ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

NGUYỄN HỮU LOAN

GIẢI PHÁP BACKUP DỮ LIỆU, SỬ DỤNG CƠ CHẾ PHÂN CỤM ĐỘNG TRONG MẠNG NGANG HÀNG CÓ CẤU TRÚC

Ngành: Công nghệ thông tin
Chuyên ngành: Hệ thống thông tin
Mã số: 60.48.01.04

TÓM TẮT LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

Hà Nội – 2017

DANH MỤC CÁC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Từ viết tắt	Giải nghĩa
Capacity	Khả năng lưu trữ của một node
Chord	Là một giao thức trong mạng ngang hàng biểu diễn mạng dưới dạng vòng tròn.
Node	Diễn tả một thực thể trong mạng như là peer hoặc máy tính kết nối mạng
DHT (Distributed Hash Table)	Bảng băm phân tán
Entry	Là một bước định tuyến trong bảng định tuyến
ID (Identification number)	Một số để định danh cho một node
Peer	Một node trong mạng ngang hàng
P2P (Peer to peer)	Mạng ngang hàng
Supernode	Là một node tương tự như server, có khả năng chuyển tiếp thông tin và kết nối tới nhiều node khác trong hệ thống

MỤC LỤC

MỞ ĐẦU

Một mạng ngang hàng không cấu trúc khi các liên kết giữa các nút mạng trong mạng phủ được thiết lập ngẫu nhiên. Hệ thống mạng ngang hàng không cấu trúc thể hiện nhược điểm là không đảm bảo quá trình tìm kiếm sẽ thành công. Đối với tìm kiếm các dữ liệu phổ biến được chia sẻ trên nhiều máy, tỉ lệ thành công là khá cao, ngược lại, nếu dữ liệu chỉ được chia sẻ trên một vài máy thì xác suất tìm thấy là khá nhỏ.

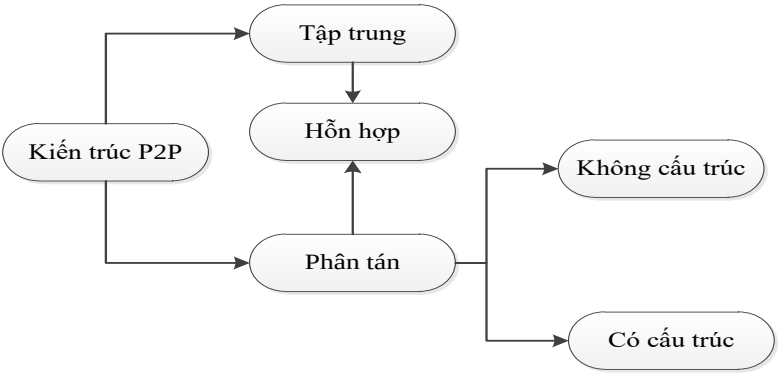
Mạng ngang hàng có cấu trúc khắc phục nhược điểm của mạng không cấu trúc bằng cách sử dụng hệ thống liên kết giữa các nút mạng trong mạng phủ theo một thuật toán cụ thể, đồng thời xác định chặt chẽ mỗi nút mạng sẽ chịu trách nhiệm đối với một phần dữ liệu chia sẻ trong mạng. Với cấu trúc này, khi một máy cần tìm một dữ liệu, nó chỉ cần áp dụng một giao thức chung để xác định nút mạng nào chịu trách nhiệm cho dữ liệu đó và sau đó liên lạc trực tiếp đến nút mạng đó để lấy kết quả.

Với những ưu điểm của mạng ngang hàng có cấu trúc, đã có rất nhiều giao thức được đưa ra để xử lý cho những bài toán cụ thể, một số giao thức được áp dụng như Chord, CAN, Kademlia, Tapestry, Kelips, mặc dù vậy trong quá trình hoạt động của mạng ngang hàng có cấu trúc nhiều vấn đề chưa được giải quyết như đảm bảo việc phục hồi dữ liệu trong mạng khi các node trong mạng thường xuyên gia nhập hoặc rời khỏi mạng, cân bằng tải giữa các node vẫn chưa xử lý được nhiều, mở rộng phạm vi hoạt động của mạng nhưng vẫn đảm bảo bảo mật của dữ liệu vẫn chưa khắc phục được triệt để, luận văn "Giải pháp backup dữ liệu, sử dụng cơ chế phân cụm động trong mạng ngang hàng có cấu trúc" sẽ đưa ra một số phương pháp mới đảm bảo việc backup dữ liệu và khắc phục các vấn đề nêu trên.

CHƯƠNG 1: TỔNG QUAN VỀ KIẾN TRÚC HỆ THỐNG MẠNG NGANG HÀNG

Trong chương này sẽ giới thiệu một số kiến trúc hệ thống mạng mang hàng, mô tả các đặc điểm chung, các thuộc tính và một số hệ thống áp dụng cho mỗi kiến trúc đưa ra.

Nhìn chung, mạng Ngang hàng được phân thành hai hệ thống chính là hệ thống tập trung và hệ thống phân tán dựa trên tính sẵn sàng của một hay nhiều server, bên cạnh đó còn có hệ thống hỗn hợp là hệ thống vừa có những đặc điểm của hệ thống tập trung và hệ thống phân tán. Các nội dung tiếp theo sẽ mô tả chi tiết cho từng hệ thống này.

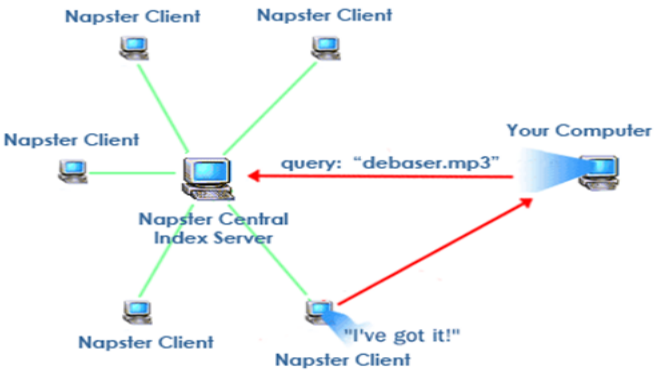


Hình 1-1 Phân loại kiến trúc P2P

1.1 HỆ THỐNG P2P TẬP TRUNG

Trong hệ thống P2P tập trung, có một hay nhiều server giúp cho các peer xác định vị trí tài nguyên mong muốn hoặc phối hợp các hoạt động giữa các peer với nhau. Để định vị tài nguyên, một peer gửi thông điệp tới server trung tâm để xác định địa chỉ peer mà chứa tài nguyên mong muốn. Khi xác định được peer có thông tin hay dữ liệu, nó có thể liên kết trực tiếp với các peer đó để trao đổi thông tin mà không qua server nữa [1].

