân chia mặt phẳng WSG thành các phân đoạn và gọi từ nguồn dữ liệu địa lý 421 để thực hiện hàm chỉ mục bằng cách này quyết định được đưa ra về phân đoạn và ô mà dữ liệu vị trí thời

gian thực 413 thuộc về.

Sau khi đưa ra quyết định này, dữ liệu thời gian thực được chuyển đến bộ phận quyết định thứ hai 433, mà được cấp dữ liệu cấu hình 423, dữ liệu liên quan đến ShardTable và kích thước bộ bản sao, để quyết định bộ bản sao nút mà phân đoạn được định vị trong.

Dữ liệu kết quả sau đó được sử dụng bởi bộ phận ghi 435 để đưa vị trí của đối tượng (xe, trong ứng dụng gọi xe) vào phân đoạn của nút bộ bản sao, hoặc nếu đã có mặt trong phân đoạn này để cập nhật vị trí đối tượng.

Bộ phận lưu trữ ghi dữ liệu này 481 vào bộ nhớ phân tán 470 để bao gồm phát hiện nút 471 và dữ liệu 473 của các bộ bản sao.

Dữ liệu yêu cầu K gần nhất 415 chuyển đến bộ phận truy vấn 450 mà chạy quy trình thứ nhất 449 để chuyển tiếp yêu cầu đến nút chứa dữ liệu phân đoạn chính, quy trình thứ hai, 451 để chuyển tiếp truy vấn trong các bộ bản sao và sau đó quy trình thứ ba 453, chạy thuật toán truy vấn K gần nhất phân tán. Kết quả được kết xuất dưới dạng dữ liệu đọc 487 đến bộ nhớ phân tán 470, và theo phương án này kết quả của thuật toán tìm kiếm còn được trả (không được thể hiện trên biểu đồ) về người gọi mà đã khởi tạo truy vấn ngay từ đầu. Kết quả tìm kiếm là ID của K người lái xe gần nhất, cộng với dữ liệu vị trí của họ và dấu thời gian.

Bộ nhớ phân tán 470 còn ghi dữ liệu bản lưu nhanh 490, qua quy trình ghi 483, và có thể sử dụng để phát hiện lỗi 485.

Kiến trúc

Dựa vào Fig.10, sơ đồ khối giản lược của phương án được đơn giản hóa của hệ thống cơ sở dữ liệu trong bộ nhớ bao gồm 3 nút lưu trữ, A, B và C, cùng với bộ phận cân bằng tải 411 được mô tả ở trên theo Fig.2.

Mỗi nút lưu trữ A, B và C bao gồm bộ xử lý tương ứng X, Y, Z và bộ nhớ chính, (ví dụ RAM) A1, A2, A3. Khi sử dụng, các bộ xử lý X, Y và Z thực hiện các quy trình 430, 450 như được mô tả theo Fig.2. Thiết bị lưu trữ trong bộ nhớ (nghĩa là RAM) được sử dụng để hỗ trợ các yêu cầu ghi/cập nhật dữ liệu lớn.

Mặc dù trong tương lai lưu trữ từ xa hoặc đám mây có thể được dự kiến hiện không thể xử lý loại tải ghi cần thiết cho ứng dụng gọi xe.

Điều quan trọng theo các phương án để định vị các nút lưu trữ đủ gần với một nhau

là thời gian truyền dữ liệu đối với lượng lớn luồng dữ liệu không trở nên đáng kể.

Các bộ bản sao, mà là các bộ ngang hàng, được lưu trữ trên các nút khác nhau. Một lý do cho điều này đó là nếu một nút lỗi, nút khác có thể vẫn phục vụ. Quy trình băm/lập chỉ mục (băm nhất quán hoặc lập chỉ mục ShardTable) được sử dụng để xác định nhiều nút mà phân đoạn cụ thể được lưu trữ trong đó. Theo phương án, dữ liệu được lưu trữ trong các nút với không có nút nào được cố định làm nhà chính cho dữ liệu đó.

Trong phần mô tả sau và các hình vẽ kèm theo, các sự kết nối được thể hiện dưới dạng đường thẳng duy nhất để dễ giải thích. Cần hiểu rằng điều này khó có thể xảy ra theo phương án thực tế, trong đó các tốc độ dữ liệu cực cao được truyền qua nhiều bus dẫn điện hoặc các liên kết khác.

Như được thể hiện, các mũi tên 713 chỉ về phía bộ phận cân bằng 411 biểu diễn người cung cấp dịch vụ (ví dụ vị trí người lái xe) cập nhật thông tin cần được đưa vào hệ thống. Mũi tên 714 biểu diễn các yêu cầu tìm kiếm lân cận gần nhất được đưa vào. Mũi tên 715 chỉ lên trên từ bộ phận 411 biểu diễn các kết quả truy vấn được kết xuất từ cơ sở dữ liệu. Bộ cân bằng tải 411 phân bố các yêu cầu tìm kiếm và dữ liệu của người cung cấp dịch vụ trong số các nút để cân bằng tải đọc và ghi. Các mũi tên 717 từ bộ phận cân bằng tải 411 đến nút A biểu diễn dữ liệu của người cung cấp dịch vụ đang được chuyển đến nút (nút A), trong khi đó các mũi tên 719 biểu diễn các kết quả truy vấn rời khỏi nút. 707 là dữ liệu từ bộ phận 411 đến nút C; 709 là các kết quả truy vấn từ nút C.

