Chương 1: Tổng quan về mạng máy tính

Mục đích

Chương này nhằm giới thiệu cho người học những nội dung sau:

- Các loại mạng truyền dữ liệu đã tồn tại trước khi của mạng máy tính ra đời.
- Cấu trúc tổng quát của một mạng máy tính.
- Hai chế độ truyền tải dữ liệu cơ bản là Chuyển mạch và Chuyển gói.
- Những lợi ích mà mạng máy tính mang lại.

Yêu cầu

Sau khi học xong chương này, người học phải có được những khả năng sau:

Phân biệt được hai loại mạng Chuyển mạch và Chuyển gói;
Định nghĩa được mạng máy tính là gì và trình bày được cấu trúc tổng quát của một mạng máy tính.
Nêu lên được những lợi ích mà mạng máy tính mang lại.

1.1 Mạng điện báo

Mạng điện báo sử dụng hệ thống mã Morse để mã hóa thông tin cần truyền đi. Mã Morse sử dụng hai tín hiệu là tít và te (ký hiệu bằng dấu chấm (•) và dấu gạch ngang (-)). Mỗi một ký tự latin sẽ được mã hóa bằng một chuỗi tíc/te riêng biệt, có độ dài ngắn khác nhau. Để truyền thông tin đi, bên gởi sẽ lần lượt mã hóa từng ký tự của thông điệp thành mã Morse, bên nhận sau đó sẽ thực hiện quá trình giải mã. Văn bản được truyền đi được gọi là một thông điệp (message) hay một thư tín (Telegram).

Vào năm 1851 mạng thư tín đầu tiên được sử dụng để nối hai thành phố London và Paris. Sau đó không lâu, hệ thống mạng này được mở rộng toàn châu Âu.

Cấu trúc của mạng gồm có hai thành phần là Trạm điện báo (Telegraph Station) và Trạm chuyển điện báo (Telegraph Switching Station) được nối lại với nhau bằng hệ thống dây truyền dẫn.

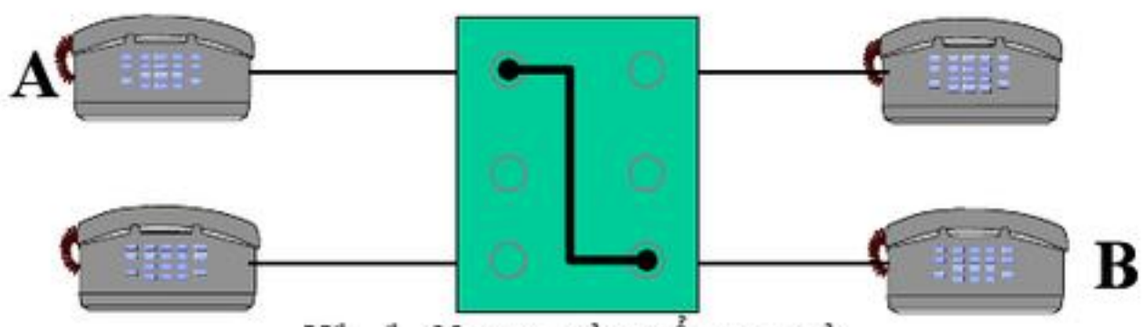
Trạm điện báo là nơi cho phép truyền và nhận các thông điệp dưới dạng các mã Morse, thông thường được thể hiện bằng âm thanh tít và te. Để truyền và nhận thông tin cần có một điện báo viên thực hiện quá trình mã hóa và giải mã thông tin truyền/nhận.

Vì không thể nối trức tiếp tất cả các trạm điện báo lại với nhau, người ta sử dụng các Trạm chuyển điện báo để cho phép nhiều trạm điện báo sử dụng chung một đường truyền để truyền tin. Tại mỗi trạm chuyển điện báo có một thao tác viên chịu trách nhiệm nhận các điện báo gởi đến, xác định đường đi để chuyển tiếp điện báo về nơi nhận. Nếu đường truyền hướng về nơi nhận đang được sử dụng để truyền một điện báo khác, thao tác viên sẽ lưu lại điện báo này để sau đó truyền đi khi đường truyền rãnh.

Để tăng tốc độ truyền tin, hệ thống Baudot thay thế mã Morse bằng mã nhị phân 5 bits (có thể mã hóa cho 32 ký tự). Các trạm điện báo cũng được thay thế bằng các máy têlêtíp (teletype terminal) cho phép xuất / nhập thông tin dạng ký tự. Hệ thống sử dụng kỹ thuật biến điệu (Modulation) và đa hợp (Multiplexing) để truyền tải thông tin.

1.2 Mạng điện thoại

Mạng điện thoại cho phép truyền thông tin dưới dạng âm thanh bằng cách sử dụng hệ thống truyền tín hiệu tuần tự.

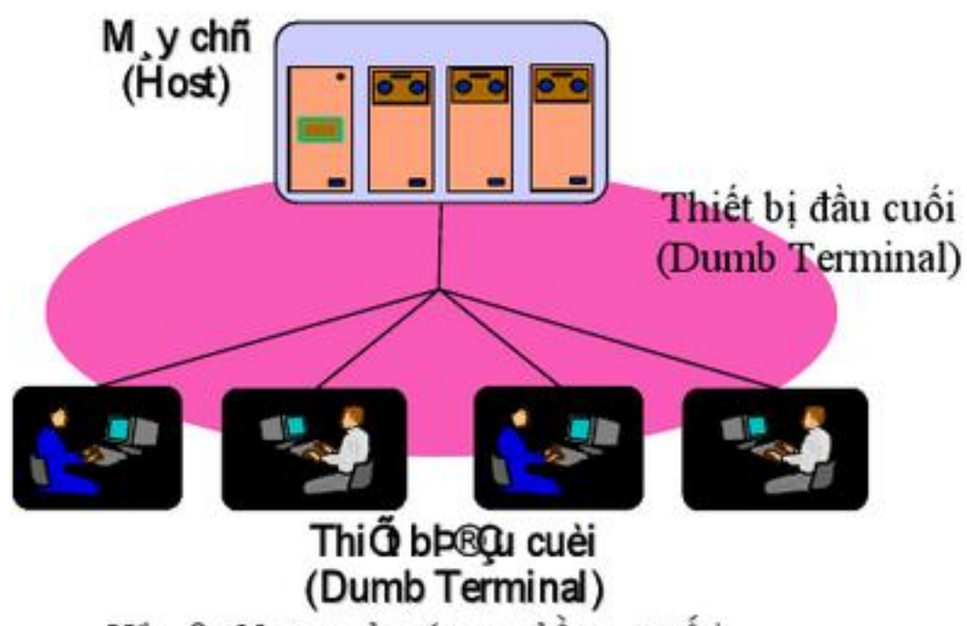


H1.1 Mạng chuyển mạch

Mạng điện thoại hoạt động theo chế độ chuyển mạch định hướng nối kết (circuit switching), tức thiết lập đường nối kết tận hiến giữa hai bên giao tiếp trước khi thông tin được truyền đi (connection oriented).

1.3 Mạng hướng đầu cuối

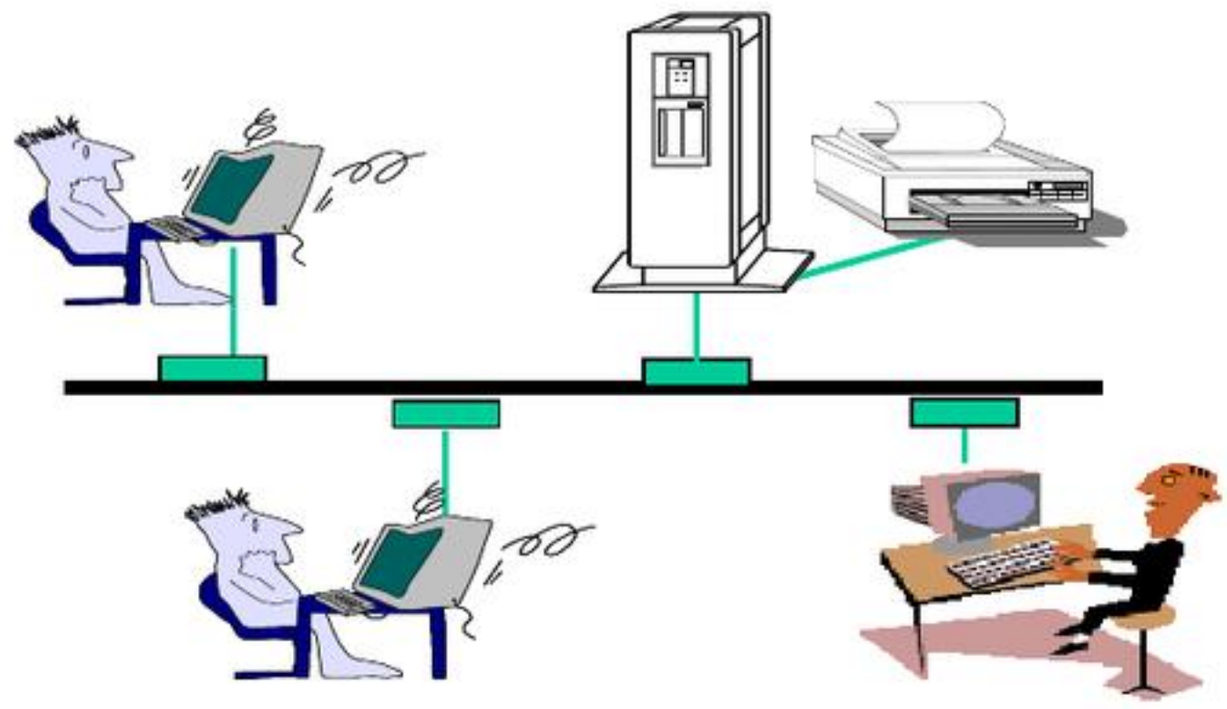
Đây là mô hình của các hệ thống máy tính lớn (Main Frame) vào những năm của thập niên 1970. Hệ thống gồm một máy chủ mạnh (Host) có năng lực tính toán cao được nối kết với nhiều thiết bị đầu cuối đần độn (Dumb terminal) chỉ làm nhiệm vụ xuất nhập thông tin, giao tiếp với người sử dụng.



H1.2 Mạng hướng đầu cuối

1.4 Mạng máy tính

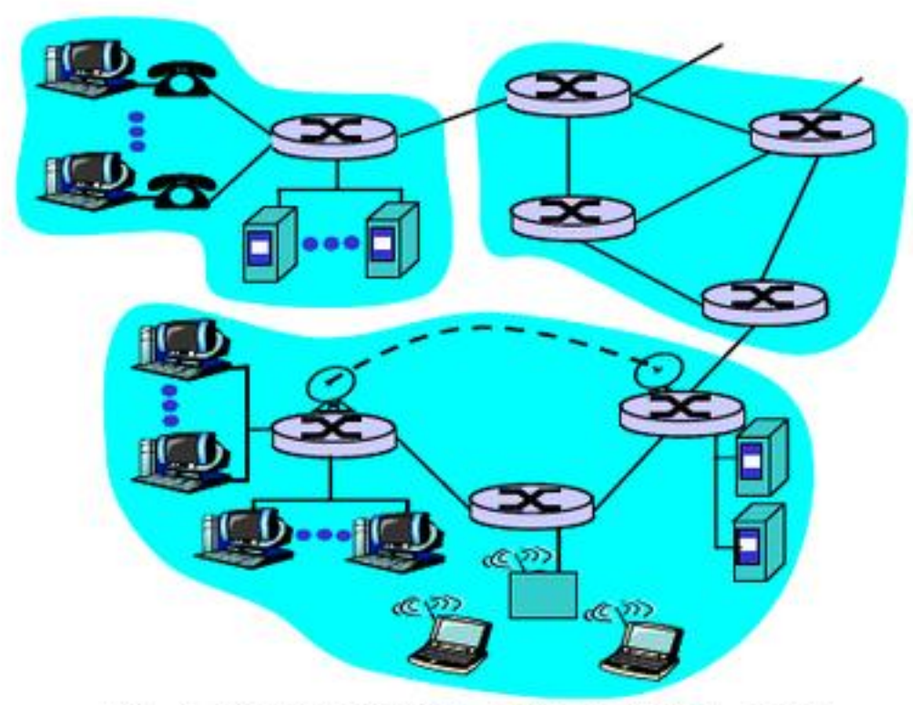
Mạng máy tính là mạng của hai hay nhiều máy tính được nối lại với nhau bằng một đường truyền vật lý theo một kiến trúc nào đó.
Mạng có thể có kiến trúc đơn giản như hình dưới đây:



H1.3 Mạng cục bộ đơn giản

Hoặc phức tạp hơn đó là hệ thống gồm nhiều mạng đơn giản nối lại với nhau như hình sau:
Một hệ thống mạng tổng quát được cấu thành từ 3 thành phần:

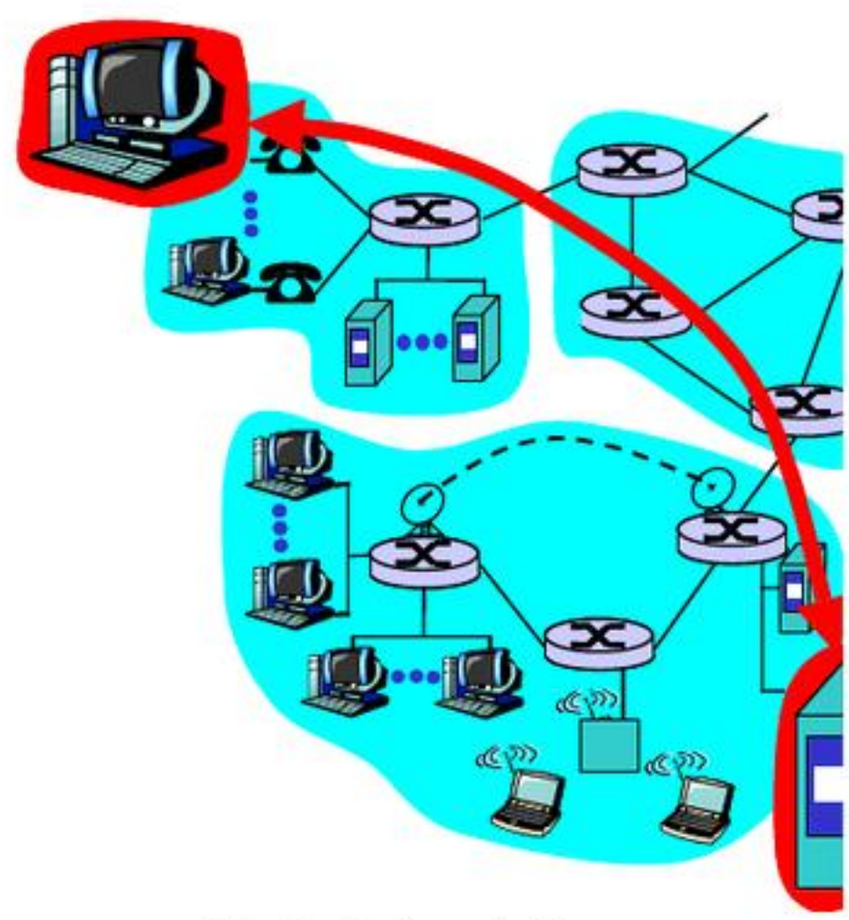
- Đường biên mạng (Network Edge): Gồm các máy tính (Host) và các chương trình ứng dụng mạng (Network Application)
- Đường trục mạng (Network Core): Gồm các bộ chọn đường (router) đóng vài trò là một mạng trung tâm nối kết các mạng lại với nhau.
- Mạng truy cập, đường truyền vật lý (Access Network , physical media): Gồm các đường truyền tải thông tin.



H1.4 Mạng diện rộng phức tạp

1.4.1 Đường biên mạng

Bao gồm các máy tính (Host) trên mạng nơi thực thi các chương trình ứng dụng mạng (Network Application). Đôi khi người ta còn gọi chúng là các Hệ thống cuối (End Systems) với ý nghĩa đây chính là nơi xuất phát của thông tin di chuyển trên mạng, cũng như là điểm dừng của thông tin.
Quá trình trao đổi thông tin giữa hai máy tính trên mạng có thể diễn ra theo hai mô hình: Mô hình Khách hàng / Người phục vụ (Client / server model) hay Mô hình ngang hàng (peer-to-peer model).



H1.5 Đường biên mạng

1.4.1.1 Mô hình khách hàng/người phục vụ (client/server):

Trong mô hình này một máy tính sẽ đóng vai trò là client và máy tính kia đóng vai trò là server. Máy tính client sẽ gởi các yêu cầu (request) đến máy tính server để yêu cầu server thực hiện công việc gì đó. Chẳng hạn khi người dùng duyệt web trên mạng Internet, trình duyệt web sẽ gởi yêu cầu đến web server đề nghị web server gởi về trang web tương ứng.
Máy tính server khi nhận được một yêu cầu từ client gởi đến sẽ phân tích yêu cầu để hiểu được client muốn đều gì, để thực hiện đúng yêu cầu của client. Server sẽ gởi kết quả về cho client trong các thông điệp trả lời (reply). Ví dụ, khi web server nhận được một yêu cầu gởi đến từ trình duyệt web, nó sẽ phân tích yêu cầu để xác định xem client cần nhận trang web nào, sau đó mở tập tin html tương ứng trên đĩa cứng cục bộ của nó để gởi về trình duyệt web trong thông điệp trả lời.
Một số ứng dụng được xây dựng theo mô hình client / server như: www, mail, ftp,...

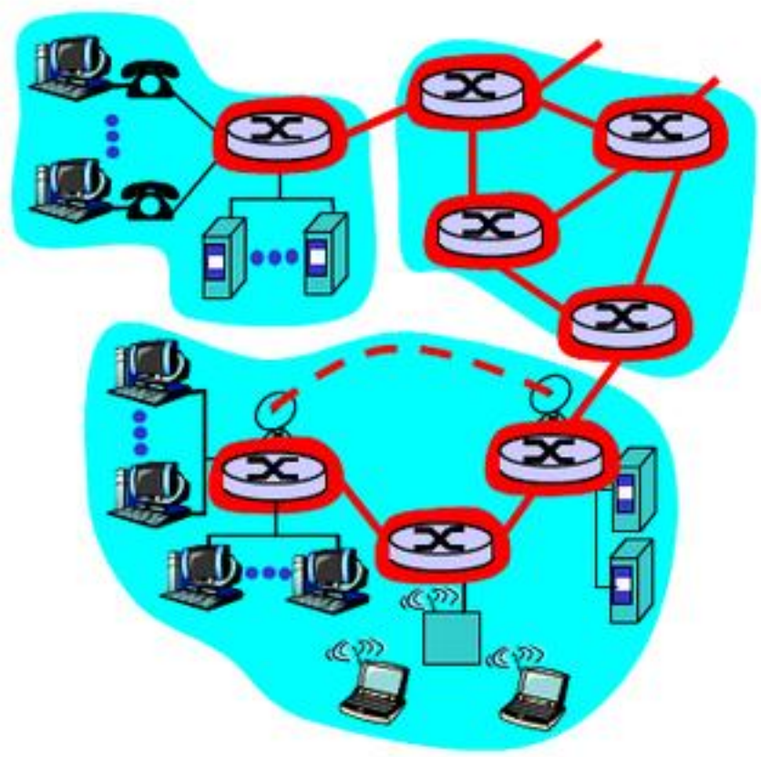
1.4.1.2 Mô hình ngang cấp (peer-to-peer):

Trong mô hình này, một máy tính vừa đóng vai trò là client, vừa đóng vai trò là server. Một số ứng dụng thuộc mô hình này như: Gnutella, KaZaA

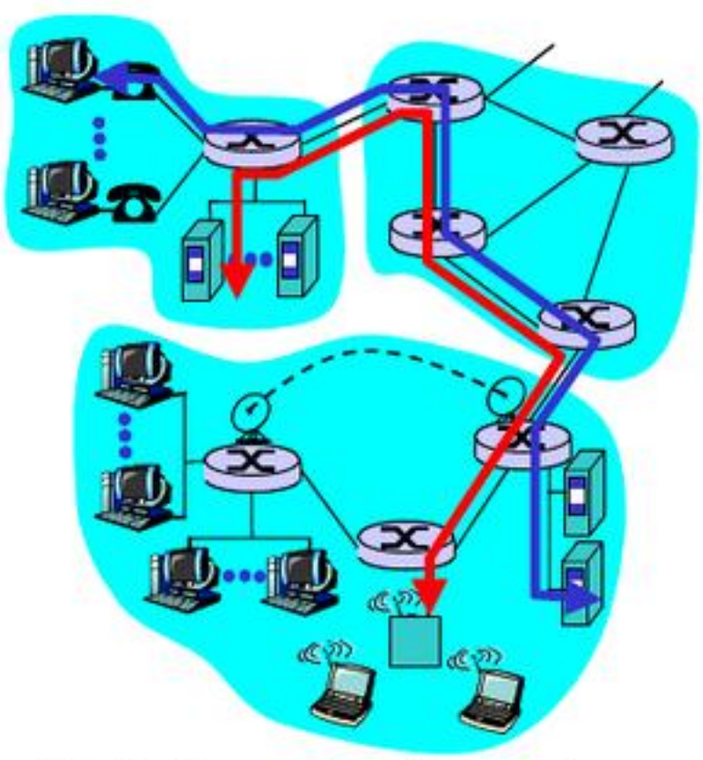
1.4.2 Đường trục mạng

Là hệ thống mạng của các bộ chọn đường (routers), làm nhiệm vụ chọn đường và chuyển tiếp thông tin, đảm bảo sự trao đổi thông tin thông suốt giữa hai máy tính nằm trên hai nhánh mạng cách xa nhau.

Câu hỏi đặt ra là làm sao thông tin có thể được truyền đi trên mạng? Người ta có thể sử dụng một trong hai chế độ truyền tải thông tin là: Chuyển mạch (circuit switching) và chuyển gói (packet switching).



Hl.6 Mạng đường trục



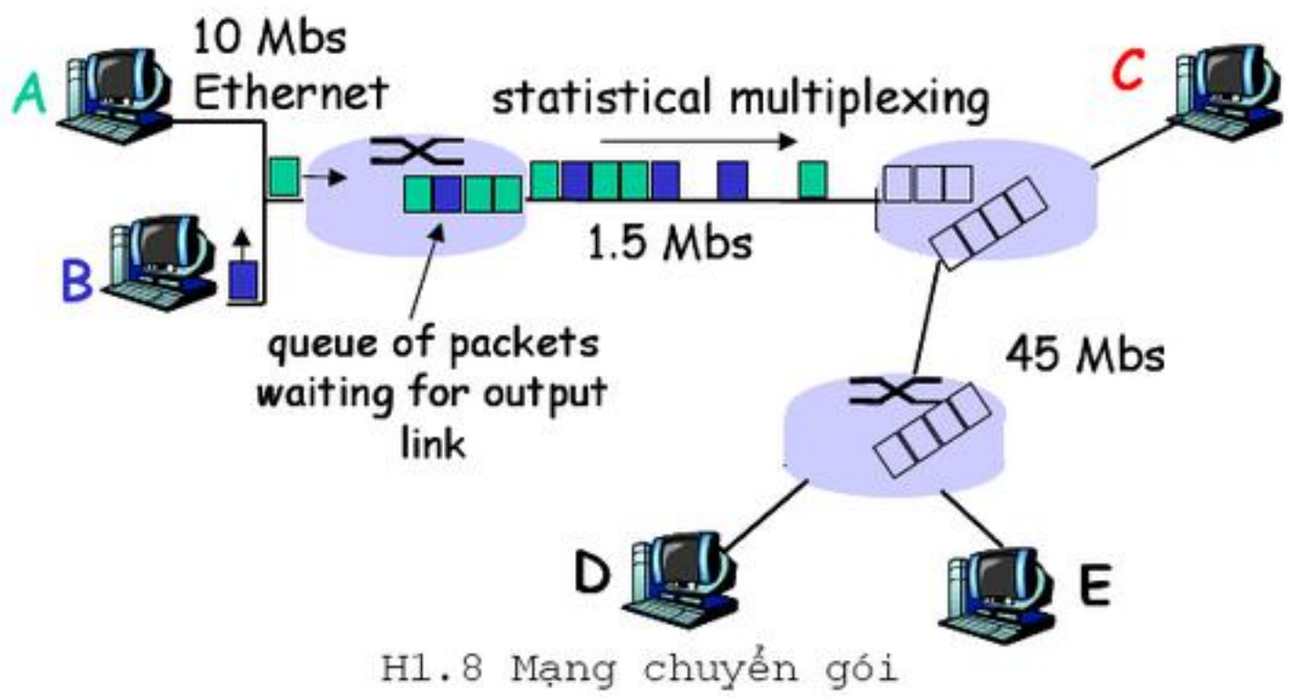
Hl.7 Mang chuyển mach

1.4.2.1 Chuyển mạch (circuit switching)

Chế độ này hoạt động theo mô hình của hệ thống điện thoại. Để có thể giao tiếp với máy B, máy A phải thực hiện một cuộc gọi (call). Nếu máy B chấp nhận cuộc gọi, một kênh ảo được thiết lập dành riêng cho thông tin trao đổi giữa A và B.
Tất cả các tài nguyên được cấp cho cuộc gọi này như băng thông đường truyền, khả năng của các bộ hoán chuyển thông tin đều được dành riêng cho cuộc gọi, không chia sẻ cho các cuộc gọi khác, mặc dù có những khoảng lớn thời gian hai bên giao tiếp "im lặng".
Tài nguyên (băng thông) sẽ được chia thành nhiều những "phần" bằng nhau và sẽ gán cho các cuộc gọi. Khi cuộc gọi sở hữu một "phần" tài nguyên nào đó, mặc dù không sử dụng đến nó cũng không chia sẻ tài nguyên này cho các cuộc gọi khác.
Việc phân chia băng thông của kênh truyền thành những "phần" có thể được thực hiện bằng một trong hai kỹ thuật: Phân chia theo tần số (FDMA-Frequency Division Multi Access) hay phân chia theo thời gian (TDMA- Time Division Multi Access).

1.4.2.2 Mạng chuyển gói

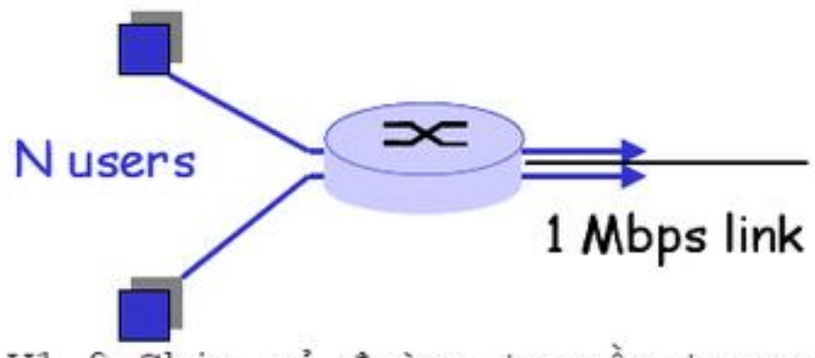
Trong phương pháp này, thông tin trao đổi giữa hai máy tính (end systems) được phân thành những gói tin (packet) có kích thước tối đa xác định.
Gói tin của những người dùng khác nhau (ví dụ của A và B) sẽ chia sẻ nhau băng thông của kênh truyền. Mỗi gói tin sẽ sử dụng toàn bộ băng thông của kênh truyền khi nó được phép. Điều này sẽ dẫn đến tình trạng lượng
thông tin cần truyền đi vượt quá khả năng đáp ứng của kênh truyền. Trong trường hợp này, các router sẽ ứng sử theo giải thuật lưu và chuyển tiếp (store and forward), tức lưu lại các gói tin chưa gởi đi được vào hàng đợi chờ cho đến khi kênh truyền rãnh sẽ lần lượt gởi chúng đi.



H1.8 Mạng chuyển gói

1.4.2.3 So sánh mạng chuyển mạch và mạng chuyển gói

Chuyển gói cho phép có nhiều người sử dụng mạng hơn:



H1.9 Chia sẻ đường truyền trong mạng chuyển gói

Giả sử:

- Một đường truyền 1 Mbit
- Mỗi người dùng được cấp 100Kbps khi truy cập "active"
- Thời gian active chiếm 10% tổng thời gian.

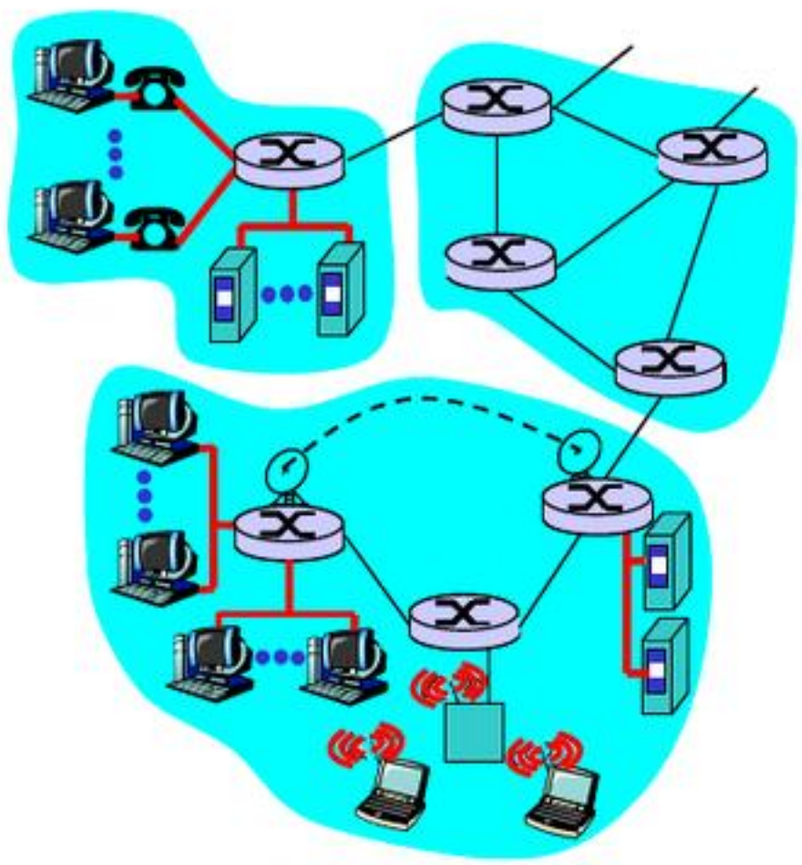
Khi đó:

- circuit-switching: cho phép tối đa 10 users
- packet switching: cho phép 35 users, (xác suất có hơn 10 "active" đồng thời là nhỏ hơn 0.004)

Chuyển gói:

- Thích hợp cho lượng lưu thông dữ liệu lớn nhờ cơ chế chia sẻ tài nguyên và không cần thiết lập cuộc.
- Cần có cơ chế điều khiển tắt nghẽn và mất dữ liệu.
- Không hỗ trợ được cơ chế chuyển mạch để đảm bảo tăng băng thông cố định cho một số ứng dụng về âm thanh và hình ảnh.

1.4.2.4 Mạng truy cập


H1.10 Mạng truy cập

Cho phép nối các máy tính vào các router ngoài biên. Nó có thể là những loại mạng sau:

- Mạng truy cập từ nhà, ví dụ như sử dụng hình thức modem dial qua đường điện thoại hay đường ADSL.
- Mạng cục bộ cho các công ty, xí nghiệp.
- Mạng không dây.

1.4.3 Các lợi ích của mạng máy tính

1.4.3.1 Mạng tạo khả năng dùng chung tài nguyên cho các người dùng.

Vấn đề là làm cho các tài nguyên trên mạng như chương trình, dữ liệu và thiết bị, đặc biệt là các thiết bị đắt tiền, có thể sẵn dùng cho mọi người trên mạng mà không cần quan tâm đến vị trí thực của tài nguyên và người dùng.
Về mặt thiết bị, các thiết bị chất lượng cao thường đắt tiền, chúng thường được dùng chung cho nhiều người nhằm giảm chi phí và dễ bảo quản.
Về mặt chương trình và dữ liệu, khi được dùng chung, mỗi thay đổi sẽ sẵn dùng cho mọi thành viên trên mạng ngay lập tức. Điều này thể hiện rất rõ tại các nơi như ngân hàng, các đại lý bán vé máy bay...

1.4.3.2 Mạng cho phép nâng cao độ tin cậy.

Khi sử dụng mạng, có thể thực hiện một chương trình tại nhiều máy tính khác nhau, nhiều thiết bị có thể dùng chung. Điều này tăng độ tin cậy trong công việc vì khi có máy tính hoặc thiết bị bị hỏng, công việc vẫn có thể tiếp tục với các máy tính hoặc thiết bị khác trên mạng trong khi chờ sửa chữa.

1.4.3.3 Mạng giúp cho công việc đạt hiệu suất cao hơn.

Khi chương trình và dữ liệu đã dùng chung trên mạng, có thể bỏ qua một số khâu đối chiếu không cần thiết. Việc điều chỉnh chương trình (nếu có) cũng tiết kiệm thời gian hơn do chỉ cần cài đặt lại trên một máy.
Về mặt tổ chức, việc sao chép dữ liệu phòng hờ tiện lợi hơn do có thể giao cho chỉ một người thay vì mọi người phải tự sao chép phần của mình.

1.4.3.4 Tiết kiệm chi phí.

Việc dùng chung các thiết bị ngoại vi cho phép giảm chi phí trang bị tính trên số người dùng. Về phần mềm, nhiều nhà sản xuất phần mềm cung cấp cả những ấn bản cho nhiều người dùng, với chi phí thấp hơn tính trên mỗi người dùng.

1.4.3.5 Tăng cường tính bảo mật thông tin.

Dữ liệu được lưu trên các máy phục vụ tập tin (file server) sẽ được bảo vệ tốt hơn so với đặt tại các máy cá nhân nhờ cơ chế bảo mật của các hệ điều hành mạng.

1.4.3.6 Việc phát triển mạng máy tính đã tạo ra nhiều ứng dụng mới

Một số ứng dụng có ảnh hưởng quan trọng đến toàn xã hội: khả năng truy xuất các chương trình và dữ liệu từ xa, khả năng thông tin liên lạc dễ dàng và hiệu quả, tạo môi trường giao tiếp thuận lợi giữa những người dùng khác nhau, khả năng tìm kiếm thông tin nhanh chóng trên phạm vi toàn thế giới,...

Chương 2: Các thành phần của mạng máy tính

Mục đích

Chương này nhằm giới thiệu cho người học những vấn đề sau:

- Các thành phần liên quan đến phần cứng của một mạng máy tính
- Sự phân loại mạng máy tính theo các tiêu chí khác nhau
- Kiến trúc phần mềm của một mạng máy tính, đặc biệt là kiến trúc có thứ bậc của các giao thức mạng
- Mô hình tham khảo OSI

Yêu cầu

Sau khi học xong chương này, người học phải có được những khả năng sau:

- Phân biệt được các loại mạng: mạng quảng bá, mạng chuyển mạch, mạng cục bộ, mạng diện rộng, mạng không dây, liên mạng, …
- Biện luận được sự cần thiết của tiếp cận phân lớp kiến trúc mạng trong việc thiết kế và xây dựng các hệ thống mạng máy
- Phân biệt được các khái niệm quan trọng trong kiến trúc phần mềm mạng như dịch vụ mạng, giao diện mạng và giao thức mạng
- Trình bày được ý nghĩa của mô hình OSI, chức năng cơ bản mỗi tầng trong mô hình

2.1 Phần cứng mạng máy tính

2.1.1 Phân loại mạng máy tính theo kỹ thuật truyền tin

Dựa theo kỹ thuật truyền tải thông tin, người ta có thể chia mạng thành hai loại là Mạng quảng bá (Broadcast Network) và mạng điểm nối điểm (Point – to – point Network)

2.1.1.1 Mạng quảng bá

Trong hệ thống mạng quảng bá chỉ tồn tại một kênh truyền được chia sẻ cho tất cả các máy tính. Khi một máy tính gởi tin, tất cả các máy tính còn lại sẽ nhận được tin đó. Tại một thời điểm chỉ cho phép một máy tính được phép sử dụng đường truyền.

2.1.1.2 Mạng điểm nối điểm

Trong hệ thống mạng này, các máy tính được nối lại với nhau thành từng cặp. Thông tin được gởi đi sẽ được truyền trực tiếp từ máy gởi đến máy nhận hoặc được chuyển tiếp qua nhiều máy trung gian trước khi đến máy tính nhận.

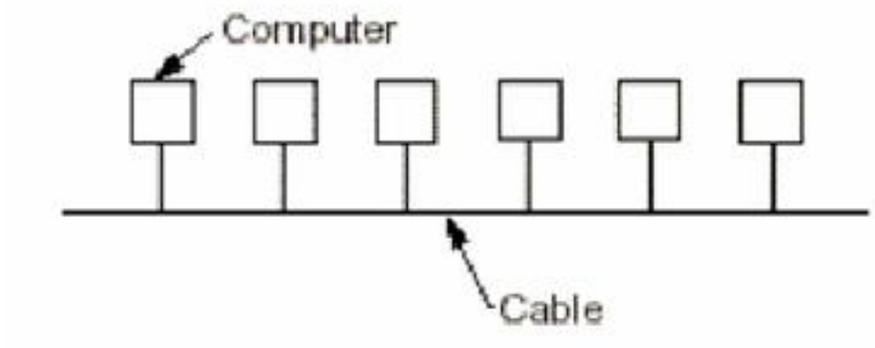
2.1.2 Phân loại mạng máy tính theo phạm vị địa lý

Trong cách phân loại này người ta chú ý đến đại lượng **Đường kính mạng** chỉ khoảng cách của hai máy tính xa nhất trong mạng. Dựa vào đại lượng này người ta có thể phân mạng thành các loại sau:

Đường kính mạng	Vị trí của các máy tính	Loại mạng
1 m	Trong một mét vuông	Mạng khu vực cá nhân
10 m	Trong 1 phòng	Mạng cục bộ, gọi tắt là mạng LAN (Local Area Network)
100 m	Trong 1 tòa nhà	
1 km	Trong một khu vực	
10 km	Trong một thành phố	Mạng thành phố, gọi tắt là mạng MAN (Metropolitan Area Network)
100 km	Trong một quốc gia	Mạng diện rộng, gọi tắt là mạng WAN (Wide Area Network)
1000 km	Trong m ột châu lục	
10000 km	Cả hành tinh	

2.1.2.1 Mạng cục bộ

Đây là mạng thuộc loại mạng quảng bá, sử dụng một đường truyền có tốc độ cao, băng thông rộng, có hình trạng (topology) đơn giản như mạng hình bus, mạng hình sao (Star topology), mạng hình vòng (Ring topology).

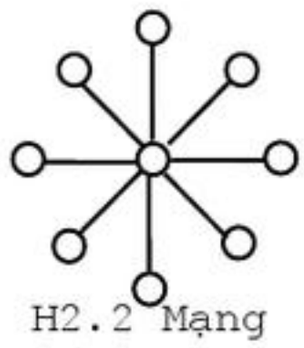


H2.1 Mạng hình Bus

2.1.2.1.1 Mạng hình bus

Tất cả các máy tính được nối lại bằng một dây dẫn (Cáp đồng trục gầy hoặc đồng trục béo). Khi một trong số chúng thực hiện truyền tin, tín hiệu sẽ lan truyền đến tất cả các máy tính còn lại. Nếu có hai máy tính truyền tin cùng một lúc thì sẽ dẫn đến tình trạng đụng độ và trạng thái lỗi xẩy ra.

2.1.2.1.2 Mạng hình sao



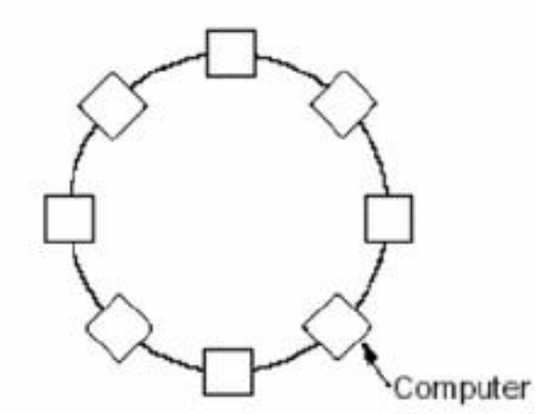
H2.2 Mạng hình sao

Các máy tính được nối trực tiếp vào một Bộ tập trung nối kết, gọi là Hub. Dữ liệu được chuyển qua Hub trước khi đến các máy nhận. Hub có nhiều cổng (port), mỗi cổng cho phép một máy tính nối vào. Hub đóng vai trò như một bộ khuyếch đại (repeater). Nó khuyếch đại tín hiệu nhận được trước khi truyền lại tín hiệu đó trên các cổng còn lại.

Ưu điểm của mạng hình sao là dễ dàng cài đặt, không dừng mạng khi nối thêm vào hoặc lấy một máy tính ra khỏi mạng, cũng như dễ dàng phát hiện lỗi. So với mạng hình Bus, mạng hình sao có tín ổn định cao hơn.

Tuy nhiên nó đòi hỏi nhiều dây dẫn hơn so với mạng hình bus. Toàn mạng sẽ bị ngưng hoạt động nếu Hub bị hư. Chi phí đầu tư mạng hình sao cao hơn mạng hình Bus.

2.1.2.1.3 Mạng hình vòng



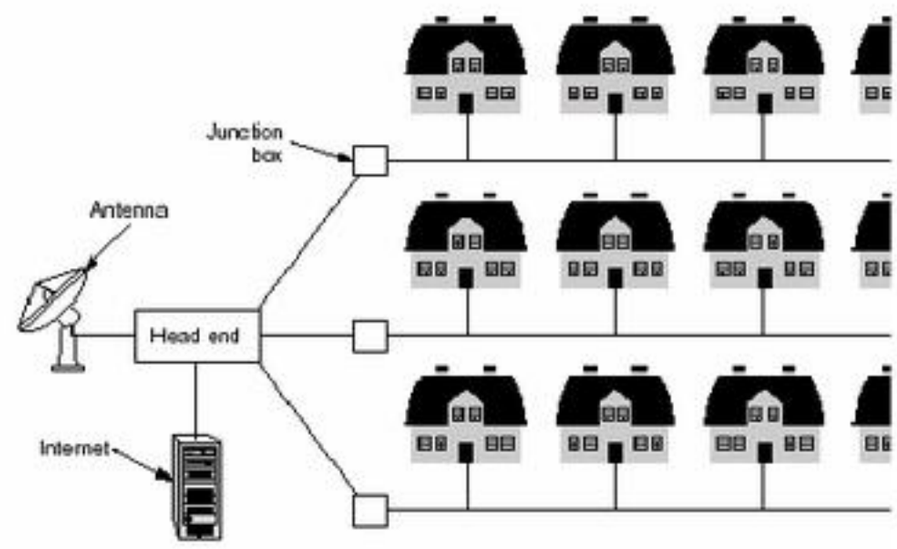
H2.3 Mạng hình vòng

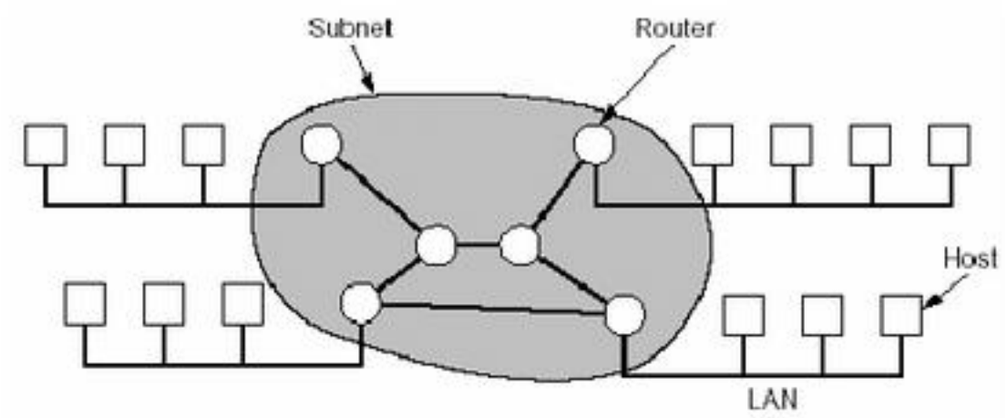
Tồn tại một thẻ bài (token: một gói tin nhỏ) lần lượt truyền qua các máy tính. Một máy tính khi truyền tin phải tuân thủ nguyên tắc sau:

- Chờ cho đến khi token đến nó và nó sẽ lấy token ra khỏi vòng tròn.
- Gởi gói tin của nó đi một vòng qua các máy tính trên đường tròn.
- Chờ cho đến khi gói tin quay về
- Đưa token trở lại vòng tròn để nút bên cạnh nhận token

2.1.2.2 Mạng đô thị

Mạng MAN được sử dụng để nối tất cả các máy tính trong phạm vi toàn thành phố. Ví dụ như mạng truyền hình cáp trong thành phố.



