(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) **1-0022735**

(51)[7] **H04L 12/54**, G06F 17/00

(13) **B**

(21) 1-2016-03331
(22) 07.09.2016
(45) 27.01.2020 382
(43) 26.12.2016 345
(73) TẬP ĐOÀN CÔNG NGHIỆP - VIỄN THÔNG QUÂN ĐỘI (VIETTEL) (VN)
Số 1 Trần Hữu Dực, Mỹ Đình 2, quận Nam Từ Liêm, thành phố Hà Nội
(72) Nguyễn Trung Hải (VN), Vũ Đức Chính (VN), Lê Thanh Bằng (VN), Nguyễn Thị Thu Thúy (VN)
(74) Công ty Luật TNHH quốc tế BMVN (BMVN INTERNATIONAL LLC)

(54) PHƯƠNG PHÁP PHÂN CHIA DỮ LIỆU NGẪU NHIÊN TRONG CÁC HỆ THỐNG PHÂN TÁN ĐA VI XỬ LÝ

(57) Sáng chế đề xuất phương pháp phân chia dữ liệu ngẫu nhiên trong các hệ thống phân tán đa vi xử lý giúp phân tán dữ liệu ngẫu nhiên đồng đều tới từng vi xử lý trong từng nút mạng và giúp giảm tải khi thực hiện tái cấu hình hệ thống bằng cách thực hiện hàm băm chỉ số của từng phần tử rồi phân chia vào các nút mạng, khác biệt ở chỗ không những dùng hàm băm chỉ số từng phần tử mà còn dùng hàm băm chuỗi mô tả từng nút để phân chia phần tử vào từng vi xử lý trong nút mạng.

Lĩnh vực kỹ thuật được đề cập

Sáng chế đề cập đến các phương pháp phân chia dữ liệu trong các hệ thống phân tán đa vi xử lý.

Tình trạng kỹ thuật của sáng chế

Trong hệ thống phân tán dữ liệu trên bộ nhớ (in-memory), về cơ bản các dữ liệu được phân chia ở các nút khác nhau và đồng thời có các bản sao của mỗi dữ liệu đó trên các nút khác để dự phòng trường hợp nút bị hỏng hoặc không thể kết nối. Hệ thống dữ liệu phân tán trên bộ nhớ có thể chia làm nhiều loại dựa trên đặc tính của hệ thống, loại dữ liệu sử dụng, phương pháp quản lý dữ liệu, v.v.. Tuy nhiên, phân loại hệ thống dựa trên phương pháp quản lý dữ liệu là khá chuẩn về mặt tính năng. Bởi vì tùy theo những yêu cầu ưu tiên về việc đọc dữ liệu, ghi dữ liệu hoặc phục hồi dữ liệu mà hệ thống sẽ đáp ứng những yêu cầu đó. Một số phương pháp quản lý dữ liệu như: lưu trữ dữ liệu theo vi xử lý - code (wide column store), lưu trữ dữ liệu dạng tài liệu (document store), lưu trữ dữ liệu theo khóa – giá trị (key/value store), v.v.. Tuy nhiên, với những cơ sở dữ liệu được tạo thành từ nhiều phần tử nhỏ, các phần tử này có thể là một số (number), giá trị (value), một chuỗi (string), hoặc một tài liệu (document), v.v.. thì phương pháp quản lý dữ liệu theo dạng khóa - giá trị là dễ dàng cho việc đọc, ghi dữ liệu cũng như thích hợp cho việc tìm kiếm và cập nhật giá trị vào các phần tử. Một số ứng dụng hệ thống phân tán dữ liệu trên bộ nhớ sử dụng phương pháp quản lý cơ sở dữ liệu theo dạng khóa – giá trị như: Redis, CouchBase, Aerospike, Cassandra, RAM Cloud, v.v..

Ở phương pháp Redis, thuật toán phân tán dữ liệu rất đơn giản trong đó các phần tử sẽ được gắn với một chỉ số (index-ID) và được phân chia đều ở các nút theo thứ tự từ thấp đến cao. Ví dụ, trong hệ thống có bốn nút: R1, R2, R3 và R4, và trong cơ sở dữ liệu có tất cả 4000 phần tử, thì các phần tử có chỉ số ID từ 1 đến 1000 sẽ nằm ở nút R1, từ 1001 đến 2000 sẽ nằm ở nút R2, v.v.. Các phiên bản Redis về sau đưa ra việc phân tán dữ liệu tốt hơn trong đó các chỉ số của phần tử sẽ được biến đổi qua một hàm băm (hash) crc32 thành một số mới, số này sẽ được chia lấy phần dư với số lượng nút trong toàn bộ hệ thống sẽ được một kết quả, số đó sẽ tương ứng với nút mà phần tử được xử lý. Ví dụ,

một phần tử có chỉ số là "foobar", sau khi biến đổi qua hàm băm crc32 sẽ được số tương ứng với phần tử này là 93024922, sau đó lấy số (93024922) chia cho 4 (số nút trong hệ thống) lấy phần dư sẽ được giá trị bằng 2, đây chính là vị trí nút mà phần tử được phân bổ đến.

Một ứng dụng khác mới được phát hành gần đây nhất là RAM Cloud. Vấn đề phân tán dữ liệu ở ứng dụng này tương đối đơn giản, cũng tương tự như các ứng dụng trên các chỉ số của phần tử được biến đổi qua một hàm băm 64 bit, và từ đó phân chia theo các khoảng giá trị mà mỗi nút chịu trách nhiệm. Tuy nhiên, các bản sao của các phần tử này lại quản lý theo phương pháp chặt chẽ hơn, đồng thời làm luôn nhiêm vụ phân tán dữ liệu bản sao. Thuật toán này được mô tả như sau: một chỉ số sau khi qua một hàm băm sẽ được một số 64 bit, sau đó số này sẽ được mô đun hóa ứng với số lượng của vùng chứa (buckets) trong hệ thống (vùng chứa là một nhóm các nút được gom thành từng nhóm và đánh số từ B1 - Bn), giá trị nhận được là vị trí mà bản sao được lưu lần lượt đến các nút nằm trong vùng chứa đó.

Tuy nhiên, thuật toán phân tán dữ liệu của hệ thống này có một số nhược điểm như sau:

- Dựa trên việc chỉ định các phần tử nằm ở nút nào, việc chỉ định này chưa thực hiện ngẫu nhiên hoàn toàn giữa các nút, mới chỉ thực hiện ngẫu nhiên của các chỉ số của các phần tử dữ liệu bằng cách sử dụng hàm băm. Khoảng giá trị mà nút nhận phần tử vẫn là khoảng chỉ định trước. Từ đó sẽ dẫn đến mất cân bằng tải giữa các nút trong hệ thống: tại một thời điểm có thể có một nút nào đó phải xử lý rất nhiều trong khi có những nút không phải xử lý gì cả.