Ở nút A, mũi tên 723 biểu diễn luồng dữ liệu người lái xe đến vị trí lưu trữ A1 từ bộ xử lý X và từ vị trí lưu trữ A1 đến bộ xử lý X. Vị trí lưu trữ A3 biểu diễn bộ bản sao của phân đoạn của dữ liệu được lưu trữ ở vị trí B2. Nút B là nút máy chủ cho phân đoạn của dữ liệu được lưu trữ ở vị trí B2.

Mũi tên 725 biểu diễn truy cập đọc và ghi đến vị trí lưu trữ A3, mà sẽ được nhắc lại là lưu trữ bộ bản sao của dữ liệu được lưu trữ ở vị trí B2. Như được lưu ý ở trên, theo một phương án, các bộ bản sao được lưu trữ trên các nút khác nhau, sao cho nếu một nút có vấn đề thì sẽ vẫn có dịch vụ nhờ sử dụng nút khác hoặc các nút khác.

Mũi tên 727 biểu diễn truyền dữ liệu giữa các bộ xử lý X và Y của các nút A và B và mũi tên 729 biểu diễn truyền dữ liệu đến và từ vị trí B2. Mũi tên 731 biểu diễn luồng dữ liệu giữa các bộ xử lý Y và Z.

Là một ví dụ đơn giản hóa của hoạt động, giả sử tìm kiếm được yêu cầu ở bộ cân bằng tải 411 để tìm kiếm dữ liệu được lưu trữ ở vị trí B2, và yêu cầu tìm kiếm này được chuyển bởi bộ cân bằng tải 411 qua đường dẫn 707 đến nút C. Sau khi nút C nhận yêu cầu, nó sẽ sử dụng băm nhất quán hoặc chỉ mục ShardTable để "biết" để chuyển tiếp yêu cầu đến nút "máy chủ", nút B, qua kết nối 731. trong đó điểm truy vấn được lưu trữ. Bộ xử lý của nút máy chủ nút B sau đó sẽ chạy truy vấn trên đường dẫn 729. Khi cập nhật dữ liệu được lưu trữ ở vị trí B2, bộ xử lý Y chuyển tiếp dữ liệu cập nhật qua kết nối 727 sao cho bộ bản sao ở vị trí A3 cũng được cập nhật.

Phần trên biểu diễn phần mô tả rất được đơn giản hóa về hệ thống không thực tế. Trong thực tế sẽ có các bộ bản sao nằm trên nhiều nút. Liên kết đơn giản của nút A đến nút B đến nút C theo nhiều phương án sẽ được thay thế bởi mạng gồm các liên kết.

Khi sử dụng, nếu truy vấn tìm kiếm được chạy ở chế độ phân nhánh ra, thì sau đó cả nút A và nút B sẽ thực thi truy vấn đối với dữ liệu và trả về. Trong trường hợp này, cả A và B là các nút máy chủ. Nếu thiết lập là theo thứ tự vòng tròn, thì ví dụ nút A và nút B lần lượt là nút máy chủ để thực thi các truy vấn.

Các ưu điểm của các phương án

Các phương án tạo sự hỗ trợ cho việc ghi thường xuyên bởi các khóa. Các hoạt động ghi được cần đến để cập nhật và theo dõi tất cả các vị trí hiện thời của các đối tượng. Người lái xe có thể di chuyển 25 mét mỗi giây ở các quốc gia phát triển như Singapore. Do đó điều quan trọng là cập nhật các vị trí của các người lái xe mỗi giây, nếu không phải là mili giây. Do đó, các cơ sở dữ liệu liên quan đến truyền thống hoặc các cơ sở dữ liệu không gian địa lý mà tạo ra các I/O đĩa cho các hoạt động ghi là quá đắt để sử dụng. Các phương án lưu trữ dữ liệu trong bộ nhớ trong môi trường phân tán.

Mặc dù tất cả các đối tượng có thể vừa trong một bộ nhớ của máy, máy duy nhất sớm trở nên bị choáng ngợp bởi số lượng ghi cỡ lớn và các truy vấn kNN, nhớ rằng số lượng người lái xe có các vị trí thời gian thực được báo cáo. Để giải quyết vấn đề này, các phương án phân bố các đối tượng (ví dụ, những người lái xe) cho các nút khác nhau (nghĩa là, các máy) theo các vị trí địa lý của chúng.

Hỗ trợ cho các tìm kiếm kNN bởi các vị trí địa lý. Lưu trữ dữ liệu khóa-giá trị đã biết chẳng hạn như Dynamo và Memcache lưu trữ các đối tượng dưới dạng các khóa và các vị trí của chúng dưới dạng các giá trị. Tìm kiếm kNN sau đó yêu cầu quét tất cả các

khóa và tính toán các khoảng cách theo cặp, độ trễ của nó là không thể chấp nhận. Các thuật toán tìm kiếm kNN truyền thống dựa vào các chỉ mục chẳng hạn như các cây R để tăng tốc các truy vấn. Tuy nhiên, việc duy trì các chỉ mục phức tạp này trong khi xử lý ghi thường xuyên là không khả thi. Các phương án áp dụng thuật toán tìm kiếm theo chiều rộng để trả lời các truy vấn K lân cận gần nhất. Bằng cách phân chia các phân đoạn hơn nữa thành các ô nhỏ, một phương án tránh được việc quét toàn bộ phân đoạn. Bắt đầu với ô mà điểm truy vấn nằm trong đó, và dần dần tìm kiếm các ô lân cận. Để làm giảm các lệnh gọi từ xa, các phương án tổng hợp các lệnh gọi ở mức phân đoạn, mà cũng đạt được tính song song.