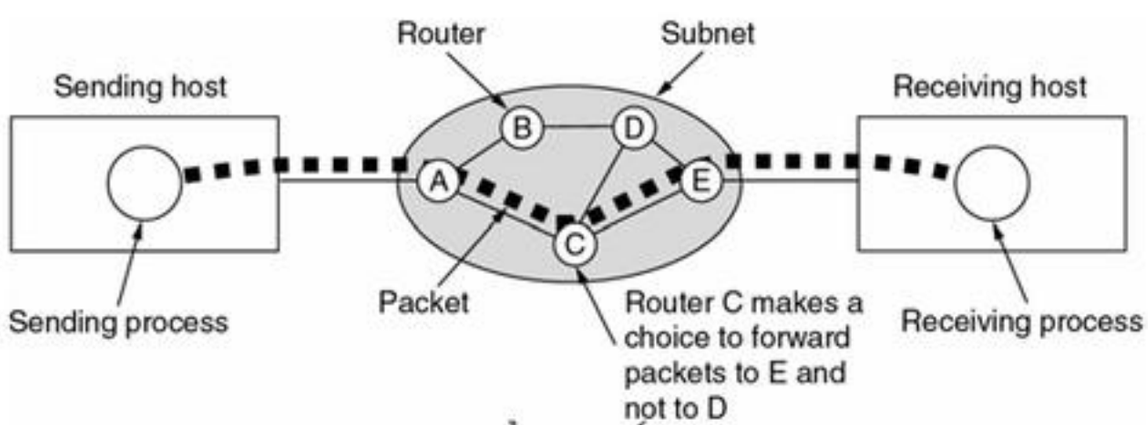


H2.5 Mạng diện rộng

2.1.2.3 Mạng diện rộng

Mạng LAN và mạng MAN thông thường không sử dụng các thiết bị chuyển mạch, điều đó hạn chế trong việc mở rộng phạm vi mạng về số lượng máy tính và khoảng cách. Chính vì thế mạng diện rộng được phát minh.

Trong một mạng WAN, các máy tính (**hosts**) được nối vào một mạng con (subnet) hay đôi khi còn gọi là đường trục mạng (Backbone), trong đó có chứa các bộ chọn đường (**routers**) và các đường truyền tải (**transmission lines**). Các Routers thông thường có nhiệm vụ lưu và chuyển tiếp các gói tin mà nó nhận được theo nguyên lý cơ bản sau: Các gói tin đến một router sẽ được lưu vào trong một hàng chờ, kế đến router sẽ



H2.6 Lưu và chuyển tiếp trong mạng WAN

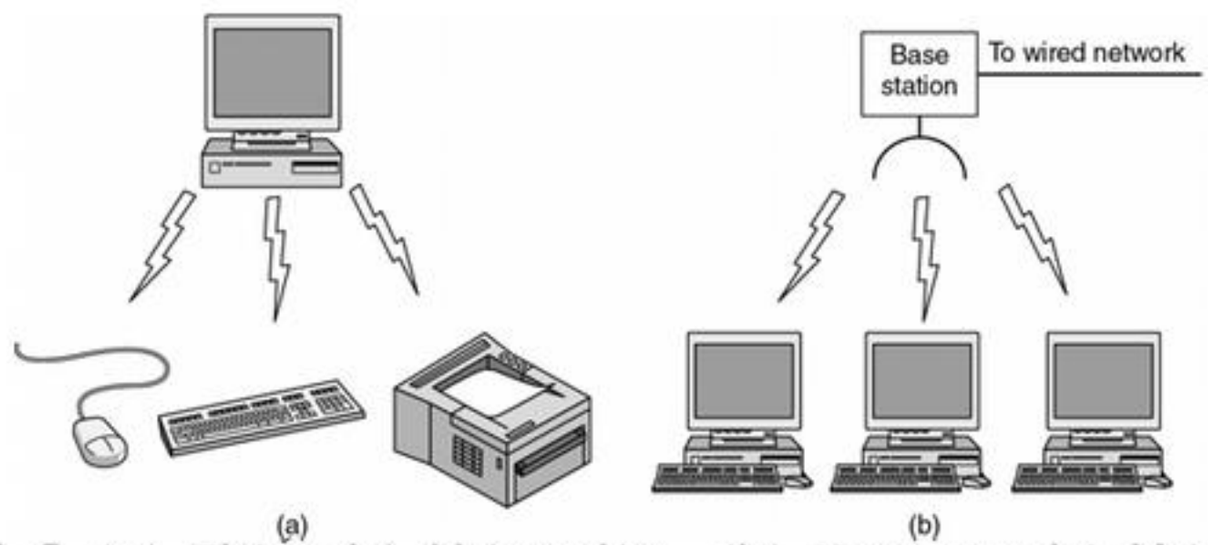
quyết định nơi gói tin cần phải đến và sau đó sẽ chuyển gói tin lên đường đã được chọn.

2.1.3 Mạng không dây

Nếu phân biệt mang theo tiêu chí hữu tuyến hay vô tuyến thì ta có thêm các loại mạng không dây sau:

2.1.3.1 Nối kết hệ thống (System interconnection)

Mạng này nhằm mục đích thay thế hệ thống cáp nối kết các thiết bị cục bộ vào máy tính như màn hình, bàn phím, chuột, phone, loa ,....



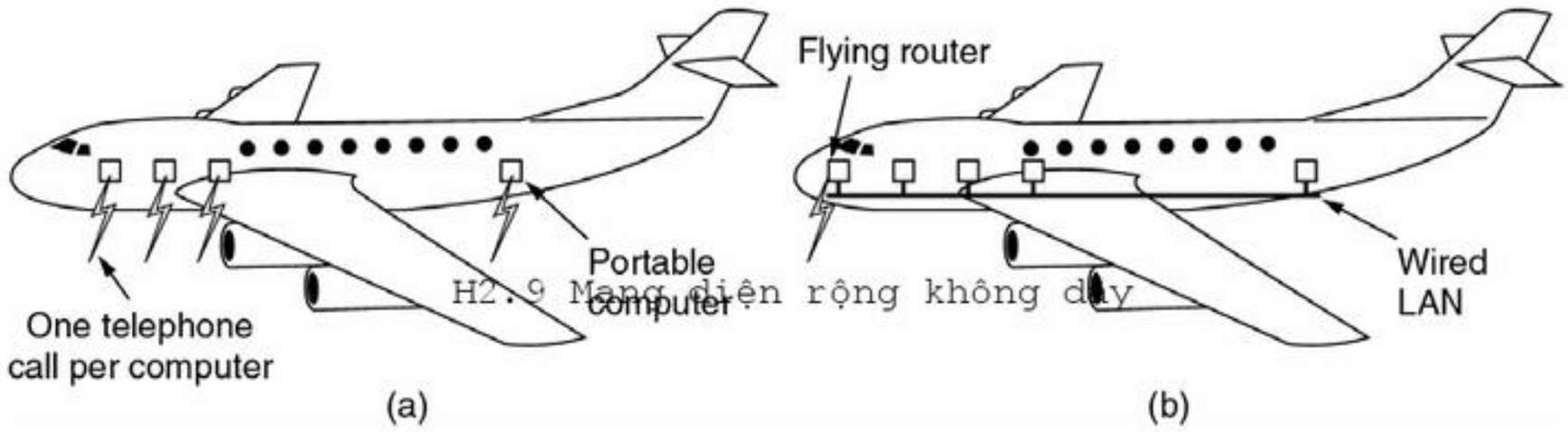
Hình 2.7 (a) Thiết bị không dây, (b) Mạng cục bộ không dây

2.1.3.2 Mạng cục bộ không dây (Wireless LANs):

Tất cả các máy tính giao tiếp với nhau thông qua một trạm cơ sở (Base Station) được nối bằng cáp vào hệ thống mạng.

2.1.3.3 Mạng diện rộng không dây (Wireless WANs):

Thông thường mạng điện thoại di động số thuộc dạng này. Với các công nghệ mới cho phép băng thông mạng có thể đạt đến 50 Mbps với khoảng cách vài kilomet



H2.9 Mạng diện rộng không dây

Trong hình (a) các máy tính sử dụng công nghệ mạng vô tuyến để nối kết với router. Ngược lại trong hình (b), các máy tính được nối bằng đường dây hữu tuyến với một router, để từ đó router sử dụng kỹ thuật vô tuyến để liên lạc với các router khác.

2.1.4 Liên mạng (Internetwork)

Thông thường một mạng máy tính có thể không đồng nhất (**homogeneous**), tức có sự khác nhau về phần cứng và phần mềm giữa các máy tính. Trong thực tế ta chỉ có thể xây dựng được các mạng lớn bằng cách liên nối kết (**interconnecting**) nhiều loại mạng lại với nhau. Công việc này được gọi là liên mạng (Internetworking).

Ví dụ:

- Nối kết một tập các mạng LAN có kiểu khác nhau như dạng Bus với dạng vòng của một công ty.

- Nối các mạng LAN lại với nhau nhờ vào một mạng diện rộng, lúc đó mạng WAN đóng vai trò là một Subnet.
- Nối các mạng WAN lại với nhau hình thành mạng WAN lớn hơn. Liên mạng lớn nhất hiện nay là mạng toàn cầu Internet.

2.2 Phần mềm mạng

Đây là thành phần quan trọng thật sự làm cho mạng máy tính vận hành chứ không phải là phần cứng. Phần mềm mạng được xây dựng dựa trên nền tảng của 3 khái niệm là giao thức (protocol), dịch vụ (service) và giao diện (interface).

- Giao thức (Protocol): Mô tả cách thức hai thành phần giao tiếp trao đổi thông tin với nhau.
- Dịch vụ (Services): Mô tả những gì mà một mạng máy tính cung cấp cho các thành phần muốn giao tiếp với nó.
- Giao diện (Interfaces): Mô tả cách thức mà một khách hàng có thể sử dụng được các dịch vụ mạng và cách thức các dịch vụ có thể được truy cập đến.

2.2.1 Cấu trúc thứ bậc của giao thức

Nền tảng cho tất cả các phần mềm làm cho mạng máy tính hoạt động chính là khái niệm kiến trúc thứ bậc của giao thức (**protocol hierachies**). Nó tổ chức các dịch vụ mà một mạng máy tính cung cấp thành các tầng/lớp (**layers**)

Hai thành phần bộ phận ở hai máy tính khác nhau, nhưng ở cùng cấp, chúng luôn luôn thống nhất với nhau về cách thức mà chúng sẽ trao đổi thông tin. Qui tắc trao đổi thông tin này được mô tả trong một giao thức (**protocol**).

Một hệ mạng truyền tải dữ liệu thường được thiết kế dưới dạng phân tầng. Để minh họa ý nghĩa của nó ta xem xét mô hình hoạt động của hệ thống gởi nhận thư tín thế giới.

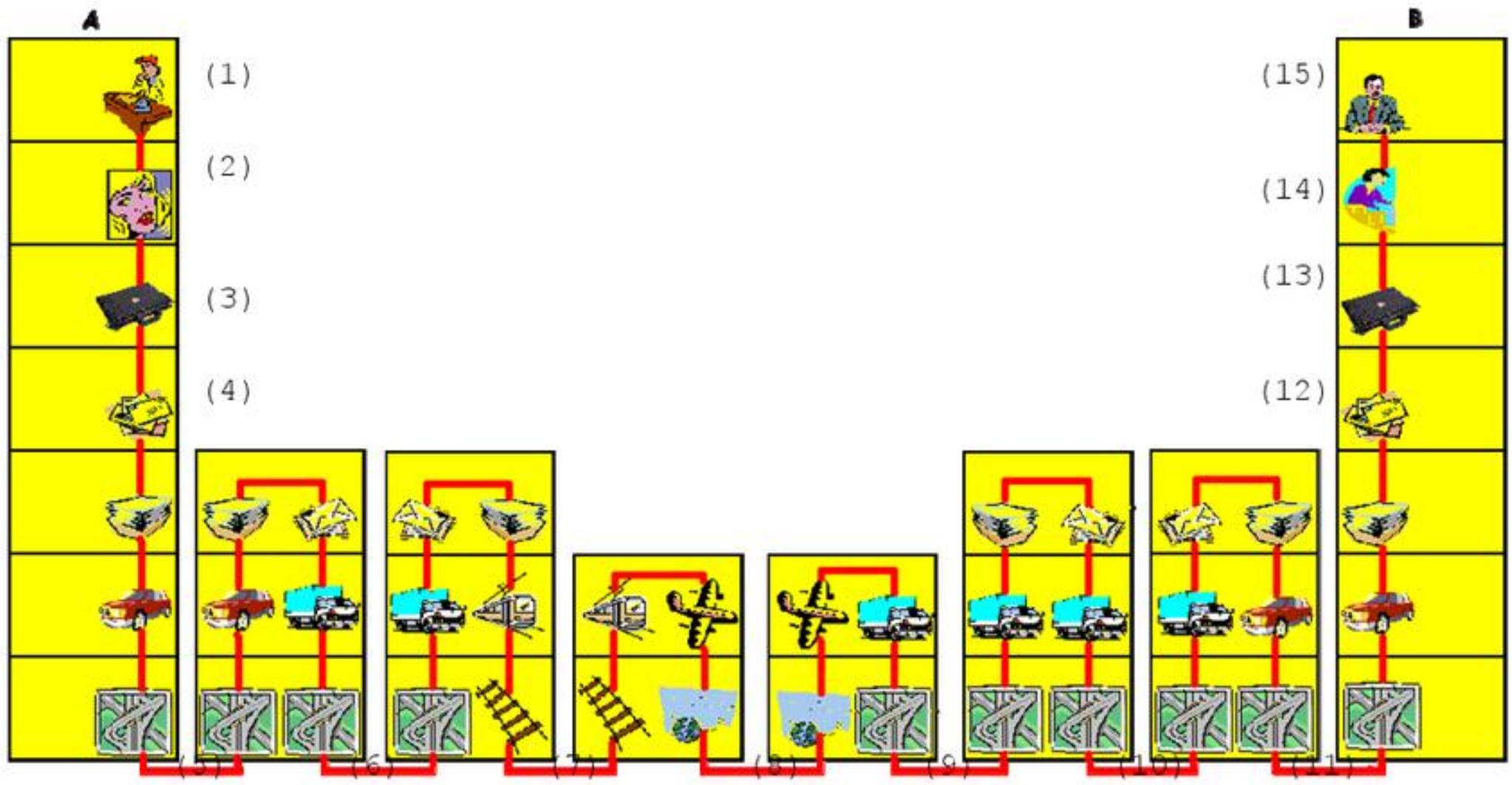
Hai đối tác A ở Paris và B ở Thành phố Cần Thơ thường xuyên trao đổi thư từ với nhau. Vì A không thể nói tiếng Việt và B không thể nói tiếng Pháp, trong khi đó cả hai có thể hiểu tiếng Anh, cho nên nó được chọn là ngôn ngữ để trao đổi thư từ, văn bản giữa A và B. Cả hai gởi thư từ cơ quan của họ. Trong công ty có bộ phận văn thư lãnh trách nhiệm tập hợp và gởi tất cả các thư của công ty ra bưu điện.

Tiến trình A gởi cho B một lá thư diễn ra như sau:

1. A viết một lá thư bằng tiếng Pháp bằng bút máy của anh ta.
2. A đưa lá thư cho thư ký, biết tiếng Anh để thông dịch lá thư ra tiếng Anh, sau đó bỏ lá thư vào bao thư với địa chỉ người nhận là địa chỉ của B.
3. Nhân viên của bộ phận văn thư chịu trách nhiệm thu thập thư của công ty ghé qua văn phòng của A để nhận thư cần gởi đi.
4. Bộ phận văn thư thực hiện việc phân loại thư và dán tem lên các lá thư bằng một máy dán tem.
5. Lá thư được gởi đến bưu điện ở Paris.
6. Lá thư được ô tô chuyển đến trung tâm phân loại ở Paris.
7. Những lá thư gởi sang Việt Nam được chuyển đến sân bay ở Paris bằng tàu điện ngầm.
8. Lá thư gởi sang Việt nam được chuyển đến sân bay Tân Sơn Nhất (Thành Phố Hồ Chí Minh) bằng máy bay.
9. Thư được ô tô chở đến trung tâm phân loại thư của Thành Phố Hồ Chí Minh.
10. Thư cho cơ quan của B được chuyển về Bưu điện Cần Thơ bằng ô tô.
11. Thư cho cơ quan của B được chuyển đến công ty của B bằng ô tô.
12. Bộ phận văn thư của công ty của B tiến hành phân loại thư.

13. Thư được phát vào một giờ đã định đến các người nhận, trong trường hợp này có văn phòng của B.
14. Thư ký của B mở thư ra và dịch nội dung lá thư gởi cho B sang tiếng Việt.
15. B đọc lá thư của A đã gởi cho anh ta.

Ta có thể tóm tắt lại tiến trình trên bằng một mô hình phân tầng với các nút của mạng thư tín này như sau:



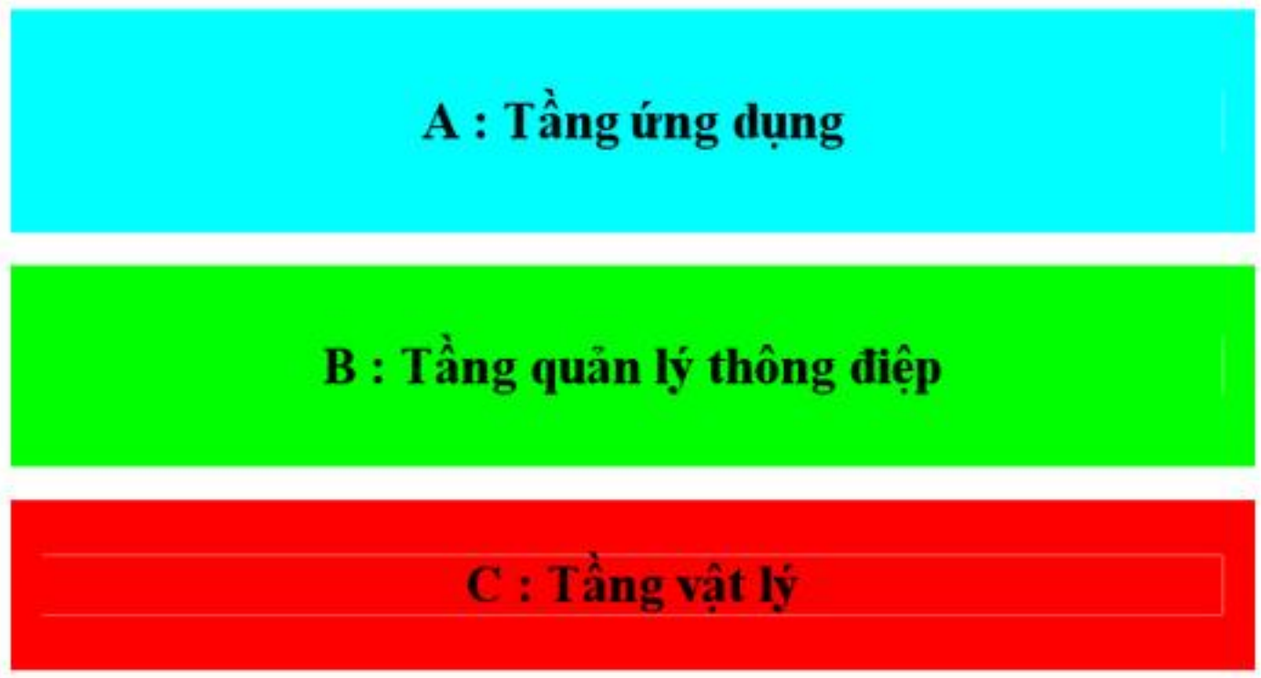
H2.10 Mô hình gởi nhận thư tín thế giới

Trong mô hình trên,mỗi tầng thì dựa trên tầng phía dưới. Ví dụ, các phương tiện của giao thông của tầng như ô tô, tàu hỏa, máy bay (của tầng liên kết dữ liệu) tầng vận chuyển thì cần hạ tầng cơ sở như đường ô tô, đường sắt, sân bay (của tầng vật lý).
Đối với mỗi tầng, các chức năng được định nghĩa là các dịch vụ cung cấp cho tầng phía trên nó. Các đường thẳng màu đỏ trong sơ đồ xác định các dịch vụ được cung cấp bởi các tầng khác nhau. Thêm vào đó, các chức năng của từng tầng tương ứng với các luật được gọi là các giao thức (Protocols).

2.2.2 Ví dụ về cấu trúc thứ bậc của giao thức

Xem xét một ví dụ khác liên quan đến hệ thống truyền tập tin từ máy tính X sang máy Y. Hai máy này được nối với nhau bởi một dây cáp tuần tự. Chúng ta xem xét một mô hình gồm 3 tầng:

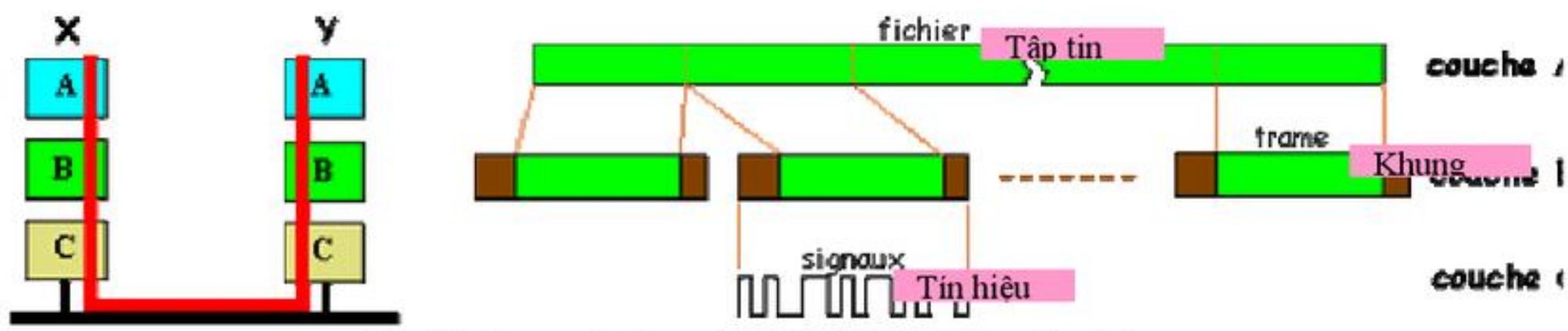
- Người sử dụng muốn truyền một tập tin sẽ thực hiện một lời gọi đến tầng A nhờ vào một hàm đã được định nghĩa sẵn, send_file(fileName, destination). Trong đó fileName là tập tin cần truyền đi, destination là địa chỉ của máy tính nhận tập tin.
- Tầng A phân chia tập tin thành nhiều thông điệp và truyền từng thông điệp nhờ lệnh send_message(MessageNo, destination) do tầng B cung cấp.
- Tầng B quản lý việc gởi các thông điệp, đảm nhiệm việc phân chia các thông điệp thành nhiều đơn vị truyền tin, gọi là các khung (frame); gởi các khung giữa X và Y tuân theo luật đã định trước (protocol) như tần suất gởi, điều khiển luồng, chờ báo nhận của bên nhận, điều khiển lỗi.



H2.11 Ứng dụng 3 tầng

- Tầng B giao cho tầng C một chuỗi các bit mà chúng sẽ được truyền lên đường truyền vật lý, không quan tâm gì về ý nghĩa của các bit, để đến nơi nhận.

Thông tin được truyền trên một kênh truyền đơn giản hoặc phức tạp và được định hướng đến nơi nhận. Bên nhận thực hiện ngược lại tiến trình của bên gởi. Cả bên nhận và bên gởi cùng có số lần gởi/nhận giống nhau.



H2.12 Đơn vị truyền dữ liệu qua các tầng

Ta cũng chú ý rằng, kích thước của các đơn vị truyền tin trong từng tầng là khác nhau. Ở tầng A đơn vị là một tập tin. Tầng B, đơn vị truyền tin là các khung theo một cấu trúc đã được định nghĩa. Tầng C, đơn vị truyền tin là các tín hiện được truyền trên đường truyền vật lý.

2.2.3 Dịch vụ mạng

Hầu hết các tầng mạng đều cung cấp một hoặc cả hai kiểu dịch vụ: Định hướng nối kết và Không nối kết.

- Dịch vụ định hướng nối kết (Connection-oriented): Đây là dịch vụ vận hành theo mô hình của hệ thống điện thoại. Đầu tiên bên gọi phải thiết lập một nối kết, kế đến thực hiện nhiều cuộc trao đổi thông tin và cuối cùng thì giải phóng nối kết.
- Dịch vụ không nối kết (Connectionless): Đây là dịch vụ vận hành theo mô hình kiểu thư tín. Dữ liệu của bạn trước tiên được đặt vào trong một bao thư trên đó có ghi rõ địa chỉ của người nhận và địa chỉ của người gởi. Sau đó sẽ gởi cả bao thư và nội dung đến người nhận.

Một số những dịch vụ thường được cung cấp ở mỗi tầng mạng cho cả hai loại có nối kết và không nối kết được liệt kê ở bảng dưới đây:

Loại	**Dịch vụ**	**Ví dụ**
Có nối kết	Luồng thông điệp tin cậy (Reliable message stream)	Ví dụ gởi tuần tự các trang
	Luồng byte tin cậy (Reliable byte stream)	Đăng nhập từ xa

	Nối kết không tin cậy (Unreliable connection)	Âm thanh số
Không nối kết	Thư tín không tin cậy (Unreliable datagram)	Mail theo kiểu bó
	Thư tín có báo nhận (Acknowledged datagram)	Mail được đăng ký
	Yêu cầu - trả lời (Request – Reply)	Truy vấn cơ sở dữ liệu

Mỗi loại dịch vụ được cung cấp với chất lượng khác nhau. Các loại dịch vụ có nối kết thường đảm bảo thứ tự đến nơi của thông tin như thứ tự chúng đã được gởi đi, cũng như đảm bảo dữ liệu luôn đến nơi. Hai điều này thường không được đảm bảo trong các dịch vụ loại không nối kết.

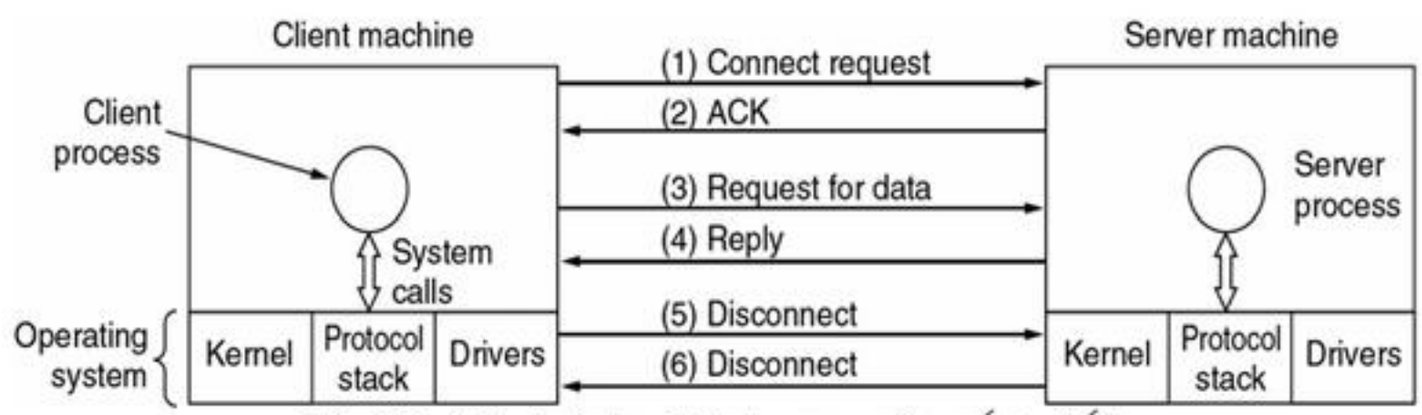
2.2.3.1 Các phép toán của dịch vụ

Một dịch vụ thường được mô tả bằng một tập hợp các hàm cơ bản (primitives) hay đôi khi còn gọi là các tác vụ (operations) sẵn có cho các khách hàng sử dụng. Một số các hàm cơ bản thường có cho một dịch vụ định hướng nối kết như sau:

Hàm cơ bản	**Chức năng**
LISTEN	Nghẽn để chờ một yêu cầu nối kết gởi đến
CONNECT	Yêu cầu thiết lập nối kết với bên muốn giao tiếp
RECIEVE	Nghẽn để chờ nhận các thông điệp gởi đến
SEND	Gởi thông điệp sang bên kia
DISCONNECT	Kết thúc một nối kết

Quá trình trao đổi thông tin giữa Client, người có nhu cầu sử dụng dịch vụ và server, người cung cấp dịch vụ được thực hiện bằng cách sử dụng các hàm cơ sở trên được mô tả như kịch bản sau:

Server	**Client**
LISTEN	
	CONNECT
RECIEVE	SEND
SEND	RECIEVE
DISCONNECT	DISCONNECT



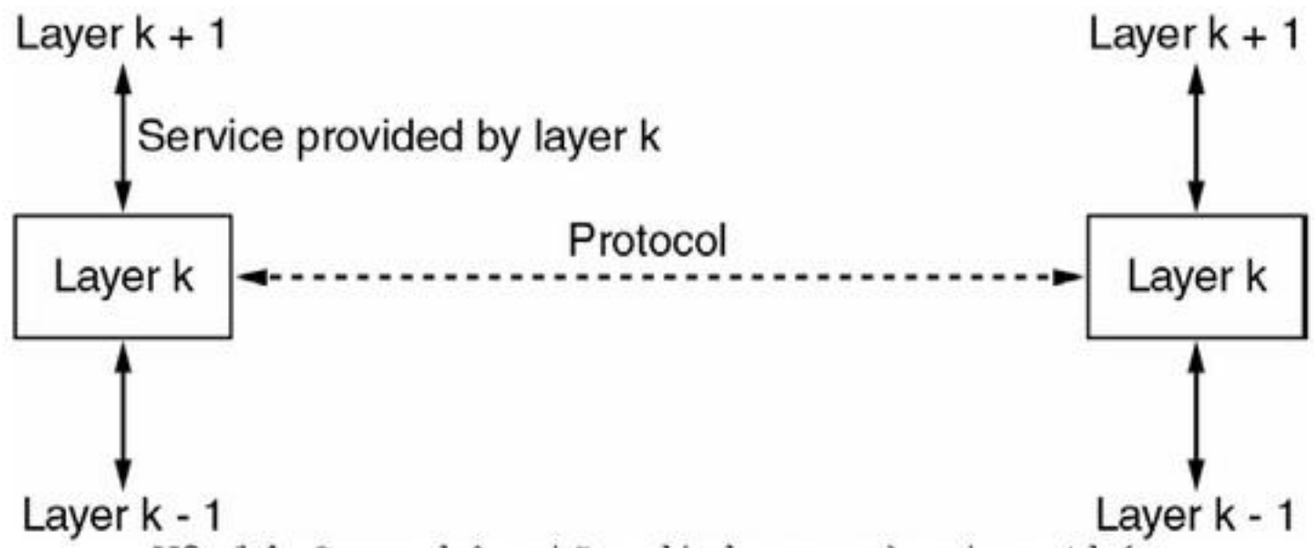
H2.13 Mô hình dịch vụ có nối kết

2.2.3.2 Sự khác biệt giữa dịch vụ và giao thức

Giao thức và dịch vụ là hai nền tảng rất quan trọng trong việc thiết kết và xây dựng một hệ thống mạng. Cần hiểu rõ ý nghĩa và phân biệt sự khác biệt giữa chúng.

- **Dịch vụ:** là một tập các phép toán mà một tầng cung cấp cho tầng phía trên của nó gọi sử dụng.

- **Giao thức:** là một tập các luật mô tả khuôn dạng dữ liệu, ý nghĩa của các gói tin và thứ tự các gói tin được sử dụng trong quá trình giao tiếp.
- **Chú ý:** Cùng một service có thể được thực hiện bởi các protocol khác nhau; mỗi protocol có thể được cài đặt theo một cách thức khác nhau (sử dụng cấu trúc dữ liệu khác nhau, ngôn ngữ lập trình là khác nhau, vv...)



H2.14 Quan hệ giữa dịch vụ và giao thức

2.3 Mô hình tham khảo OSI

Để dễ dàng cho việc nối kết và trao đổi thông tin giữa các máy tính với nhau, vào năm 1983, tổ chức tiêu chuẩn thế giới ISO đã phát triển một mô hình cho phép hai máy tính có thể gởi và nhận dữ liệu cho nhau. Mô hình này dựa trên tiếp cận phân tầng (lớp), với mỗi tầng đảm nhiệm một số các chức năng cơ bản nào đó.
Để hai máy tính có thể trao đổi thông tin được với nhau cần có rất nhiều vấn đề liên quan. Ví dụ như cần có Card mạng, dây cáp mạng, điện thế tín hiệu trên cáp mạng, cách thức đóng gói dữ liệu, điều khiển lỗi đường truyền vv... Bằng cách phân chia các chức năng này vào những tầng riêng biệt nhau, việc viết các phần mềm để thực hiện chúng trở nên dễ dàng hơn. Mô hình OSI giúp đồng nhất các hệ thống máy tính khác biệt nhau khi chúng trao đổi thông tin. Mô hình này gồm có 7 tầng:

Tầng 7: Tầng ứng dụng (Application Layer)

Đây là tầng trên cùng, cung cấp các ứng dụng truy xuất đến các dịch vụ mạng. Nó bao gồm các ứng dụng của người dùng, ví dụ như các Web Browser (Netscape Navigator, Internet Explorer), các Mail User Agent (Outlook Express, Netscape Messenger, ...) hay các chương trình làm server cung cấp các dịch vụ mạng như các Web Server (Netscape Enterprise, Internet Information Service, Apache, ...), Các FTP Server, các Mail server (Send mail, MDeamon). Người dùng mạng giao tiếp trực tiếp với tầng này.

Tầng 6: Tầng trình bày (Presentation Layer)

Tầng này đảm bảo các máy tính có kiểu định dạng dữ liệu khác nhau vẫn có thể trao đổi thông tin cho nhau. Thông thường các máy tính sẽ thống nhất với nhau về một kiểu định dạng dữ liệu trung gian để trao đổi thông tin giữa các máy tính. Một dữ liệu cần gởi đi sẽ được tầng trình bày chuyển sang định dạng trung gian trước khi nó được truyền lên mạng. Ngược lại, khi nhận dữ liệu từ mạng, tầng trình bày sẽ chuyển dữ liệu sang định dạng riêng của nó.

Tầng 5: Tầng giao dịch (Session Layer)

Tầng này cho phép các ứng dụng thiết lập, sử dụng và xóa các kênh giao tiếp giữa chúng (được gọi là giao dịch). Nó cung cấp cơ chế cho việc nhận biết tên và các chức năng về bảo mật thông tin khi truyền qua mạng.

Tầng 4: Tầng vận chuyển (Transport Layer)

Tầng này đảm bảo truyền tải dữ liệu giữa các quá trình. Dữ liệu gởi đi được đảm bảo không có lỗi, theo đúng trình tự, không bị mất mát, trùng lắp. Đối với các gói tin có kích thước lớn, tầng này sẽ phân chia chúng thành các phần nhỏ trước khi gởi đi, cũng như tập hợp lại chúng khi nhận được.

Tầng 3: Tầng mạng (Network Layer)

Tầng này đảm bảo các gói tin dữ liệu (Packet) có thể truyền từ máy tính này đến máy tính kia cho dù không có đường truyền vật lý trực tiếp giữa chúng. Nó nhận nhiệm vụ tìm đường đi cho dữ liệu đến các đích khác nhau trong mạng.

Tầng 2: Tầng liên kết dữ liệu (Data-Link Layer)

Tầng này đảm bảo truyền tải các khung dữ liệu (Frame) giữa hai máy tính có đường truyền vật lý nối trực tiếp với nhau. Nó cài đặt cơ chế phát hiện và xử lý lỗi dữ liệu nhận.

Tầng 1: Tầng vật ký (Physical Layer)

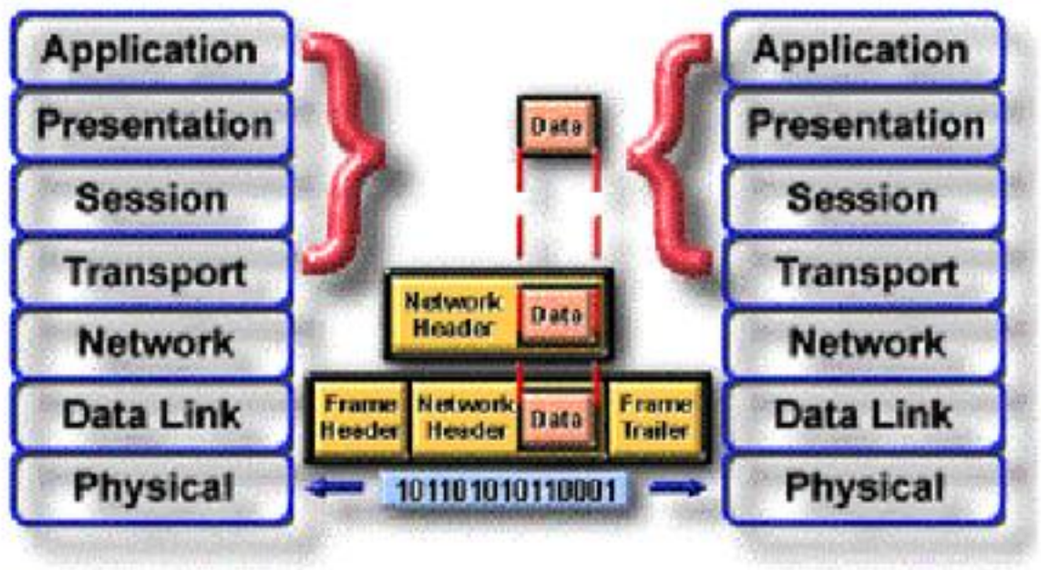
Điều khiển việc truyền tải thật sự các bit trên đường truyền vật lý. Nó định nghĩa các tín hiệu điện, trạng thái đường truyền, phương pháp mã hóa dữ liệu, các loại đầu nối được sử dụng.

Về nguyên tắc, tầng n của một hệ thống chỉ giao tiếp, trao đổi thông tin với tầng n của hệ thống khác. Mỗi tầng sẽ có các đơn vị truyền dữ liệu riêng:

- Tầng vật lý: bit
- Tầng liên kết dữ liệu: Khung (Frame)
- Tầng Mạng: Gói tin (Packet)
- Tầng vận chuyển: Đoạn (Segment)

H2.15-Xử lý dữ liệu qua các tầng

Trong thực tế, dữ liệu được gởi đi từ tầng trên xuống tầng dưới cho đến tầng thấp nhất của máy tính gởi. Ở đó, dữ liệu sẽ được truyền đi trên đường truyền vật lý. Mỗi khi dữ liệu được truyền xuống tầng phía dưới thì nó bị "gói" lại trong đơn vị dữ liệu của tầng dưới. Tại bên nhận, dữ liệu sẽ được truyền ngược lên các tầng cao dần. Mỗi lần qua một tầng, đơn vị dữ liệu tương ứng sẽ được tháo ra.



Đơn vị dữ liệu của mỗi tầng sẽ có một tiêu đề (header) riêng.

OSI chỉ là mô hình tham khảo, mỗi nhà sản xuất khi phát minh ra hệ thống mạng của mình sẽ thực hiện các chức năng ở từng tầng theo những cách thức riêng. Các cách thức này thường được mô tả dưới dạng các chuẩn mạng hay các giao

thức mạng. Như vậy dẫn đến trường hợp cùng một chức năng nhưng hai hệ thống mạng khác nhau sẽ không tương tác được với nhau. Hình dưới sẽ so sánh kiến trúc của các hệ điều hành mạng thông dụng với mô hình OSI.
Để thực hiện các chức năng ở tầng 3 và tầng 4 trong mô hình OSI, mỗi hệ thống mạng sẽ có các protocol riêng:

- UNIX: Tầng 3 dùng giao thức IP, tầng 4 giao thức TCP/UDP
- Netware: Tầng 3 dùng giao thức IPX, tầng 4 giao thức SPX
- Microsoft định nghĩa giao thức NETBEUI để thực hiện chức năng của cả tầng 3 và tầng 4
-

Nếu chỉ dừng lại ở đây thì các máy tính UNIX, Netware và NT sẽ không trao đổi thông tin được với nhau. Với sự lớn mạnh của mạng Internet, các máy tính cài đặt các hệ điều hành khác nhau đòi hỏi phải giao tiếp được với nhau, tức phải sử dụng chung một giao thức. Đó chính là bộ giao thức TCP/IP, giao thức của mạng Internet.

H2.16 - Kiến trúc của một số hệ điều hành mạng thông dụng

Chương 3: Tầng vật lý

Mục đích

Chương này nhằm giới thiệu những nội dung cơ bản sau:

- Giới thiệu mô hình của một hệ thống truyền dữ liệu đơn giản và các vấn đề có liên quan đến trong một hệ thống truyền dữ liệu sử dụng máy tính
- Giới thiệu các phương pháp số hóa thông tin
- Giới thiệu về đặc điểm kênh truyền, tính năng kỹ thuật của các loại cáp truyền dữ liệu
- Giới thiệu các hình thức mã hóa dữ liệu số để truyền tải trên đường truyền

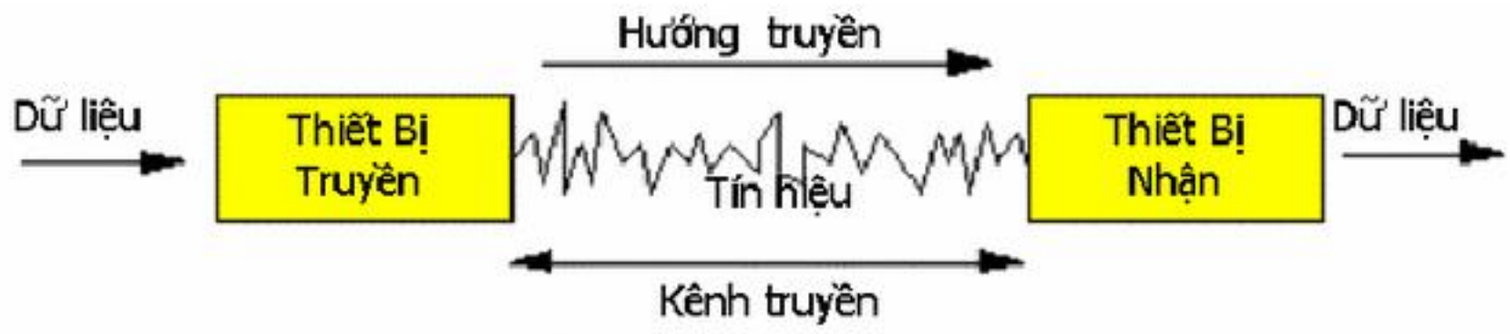
Yêu cầu

Sau khi học xong chương này, người học phải có được những khả năng sau:

- Liệt kê được những vấn đề cơ bản có liên quan đến một hệ thống truyền dữ liệu
- Mô tả được các hình thức số hóa thông tin
- Phân biệt và tính toán được các đại lượng liên quan đến đặc tính của một kênh truyền như: Băng thông, tần số biến điệu, tốc độ dữ liệu, nhiễu, dung lượng và giao thông của một kênh truyền
- Mã hóa được dữ liệu số nhờ vào các tín hiệu số và tuần tự theo các kỹ thuật khác nhau.

3.1 Giới thiệu

Về cơ bản, một hệ thống mạng truyền dữ liệu đơn giản nhất được mô tả như sau:



H3.1 Hệ thống truyền dữ liệu đơn giản

Trong mô hình trên, dữ liệu gồm có văn bản, hình ảnh, âm thanh, phim ảnh cần được số hóa dưới dạng nhị phân (bit 0, 1) để dễ dàng cho xử lý và truyền tải. Thiết bị truyền được nối với thiết bị nhận bằng một đường truyền hữu tuyến hoặc vô tuyến.

Truyền tin là quá trình thiết bị truyền gởi đi lần lượt các bit của dữ liệu lên kênh truyền để nó lan truyền sang thiết bị nhận và như thế là dữ liệu đã được truyền đi. Các thiết bị truyền và nhận là các máy tính. Để cho hệ thống này có thể hoạt động được thì các vấn đề sau cần phải được xem xét:

- Cách thức mã hóa thông tin thành dữ liệu số.
- Các loại kênh truyền dẫn có thể sử dụng để truyền tin.
- Sơ đồ nối kết các thiết bị truyền và nhận lại với nhau.
- Cách thức truyền tải các bit từ thiết bị truyền sang thiết bị nhận.