- Các phương án phân chia dữ liệu ở trên chưa thực hiện phân tán dữ liệu đến từng vi xử lý của mỗi nút. Ví dụ, trong mỗi nút có số lượng vi xử lý khác nhau, do đó tốc độ xử lý dữ liệu ở mỗi nút là hoàn toàn khác nhau, vì vậy nếu mặc định thông số kỹ thuật mỗi nút khác nhau là giống hệt nhau sẽ dẫn đến hoạt động không đồng đều giữa các nút trong toàn bộ hệ thống.

- Việc mở rộng hay thu gọn hệ thống sẽ không đạt được hiệu năng tốt nhất. Cụ thể là khi thêm hay bớt một nút nào đó trong hệ thống, toàn bộ hệ thống sẽ phải cấu hình

và phân chia các phần tử lại từ đầu, dẫn đến phải di chuyển rất nhiều dữ liệu qua đường truyền mạng, gây ra tắc nghẽn hoặc nghiêm trọng hơn là có thể treo toàn bộ hệ thống.

Bản chất kỹ thuật của sáng chế

Mục đích của sáng chế là phân chia dữ liệu trong hệ thống phân tán đa vi xử lý sao cho dữ liệu được phân bố đồng đều nhất tới từng vi xử lý trong hệ thống, các dữ liệu này cũng phải được sao lưu trên những nút khác trên hệ thống để phục hồi dữ liệu kịp thời khi có một nút bị hỏng, đồng thời việc di chuyển dữ liệu xảy ra ít nhất khi tái cấu hình (thu gọn hay mở rộng số nút) trong hệ thống.

Để đạt được mục đích nêu trên, sáng chế đề xuất phương pháp phân chia dữ liệu ngẫu nhiên trong hệ thống phân tán đa vi xử lý bao gồm các bước sau:

a) xác định danh sách nút mạng sẽ lưu trữ dữ liệu của toàn bộ hệ thống dưới dạng địa chỉ IP của nút đó, sau đó tạo thành bảng băm các chỉ số trong nút, bước này được thực hiện như sau:

(i) đầu tiên, các nút trong hệ thống được biểu diễn bằng một chuỗi các kí tự bằng địa chỉ IP của nút đó.

(ii) sau đó, chuỗi này được băm thành một chuỗi 64 bit bằng một hàm băm, N bit đầu tiên của chuỗi băm này sẽ được dùng làm chỉ số băm của nút ID_nút định danh phân tán dữ liệu trong hệ thống.

(iii) các chỉ số băm của nút trong hệ thống được xắp xếp theo thứ tự tăng dần thành một bảng băm chỉ số nút.

b) khi cần xác định vi xử lý nào của nút mạng nào trong hệ thống sẽ xử lý dữ liệu, hệ thống cần xác định chỉ số đại diện cho phần tử đó để phân tán dữ liệu, cụ thể bước này được thực hiện như sau:

(i) chỉ số của phần tử cập nhật cũng được băm thành một số 64 bit dựa trên một hàm băm, chỉ số băm 64 bit này được lấy N bit đầu tiên làm chỉ số băm của phần tử.

(ii) chỉ số băm của phần tử được so sánh với bảng băm của chỉ số nút, chỉ số băm của nút nào nhỏ hơn và gần nhất với chỉ số băm của phần tử, thì phần tử đó sẽ được chỉ định đến xử lý ở nút đó, đồng thời phần tử nằm ở nút này sẽ được xem là phần tử bản lưu chính (phần tử chính) và nút chứa phần tử

chính gọi là nút chính, các bản sao của phần tử này sẽ được sao lưu đến các nút có chỉ số băm lớn hơn và gần nhất với chỉ số băm của nút chính tùy theo hệ số sao chép đã cài đặt cho hệ thống, nút chứa các bản sao của phần tử gọi là nút phụ, các xử lý đối với phần tử này như cập nhật và xóa phần tử sẽ được thực hiện trên bản lưu chính, sau đó mới cập nhật đến các bản sao của nó.

(iii) sau khi xác định được nút chứa phần tử, thì sẽ lấy chuỗi băm của phần tử chia cho số vi xử lý của nút đó, phần dư của phép chia sẽ xác định vi xử lý nào trong nút sẽ xử lý và lưu phần tử.

Theo phương pháp của sáng chế, việc phân tán dữ liệu vẫn sử dụng hàm băm chỉ số của mỗi phần tử, nhưng đồng thời sử dụng thêm hàm băm địa chỉ IP của từng nút trong hệ thống. Hệ thống sẽ sử dụng những kết quả băm này để lựa chọn một nút lưu dữ liệu chính thông qua so sánh khoảng cách từ phân khu (partition) chứa dữ liệu đến địa chỉ IP của nút, và một vài nút khác để lưu dữ liệu sao lưu sao cho khoảng cách từ phân khu chứa dữ liệu đến địa chỉ IP của các nút lưu là nhỏ nhất. Việc này sẽ tạo ra sự phân tán ngẫu nhiên đồng đều hơn tới từng nút, đồng thời cũng phục hồi dữ liệu kịp thời khi có một hoặc một vài nút bị hỏng, tránh được mất mát dữ liệu.

Sau đó khi chọn được nút lưu trữ dữ liệu, thì việc phân chia dữ liệu trên từng vi xử lý sẽ được thực hiện bằng cách lấy toàn bộ hoặc một phần kết quả băm chỉ số của mỗi phần tử chia cho số vi xử lý của nút đó lấy phần dư sẽ là chỉ số của vi xử lý sẽ lưu trữ dữ liệu. Điều này đảm bảo phân chia dữ liệu tới từng vi xử lý trong nút.

Khi xảy ra tái cấu hình hệ thống (thêm hay bớt một hoặc một vài nút nào đó), do sáng chế không sử dụng phương pháp chia chỉ số băm của mỗi phần tử cho số nút nên sẽ không xảy ra việc tính toán lại và di chuyển toàn bộ dữ liệu qua đường truyền, mà thay vào đó sẽ chỉ di chuyển một phần dữ liệu liên quan tới nút được thêm mới hay nút vừa mất đi. Điều này sẽ giảm bớt rất nhiều nguyên nhân gây tắc nghẽn đường truyền, việc tái cấu hình hay sao lưu, phục hồi sẽ nhanh hơn, đồng thời hệ thống cũng bớt được nguy cơ bị treo.