Hỗ trợ các tải không cân bằng. Vì các phân đoạn địa lý có kích thước vật lý cố định (ví dụ, 20km x 20km), không bất ngờ là các phân đoạn nhất định có nhiều dữ liệu và các truy vấn hơn so với các phân đoạn khác. Ví dụ, phân đoạn trong thành phố lớn hơn (ví dụ, Singapore) có thể có số lượng người lái xe lớn hơn năm lần so với trong thành phố nhỏ hơn (ví dụ, Bali). Kết quả là, phân đoạn trong thành phố lớn hơn có số lần ghi lớn hơn năm lần so với phân đoạn trong thành phố nhỏ hơn. Các phân đoạn ở các vùng nhu cầu cao (ví dụ, vùng trung tâm thành phố) được truy vấn thường xuyên hơn nhiều so với các vùng nông thôn. Các tải không cân bằng này dẫn đến các sự khó khăn đối với các chính sách mở rộng quy mô. Phép băm nhất quán được sử dụng rộng rãi để mở rộng quy mô vì nó giảm thiểu lượng dữ liệu cần được di chuyển qua các nút. Tuy nhiên, khi một nút trở thành điểm nóng và nút mới được thêm vào, thì băm nhất quán sẽ chọn nút ngẫu nhiên và chuyển một phần của dữ liệu của nó sang nút mới. Nếu không may thì, nút điểm nóng không được chọn, trạng thái của nó sẽ hầu như không được giảm. Trạng thái này rất có thể kết thúc với vòng lặp cụt của việc thêm các nút nhàn rỗi.

Một phương án đề xuất ShardTable dưới dạng bổ sung để, và để sử dụng cùng với, băm nhất quán để cân bằng tải. Mặc dù băm nhất quán phân bố xấp xỉ số lượng phân đoạn bằng nhau cho các nút, ShardTable được tạo cấu hình để dành một hoặc nhiều nút cho một phân đoạn cụ thể. ShardTable là cấu trúc bán tự động, mặc dù, trong thực tế, các sự can thiệp của con người hiếm khi được cần đến.

Độ tin cậy, phát hiện và khôi phục lỗi nhanh. Các phương án sử dụng các bộ bản sao để hi sinh tính nhất quán mạnh mẽ đổi lấy tính khả dụng cao. Đồng thời các bản sao khác nhau có thể xem các trạng thái dữ liệu khác nhau, mà không phải là quan trọng trong trường hợp sử dụng của sáng chế. Các bộ bản sao khiến cho toàn bộ hệ thống khả dụng

cao. Các phương án tận dụng giao thức kiểu tin đồn SWIM để đạt được phát hiện lỗi nhanh. Trong trường hợp mất điện theo vùng, các phương án có thể khôi phục nhanh chóng từ kho dữ liệu bên ngoài trong đó các bản lưu nhanh dữ liệu được giữ không đồng bộ.

Rõ ràng là sáng chế được mô tả chỉ bằng cách lấy ví dụ. Các sự cải biến khác nhau có thể được thực hiện đối với các kỹ thuật được mô tả ở đây mà không lệch khỏi tinh thần và phạm vi của các điểm yêu cầu bảo hộ kèm theo. Các kỹ thuật được mô tả bao gồm các kỹ thuật mà có thể được đưa ra theo cách độc lập hoặc kết hợp với nhau. Do đó, các dấu hiệu được mô tả đối với một kỹ thuật có thể còn được thể hiện kết hợp với kỹ thuật khác.

YÊU CẦU BẢO HỘ

1. Hệ thống cơ sở dữ liệu được tạo cấu hình để tìm kiếm các đối tượng di động, mỗi đối tượng này có các thuộc tính bao gồm dữ liệu vị trí, để xác định các đối tượng lân cận gần nhất với vị trí cụ thể, đối tượng này nằm trong không gian địa lý được tạo thành từ các không gian con khác biệt về không gian, mỗi không gian con này được tạo thành từ các ô, hệ thống cơ sở dữ liệu bao gồm các nút lưu trữ; và hệ điều hành, hệ điều hành này được tạo cấu hình để điều khiển việc lưu trữ dữ liệu đối tượng trong số các nút lưu trữ, trong đó hệ điều hành được tạo cấu hình để thực hiện việc lưu trữ dữ liệu biểu diễn một hoặc nhiều không gian con khác biệt về không gian trong một nút lưu trữ tương ứng trong số các nút lưu trữ, trong đó dữ liệu vị trí của mỗi đối tượng được sử dụng để lập chỉ mục rằng đối tượng đối với các ô tạo thành mỗi không gian con khác biệt về không gian trong mỗi nút, và trong đó dữ liệu được lưu trữ trong các nút lưu trữ nhờ sử dụng phép ánh xạ tạo cấu hình được từ các không gian con đến các nút lưu trữ mà xác định rõ ràng không gian con nào thuộc về nút lưu trữ nào, dựa trên tải đọc và/hoặc ghi trên mỗi trong số các không gian con khác biệt về không gian.