Hệ thống trên là hệ thống cơ bản nhất cho các hệ thống truyền dữ liệu. Nó thực hiện đầy đủ các chức năng mà tầng vật lý trong mô hình OSI qui định.

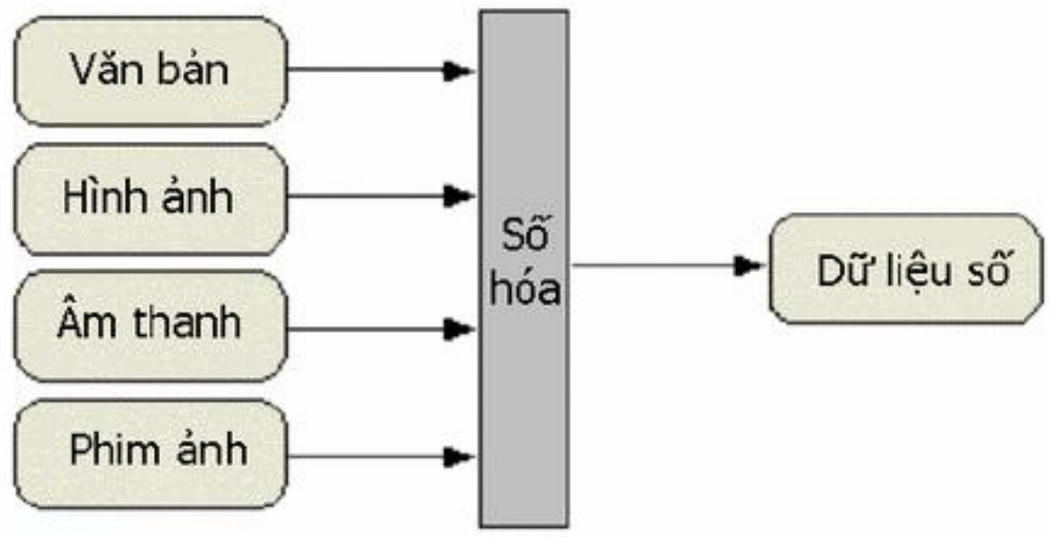
3.2 Vấn đề số hóa thông tin

Thông tin tồn tại dưới nhiều hình thức khác nhau. Để xử lý, mà đặc biệt để truyền tải thông tin ta cần phải mã hóa chúng.

Lời nói : Hệ thống : điện thoại Bộ mã hóa : micro Bộ giải mã : Loa Truyền tải : tín hiệu tuần tự hay tín hiệu số	***Ảnh tĩnh :*** Hệ thống: fax Bộ mã hóa : scanner Bộ giải mã : Bộ thông dịch tập tin Truyền tải : Tín hiệu tuần tự hoặc tín hiệu số.
Dữ liệu tin học : Hệ thống : mạng truyền tin. Bộ mã hóa : Bộ điều khiển truyền thông. Bộ giải mã:Bộ điều khiển truyền thông Truyền tải : Tín hiệu tuần tự hoặc tín hiệu số.	***Truyền hình :*** Hệ thống : truyền quảng bá Bộ mã hóa : caméra Bộ giải mã : bộ thu TV + antenne Truyền tải : Tín hiệu tuần tự hoặc tín hiệu số.

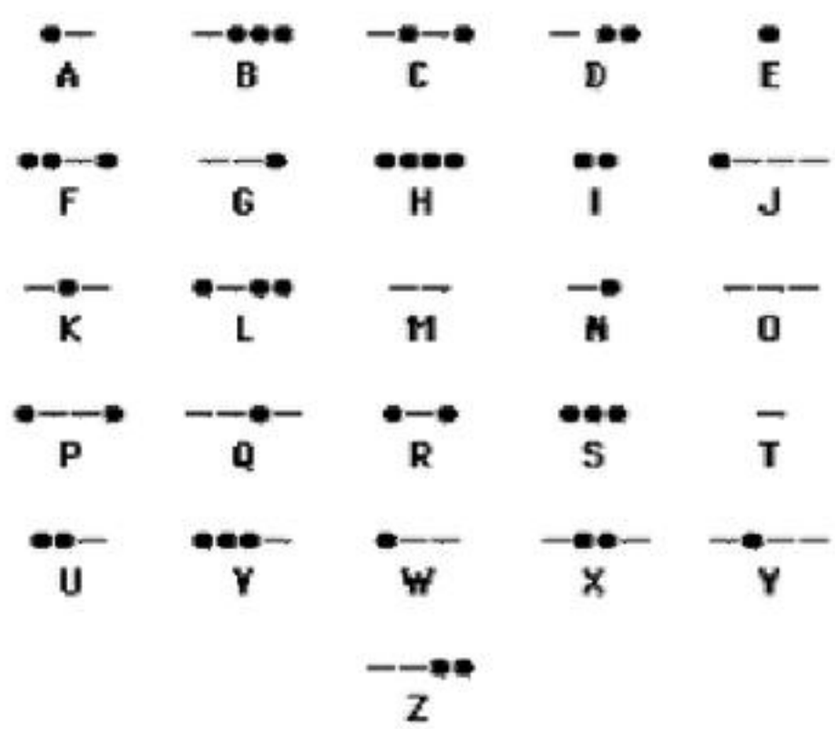
Trong thời đại chúng ta, thông tin thường được thể hiện dưới dạng các trang tài liệu hỗn hợp, như các trang web, mà ở đó đồng thời có thể thể hiện văn bản, hình ảnh tĩnh, hình ảnh động, phim ảnh,.... Thông tin thực tế được thể hiện dưới dạng đa phương tiện. Mỗi một loại thông tin sở hữu

hệ thống mã hóa riêng, nhưng kết quả thì giống nhau: một chuỗi các số 0 và 1. Việc truyền tải thông tin bao gồm việc truyền tải các bit này.
Mô hình mã hóa như sau:



H3.2 Sơ đồ số hóa dữ liệu

3.2.1 Số hóa văn bản



H3.3 Mã Morse

Hệ thống mã hóa đầu tiên liên quan đến văn bản là hệ thống mã Morse, được sử dụng rộng rãi trước khi có máy tính. Đây là một bộ mã nhị phân sử dụng 2 ký tự chấm (.) và gạch (-) để số hóa văn bản (có thể xem tương đương với các bit 0 và 1).
Tuy nhiên nó có nhiều điểm bất lợi sau:

- Nghèo nàn: ít các ký tự được mã hóa;
- Nó sử dụng sự phối hợp của các dấu gạch và dấu chấm với độ dài khác nhau, điều này không được tiện lợi đặc biệt cho các ký tự có tần suất xuất hiện giống nhau.

Chính vì thế nó không được dùng để số hóa thông tin.
Nếu chúng ta qui định rằng số bit dùng để mã hóa cho một ký tự phải bằng nhau thì với p bit ta có thể mã hóa cho 2^p ký tự. Hệ thống mã hóa như thế đã được dùng trong quá khứ.

Ví dụ :
5 bit: dùng trong hệ thống ATI (Alphabet Télégraphique International)
7 bit : gọi là mã ASCII (American Standard Code for Informatics Interchange) được dùng rộng rãi trong máy tính.

poids forts

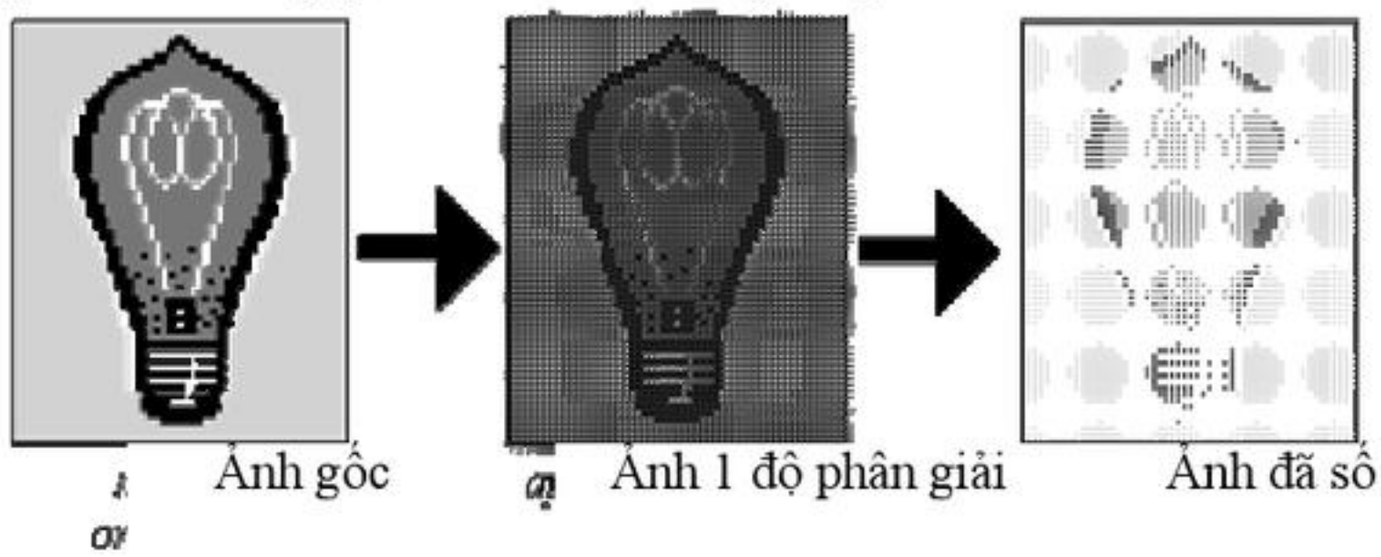
poids faibles	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	\	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	Ç	l	ù
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	↑	n	~
1111	SI	US	/	?	O	<--	o	DEL

H3.4 Mã ASCII chuẩn

Bảng mã này có cả các ký tự không in được gọi là các ký tự điều khiển được dùng để tạo ra các tác vụ trên các thiết bị tin học hay dùng để điều khiển thông tin truyền tải.
Bảng mã 8 bits: có mã ASCII mở rộng và mã EBCDIC
Vì máy tính lưu thông tin dưới dạng các byte 8 bit nên khi sử dụng mã ASCII 7 bít thì bit có trọng số lớn nhất (vị trí thứ 7) luôn có giá trị là 0. Chúng ta có thể sử dụng bit này để định nghĩa các ký tự đặc biệt bằng cách đặt nó giá trị 1. Và như thế chúng ta có một bảng mã ASCII mở rộng. Tuy nhiên, điều này sẽ dẫn đến việc tồn tại nhiều bảng mã ASCII mở rộng khác nhau làm khó khăn trong việc trao đổi thông tin trên phạm vi toàn thế giới.
Mã EBCDIC dùng 8 bits để mã hóa nhờ đó có thể thể hiện được 256 ký tự. Nó được sử dụng trong các máy tính IBM. Tuy nhiên nó không thông dụng như mã ASCII.
Mã 16 bits : Mã Unicode
Mã này được phát triển gần đây để thỏa mãn nhu cầu trao đổi thông tin giữa những người dùng Web. Nó mã hóa hầu hết tất cả các ký tự của các ngôn ngữ trên thế giới. Nó tương thích với mã ASCII 7 bits ở 127 ký tự đầu tiên. Hiện nay mã Unicode bắt đầu được sử dụng rộng rãi.

3.2.2 Số hóa hình ảnh tĩnh

Ảnh số thật sự là một ảnh được vẽ nên từ các đường thẳng và mỗi đường thẳng được xây dựng bằng các điểm. Một ảnh theo chuẩn VGA với độ phân giải 640x480 có nghĩa là một ma trận gồm 480 đường ngang và mỗi đường gồm 640 điểm ảnh (pixel).



H3.5 Mã hóa hình ảnh tĩnh

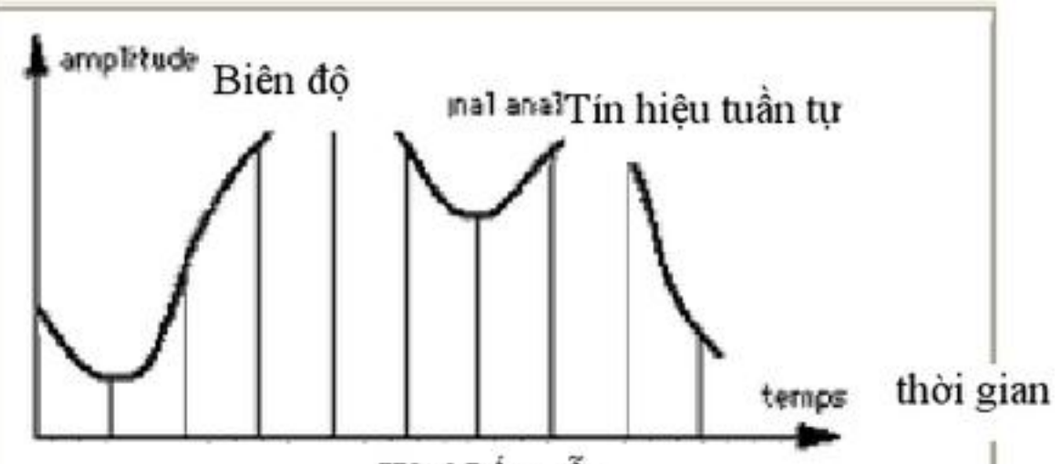
Một điểm ảnh được mã hóa tùy thuộc vào chất lượng của ảnh:
Ảnh đen trắng : sử dụng một bit để mã hóa một điểm : giá trị 0 cho điểm ảnh màu đen và 1 cho điểm ảnh màu trắng.
Ảnh gồm 256 mức xám: mỗi điểm được thể hiện bằng một byte (8 bits) ;

Ảnh màu: người ta chứng minh rằng một màu là sự phối hợp của ba màu cơ bản là đỏ (Red), xanh lá (Green) và xanh dương (Blue). Vì thế một màu bất kỳ có thể được biểu biễn bởi biểu thức:
x = aR + bG +cB
Trong đó a, b, c là các lượng của các màu cơ bản. Thông thường một ảnh đẹp sẽ có lượng màu với giá trị từ 0 đến 255. Và như thế, một ảnh màu thuộc loại này được thể hiện bằng 3 ma trận tương ứng cho 3 loại màu cơ bản. Mỗi phần tử của mảng có giá trị của 8 bits. Chính vì thế cần có 24 bit để mã hóa cho một điểm ảnh màu.
Kích thước của các ảnh màu là đáng kể, vì thế người ta cần có phương pháp mã hóa để giảm kích thước của các ảnh.

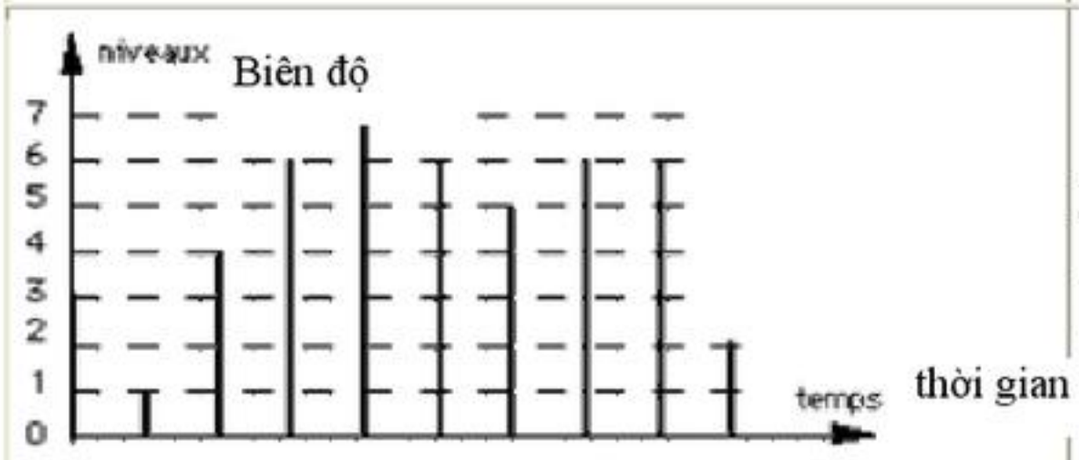
3.2.3 Số hóa âm thanh và phim ảnh

Dữ liệu kiểu âm thanh và phim ảnh thuộc kiểu tín hiệu tuần tự. Các tín hiệu tuần tự được số hóa theo cách thức sau đây:

1 - Lấy mẫu
Tín hiệu được lấy mẫu: với tần số f, ta đo biên độ của tín hiệu, như thế ta được một loạt các số đo.



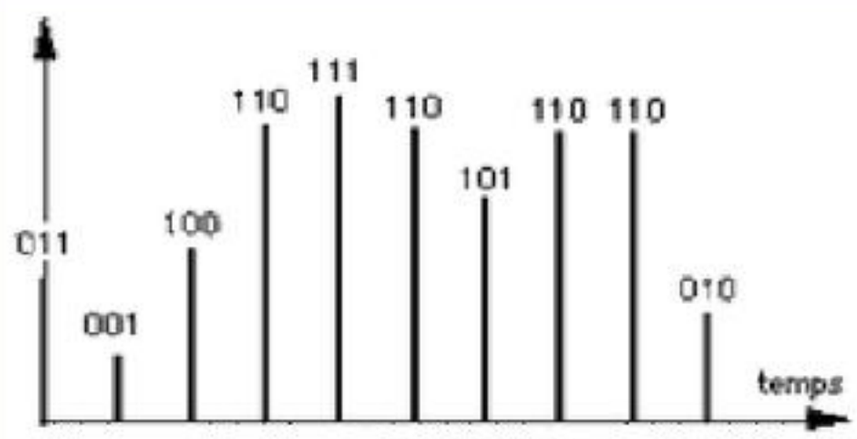
H3.6 Lấy mẫu



H3.7 Lượng hóa

2 - Lượng hóa
Ta xác định một thang đo với các giá trị là lũy thừa của 2 (2^p) và thực hiện việc lấy tương ứng các số đo vào giá trị thanh đo.

3- Mã hóa
Mỗi một giá trị sau đó được mã hóa thành các giá trị nhị phân và đặt vào trong các tập tin.
011001100110111110101110110010.....



H3.8 Mã hóa

Dung lượng tập tin nhận được phụ thuộc hoàn toàn vào tần số lấy mẫu f và số lượng bit dùng để mã hóa giá trị thang đo p (chiều dài mã cho mỗi giá trị).

3.3 Các loại kênh truyền

3.3.1 Kênh truyền hữu tuyến

Cáp thuộc loại kênh truyền hữu tuyến được sử dụng để nối máy tính và các thành phần mạng lại với nhau. Hiện nay có 3 loại cáp được sử dụng phổ biến là: Cáp xoắn đôi (twisted pair), cáp đồng trục (coax) và cáp quang (fiber optic). Việc chọn lựa loại cáp sử dụng cho mạng tùy thuộc vào nhiều yếu tố như: giá thành, khoảng cách, số lượng máy tính, tốc độ yêu cầu, băng thông

3.3.1.1 Cáp xoắn đôi (Twisted Pair)

Cáp xoắn đôi có hai loại: Có vỏ bọc (Shielded Twisted Pair) và không có vỏ bọc (Unshielded Twisted Pair). Cáp xoán đôi có vỏ bọc sử dụng một vỏ bọc đặc biệt quấn xung quanh dây dẫn có tác dụng chống nhiễu. Cáp xoán đôi trở thành loại cáp mạng được sử dụng nhiều nhất hiện nay. Nó hỗ trợ hầu hết các khoảng tốc độ và các cấu hình mạng khác nhau và được hỗ trợ bởi hầu hết các nhà sản xuất thiết bị mạng.

(a) đôi khó

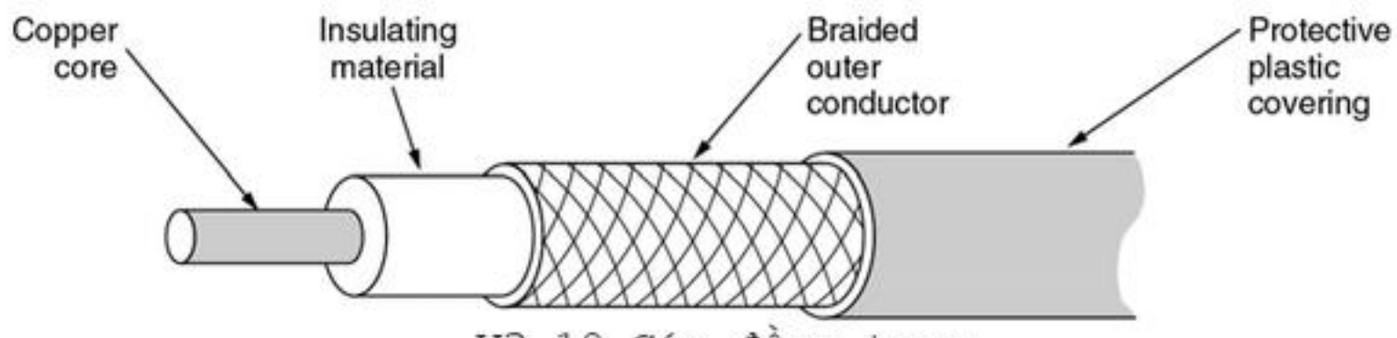
(b)

Các đặc tính của cáp xoán đôi là:

- Được sử dụng trong mạng token ring (cáp loại 4 tốc độ 16MBps), chuẩn mạng Ethernet 10BaseT (Tốc độ 10MBps), hay chuẩn mạng 100BaseT (tốc độ 100Mbps)
- Giá cả chấp nhận được.
- UTP thường được sử dụng bên trong các tòa nhà vì nó ít có khả năng chống nhiễu hơn so với STP.
- Cáp loại 2 có tốc độ đạt đến 1Mbps (cáp điện thoại) .
- Cáp loại 3 có tốc độ đạt đến 10Mbps (Dùng trong mạng Ethernet 10BaseT) (Hình a)
- Cáp loại 5 có tốc độ đạt đến 100MBps (dùng trong mạng 10BaseT và 100BaseT) (Hình b)
- Cáp loại 5E và loại 6 có tốc độ đạt đến 1000 MBps (dùng trong mạng 1000 BaseT)

3.3.1.2 Cáp đồng trục (Coaxial Cable)

Cáp đồng trục là loại cáp được chọn lựa cho các mạng nhỏ ít người dùng, giá thành thấp. Có cáp đồng trục gầy (thin coaxial cable) và cáp đồng trục béo (thick coaxial cable)



H3.10 Cáp đồng trục

- Cáp đồng trục gầy, ký hiệu RG-58AU, được dùng trong chuẩn mạng Ethernet 10Base2.

H3.11 Cáp đồng trục gầy

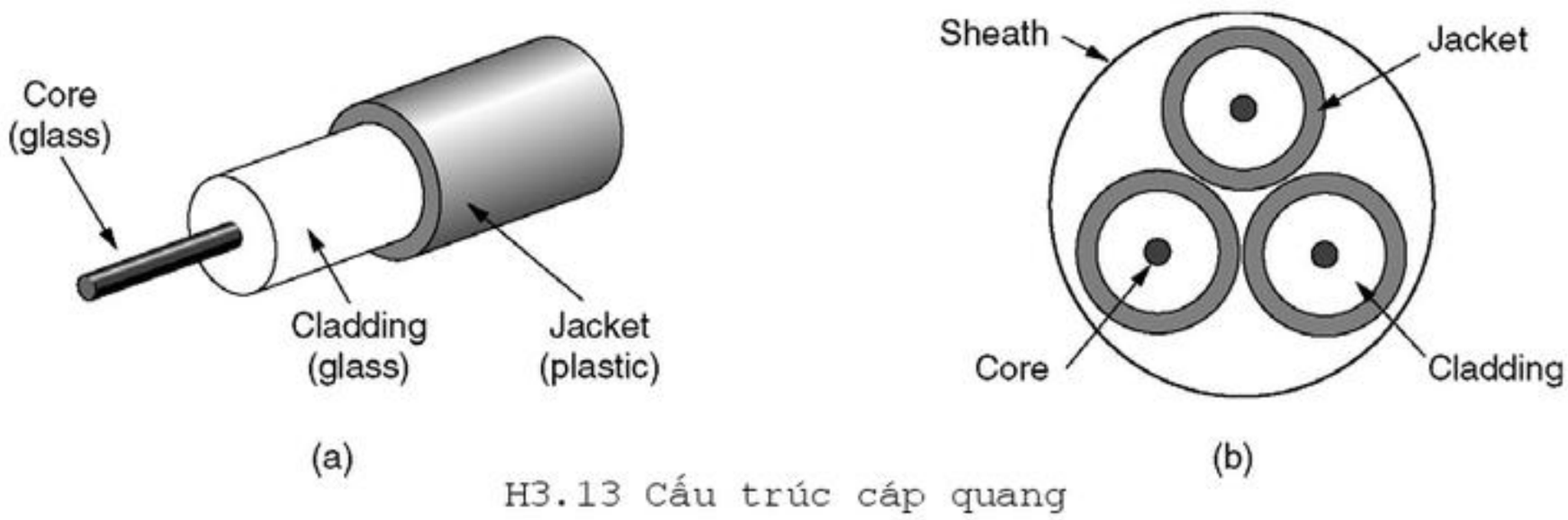
- Cáp đồng trục béo, ký hiệu RG-11, được dùng trong chuẩn mạng 10Base5

Các loại đầu nối được sử dụng với cáp đồng trục gầy là đầu nối chữ T (T connector), đầu nối BNC và thiết bị đầu cuối (Terminator)

H3.12 Đầu nối chữ T và BNC

3.3.1.3 Cáp quang (Fiber Optic)

Cáp quang truyền tải các sóng điện từ dưới dạng ánh sáng. Thực tế, sự xuất hiện của một sóng ánh sáng tương ứng với bit "1"và sự mất ánh sáng tương ứng với bit "0". Các tín hiệu điện tử được chuyển sang tín hiệu ánh sáng bởi bộ phát, sau đó các tín hiệu ánh sáng sẽ được chuyển thành các sung điện tử bởi bộ nhận. Nguồn phát quang có thể là các đèn LED (Light Emitting Diode) cổ điển, hay các diod laser. Bộ dò ánh sáng có thể là các tế bào quang điện truyền thống hay các tế bào quang điện dạng khối.

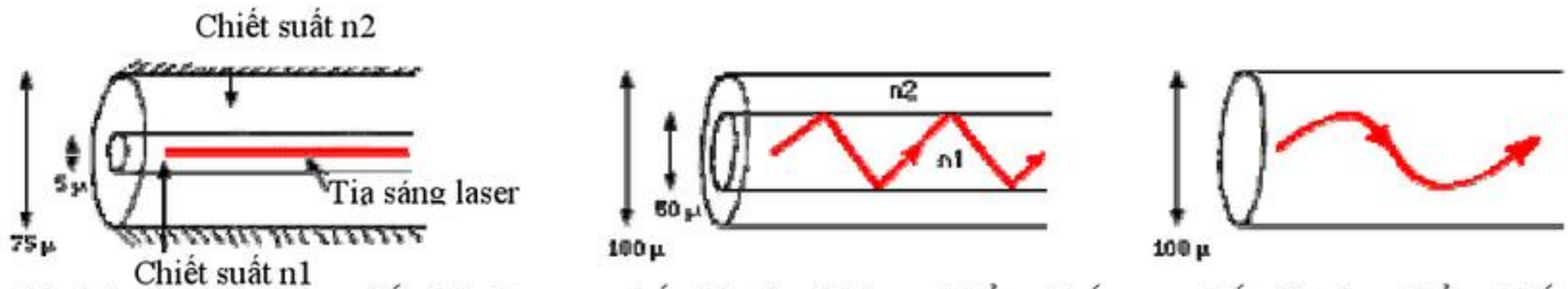


H3.13 Cấu trúc cáp quang

Sự lan truyền tín hiệu được thực hiện bởi sự phản xạ trên bề mặt. Thực tế, tồn tại 3 loại cáp quang.

- Chế độ đơn: một tia sáng trên đường truyền tải
- Hai chế độ còn lại gọi là chế độ đa: nhiều tia sáng cùng được truyền song song nhau

Trong chế độ đơn, chiết suất n2 > n1. Tia laser có bước sóng từ 5 đến 8 micromètres được tập trung về một hướng. Các sợi loại này cho phép tốc độ bit cao nhưng khó xử lý và phức tạp trong các thao tác nối kết.



H3.14 Cáp quang chế độ đơn - chế độ đa không thẩm thấu - chế độ đa thẩm thấu

- Chế độ đa không thẩm thấu

Các tia sáng di chuyển bằng cách phản xạ giữa bề mặt của 2 môi trường có chiết suất khác nhau (n2>n1) và mất nhiều thời gian hơn để các sóng di chuyển so với chế độ đơn. Độ suy giảm đường truyền từ 30 dB/km đối với các loại cáp thủy tinh và từ 100 dB/km đối với loại cáp bằng chất dẽo.

- Chế độ đa bị thẩm thấu

Chiết suất tăng dần từ trung tâm về vỏ của ống. Vì thế sự phản xạ trong trường hợp này thì rất nhẹ nhàng.

Từ cách đây nhiều năm người ta có thể thực hiện đa hợp trên cùng một sợi quang nhiều thông tin bằng cách dùng các sóng có độ dài khác nhau. Kỹ thuật này được gọi là WDM (Wavelength Division Multiplexing).

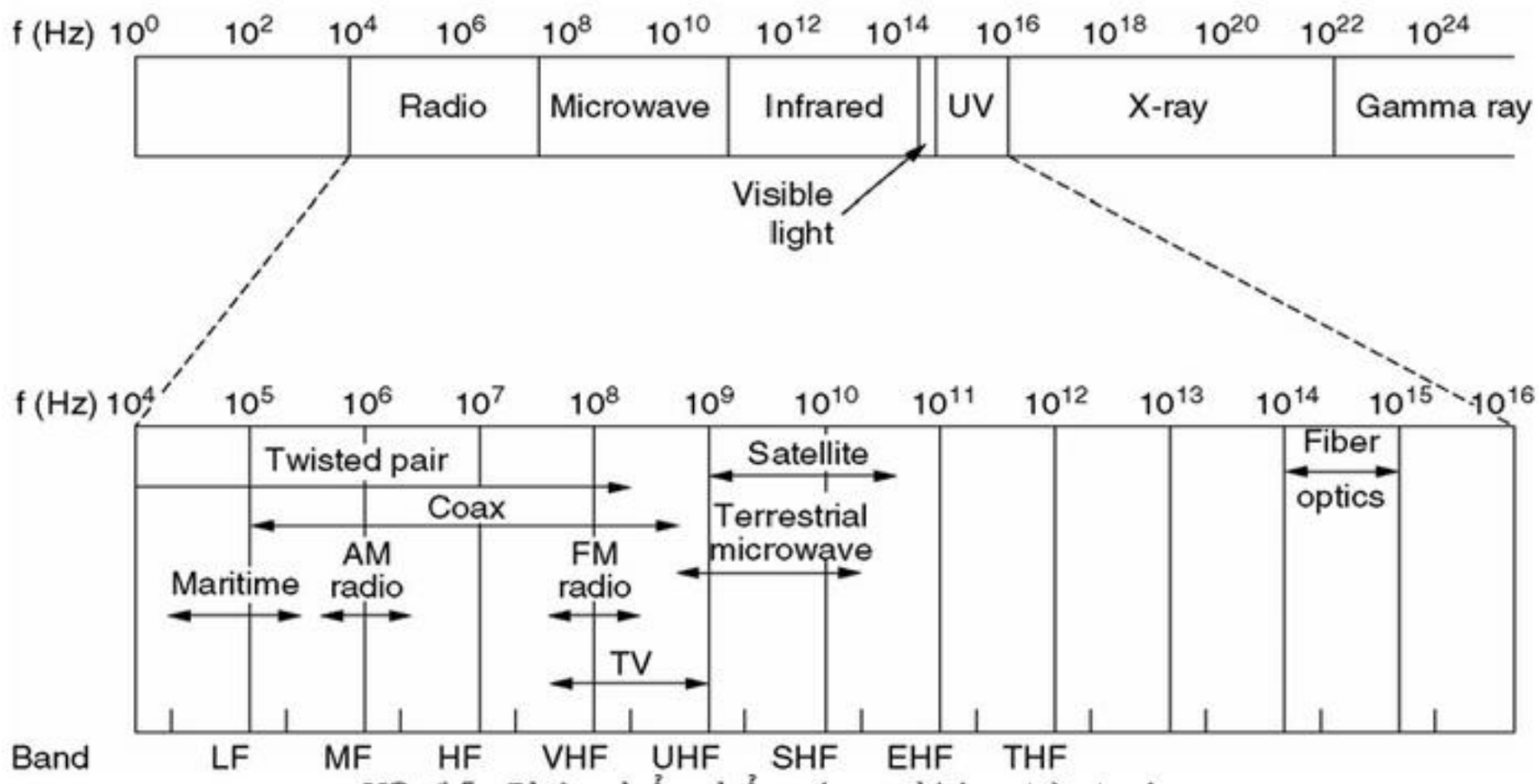
3.3.2 Kênh truyền vô tuyến

Kênh truyền vô tuyến thì thật sự tiện lợi cho tất cả chúng ta, đặc biệt ở những địa hình mà kênh truyền hữu tuyến không thể thực hiện được hoặc phải tốn nhiều chi phí (rừng rậm, hải đảo, miền núi). Kênh truyền vô tuyến truyền tải thông tin ở tốc độ ánh sáng.

Gọi:

- c là tốc độ ánh sáng,
- f là tần số của tín hiệu sóng
- λ là độ dài sóng. Khi đó ta có

$$c = \lambda f$$



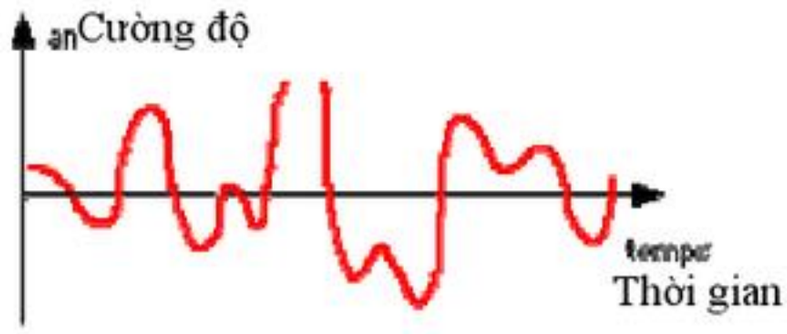
H3.15 Phân bố phổ sóng điện từ trên

Tín hiệu có độ dài sóng càng lớn thì khoảng cách truyền càng xa mà không bị suy giảm, ngược lại những tín hiệu có tần số càng cao thì có độ phát tán càng thấp.

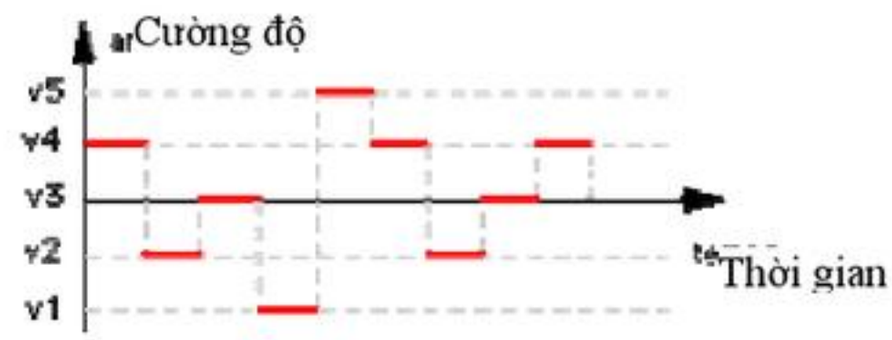
Hình H3.15 mô tả phổ của sóng điện tử được dùng cho truyền dữ liệu. Khoảng tần số càng cao càng truyền tải được nhiều thông tin.

3.4 Đặc điểm kênh truyền

Phương tiện thường được dùng để truyền tải dữ liệu (các bits 0,1) từ thiết bị truyền đến thiết bị nhận trên một kênh truyền nhận vật lý là các tín hiệu tuần tự hay tín hiệu số.



H3.16 Tín hiệu tuần tự : được biểu diễn bằng một độ lớn vật lý thay đổi một cách liên tục

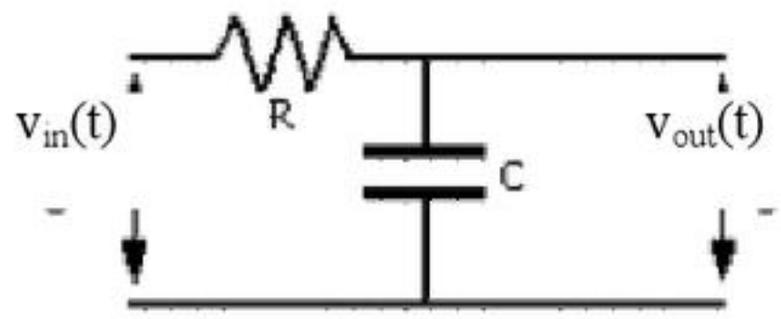


H3.17 Tín hiệu số : được biểu diễn bởi một độ lớn vật lý với một vài giá trị xác định riêng rời

3.4.1 Truyền tải tín hiệu sóng dạng hình sin

Sóng dạng hình sin, không kết thúc hoặc suy giảm sau một khoảng thời gian là dạng tín hiệu tuần tự đơn giản nhất, dễ dàng tạo ra được. Hơn thế nó còn đặc biệt được chú ý đến bởi yếu tố sau: **bất kỳ một dạng tín hiệu nào cũng có thể được biểu diễn lại bằng các sóng hình sin**. Yếu tố này được rút ra từ một nghiên cứu cụ thể nó cho phép chúng ta có thể định nghĩa một vài đặc điểm của kênh truyền vật lý.

Xem xét một kênh truyền, giả sử rằng các điểm nối kết là trực tiếp, không có ngắt quảng, được hình thành từ hai sợi kim loại. Một đoạn của kênh truyền được xem như một đèn 4 cực gồm một điện trở R và một tụ điện C.



H3.18 Mô hình kênh truyền dữ liệu vật lý

Tín hiệu hình sin được áp vào giữa các cực (giữa 2 sợi dây) được tín theo biểu thức:

$$v_{in}(t) = \mathbf{V_{in}} \sin wt$$

Trong đó

- V_{in} : là hiệu điện thế cực đại;
- w : nhịp ; f = w/2pi : là **tần số**;
- T = 2pi/w = 1/f : là **chu kỳ**.

Tín hiệu đầu ra sẽ là:

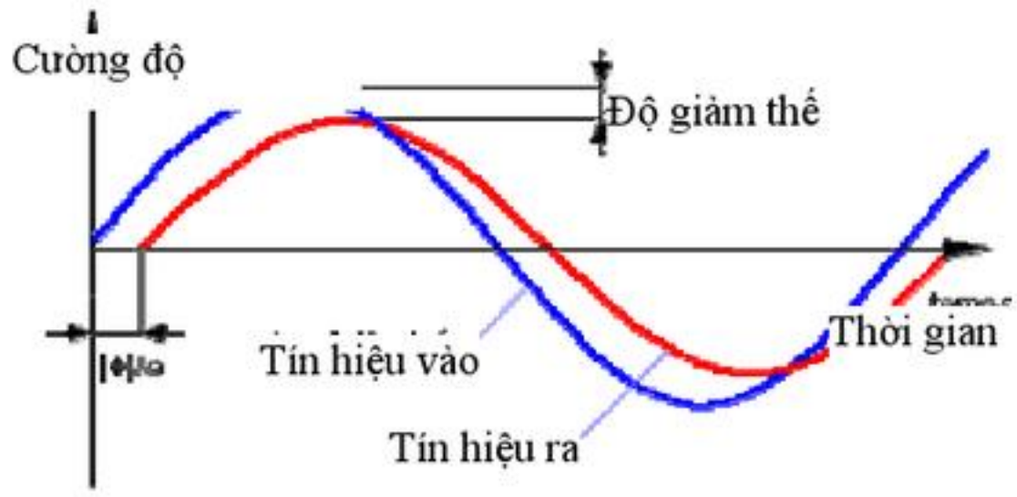
$$v_{out}(t) = V_{out} \sin (wt + F)$$

- Với : F : là độ trễ pha.

Mức điện thế ngỏ ra tùy thuộc vào điện thế ngỏ vào và đặc điểm vật lý của đèn bốn cực. Các luật trường điện tử chứng minh rằng trong trường hợp đơn giản nhất ta có:

$$V_{out}/V_{in} = (1 + R^2C^2w^2)^{-1/2} \qquad F = \text{atan}(-RC\ w)$$

Ta nhận thấy rằng điện thế ngỏ ra V_{out} thì yếu hơn điện thế ngỏ vào V_{in}. Ta nói có một sự giảm thế và một sự lệch pha F giữa hiệu điện thế ngỏ vào và hiệu điện thế ngỏ ra. Nếu ta chồng 2 sóng điện thế ngỏ vào và điện thế ngỏ ra trong một sơ đồ thời gian, ta có kết quả như sau:

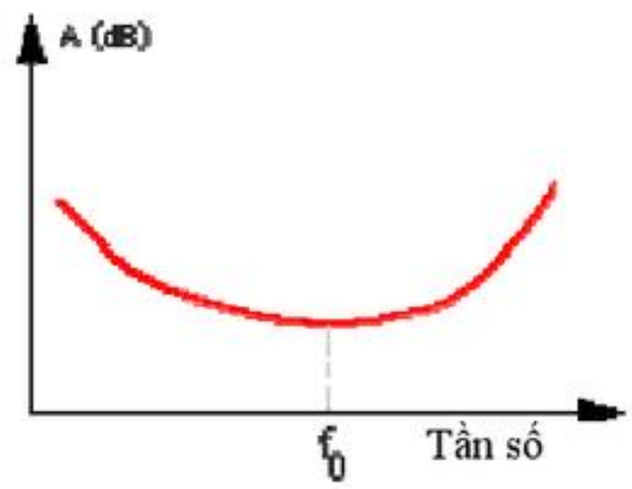


H3.19 Sự trễ pha và giảm thế của tín hiệu ngỏ ra

Độ suy giảm trên kênh truyền A của tín hiệu là một tỷ lệ về công suất P_{in}/P_{out} của tín hiệu phát P_{in} và tín hiệu nhận được P_{out}. Mỗi công suất được tính với đơn vị là watts. Ta biểu diễn độ suy giảm bằng đơn vị decibel:

$$A(w) = 10 \log_{10}(P_{in}/P_{out})$$

Hình bên mô tả đồ thị biểu diễn mối tương quan giữa độ suy giảm và **tần số** sóng phát trên một kênh truyền nào đó.
Ta thấy rằng **tần số** tối ưu nhất là f_0 và như thế, nếu chúng ta muốn độ suy giảm là nhỏ nhất thì chúng ta sẽ chọn sóng phát hình sin có **tần số** càng gần f_0 càng tốt.



H3.20 Tương quan giữa tần số và độ suy giảm của tín hiệu

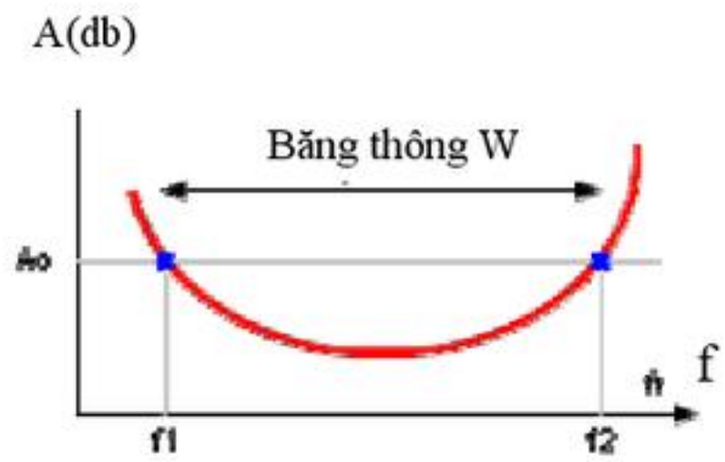
3.4.2 Truyền tín hiện bất kỳ

Lý thuyết toán Fourrier đã chứng minh rằng bất kỳ một tín hiện nào cũng có thể xem như được tạo thành từ một tổng của một số hữu hạn hoặc vô hạn các sóng hình sin. Không đi sâu vào chứng minh ta có kết quả sau:

- Một tín hiệu bất kỳ x(t) thì có thể phân tích thành một tập hợp các tín hiệu dạng sóng hình sin.
- Nếu là tín hiệu tuần hoàn, thì ta có thể phân tích nó thành dạng một chuỗi Fourier. Thuật ngữ chuỗi ở đây ý muốn nói đến một loạt các sóng hình sin có **tần số** khác nhau như là các bội số của **tần số** tối ưu f_0.
- Nếu tín hiệu không là dạng tuần hoàn, thì ta có thể phân tích nó dưới dạng một bộ Fourier ; với các sóng hình sin có **tần số** rời rạc.