Mô tả vắn tắt các hình vẽ

Các mục đích, khía cạnh, dấu hiệu và ưu điểm của sáng chế nêu trên sẽ trở nên rõ ràng và được hiểu dễ dàng hơn bằng cách tham khảo các mô tả sau đây có kết hợp với các hình vẽ kèm theo, trong đó:

Hình 1 là hình vẽ dạng sơ đồ khối mô tả mô hình tổng quan một hệ thống phân tán dữ liệu lớn theo phương pháp của sáng chế.

Hình 2 là hình vẽ dạng sơ đồ khối mô tả cách tạo bảng băm chỉ số các nút trong toàn bộ hệ thống theo phương pháp của sáng chế.

Hình 3 là hình vẽ dạng sơ đồ khối mô tả thuật toán sử dụng hàm băm để biến đổi một chỉ số của phần tử sang một số N bit.

Hình 4 là hình vẽ dạng sơ đồ khối mô tả cách tìm ra vi xử lý lưu trữ và xử lý một phần tử nào đó.

Hình 5 là hình vẽ dạng sơ đồ khối biểu diễn việc di chuyển dữ liệu khi thêm một nút mới vào hệ thống.

Hình 6 là hình vẽ dạng sơ đồ khối biểu diễn việc di chuyển dữ liệu khi bớt một nút trong hệ thống và nút này không bị treo.

Hình 7 là hình vẽ dạng sơ đồ khối mô tả biểu diễn việc di chuyển dữ liệu khi bớt một nút trong hệ thống và nút này bị treo.

Mô tả chi tiết sáng chế

Trong hệ thống phân tán, để dễ dàng cho việc truy cập vào các phần tử trong cơ sở dữ liệu và phân tán dữ liệu trên hệ thống đồng đều nhất thì mỗi phần tử nhỏ nhất của cơ sở dữ liệu sẽ có một chỉ số, chỉ số này sẽ được dùng để xác định vị trí nút nào trong hệ thống lưu trữ, xử lý đối với dữ liệu đó.

Khi truy vấn một phần tử trong cơ sở dữ liệu, thì vấn đề trả kết quả truy vấn lại là vấn đề lớn đối với một cơ sở dữ liệu tập trung do việc giới hạn của băng thông, thời gian tìm kiếm và xử lý giới hạn trên mỗi nút. Do đó, để giảm thời gian truy vấn, phân tán dữ liệu phải được xử lý đồng đều giữa các nút mạng, nhằm giảm sự tập trung của dữ liệu tại một vị trí. Đồng thời, hệ thống phân tán dữ liệu cũng tạo nhiều bản sao của các phần tử ở các nút khác nhau, để phòng ngừa trường hợp một nút trong hệ thống bị hỏng hoặc mất kết nối. Sự phân tán các bản sao này cũng phải đồng đều hoặc tự động cập nhật dữ liệu

mới cho các phần tử cũ để không làm tăng khối lượng của cơ sở dữ liệu và đồng thời dễ dàng phục hồi dữ liệu khi một nút trong hệ thống bị mất.

Bình thường, một hệ thống cơ sở dữ liệu phân tán sẽ có nhiều triệu bản ghi trong một hệ thống. Các yêu cầu truy vấn dữ liệu liên tục xảy ra, do đó các dữ liệu phân tán đồng đều ở các nút mạng là rất quan trọng, tuy nhiên việc xử lý ở mỗi nút có hiệu năng khác nhau phụ thuộc vào hiệu năng phần cứng của hệ thống. Vì vậy, ngoài vấn đề phân dữ liệu đều ở các nút mạng, vấn đề quan tâm sâu hơn là hiệu năng xử lý của mỗi nút.

Hình 1 biểu diễn tổng quan một hệ thống phân tán dữ liệu, bao gồm các thành phần máy khách (client) 1, nút (node) 2 và các vi xử lý (core) 3 trong từng nút, các nút 2 và máy khách 1 kết nối với nhau thông qua các kết nối mạng (connection) 4. Một hệ thống cơ sở dữ liệu phân tán cũng đồng thời gặp vấn đề khi có một hay một vài nút nào đó bị sự cố (crash), đường truyền mạng không ổn định, hay khi người quản trị muốn thu gọn hay mở rộng hệ thống. Khi đó, sự phân tán dữ liệu cũng cần tính toán lại. Nếu phương án lựa chọn để phân tán dữ liệu không tốt thì sẽ gây ra phải di chuyển rất nhiều dữ liệu gây gánh nặng cho hệ thống.

Phương pháp theo sáng chế đáp ứng được những vấn đề này cho việc phân tán dữ liệu đồng đều trong hệ thống có hiệu năng xử lý của các vi xử lý khác nhau. Dữ liệu đồng thời phân tán ngẫu nhiên và có các bản sao cũng ngẫu nhiên trên toàn bộ hệ thống. Do đó, cân bằng được phân tán dữ liệu trên toàn bộ hệ thống với hệ cơ sở dữ liệu lớn. Bên cạnh đó, khi xảy ra tái cấu hình hệ thống (thêm hay bớt một hoặc một vài nút nào đó), do sáng chế không sử dụng phương pháp chia chỉ số băm của mỗi phần tử cho số nút nên sẽ không xảy ra việc tính toán lại và di chuyển toàn bộ dữ liệu qua đường truyền, mà thay vào đó sẽ chỉ di chuyển một phần dữ liệu liên quan tới nút được thêm mới hay nút vừa mất đi. Điều này sẽ giảm bớt rất nhiều nguyên nhân gây tắc nghẽn đường truyền, việc tái cấu hình hay sao lưu, phục hồi sẽ nhanh hơn, đồng thời hệ thống cũng bớt được nguy cơ bị sự cố (crash).

Việc thực hiện phương án đề xuất có thể được lập trình bằng các ngôn ngữ lập trình nền tảng như C, C++, Java, v.v. theo các bước được mô tả cụ thể ở dưới đây:

Bước 1: Hệ thống phân tán phải xác định được danh sách nút mạng sẽ lưu trữ dữ liệu của toàn bộ hệ thống dưới dạng địa chỉ IP của nút đó, sau đó tạo thành bảng băm các chỉ số trong nút bằng các bước được mô tả như ở hình 2:

- Đầu tiên, các nút trong hệ thống được biểu diễn bằng một chuỗi các kí tự bằng địa chỉ IP của nút 5.

- Chuỗi này sau đó sẽ được băm thành một chuỗi 64 bit bằng một hàm băm. N bit đầu tiên của chuỗi băm này sẽ được dùng làm chỉ số băm của nút ID_Nút 6 định danh phân tán dữ liệu trong hệ thống. Trong đó, 2 mũ N sẽ xác định số lượng phân vùng trong hệ thống.

- Các chỉ số băm của nút trong hệ thống được sắp xếp theo thứ tự tăng dần thành một bảng băm chỉ số nút 7.