2. Hệ thống cơ sở dữ liệu theo điểm 1, trong đó dữ liệu của mỗi không gian con khác biệt về không gian được lưu trữ hoàn toàn trong một nút lưu trữ duy nhất.

3. Hệ thống cơ sở dữ liệu theo điểm 1 hoặc điểm 2, trong đó hệ điều hành được tạo cấu hình sao cho dữ liệu của mỗi không gian con khác biệt về không gian được sao chép đến các nút lưu trữ để tạo ra các bản sao dữ liệu.

4. Hệ thống cơ sở dữ liệu theo điểm 3, trong đó hệ điều hành được tạo cấu hình sao cho các hoạt động ghi liên quan đến không gian con khác biệt về không gian được truyền đến tất cả các bản sao dữ liệu liên quan.

5. Hệ thống cơ sở dữ liệu theo điểm 3, trong đó số lượng bản sao có thể tạo cấu hình dựa trên các trường hợp sử dụng.

6. Hệ thống cơ sở dữ liệu theo điểm 1 hoặc điểm 2, trong đó hệ điều hành được tạo cấu hình để thao tác thuật toán tìm kiếm theo chiều rộng để trả lời các truy vấn K lân cận gần nhất.

7. Hệ thống cơ sở dữ liệu theo điểm 1, trong đó dữ liệu được lưu trữ trong các nút lưu trữ bằng cách băm nhất quán.

8. Hệ thống cơ sở dữ liệu theo điểm 1, trong đó để cân bằng tải, hệ điều hành được tạo cấu hình để sử dụng cả phép ánh xạ tạo cấu hình được bởi người dùng từ các không gian con đến các nút lưu trữ mà xác định rõ ràng không gian con nào thuộc về nút nào và băm nhất quán.

9. Hệ thống cơ sở dữ liệu theo điểm 8, trong đó đối với dữ liệu không có trong phép ánh xạ, thì phép băm nhất quán được sử dụng.

10. Hệ thống cơ sở dữ liệu theo điểm 1, trong đó một nút trong phép ánh xạ được sử dụng làm bộ điều phối tĩnh để phát rộng các kết nối mới.

11. Hệ thống cơ sở dữ liệu theo điểm 1, trong đó hệ điều hành áp dụng truyền thông điệp kiểu tin đồn để phát hiện nút.

12. Hệ thống cơ sở dữ liệu theo điểm 1 hoặc điểm 2, trong đó các đối tượng là các xe của người cung cấp dịch vụ.

13. Hệ thống cơ sở dữ liệu theo điểm 1 hoặc điểm 2, trong đó cơ sở dữ liệu được lưu trữ trong bộ nhớ.

14. Phương pháp lưu trữ dữ liệu biểu diễn các đối tượng di động, mỗi đối tượng này có các thuộc tính bao gồm dữ liệu vị trí, để cho phép tìm kiếm nhanh các lân cận gần nhất với vị trí cụ thể trong không gian địa lý được tạo thành từ các không gian con khác biệt về không gian, mỗi không gian con này được tạo thành từ các ô, hệ thống cơ sở dữ liệu bao gồm các nút lưu trữ; phương pháp này bao gồm các bước:

lưu trữ dữ liệu đối tượng trong số các nút lưu trữ, sao cho dữ liệu biểu diễn một hoặc nhiều không gian con khác biệt về không gian được lưu trữ trong một nút lưu trữ tương ứng trong số các nút lưu trữ,

sử dụng dữ liệu vị trí hiện thời của mỗi đối tượng để lập chỉ mục rằng đối tượng đối với các ô tạo thành mỗi không gian con khác biệt về không gian trong mỗi nút lưu trữ, và

sử dụng tải đọc và/hoặc ghi trên mỗi trong số các không gian con khác biệt về không gian để ánh xạ các không gian con đến các nút lưu trữ để xác định rõ ràng không gian con nào thuộc về nút lưu trữ nào.

15. Phương pháp tăng tốc tìm kiếm lân cận gần nhất bao gồm bước phân bố dữ liệu trong các nút lưu trữ theo mối quan hệ địa lý giữa dữ liệu, sao cho dữ liệu biểu diễn một hoặc nhiều không gian con khác biệt về không gian được lưu trữ trong một nút lưu trữ tương

ứng trong số các nút lưu trữ và lập chỉ mục dữ liệu đối với vị trí nằm trong các ô tạo thành mỗi không gian con khác biệt về mặt không gian và sử dụng tải đọc và/hoặc ghi trên mỗi trong số các không gian con khác biệt về không gian để ánh xạ các không gian con đến các nút lưu trữ để xác định rõ ràng không gian con nào thuộc về nút lưu trữ nào, nhờ đó cho phép việc tìm kiếm dữ liệu được thực hiện nhờ sử dụng số lượng cuộc gọi từ xa được giảm bớt.

16. Hệ thống kho dữ liệu không gian trong bộ nhớ mở rộng được cho các tìm kiếm kNN bao gồm hệ thống cơ sở dữ liệu theo điểm bất kỳ trong số các điểm từ 1 đến 13.