Kiến trúc hệ thống tập trung này dễ bị tấn công vào liên kết đến server, mặt khác nó còn là nút thắt cổ chai đối với hệ thống có số peer lớn, tiềm ẩn việc làm giảm hiệu năng một cách đột ngột, ngoài ra mô hình này hạn chế khả năng mở rộng, điển hình của mô hình này là Napster [16].



Hình 1-2 Mô hình mạng Napster

Mô hình cho thấy có một server trung tâm duy trì siêu dữ liệu (metadata) của file hoặc đối tượng chia sẻ bởi các peer trong mạng. Metadata này có thể xem như là cặp (dataID, PeerID), bất kỳ truy vấn nào trước tiên đều kết nối trực tiếp đến server tung tâm, server trung tâm trả lại danh sách các node chứa các đối tượng/dữ liệu yêu cầu. Sau đó truy vấn khởi tạo kết nối trực tiếp tới những node chứa đối tượng/dữ liệu mà không thông qua server nữa.

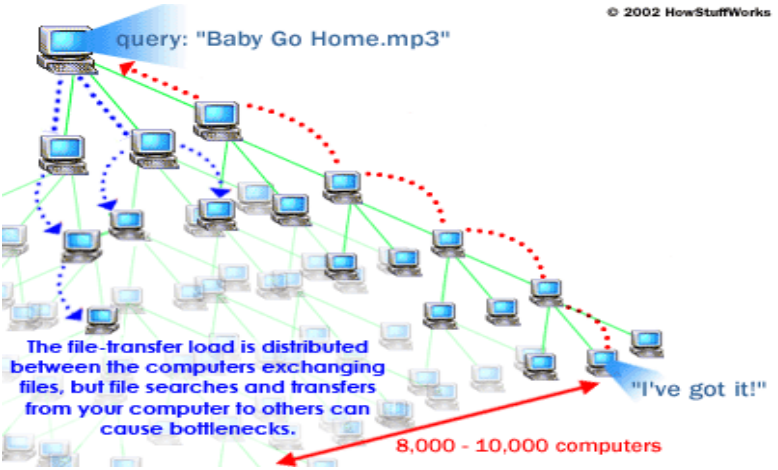
1.2 HỆ THỐNG P2P PHÂN TÁN ĐẦY ĐỦ

Trong hệ thống phân tán các peer có quyền và trách nhiệm như nhau. Mỗi peer chỉ có thông tin một phần trong mạng và yêu cầu dữ liệu hay dịch vụ thông qua một số peer khác[2]. Như vậy việc xác định các peer yêu cầu dữ liệu hay dịch vụ nhanh là một vấn đề và thách thức đối với hệ thống này. Hệ thống P2P phân tán được chia thành hai hệ thống là hệ thống P2P phân tán không cấu trúc và hệ thống P2P phân tán có cấu trúc, khác nhau giữa hai hệ thống này là phương pháp các truy vấn chuyển đến các node.

1.2.1 Hệ thống P2P không cấu trúc - Gnutella

Gnutella là hệ thống phân tán thuần túy, không có node trung tâm chịu trách nhiệm tổ chức mạng và không phân biệt giữa client và server [1,12]. Các node trong hệ thống kết nối với nhau thông qua một phần mềm

ứng dụng cụ thể. Mạng Gnutella được mở rộng khi node mới tham gia vào mạng và bị thu hẹp khi các node rời mạng. Hoạt động cơ bản của Gnutella bao gồm việc tham gia, rời mạng, tìm kiếm và tải các file.

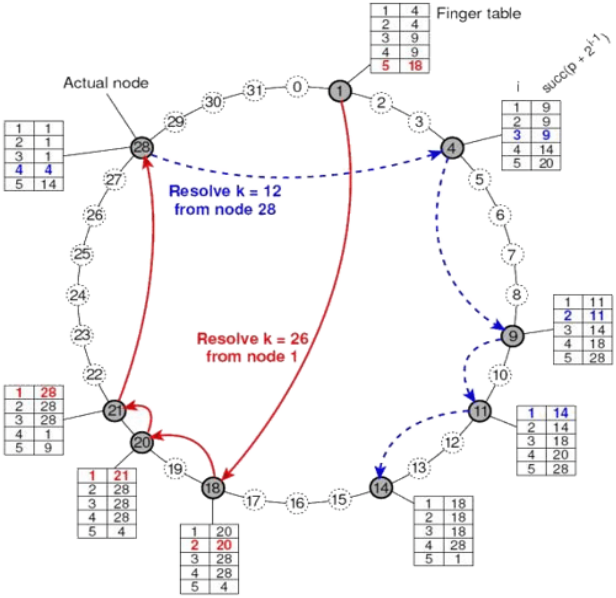


Hình 1-3 Mô hình trao đổi và tìm kiếm thông tin trong Gnutella

1.2.2 Hệ thống P2P có cấu trúc - Giao thức Chord

Chord là một giao thức tìm kiếm phân tán sử dụng mô hình dạng vòng để kết nối các node với nhau. Giao thức này nằm trong hệ thống phân tán có cấu trúc, sử dụng bảng băm phân tán DHT để xác định các cặp khóa (key, value) phục vụ cho việc tra cứu, tìm kiếm trong mạng [8]. Hình 1-4 mô tả các node được xếp thành hình vòng tròn và sơ đồ kết nối giữa các node với nhau trong mạng Chord.

Chord được biểu diễn dưới dạng vòng tròn, với vòng tròn có N bít sẽ có 2n không gian định danh, mỗi node có một node liền trước (successor) và 1 node liền sau (predecessor), các node định tuyến cho nhau thông qua bảng định tyến (finger table). Mỗi dòng trong bảng định tuyến sẽ lưu thông tin một node ở xa gọi là entry. Bảng định tuyến được xác định dựa trên số bit đưa vào hệ thống, với n bit sẽ có n entry trong bảng định tuyến.