Bước 2: Khi cần xác định vi xử lý nào của nút mạng nào trong hệ thống sẽ xử lý dữ liệu, mỗi dữ liệu hay còn gọi là một phần tử sẽ có một chỉ số đại diện cho phần tử đó. Hệ thống sẽ dựa vào chỉ số này để xác định vi xử lý và nút mạng xử lý phần tử theo các bước được mô tả như ở hình 3:

- Mỗi phần tử có một chỉ số của phần tử 8 đại diện cho phần tử đó để phân tán dữ liệu.

- Chỉ số của phần tử 8 cập nhật cũng được băm thành một số 64 bit dựa trên một hàm băm. Chỉ số băm 64 bit này được lấy N bit đầu tiên làm chỉ số băm của phần tử 9 để biết được dữ liệu được phân bố và xử lý ở phân vùng nào.

- Chỉ số băm của phần tử 9 được so sánh với bảng băm của chỉ số nút 7, chỉ số băm của nút nào nhỏ hơn và gần nhất với chỉ số băm của phần tử, thì phần tử đó sẽ được chỉ định đến xử lý ở nút đó, đồng thời phần tử nằm ở nút này sẽ được xem là phần tử bản lưu chính (phần tử chính - object master). Trong một phân vùng, nút chứa phần tử chính gọi là nút chính (master). Các bản sao của phần tử này sẽ được sao lưu đến các nút có chỉ số băm lớn hơn và gần nhất với chỉ số băm của nút chính tùy theo hệ số sao chép đã cài đặt cho hệ thống. Nút chứa các bản sao của phần tử gọi là nút phụ (slave). Các xử lý đối với phần tử này như cập nhật, xóa phần tử sẽ được thực hiện trên bản lưu chính, sau đó mới cập nhật đến các bản sao của nó.

- Sau khi xác định được nút chứa phần tử, thì sẽ lấy chuỗi băm của phần tử chia cho số vi xử lý của nút đó. Phần dư của phép chia sẽ xác định vi xử lý nào trong nút sẽ xử lý và lưu phần tử thuộc phân vùng tương ứng với chỉ số băm của phần tử. Quá trình áp dụng cho cả các phần tử bản lưu chính và phụ.

Hình 4 là một ví dụ với hệ thống có 6 nút, phần tử có chỉ số băm ID_Object = 71, hệ số sao chép R =2:

- Bước 1: Xác định ID_Nút 2 nhỏ hơn và gần nhất với ID_Object nên phần tử sẽ được xử lý tại phân vùng 71 với nút 2 sẽ là bản lưu chính của phần tử, các nút 3 và 4 là nút phụ của phần tử.

- Bước 2: Xác định vi xử lý sẽ xử lý phần tử:

+ Nút 2 có 8 vi xử lý, 71 chia 8 dư 7 nên vi xử lý 7 của nút 2 thuộc phân vùng 71 chứa bản chính của phần tử.

+ Nút 3 có 2 vi xử lý, 71 chia 2 dư 1 nên vi xử lý 1 của nút 3 thuộc phân vùng 71 chứa bản sao của phần tử.

+ Nút 4 có 4 vi xử lý, 71 chia 4 dư 3 nên vi xử lý 3 của nút 4 thuộc phân vùng 71 chứa bản sao của phần tử.

Trong trường hợp hệ thống có thêm nút mạng mới như biểu diễn trong Hình 5, bảng băm chỉ số nút hệ thống sẽ tăng thêm một hàng, do đó có thể thay đổi cấu hình nút trong một phân vùng và hệ thống phải tự động sắp xếp lại dữ liệu. Khi đó, chỉ có phần dữ liệu có chỉ số băm thuộc về nút mạng mới thuộc phân vùng thay đổi phải di chuyển trong mạng.

Trong trường hợp người quản trị muốn bỏ một nút mạng, nút mạng này không bị treo trong Hình 6, bảng băm chỉ số nút của hệ thống giảm đi một hàng, danh sách nút trước đó của phân vùng bao gồm nút 2, nút 3 sẽ chuyển thành bao gồm nút 1, nút 3. Do đó, những dữ liệu được lưu trong nút này (nút 2) sẽ phải di chuyển sang nút có chỉ số băm liền trước đó (nút 1) trong bảng.

Trường hợp thể hiện trên hình 7, hệ thống có một nút mạng bị treo (nút 2). Nút này sẽ chứa dữ liệu bản chính của một vài phần dữ liệu nào đó và dữ liệu bản sao của một vài phần dữ liệu khác. Những dữ liệu bản sao thì không cần phải sao lưu. Đối với những dữ liệu bản chính thì nút có chỉ số băm ngay sau nút mạng bị treo (nút 3) sẽ lưu bản sao

của những dữ liệu đó, vì vậy hệ thống chỉ cần sao chép những dữ liệu này sang nút có chỉ số băm liền trước nút mạng bị treo (nút 1) là được.

Như vậy, phương án đề xuất có thể được áp dụng đối với những hệ thống phân tán đa vi xử lý, cần lưu trữ dữ liệu của hệ thống trong một phạm vi hệ thống mạng nhiều nút mạng, mỗi nút mạng có nhiều vi xử lý khác nhau. Phương án đề xuất này sẽ giúp phân chia dữ liệu đồng đều tới từng vi xử lý trong từng nút mạng, có thể được thực hiện bằng các ngôn ngữ lập trình nền tảng như C, C++, Java, v.v..

Hiệu quả đạt được

Hiệu quả đạt được của sáng chế là phân chia dữ liệu trong hệ thống phân tán đa vi xử lý để:

- Dữ liệu được phân bố đồng đều nhất tới từng vi xử lý trong hệ thống.

- Dữ liệu được sao lưu trên những vi xử lý khác của những nút khác trên hệ thống.

- Việc di chuyển, sao chép dữ liệu xảy ra ít nhất khi tái cấu hình (thu gọn hay mở rộng số nút) trong hệ thống.