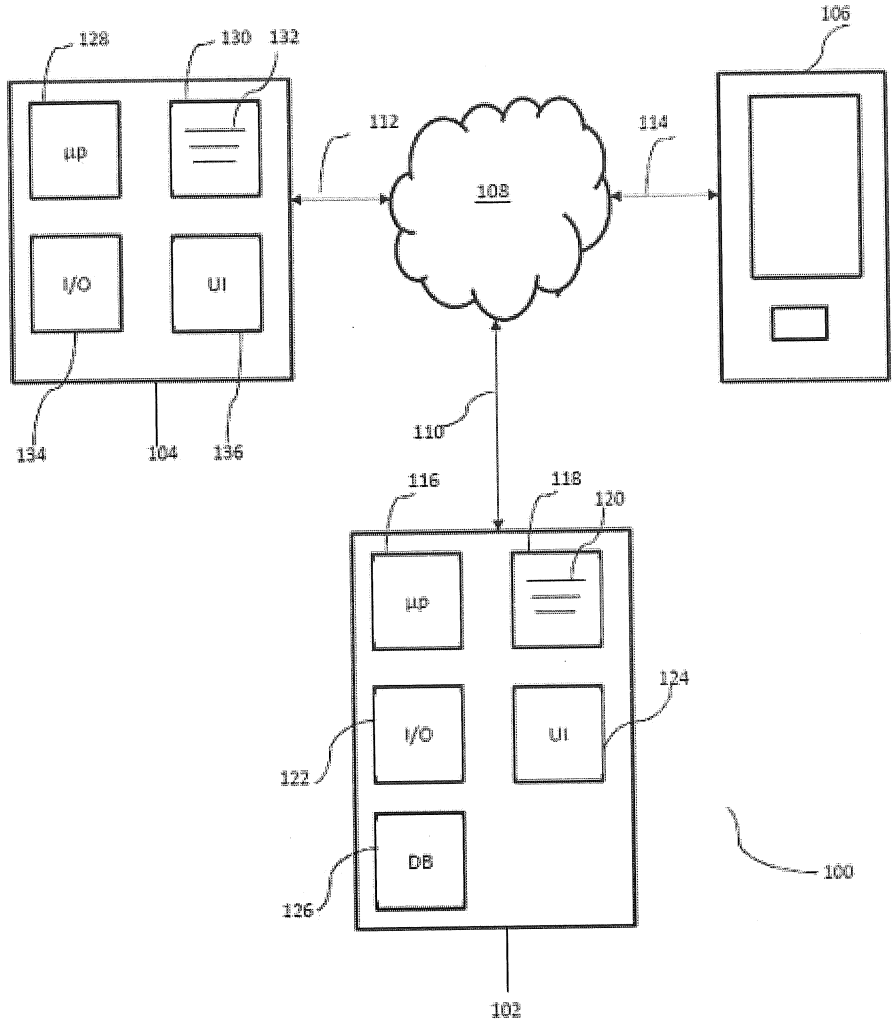

106
128
130
132
112
114
108
µp
I/O
UI
134
104
136
110
116
118
120
µp
124
I/O
UI
122
DB
126
100
102