3.4.3 Băng thông của một kênh truyền (Bandwidth)

Bởi vì một tín hiệu bất kỳ có thể được xem như là một sự kết hợp của một chuỗi các sóng hình sin, nên ta có thể xem rằng, sự truyền tải một tín hiệu bất kỳ tương đương với việc truyền tải các sóng hình sin thành phần. Vì **tần số** của chúng là khác nhau, chúng có thể đến nơi với độ suy giảm là khác nhau, một trong số chúng có thể không còn nhận ra được. Nếu ta định nghĩa một ngưỡng còn "nghe" được A_0, thì tất cả các tín hiệu hình sin có **tần số** nhỏ hơn f_1 được xem như bị mất. Tương tự các tín hiệu có **tần số** lớn hơn f_2 cũng được xem là bị mất. Những tín hiện có thể nhận ra được ở bên nghe là các tín hiệu có **tần số** nằm giữa f_1 và f_2. Khoản **tần số** này được gọi là băng thông của một kênh truyền.



H3.21 Băng thông của kênh truyền

Nói một các khác, với một tín hiệu phức tạp bất kỳ, tín hiệu này sẽ truyền tải được nếu như **tần số** của các sóng hình sin thành phần của nó có **tần số** nằm trong khoảng băng thông của kênh truyền. Chúng ta cũng nhận thấy rằng, băng thông càng lớn thì càng có nhiều tín hiệu được truyền đến nơi. Chính vì thế chúng ta thường quan tâm đến các kênh truyền có băng thông rộng..
Ví dụ :độ rộng băng thông của kênh truyền điện thoại là 3100 Hz vì các tín hiệu âm thanh có thể nghe được nằm ở khoảng **tần số** từ 300 Hz đến 3400 Hz.

3.4.4 Tần số biến điệu và tốc độ dữ liệu (Baund rate and bit rate)

Một thông điệp thì được hình thành từ một chuỗi liên tiếp các tín hiệu số hay tuần tự. Mỗi tín hiệu có độ dài thời gian là **t**. Các tín hiệu này được lan truyền trên kênh truyền với vận tốc 10^8 m/s trong kênh truyền cáp quang hay 2.10^6 m/s trong kênh kim loại. Chúng ta thấy rằng tốc độ lan truyền không phải là yếu tố quyết định. Yếu tố quyết định chính là nhịp mà ta đặt tín hiệu lên kênh truyền. Nhịp này được gọi là tần số biến điệu:

R = 1/t (đơn vị là bauds).

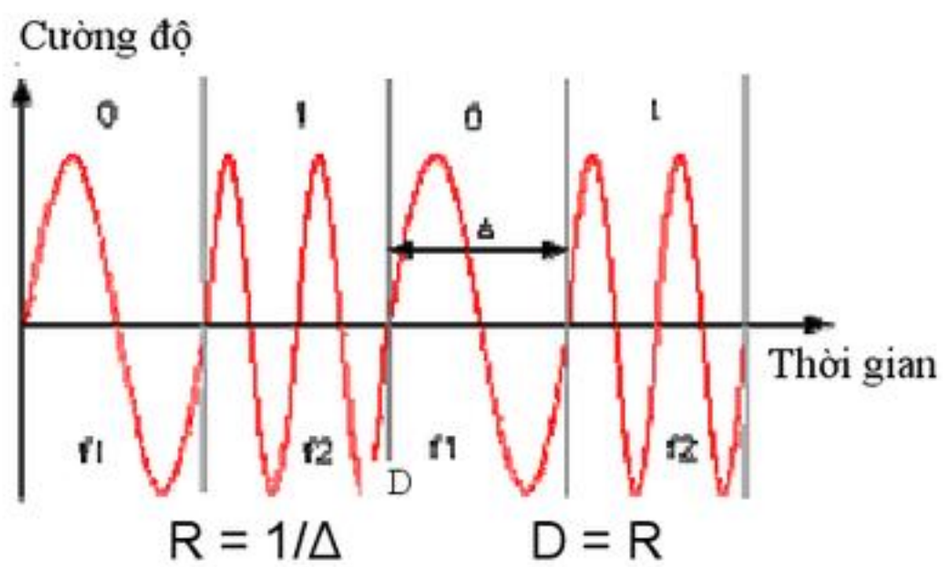
Nếu thông điệp dạng nhị phân, và mỗi tín hiện chuyển tải n bit, khi đó ta có tốc độ bit được tính như sau:

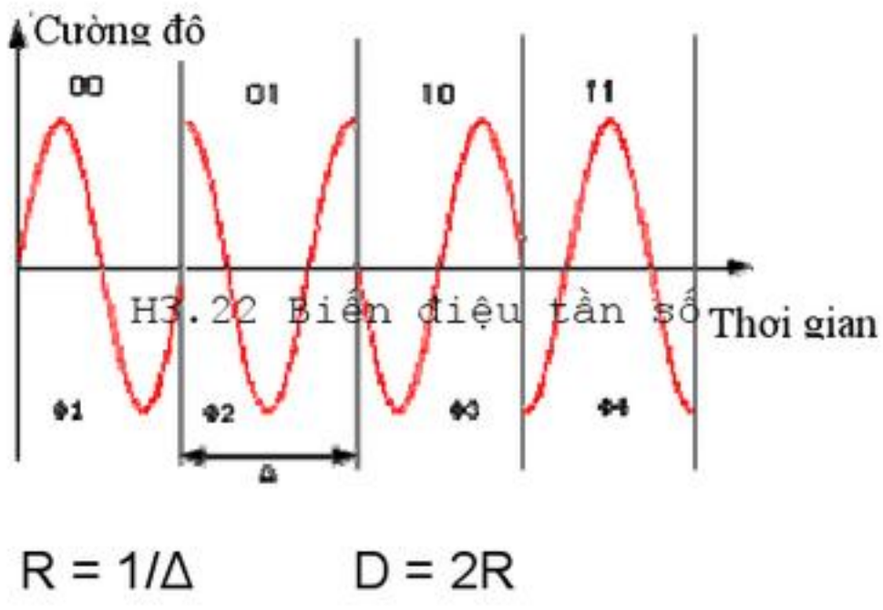
D = nR (đơn vị là bits/s)

Giá trị này thể hiện nhịp mà ta đưa các bit lên đường truyền.
Ví dụ : Cho hệ thống có R = 1200 bauds và D = 1200 bits/s. Ta suy ra một tín hiện cơ bản chỉ chuyển tải một bit.

Một số ví dụ về tần số biến điệu và tốc độ dữ liệu:

Ví dụ 1 : Truyền tải các dữ liệu số bằng các tín hiệu tuần tự.
Ta sử dụng hai kiểu tín hiệu tuần tự, mỗi loại có độ dài sóng D, sóng thứ nhất có tần số f_1, sóng thứ hai có tần số f_2 (gấp đôi tần số f_1). Cả hai tín hiệu đều có thể nhận được ở ngõ ra. Ta qui định rằng tín hiệu thứ nhất truyền bit "0" và tín hiệu thứ hai truyền bit "1". Nhịp được sử dụng để đưa các tín hiệu lên đường truyền bằng với nhịp truyền các bit bởi vì mỗi tín hiệu thì truyền một bit. Sự phân biệt giữa tín hiệu 0 và 1 dựa trên sự khác biệt về tần số của 2 tín hiệu sin. Sự mã hóa này được gọi là biến điệu tần số.



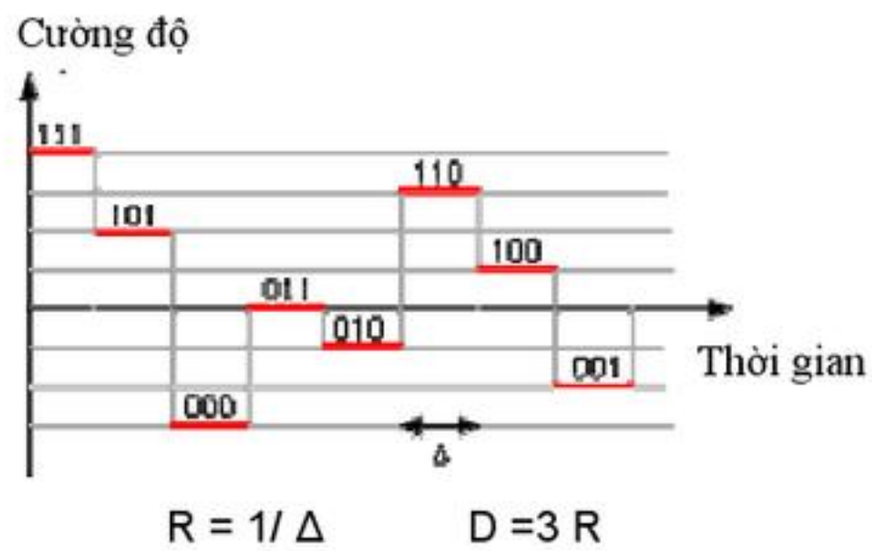


H3.22 Biến điệu tần số

Ví dụ 2 : Truyền dữ liệu số bởi các tín hiệu tuần tự. Trong trường hợp này ta sử dụng 4 loại tín hiệu hình sin lệch pha nhau Pi/4. Mỗi loại tín hiệu có thể

vận chuyển 2 bits hoặc 00, 01, 10 hay 11. Với cách thức như thế, tốc độ dữ liệu sẽ gấp đôi tần số biến điệu.
Sự phân biệt giữa các tín hiệu trong trường hợp này dự vào pha của tín hiệu. Ta gọi là biến điệu pha.

Ví dụ 3 : Truyền tải các dữ liệu số bằng các tín hiệu số.
Ta sử dụng 8 tín hiệu số cùng độ dài nhưng có biên độ khác nhau. Mỗi tín hiệu truyền tải 3 bits bởi chúng có thể đại diện cho 8 sự kết hợp khác nhau của 3 bit. Sự phân biệt giữa các tín hiệu trong trường hợp này dựa vào biên độ của các tín hiệu. Ta gọi là biến điệu biên độ.
Để có được một tốc độ truyền dữ liệu cao nhất, ta tìm cách cải thiện tốc độ bit. Bởi vì D = n R, ta có thể tăng tốc độ bit bằng cách tăng một trong các yếu tố sau:



H3.24 Biến điệu biên độ

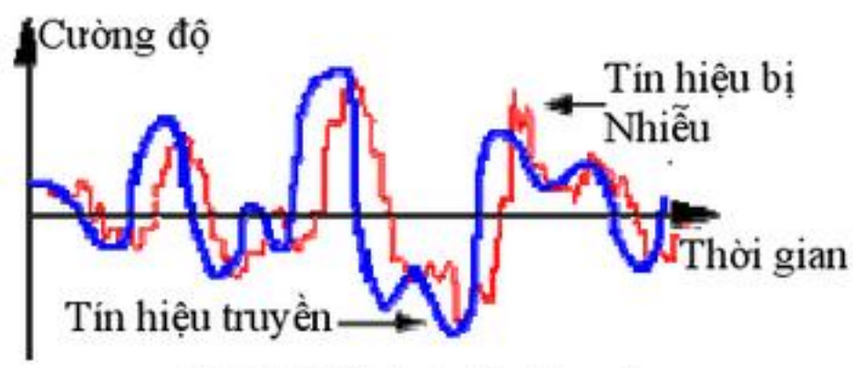
- Hoặc tăng n (số bit truyền tải bởi một tín hiệu), tuy nhiên nhiễu là một rào cản quan trọng.
- Hoặc R(tần số biến điệu), tuy nhiên chúng ta cũng không thể vượt qua tần số biến điệu cực đại R_{max} .

Kết quả sau đây đã được chứng minh bởi Nyquist (1928) xác định mối ràng buộc giữa tần số biến điệu cực đại và băng thông của kênh truyền W:

- $R_{max} = 2$ W,
- Kết quả này được tính toán trên lý thuyết, trong thực tế thì $R_{max} = 1,25$ W

3.4.5 Nhiễu và khả năng kênh truyền

Nhiễu bao gồm các tín hiệu ký sinh chúng chồng lên các tín hiệu được truyền tải và chúng làm cho các tín hiệu này bị biến dạng



H3.25 Nhiễu trên kênh truyền

Chúng ta có thể phân biệt thành 3 loại nhiễu :

- Nhiễu xác định: phụ thuộc vào đặc tính kênh truyền
- Nhiễu không xác định
- Nhiễu trắng từ sự chuyển động của các điện tử

Nhiễu phiền tối nhất dĩ nhiên là loại nhiễu không xác định. Chúng có thể làm thay đổi tín hiệu vào những khoảng thời gian nào đó làm cho bên nhận khó phân biệt được đó là bit "0" hay bit "1". Chính vì thế mà công suất của tín hiệu nên lớn hơn nhiều so với công suất của nhiễu. Tỷ lệ giữa công suất tín hiệu và công suất nhiễu tính theo đơn vị décibels được biểu diễn như sau :

$$S/B = 10\log_{10}(P_S(Watt)/P_B(Watt))$$

Trong đó P_S và P_B là công suất của tín hiệu và công suất của nhiễu.
Định lý Shannon (1948) giải thích tầm quan trọng của ty lệ S/B trong việc xác định số bit tối đa có thể chuyên chở bởi một tín hiệu như sau:

$$n_{max} = \log_2 \sqrt{1+\frac{P_S}{P_B}}$$

Kết hợp với định lý của Nyquist, ta có thể suy ra tốc độ bit tối đa của một kênh truyền được tính theo công thức sau:

$$C = D_{max} = R_{max} n_{max} = 2W \log_2 \sqrt{1+\frac{P_S}{P_B}} = W \log_2 [1+\frac{P_S}{P_B}]$$

C được gọi là khả năng của kênh truyền , xác định tốc độ bit tối đa có thể chấp nhận được bởi kênh truyền đó.

Ví dụ : Kênh truyền điện thoại có độ rộng băng thông là W = 3100 Hz tỷ lệ S/B = 20 dB. Từ đó ta tính được khả năng của kênh truyền điện thoại là :

C = 20,6 Kbits/s .

3.4.6 Giao thông (Traffic)

Giao thông là một khái niệm liên quan đến sự sử dụng một kênh truyền tin. Giao thông cho phép biết được mức độ sử dụng kênh truyền từ đó có thể chọn một kênh truyền phù hợp với mức độ sử dụng hiện tại.

Để đánh giá giao thông, ta có thể xem một cuộc truyền tải hay một cuộc giao tiếp là một **phiên giao dịch** (session) với độ dài trung bình là T (đơn vị là giây). Cho N_c là số lượng phiên giao dịch trung bình trên một giờ. Mật độ giao thông E được tính theo biểu thức sau :

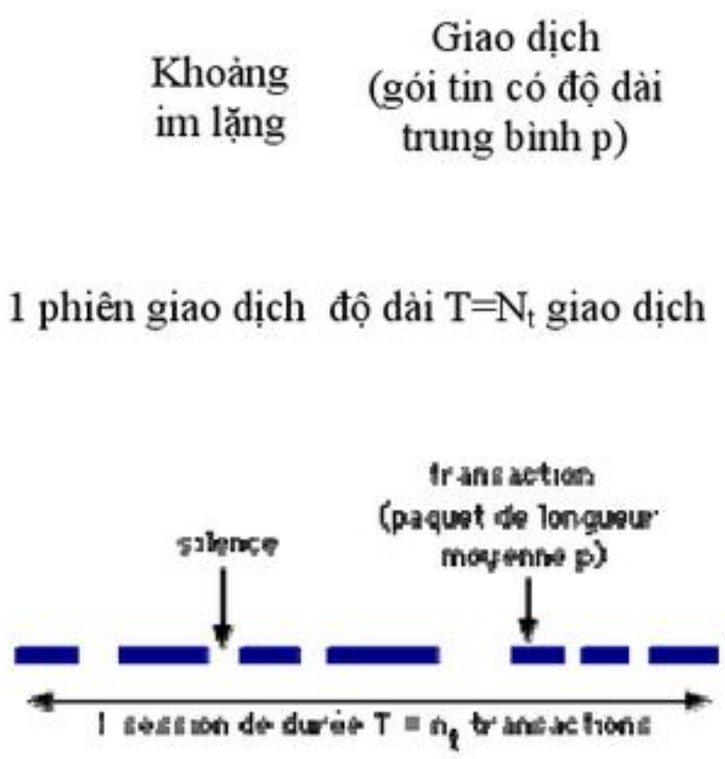
$$E = T N_c / 3600$$

Nói cách khác, mật độ giao thông là đại lượng dùng để đo mức độ sử dụng kênh truyền trong một giây.

Thực tế, khi phân tích kỹ hơn ta sẽ thấy rằng trong một phiên giao dịch sẽ chứa nhiều khoảng im lặng (không dùng kênh truyền), ta có thể phân biệt thành 2 loại phiên giao dịch sau:

- Các phiên giao dịch mà ở đó thời gian sử dụng T được sử dụng hết.
- Các phiên giao dịch mà ở đó thời gian T có chứa các khoảng im lặng.

Trong trường hợp thứ hai, mật độ giao thông thì không phản ánh đúng mức độ bận rộn thật sự của kênh truyền. Ta tách một phiên giao dịch thành nhiều **giao dịch** (transaction) với độ dài trung bình là p bit, cách khoảng nhau bởi những khoảng im lặng. Giả sử N_t là số giao dịch trung bình trong một phiên giao dịch.



Gọi **D** là tốc độ bit của kênh truyền, tốc độ bit thật sự **d** trong trường hợp này là:

$$d = \frac{N_t p}{T}$$

và tần suất sử dụng kênh truyền được định nghĩa bởi tỷ số: $\theta = \frac{d}{D}$

Ví dụ : Trong một tính toán khoa học từ xa, người dùng giao tiếp với máy tính trung tâm. Cho :

- p = 900 bits, N_t = 200, T = 2700 s, N_c = 0.8, D = 1200 b/s.
- Khi đó
  - Mật độ giao thông trung bình là E = 0.6
  - Tầng suất sử dụng kênh truyền θ = 0.05

3.5 Mã hóa đường truyền (Line Coding)

Sau khi số hóa thông tin, vấn đề chúng ta phải quan tâm kế tiếp là cách truyền tải các bit "0" và "1". Ta có thể sử dụng tín hiệu số hoặc tín hiệu tuần tự để truyền tải các bit "0", "1". Công việc này còn được gọi là mã hóa đường truyền (line coding).

3.5.1 Mã hóa đường truyền bằng tín hiệu số

Trong phương pháp này ta sử dụng một tín hiệu số cho bit "0" và một tín hiệu số khác cho bit "1". Có nhiều cách thức để thực hiện điều này. Một số phương pháp mã hóa phổ biến như:

- Mã NRZ (Non Return to Zero), RZ (Return to Zero), lưỡng cực (bipolar) NRZ và RZ:

a) NRZ : Điện thế mức 0 để thể hiện bit 0 và điện thế khác không V_0 cho bit "1"

b) RZ : Mỗi bit "1" được thể hiện bằng một chuyển đổi điện thế từ V_0 về 0.

c) Lưỡng cực NRZ : Các bit "1" được mã hóa bằng một điện thế dương, sau đó đến một điện thế âm và tiếp tục như thế.

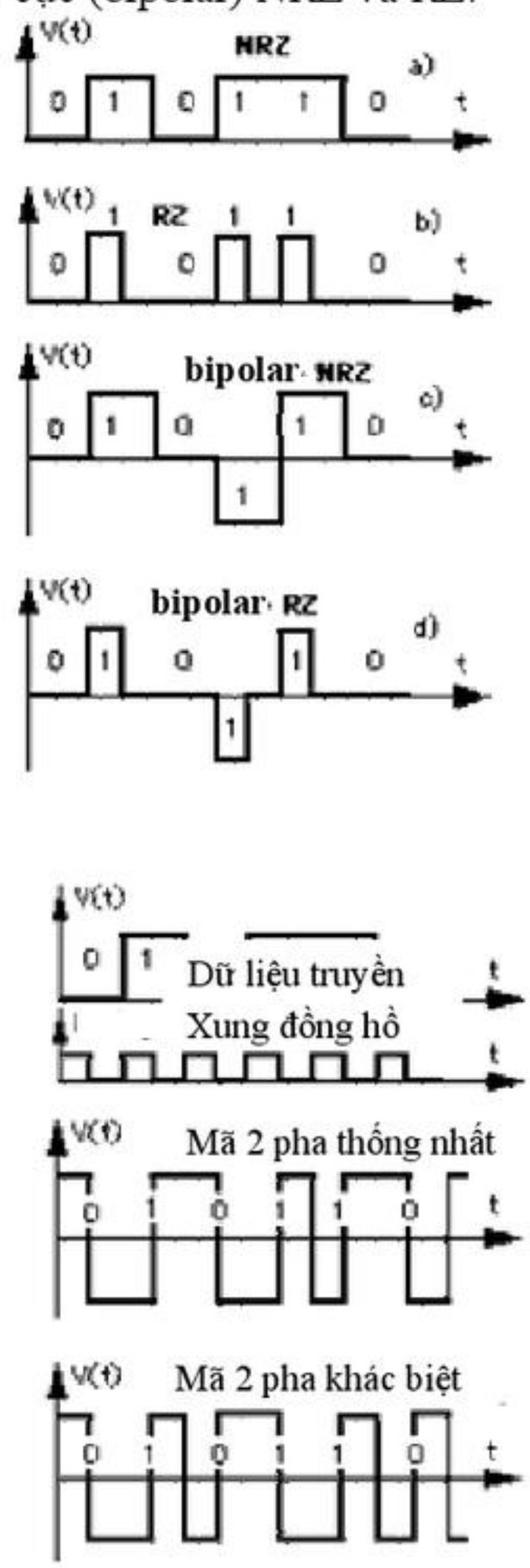
d) Lưỡng cực RZ : Mỗi bit "1" được thể hiện bằng một chuyển đổi từ điện thế khác không về điện thế không. Giá trị của điện thế khác không đầu tiên là dương sau đó là âm và tiếp tục chuyển đổi qua lại như thế.

- Mã hóa hai pha (biphase):

Các mã loại này được định nghĩa so với phương pháp mã NRZ như sau:

a) Mã hai pha thống nhất đôi khi còn gọi là mã Manchester: bit "0" được thể hiện bởi một chuyển đổi từ tín hiệu dương về tín hiệu âm và ngược lại một bit "1" được thể hiện bằng một chuyển đổi từ tín hiệu âm về tín hiệu dương.

b) Mã hai pha khác biệt : Nhảy một pha 0 để thể hiện bit "0" và nhảy một pha Pi để thể hiện bit "1".



H3.26 Mã hóa đường truyền bằng tín hiệu số

3.5.2 Mã hóa đường truyền bằng tín hiệu tuần tự

Thông thường người ta sử dụng một sóng mang hình sin v(t) = V sin(t +) để mã hóa đường truyền. Trong đó ta thay đổi một số tham số để thể hiện các bit "0" và "1" :

- Thay đổi V, ta có biến điệu biên độ (Amplitude modulation)
- Thay đổi , ta có biến điệu tần số (Frequency modulation)
- Thay đổi , ta có biến điệu pha (Phase modulation)

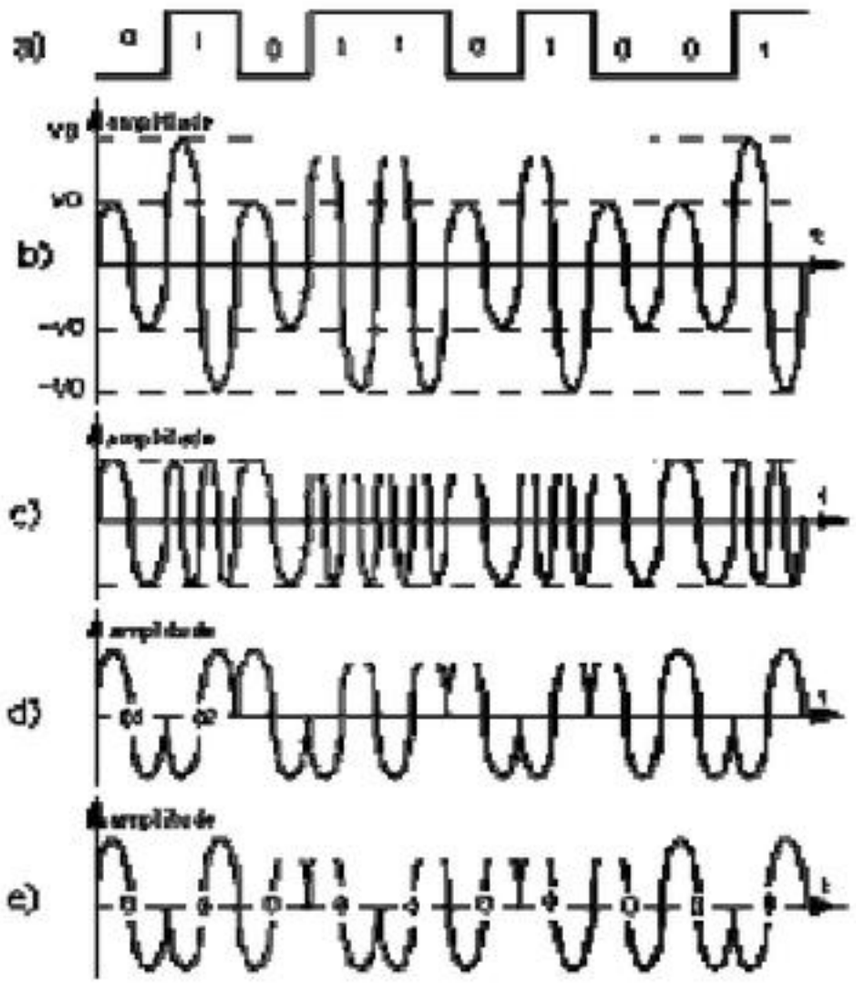
Bên truyền thực hiện quá trình mã hóa một bit thành tín hiệu tuần tự gọi là biến điệu (modulation). Ngược lại bên nhận, nhận được tín hiệu tuần tự phải giải mã thành một bit, gọi là hoàn điệu (demodulation).

a) Sử dụng tín hiệu số theo mã NRZ

b) Sử dụng biến điệu biên độ

c) Sử dụng biến điệu tần số
d) Sử dụng biến điệu pha

e) Sử dụng biến điệu pha



H3.26 Mã hóa đường truyền bằng tín hiệu hiệu tuần tự

Chương 4: Tầng liên kết dữ liệu (Data link layer)

Mục đích

Chương này nhằm giới thiệu cho người học những nội dung chủ yếu sau:

- Các chức năng cơ bản mà tầng liên kết dữ liệu đảm trách
- Vai trò của khung trong vấn đề xử lý lỗi đường truyền và các phương pháp xác định khung
- Giới thiệu các phương pháp phát hiện lỗi như Phương pháp kiểm tra chẵn lẽ, Phương pháp kiểm tra theo chiều dọc và Phương pháp kiểm tra phần dư tuần hoàn.
- Giới thiệu các giao thức điều khiển lỗi cho phép theo dõi tình trạng lỗi của dữ liệu gởi đi
- Giới thiệu các giao thức xử lý lỗi, chỉ ra các cách giải quyết trường hợp dữ liệu truyền đi bị lỗi.

Yêu cầu

Sau khi học xong chương này, người học phải có được các khả năng sau:

- Biện luận được vai trò của tầng liên kết dữ liệu trong vấn đề xử lý lỗi dữ liệu truyền nhận
- Trình bày được các phương pháp định khung đếm ký tự, phương pháp sử dụng byte là cờ và phương pháp sử dụng cờ đặc biệt
- Phân biệt được sự khác nhau giữa các chức năng phát hiện lỗi, điều khiển lỗi và xử lý lỗi của tầng hai.
- Cài đặt được cơ chế phát hiện lỗi theo các phương pháp kiểm tra chẵn lẽ, Phương pháp kiểm tra theo chiều dọc và Phương pháp kiểm tra phần dư tuần hoàn
- Cài đặt được các giao thức điều khiển lỗi Dừng và chờ, giao thức cửa sổ trượt
- Cài đặt được giao thức xử lý lỗi Go-Back-N và giao thức Selective Repeat
- Trình bày được ý tưởng cơ bản của giao thức HDLC

4.1 Chức năng của tầng liên kết dữ liệu

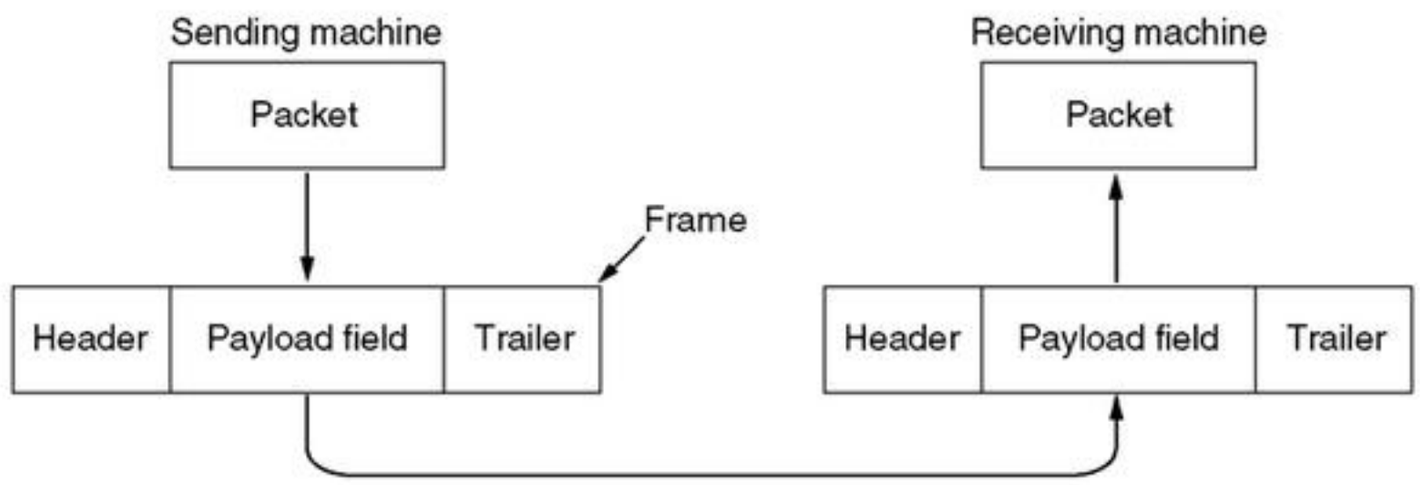
Tầng liên kết dữ liệu đảm nhận các chức năng sau:

- Cung cấp một giao diện được định nghĩa chuẩn cho các dịch vụ cung cấp cho tầng mạng.
- Xử lý lỗi đường truyền.
- Điều khiển luồng dữ liệu nhờ đó bên truyền nhanh không làm tràn dữ liệu bên nhận chậm

4.1.1 Các dịch vụ cơ bản của tầng liên kết dữ liệu

Nhiệm vụ của tầng liên kết dữ liệu là cung cấp các dịch vụ cho tầng mạng. Dịch vụ chính của tầng liên kết dữ liệu là truyền tải dữ liệu nhận được từ tầng mạng trên máy gởi đến tầng mạng trên máy nhận.

Để làm được điều này, tầng liên kết dữ liệu lấy các gói tin (Packet) mà nó nhận được từ tầng mạng và gói chúng vào trong các khung (frame) để truyền đi. Mỗi khung chứa phần tiêu đề (Header), thông tin cần truyền đi (Payload field) và thông tin theo dõi khác (Trailer).



H4.1 Vị trí và chức năng của tầng liên kết dữ liệu

Có 3 dịch vụ cơ bản mà tầng liên kết dữ liệu thường cung cấp là:

- Dịch vụ không nối kết không báo nhận (unacknowledged connectionless service), thường được sử dụng trong mạng LAN.
- Dịch vụ không nối kết có báo nhận (acknowledged connectionless service), thường dùng cho mạng không dây.
- Dịch vụ nối kết định hướng có báo nhận (acknowledged connection-oriented service), thường dùng trong mạng WANs.

4.1.2 Xử lý lỗi

Để có thể truyền tải được dữ liệu nhận từ tầng mạng đến máy nhận, tầng liên kết dữ liệu phải sử dụng các dịch vụ được cung cấp bởi tầng vật lý. Tất cả những gì tầng vật lý thực hiện là nhận một chuỗi các bits thô và cố gắng truyền chúng đến máy đích. Tầng vật lý không đảm bảo về độ tin cậy của các bits được truyền đi. Số lượng bits đến nơi nhận có thể nhiều, ít, hay bằng số bits đã gởi đi, thậm chí giá trị của chúng cũng có thể khác với giá trị mà chúng đã được gởi đi. Chính vì thế mà tầng liên kết dữ liệu phải dò tìm và xử lý các lỗi trên dữ liệu nhận được.

4.1.3 Định khung

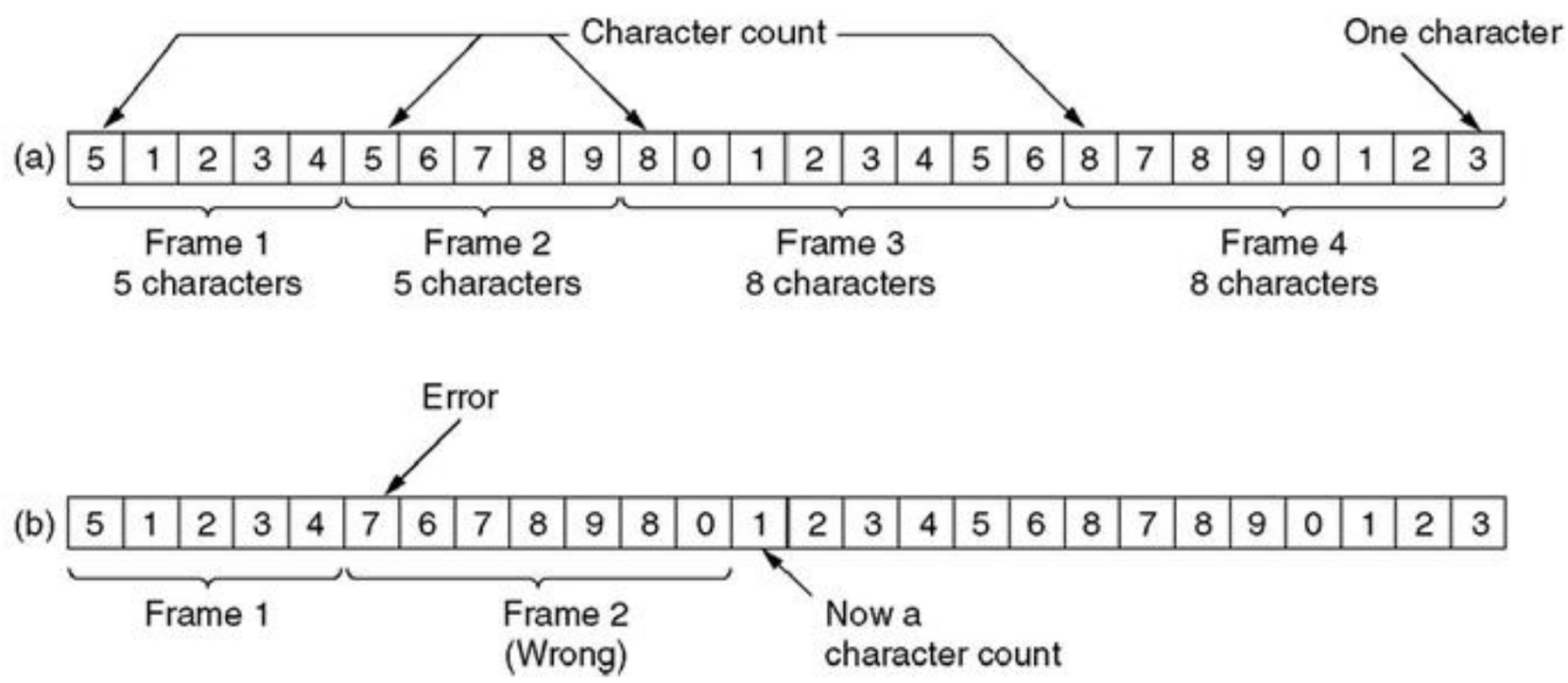
Như đã nói ở phần trên, đơn vị truyền tin của tầng liên kết dữ liệu là các khung. Vấn đề đặt ra là làm sao bên nhận biết được điểm bắt đầu và điểm kết thúc của khung. Chính vì vậy mà tầng liên kết dữ liệu cần thiết phải qui định khuôn dạng của khung mà mình sử dụng. Có 3 phương pháp để định khung phổ biến sau:

- Đếm ký tự (Character count)
- Sử dụng các bytes làm cờ hiệu và các bytes độn (Flag byte with byte stuffing)

- Sử dụng cờ bắt đầu và kết thúc khung cùng với các bit độn (Starting and ending flags with bit stuffing)
-

4.1.3.1 Phương pháp đếm ký tự (Character Count)

Phương pháp này sử dụng một trường trong phần tiêu đề để mô tả số lượng các ký tự có trong khung. Bất lợi của phương pháp này là nếu một ký tự đếm của một khung nào đó bị lỗi sẽ làm cho các khung phía sau không thể xác định được. Phương pháp này vì thế mà ít được sử dụng.

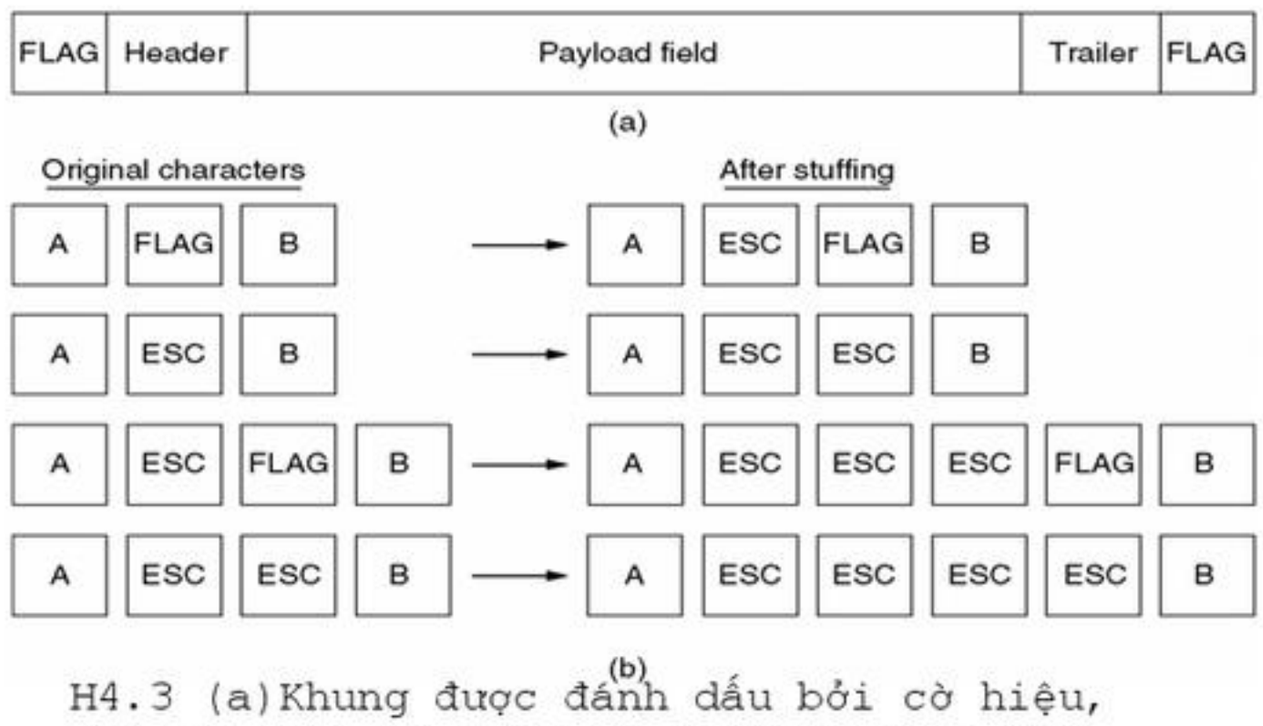


H4.2 Định khung bằng phương pháp đếm ký tự

4.1.3.2 Phương pháp sử dụng byte làm cờ và các byte độn (Flag byte with byte stuffing)

Phương pháp này sử dụng một byte có giá trị đặc biệt để làm cờ hiệu (flag byte) đánh dấu điểm bắt đầu và kết thúc của khung.

Một vấn đề phát sinh trong phương pháp này là, trong dữ liệu có thể chứa byte có giá trị của cờ hiệu. Điều này sẽ làm gẫy khung. Để giải quyết vấn đề này, người ta đưa vào phía trước byte dữ liệu có giá trị của cờ hiệu một byte đặc biệc gọi là byte ESC. Bên nhận khi nhận được byte ESC theo sau là giá trị của cờ hiệu thì sẽ bỏ đi ký tự ESC đồng thời biết đây chưa phải là điểm kết thúc của khung. Tương tự, nếu trong dữ liệu có chứa ký tự ESC thì ta cũng đưa thêm vào phía trước nó một ký tự ESC. Kỹ thuật này được gọi là ký tự độn (character stuffing).



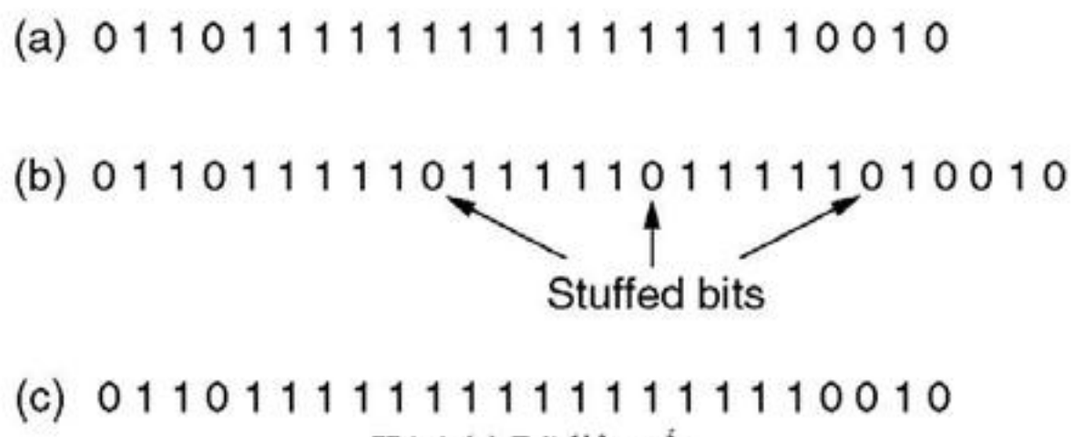
H4.3 (a)Khung được đánh dấu bởi cờ hiệu,
(b) Dữ liệu có chứa cờ hiệu và byte ESC.

Yếu điểm của phương pháp này là nó dựa trên ký tự dạng 8 bits vì thế sẽ không sử dụng được trong các hệ thống sử dụng chuẩn mã 16 bits như Unicode chẳng hạn.

4.1.3.3 Sử dụng cờ bắt đầu và kết thúc khung cùng với các bit độn (Starting and ending flags with bit stuffing).

Phương pháp này sử dụng mẫu bit đặc biệt, 01111110, để làm cờ đánh dấu điểm bắt đầu và kết thúc khung. Khi bên gởi phát hiện có 5 bits 1 liên tiếp trong dữ liệu gởi đi, nó sẽ thêm vào bit 0.

Ngược lại, nếu bên nhận phát hiện 5 bits liên tiếp và theo sau bằng một bit 0, nó sẽ loại bỏ bit 0 ra khỏi dữ liệu. Nhờ thế cờ sẽ không xuất hiện trong dữ liệu gởi.



H4.4 (a) Dữ liệu gốc,
(b)Dữ liệu chuyển lên đường truyền,
(c) Dữ liệu nhận sau khi loại bỏ các bit độn.

4.1.4 Điều khiển lỗi (Error Control)

Một vấn đề khác cần phải xem xét là cách nào để đảm bảo rằng toàn bộ các khung đã được phân phát đến tầng mạng và được phân phát theo đúng trình tự chúng đã được gởi. Điều này không cần quan tâm trong dịch vụ không nối kết không báo nhận. Tuy nhiên nó cần phải được đảm bảo trong dịch vụ nối kết định hướng.

Cách thường được dùng để đảm bảo việc phân phát tin cậy là cung cấp cho người gởi một vài phản hồi từ người nhận về tình trạng nhận khung. Hệ thống sẽ định nghĩa một khung đặc biệt, gọi là khung báo nhận (acknowledgement), để cho người nhận thông báo cho người gởi tình trạng của dữ liệu nhận là tốt hay xấu. Nếu người gởi nhận được một báo hiệu tốt về gói tin, người gởi an tâm rằng gói tin đã được phân phát một cách an toàn. Ngược lại, một khung báo không nhận (unacknowledgement) báo hiệu rằng có một số vấn đề gì đó đối với khung nhận và nó cần phải được truyền lại.