Yêu cầu bảo hộ

1. Phương pháp phân chia dữ liệu ngẫu nhiên trong hệ thống phân tán đa vi xử lý bao gồm các bước sau:

a) xác định danh sách nút mạng lưu trữ dữ liệu của toàn bộ hệ thống dưới dạng địa chỉ IP của nút đó, trong đó bước này bao gồm:

(i) biểu diễn các nút trong hệ thống bằng một chuỗi các kí tự bằng địa chỉ IP của nút đó;

(ii) băm chuỗi kí tự này thành một chuỗi 64 bit bằng một hàm băm, N bit đầu tiên của chuỗi băm này được dùng làm chỉ số băm của nút (ID_nút) định danh phân tán dữ liệu trong hệ thống;

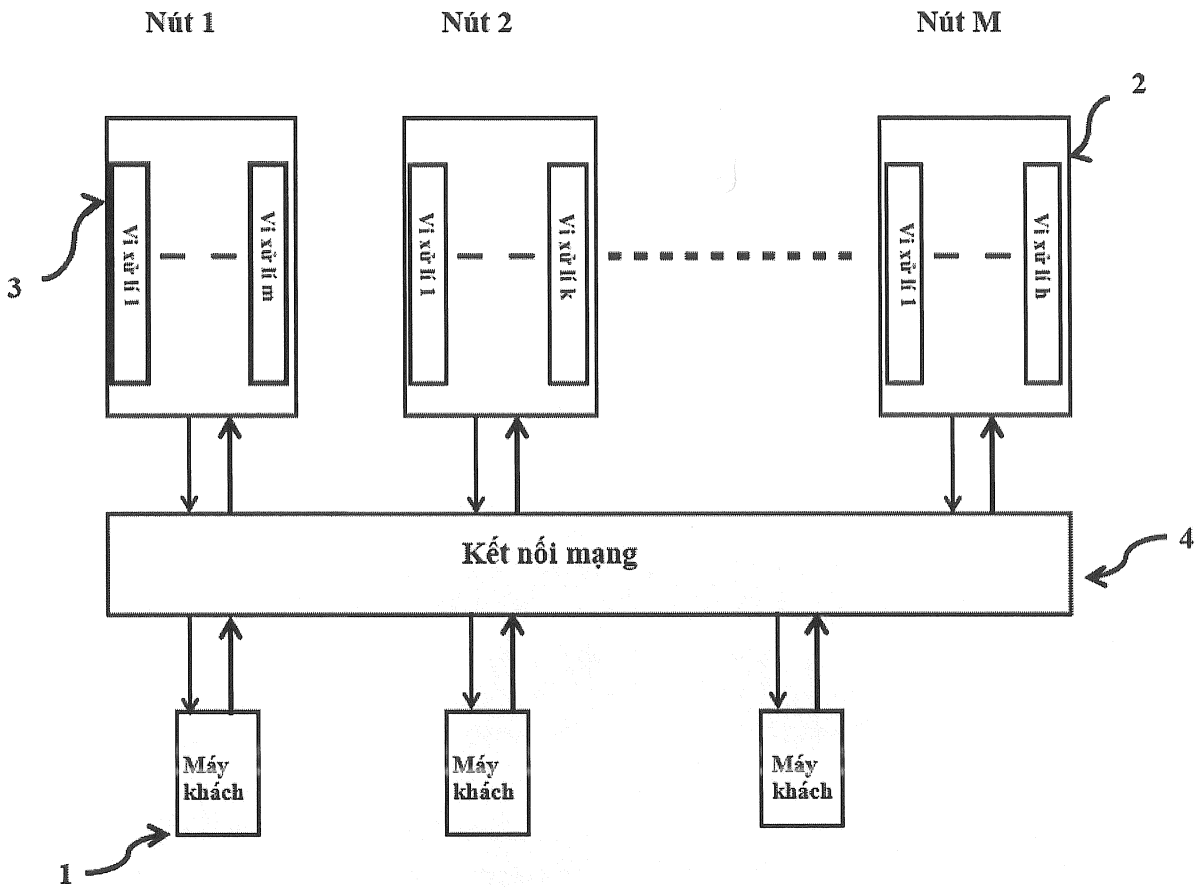
(iii) sắp xếp các chỉ số băm của nút trong hệ thống theo thứ tự tăng dần thành một bảng băm chỉ số nút;

b) xác định vi xử lý cụ thể của nút mạng trong hệ thống để xử lý dữ liệu bằng cách xác định chỉ số đại diện cho dữ liệu (phần tử) này để phân tán dữ liệu, trong đó bước này bao gồm:

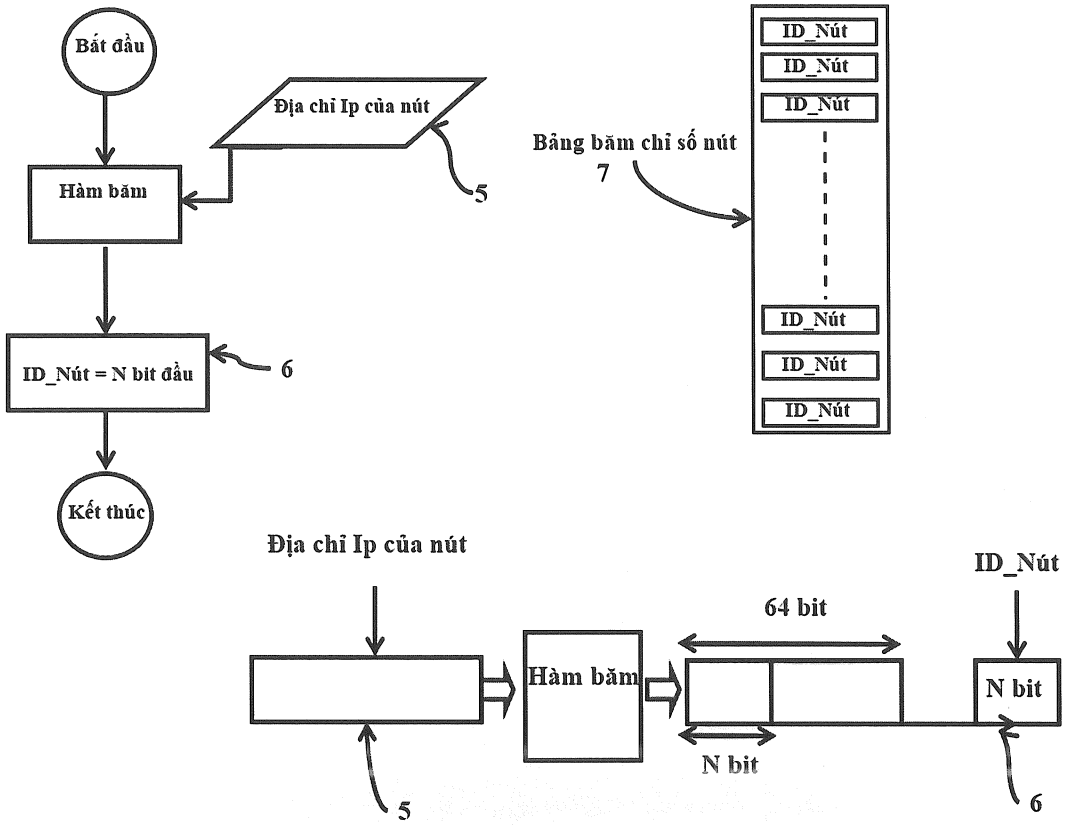
(i) băm chỉ số của phần tử thành một số 64 bit dựa trên một hàm băm, chỉ số băm 64 bit này được lấy N bit đầu tiên làm chỉ số băm của phần tử;