FIG. 1

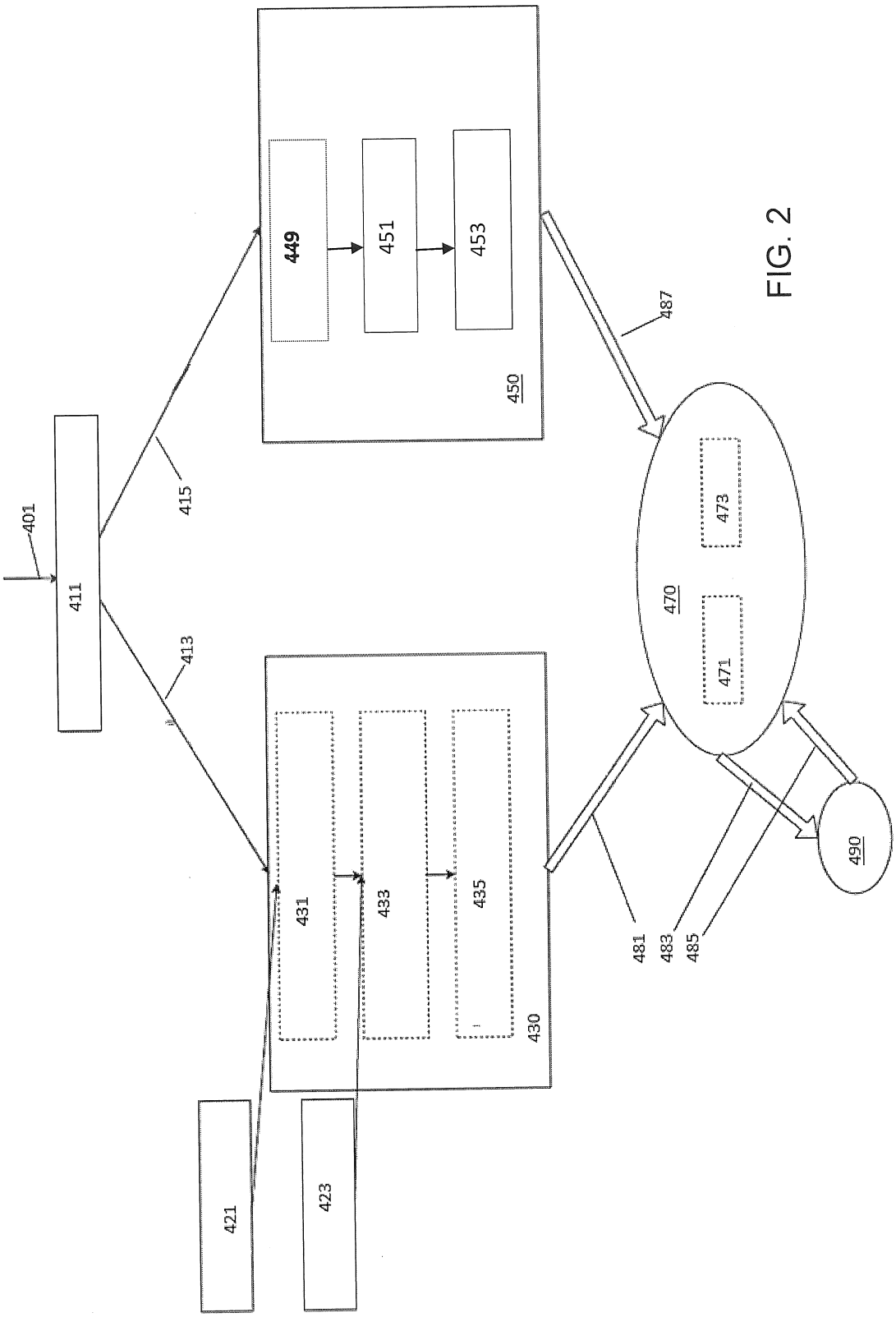

401
411
415
413
449
451
453
450
487
470
473
471
490
481
483
485
431
433
435
430
421
423

FIG. 2

3/9

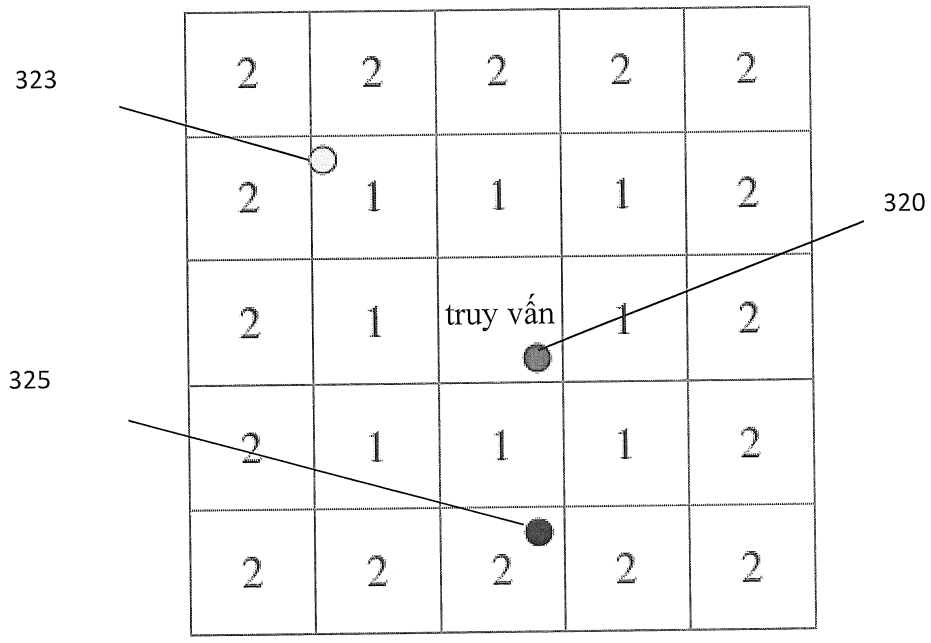


FIG. 3

Input: K, loc
Output: the K nearest Objects

Initialization
result ← priority queue of size K
min_iter ← MAX
i = 0 // current iteration

```
1: cell_list.add(index(loc))
2: while len(cell_list) > 0 do
3:    neighbours ← ∅
4:    for cell in cell_list do
5:       if cell.visited then
6:          continue
7:       end if
8:       cell.visited ← true
9:       nearest_objects ← KNearest_InCell(K, loc, cell)
         // on node sharding(cell)
10:      result, min_iter ← Merge(result, nearest_objects)
11:      neighbours.add(neighbour(cell))
12:   end for
13:   if i≥ ⌈√2 * (min_iter+1) + 1⌉ && result.size==K
      then
14:      break
15:   end if
16:   i ← i+1
17:   cell_list ← neighbours
18: end while
19: return result
```

Algorithm 1: Naive_BFS(K, loc)

FIG. 4

Input: K, loc
Output: the K nearest Objects
Initialization
result ← priority queue of size K
shard_list ← FindNearbyCrossingShards(loc)
for shard_id in shard_list **do**
    nearest_objects ← Naive_BFS(K, loc) on node=sharding(shard_id)
    result ← Merge(result, nearest_objects)
end for
return $result$
Algorithm 2: Optimised_KNearest(K, loc)