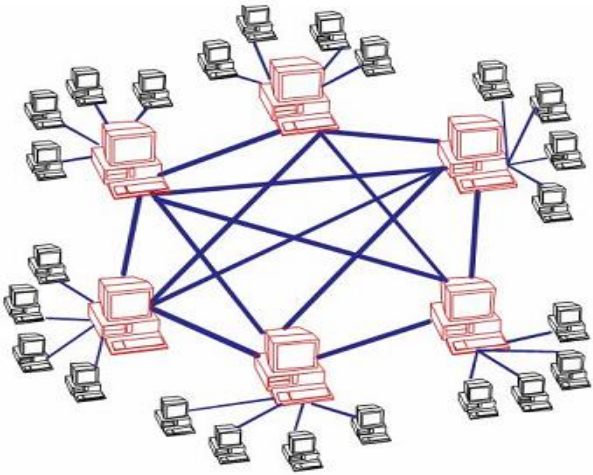
Hình 1-4 Mô hình mạng sử dụng giao thức Chord

1.3 HỆ THỐNG P2P HỖN HỢP

Hệ thống P2P hỗn hợp tận dụng được các ưu điểm so với Hệ thống phân tán đầy đủ và hệ thống tập trung. Trong hệ thống này có một vài peer xử lý nhiều chức năng hơn và chịu trách nhiệm nhiều hơn các peer khác còn gọi là supernode, những node này được thành lập ở mức cao trong hệ thống hỗn hợp, các peer này như là các server trung tâm để cung cấp các dịch vụ cho các peer khác [13]. Mặc dù các supernode chia sẻ một vài chức năng tới server trung tâm nhưng nó cũng có một vài điểm khác biệt như sau:

- Một supernode không mạnh bằng server trung tâm và chỉ chịu trách nhiệm tập hợp các peer trong mạng.
- Một server như mô hình Napster giúp đỡ các peer để định vị các file mà không chia sẻ file, tuy nhiên một supernode không chỉ

phối hợp các hoạt động trong peer mà chính nó còn thực hiện các hoạt động tương tự và đóng góp tài nguyên của nó như là các peer thông thường khác.



Hình 1-5 Mô hình mạng hỗn hợp

CHƯƠNG 2: CÁC PHƯƠNG PHÁP BACKUP DỮ LIỆU TRÊN MẠNG NGANG HÀNG CÓ CẤU TRÚC

Trong chương này mô tả hai phương pháp backup khác nhau, đó là phương pháp successor list và phân cụm tĩnh, sử dụng giao thức Chord. Giao thức Chord là giao thức chính được sử dụng cho nghiên cứu, cải tiến của luận văn thông qua việc mở rộng phân cụm động và phân cụm tĩnh so với phương pháp Chord nguyên thủy (successor list).

2.1 CƠ CHẾ BACKUP THEO SUCCESSOR LIST

Backup theo successor list là phương pháp backup nguyên thủy trong mạng Chord. Dựa trên bảng băm phân tán DHT, sử dụng mã xóa IDA (information dispersal algorithm) [9], nhằm phân tán và lưu trữ khối với bảng băm.

DHT là một hàm băm được cài đặt như một hệ thống phân tán. Cũng như một hàm băm thông thường, DHT cung cấp ánh xạ từ key đến value. Điểm khác của DHT so với hàm băm thông thường là các value trong một DHT được lưu đến các node khác nhau trong mạng, chứ không phải lưu trong một cấu trúc dữ liệu cục bộ. Nhờ khả năng phân tán làm cho DHT trở nên mạnh mẽ, hiệu quả và đáp ứng được những ứng dụng thực tế. Các ứng dụng này yêu cầu DHT duy trì tính sẵn sàng của dữ liệu ngay cả khi gặp lỗi và hiệu quả trong trường hợp xử lý một khối dữ liệu lớn.

Theo phương pháp mã xóa, bảng băm chia khối dữ liệu (file dữ liệu) ra làm f mảnh, trong đó với k mảnh là có thể khôi phục lại được khối dữ liệu. Các mảnh dữ liệu ở đây là riêng biệt nhau, chứa thông tin độc lập và duy nhất. Chẳng hạn, khối dữ liệu chia ra làm 14 mảnh, nhưng với 7 mảnh dữ liệu có thể khôi phục lại khối dữ liệu 14 mảnh. Để duy trì các mảnh dữ liệu luôn đảm bảo có thể khôi phục được khối dữ liệu ban đầu, DHT chuyển đổi các mảnh giữa các node khi các node tham gia hoặc rời mạng.

- **Đảm bảo tính sẵn sàng khối dữ liệu**

Giống như khả năng chịu lỗi của nhiều hệ thống lưu trữ khác, bảng băm sử dụng mã xóa để làm tăng tính sẵn sàng với chi phí thấp.

Khi thêm khối dữ liệu: put (k,b) [6]

Khi một ứng dụng muốn thêm một khối dữ liệu mới, nó gọi hàm băm put(k,b) thực hiện như sau:

```
Void put (k,f) // đặt một mảnh vào mỗi successor
{
Frags=IDAencode (f)
Succs=lookup (k,14)
For i (0…13)
Send (succs[i].ipaddr, k, frags[i])
}
```

- **Lấy khối dữ liệu: get(k)**

Để lấy khối dữ liệu, một client phải định vị và truy hồi dủ các mảnh theo thuật toán phân mảnh thông tin IDA để lắp ghép lại thành khối dữ liệu ban đầu. Khi một ứng dụng client gọi get(k) bảng băm tại client trước tiên khởi tạo việc tìm kiếm qua hàm lookup(k,7) để tìm danh sách các node có khả năng lưu trữ các mảnh của khối dữ liệu. Kết quả tìm kiếm sẽ trả về danh sách từ 7 đến 14 node successor trực tiếp của khóa k.

Sau đó get() chọn 7 successor với độ trễ thấp nhất để thiết lập đồng bộ, gửi mỗi node một RPC (remote procedure call) để yêu cầu một mảnh của khóa k theo phương pháp đồng bộ song song. Với mỗi RPC quá thời gian cho phép (time out) hoặc bị lỗi, get() gửi một mảnh yêu cầu RPC để kết nối lại với danh sách các successor tìm thấy qua hàm lookup() mà chưa kết nối để thiết lập kết nối lại [6].