Một khả năng khác có thể xảy ra là khung gởi đi hoàn toàn bị mất không đến được người nhận. Trong trường hợp này sẽ không có một khung báo nhận nào được gởi về cho người gởi, làm cho người gởi rơi vào trạng thái chờ đợi vĩnh viễn.

Để giải quyết vấn đề này, người ta thêm vào tầng liên kết dữ liệu một bộ đếm thời gian (timer). Khi bên gởi truyền một khung đi, nó sẽ thiết lập bộ đếm thời gian. Bộ đếm thời gian sẽ không còn hiệu lực (time-out) sau một khoảng thời đủ lớn để khung được truyền đến người nhận, xử lý ở đó, và khung báo nhận đến được người gởi. Thông thường nếu khung được nhận tốt, khung báo nhận sẽ trở về người gởi trước thời gian qui định. Khi đó bộ đếm thời gian sẽ bị hủy.

Tuy nhiên, nếu khung báo nhận bị mất, bộ đếm thời gian sẽ trôi qua, báo hiệu cho người gởi về vấn đề phát sinh. Giải pháp trong trường hợp này là bên gởi gởi lại khung. Như thế khung được truyền đi nhiều lần có thể làm cho khung được gởi lên tầng mạng nhiều hơn một lần. Để phòng ngừa trường hợp này, người ta gán vào mỗi khung gởi đi một Số thứ tự (sequence number), nhờ đó bên nhận phân biệt được các khung được truyền lại.

4.1.5 Điều khiển luồng (Flow Control)

Một vấn đề thiết kế quan trọng khác cần phải xem xét trong tầng liên kết dữ liệu là sự khác biệt về tốc độ truyền / nhận dữ liệu của bên truyền và bên nhận. Có hai tiếp cận để giải quyết vấn đề này.

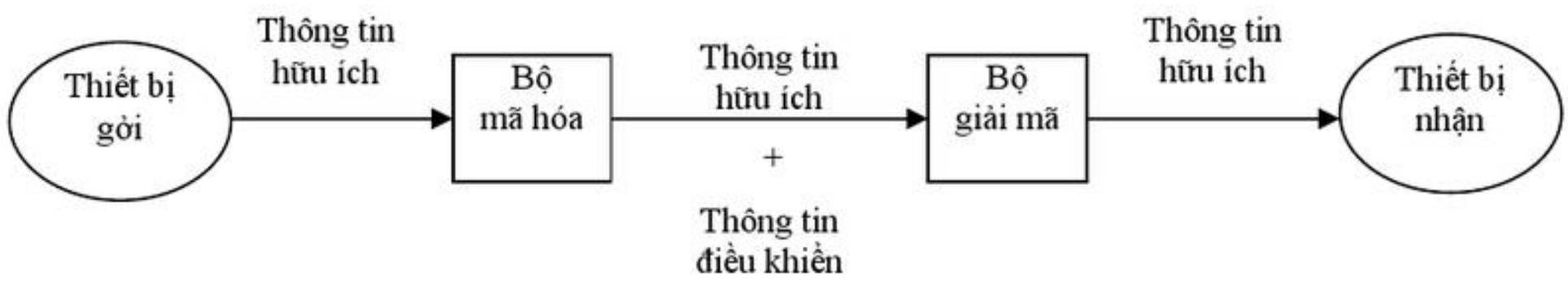
Tiếp cận điều khiển luồng dựa trên phản hồi (feedback based flow control): Người nhận gởi thông tin về cho người gởi cho phép người gởi gởi thêm dữ liệu, cũng như báo với người gởi những gì mà người nhận đang làm.

Tiếp cận điều khiển luồng dựa trên tần số (rate based flow control): Trong giao thức truyền tin cài sẵn cơ chế giới hạn tần suất mà người gởi có thể truyền tin.

4.2 Vấn đề xử lý lỗi

4.2.1 Bộ mã phát hiện lỗi

Khi truyền tải một chuỗi các bit, các lỗi có thể phát sinh ra, bit 1 có thể biến thành bit 0 hay ngược lại.
Ta định nghĩa tỷ lệ lỗi bởi tỷ số sau:
τ = Số bít bị lỗi / Tổng số bít được truyền
Tỷ lệ lỗi này có giá trị từ 10^{-5} đến 10^{-8}. Tùy thuộc vào từng loại ứng dụng, một lỗi có mức độ nghiêm trọng khác nhau, chính vì thế cần có các cơ chế cho phép phát hiện lỗi cũng như sửa lỗi. Các thống kê cho thấy rằng 88% các lỗi xẩy ra do sai lệch một bit và 10% các lỗi xảy ra do sự sai lệch 2 bit kề nhau. Chính vì thế ta ưu tiên cho vấn đề phát hiện các lỗi trên một bit và sửa đổi chúng một cách tự động.
Với ý tưởng như thế, ta sử dụng các mã phát hiện lỗi: bên cạnh các thông tin hữu ích cần truyền đi, ta thêm vào các thông tin điều khiển. Bên nhận thực hiện việc giải mã các thông tin điều khiển này để phân tích xem thông tin nhận được là chính xác hay có lỗi.



H4.5 Mô hình xử lý lỗi trong truyền dữ liệu

Thông tin điều khiển được đưa vào có thể theo 2 chiến lược. Chiến lược thứ nhất gọi là bộ mã sửa lỗi (Error-correcting codes) và chiến lược thứ hai gọi là bộ mã phát hiện lỗi (Error-detecting codes). Bộ mã sửa lỗi cho phép bên nhận có thể tính toán và suy ra được các thông tin bị lỗi (sửa dữ liệu bị lỗi). Trong khi bộ mã phát hiện lỗi chỉ cho phép bên nhận phát hiện ra dữ liệu có lỗi hay không. Nếu có lỗi bên nhận sẽ yêu cầu bên gởi gởi lại thông tin. Với tốc độ của đường truyền ngày càng cao, người ta thấy rằng việc gởi lại một khung thông tin bị lỗi sẽ ít tốn kém hơn so với việc tính toán để suy ra giá trị ban đầu của các dữ liệu bị lỗi. Chính vì thế đa số các hệ thống mạng ngày nay đều chọn bộ mã phát hiện lỗi.

4.2.2 Những bộ mã phát hiện lỗi (Error-Detecting Codes)

Có nhiều sơ đồ phát hiện lỗi, trong đó có các sơ đồ thông dụng là:

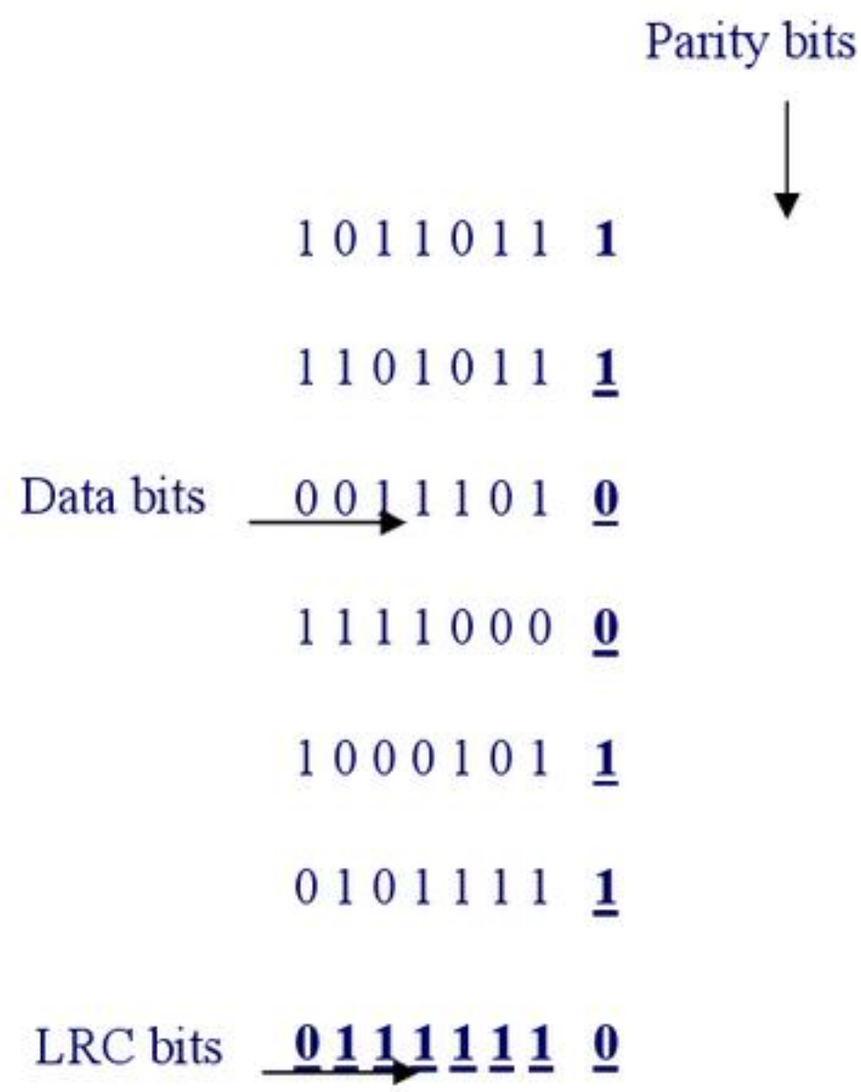
- Kiểm tra chẵn lẻ (Parity checks)
- Kiểm tra thêm theo chiều dọc (Longitudinal reduncy check)
- Kiểm tra phần dư tuần hoàn (Cyclic redundancy check)

4.2.2.1 Kiểm tra chẵn lẻ (Parity Check)

Sơ đồ phát hiện bit lỗi đơn giản nhất là nối một bit chẵn-lẻ vào cuối của mỗi từ trong khung. Một ví dụ tiêu biểu là việc truyền các ký tự ASCII, mà trong đó một bit chẵn-lẻ được nối vào mỗi ký tự ASCII 7 bit. Giá trị của bit này được lựa chọn sao cho có một số chẵn của bit 1, với kiểm tra chẵn (even parity) hoặc một số lẻ của bit 1, với kiểm tra lẻ (odd parity).
Ví dụ, nếu bên gởi đang truyền một ký tự ASCII G (mã ASCII là1110001) và đang dùng phương pháp kiểm tra lẽ, nó sẽ nối một bit 1 và truyền đi 11100011. Bên nhận sẽ kiểm tra ký tự nhận được và nếu tổng của các bit 1 là lẻ, nó xem như không có lỗi. Nếu một bit hoặc một số lẻ bất kỳ các bit bị lỗi đảo ngược thì rõ ràng bên nhận sẽ phát hiện được lỗi. Tuy nhiên, nếu hai hay một số chẵn bất kỳ các bit bị lỗi đảo ngược thì nó sẽ không phát hiện được lỗi. Trên thực tế những xung nhiễu lại thường đủ dài để có thể phá hủy hơn một bit, đặc biệt là với tốc độ dữ liệu cao. Do đó, cần phải có một phương pháp cải thiện trường hợp này.

4.2.2.2 Kiểm tra thêm theo chiều dọc (Longitudinal Redundancy Check or Checksum)

Có thể cải thiện sơ đồ trên bằng cách dùng phương pháp LRC. Trong phương pháp này, khung được xem như một khối nhiều ký tự được sắp xếp theo dạng hai chiều, và việc kiểm tra được thực hiện cả chiều ngang lẫn chiều dọc.
Theo chiều ngang, mỗi ký tự được thêm vào một bit kiểm tra chẵn lẽ như trường hợp trên, và được gọi là bit Kiểm tra chiều ngang VRC (Vertical Redundancy Check).
Theo chiều dọc, cung cấp thêm một ký tự kiểm tra, được gọi là ký tự Kiểm tra chiều dọc LRC (Longitudinal Redundancy Check) hay Checksum. Trong đó, bít thứ i của ký tự này chính là bit kiểm tra cho tất cả các bit thứ i của tất cả các ký tự trong khối.



H4.6 Kiểm tra chiều dọc tuần hoàn

Các phép đo chỉ ra rằng việc dùng cả hai VRC và LRC giảm đi tỷ lệ lỗi không phát hiện được hai đến bốn bậc so với dùng chỉ VRC. Hãy xem trường hợp bit 1 và 3 trong ký tự 1 đang bị lỗi. Khi bên nhận tính toán được bit VRC cho ký tự 1, nó sẽ kiểm tra với bit VRC đã nhận, và sẽ không phát hiện được lỗi. Tuy nhiên, khi nó tính toán được ký tự LRC, bit 1 và 3 của ký tự này sẽ khác với những bit đó trong ký tự LRC nhận được, và sẽ phát hiện được lỗi.
Tuy nhiên, ngay sơ đồ này cũng không phải là thật sự tốt. Bây giờ, nếu giả sử bit 1 và 3 của ký tự 5 cũng bị lỗi, phương pháp này sẽ không phát hiện được điểm sai.

4.2.2.3 Kiểm tra phần dư tuần hoàn (Cyclic Redundancy Check)

Để cải tiến hơn nữa các nhà thiết kế đã dùng kỹ thuật mới dễ dàng và hiệu quả được gọi là kiểm tra phần dư tuần hoàn, trong đó có thể sử dụng một số phương pháp cài đặt khác nhau như: modulo 2, đa thức, thanh ghi dịch và các cổng Exclusive-or.
Các thủ tục với modulo 2 diễn ra như sau. Với một thông điệp **M** có **k** bit cần gởi đi, bên gởi sẽ nối vào cuối thông điệp một chuỗi **F** có **r** bit, được gọi là **Chuỗi kiểm tra khung** (FCS: Frame Check Sequence). Chuỗi kiểm tra khung sẽ tính toán sao cho khung kết quả **T** được hình thành từ việc nối **M** với **F** (gồm **k + r** bit) có thể chia hết bởi số **P** nào đó được định trước. Bên gởi sẽ gởi T đi. Khi bên nhận nhận được T, nó sẽ thực hiện phép chia modulo T cho P. Nếu phép chia không hết, tức có số dư, bên nhận xác định rằng khung T đã bị lỗi, ngược lại là không có lỗi. Nếu khung không có lỗi, bên nhận sẽ tách thông điệp M từ T, là k bits trọng số cao của T.
Phương pháp này dùng phép chia modulo 2 trong việc chia T cho P, phép toán modulo 2 dùng một phép cộng nhị phân không nhớ và đó cũng chính là phép toán Exclusive-or. Ví dụ sau mô tả phép toán cộng và nhân modulo 2:

```
    1111          11001
+   1010      x      11
  ------        -------
    0101          11001
                 11001
                 ------
                 101011
```

- Giả sử ta có:
  - M: Thông điệp k bit cần được gởi sang bên nhận.
  - F : Chuỗi kiểm tra khung FCS gồm r bit là thông tin điều khiển được gởi theo M để giúp bên nhận có thể phát hiện được lỗi.
  - T =MF là khung (k + r) bit, được hình thành bằng cách nối M và F lại với nhau. T sẽ được truyền sang bên nhận, với r < k
- Với M (k bit) , P (r+1 bit), F (r bit), T (k+r bit), thủ tục tiến hành để xác định checksum F và tạo khung truyền như sau:
  - Nối r bit 0 vào cuối M, hay thực hiện phép nhân M với 2^r
  - Dùng phép chia modulo 2 chia chuỗi bit $M*2^r$ cho P.
  - Phần dư của phép chia sẽ được cộng với $M*2^r$ tạo thành khung T truyền đi.
  - Trong đó P được chọn dài hơn F một bit, và cả hai bit cao nhất và thấp nhất phải là 1

Ví dụ:

- Giả sử ta có:
  - M = 1010001101 (10 bit)
  - P = 110101 (6 bit)
  - FCS cần phải tính toán (5 bit)
- Ta lần lượt thực hiện các bước sau:
  - Tính $M*2^5$ = 101000110100000.
  - Thực hiện phép chia modulo $M*2^5$ cho P như hình dưới, ta được phần dư F = **01110**
  - Tạo khung gởi đi là T = $M*2^r$ + F = 1010001101**01110**

```
                    1 1 0 1 0 1 0 1 1 0     ( Q : Kết quả phép chia)
(P) 1 1 0 1 0 1 | 1 0 1 0 0 0 1 1 0 1 0 0 0 0 0 ( M*2^r)
                  1 1 0 1 0 1
                  -----------
                    1 1 1 0 1 1
                    1 1 0 1 0 1
                  -----------
                      0 1 1 1 0 1
                      0 0 0 0 0 0
                      -----------
                        1 1 1 0 1 0
                        1 1 0 1 0 1
                        -----------
                          0 1 1 1 1 1
                          0 0 0 0 0 0
                          -----------
                            1 1 1 1 1 0
                            1 1 0 1 0 1
                            -----------
                              0 1 0 1 1 0
                              0 0 0 0 0 0
                              -----------
                                1 0 1 1 0 0
                                1 1 0 1 0 1
                                -----------
                                  1 1 0 0 1 0
                                  1 1 0 1 0 1
                                  -----------
                                    0 0 1 1 1 0
                                    0 0 0 0 0 0
                                    -----------
                                    0 1 1 1 0 = F
```

Ngoài ra người ta còn có thể sử dụng phương pháp đa thức để biểu diễn phương pháp kiểm tra phần dư tuần hòan. Trong phương pháp này người ta biểu diễn các chuỗi nhị phân dưới dạng những đa thức của biến x với các hệ số nhị phân. Các hệ số tương ứng với các bit trong chuỗi nhị phân cần biểu diễn.

Giả sử ta có M=110011và P = 11001, khi đó M và P sẽ được biểu diễn lại bằng 2 đa thức sau:

$$M(x) = x^5 + x^4 + x + 1$$

$$P(x) = x^4 + x^3 + 1$$

Những phép toán trên đa thức vẫn là modulo 2. Quá trình tính CRC được mô tả dưới dạng các biểu thức sau:

1. $\dfrac{X^n M(X)}{P(X)} = Q(X) + \dfrac{R(X)}{P(X)}$
2. $T(X) = X^n M(X) + R(X)$

Các version thường được sử dụng của P là :

CRC-12 $= X^{12} + X^{11} + X^3 + X^2 + X + 1$

CRC-16 $= X^{16} + X^{15} + X^2 + 1$

CRC-CCITT $= X^{16} + X^{12} + X^5 + 1$

CRC-32 $= X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

Ví dụ:

- Cho: M=1010001101, P=110101
- Ta có: r=5

  $M(x)= x^9+x^7+x^3+x^2+1$

  $x^5M(x)= x^{14}+x^{12}+x^8+x^7+x^5$

  $P(x) = x^5+x^4+x^2+1$
- Thực hiện phép toán:

  $\dfrac{x^rM(x)}{P(x)} = Q(x) + \dfrac{F(x)}{P(x)}$

  => $Q(x)= x^9+ x^8+ x^6+x^4+x^2 +x^1$

  $F(x)= x^3+ x^2+ x^1$ <> **01110**

  ⇨ Khung cần truyền đi là T= 1010001101**01110**

4.3 Một số giao thức điều khiển lỗi (Error Control)

Phần kế tiếp chúng ta xem xét một số giao thức cơ bản được sử dụng nhiều trong việc điều khiển lỗi. Các giao thức này được xây dựng dựa trên các giả định sau:

- Chúng ta có máy tính A muốn gởi dữ liệu cho máy tính B.
- Luôn luôn có đủ dữ liệu cho máy A gởi đi .
- Các giao diện giao tiếp với tầng mạng và tầng vật lý đã được định nghĩa chuẩn.
- Bên nhận thông thường thực hiện việc chờ đợi một sự kiện nào đó phát sinh bằng cách gọi hàm wait_for_event().

Các giao thức được trình bày dưới dạng các chương trình viết bằng ngôn ngữ c. Chúng sử dụng các định nghĩa trong tập tin **protocol.h** có nội dung như sau:

```
#define MAX_PKT 1024                                  /* Kích thước tối đa của một gói tin */

typedef enum {false, true} boolean;                   /* Kiểu luận lý */
typedef unsigned int seq_nr;                          /* Số thứ tự của khung gởi hoặc khung báo nhận*/
typedef struct {unsigned char data[MAX_PKT];} packet; /* Định nghĩa kiểu của gói tin */
typedef enum {data, ack, nak} frame_kind;             /* Các loại khung */
typedef struct {                                      /* Kiểu dữ liệu của khung: */
  frame_kind kind;                                    //Loại khung
  seq_nr seq;                                         //Số thứ tự của khung gởi đi
  seq_nr ack;                                         //Số thức tự của khung muốn báo nhận
  packet info;                                        //Thông tin gởi nhận,
} frame;                                              // là gói tin nhận của tầng mạng

/* Chờ một sự kiện xuất hiện; trả về kiểu của sự kiện */
void wait_for_event(event_type *event);
/* Nạp gói tin nhận được từ tầng mạng vào khung để gởi đi */
void from_network_layer(packet *p);
/* Chuyển dữ liệu từ khung nhận được cho tầng mạng */
void to_network_layer(packet *p);
/* Nhận khung đến từ tầng vật lý và lưu nó vào khung r */
void from_physical_layer(frame *r);
/* Chuyển một khung xuống tầng vật lý để truyền đi */
void to_physical_layer(frame *s);
/* Khởi động đồng hồ và bật sự kiện quá thời hạn cho khung thứ k đang gởi đi */
void start_timer(seq_nr k);
/* Dừng đồng hồ và tắt sự kiện quá thời hạn cho khung thứ k đang gởi đi */
void stop_timer(seq_nr k);
/* Khởi động đồng hồ phụ và bật sự kiện quá thời hạn cho khung phản hồi*/
void start_ack_timer(void);
/* Dừng đồng hồ phụ và tắt sự kiện quá thời hạn cho khung phản hồi*/
void stop_ack_timer(void);
/* Cho phép tầng mạng tạo sự kiện tầng mạng đã sẵn sàng */
void enable_network_layer(void);
/* Cấm tầng mạng tạo sự kiện tầng mạng đã sẵn sàng */
void disable_network_layer(void);
/* Macro để tăng giá trị K theo kiểu quay vòng */
#define inc(k) if (k < MAX_SEQ) k = k + 1; else k = 0
```

4.3.1 Giao thức truyền đơn công không ràng buộc (Unrestricted Simplex Protocol)

Protocol 1 (utopia) được dùng cho việc truyền tải thông tin theo một chiều từ người gởi sang người nhận. Kênh truyền được giả định là không có lỗi và bên nhận được giả định rằng có thể xử lý được hết tất cả các thông tin gởi đến một cách nhanh chóng. Chính vì thế mà bên gởi chỉ đơn thuần thực hiện một vòng lặp đưa dữ liệu lên đường truyền với tốc độ nhanh nhất có thể.

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender1(void)
{
  frame s;                          /* Vùng đệm để chứa khung gởi đi */
  packet buffer;                    /*  Vùng đệm để chứa gói tin gởi đi */

  while (true) {
      from_network_layer(&buffer); /* Nhận gói tin từ tầng mạng để gởi đi */
      s.info = buffer;              /* Đưa gói tin vào khung để gởi đi */
      to_physical_layer(&s);        /* Gởi khung xuống tầng vật lý để gởi lên đường truyền */
  }                                 /



}

void receiver1(void)
{
  frame r;
  event_type event;

  while (true) {
      wait_for_event(&event);       /* Chờ sự kiện, chỉ xuất hiện khi khung đến */
      from_physical_layer(&r);      /* Nhận khung từ tầng vật lý */
      to_network_layer(&r.info);    /* Lấy thông tin ra khỏi khung và gởi lên tầng mạng */
  }
}
```

P4.1 Giao thức truyền đơn công không ràng buộc

4.3.2 Giao thức truyền đơn công dừng và chờ (Simplex Stop-and-wait Protocol)

Giao thức Stop-and-wait cũng được thiết kế cho các cuộc truyền tải thông tin một chiều từ người gởi sang người nhận. Kênh truyền tải thông tin một lần nữa cũng được giả định rằng không có lỗi như giao thức Unrestricted Simplex Protocol. Tuy nhiên, trong trường hợp này, bên nhận chỉ có một vùng lưu trữ có khả năng hạn chế và một tốc độ xử lý giới hạn, vì thế giao thức phải được thiết kế dự phòng cho trường hợp dữ liệu máy gởi đến nhanh làm tràn vùng lưu trữ thông tin của bên nhận.

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender2(void)
{
  frame s;                          /* Vùng đệm để chứa khung gởi đi */
  packet buffer;                    /* Vùng đệm để chứa gói tin gởi đi */
  event_type event;                 /* Sự kiện báo hiệu khung đến */

  while (true) {
      from_network_layer(&buffer);  /* Nhận gói tin từ tầng mạng để gởi đi */
      s.info = buffer;              /* Đưa gói tin vào khung để gởi đi */
      to_physical_layer(&s);        /* Gởi khung xuống tầng vật lý để gởi lên đường truyền */
      wait_for_event(&event);       /* Chờ sự kiện đến của khung báo nhận gởi về từ bên gởi*/
  }
}

void receiver2(void)
{
  frame r, s;
  event_type event;
  while (true) {
      wait_for_event(&event);       /* Chờ sự kiện, chỉ xuất hiện khi khung đến */
      from_physical_layer(&r);      /* Nhận khung từ tầng vật lý */
      to_network_layer(&r.info);    /* Lấy thông tin ra khỏi khung và gởi lên tầng mạng */
      to_physical_layer(&s);        /* Gởi khung báo nhận sang bên gởi */
  }
}
```

P4.2 Giao thức truyền đơn công truyền và chờ

4.3.3 Giao thức truyền đơn công cho kênh truyền có nhiễu (Simplex Protocol for Noisy Channel)

Giả sử ta bỏ đi giả thuyết kênh truyền không có lỗi. Trong trường hợp này, với các kỹ thuật xử lý lỗi (Parity check, CRC), bên nhận có thể phát hiện ra được các khung bị lỗi. Tuy nhiên, điều gì sẽ xảy ra nếu khung gởi đi bị mất, không đến được nơi nhận. Khi đó sẽ dẫn đến tình trạng như sau:

- Người gởi không biết được khung có đến nơi nhận tốt hay không.
  - Giải pháp là yêu cầu người nhận gởi các khung báo nhận thông báo về tình hình các khung bị lỗi.
- Các khung báo nhận có thể bị mất.
  - Giải pháp: Mỗi khi gởi một khung đi, Bên gởi sẽ thiết lập một bộ đếm thời gian. Nếu sau một khoảng thời gian qui định mà không nhận được khung báo nhận, bên gởi sẽ gởi lại các khung không được báo nhận
- Bên nhận không phân biệt được các khung trùng lắp do bên gởi gởi lại.
  - Giải pháp: Mỗi khung sẽ có một số thứ tự để phân biệt lẫn nhau. Số thứ tự này sẽ được tăng dần cho đến một giá trị cực đại sau đó lại quay về giá trị 0. Trong ví dụ sau, số thứ tự có giá trị cực đại là 1. Như vậy ta chỉ sử dụng 2 giá trị là 0 và 1 để đánh số thứ tự cho khung.

```
/* Protocol 3 (par) allows unidirectional data flow over an unreliable channel. */
#define MAX_SEQ 1                          /* Giá trị tối đa của số thứ tự khung là 1 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"

void sender3(void)
{
  seq_nr next_frame_to_send;               /* Số thứ tự của gói tin của lần gởi kế tiếp */
  frame s;                                 /* Khung để gởi dữ liệu đi */
  packet buffer;                           /* Vùng lưu trữ cho gói tin gởi */
  event_type event;

  next_frame_to_send = 0;
  from_network_layer(&buffer);
  while (true) {
      s.info = buffer;                     /* Khởi động số thứ tự cho khung gởi */
      s.seq = next_frame_to_send;          /* Nhận gói tin đầu tiên từ tầng mạng để gởi đi */
      to_physical_layer(&s);
      start_timer(s.seq);
      wait_for_event(&event);              /* Xây dựng khung để gởi đi */
      if (event == frame_arrival) {        /* Đánh số thứ tự cho khung */
            from_physical_layer(&s);       /* Gởi khung xuống tầng vật lý để truyền đi */
            if (s.ack == next_frame_to_send) {
                  stop_timer(s.ack);       /* Nếu khung báo nhận đến chậm, tạo sự kiện
                  from_network_layer(&buffer);
                  inc(next_frame_to_send);
            }                              time-out */
      }
  }
}

void receiver3(void)
{
  seq_nr frame_expected;                   /* Số thứ tự của gói tin chờ nhận kế tiếp */
  frame r, s;                              /* Khung nhận và khung báo nhận */
  event_type event;

  frame_expected = 0;
  while (true) {
      wait_for_event(&event);              /* Khởi động số thứ tự cho khung nhận */
      if (event == frame_arrival) {
            from_physical_layer(&r);
            if (r.seq == frame_expected) {
                  to_network_layer(&r.info);
                  inc(frame_expected);     /* Chờ một sự kiện xảy ra*/
            }                              /* Nếu là sự kiện khung đến */
            s.ack = 1 – frame_expected;    /* - Nhận khung dữ liệu từ tầng vật lý */
            to_physical_layer(&s);         /* - Nếu đúng là khung đang chờ */
      }                                    /* - Gởi dữ liệu nhận được lên tầng
  }
}                                          mạng */
```

P4.3 Giao thức truyền đơn công cho kênh truyền có nhiễu

4.4 Giao thức của sổ trượt (Sliding windows)

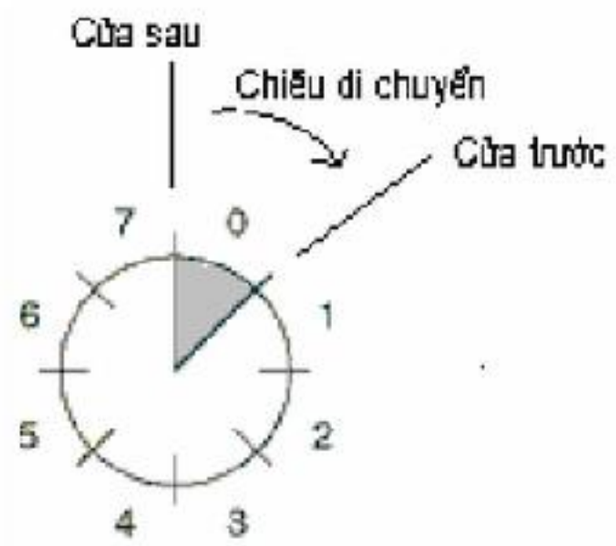
4.4.2 Vấn đề truyền tải thông tin theo hai chiều (Duplex)

Chúng ta muốn việc truyền tải thông tin giữa hai bên giao tiếp diễn ra một cách đồng thời theo hai chiều hơn là chỉ một chiều để khai thác tối đa khả năng của kênh truyền.
Để thực hiện được điều này, chúng ta thực sử dụng chế độ truyền tải hai chiều, gọi là song công (Duplex). Nguyên tắc thực hiện như sau:
Vẫn thực hiện việc truyền tải khung, tuy nhiên ta có phân biệt thành các loại khung: dữ liệu (data), báo nhận ACK (acknowledgement), và báo không nhận NACK(Not Acknowledgement) trong trường xác định loại (Type) của khung.
Khi một bên nào đó truyền tin, nó có thể kết hợp đưa thông tin báo cho bên kia biết tình trạng của gói tin mà nó đã nhận trước đó. Ta gọi là kỹ thuật **piggyback**.

4.4.3 Giới thiệu về giao thức cửa sổ trượt

Thay vì chỉ truyền đi một khung tại một thời điểm (simplex), giao thức cửa sổ trượt cho phép bên gởi có thể gởi đi nhiều khung.
Giao thức này sử dụng một cửa sổ để cho phép bên gởi theo dõi các khung mà nó được phép gởi đi và các khung mà nó đang chờ báo nhận, gọi là cửa sổ gởi (Sending Windows). Một cửa sổ khác để bên nhận theo dõi các khung mà nó được phép nhận, gọi là cửa sổ nhận (Receiving Windows).

Cấu trúc của cửa sổ được mô tả như sau:

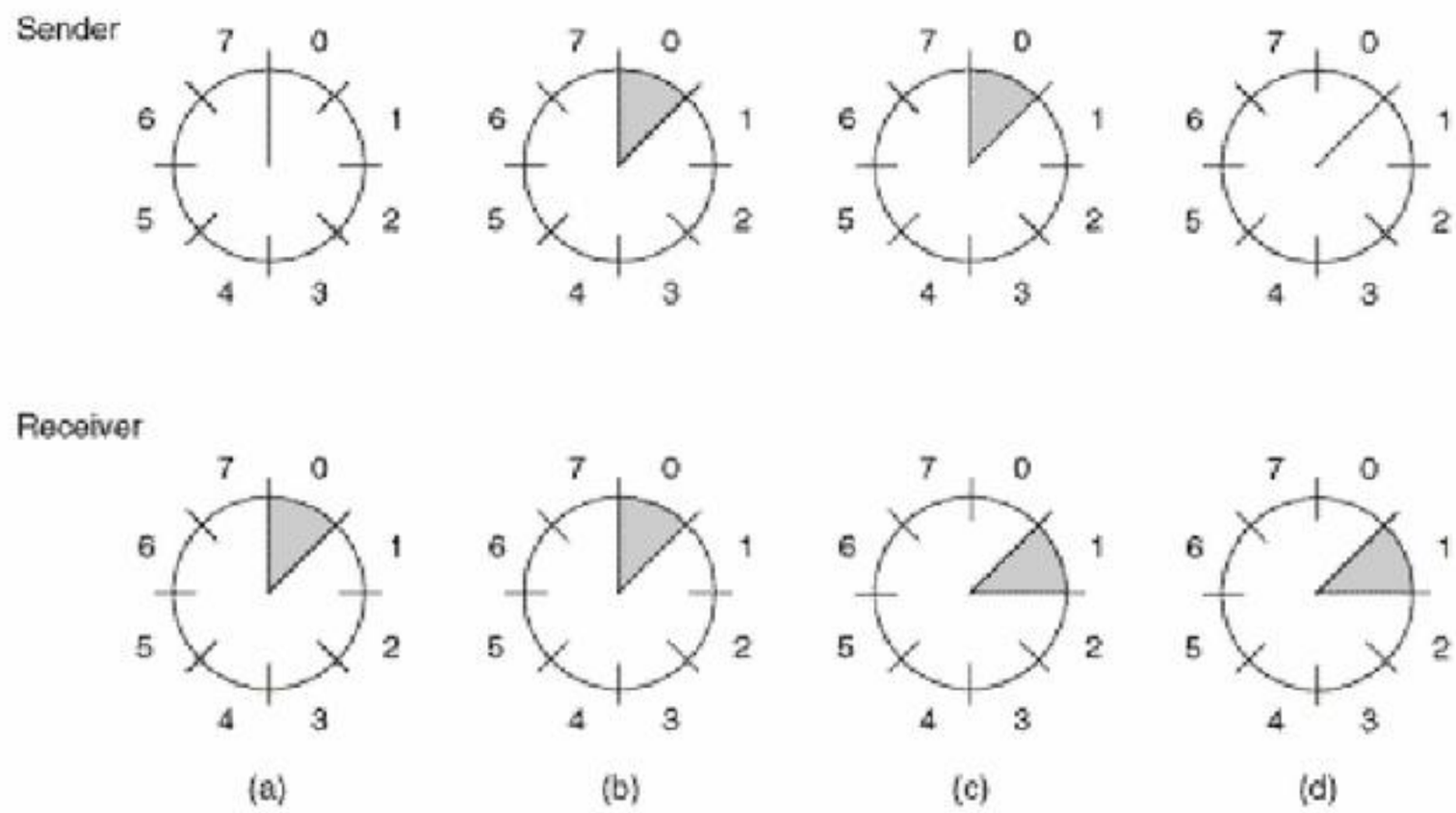


H4.7 Cấu trúc cửa sổ trượt

- Phần tô đen là phạm vi của cửa sổ gồm có **cửa trước** và **cửa sau** cùng di chuyển theo một chiều.
- Kích thước của cửa sổ là chiều của cung giới hạn từ cửa sau đến cửa trước.
- Kích thước của cửa sổ có thể thay đổi. Khi cửa trước di chuyển, cửa sổ được mở rộng ra. Ngược lại khi cửa sau di chuyển, kích thước của cửa sổ bị thu hẹp lại và nó làm cho cửa sổ thay đổi vị trí, trượt / quay quanh một tâm của vòng tròn.
- Kích thước nhỏ nhất của cửa sổ là 0, khi đó cửa trước và cửa sau nằm cùng một vị trí. Giả sử, có $n=2^k$ vị trí cho các cửa, khi đó kích thước tối đa của cửa sổ là n-1 (không là n để phân biệt với kích thước là 0).
- Giả sử ta dùng k bit để đánh số thứ tự cho các khung. Ta sẽ có 2^k khung, đánh số từ 0 đến 2^k-1. Khi đó cửa sổ trượt sẽ được chia thành 2^k vị trí tương ứng với 2^k khung.
- Đối với cửa sổ gởi, các vị trí nằm trong cửa sổ trượt biểu hiện số thứ tự của các khung mà bên gởi đang chờ bên nhận báo nhận. Phần bên ngoài cửa sổ là các khung có thể gởi tiếp. Tuy nhiên phải đảm bảo rằng, cửa sổ gởi không được vượt quá kích thước tối đa của cửa sổ.
- Đối với bên nhận, các vị trí nằm trong cửa sổ biểu hiện số thứ tự các khung mà nó đang sẵn sàng chờ nhận.
- Kích thước tối đa của cửa sổ biểu thị dung lượng bộ nhớ đệm của bên nhận có thể lưu tạm thời các gói tin nhận được trước khi xử lý chúng. Giả sử bên nhận có một vùng bộ nhớ đệm có khả năng lưu trữ 4 khung nhận được. Khi đó, kích thước tối đa của cửa sổ sẽ là 4.

4.4.4 Hoạt động của cửa sổ trượt

Ví dụ sau mô tả hoạt động của cửa sổ trượt với kích thước cửa sổ là 1, sử dụng 3 bits để đánh số thứ tự khung (từ 0 đến 7).



H4.8 Hoạt động của cửa sổ trượt

Khởi đầu, Hình (a):
Bên gởi: chưa gởi khung nào nên kích thước của cửa sổ là 0.
Bên nhận đang chờ nhận khung 0, kích thước cửa sổ là 1
Bên gởi gởi khung số 0: Nó kiểm tra kích thước của cửa sổ trượt là 0, nhỏ hơn kích thước tối đa nên nó được phép gởi. Cửa trước của cửa sổ gởi di chuyển lên một bước chứa giá trị 0 là số thứ tự của khung báo nhận bên gởi đang chờ. Kích thước cửa sổ trượt lúc này là 1, đạt đến kích thước tối đa nên nó không được phép gởi thêm khung nữa (Hình b).
Bên nhận nhận được khung 0: nó kiểm tra và nhận thấy khung không có lỗi. Nó gởi khung báo nhận số 0 về cho bên nhận. Đồng thời cửa sau của nó di chuyển để loại khung số 0 ra khỏi cửa sổ trượt. Cửa trước cũng di chuyển để mở rộng kích thước cửa sổ đến giá trị tối đa. Lúc này cửa sổ nhận chứa khung số 1 là khung mà nó đang chờ nhận tiếp (Hình c).
Bên gởi nhận được khung báo nhận số 0: Vì đây là khung báo hiệu bên nhận đã nhận tốt nên cửa sau của cửa sổ gởi di chuyển để loại khung số 0 ra khỏi cửa sổ gởi. Lúc này cửa sổ gởi có kích thước là 0, bên gởi có quyền gởi tiếp khung (Hình d)

Như vậy khi kích thước của cửa sổ trượt là 1, ta có giao thức stop-and-wait.

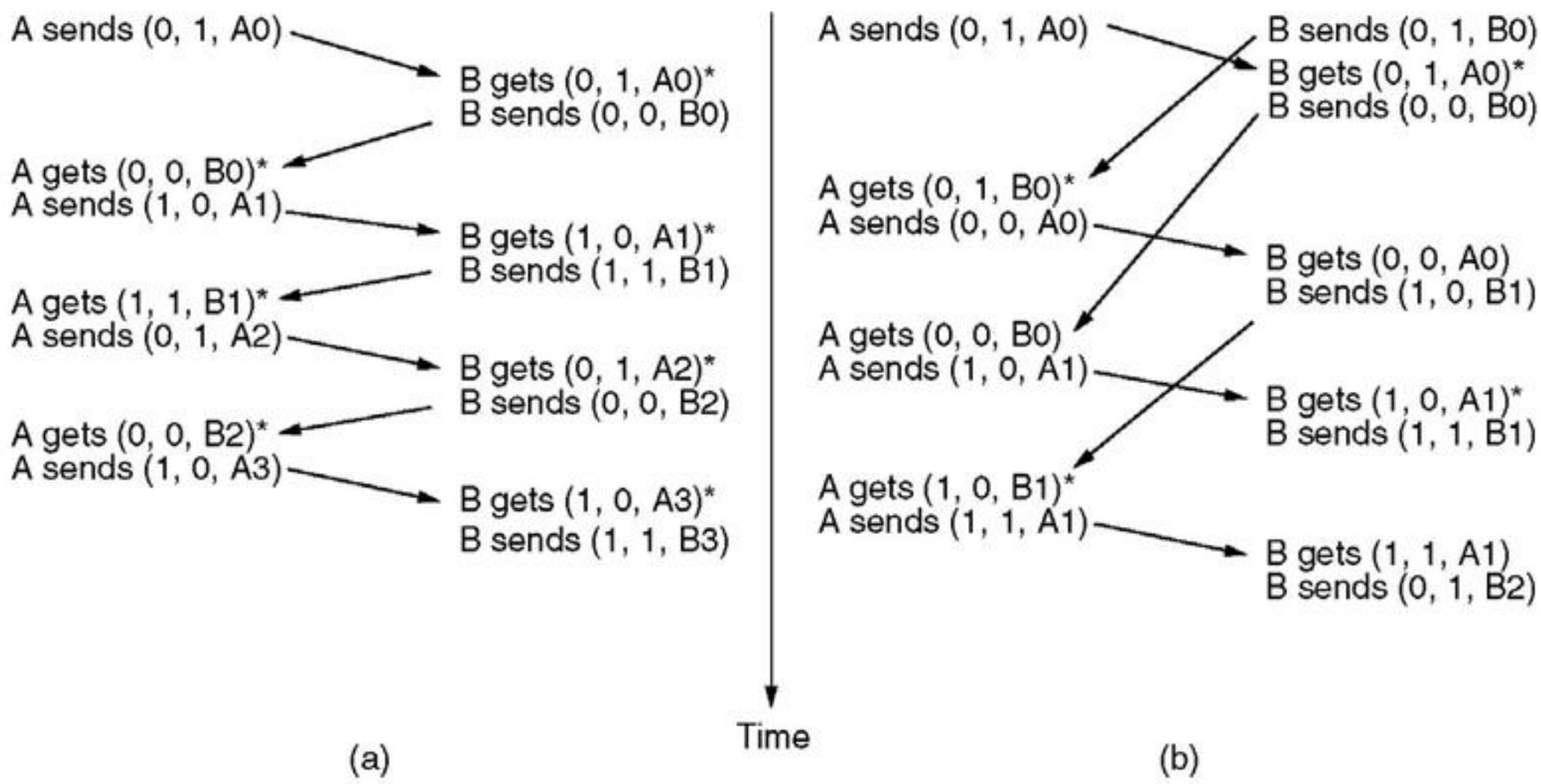
4.4.5 Cài đặt giao thức cửa sổ trượt kích thước 1 bit (A One-Bit Sliding Window Protocol)

```
/* Protocol 4 (sliding window) is bidirectional */
#define MAX_SEQ 1                                   /* Kích thước cửa sổ là 1 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"
void protocol4 (void)
{
    seq_nr next_frame_to_send;                      /* Số thứ tự của khung gởi đi kế tiếp */
    seq_nr frame_expected;                          /* Số thứ tự của khung báo nhận đang chờ nhận */
    frame r, s;                                     /* Khung nhận và khung gởi */
    packet buffer;                                  /* Gói tin chờ gởi */
    event_type event;

    next_frame_to_send = 0;                         /* Khởi động số thứ tự khung gởi */
    frame_expected = 0;                             /* Khởi động số thứ tự khung báo nhận chờ nhận */
    from_network_layer(&buffer);                    /* Nhận gói tin từ tầng mạng để gởi đi */
    s.info = buffer;                                /* Đưa gói tin dữ liệu vào khung để gởi */
    s.seq = next_frame_to_send;                     /* Đặt số thứ tự cho khung */
    s.ack = 1 – frame_expected;                     /* Đặt số thứ tự báo nhận vào khung */
    to_physical_layer(&s);                          /* Đưa khung xuống tầng vật lý để gởi */
    start_timer(s.seq);                             /* Khởi động bộ đếm thời gian */

    while (true) {
        wait_for_event(&event);                     /* Chờ sự kiện Khung đến, Khung bị lỗi, quá thời gian */
        if (event == frame_arrival) {               /* Một khung đến không bị lỗi */
            from_physical_layer(&r);                /* Nhận khung từ tầng vật lý */

            if (r.seq == frame_expected) {          /* Kiểm tra có phải là khung đang chờ nhận không */
                to_network_layer(&r.info);          /* Lấy gói tin ra khỏi khung và chuyển lên tầng mạng */
                inc(frame_expected);                /* Tăng số thứ tự của khung chờ nhận kế tiếp */
            }

            if (r.ack == next_frame_to_send) {      /* Nếu bên kia đã báo nhận khung vừa gởi */
                stop_timer(r.ack);                  /* Xóa bộ đếm thời gian */
                from_network_layer(&buffer);        /* Nhận gói tin kế tiếp từ tầng mạng để gởi đi */
                inc(next_frame_to_send);            /* Tăng số thứ tự của khung kế tiếp */
            }
        }
        s.info = buffer;                            /* Đưa gói tin vào khung để gởi */
        s.seq = next_frame_to_send;                 /* Đặt số thứ tự cho khung gởi */
        s.ack = 1 – frame_expected;                 /* Đặt số thứ tự khung báo nhận */
        to_physical_layer(&s);                      /* Đưa khung xuống tầng vật lý để gởi */
        start_timer(s.seq);                         /* Khởi động bộ đếm thời gian */
    }
}
```

P4.4 Cài đặt cửa sổ trượt với kích thước là 1

4.4.6 Ví dụ về 2 kịch bản của giao thức trên

(a): Việc gởi nhận diễn ra bình thường theo đúng tuần tự
(b): Việc gởi nhận diễn ra theo một trình tự bất kỳ
Ký hiệu **A send (seq, ack, packet number)** để chỉ rằng A gởi B một khung có số thứ tự là **seq**, đồng thời báo cho B biết A đã nhận được tốt khung có số thứ tự **ack** của B gởi sang. Khung chứa gói tin thứ **packet number**. Dấu * biểu thị rằng khung tốt, và gói tin được lấy ra khỏi khung để chuyển cho tầng mạng.