(ii) so sánh chỉ số băm của phần tử với bảng băm của chỉ số nút, chỉ số băm của nút nào nhỏ hơn và gần nhất với chỉ số băm của phần tử, thì phần tử đó sẽ được chỉ định đến xử lý ở nút đó, trong đó phần tử nằm ở nút này là phần tử bản lưu chính (phần tử chính) và nút chứa phần tử chính là nút chính, các bản sao của phần tử này được sao lưu đến các nút có chỉ số băm lớn hơn và gần nhất với chỉ số băm của nút chính tùy theo hệ số sao chép đã cài đặt cho hệ thống và nút chứa các bản sao của phần tử là nút phụ;

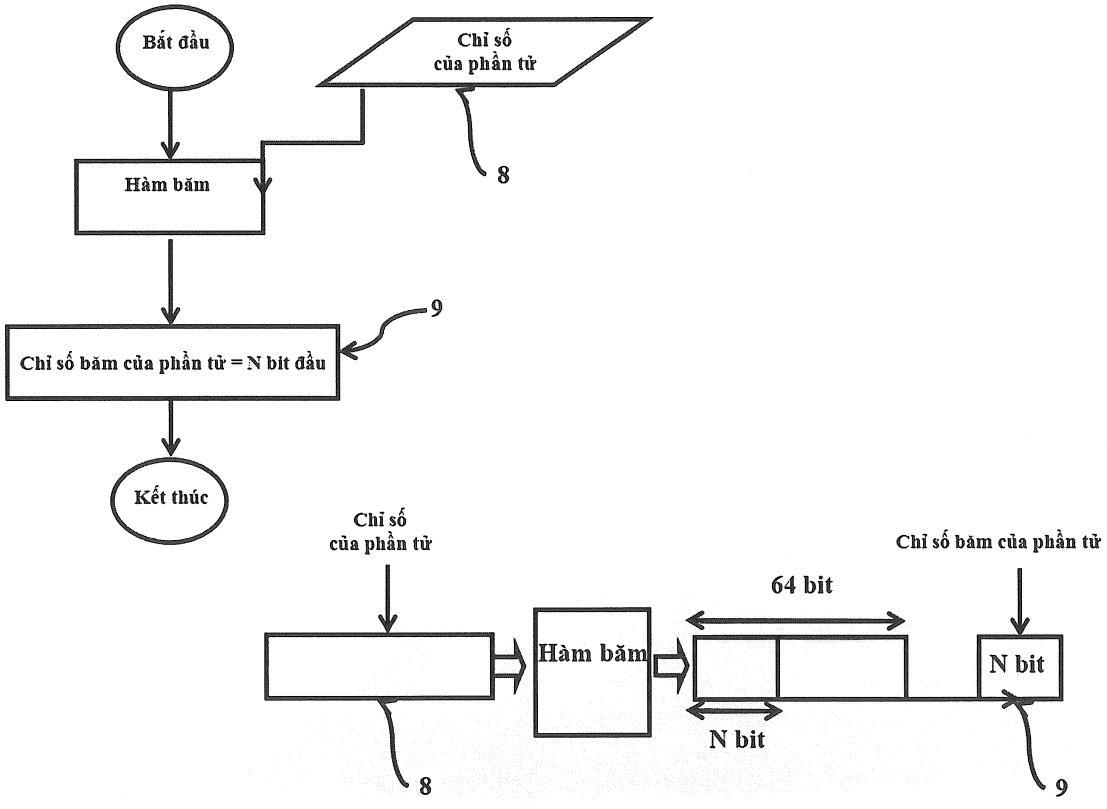
(iii) xác định vi xử lý của nút thực hiện xử lý và lưu phần tử bằng cách xác định phần dư của phép chia chỉ số băm của phần tử cho số vi xử lý của nút đó.


Nút 1
Nút 2
Nút M
2
3
Vi xử lí 1
Vi xử lí m
Vi xử lí 1
Vi xử lí k
Vi xử lí 1
Vi xử lí h
Kết nối mạng
4
Máy khách
Máy khách
Máy khách
1

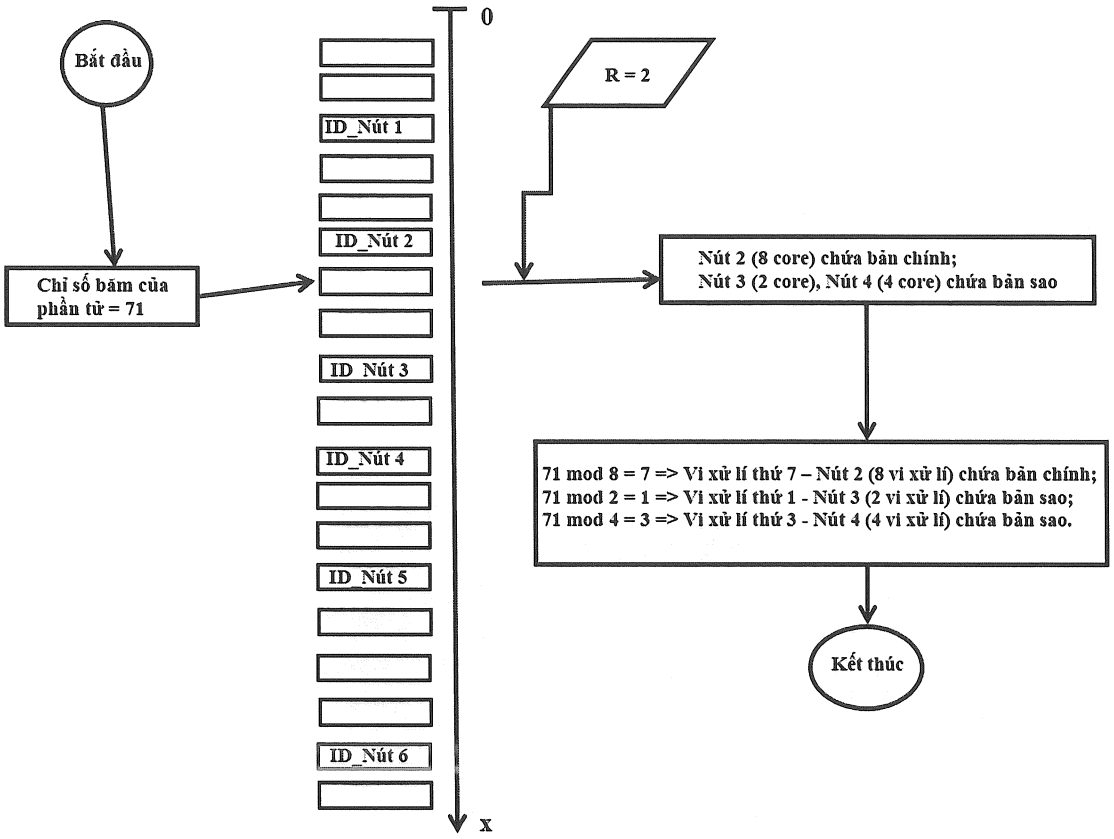
Hình 1


Bắt đầu
Địa chỉ Ip của nút
5
Hàm băm
ID_Nút = N bit đầu
6
Kết thúc
Bảng băm chỉ số nút
7
ID_Nút
ID_Nút
ID_Nút
ID_Nút
ID_Nút
ID_Nút
Địa chỉ Ip của nút
Hàm băm
64 bit
N bit
ID_Nút
N bit
5
6