Trong trường hợp gọi hàm lookup() nhưng kết quả trả về ít hơn 7 successor chứa các mảnh, get() hỏi một trong successor của nó tìm kiếm mở rộng thêm các node khác để tạo lại khối dữ liệu. Nếu không thể xây

dựng lại được khối dữ liệu sau quá trình trao đổi, tìm kiếm trên hệ thống, get() trả lại kết quả không thành công.

Một ứng dụng có thể gọi hàm get(k) nhiều lần để lấy khóa cho sẵn. Khi các node tham gia hoặc rời hệ thống, các mảnh cần phải chuyển đến các successor node của nó. Nếu tỷ lệ tham gia, rời mạng của các node tăng cao có thể dẫn đến mảnh dữ liệu bị sai vị trí và dẫn đến việc không lấy được mảnh bị mất. Để khắc phục việc này, bảng băm đưa ra cơ chế duy trì mảnh dữ liệu để phục hồi lại các mảnh đã bị mất.

- **Duy trì mảnh dữ liệu**

Trạng thái lý tưởng khi tồn tại đủ số mảnh của khối dữ liệu, tuy nhiên các node tham gia và rời mạng liên tục dẫn tới bị lỗi ở một số node làm cho các mảnh bị mất hoặc bị đặt sai vị trí. Để duy trì trạng thái lý tưởng, bảng băm sử dụng hai giao thức là giao thức duy trì cục bộ và giao thức duy trì toàn cục.

Giao thức duy trì cục bộ phục hồi số mảnh còn thiếu, còn giao thức toàn cục di chuyển các mảnh đặt sai vị trí vào vào đúng số node, khôi phục lại vị trí, ngoài ra nó cũng có chức năng phục hồi các mảnh đã bị mất, xóa đi các mảnh dữ liệu dư thừa trong hệ thống.

2.2 PHÂN CỤM TĨNH TRONG MẠNG CHORD

2.2.1 Phương pháp tách cụm tĩnh

Ý tưởng chính của phương pháp là chia mạng Chord thành một số cụm với không gian ID mỗi cụm bằng nhau dựa vào bảng băm phân tán DHT, mỗi cụm sẽ lưu trữ cục bộ và thực hiện duy trì dữ liệu trong cụm, đảm bảo dữ liệu luôn sẵn sàng ngay cả khi các node tham gia hoặc rời mạng.

Trong mỗi cụm đưa ra một số node có khả năng lưu trữ với dung lượng lớn để đảm bảo việc backup dữ liệu luôn được cân bằng tải giữa các node trong một cụm.

2.2.2 Phương pháp backup file

Để duy trì tính sẵn sàng của file dữ liệu ngay cả có sự vào ra của các node trong mạng, mỗi file được mã hóa thành n mảnh sử dụng hình thức mã xóa [14], các mảnh này được lưu ở một số node trong mạng. Đặc trưng của mã xóa là với k mảnh (k<n) dữ liệu file được tập hợp thì có thể khôi phục lại file ban đầu. Ở đây k và n được định nghĩa trước trong hệ thống. Nội dung phương pháp sao lưu dựa trên việc phân cụm là các node trong cùng một cụm sẽ lưu trữ các mảnh dữ liệu của file để đảm bảo trong một cụm luôn có số mảnh của một file lớn hơn k nhằm duy trì và phục hồi lại dữ liệu file[4].

Quản lý thông tin cụm

Không gian khóa DHT được chia thành m phần bằng nhau (m cụm), biên của cụm thứ k sẽ được lưu ở node đầu và node cuối cụm, các node có định danh ID nằm ở giữa định danh đầu cụm và định danh cuối cụm thì thuộc cụm đó. Như vậy trong mỗi cụm có một node đầu cụm, một node cuối cụm. Node cuối cụm này nhưng cũng là đầu cụm kế tiếp

Truy vấn và sao lưu dữ liệu

Trong phương pháp backup được đưa ra, một node chịu trách nhiệm về khóa của một file DHT sẽ quản lý tính sẵn có của file đó (quản lý file đó còn tồn tại hay không và thông tin lưu trữ các mảnh của file). Khóa của một file DHT là khóa duy nhất sinh ra từ việc băm nội dung của file và được sử dụng để truy vấn phục hồi một file

Duy trì tính ổn định file

Khi một node rời mạng chủ động, nó sẽ chuyển dữ liệu lưu trữ của nó cho successor node và gửi thông tin thông báo tới node đầu cụm về trạng thái rời mạng. Tuy nhiên, nếu một node rời mạng do bị lỗi (rời mạng đột ngột), dữ liệu bao gồm mảnh dữ liệu và thông tin về các file sao lưu được lưu trong node đó sẽ bị mất, trong trường hợp này chúng ta cần duy trì ít nhất k mảnh của bất kỳ file nào trong mạng để đảm bảo file luôn sẵn sàng.

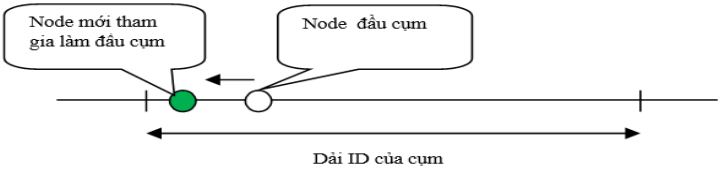
CHƯƠNG 3: PHƯƠNG PHÁP PHÂN CỤM ĐỘNG VÀ CƠ CHẾ BACKUP.

Ý tưởng chính của phương pháp này là chia cụm và giới hạn số node trong một cụm để đảm bảo quá trình backup định kỳ thường xuyên và ổn định hơn. Bên cạnh đó, cũng xử lý các cụm liền kề có số node trong mỗi cụm nhỏ có thể nhập cụm lại nhằm đảm bảo hai vấn đề:

- Cân bằng tải cho các node: Việc nhập các cụm sẽ đảm bảo các cụm luôn ổn định số node trong một cụm, trong khi mỗi cụm lại có cơ chế chọn các node tốt nhất để phục vụ việc backup dữ liệu.
- Giảm số lượng cụm trong một mạng khi quá trình tách cụm liên tục tạo ra nhiều cụm, từ đó cân bằng số node trong cụm và làm cho cân bằng tải trong cụm tốt hơn, thỉ lệ phục hồi file thành công cao hơn.