H4.9 Kịch bản giao thức cửa sổ trượt với kích thước là 1

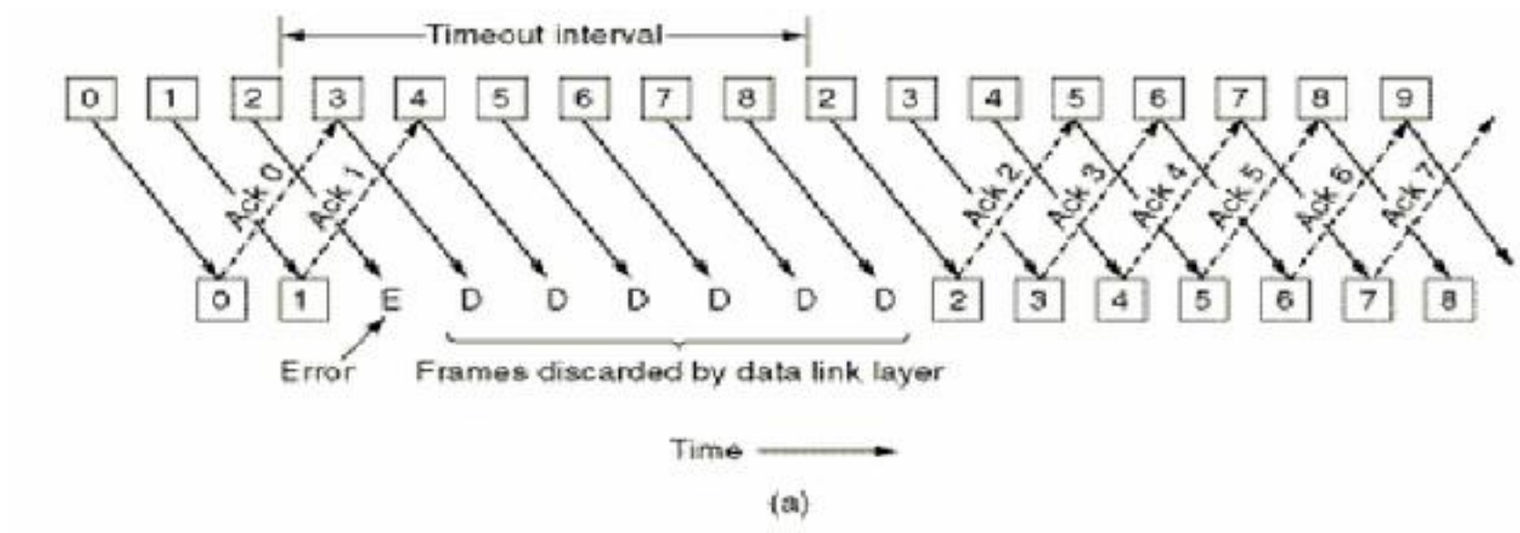
1.1.1 Vấn đề điều khiển lỗi (Error Control)

Vấn đề kế tiếp cần phải quan tâm là bên nhận sẽ làm gì nếu khung bị lỗi.
Giải pháp đơn giản là truyền lại tất cả các khung bắt đầu từ khung thứ N bị lỗi. Nếu có những khung khác được nhận trong khoảng thời gian này thì chúng đều bị bỏ qua. Đây gọi là giao thức **Go-Back-N.**
Giải pháp thứ hai là chỉ truyền lại những khung bị lỗi, và chờ đến khi nó được gởi lại trước khi tiếp tục việc gởi tin, gọi là giao thức **Selective Repeat.**

4.4.6.1 Giao thức Go-Back-N

Giao thức Go-Back-N thì rất đơn giản. Khi một khung bị lỗi. Bên nhận bỏ qua khung. Vì không một báo nhận nào gởi về cho bên nhận nên sự kiện quá thời gian xảy ra, bên gởi phải gởi lại ung bị lỗi và toàn bộ các khung phía sau nó.

Ví dụ:



H4.10 Giao thức Go-Back-N

Trong ví dụ trên, bên nhận phát hiện ra khung số 2 bị lỗi nó bỏ qua các khung sau đó (3,4,5,6,7,8), chỉ chờ nhận lại khung số 2. Phía bên gởi chờ báo nhận từ bên nhận cho đến khi quá thời gian, nó sẽ thực hiện gởi lại các khung 2, 3, 4, 5, 6, ...
Đoạn chương trình sau cài đặt giao thức Go-Back-N

/* Giao thức này cho phép nhiều khung được gởi đi. Bên gởi có thể gởi trước đến MAX_SEQ khung mà không cần chờ một báo nhận. Điểm lưu ý khác là tầng mạng không phải luôn luôn có dữ liệu sẵn sàng để gởi. Khi nào có dữ liệu để gởi, tầng mạng sẽ sinh ra một sự kiện network-layer- ready.*/

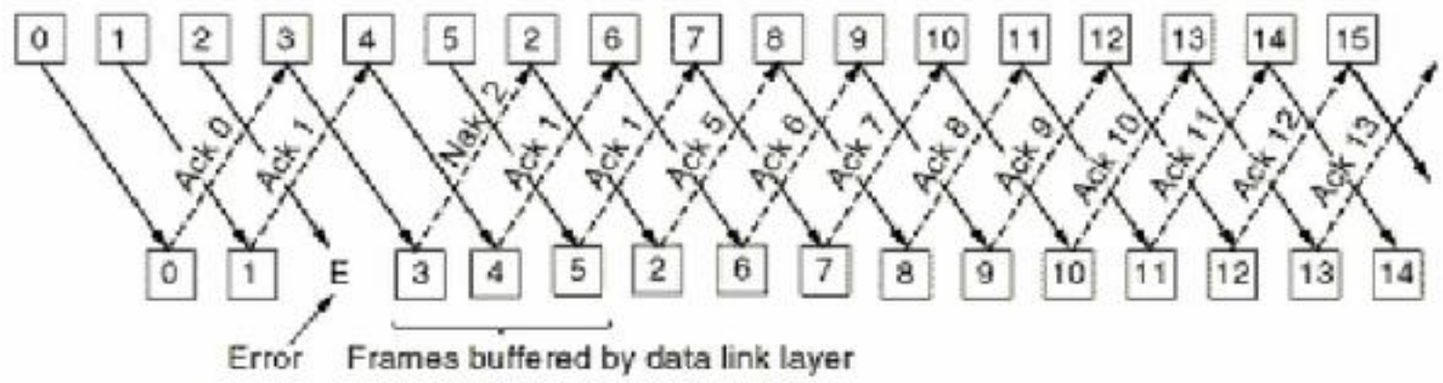
```
#define MAX_SEQ 7                    /* Kích thước lớn nhất của cửa sổ trượt, phải là 2^k-1*/
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready} event_type;
#include "protocol.h"

static boolean between(seq_nr a, seq_nr b, seq_nr c)
{
   /*          True nếu a<=b<c                          */
  if (((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a)))
      return(true);
   else
      return(false);
}

static void send_data(seq_nr frame_nr, seq_nr frame_expected, packet buffer[ ])
{
/*          Tạo khung gởi gói tin đi                          */
  frame s;                              /* Khung để gởi gói tin đi */

  s.info = buffer[frame_nr];            /* Đưa gói tin vào khung */
  s.seq = frame_nr;                     /* Đặt số thứ tự cho khung gởi*/
  s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1/* Đặt số thứ tự cho khung cần báo nhận
  to_physical_layer(&s);                /* Gởi khung xuống tầng vật lý để truyền đi */
  start_timer(frame_nr);                /* Khởi động bộ đếm thời gian cho khung gởi đi*/
}

void protocol5(void)
{
  seq_nr next_frame_to_send;            /* Số thứ tự cho khung gởi kế tiếp */
  seq_nr ack_expected;                  /* Khung lâu nhất chưa được báo nhận */
  seq_nr frame_expected;                /* Khung chờ nhận kế tiếp /
  frame r;                              /* Khung */
  packet buffer[MAX_SEQ + 1];           /* Vùng bộ nhớ đệm cho các khung gởi đi */
  seq_nr nbuffered;                     /* Số lượng bộ nhớ đệm đang được dùng */
  seq_nr i;                             /* Chỉ số mảng của vùng nhớ đệm */
  event_type event;

  enable_network_layer();
  ack_expected = 0;
  next_frame_to_send = 0;               /* Cho phép tầng mạng tạo sự kiện network_layer_ready */
  frame_expected = 0;                   /* Đặt báo nhận đầu tiên chờ nhận là 0 */
  nbuffered = 0;                        /* Khung đầu tiên gởi đi là 0 */
```

```
while (true) {
  wait_for_event(&event);          /* Chờ 1 trong 4 sự kiện liệt kê ở trên xảy ra */

  switch(event) {
    case network_layer_ready:      /* Tầng mạng có một gói tin cần gởi đi */
        /* Chấp nhận, lưu và truyền đi một khung mới */
        from_network_layer(&buffer[next_frame_to_send]);/* Nhận khung từ tầng vật lý*/
        nbuffered = nbuffered + 1; /* Tăng kích thước cửa sổ gởi */
        send_data(next_frame_to_send, frame_expected, buffer)/* Gởi khung đi */
        inc(next_frame_to_send); /* Di chuyển cửa trước của cửa sổ gởi */
        break;

    case frame_arrival:            /* Một khung dữ liệu hay điều khiển vừa đến */
        from_physical_layer(&r);   /* Nhận khung từ tầng vật lý */

        if (r.seq == frame_expected) {
            /* Là khung đang được chờ đợi */
            to_network_layer(&r.info)/* Chuyển gói tin lên tầng mạng */
            inc(frame_expected);   /* Di chuyển cửa sau của cửa sổ nhận */
        }

    /* Nếu là Ack thứ n, sẽ không quan tâm đến các ACK n-1, n-2, ... */
        while (between(ack_expected, r.ack, next_frame_to_send)) {
    /* Xử lý các báo nhận */
            nbuffered = nbuffered —1;/* Giảm kích thước cửa sổ gởi */
            stop_timer(ack_expected);/* Khung đã đến, xóa bộ đếm thời gian */
            inc(ack_expected);/* Tăng số thứ tự khung chờ nhận kế tiếp */
        }
        break;

    case cksum_err: break;         /* Khung nhận bị lỗi, bỏ qua */

    case timeout:   /* Quá thời gian, truyền lại tất cả các khung đang chờ báo nhận */
        next_frame_to_send = ack_expected;  /* Bắt đầu truyền lại */
        for (i = 1; i <= nbuffered; i++) {
            send_data(next_frame_to_send, frame_expected, buffer);/* Truyền lại */
            inc(next_frame_to_send); /* Chuẩn bị để truyền khung kế tiếp */
        }

  }

  if (nbuffered < MAX_SEQ)         /* Vùng đệm còn khả năng chứa gói tin ? */
        enable_network_layer();
  else
        disable_network_layer();
}
}
```

P4.5 Cài đặt giao thức Go-Back-N

4.4.6.2 Giao thức Selective Repeat

Trong giao thức này, khung bị lỗi bị bỏ đi, nhưng các khung nhận tốt sau đó đều được lưu lại tạm thời trong vùng nhớ đệm. Khi quá thời gian, bên gởi chỉ gởi lại khung cũ nhất chưa được báo nhận. Nếu khung này đến nơi chính xác, bên nhận có thể chuyển lên tầng mạng tất cả các khung đã được lưu vào bộ nhớ đệm theo đúng thứ tự.
Trong giao thức này, bên nhận sử dụng khung **Báo không nhận** NAK (Negative Acknowledge) khi phát hiện ra khung bị lỗi, ví dụ lỗi CRC, sai thứ tự gói tin. NAK sẽ được gởi về bên nhận trước khi sự kiện quá thời gian báo nhận của khung bị lỗi xảy ra. Nhờ đó tăng được hiệu xuất truyền tin.



(b)

H4.11 Giao thức Selective Repeat với cửa sổ trượt lớn hơn 1

Trong ví dụ trên các khung 0, 1 được nhận tốt và đã được báo nhận, còn khung số 2 thì bị lỗi trên đường truyền. Khi khung số 3 đến, tầng liên kết dữ liệu phát hiện lỗi về số thứ tự khung chờ nhận, vì thế nó gởi khung NAK cho khung số 2 và lưu tạm thời khung số 3 vào vùng nhớ đệm. Tương tự, các khung 4 và 5 cũng được lưu lại mà chưa chuyển lên tầng mạng (vì phải chờ nhận khung số 2).
Khi khung NAK 2 đến bên gởi, nó truyền lại ngay khung số 2.
Khi khung số 2 đến bên nhận, nó đã có đủ các khung 2,3,4,5 theo đúng thứ tự vì thế nó chuyển 4 khung này lên tầng mạng theo một thứ tự đúng đắn. Đồng thời bên nhận gởi về bên gởi khung ACK 5 để báo rằng đã nhận tốt đến khung số 5.
Trong trường hợp khung NAK2 bị mất, không đến được bên gởi, thì sự kiện quá thời gian sẽ xảy ra. Khi đó bên gởi cũng chỉ gởi lại khung số 2 mà thôi.

/* Giao thức này chấp nhận các khung đến không đúng thứ tự, nhưng chúng lại được chuyển lên tầng mạng theo một thứ tự đúng đắn. Với mỗi khung gởi đi, sẽ có một bộ đếm thời gian đi kèm. Khi quá thời hạn chỉ khung tương ứng được gởi lại, thay vì gởi lại tất cả các khung như giao thức Go-Back-N */

```
#define MAX_SEQ 7                                   /* Số thứ tự khung lớn nhất*/
#define NR_BUFS ((MAX_SEQ + 1)/2)                   /* Kích thước tối đa cửa sổ gởi và nhận
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready, ack_timeout} event_type;
#include "protocol.h"
boolean no_nak = true;                              /* Chưa gởi khung NAK*/
seq_nr oldest_frame = MAX- SEQ + 1;                 /* Khung cũ nhất đã gởi */

static boolean between(seq_nr a, seq_nr b, seq_nr c)
{
/*  Kiểm tra b có là giá trị giữa a và c không                                  */
  return ((a <= b) && (b < c)) ll ((c < a) && (a <= b)) ll ((b < c) && (c < a));
}

static void send_frame(frame_kind fk, seq_nr frame_nr, seq_nr frame_expected, packet buffer[ ])
{
/* Tạo và gởi khung dữ liệu / báo nhận / báo lỗi */
  frame s;                                          /* Khung */

  s.kind = fk;                                      /* Kiểu khung: data, ack, nak */
  if (fk == data) s.info = buffer[frame_nr % NR_BUFS];
  s.seq = frame_nr;                                 /* Đặt số thứ tự cho khung dữ liệu gởi đi */
  s.ack = (frame_expected + MAX_SEQ) % (MAX  SEQ + 1):
  if (fk == nak) no_nak = false;                    /* Chỉ gởi 1 NAK cho một khung */
  to_physical_layer(&s);                            /* Gởi khung đi */
  if (fk == data) start_timer(frame_nr % NR_BUFS)/* Khởi động bộ đếm thời gian cho khung dữ liệu gởi */
  stop_ack_timer();                                 /* Ngừng bộ đếm thời gian chờ báo nhận*/
}
void protocol6(void)
{
  seq_nr ack_expected;                              /* Khung chờ được báo nhận */
  seq_nr next_frame_to_send;                        /* Khung kế tiếp gởi đi */
  seq_nr frame_expected;                            /* Khung đang chờ nhận */
  seq_nr too_far;                                   /* Khung kế tiếp chưa được nhận */
  int i;                                            /* Chỉ số vùng nhớ tạm */
  frame r;                                          /* Khung nhận và gởi */
  packet out_buf[NR_BUFS];                          /* Vùng đệm cho dữ liệu gởi */
  packet in_buf[NR_BUFS];                           /* Vùng đệm cho dữ liệu nhận */
  boolean arrived[NR_BUFS];                         /* Theo dõi sử dụng vùng đệm dữ liệu nhận */
  seq_nr nbuffered;                                 /* Số lượng vùng đệm gởi đang được sử dụng */
  event_type event;

  enable_network_layer();
  ack_expected = 0;                                 /* Gán giá trị khởi động */                    /
  next_frame_to_send = 0;
  frame_expected = 0;
  too_far = NR_BUFS;
  nbuffered = 0;
  for (i = 0; i < NR_BUFS; i++) arrived[i] = false;  /* Khởi tạo các vùng đệm nhận rỗng */
```

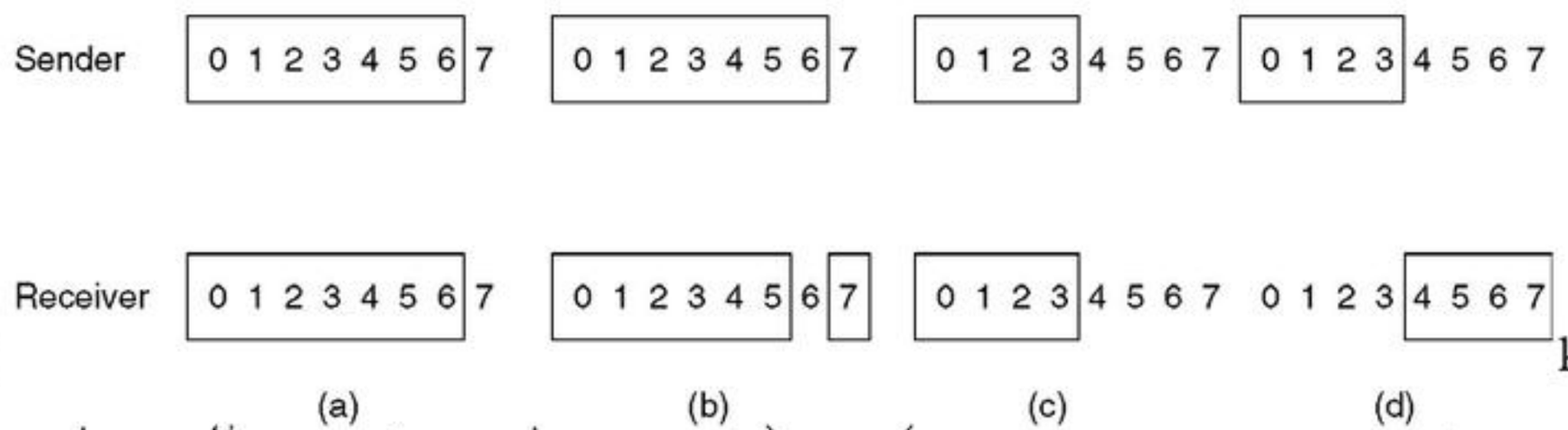
```
while (true) {
  wait_for_event(&event);                    /* Chờ 1 trong 5 sự kiện phát sinh */
  switch(event) {
    case network_layer_ready:                /* Nhận, lưu và truyền một khung mới */
          nbuffered = nbuffered + 1;         /* Mở rộng kích thước cửa sổ gởi */
          from_network_layer(&out_buf[next_frame_to_send % NR_BUFS]); /* Nhận gói tin từ tầng mạng */
          send_frame(data, next_frame_to_send, frame_expected, out_buf);/* Gởi khung đi */
          inc(next_frame_to_send);           /* Tăng số thứ tự khung gởi kế tiếp */
          break;

    case frame_arrival:                      /* Một khung dữ liệu hoặc điều khiển vừa đến */
          from_physical_layer(&r);           /* Nhận khung từ tầng vật lý */
          if (r.kind == data) {
               /* Là khung dữ liệu có lỗi thì gởi khung NAK, ngược lại tính giờ để gởi khung ACK  */
               if ((r.seq != frame_expected) && no_nak)
                  send_frame(nak, 0, frame_expected, out_buf); else start_ack_timer();
               if (between(frame_expected, r.seq, too_far) && (arrived[r.seq%NR_BUFS] == false)) {
               /* Là khung có thứ tự chấp nhận và chưa được lưu dữ liệu lại*/
                    arrived[r.seq % NR_BUFS] = true;    /* Đưa dữ liệu vào vùng đệm */
                    in_buf[r.seq % NR_BUFS] = r.info;   /* Đánh dấu vùng đệm đã sử dụng */
                    while (arrived[frame_expected % NR_BUFS]) {
               /* Chuyển tất cả các khung đang trong vùng đệm lên tầng mạng và mở rộng cửa sổ nhận */
                         to_network_layer(&in_buf[frame_expected % NR_BUFS]);
                         no_nak = true;
                         arrived[frame_expected % NR_BUFS] = false;
                         inc(frame_expected);   /* Di chuyển cửa sau cửa sổ nhận */
                         inc(too_far);          /* Di chuyển cửa trước cửa sổ nhận */
                         start_ack_timer();     /* Khởi động đồng hồ cho các báo nhận */
                    }
               }
          }

          if((r.kind==nak) && between(ack_expected,(r.ack+1)%(MAX_SEQ+1),next_frame_to_send))
               send_frame(data, (r.ack+1) % (MAX_SEQ + 1), frame_expected, out_buf);

          while (between(ack_expected, r.ack, next_frame_to_send)) {
               nbuffered = nbuffered—1;         /* Giảm kích thước cửa sổ gởi */
               stop_timer(ack_expected % NR_BUFS);/* Khung đã đến nơi thật sự */
               inc(ack_expected);               /* Di chuyển cửa dưới của cửa sổ gởi */
          }
          break;
    case cksum_err:
          if (no_nak) send_frame(nak, 0, frame_expected, out_buf);/* Khung bị lỗi */
          break;
    case timeout:
          send_frame(data, oldest_frame, frame_expected, out_buf);/* Bên gởi xử lý chậm */
          break;
    case ack_timeout:
          send_frame(ack,0,frame_expected, out_buf);   /* Quá thời gian cho các báo nhận */
    }
    if (nbuffered < NR_BUFS) enable_network_layer(); else disable_network_layer();
  }
}
```

P4.6 Cài đặt giao thức Selective-Repeat

Một số điểm cần lưu ý khi sử dụng cửa số trượt với kích thước lớn hơn 1:

4.4.6.1.1 Kích thước tối đa của cửa sổ gởi và nhận là bao nhiêu ?

Giả sử ta dùng 3 bit để đánh số cho khung. Như vậy bên gởi được phép gởi trước tối đa 7 khung trước khi chờ bên nhận gởi báo nhận về.



- kỳ
- Tất cả các khung đến nơi không có lỗi, bên nhận gởi các báo nhận và chuyển cửa số nhận về vị trí sẵn sàng để nhận các khung 7,0,1,2,3,4 và 5 (Hình b).
- Tại thời điểm đó, đường truyền có sự cố làm cho tất cả các khung báo nhận đều mất. Quá thời gian, bên gởi gởi lại khung 0. Khi khung này đến bên nhận, nó kiểm tra xem khung có nằm trong cửa sổ nhận không. Điều không may mắn đã xảy ra: khung 0 nằm trong cửa sổ nhận mới (Hình b). Bên nhận nhận khung 0 xem như một khung mới hoàn toàn và chuyển khung 0 lên tầng mạng. Như vậy tầng mạng đã nhận 2 lần cùng một gói tin, tức giao thức vận hành sai.
- Tình trạng này có thể tránh được nếu ta đảm bảo rằng cửa sổ nhận mới không đè chồng lên cửa sổ trước đó. Điều này có thể thực hiện được nếu ta giới hạn kích thước tối đa của của sổ nhận bằng một nửa khoảng đánh số thứ tự của khung.
  - Ví dụ: Nếu dùng 3 bit để đánh số thứ tự khung từ 0 đến 7 thì kích thước tối đa cửa sổ nhận là (7-0+1)/2 =4.
  - Nếu dùng 4 bit để đánh số thứ tự khung từ 0 đến 15 thì kích thước tối đa cửa sổ nhậnt là (15-0+1)/2 =8.

4.4.6.2.1 Số lượng buffer để lưu khung là bao nhiêu?

Số lượng buffer chỉ cần bằng kích thước tối đa của cửa sổ nhận, không cần thiết phải bằng số lượng khung. Ví dụ: Nếu dùng 3 bit để đánh số thứ tự khung từ 0 đến 7 thì kích thước tối đa cửa sổ nhận là (7-0+1)/2 =4 và số lượng buffer cần thiết cũng là 4.

4.4.6.2.2 Khi nào gởi báo nhận cho một gói tin?

Ta thấy rằng, khi một khung đến, báo nhận của khung này sẽ không được gởi ngược về một cách tức thì. Thay vào đó, nó sẽ được gởi kèm trong khung dữ liệu kế tiếp của bên nhận. Nếu bên nhận không có dữ liệu để gởi đi, báo nhận sẽ bị giữ lại khá lâu. Chính vì thế, mỗi khi có khung đến thì bộ đếm thời gian start_ack_timer được khởi động. Nếu trong suốt khoảng thời gian này không có một khung dữ nào cần gởi đi, thì sau đó một khung báo nhận riêng biệt sẽ được gởi đi. Bộ đếm thời gian start_ack_timer sinh ra sự kiện ack_timeout. Chúng ta cũng phải đảm bảo rằng, bộ đếm thời gian start_ack_timer thì ngắn hơn bộ đếm thời gian chờ báo nhận cho các khung dữ liệu.

4.4.7 Giao thức HDLC (High-Level Data Link Control)

Giao thức điều khiển liên kết dữ liệu quan trong nhất là HDLC. Không phải vì nó được sử dụng rộng rãi mà nó còn là cơ sở cho nhiều giao thức điều khiển liên kết dữ liệu khác.

4.4.7.2 Các đặc tính của giao thức HDLC

Giao thức HDLC định nghĩa 3 loại máy trạm, hai cấu hình đường nối kết và 3 chế độ điều khiển truyền tải

4.4.7.2.1 Ba loại trạm trong HDLC

- Trạm chính (Primary Station): Có trách nhiệm điều khiển các thao thác về đường truyền. Các khung được gởi từ trạm chính gọi là lệnh (Command).
- Trạm phụ (Secondary Station): Hoạt động dưới sự kiểm soát của trạm chính. Khung gởi từ trạm phụ gọi là các trả lời. Trạm chính duy trì nhiều đường nối kết luận lý đến các trạm phụ trên đường truyền.
- Trạm hỗn hợp (Combined Station): Bao gồm đặc điểm của trạm chính và trạm phụ. Một trạm hỗn hợp có thể gởi đi các lệnh và các trả lời.

4.4.7.2.2 Hai cấu hình đường nối kết:

- Cấu hình không cân bằng (Unbalanced Configuration): Gồm một máy trạm chính (Primary Station) và nhiều máy trạm phụ (Secondary station) và hỗ trợ cả 2 chế độ truyền song công và bán song công
- Cấu hình cân bằng (Balanced Configuration): Bao gồm 2 máy trạm hỗn hợp, và hỗ trợ cả 2 chế độ truyền song công và bán song công.

4.4.7.2.3 Có 3 chế độ truyền tải là:

- Chế độ trả lời bình thường (NRM- Normal Response Mode), được sử dụng với cấu hình đường nối kết không cân bằng. Máy chính có thể khởi động một cuộc truyền tải dữ liệu về cho máy phụ. Nhưng máy phụ chỉ có thể thực hiện việc truyền dữ liệu cho máy chính như là những trả lời cho các yêu cầu của máy chính.
- Chế độ cân bằng bất đồng bộ (ABM - Asynchronous Response Mode): Được sử dụng với cấu hình nối kết cân bằng. Cả hai máy đều có quyền khởi động các cuộc truyền tải dữ liệu mà không cần sự cho phép của máy kia.
- Chế độ trả lời bất đồng bộ (ARM-Asynchronous Response Mode): Sử dụng cấu hình không cân bằng. Một máy phụ có thể khởi động một cuộc truyền tải và không cần sự cho phép tường minh của máy chính. Máy chính vẫn đảm trách vai trò bảo trì đường truyền bao gồm việc khởi động, phục hồi lỗi và xóa nối kết.

Chế độ NRM đòi hỏi phải có nhiều đường dây để nối một máy chính với nhiều thiết bị đầu cuối. Chế độ ABM được sử dụng nhiều nhất trong 3 chế độ, nó cho phép sử dụng hiệu quả đường truyền. Chế độ ARM thì ít được dùng đến.

4.4.7.3 Cấu trúc khung

HDLC sử dụng chế độ truyền tải đồng bộ, các bits dữ liệu truyền đi được gói vào trong các khung và sử dụng một cấu trúc khung cho tất cả các loại dữ liệu cũng như thông tin điều khiển.
Khung trong giao thức HDLC có cấu trúc như sau:

Bits	8	8	8	≥ 0	16	8
	0 1 1 1 1 1 1 0	Address	Control	Data	Checksum	0 1 1 1 1 1 1 0

H4.12 Cấu trúc khung của HDLC

Flag (8 bit)	Là cờ dùng để xác định điểm bắt đầu và kết thúc của khung, giá trị nó là 01111110. HDLC sử dụng kỹ thuật bit độn để loại trừ sự xuất hiện của cờ trong dữ liệu.
Address (8 bit)	Vùng ghi địa chỉ để xác định máy phụ được phép truyền hay nhận khung.
Control (8bit)	Được dùng để xác định loại khung. Mỗi loại có thông tin điều khiển khác nhau. Có 3 loại khung: Thông tin (I), Điều khiển (S) và không đánh số (U).
Information (128-1024 bytes)	Vùng chứa dữ liệu cần truyền.

FCS (Frame Check Sequence- 8 bit)	Vùng chứa mã kiểm soát lỗi, dùng phương pháp đa thức **CRC-CCITT = $X^{16} + X^{12} + X^5 + 1$**

Giá trị 8 bit của trường control hình thành 3 loại khung như sau:

Bits	1	3	1	3
Khung I	0	Seq	P/F	Next
Khung S	1	0	Type	P/F	Next
Khung U	1	1	Type	P/F	Modifier

H4.13 Cấu trúc trường điều khiển trong khung HDLC

Giao thức HDLC sử dụng một cửa sổ trượt với số thứ tự khung 3 bít. Trường seq trong khung I để chỉ số thứ tự của khung thông tin hiện tại. Trường Next để chỉ số thứ tự của khung thông tin mà bên gởi đang chờ nhận (thay vì là khung đã nhận tốt như giao thứ cửa sổ trượt đã giới thiệu ở phần trước).
Bit P/F có ý nghĩa là Poll/Final, tức chọn hoặc kết thúc. Khi máy tính chính mời một máy phụ truyền tin, thì bit này được đặt lên 1 có ý nghĩa là P (Poll, chọn). Ngược lại khi thông tin được truyền từ máy phụ lên máy chính thì nó được đặt xuống 0, để báo với máy chính rằng máy phụ hiện tại vẫn còn dữ liệu để gởi đi. Khi máy phụ gởi khung cuối cùng, bit này được đặt lên 1, có ý nghĩa là F (Final, kết thúc), để báo cho máy chính biết rằng nó đã hoàn thành việc truyền tải thông tin.
Khung S (Supervisory Frame) là khung điều khiển, dùng để kiểm soát lỗi và luồng dữ liệu trong quá trình truyền tin. Khung S có 4 kiểu được xác định bởi tổ hợp giá trị của 2 bit trong trường Type.

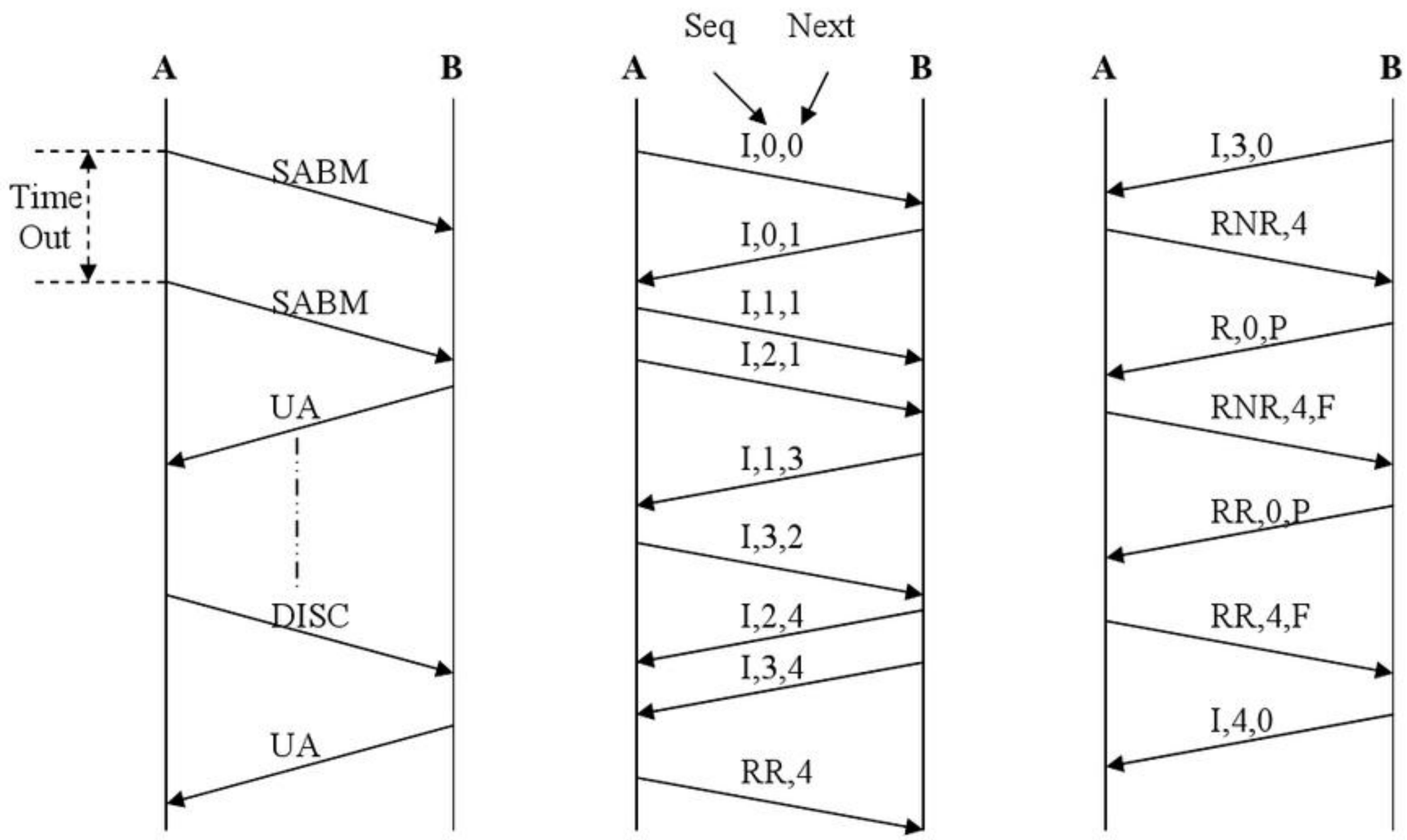
SS=0 0	RR (Receive Ready), là khung báo nhận, thông báo sẵn sàng nhận dữ liệu, đã nhận tốt đến khung Next-1, và đang đợi nhận khung Next. Được dùng đến khi không còn dữ liệu gởi từ chiều ngược lại để vừa làm báo nhận (figgyback)
SS=0 1	REJ (Reject): đây là một khung báo không nhận (negative acknowledge), yêu cầu gởi lại các khung, từ khung Next.
SS=1 0	RNR (Receive Not Ready): thông báo không sẵn sàng nhận tin, đã nhận đến đến khung thứ Next-1, chưa sẵn sàng nhận khung Next
SS=1 1	SREJ (Selective Reject): yêu cầu gởi lại một khung có số thức tự là Next

Khung U (Unnumbered Frame) thường được sử dụng cho mục đích điều khiển đường truyền, nhưng đôi khi cũng được dùng để gởi dữ liệu trong dịch vụ không nối kết. Các lệnh của khung U được mô tả như sau:

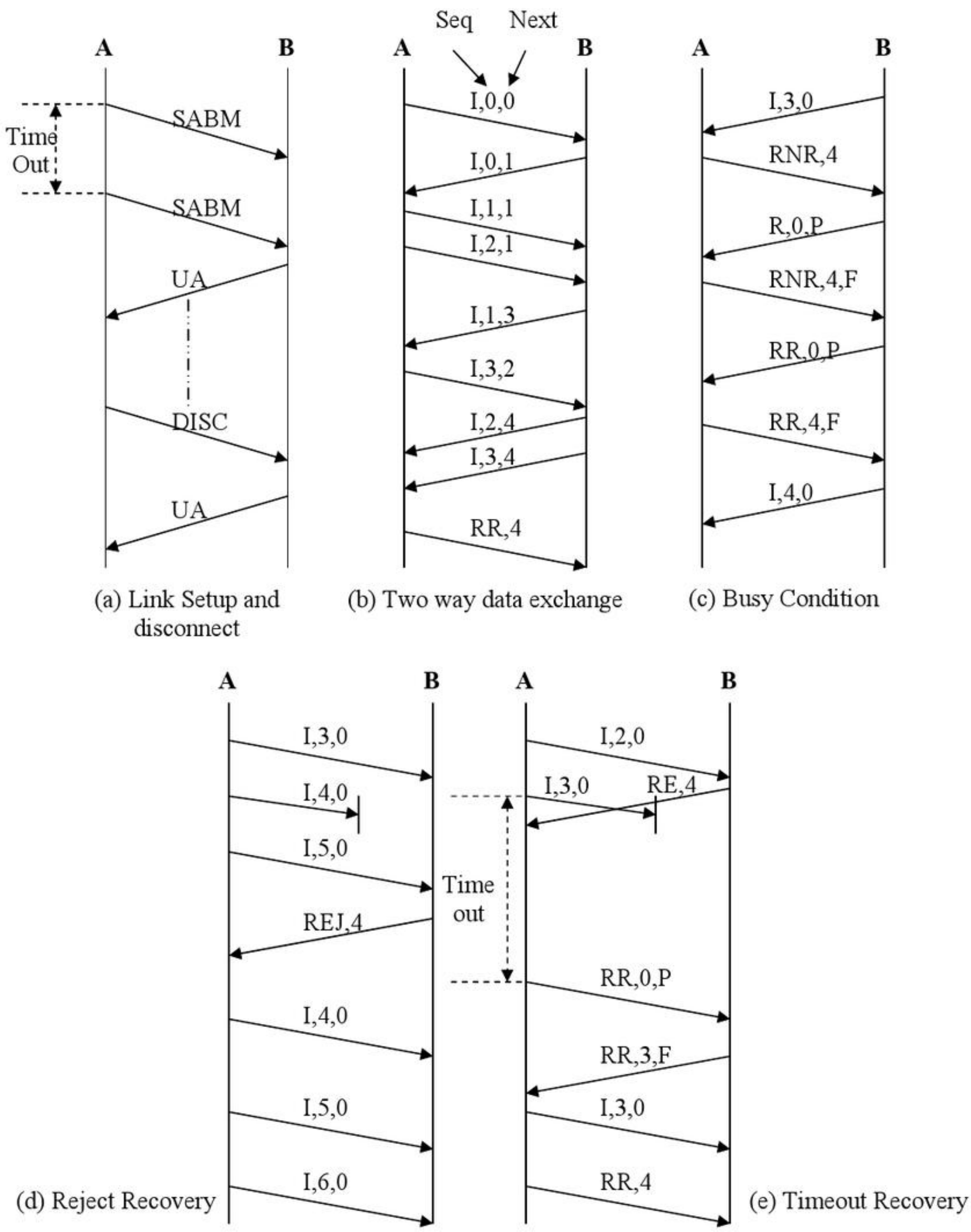
1111P100	Lệnh này dùng để thiết lập chế độ truyền tải SABM (Set Asynchronous Balanced Mode).
1100P001	Lệnh này dùng để thiết lập chế độ truyền tải SNRM (Set Normal Response Mode).
1111P000	Lệnh này dùng để thiết lập chế độ truyền tải SARM (Set Asynchronous Response Mode).
1100P010	Lệnh này để yêu cầu xóa nối kết DISC (Disconnect).
1100F110	UA (Unumbered Acknowledgment). Được dùng bởi các trạm phụ để báo với trạm chính rằng nó đã nhận và chấp nhận các lệnh loại U ở trên.
1100F001	CMDR/FRMR (Command Reject/Frame Reject). Được dùng bởi trạm phụ để báo rằng nó không chấp nhận một lệnh mà nó đã nhận chính xác.

4.4.7.4 Một vài kịch bản về giao thức HDLC

Kịch bản (a) mô tả các khung liên quan trong quá trình thiết lập và xóa nối kết. Đầu tiên một trong hai bên giao tiếp sẽ gởi khung SABM sang bên kia và thiết lập một bộ đếm thời gian. Bên phía còn lại khi nhận được khung SABM sẽ trả lời bằng khung UA. Bên yêu cầu nối kết khi nhận được khung UA sẽ xóa bỏ bộ đếm thời gian. Nối kết đã được hình thành và hai bên có thể truyền khung qua lại cho nhau. Nối kết sẽ xóa đi nếu một trong hai bên giao tiếp gởi khung DISC. Trong một trường hợp khác, nếu sau một khoảng thời gian trôi qua, bên yêu cầu nối kết không nhận được khung UA, nó sẽ cố gắng gởi lại khung SABM một số lần qui định. Nếu vẫn không nhận được khung UA, bên yêu cầu nối kết sẽ thông báo lỗi lên tầng cao hơn.