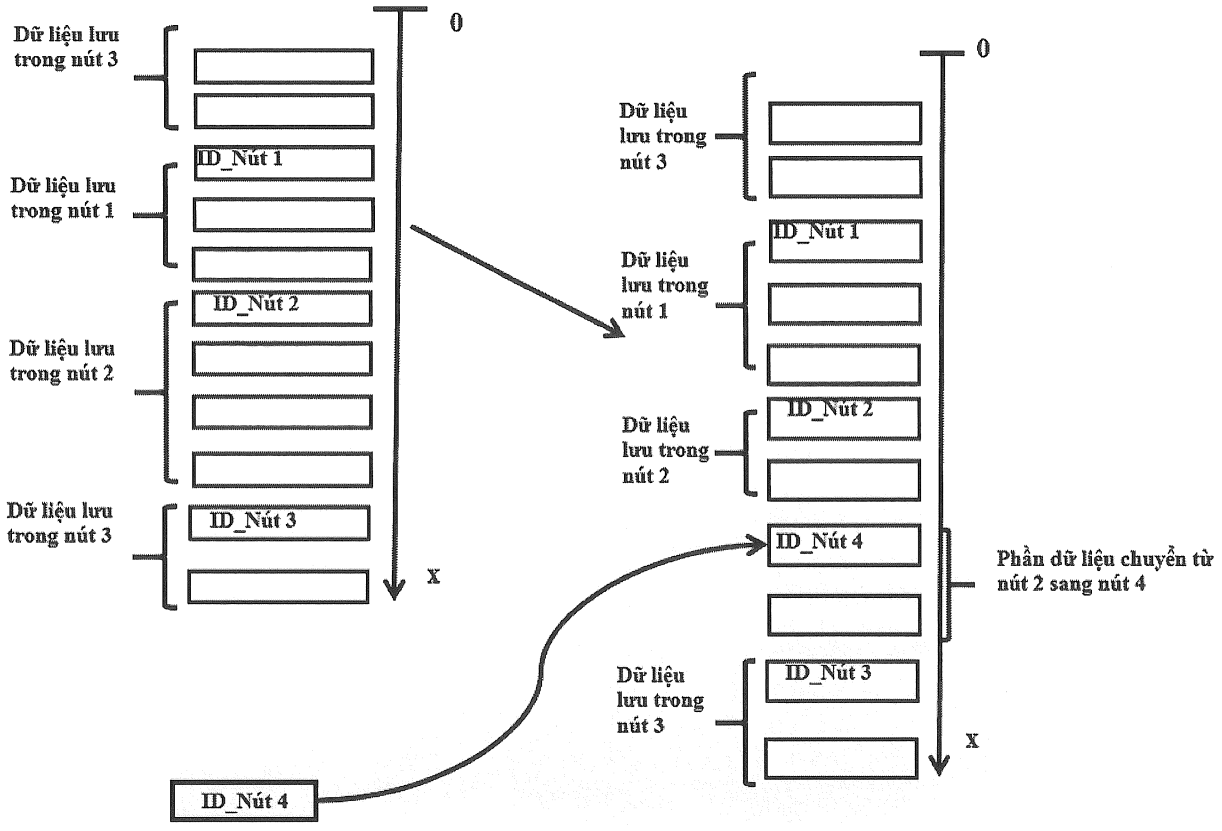
Hình 2


Bắt đầu
Chỉ số
của phần tử
8
Hàm băm
Chỉ số băm của phần tử = N bit đầu
9
Kết thúc
Chỉ số
của phần tử
Hàm băm
64 bit
N bit
Chỉ số băm của phần tử
N bit
8
9