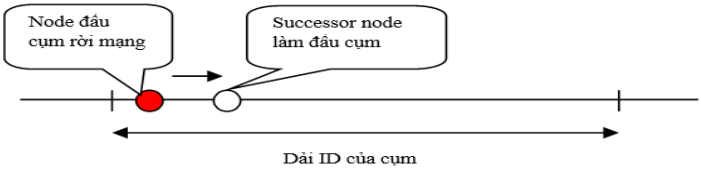
3.1 PHƯƠNG PHÁP TÁCH NHẬP CỤM

Khi một node tham gia vào hệ thống, nó sẽ thông báo cho successor node. Successor node sẽ thông báo cho node đầu cụm. Node đầu cụm sẽ tăng tổng số node trong cụm lên 1 đơn vị



Hình 3-1 Mô tả việc tham gia một node vào hệ thống

Khi một node rời mạng, nó sẽ thông báo cho successor node. Successor node sẽ thông báo cho node đầu cụm. Node đầu cụm sẽ giảm tổng số node trong cụm xuống 1 đơn vị.

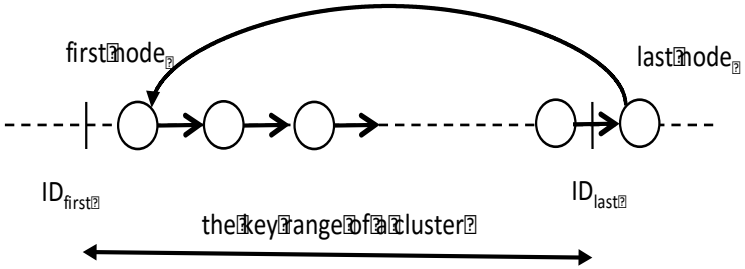


Hình 3-2 Mô tả một node rời hệ thống

Mục đích của việc tính tăng, giảm ở node đầu cụm nhằm xem xét tổng số lượng node trong cụm để thực hiện tách cụm hay nhập cụm

Định kỳ các cụm cập nhật lại thông tin trong cụm. Node đầu cụm sau khi cập nhật số lượng các node trong cụm để xem xét các bước sau:

- Có tách tách/nhập cụm không?
- Sau khi tách/nhập cụm, node đầu cụm thực hiện cập nhật tới successor node, cứ như vậy đến node cuối cụm. Sau đó node cuối nhánh cập nhật lại cho node đầu cụm.



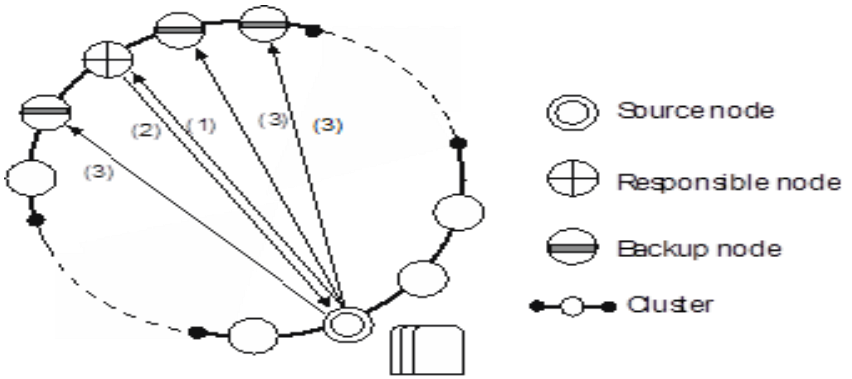
Hình 3-3 Quá trình cập nhật dữ liệu trong một cụm

3.2 PHÂN MẢNH KHI ĐƯA MỘT FILE MỚI VÀO MẠNG

Mặc định mỗi file khi được đưa vào mạng sẽ được chia thành k mảnh để phục vụ cho việc backup file.

Đối với 1 file mới đưa vào mạng Chord. File đưa vào dựa theo thuật toán DHT băm nội dung của file thành key, key được sinh ra sẽ được successor node quản lý, từ đó biết được cụm nào và danh sách các node tốt nhất, tiếp theo successor node sẽ thông báo cho node lưu trữ file gốc thực hiện quá trình backup các mảnh vào các node tốt nhất.

Như vậy key của một file vừa có thông tin của file gốc vừa có thông tin các mảnh. Việc tìm kiếm dữ liệu thông qua các key, từ key truy vấn tới các mảnh và trả lại thông tin tìm kiếm [11, 12]. File gốc được sử dụng trong trường hợp các mảnh còn lại không đủ số lượng để phục hồi lại file gốc, khi đó sử dụng file gốc ban đầu để tạo thêm các mảnh mới.



Hình 3-4 Quá trình backup và phân mảnh một file mới đưa vào mạng

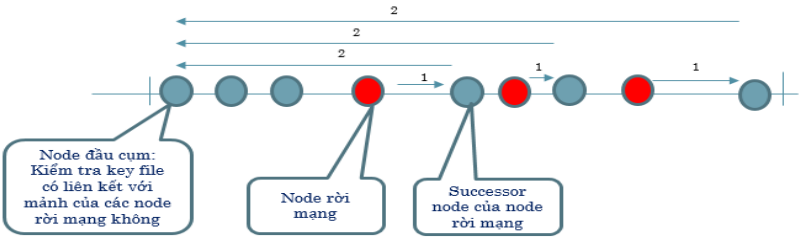
3.3 BACKUP KHI MỘT NODE RỜI MẠNG

Khi 1 node rời khỏi mạng Chord, nó sẽ thực hiện các bước sau:

1. Thông báo cho successor node tình trạng rời mạng.

2. Successor node thông báo cho node đầu cụm thông tin ID node rời mạng.

Định kỳ, node đầu cụm tập hợp danh sách các node rời mạng và thông báo cho các node trong hệ thống các node trong cụm đã rời mạng



Hình 3-1 Quá trình các node rời mạng và cập nhật thông tin

Mỗi khi một node nhận được thông tin về danh sách các node rời mạng trong cụm, nó thực hiện kiểm tra lần lượt các key của file mà nó quản lý để kiểm tra các lại các mảnh dữ liệu mà key quản lý.