(a) Link Setup and disconnect (b) Two way data exchange (c) Busy Condition

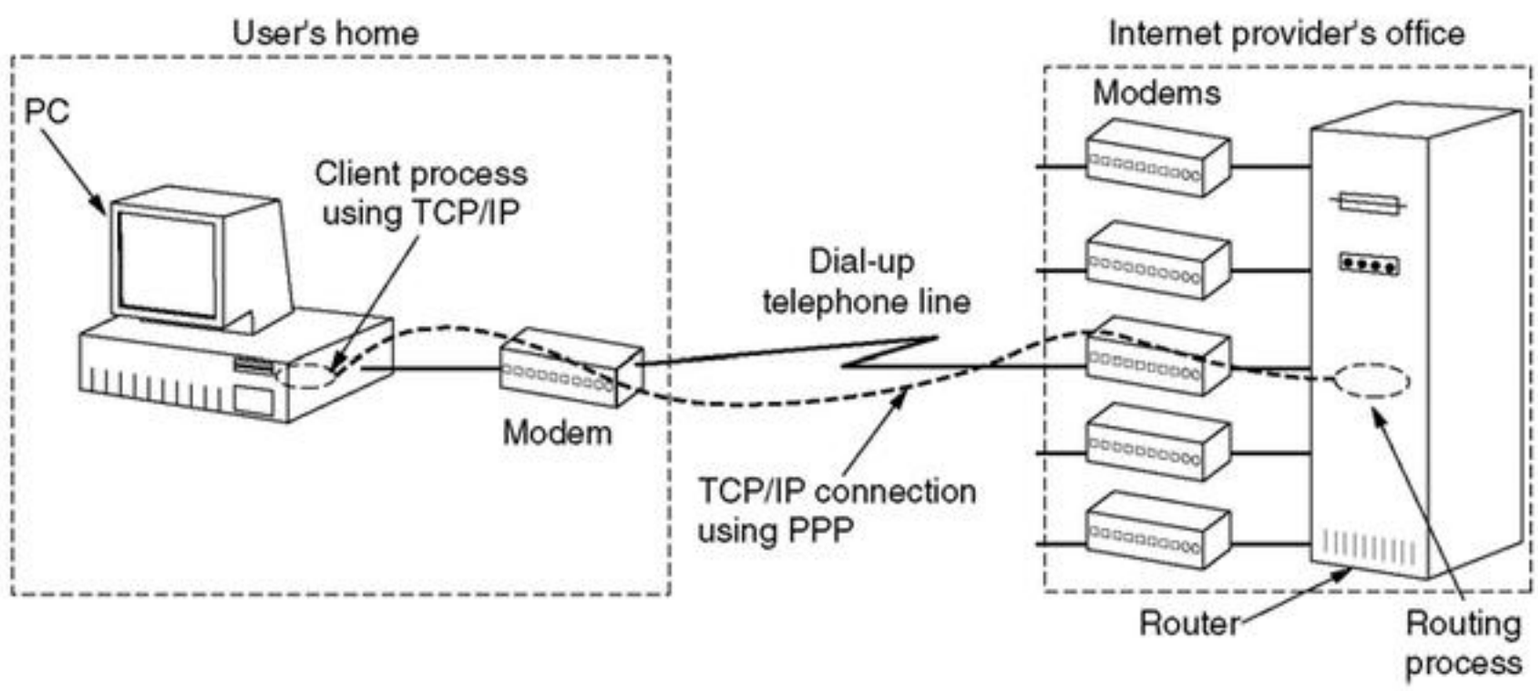


H4.14 Một vài kịch bản của HDLC

Kịch bản (b) mô tả tiến trình trao đổi khung I giữa hai bên. Ta thấy rằng bên A gởi liên tiếp các khung (I,1,1 và I,2,1) mà không nhận được khung báo nhận thì số thứ tự của khung chờ nhận vẫn không thay đổi, trong trường hợp này là 1. Ngược lại khi bên B nhận liên tiếp các khung (I,1,1 và I,2,1) mà không gởi khung nào đi, thì khung chờ nhận kế tiếp của khung thông tin truyền đi phải là số kế tiếp của khung vừa nhận, là 3.
Trong kịch bản (c) máy A không thể xử lý kịp các khung do B gởi đến vì thế nó gởi khung RNR để yêu cầu B tạm dừng việc việc truyền tải. Bên B định kỳ gởi thăm dò bên A bằng cách gởi khung RR với bit P được đặt lên 1. Nếu bên A vẫn chưa thể nhận thông tin từ bên B nó sẽ trả lời bằng khung RNR, ngược lại nếu A đã sẵn sàng thì nó sẽ trả lời bằng khung RR.
Trong kịch bản (d), bên A gởi sang B ba khung thông tin 3,4 và 5. Khung 4 bị mất hoàn toàn trên đường truyền. Khi bên B nhận được khung 5, nó sẽ bỏ qua khung này vì sai thứ tự khung. B gởi REJ với trường Next là 4 để yêu cầu A gởi lại tất cả các khung từ khung số 4.
Kịch bản (e) minh họa cách thức phục hồi lỗi dựa vào thời gian (timeout). Khung số 3 bị lỗi và do đó B bỏ nó. B không thể gởi khung REJ vì nó không thể xác định được đó có phải là khung I hay không. Bên A sau một khoảng thời gian trôi qua không thấy khung trả lời từ B, nó sẽ gởi khung RR với bit P=1 để kiểm tra trạng thái của bên kia. Bên B sẽ đáp lại bằng khung RR với trường Next là 3 để báo hiệu khung số 3 đã mất. Sau đó A sẽ truyền lại khung số 3.

4.4.7.5 Giao thức Điểm nối điểm (PPP- Point-to-Point Protocol)

PPP là một giao thức đặc biệt quan trọng trong mạng Internet. Nó cho phép truyền tải thông tin giữa các router trên mạng hay để cho phép nối các máy tính người dùng vào mạng của nhà cung cấp dịch vụ Internet (ISP).



H4.15 Sơ đồ nối kết của giao thức PPP

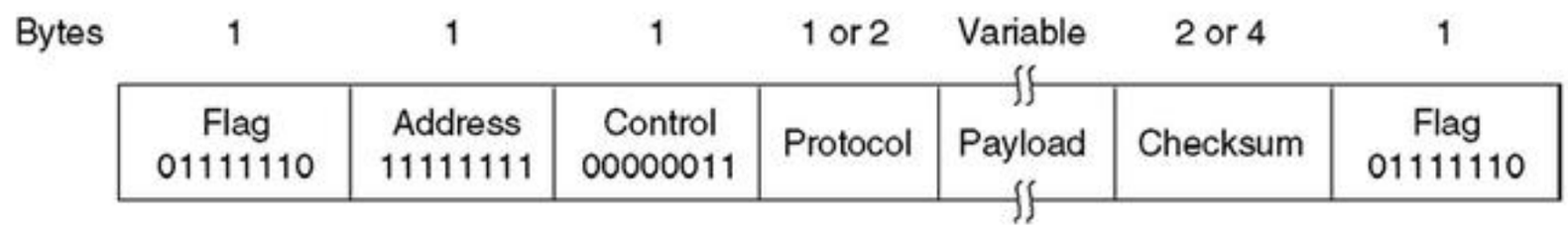
Giao thức PPP được định nghĩa trong RFC (Request For Comments) 1661 và sau đó được mở rộng thêm bằng các RFC 1662, RFC 1663. PPP thực hiện chức năng phát hiện lỗi trên dữ liệu truyền, hỗ trợ nhiều giao thức vận hành trên nó, phân phối địa chỉ IP khi máy tính nối kết vào mạng, kiểm tra quyền đăng nhập và nhiều tính năng khác.
PPP cung cấp 3 đặc tính sau:

Định nghĩa một phương pháp định khung cùng với phương pháp phát hiện lỗi.

1. Giao thức điều khiển đường truyền cho phép thiết lập kênh giao tiếp, kiểm tra kênh, thỏa thuận về các thông số truyền tin và xóa kênh truyền khi không cần thiết nữa. Giao thức này được gọi là giao thức LCP (Link Control Protocol).
2. Có phương pháp thương lượng về các tùy chọn tầng mạng một cách độc lập với giao thức mạng được sử dụng. Phương pháp được chọn lựa NCP (Network Control Protocol) khác nhau cho mỗi giao thức mạng.

Để hiểu rõ về giao thức PPP, ta xét trường hợp quay số nối kết máy tính ở nhà vào mạng của một ISP.

Đầu tiên máy tính các nhân sẽ quay số thông qua modem đến router của ISP. Router sẽ tiếp nhận cuộc gọi và một nối kết vật lý được hình thành. Máy tính sẽ gởi một loạt các gói tin theo giao thức LCP trong một hoặc nhiều khung của giao thức PPP để thỏa thuận về các thông số mà PPP sẽ sử dụng.
Sau đó một loạt các gói tin của giao thức NCP sẽ được gởi đi để thực hiện cấu hình tầng mạng. Thông thường máy tính muốn sử dụng giao thức TCP/IP nên nó cần một địa chỉ IP. Giao thức NCP sẽ gán địa chỉ IP cho máy tính. Từ lúc này, máy tính đóng vai trò như một máy trên mạng Internet. Nó có thể gởi và nhận các gói tin của giao thức IP. Khi người dùng kết thúc, NCP xóa đi nối kết của tầng mạng và giải phóng địa chỉ IP của máy tính để sử dụng cho các máy tính khác nối vào sau đó. Giao thức LCP sẽ xóa nối kết của tầng liên kết dữ liệu. Và cuối cùng máy tính sẽ yêu cầu modem kết thúc cuộc gọi (Hang up) và giải phóng nối kết ở tầng vật lý.
Khung của giao thức PPP tương tự như khung của giao thức HDLC, tuy nhiên đây là khung theo kiểu hướng ký tự. Nó sử dụng kỹ thuật byte độn.



H4.16 Cấu trúc khung của giao thức PPP

PPP sử dụng byte đặc biệt 01111110 để làm cờ đánh dấu điểm bắt đầu và kết thúc của khung.
Địa chỉ 11111111 để chỉ rằng tất cả các trạm đều nhận khung. Nhờ đó giao thức LCP không cần thiết phải đánh địa chỉ cho các trạm.
Trường Control có giá trị 00000011 để biểu thị rằng giao thức không sử dụng cơ chế báo nhận dựa trên số thứ tự của khung.
Trường Protocol để xác định phần gói tin được chứa đựng trong phần Payload được định nghĩa bởi giao thức mạng nào. Mỗi protocol đã được qui định một giá trị riêng. Bit đầu tiên là 0 được sử dụng cho các giao thức mạng IP, IPX, OSI CLNP, XNS. Kích thước mặc định là 2 bytes, tuy nhiên giao thức LCP có thể thỏa thuận để sử dụng 1 byte.
Payload là nơi chứa gói tin với chiều dài khác nhau. Chiều dài tối đa mặc định là 1500 bytes, tuy nhiên LCP có thể thỏa thuận để thay đổi.
Cuối cùng là trường checksum dùng để kiểm tra lỗi trong khung.

Chương 5: MẠNG NỘI BỘ & LỚP CON ĐIỀU KHIỂN TRUY CẬP

Mục đích

Chương này nhằm giới thiệu với người học những nội dung sau:

- Các phương chia sẻ đường truyền chung giữa các máy tính trong một mạng cục bộ như: các phương pháp chia kênh, các phương pháp truy cập đường truyền ngẫu nhiên và các phương pháp phân lượt truy cập đường truyền.
- Giới thiệu chi tiết về nguyên tắc hoạt động của các chuẩn mạng cục bộ như họ các chuẩn mạng Ethernet, FDDI và mạng không dây

Yêu cầu

Sau khi học xong chương này, người học phải có được các khả năng sau:

- Trình bày được sự khác biệt cơ bản về cách thức chia sẻ đường truyền chung giữa các máy tính trong các phương pháp chia kênh, truy cập đường truyền ngẫu nhiên và phân lượt truy cập đường truyền.
- Trình bày được nguyên tắc chia sẻ đường truyền chung giữa các máy tính theo các phương pháp FDMA, TDMA, CDMA, ALOHA, CSMA, CAMA/CD, Token Passing, …
- Trình bày được những đặc điểm và nguyên tắc hoạt động của các chuẩn thuộc họ mạng Ethernet, mạng FDDI và chuẩn mạng không dây 802.11

5.1 Tổng quan về LAN

Như đã trình bày trong phần 2.1, theo tiêu chí đánh giá là khoảng cách địa lý thì người ta thường phân loại mạng máy tính thành ba kiểu:

- Mạng nội bộ - Local Area Network (LAN)
- Mạng đô thị - Metropolitan Area Network (MAN)
- Mạng diện rộng - Wide Area Network (WAN)

Trong thực tế, LAN và WAN thường được cài đặt nhất.

Mạng LAN được sử dụng để nối kết một dãi rộng các thiết bị trong một phạm vi hẹp, ví dụ: trên cùng một tầng, một tòa nhà hay một khuôn viên (thường không vượt quá 10Km). Ngày nay, LAN là loại mạng được sử dụng rất phổ biến trong mọi lĩnh vực của xã hội. Người ta thường nghĩ đến LAN như là mạng có thông lượng cao, độ trì hoãn thấp.

Hiện tại có rất nhiều công nghệ xây dựng mạng LAN mà chúng ta sẽ xem xét đến ngay sau đây. Nhiều chuẩn mạng LAN đã được phát triển trong đó **Ethernet** và **FDDI** là phổ biến nhất. Người ta thường gọi chung họ các chuẩn mạng LAN là **IEEE 802.**

Về góc độ kỹ thuật, LAN có các tính chất quan trọng sau:

- Tất cả các host trong mạng LAN cùng chia sẻ đường truyền chung. Do đó chúng hoạt động dựa trên kiểu quảng bá (broadcast).
- Không yêu cầu phải có hệ thống trung chuyển (routing/switching) trong một LAN đơn.

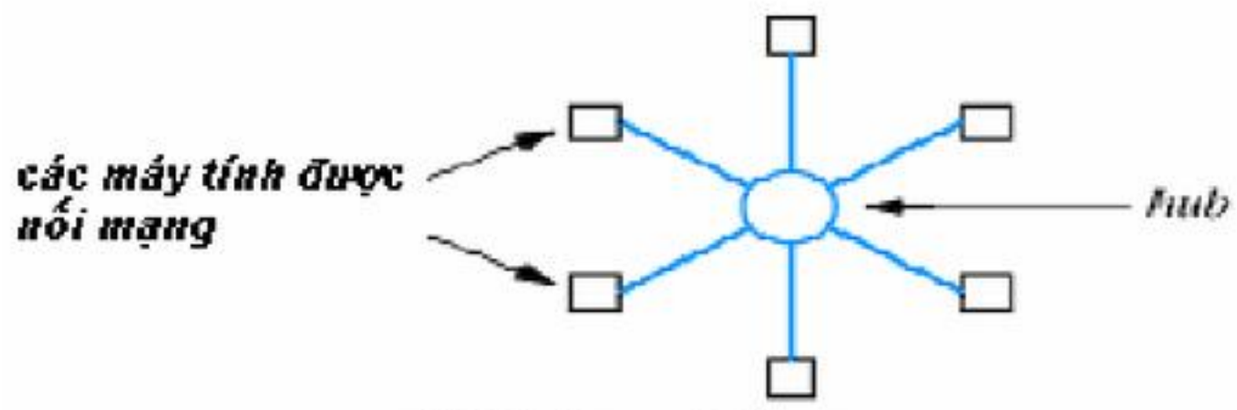
Thông thường, một mạng LAN được định nghĩa dựa trên các thông số sau:

- Hình thái (topology): Chỉ ra kiểu cách mà các host trong mạng được đấu nối với nhau.
- Đường truyền chia sẻ (xoắn đôi, đồng trục, cáp quang): Chỉ ra các kiểu đường truyền mạng (network cables) được dùng để đấu nối các host trong LAN lại với nhau. (Xin xem lại mô tả chi tiết các kiểu đường truyền trong chương Tầng Vật Lý).
- Kỹ thuật truy cập đường truyền (Medium Access Control - MAC): Chỉ ra cách thức mà các host trong mạng LAN sử dụng để truy cập và chia sẻ đường truyền mạng. MAC sẽ quản trị việc truy cập đến đường truyền trong LAN và cung cấp cơ sở cho việc định danh các tính chất của mạng LAN theo chuẩn IEEE.

5.2 Hình thái mạng

Hình thái mạng sẽ xác định hình dáng tổng quát của một mạng. Hiện tại, người ta đã định nghĩa ra được nhiều hình thái mạng khác nhau tương ứng với những tính chất đặc thù của chúng. Hình thái mạng là tiêu chí bắt buộc dùng để xây dựng mạng LAN và nó chủ yếu quan tâm đến việc làm cho mạng được liên thông, che dấu chi tiết về các thiết bị thực đối với người dùng.

5.2.1 Mạng hình sao

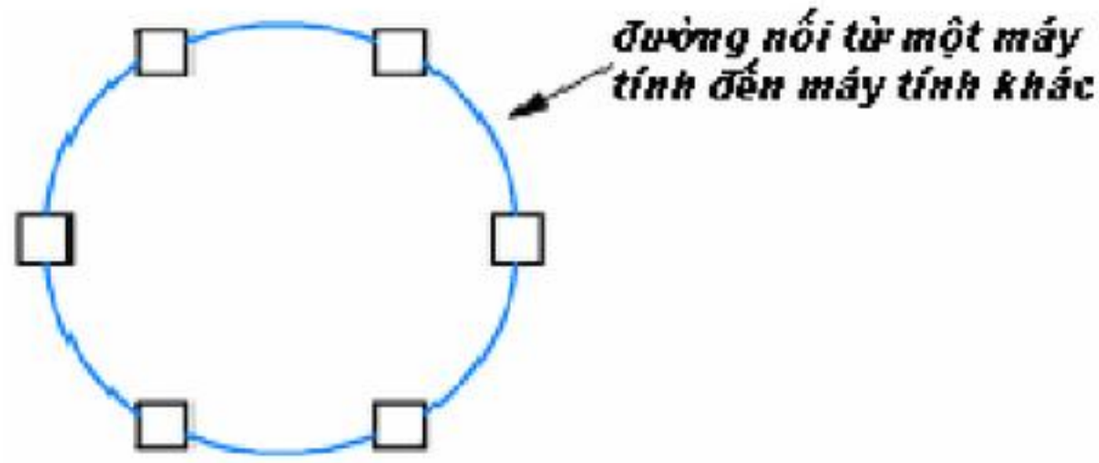


H5.1 Sơ đồ mạng hình sao

Tất cả các máy tính trong mạng được đấu nối tới một thiết bị tập trung tín hiệu trung tâm. Thành phần trung tâm của mạng được gọi là H*ub*.

Phương thức hoạt động của mạng hình sao như sau: Mọi máy tính đều phát tín hiệu ra Hub và Hub phát lại tín hiệu vào đến tất cả các đầu ra. Mỗi máy tính có một nối kết riêng lẻ đến Hub

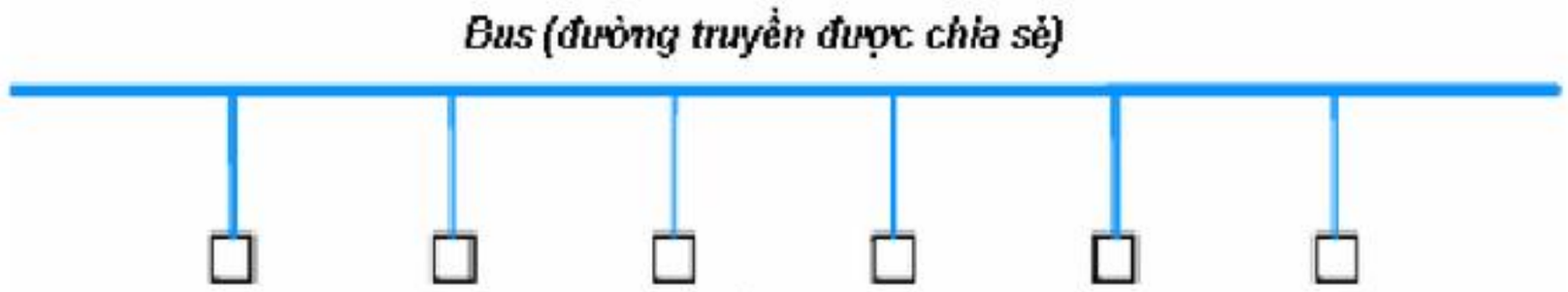
5.2.2 Mạng hình vòng



H5.2 Sơ đồ mạng hình vòng

Không có thiết bị trung tâm trong sơ đồ nối mạng hình vòng. Đường nối kết mạng sẽ đi trực tiếp từ một máy tính đến máy tính khác.
Thực tế, có một đoạn cable ngắn nối máy tính với vòng.

5.2.3 Mạng hình bus



H5.3 Sơ đồ mạng hình bus

Trong sơ đồ mạng hình bus, người ta dùng một dây cáp (cable) đơn nối kết toàn bộ LAN. Mỗi máy tính có một đầu nối đến cáp được chia sẻ.
Với một đường truyền chia sẻ như thế thì sẽ có khả năng đụng độ xảy ra khi các máy tính cùng phát tín hiệu ra đường truyền cùng một lúc. Do đó, phải có giải pháp làm cho các máy tính hoạt động đồng bộ với nhau nhằm cho phép chỉ một máy tính truyền thông tin tại một thời điểm.

5.3 Lớp con MAC (Media Access Control Sublayer)

Như đã trình bày ở trên, chương này trình bày về mạng LAN – mạng dạng truyền quảng bá và các giao thức truyền quảng bá của nó.
Trong bất kỳ mạng dạng quảng bá nào, vấn đề then chốt luôn là cách thức người ta quyết định ai có quyền truy cập kênh truyền tại một thời điểm. Để làm rõ vấn đề hơn, hãy xem xét ví dụ sau: Có sáu người đang họp thông qua hệ thống điện thoại, mọi người đều được nối kết để có thể nghe và nói với những người khác. Khi một người ngừng nói mà có hai người hoặc nhiều hơn cùng phát biểu tiếp sẽ tạo ra tình trạng lộn xộn. Trong các cuộc họp dạng gặp mặt trực tiếp, tình trạng lộn xộn này có thể được giải quyết bằng cách đưa tay xin phát biểu. Nhưng trong hệ thống hội thảo thông qua điện thoại này, khi mà đường truyền rảnh, việc quyết định ai sẽ nói tiếp có vẻ khó làm hơn. Đã có nhiều giao thức dùng giải quyết vấn đề trên. Và chúng chính là nội dung trình bày của phần này. Nói một cách khác, các kênh truyền dạng quảng bá thỉnh thoảng còn được gọi là các kênh đa truy cập (multiaccess channels) hay là các kênh truy cập ngẫu nhiên (random access channels).
Các giao thức được sử dụng để quyết định ai có quyền truy cập đường truyền quảng bá trước được gom vào trong một lớp con của tầng liên kết dữ liệu gọi là lớp con MAC. Lớp con MAC là đặc biệt quan trọng trong mạng LAN, do nhiều mạng LAN sử dụng đường truyền dạng quảng bá như là phương tiện truyền thông nền tảng. Các mạng WAN, theo xu hướng ngược lại, lại dùng các nối kết dạng điểm-điểm (ngoại trừ các mạng dùng vệ tinh).
Về cơ bản, có ba phương pháp điều khiển truy cập đường truyền: Chia kênh, truy cập ngẫu nhiên (Random Access) và phân lượt ("Taking-turns"). Giải thích cụ thể về ba phương pháp điều khiển truy cập đường truyền trên sẽ được trình bày ngay sau đây.

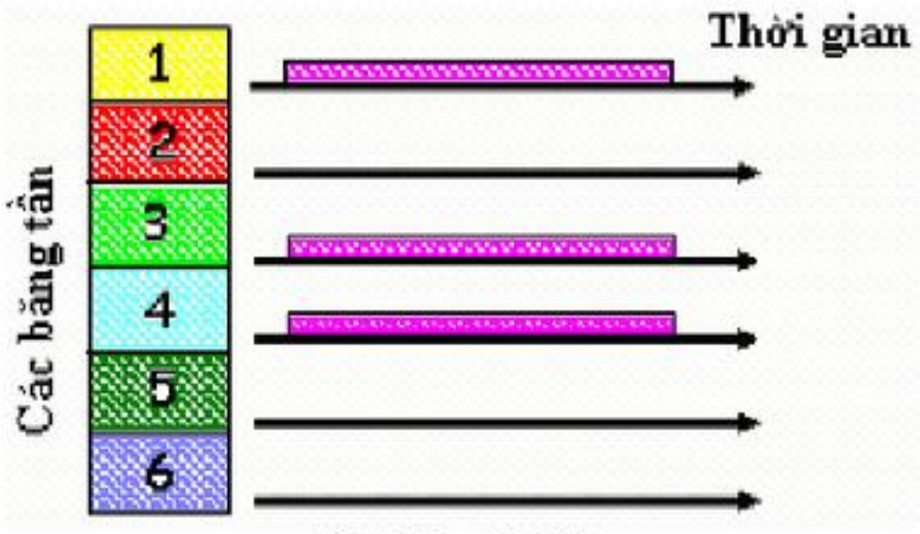
5.3.1 Phương pháp chia kênh

Ý tưởng chung của phương pháp này là: đường truyền sẽ được chia thành nhiều kênh truyền, mỗi kênh truyền sẽ được cấp phát riêng cho một trạm. Có ba phương pháp chia kênh chính: FDMA, TDMA, CDMA.

5.3.1.1 Chia tần số (FDMA – Frequency Division Multiple Access)

Một phương thức truyền thống để chia sẻ một kênh truyền đơn cho nhiều người dùng cạnh tranh là Chia tần số (FDMA). Phổ của kênh truyền được chia thành nhiều băng tần (frequency bands) khác nhau. Mỗi trạm được gán cho một băng tần cố định. Những trạm nào được cấp băng tần mà không có dữ liệu để truyền thì ở trong trạng thái nhàn rỗi (idle).

Ví dụ: Một mạng LAN có sáu trạm, các trạm 1, 3, 4 có dữ liệu cần truyền, các trạm 2, 5, 6 nhàn rỗi.



H5.4 Mạng FDMA

Nhận xét:

- Do mỗi người dùng được cấp một băng tần riêng, nên không có sự đụng độ xảy ra. Khi chỉ có số lượng người dùng nhỏ và ổn định, mỗi người dùng cần giao tiếp nhiều thì FDMA chính là cơ chế điều khiển truy cập đường truyền hiệu quả.
- Tuy nhiên, khi mà lượng người gởi dữ liệu là lớn và liên tục thay đổi hoặc đường truyền vượt quá khả năng phục vụ thì FDMA bộc lộ một số vấn đề. Nếu phổ đường truyền được chia làm N vùng và có ít hơn N người dùng cần truy cập đường truyền, thì một phần lớn phổ đường truyền bị lãng phí. Ngược lại, có nhiều hơn N người dùng có nhu cầu truyền dữ liệu thì một số người dùng sẽ phải bị từ chối không có truy cập đường truyền vì thiếu băng thông. Tuy nhiên, nếu lại giả sử rằng số lượng người dùng bằng cách nào đó luôn được giữ ổn định ở con số N, thì việc chia kênh truyền thành những kênh truyền con như thế tự thân là không hiệu quả. Lý do cơ bản ở đây là: nếu có vài người dùng rỗi, không truyền dữ liệu thì những kênh truyền con cấp cho những người dùng này bị lãng phí.
- Có thể dễ dàng thấy được hiệu năng nghèo nàn của FDMA từ một phép tính theo lý thuyết xếp hàng đơn giản. Bắt đầu là thời gian trì hoãn trung bình T trong một kênh truyền có dung lượng C bps, với tỉ lệ đến trung bình là λ khung/giây, mỗi khung có chiều dài được chỉ ra từ hàm phân phối mũ với giá trị trung bình là $1/\mu$ bit/khung. Với các tham số trên ta có được tỉ lệ phục vụ là μC khung/giây. Từ lý thuyết xếp hàng ta có:

$$T = \frac{1}{\mu C - \lambda}$$

Ví dụ: nếu C = 100 Mbps, $1/\mu$ = 10000 bits và λ = 5000 khung/giây thì T = 200 μs.
Bây giờ nếu ta chia kênh lớn này thành N kênh truyền nhỏ độc lập, mỗi kênh truyền nhỏ có dung lượng C/N bps. Tỉ lệ trung bình các khung đến các kênh truyền nhỏ bây giờ là λ/N. Tính toán lại T chúng ta có:

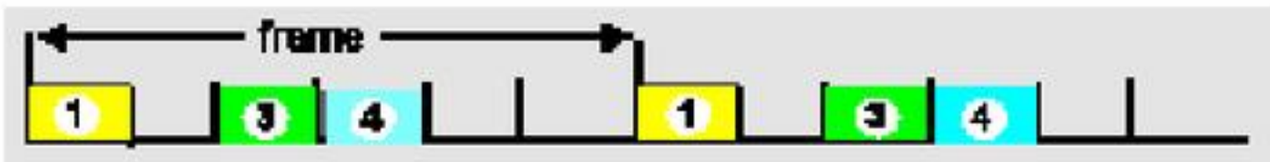
$$T_{FDMA} = \frac{1}{\mu(C/N) - (\lambda/N)} = \frac{N}{\mu C - \lambda} = NT$$

Thời gian chờ đợi trung bình trong các kênh truyền con sử dụng FDMA là xấu hơn gấp N lần so với trường hợp ta sắp xếp cho các khung được truyền tuần tự trong một kênh lớn.

5.3.1.2 Chia thời gian (TDMA – Time Division Multiple Access)

Trong phương pháp này, các trạm sẽ xoay vòng (round) để truy cập đường truyền. Vòng ở đây có thể hiểu là vòng thời gian. Một vòng thời gian là khoảng thời gian đủ để cho tất cả các trạm trong LAN đều được quyền truyền dữ liệu. Qui tắc xoay vòng như sau: một vòng thời gian sẽ được chia đều thành các khe (slot) thời gian bằng nhau, mỗi trạm sẽ được cấp một khe thời gian – đủ để nó có thể truyền hết một gói tin. Những trạm nào tới lượt được cấp cho khe thời gian của mình mà không có dữ liệu để truyền thì vẫn chiếm lấy khe thời gian đó, và khoảng thời gian bị chiếm này được gọi là thời gian nhàn rỗi (idle time). Tập hợp tất cả các khe thời gian trong một vòng được gọi lại là khung (frame).

Ví dụ:



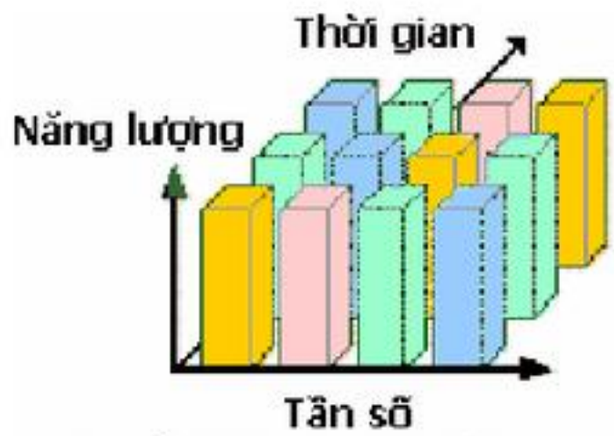
H5.5 Mạng TDMA

Mạng LAN dùng cơ chế truy cập đường truyền TDMA trên có sáu trạm. Các trạm 1, 3, 4 có dữ liệu cần truyền. Các trạm 2, 5, 6 nhàn rỗi.
Chúng ta cũng áp dụng cùng một nhận xét về mạng TDMA như mạng FDMA. Mỗi người dùng được cấp phát một khe thời gian. Và nếu người dùng không sử dụng khe thời gian này để truyền dữ liệu thì thời gian sẽ bị lãng phí.

5.3.1.3 Kết hợp giữa FDMA và TDMA

Trong thực tế, hai kỹ thuật TDMA và FDMA thường được kết hợp sử dụng với nhau, ví dụ như trong các mạng điện thoại di động.

- Các điện thoại di động TDMA sử dụng các kênh 30 KHz, mỗi kênh lại được chia thành ba khe thời gian. Một thiết bị cầm tay sử dụng một khe thời gian cho việc gởi và một khe khác cho việc nhận dữ liệu. Chẳng hạn như các hệ thống: Cingular (Nokia 8265, TDMA 800/ 1900 MHz, AMPS 800 mHz), AT&T Wireless.
- Hệ thống GSM sử dụng các kênh 200 KHz được chia thành 8 khe thời gian. Một thiết bị cầm tay sẽ sử dụng một khe thời gian trong hai kênh khác nhau để gởi và nhận thông tin. Các hệ thống Cingular, T-Mobile, AT&T đang chuyển sang dùng kỹ thuật này.
-



H5.6 Kết hợp giữa TDMA và FDMA

5.3.1.4 Phân chia mã (CDMA – Code Division Multiple Access)

CDMA hoàn toàn khác với FDMA và TDMA. Thay vì chia một dãy tần số thành nhiều kênh truyền băng thông hẹp, CDMA cho phép mỗi trạm có quyền phát dữ liệu lên toàn bộ phổ tần của đường truyền lớn tại mọi thời điểm. Các cuộc truy cập đường truyền xảy ra đồng thời sẽ được tách biệt với nhau bởi kỹ thuật mã hóa. CDMA cũng xóa tan lo lắng cho rằng những khung dữ liệu bị đụng độ trên đường truyền sẽ bị biến dạng. Thay vào đó CDMA chỉ ra rằng nhiều tín hiệu đồng

thời sẽ được cộng lại một cách tuyến tính! Kỹ thuật CDMA thường được sử dụng trong các kênh truyền quảng bá không dây (mạng điện thoại di động, vệ tinh …).
Trước khi đi vào mô tả giải thuật CDMA, hãy xem xét một ví dụ gần giống như sau: tại một phòng đợi trong sân bay có nhiều cặp hành khách đang chuyện trò. TDM có thể được so sánh với cảnh tượng: tất cả mọi người đều đứng giữa phòng, chờ đến lượt mình được phát biểu. FDM thì giống như cảnh tượng: mỗi một cặp được sắp vào một ô nói chuyện riêng. Còn CDMA lại giống như cảnh: mọi người đều đứng ngay trong phòng đợi, nói chuyện đồng thời, nhưng mỗi cặp chuyện trò sẽ sử dụng một ngôn ngữ riêng. Cặp nói tiếng Pháp chỉ líu lo với nhau bằng tiếng Pháp, bỏ qua mọi tiếng động không phải là tiếng Pháp và coi đó như là tiếng ồn. Vì thế, vấn đề then chốt trong CDMA là khả năng rút trích ra được tín hiệu mong muốn trong khi từ chối mọi thứ khác và coi đó là tiếng ồn ngẫu nhiên.
Trong CDMA, thời gian gởi một bit (bit time) lại được chia thành *m* khoảng nhỏ hơn, gọi là chip. Thông thường, có 64 hay 128 chip trên một bit, nhưng trong ví dụ phía dưới, chúng ta dùng 8 chip cho đơn giản.
Nhiều người dùng đều chia sẻ chung một băng tần, nhưng mỗi người dùng được cấp cho một mã duy nhất dài *m* bit gọi là dãy chip (chip sequence). Dãy chip này sẽ được dùng để mã hóa và giải mã dữ liệu của riêng người dùng này trong một kênh truyền chung đa người dùng. Ví dụ, sau đây là một dãy chip: (11110011). Để gởi bit 1, người dùng sẽ gởi đi dãy chip của mình. Còn để gởi đi bit 0, người dùng sẽ gởi đi phần bù của dãy chip của mình. Ví dụ với dãy chip trên, khi gởi bit 1, người dùng sẽ gởi 11110011; khi gởi bit 0 thì người dùng sẽ gởi 00001100.
Để tiện cho việc minh họa, chúng ta sẽ sử dụng các ký hiệu lưỡng cực sau: bit 0 được ký hiệu là -1, bit 1 được ký hiệu là +1.
Cũng cần phải đưa ra một định nghĩa mới: tích trong (inner product): Tích trong của hai mã S và T, ký hiệu là S•T, được tính bằng trung bình tổng của tích các bit nội tại tương ứng của hai mã này.

$$S \bullet T = \frac{1}{m}\sum_{i=1}^{m} S_i T_i$$

<u>Ví dụ</u>:

$$S = +1+1+1-1-1+1+1-1$$

$$T = +1+1+1+1-1-1+1-1$$

$$S \bullet T = \frac{+1+1+1+(-1)+1+(-1)+1+1}{8} = \frac{1}{2}$$

Bây giờ ta xem xét cách thức cấp phát chuỗi chip cho các trạm, sao cho không gây ra lẫn lộn thông tin giữa các trạm với nhau.
Định nghĩa: Hai mã S và T có cùng chiều dài m bits được gọi là trực giao khi: S•T = 0.
Ví dụ:

$$S = +1+1-1-1-1-1-1+1$$

$$T = -1-1+1-1-1-1+1+1$$

$$S \bullet T = \frac{(-1)+(-1)+(-1)+1+1+1+(-1)+1}{8} = 0$$

Nếu các người dùng trong hệ thống có các mã trực giao với nhau thì họ có thể cùng tồn tại và truyền dữ liệu một cách đồng thời với khả năng bị giao thoa dữ liệu là ít nhất.
<u>Qui ước</u>:

- Gọi D_i là bit dữ liệu mà người dùng i muốn mã hóa để truyền trên mạng.
- C_i là chuỗi chip (mã số) của người dùng i.

Sau đây là cách thức mã hóa tín hiệu để gởi lên đường truyền và giải mã để lấy dữ liệu đó ra:

- *Tín hiệu được mã của người dùng i:*
  $Z_i = D_i \times C_i$

- *Tín hiệu tổng hợp được gởi trên đường truyền:*

  $$Z = \sum_{i=1}^{n} Z_i$$

  với n là tổng số người dùng gởi tín hiệu lên đường truyền tại cùng thời điểm
- *Giải mã:*

  Dữ liệu mà người dùng i lấy về từ tín hiệu tổng hợp chung:

  $$D_i = Z \bullet C_i$$

  Nếu D_i > "ngưỡng", coi nó là 1, ngược lại coi nó là -1

Ví dụ:

Hệ thống có 4 người dùng A, B, C, D. Các mã số tương ứng của họ như sau:

A: 0 0 0 1 1 0 1 1
B: 0 0 1 0 1 1 1 0
C: 0 1 0 1 1 1 0 0
D: 0 1 0 0 0 0 1 0

Nếu ký hiệu theo kiểu lưỡng cực thì:

A: (–1 –1 –1 +1 +1 –1 +1 +1)
B: (–1 –1 +1 –1 +1 +1 +1 –1)
C: (–1 +1 –1 +1 +1 +1 –1 –1)
D: (–1 +1 –1 –1 –1 –1 +1 –1)

Để ý các mã số A, B, C, D là trực giao!

Có sáu ví dụ:

1) Chỉ có người dùng C gởi bit 1:
2) B gởi bit 1, C gởi bit 1
3) A gởi bit 1, B gởi bit 0
4) A, C đều gởi bit 1, B gởi bit 0
5) A, B, C, D đều gởi bit 1
6) A, B, D gởi bit 1, C gởi bit 0

Ta tính toán được các mã tổng hợp gởi lên đường truyền như sau:

1) – – 1 – C $Z = (-1\ +1\ -1\ +1\ +1\ +1\ -1\ -1)$
2) – 1 1 – B + C $Z = (-2\ \ 0\ \ 0\ \ 0\ +2\ +2\ \ 0\ -2)$
3) 1 0 – – A + $\overline{B}$ $Z = (\ 0\ \ 0\ -2\ +2\ \ 0\ -2\ \ 0\ +2)$
4) 1 0 1 – A + $\overline{B}$ + C $Z = (-1\ +1\ -3\ +3\ +1\ -1\ -1\ +1)$
5) 1 1 1 1 A + B + C + D $Z = (-4\ \ 0\ -2\ \ 0\ +2\ \ 0\ +2\ -2)$
6) 1 1 0 1 A + B + $\overline{C}$ + D $Z = (-2\ -2\ \ 0\ -2\ \ 0\ -2\ +4\ \ 0)$

Bây giờ, ta tính được dữ liệu nguyên thủy của người dùng ở trạm C, sau khi đã rút trích ra từ mã tổng hợp như sau:

1) $Z \bullet C = (1 + 1 + 1 + 1 + 1 + 1 + 1 + 1)/8 = 1$
2) $Z \bullet C = (2 + 0 + 0 + 0 + 2 + 2 + 0 + 2)/8 = 1$
3) $Z \bullet C = (0 + 0 + 2 + 2 + 0 - 2 + 0 - 2)/8 = 0$
4) $Z \bullet C = (1 + 1 + 3 + 3 + 1 - 1 + 1 - 1)/8 = 1$
5) $Z \bullet C = (4 + 0 + 2 + 0 + 2 + 0 - 2 + 2)/8 = 1$
6) $Z \bullet C = (2 - 2 + 0 - 2 + 0 - 2 - 4 + 0)/8 = -1$

Nhận xét:

- Đầu tiên, chúng ta phải giả sử rằng tất cả các dãy chip được đồng bộ hóa để được gởi nhận cùng thời điểm. Nhưng trong thực tế, kiểu đồng bộ hóa như vậy là không thể có được. Những gì người ta có thể làm được để đồng bộ hóa là: người gởi và người nhận đồng bộ hóa với nhau bằng cách cho người gởi gởi một dãy chip được định nghĩa trước, dãy này phải đủ dài để cho bên nhận có thể theo kịp bên gởi. Tất cả các cuộc truyền nhận khác được xem như là nhiễu ngẫu nhiên. Người ta chứng minh được rằng, chuỗi chip càng dài thì xác suất phát hiện ra chuỗi này một cách chính xác là càng cao với sự hiện diện của nhiễu.
- Cũng cần phải giả thiết rằng: bên nhận biết chính xác bên gởi là ai. Tuy trong thực tế, cần phải trung thực mà nói rằng: đặt giả thiết thì dễ hơn là làm. Nhưng hãy tin tưởng là CDMA có nhiều chi tiết phức tạp hơn và thông minh hơn để làm được chuyện đó.

5.3.2 Phương pháp truy cập đường truyền ngẫu nhiên (Random Access)

Trong phương pháp này, người ta để cho các trạm tự do tranh chấp đường truyền chung để truyền từng khung dữ liệu một. Nếu một trạm cần gởi một khung, nó sẽ gởi khung đó trên toàn bộ dải thông của kênh truyền. Sẽ không có sự phối hợp trình tự giữa các trạm. Nếu có hơn hai trạm phát cùng một lúc, "đụng độ" (collision) sẽ xảy ra, các khung bị đụng độ sẽ bị hư hại.
Giao thức truy cập đường truyền ngẫu nhiên được dùng để xác định:

- Làm thế nào để phát hiện đụng độ.
- Làm thế nào để phục hồi sau đụng độ.

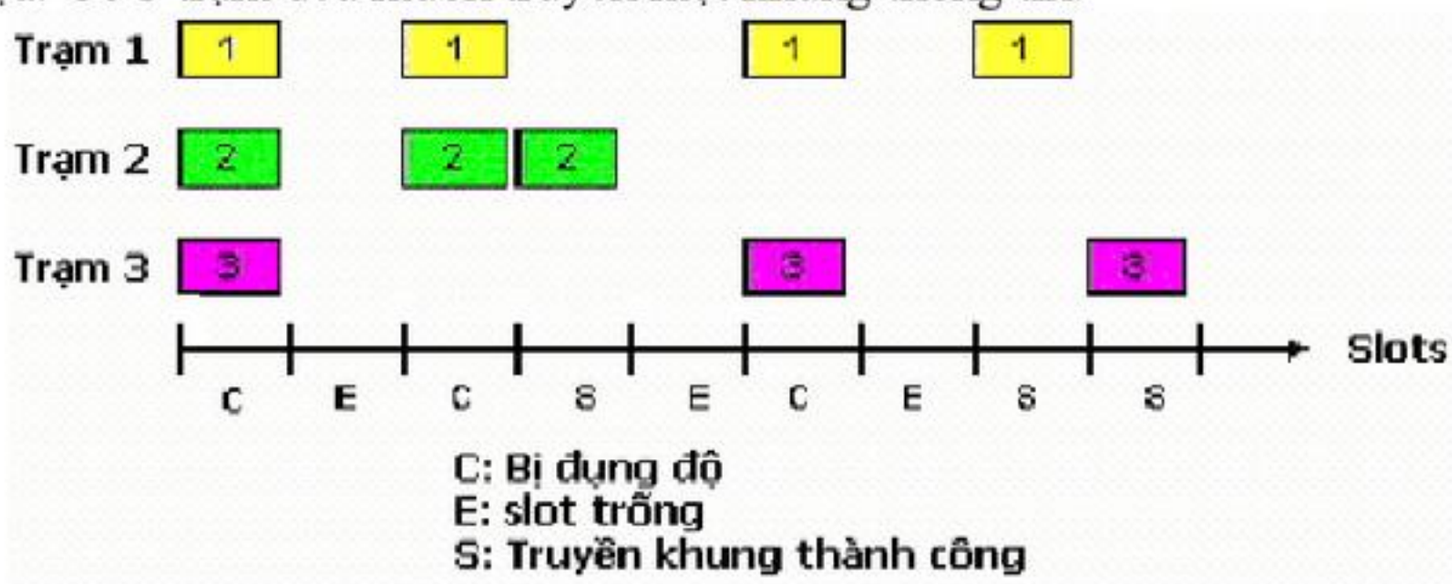
Ví dụ về các giao thức truy cập ngẫu nhiên: slotted ALOHA và pure ALOHA, CSMA và CSMA/CD, CSMA/CA.