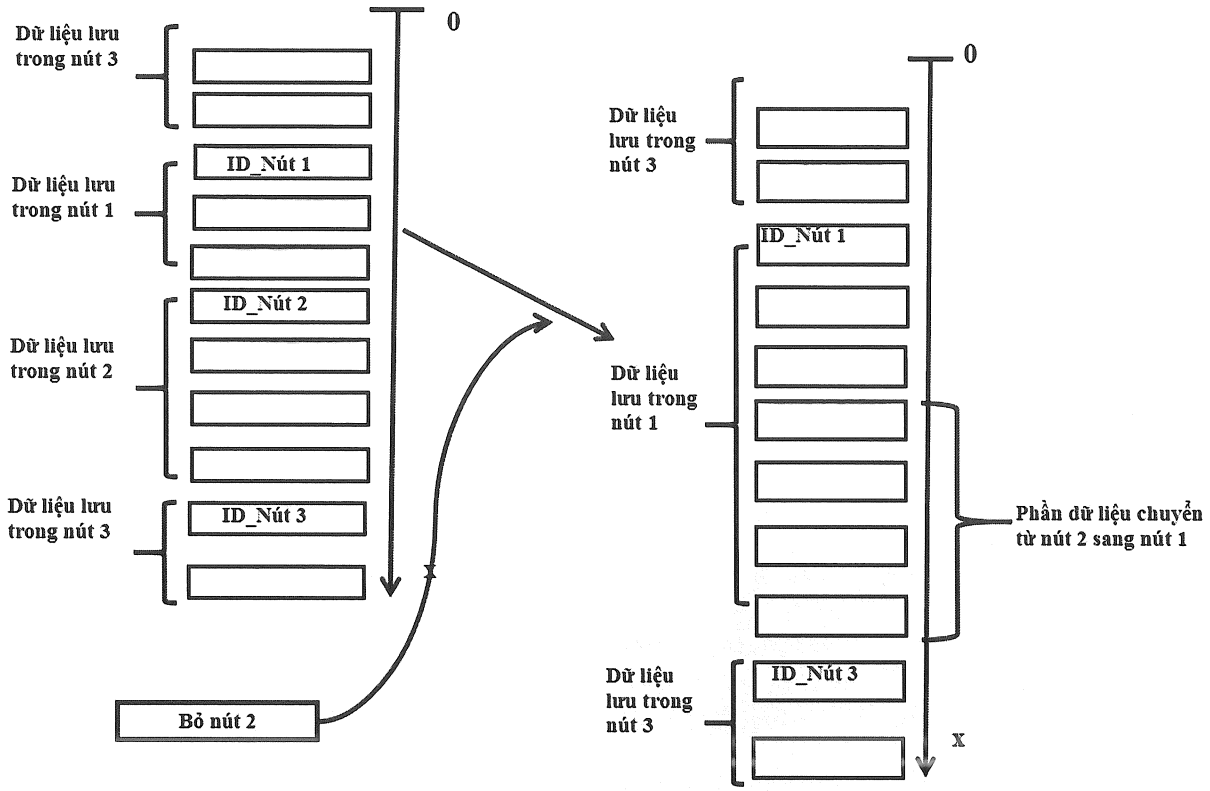
Hình 3


Bắt đầu
Chỉ số băm của phần tử = 71
0
ID_Nút 1
ID_Nút 2
ID_Nút 3
ID_Nút 4
ID_Nút 5
ID_Nút 6
x
R = 2
Nút 2 (8 core) chứa bản chính;
Nút 3 (2 core), Nút 4 (4 core) chứa bản sao
71 mod 8 = 7 => Vi xử lí thứ 7 – Nút 2 (8 vi xử lí) chứa bản chính;
71 mod 2 = 1 => Vi xử lí thứ 1 - Nút 3 (2 vi xử lí) chứa bản sao;
71 mod 4 = 3 => Vi xử lí thứ 3 - Nút 4 (4 vi xử lí) chứa bản sao.
Kết thúc

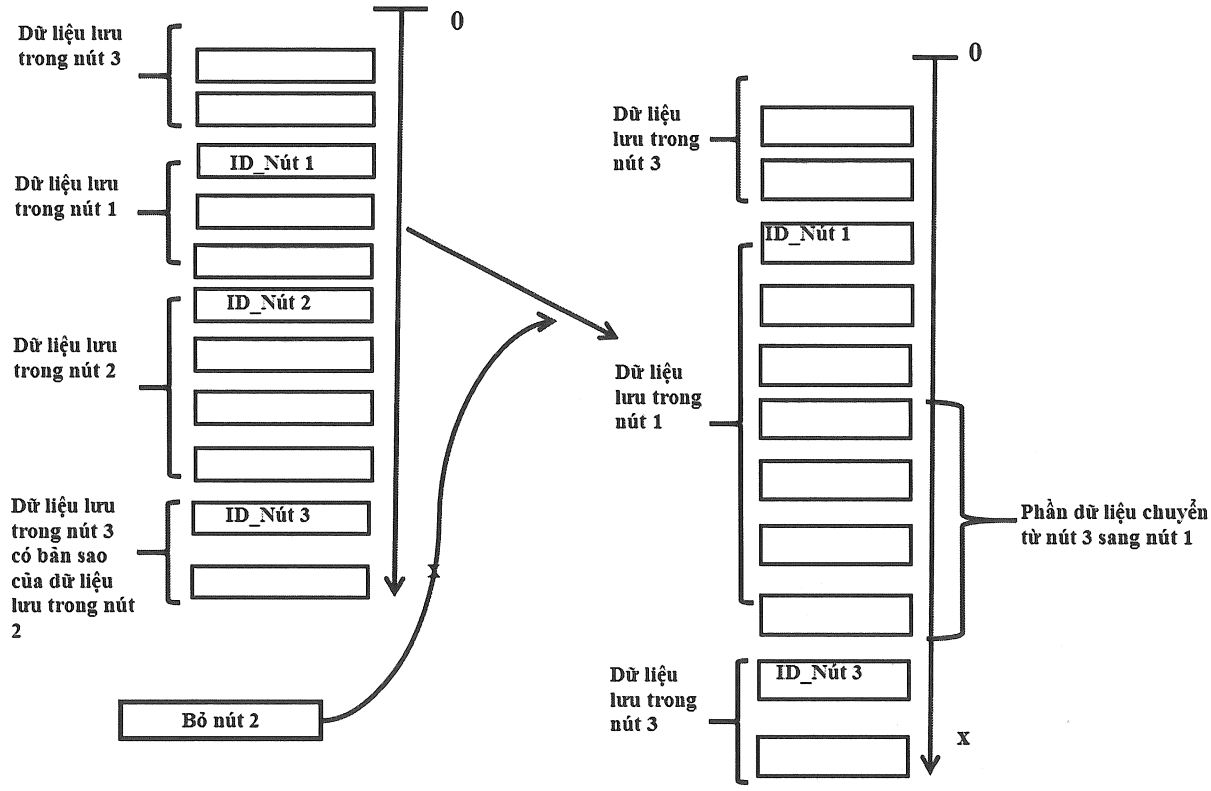
Hình 4


Dữ liệu lưu trong nút 3
Dữ liệu lưu trong nút 1
Dữ liệu lưu trong nút 2
Dữ liệu lưu trong nút 3
ID_Nút 1
ID_Nút 2
ID_Nút 3
0
x
ID_Nút 4
Dữ liệu lưu trong nút 3
Dữ liệu lưu trong nút 1
Dữ liệu lưu trong nút 2
Dữ liệu lưu trong nút 3
ID_Nút 1
ID_Nút 2
ID_Nút 4
ID_Nút 3
0
x
Phần dữ liệu chuyển từ nút 2 sang nút 4

Hình 5


Dữ liệu lưu trong nút 3
0
ID_Nút 1
Dữ liệu lưu trong nút 1
ID_Nút 2
Dữ liệu lưu trong nút 2
Dữ liệu lưu trong nút 3
ID_Nút 3
Bỏ nút 2
Dữ liệu lưu trong nút 3
0
ID_Nút 1
Dữ liệu lưu trong nút 1
Phần dữ liệu chuyển từ nút 2 sang nút 1
Dữ liệu lưu trong nút 3
ID_Nút 3
x

Hình 6


Dữ liệu lưu trong nút 3
0
ID_Nút 1
Dữ liệu lưu trong nút 1
ID_Nút 2
Dữ liệu lưu trong nút 2
Dữ liệu lưu trong nút 3 có bản sao của dữ liệu lưu trong nút 2
ID_Nút 3
Bỏ nút 2
Dữ liệu lưu trong nút 3
0
ID_Nút 1
Dữ liệu lưu trong nút 1
Phần dữ liệu chuyển từ nút 3 sang nút 1
Dữ liệu lưu trong nút 3
ID_Nút 3
x

Hình 7