Trường hợp các key của file kiểm tra thấy số lượng các mảnh còn lại nhỏ hơn giá trị ngưỡng các mảnh (k mảnh) có thể phục hồi lại file, node chịu trách nhiệm quản lý key sẽ thực hiện backup lại các mảnh đã mất.

Trường hợp các key của file kiểm tra thấy tổng số các mảnh còn lại không có khả năng phục hồi lại file gốc, node chịu trách nhiệm quản lý key sẽ tìm lại node chứa file gốc để backup lại các mảnh. Nếu node chứa file gốc bị bị rời mạng thì không backup được các mảnh, đồng thời nó sẽ thông báo các node lưu trữ các mảnh xóa các mảnh đó trong hệ thống.

CHƯƠNG 4: ĐÁNH GIÁ HIỆU QUẢ PHƯƠNG PHÁP TÁCH NHẬP CỤM SỬ DỤNG CƠ CHẾ PHÂN CỤM ĐỘNG

4.1 CHƯƠNG TRÌNH MÔ PHỎNG

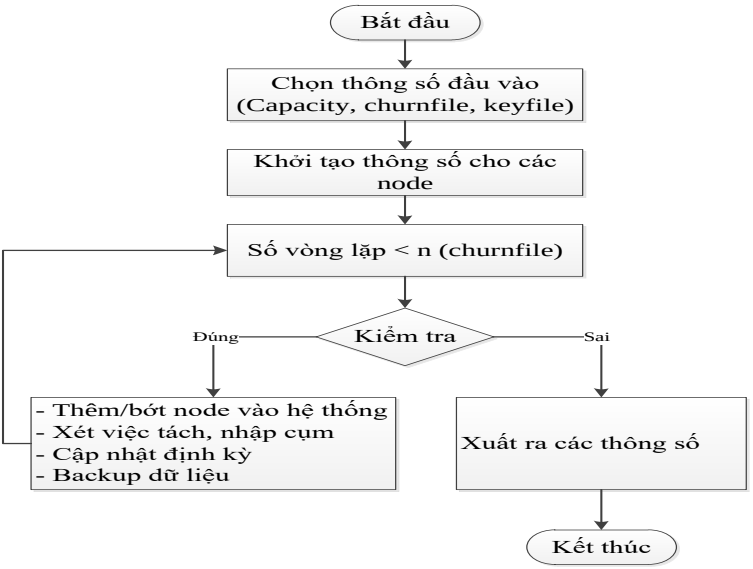
Chương trình mô phỏng phân cụm động được mở rộng từ chương trình mô phỏng của Jonathan Ledlie [14] và được xây dựng trên ngôn ngữ Microsoft Visual studio C++;

Một số thông số sử dụng mô tả hoạt động của chương trình

- Dung lượng (capacity) của một node: Trung bình một node có khả năng chứa 120 mảnh dữ liệu, mỗi file tạo ra một key file quản lý 6 mảnh dữ liệu. Trong chương trình mô phỏng tạo ra file đầu vào chứa key file là 5%, 10%, 15%, 20%, 30% có ngĩa là chương trình phân bổ các key file cho cả không gian ID của mạng (4096 ID) đồng đều cho mỗi ID 5 key file.
- Thời gian sống của một node: Được sinh ra khi tạo file chứa các node tham gia hoặc rời mạng khi tạo file pareto.

Chương trình mô phỏng

Hoạt động chương trình mô phỏng theo lưu đồ sau:



Hình 4-1 Lưu đồ chạy chương trình mô phỏng

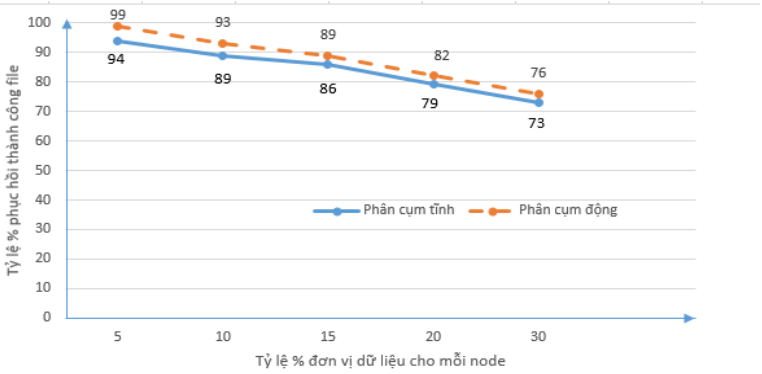
Một số điểm phân biệt giữa chương trình mô phỏng phân cụm động và phân cụm tĩnh

Phân cụm tĩnh	Phân cụm động
Khởi tạo ban đầu với số cụm xác định	Khởi tạo ban đầu với chỉ 1 cụm
Số node trong một cụm không giới hạn	Giới hạn số node trong một cụm
Các mảnh dữ liệu luôn nằm trong một cụm	Các mảnh dữ liệu có thể nằm ở nhiều cụm khác nhau
Số lượng cụm cố định	Số lượng cụm thay đổi tùy thuộc vào số node tham gia hoặc rời mạng

Bảng 4-1: So sánh sự khác nhau giữa phân cụm tĩnh và phân cụm động

4.2 ĐÁNH GIÁ VÀ SO SÁNH MỘT SỐ THÔNG SỐ CỦA PHƯƠNG PHÁP TÁCH NHẬP CỤM THEO CƠ CHẾ PHÂN CỤM ĐỘNG SO VỚI PHÂN CỤM TĨNH.