FIG. 5

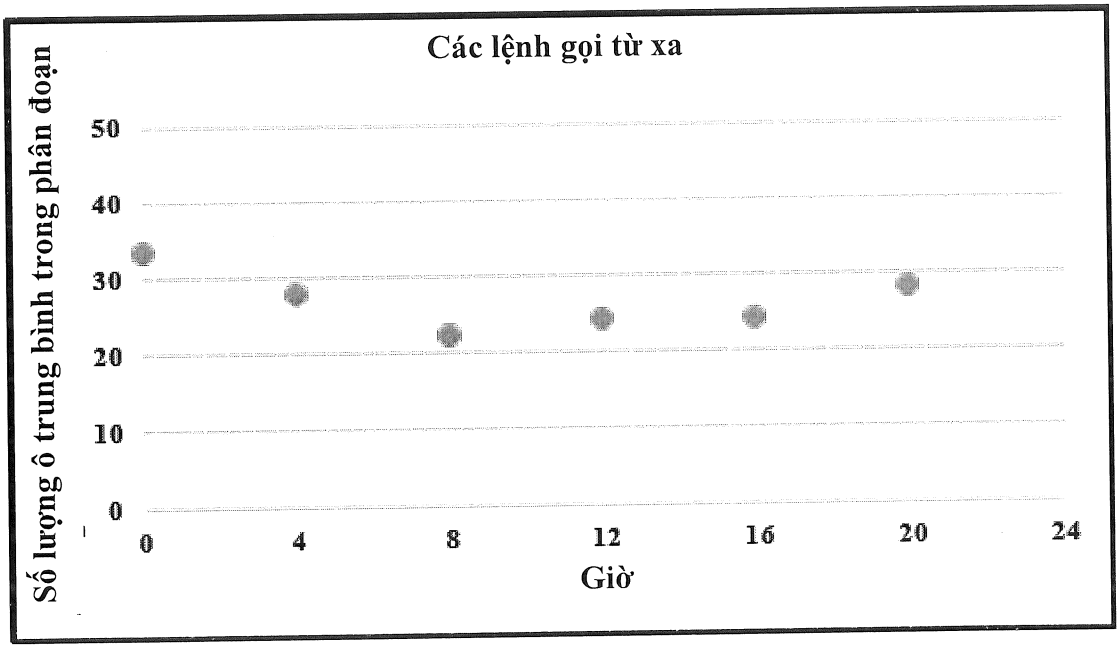
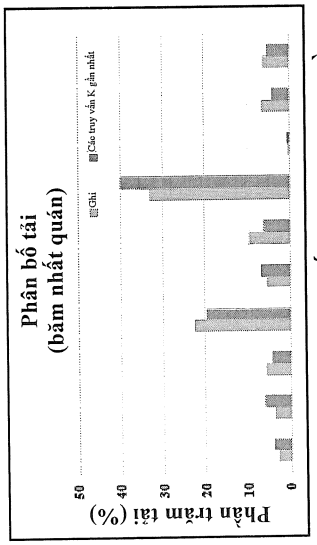
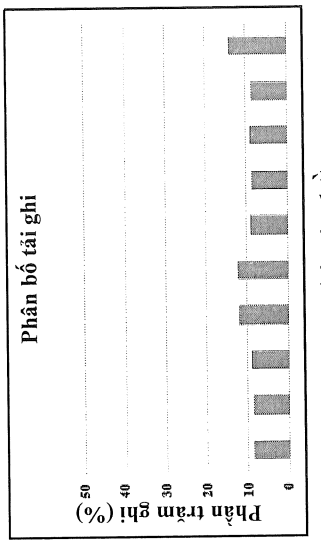