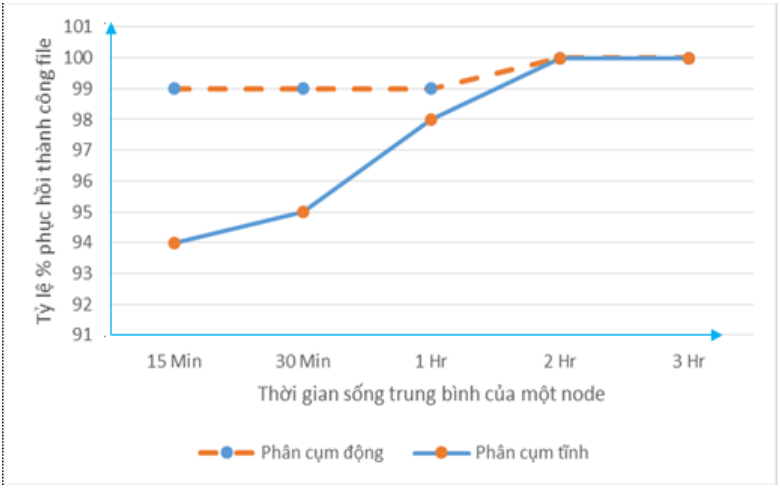
4.2.1 Tỷ lệ khôi phục file ban đầu thành công (khi cố định thời gian sống 1 node và tăng số file)



Biểu đồ 4-1 So sánh tỷ lệ khôi phục file ban đầu thành công giữa phân cụm tĩnh và phân cụm động

Từ kết quả của chương trình cho thấy khi dữ liệu đưa vào node tăng lên thì tỷ lệ truy vấn thành công giảm theo, với 5 đơn vị dữ liệu đưa vào một node tỷ lệ thành công là 99% và thấp dần xuống 76% khi dữ liệu đưa vào là 30 đơn vị. Kết quả này cũng cho thấy tỷ lệ thành công của phương pháp phân cụm động cao hơn so với phân cụm tĩnh, trung bình khoảng 3% do thời gian cập nhật thông tin trong một cụm nhanh hơn nên quá trình backup tốt hơn và tỷ lệ truy vấn thành công cao hơn.

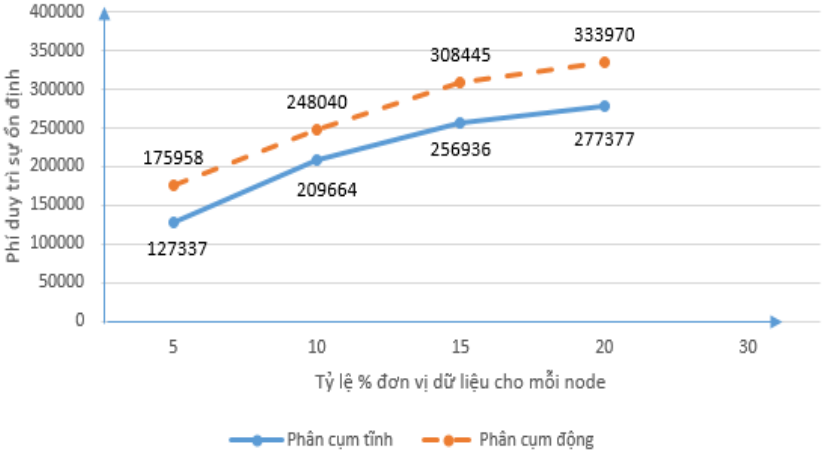
4.2.2 Tỷ lệ file ban đầu thành công (cố định số lượng file và thay đổi thời gian sống)



Biểu đồ 4-2 So tỷ lệ file ban đầu thành công giữa phân cụm tĩnh và phân cụm động khi thay đổi thời gian sống của một node.

Kết quả trên biểu đồ 4-2 cho thấy, thời gian sống của một node càng lâu, tỉ lệ rời mạng của các node trong cụm thấp hơn, các mảnh dữ liệu bị phân tán ra các cụm ít hơn dẫn tới tỷ lệ tìm thấy các mảnh trong cụm cao hơn và tỷ lệ phục hồi thành công file cao. Tỷ lệ phục hồi thành công file của phương pháp phân cụm tĩnh thấp hơn so với phân cụm động do quá trình cập nhật các node tốt nhất trong cụm của phân cụm động nhanh hơn nên backup được nhiều mảnh đã mất cho các node tốt nhất, dẫn tới tỷ lệ truy vấn thành công cao hơn, tỷ lệ phục hồi thành công file cao hơn. Trường hợp thời gian sống của một node từ 2 giờ trở lên, tỷ lệ phục hồi thành công file của cả phân cụm tĩnh và phân cụm động là 100%.

4.2.3 Chi phí cho việc duy trì các mảnh là bao nhiêu.

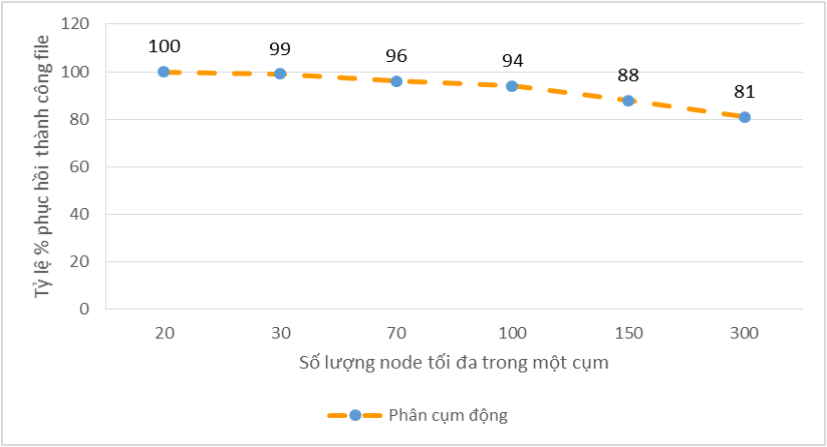


Biểu đồ 4-3 So sánh chi phí duy trì các mảnh giữa phân cụm tĩnh và phân cụm động.

Kết quả trên biểu đồ chỉ ra rằng chi phí để backup dữ liệu cho các mảnh của phương pháp phân cụm động cao hơn phân cụm tĩnh, điều này được giải thích như sau:

- Phương pháp phân cụm động được phân chia thành nhiều cụm trong quá trình chạy, đo đó các mảnh dữ liệu có thể nằm ở nhiều cụm khác nhau, việc định kỳ backup dữ liệu giữa các cụm với nhau mất nhiều thời gian do thông tin các cụm cập nhập cho nhau chậm hơn thông tin cập nhật trong một cụm dẫn tới một số mảnh dữ liệu không có thông tin. Khi khôi phục lại file ban đầu phải mất chi phí để phục hồi các mảnh này nên tốn chi phí hơn
- Theo phương pháp phân cụm tĩnh dữ liệu luôn nằm ở các node trong cụm do đó chi phí để tìm thấy và phục hồi các dữ liệu thấp hơn so với phân cụm động.