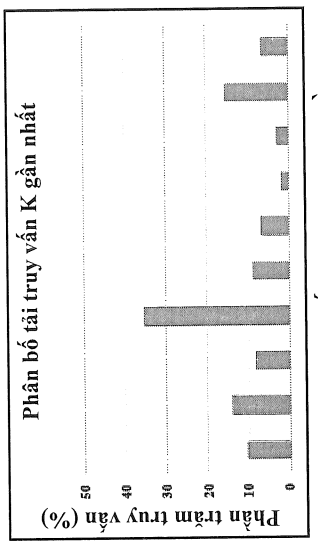
FIG. 6



(a) Tải ghi và truy vấn không cân bằng



(b) Tải ghi cân bằng



(c) Tải truy vấn không cân bằng

FIG. 7

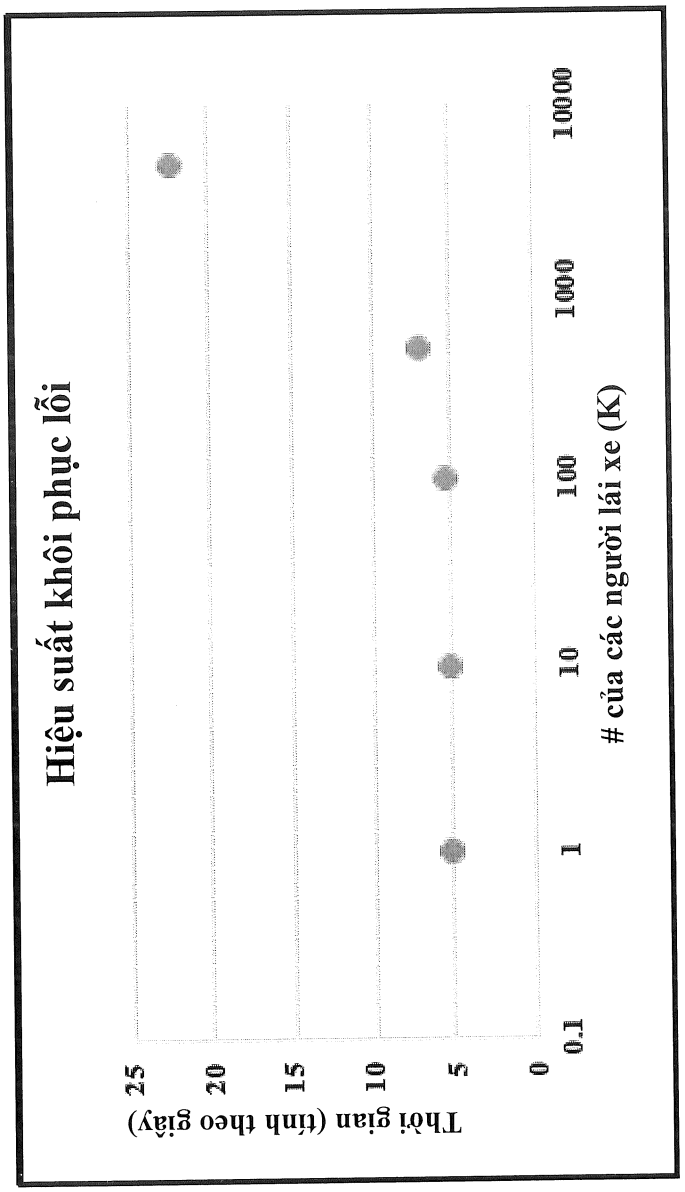

Hiệu suất khôi phục lỗi
Thời gian (tính theo giây)
25
20
15
10
5
0
0.1
1
10
100
1000
10000
của các người lái xe (K)

FIG. 8

Chỉ mục không gian địa lý	hiệu suất (ns/op)
Phương án	0,57
Geohash	108
S2	784
H3	2084

FIG. 9

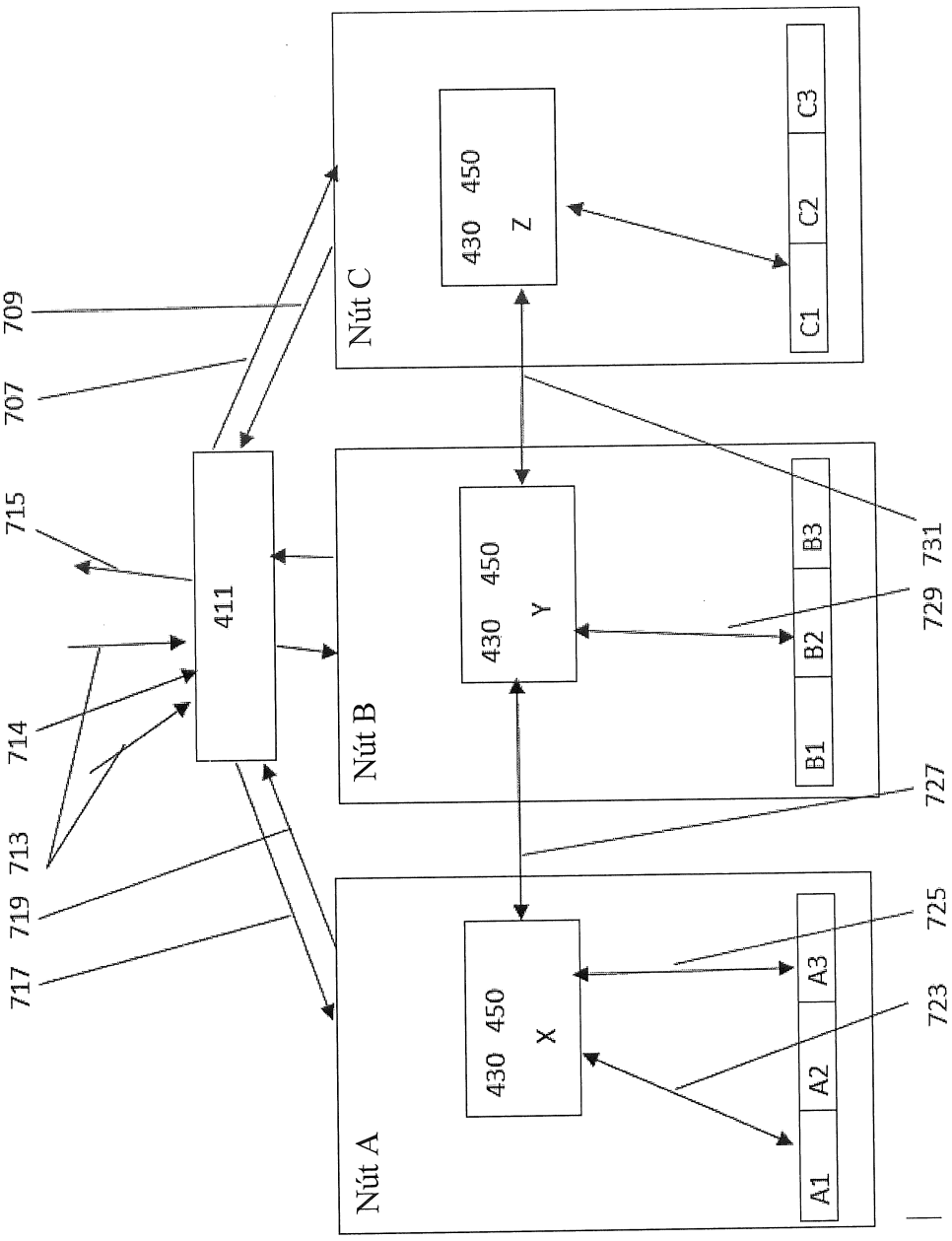

709
707
715
714
713
719
717
411
Nút A
430 450 X
A1
A2
A3
Nút B
430 450 Y
B1
B2
B3
Nút C
430 450 Z
C1
C2
C3
723
725
727
729
731

FIG. 10