4.2.4 So sánh file ban đầu thành công khi thay đổi số lượng node trong cụm



Biểu đồ 4-4 Tỷ lệ phục hồi công file khi thay đổi số lượng node tách, nhập trong một cụm

Thí nghiệm mô phỏng trong trường hợp thay đổi số lượng node tách, nhập cụm cho thấy khi số node tách, nhập cụm thấp tỷ lệ phục hồi thành công file cao hơn so với số node tách, nhập cụm lớn. Điều này chứng tỏ với cụm có số lượng node nhỏ việc cập nhật định kỳ nhanh hơn so với cụm có số lượng node lớn, từ đó việc phục hồi file và các mảnh dữ liệu nhanh hơn dẫn tới tỷ lệ phục hồi thành công file cao hơn.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Thông qua việc nghiên cứu về backup dữ liệu theo cơ chế phân cụm động phần nào cũng cho thấy những ưu điểm và linh hoạt trong mạng ngang hàng có cấu trúc sử dụng thông qua giao thức Chord. Nhìn chung cơ chế phân cụm tĩnh phù hợp với những mạng được ước lượng trước số node tham gia hoặc rời mạng trong hệ thống, qua đó việc chia cụm cố định sẽ phù hợp để đảm bảo cả việc backup, thời gian backup cũng như duy trì các mảnh dữ liệu.

Với cơ chế phân cụm động, không phụ thuộc vào số lượng các node tham gia hoặc rời mạng, thời gian backup được ổn định, tỷ lệ khôi phục thành công file dữ liệu cao hơn nhưng chi phí duy trì thì tốn hơn, đòi hỏi những node tham gia gia mạng với cấu hình cao hơn để tăng thời gian xử lý backup.

Mặc dù đã đạt được một số kết quả cho thấy ở trên tuy nhiên việc mô phỏng này vẫn còn một số hạn chế cần được bổ sung, nghiên cứu thêm để phù hợp với thực tế như: tính đến khoảng cách của các node khi tham gia vào hệ thống, từ việc xác định được khoảng cách các node tham gia vào hệ thống sẽ phân bổ vào các cụm hợp lý hơn nhằm giảm tải cho việc duy trì dữ liệu

Trong tương lai có thể mở rộng nội dung của luận văn thông qua việc tính khoảng cách các node khi tham gia vào cụm.

TÀI LIỆU THAM KHẢO

Tiếng Việt

[1] Nguyễn Hoài Sơn, Hồ Sĩ Đàm (2008), "*Tìm kiếm thông tin theo các giá trị thuộc tính trên mạng ngang hàng có cấu trúc*", Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội.

[2] Nguyễn Đại Thọ (2007), "*Công nghệ mạng ngang hàng*", Bộ môn Mạng & Truyền thông Máy tính Khoa Công nghệ Thông tin, trường Đại học Công nghệ - Đại học Quốc gia Hà Nội.

[3] Ngô Hoàng Giang (2008), *"Đánh giá hiệu năng của một số thuật toán bảng băm phân tán DHT và đưa ra giải pháp cải tiến hiệu năng của thuật toán Chord",* luận văn thạc sỹ Công nghệ thông tin trường đại học Bách khoa Hà Nội.

Tiếng Anh

[4] Nguyen Dinh Nghia, Nguyen Hoai Son (2016), "A Cluster-based File Replication Scheme for DHT-based File Backup Systems", "*Advanced Technologies for Communications (ATC), 2016 International Conference on",* ISSN: 2162-1039, No 16520217.

[5] Kale A.R and SHIRBHATE D.D (Mar 2012), "An advanced hybrid peer to peer botnet", "*International Journal of wireless Comunication*", ISSN: 2231-3559, Vol.2.

[6] John cates (2003), *"Robust and Efficient Data Management for Distributed Hash table",* Submitted to Departmant of Electrical and computer science - Massachusetts institute of technology, USA.

[7] IonStoca RobMorris, David Karger, M.Frans Kaashoek, Hari Balakrishnan (2001), "Chord: A Ccalable peer-to-peer lookup service for internet Applications", *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*", ISBN:1-58113-411-8, Vol.31.

[8] L. Garc´es-Erice, P.A. Felber, E.W. Biersack, G. Urvoy-Keller K.W. Ross (March 2004), "Data Indexing in Peer-to-Peer DHT Networks", *"Proceedings of the 24th International Conference on Distributed Computing Systems",* ISBN: 0-7695-2086-3.

[9] Micheael Rabin (April 1989), "Efficient dispersal of information for security, load balancing, and fault tolerance", "*Journal of the Association for Computing Machinery*", Vol. 36, No.2

[10] Sameh El-Alsary and Seif Haridi (July 2004), "*An overview of structured P2P overlay network*", Swedish Institute of Computer Science, Swedish.

[11] S. Legtchanko, P. Sen, Cilles Muller (April 2009), *"*Churn-resilent replecation strategе for peer to peer distributed hash-tables*"*, "*Proceedings of the 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems*", ISBN: 978-3-642-05117-3, No 6897.

[12] S. Ratnasamy, P. Francis, M. Handley and R. Karp, (Aug. 2001), "A Scalable Content-Addressable Network", *"Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications"*, ISBN:1-58113-411-8, Vol.31.

[13] Alberto Montresor (Arp 2016), "*Distributed algorithms peer to peer system*", University of trento, Italy.

[14] J.Ledlie, M.Seltzer. (Mar 2005), "Distributed, Secure Load Balancing with Snew, Heterogeneity and Churrn", *"In Proceedings of 24th Annual Joint Conference of the IEEE Computer and Comunications Societies"*, ISSN: 0743-166X, Vol.2.

[15] Christos Gkantsidis, Milena Mihail (Mar 2004),*"Random walks in peer to peer networks", "Performance Evaluation - P2P computing systems"* ISSN: 0743-166X, No 8410756.

[16] Vu Q.H, LuLu M, Ooi P.C (2010), *"Peer to peer computing principles and applications",* Spinger 2010, XVI, 317 p. ISPN 978-3-642-03513-5.