5.3.2.1 ALOHA

Vào những năm 1970, Norman Abramson cùng các đồng sự tại Đại học Hawaii đã phát minh ra một phương pháp mới ưu hạng dùng để giải quyết bài toán về cấp phát kênh truyền. Sau đó công việc của họ tiếp tục được mở rộng bởi nhiều nhà nghiên cứu khác. Mặc dù công trình của Abramson, được gọi là hệ thống ALOHA, sử dụng hệ thống truyền quảng bá trên sóng radio mặt đất, nhưng ý tưởng cơ sở của nó có thể áp dụng cho bất kỳ hệ thống nào trong đó những người dùng không có phối hợp với nhau sẽ tranh chấp sử dụng đường truyền chung duy nhất.
Ở đây, chúng ta sẽ thảo luận về hai phiên bản của ALOHA: pure (thuần túy) và slotted (được chia khe).

5.3.2.1.1 Slotted ALOHA

Thời gian được chia thành nhiều slot có kích cỡ bằng nhau (bằng thời gian truyền một khung). Một trạm muốn truyền một khung thì phải đợi đến đầu slot thời gian kế tiếp mới được truyền. Dĩ nhiên là sẽ xảy ra đụng độ và khung bị đụng độ sẽ bị hư. Tuy nhiên, dựa trên tính phản hồi của việc truyền quảng bá, trạm phát luôn có thể theo dõi xem khung của nó phát đi có bị hủy hoại hay không bằng cách lắng nghe kênh truyền. Những trạm khác cũng làm theo cách tương tự. Trong trường hợp vì lý do nào đó mà trạm không thể dùng cơ chế lắng nghe đường truyền, hệ thống cần yêu cầu bên nhận trả lời một khung báo nhận (acknowledgement) cho bên phát. Nếu phát sinh đụng độ, trạm phát sẽ gởi lại khung tại đầu slot kế tiếp với xác suất ***p*** cho đến khi thành công.
Ví dụ minh họa: Có 3 trạm đều muốn truyền một khung thông tin.



H5.7 Minh họa giao thức Slotted ALOHA

Do sẽ có đụng độ mà mất khung thông tin, một câu hỏi đặt ra là: đâu là tỉ suất truyền khung thành công của các trạm trong mạng?
Giả sử có N trạm muốn truyền dữ liệu, mỗi trạm truyền khung thông tin của mình trong một slot với xác suất ***p***. Xác suất để một trạm trong N trạm truyền thành công S(p) được tính như sau:

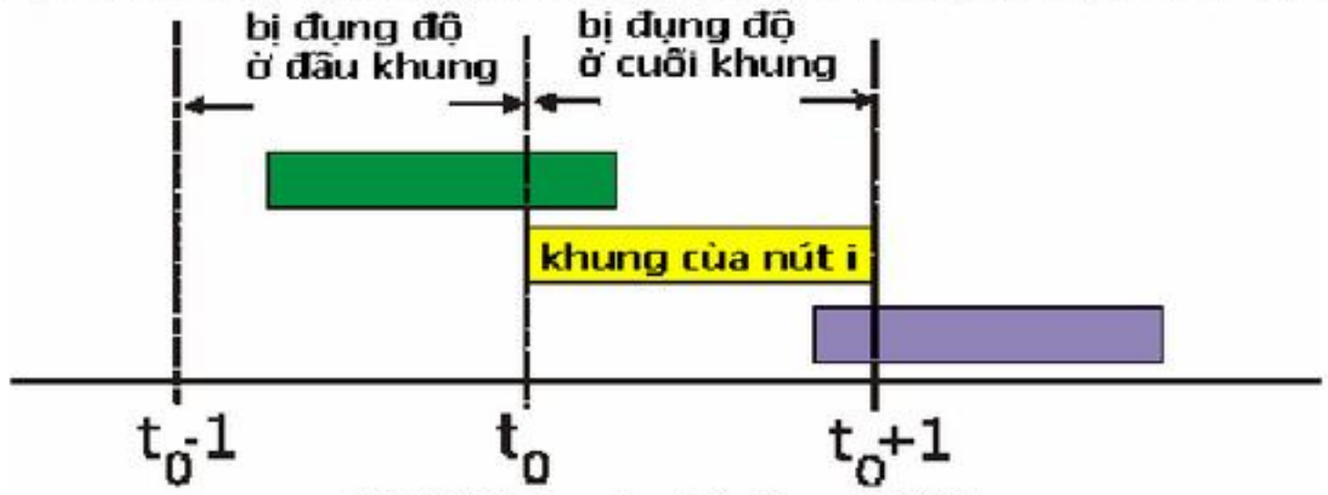
$$S(p) = Np(1-p)^{N-1}$$

Khi $p = 1/N$, S(p) đạt giá trị cực đại $(1-1/N)^{N-1}$

5.3.2.1.2 Pure ALOHA

Kỹ thuật Pure ALOHA đơn giản hơn Slotted ALOHA do không có sự đồng bộ hóa giữa các trạm. Mỗi khi muốn truyền một khung thông tin, trạm sẽ truyền nó ngay mà không cần đợi đến đầu của

slot thời gian kế tiếp. Vì thế xác xuất bị đụng độ tăng thêm! Nghĩa là khung thông tin được gởi tại thời điểm t_0 sẽ đụng độ với những khung được gởi trong khoảng thời gian $[t_0-1, t_0+1]$.



H5.8 Minh họa giao thức Pure ALOHA

Gọi P là xác xuất của một sự kiện nào đó, ta có những phân tích sau:

P(nút i truyền thành công) = P(để nút i truyền)

× P(không có nút nào khác truyền trong khoảng $[t_0-1,t_0]$) ·

× P(không có nút nào khác truyền trong khoảng $[t_0, t_0+1]$)

$$= p(1-p)^{N-1}(1-p)^{N-1}$$

S(p) = P(một nút bất kỳ trong N nút truyền thành công) = $Np(1-p)^{N-1}(1-p)^{N-1}$

Những phân tích vừa nêu giả sử rằng luôn có thường trực N trạm trong mạng. Và trong trường hợp tối ưu, mỗi trạm thử truyền vơi xác suất 1/N.

Trong thực tế, số lượng các trạm thường trực trong mạng luôn thay đổi. Giả sử chúng ta có tổng cộng m trạm làm việc. n trạm là thường trực trên mạng, mỗi trạm thường trực trên mạng sẽ cố gởi khung thông tin với xác suất cố định p. m-n trạm còn lại là không thường trực, và chúng có thể gởi khung thông tin với xác suất p_a, với p_a có thể nhỏ hơn p.

Nhận xét chung về ALOHA:

- Hiệu năng thấp do không có thăm dò đường truyền trước khi gởi khung, dẫn đến việc mất nhiều thời gian cho việc phát hiện đụng độ và phục hồi sau đụng độ.
- Hoạt động theo kiểu ALOHA có khả năng dẫn đến việc hệ thống bị "chết đứng" do mọi nỗ lực gởi gói tin của tất cả các trạm đều bị đụng độ.

Slotted ALOHA trở nên quan trọng với lý do không mấy rành mạch lắm. Nó ra đời vào những năm 1970, được sử dụng trong một số hệ thống thí nghiệm thời đó, và rồi hầu như bị lãng quên. Và khi công nghệ truy cập Internet không qua cable được phát minh, đột nhiên lại phát sinh vấn đề làm sao để cấp phát đường truyền được chia sẻ cho nhiều người dùng cạnh tranh, Slotted ALOHA lại được lôi ra từ thùng rác để cứu rỗi cuộc đời. Vẫn thường có chuyện là các giao thức hợp lý một cách hoàn hảo lại không được sử dụng vì những lý do về chính trị, nhưng nhiều năm sau đó, một số người thông thái lại nhận ra rằng những giao thức bỏ đi từ lâu rồi đó lại có thể giúp họ giải quyết được vấn đề. Với lý do như vậy, trong chương này, chúng ta sẽ nghiên cứu một số giao thức ưu hạng hiện tại không được sử dụng rộng rãi lắm nhưng biết đâu có thể được sử dụng dễ dàng trong các ứng dụng ở tương lai. Dĩ nhiên là chúng ta cũng sẽ nghiên cứu nhiều giao thức đang được sử dụng rộng rãi hiện nay.

5.3.2.2 CSMA – Carrier Sense Multiple Access

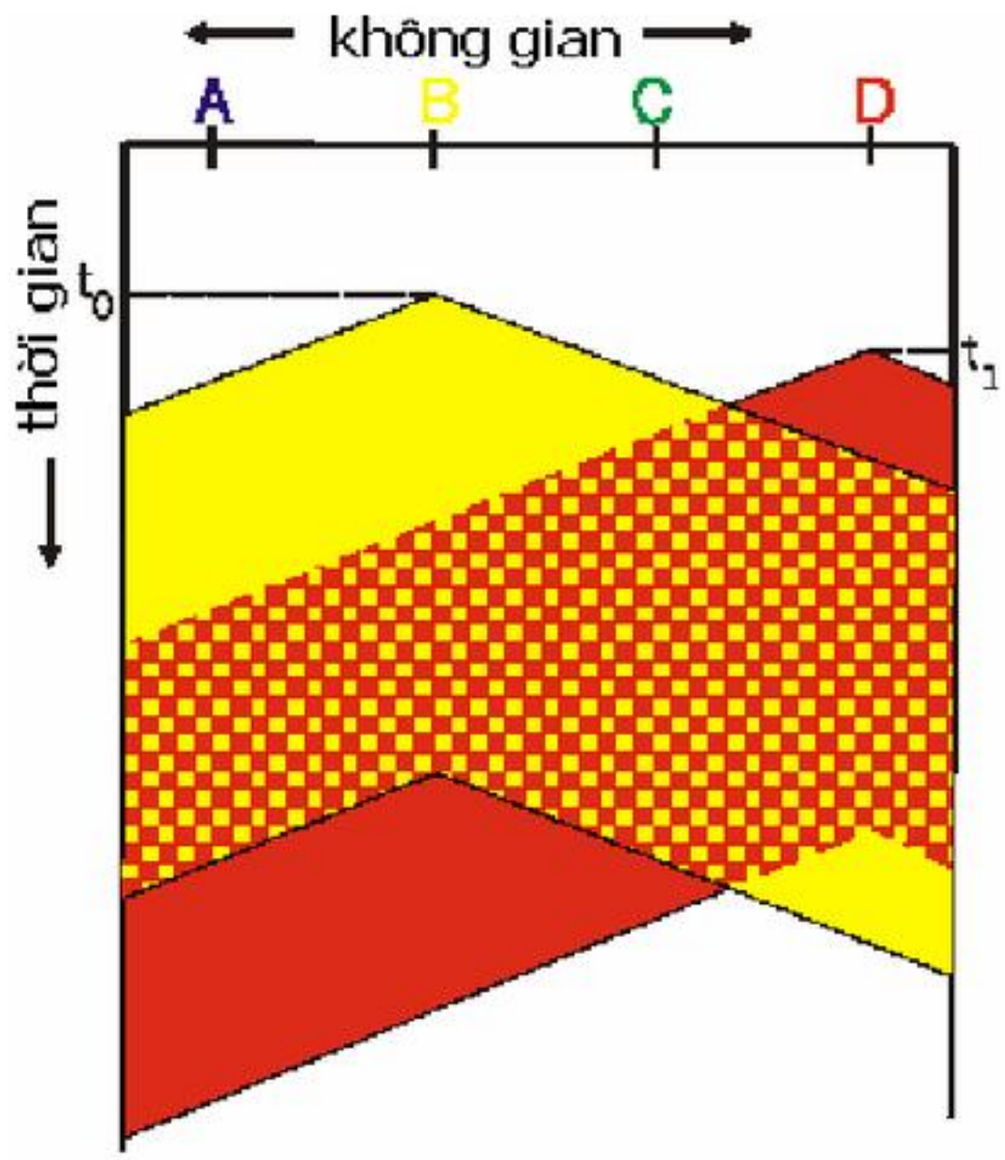
Giao thức ALOHA mặc dù đã chạy được, nhưng một điều đáng ngạc nhiên là người ta lại để cho các trạm làm việc tự do gởi thông tin lên đường truyền mà chẳng cần quan tâm đến việc tìm hiểu xem những trạm khác đang làm gì. Và điều đó dẫn đến rất nhiều vụ đụng độ tín hiệu. Tuy nhiên, trong mạng LAN, người ta có thể thiết kế các trạm làm việc sao cho chúng có thể điều tra xem các trạm khác đang làm gì và tự điều chỉnh hành vi của mình một cách tương ứng. Làm như vậy sẽ giúp cho hiệu năng mạng đạt được cao hơn. CSMA là một giao thức như vậy!

Các giao thức mà trong đó các trạm làm việc lắng nghe đường truyền trước khi đưa ra quyết định mình phải làm gì tương ứng với trạng thái đường truyền đó được gọi là các giao thức có "cảm nhận" đường truyền (carrier sense protocol). Cách thức hoạt động của CSMA như sau: lắng nghe

kênh truyền, nếu thấy kênh truyền rỗi thì bắt đầu truyền khung, nếu thấy đường truyền bận thì trì hoãn lại việc gởi khung.
Thế nhưng trì hoãn việc gởi khung cho đến khi nào?
Có ba giải pháp:

- Theo dõi không kiên trì (Non-persistent CSMA): Nếu đường truyền bận, đợi trong một khoảng thời gian ngẫu nhiên rồi tiếp tục nghe lại đường truyền.
- Theo dõi kiên trì (persistent CSMA): Nếu đường truyền bận, tiếp tục nghe đến khi đường truyền rỗi rồi thì truyền gói tin với xác suất bằng 1.
- Theo dõi kiên trì với xác suất p (P-persistent CSMA): Nếu đường truyền bận, tiếp tục nghe đến khi đường truyền rỗi rồi thì truyền gói tin với xác suất bằng p.

Dễ thấy rằng giao thức CSMA cho dù là theo dõi đường truyền kiên trì hay không kiên trì thì khả năng tránh đụng độ vẫn tốt hơn là ALOHA. Tuy thế, đụng độ vẫn có thể xảy ra trong CSMA!
Tình huống phát sinh như sau: khi một trạm vừa phát xong thì một trạm khác cũng phát sinh yêu cầu phát khung và bắt đầu nghe đường truyền. Nếu tín hiệu của trạm thứ nhất chưa đến trạm thứ hai, trạm thứ hai sẽ cho rằng đường truyền đang rảnh và bắt đầu phát khung. Như vậy đụng độ sẽ xảy ra.

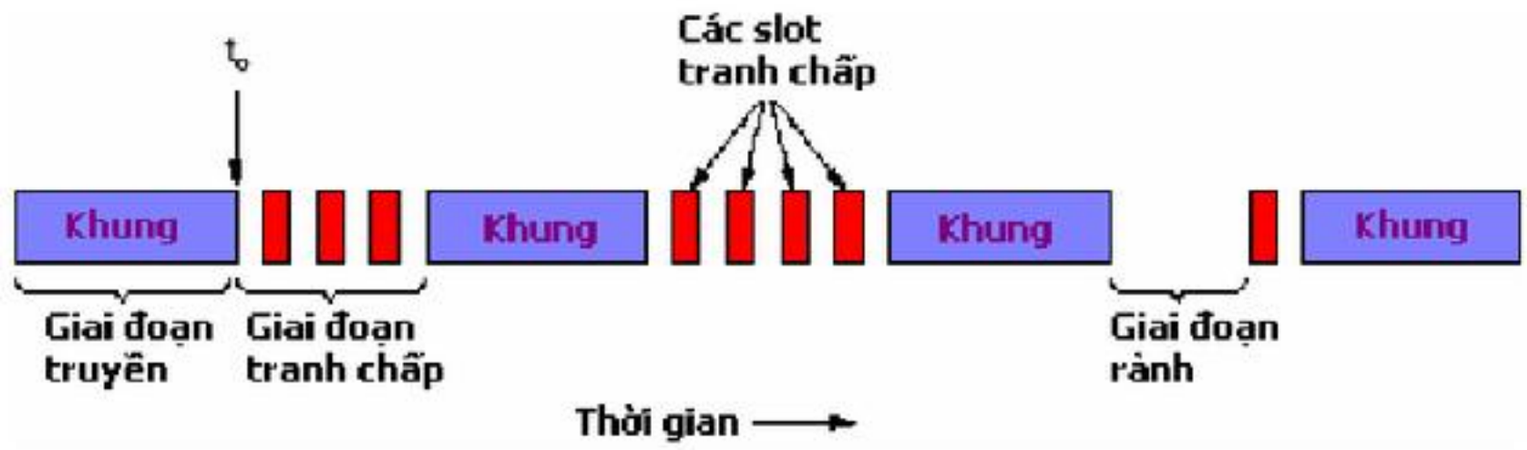


H5.9 Mô tả không gian và thời gian diễn ra đụng độ

Hậu quả của đụng độ là: khung bị mất và toàn bộ thời gian từ lúc đụng độ xảy ra cho đến khi phát xong khung là lãng phí!
Bây giờ phát sinh vấn đề mới: các trạm có quan tâm theo dõi xem có đụng độ xảy ra không và khi đụng độ xảy ra thì các trạm sẽ làm gì?

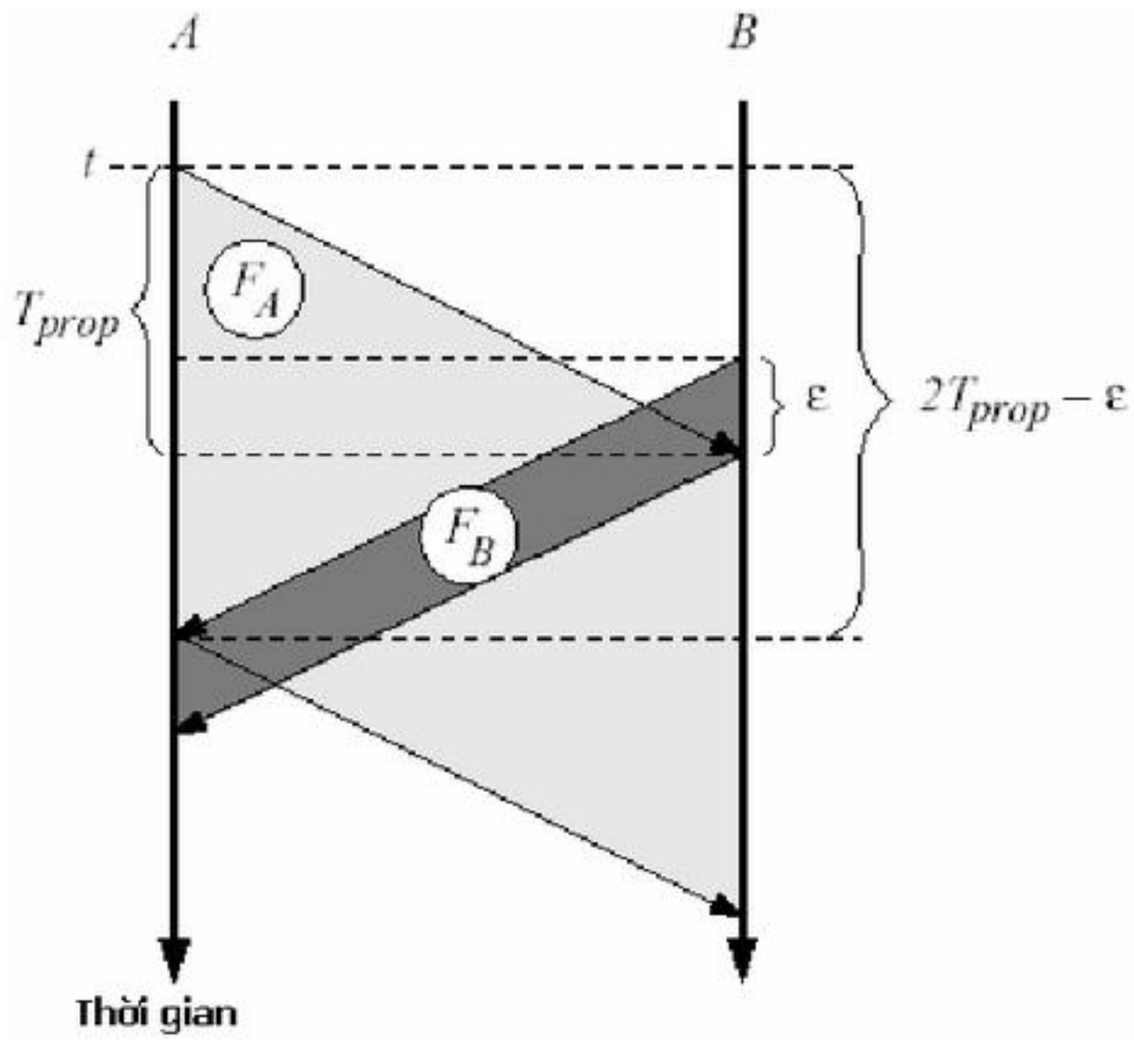
5.3.2.2.1 CSMA với cơ chế theo dõi đụng độ (CSMA/CD – CSMA with Collision Detection)

CSMA/CD về cơ bản là giống như CSMA: lắng nghe trước khi truyền. Tuy nhiên CSMA/CD có hai cải tiến quan trọng là: phát hiện đụng độ và làm lại sau đụng độ.
Phát hiện đụng độ: Trạm vừa truyền vừa tiếp tục dò xét đường truyền. Ngay sau khi đụng độ được phát hiện thì trạm ngưng truyền, phát thêm một dãy nhồi (dãy nhồi này có tác dụng làm tăng cường thêm sự va chạm tín hiệu, giúp cho tất cả các trạm khác trong mạng thấy được sự đụng độ), và bắt đầu làm lại sau đụng độ.
CSMA/CD, cũng giống như các giao thức trong LAN khác, sử dụng mô hình quan niệm như trong hình sau:



H5.10 CSMA/CD có thể ở một trong ba trạng thái: tranh chấp, truyền, rảnh

Tại thời điểm t_0, một trạm đã phát xong khung của nó. Bất kỳ trạm nào khác có khung cần truyền bây giờ có thể cố truyền thử. Nếu hai hoặc nhiều hơn các trạm làm như vậy cùng một lúc thì sẽ xảy ra đụng độ. Đụng độ có thể được phát hiện bằng cách theo dõi năng lượng hay độ rộng của xung của tín hiệu nhận được và đem so sánh với độ rộng của xung vừa truyền đi.
Bây giờ ta đặt ra câu hỏi: Sau khi truyền xong khung (hết giai đoạn truyền), trạm sẽ bỏ ra thời gian tối đa là bao lâu để biết được là khung của nó đã bị đụng độ hoặc nó đã truyền thành công? Gọi thời gian này là "cửa sổ va chạm" và ký hiệu nó là T_w. Phân tích sau đây sẽ cho ra câu trả lời. Hình sau sẽ mô phỏng chi tiết về thời gian phát khung giữa hai trạm A và B ở hai đầu mút xa nhất trên đường truyền tải.

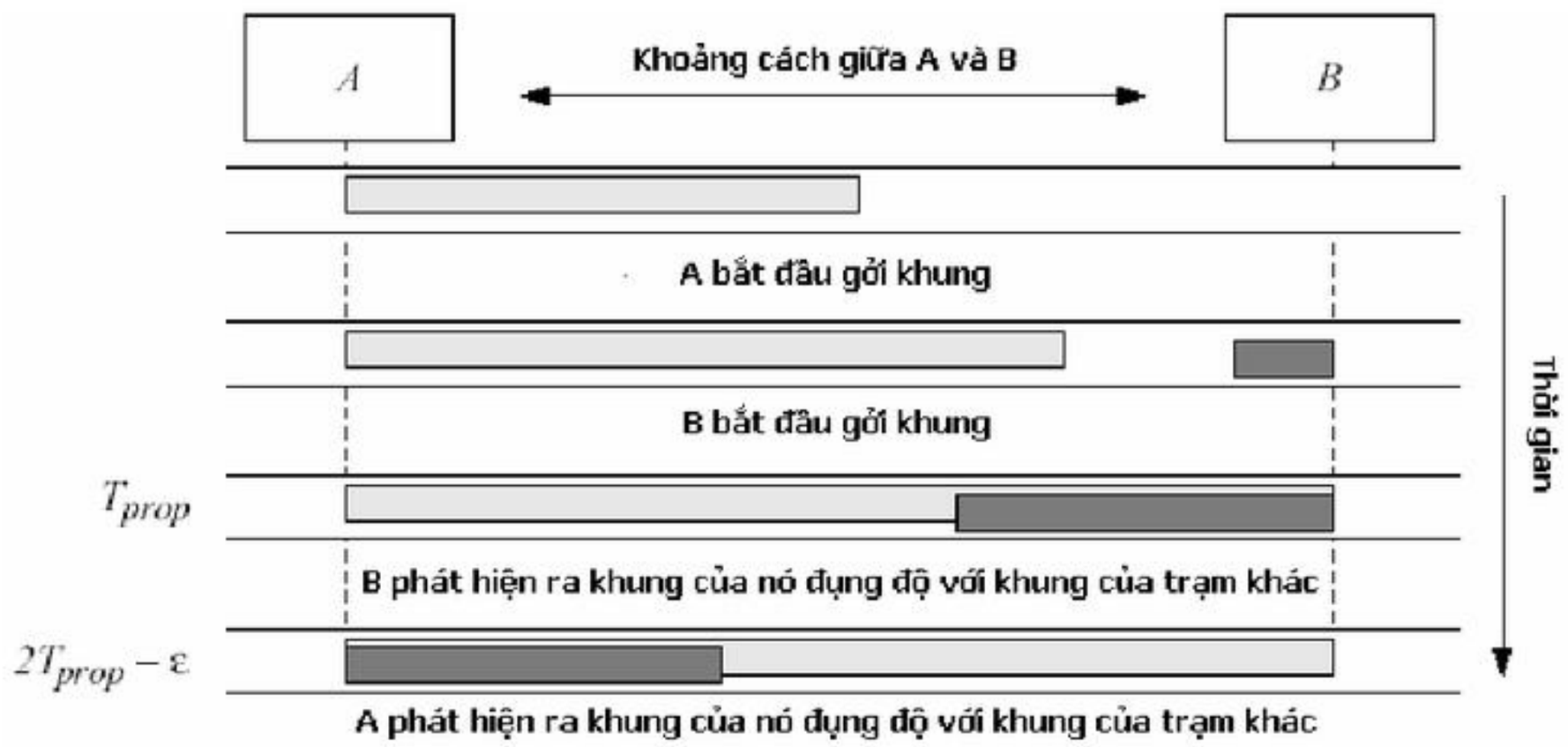


H5.11 Thời gian cần thiết để truyền một khung

Đặt T_{prop} là thời gian lan truyền tín hiệu giữa hai đầu mút xa nhau nhất trên đường truyền tải.

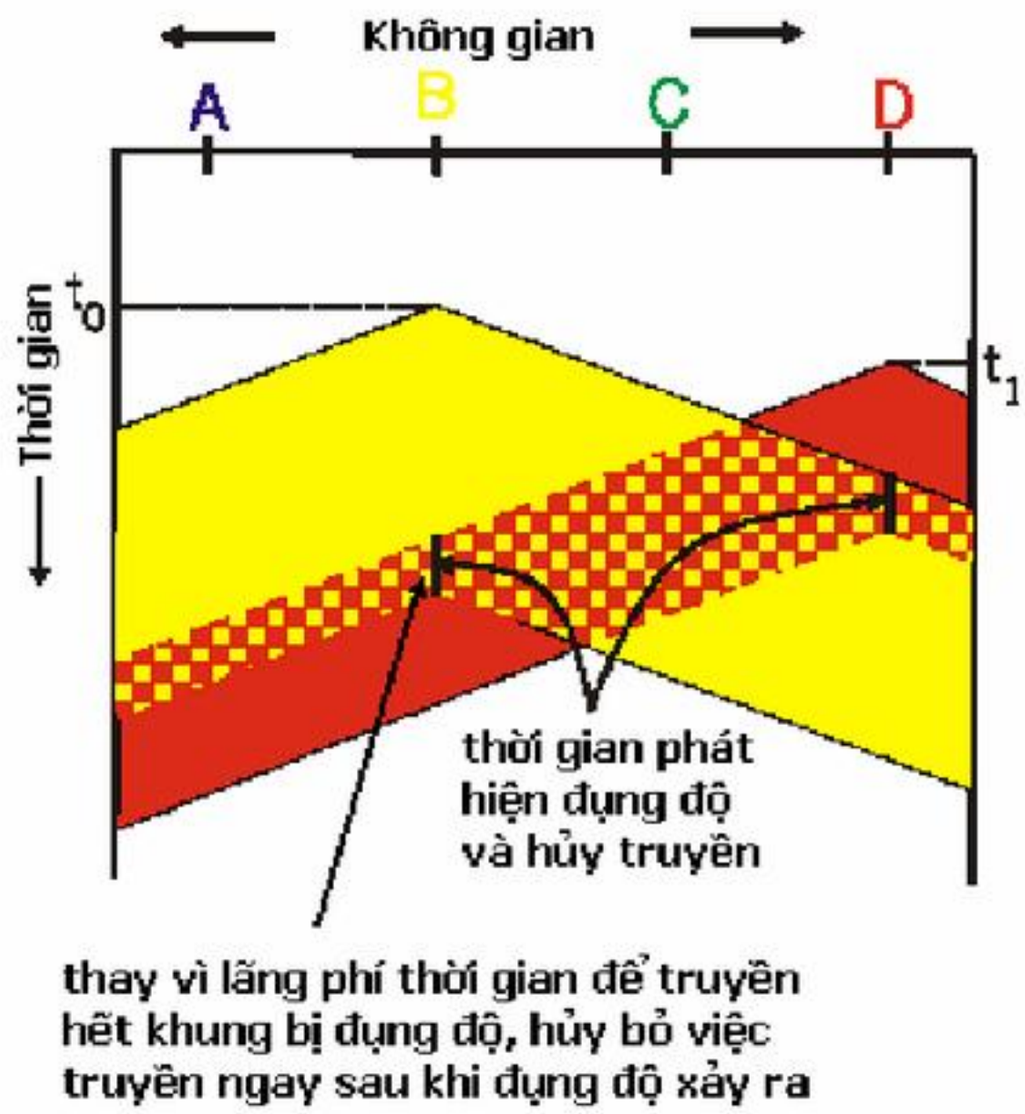
- Tại thời điểm t, A bắt đầu phát đi khung dữ liệu của nó.
- Tại $t+T_{prop}-\varepsilon$, B phát hiện kênh truyền rảnh và phát đi khung dữ liệu của nó.
- Tại $t+T_{prop}$, B phát hiện sự đụng độ.
- Tại $t+2T_{prop}-\varepsilon$, A phát hiện sự đụng độ.

Theo phân tích trên, thì $\mathbf{T_w = 2T_{prop}}$



H5.12 Phát hiện đụng độ khi truyền tin

Việc hủy bỏ truyền khung ngay khi phát hiện có đụng độ giúp tiết kiệm thời gian và băng thông, vì nếu cứ tiếp tục truyền khung đi nữa, khung đó vẫn hư và vẫn phải bị hủy bỏ.



H5.13 Xử lý khu đụng độ

<u>Làm lại sau khi đụng độ</u>: Sau khi bị đụng độ, trạm sẽ chạy một thuật toán gọi là back-off dùng để tính toán lại lượng thời gian nó phải chờ trước khi gởi lại khung. Lượng thời gian này phải là ngẫu nhiên để các trạm sau khi quay lại không bị đụng độ với nhau nữa.

Thuật toán back-off hoạt động như sau:

- Rút ngẫu nhiên ra một con số nguyên M thõa: $0 \le M \le 2^k$. Trong đó $k = \min(n,10)$, với n là tổng số lần đụng độ mà trạm đã gánh chịu.
- Kỳ hạn mà trạm phải chờ trước khi thử lại một lần truyền mới là $M*T_w$.
- Khi mà n đạt đến giá trị 16 thì hủy bỏ việc truyền khung. (Trạm đã chịu đựng quá nhiều vụ đụng độ rồi, và không thể chịu đựng hơn được nữa!)

ĐÁNH GIÁ HIỆU SUẤT CỦA GIAO THỨC CSMA/CD:

Gọi:

P là kích thước của khung, ví dụ như 1000 bits.

C là dung lượng của đường truyền, ví dụ như 10 Mbps.

Ta có thời gian phát hết một khung thông tin là P/C giây.

Trung bình, chúng ta sẽ thử e lần trước khi truyền thành công một khung. Vì vậy, với mỗi lần phát thành công một khung (tốn P/C giây), ta đã mất tổng cộng $2eT_{prop}$ ($\approx 5T_{prop}$) vì đụng độ.

Thành thử hiệu năng của giao thức (tỉ lệ giữa thời gian hoạt động hữu ích trên tổng thời gian hoạt động) là:

$$\frac{\frac{P}{C}}{\frac{P}{C}+5T_{prop}} = \frac{1}{1+\frac{5T_{prop}}{\frac{P}{C}}} = \frac{1}{1+5a}, \text{ với } a = \frac{T_{prop}C}{P}$$

Giá trị của a đóng vai trò rất quan trọng đến hiệu suất hoạt động của mạng kiểu CSMA/CD.

5.3.3 Phương pháp phân lượt truy cập đường truyền

Bây giờ thử nhìn lại hai phương pháp điều khiển truy cập đường truyền "chia kênh" và "truy cập ngẫu nhiên", ta sẽ thấy chúng đều có những điểm hay và hạn chế:

- Trong các giao thức dạng chia kênh, kênh truyền được phân chia một cách hiệu quả và công bằng khi tải trọng đường truyền là lớn. Tuy nhiên chúng không hiệu quả khi tải trọng của đường truyền là nhỏ: có độ trì hoãn khi truy cập kênh truyền, chỉ 1/N băng thông được cấp cho người dùng ngay cả khi chỉ có duy nhất người dùng đó hiện diện trong hệ thống.
- Các giao thức dạng truy cập ngẫu nhiên thì lại hoạt động hiệu quả khi tải trọng của đường truyền thấp. Nhưng khi tải trọng đường truyền cao thì phải tốn nhiều chi phí cho việc xử lý đụng độ.

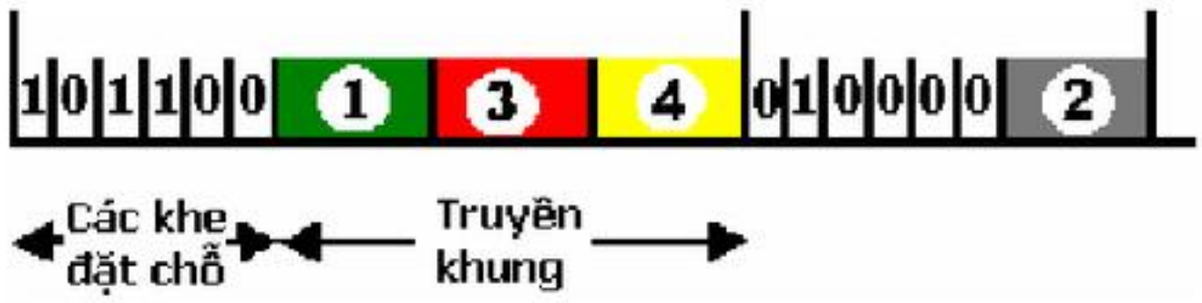
Các giao thức dạng "phân lượt" sẽ để ý đến việc tận dụng những mặt mạnh của hai dạng nói trên. Ý tưởng chính của các giao thức dạng "phân lượt" là không để cho đụng độ xảy ra bằng cách cho các trạm truy cập đường truyền một cách tuần tự.

Về cơ bản, có hai cách thức để "phân lượt" sử dụng đường truyền:

- **Thăm dò (polling)**: Trạm chủ (master) sẽ mời các trạm tớ (slave) truyền khi đến lượt. Lượt truyền được cấp phát cho trạm tớ có thể bằng cách: trạm chủ dành phần cho trạm tớ hoặc trạm tớ yêu cầu và được trạm chủ đáp ứng. Tuy nhiên có thể thấy những vấn đề sẽ gặp phải của giải pháp này là: chi phí cho việc thăm dò, độ trễ do phải chờ được phân lượt truyền, hệ thống rối loạn khi trạm chủ gặp sự cố.
- **Chuyền thẻ bài (token passing)**: Thẻ bài điều khiển sẽ được chuyển lần lượt từ trạm này qua trạm kia. Trạm nào có trong tay thẻ bài sẽ được quyền truyền, truyền xong phải chuyền thẻ bài qua trạm kế tiếp. Những vấn đề cần phải quan tâm: chi phí quản lý thẻ bài, độ trễ khi phải chờ thẻ bài, khó khăn khi thẻ bài bị mất.

5.3.3.1 Ví dụ về phương pháp thăm dò: Thăm dò phân tán (Distributed Polling)

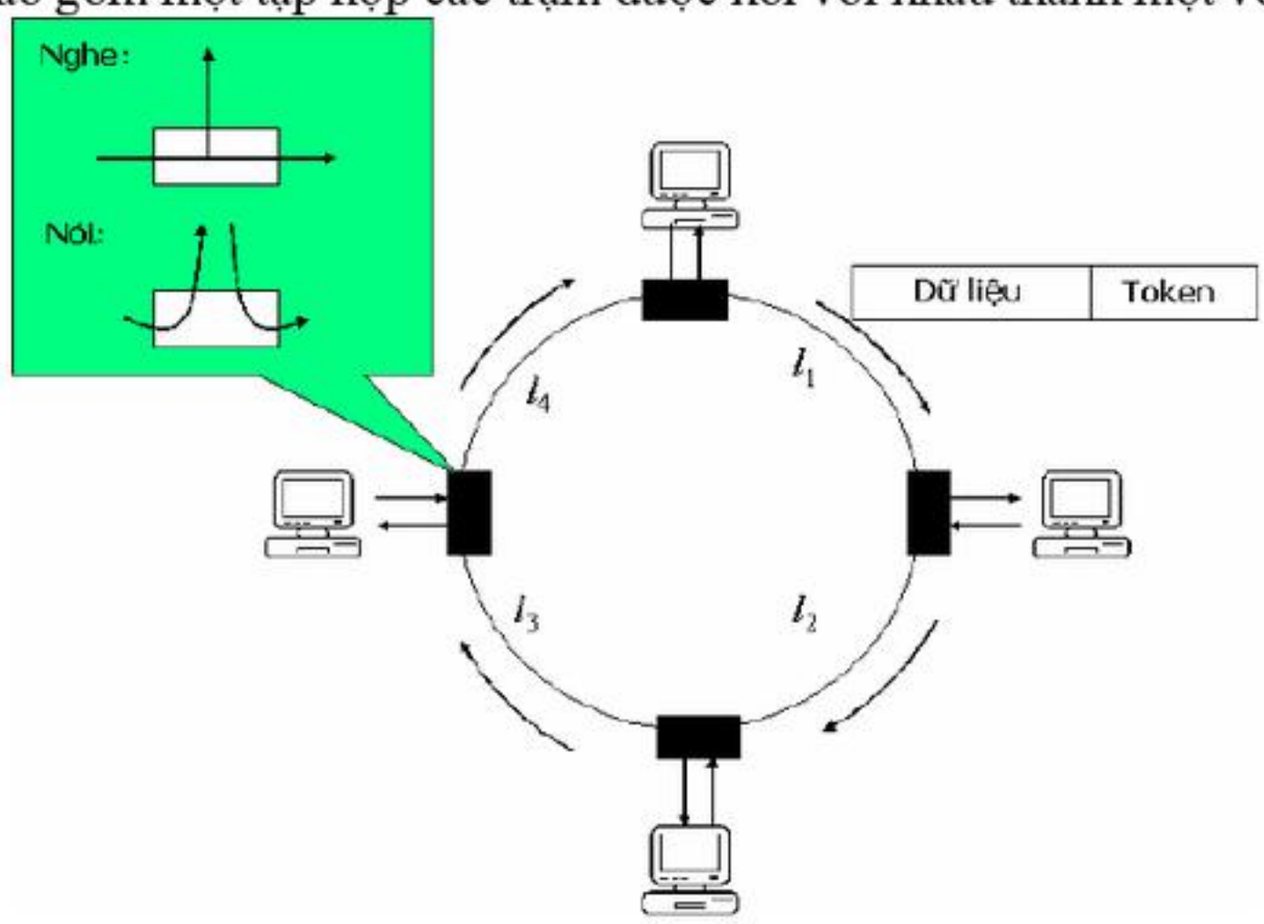
Thời gian được chia thành những "khe" (slot). Giả sử hệ thống hiện có N trạm làm việc. Một chu kỳ hoạt động của hệ thống bắt đầu bằng N khe thời gian ngắn dùng để đặt chỗ (reservation slot). Khe thời gian dùng để đặt chỗ bằng với thời gian lan truyền tín hiệu giữa hai đầu mút xa nhất trên đường truyền. Tới khe đặt chỗ thứ i, trạm thứ i nếu muốn truyền dữ liệu sẽ phát tín hiệu đặt chỗ của mình lên kênh truyền, và tín hiệu này sẽ được nhìn thấy bởi tất cả các trạm khác trong mạng. Sau thời gian đặt chỗ, các trạm bắt đầu việc truyền dữ liệu của mình theo đúng trình tự đã đăng ký.



H5.14 Mô tả các chu kỳ hoạt động của hệ thống Thăm dò phân tán

5.3.3.2 Ví dụ về phương pháp chuyển thẻ bài: Token Ring

Giao thức này sử dụng mạng kiểu hình vòng, dùng thẻ bài để cấp quyền sử dụng đường truyền. Mạng token ring bao gồm một tập hợp các trạm được nối với nhau thành một vòng.



H5.15 Mô hình hoạt động của mạng Token Ring

Dữ liệu luôn chạy theo một hướng vòng quanh vòng. Mỗi trạm nhận khung từ trạm phía trên của nó và rồi chuyển khung đến trạm phía dưới. Thẻ bài là công cụ để quyết định ai có quyền truyền tại một thời điểm.

Cách thức hoạt động của mạng token ring như sau: một thẻ bài, thực chất chỉ là một dãy bit, sẽ chạy vòng quanh vòng; mỗi nút sẽ nhận thẻ bài rồi lại chuyển tiếp thẻ bài này đi. Khi một trạm có khung cần truyền và đúng lúc nó thấy có thẻ bài tới, nó liền lấy thẻ bài này ra khỏi vòng (nghĩa là không có chuyển tiếp chuỗi bit đặc biệt này lên vòng nữa), và thay vào đó, nó sẽ truyền khung dữ liệu của mình đi. Khi khung dữ liệu đi một vòng và quay lại, trạm phát sẽ rút khung của mình ra và chèn lại thẻ bài vào vòng. Hoạt động cứ xoay vòng như thế.

Card mạng dùng cho token ring sẽ có trên đó một bộ nhận, một bộ phát và một bộ đệm dùng chứa dữ liệu. Khi không có trạm nào trong vòng có dữ liệu để truyền, thẻ bài sẽ lưu chuyển vòng quanh. Nếu một trạm có dữ liệu cần truyền và có thẻ bài, nó có quyền truyền một hoặc nhiều khung dữ liệu tùy theo qui định của hệ thống.

Mỗi khung dữ liệu được phát đi sẽ có một phần thông tin chứa địa chỉ đích của trạm bên nhận; ngoài ra nó còn có thể chứa địa chỉ muticast hoặc broadcast tùy theo việc bên gởi muốn gởi khung cho một nhóm người nhận hay tất cả mọi người trong vòng. Khi khung thông tin chạy qua mỗi trạm trong vòng, trạm này sẽ nhìn vào địa chỉ đích trong khung đó để biết xem có phải nó là đích đến của khung không. Nếu phải, trạm sẽ chép nội dung của khung vào trong bộ đệm của nó, chỉ chép thôi chứ không được xóa khung ra khỏi vòng.

Một vấn đề cần phải quan tâm đến là một trạm đang giữ thẻ bài thì nó có quyền truyền bao nhiêu dữ liệu, hay nói cách khác là trạm được cho bao nhiêu thời gian để truyền dữ liệu? Chúng ta gọi thời gian này là thời gian giữ thẻ bài – THT (Token Holding Time). Trong trường hợp trong vòng chỉ có một trạm cần truyền dữ liệu và các trạm khác không có nhu cầu truyền, thì ta có thể cấp THT cho trạm có nhu cầu càng lâu càng tốt. Điều này sẽ làm tăng hiệu suất sử dụng hệ thống một cách đáng kể. Bởi vì sẽ thật là ngớ ngẩn nếu bắt trạm ngừng, chờ thẻ bài chạy hết một vòng, rồi lại

truyền tiếp. Tuy nhiên, giải pháp trên sẽ không hoạt động tốt nếu có nhiều trạm trong vòng cần gởi dữ liệu. THT dài chỉ thích hợp cho những trạm cần truyền nhiều dữ liệu, nhưng lại không phù hợp với những trạm chỉ có ít thông điệp cần gởi đi ngay cả khi thông điệp này là tối quan trọng. Điều này cũng giống như tình huống mà bạn xếp hàng để sử dụng máy ATM ngay sau một anh chàng định rút ra 10 triệu đồng, trong khi bạn chỉ cần vào đấy để kiểm tra tài khoản của mình còn bao nhiêu tiền! Trong các mạng 802.5, THT mặc định là 10 ms.

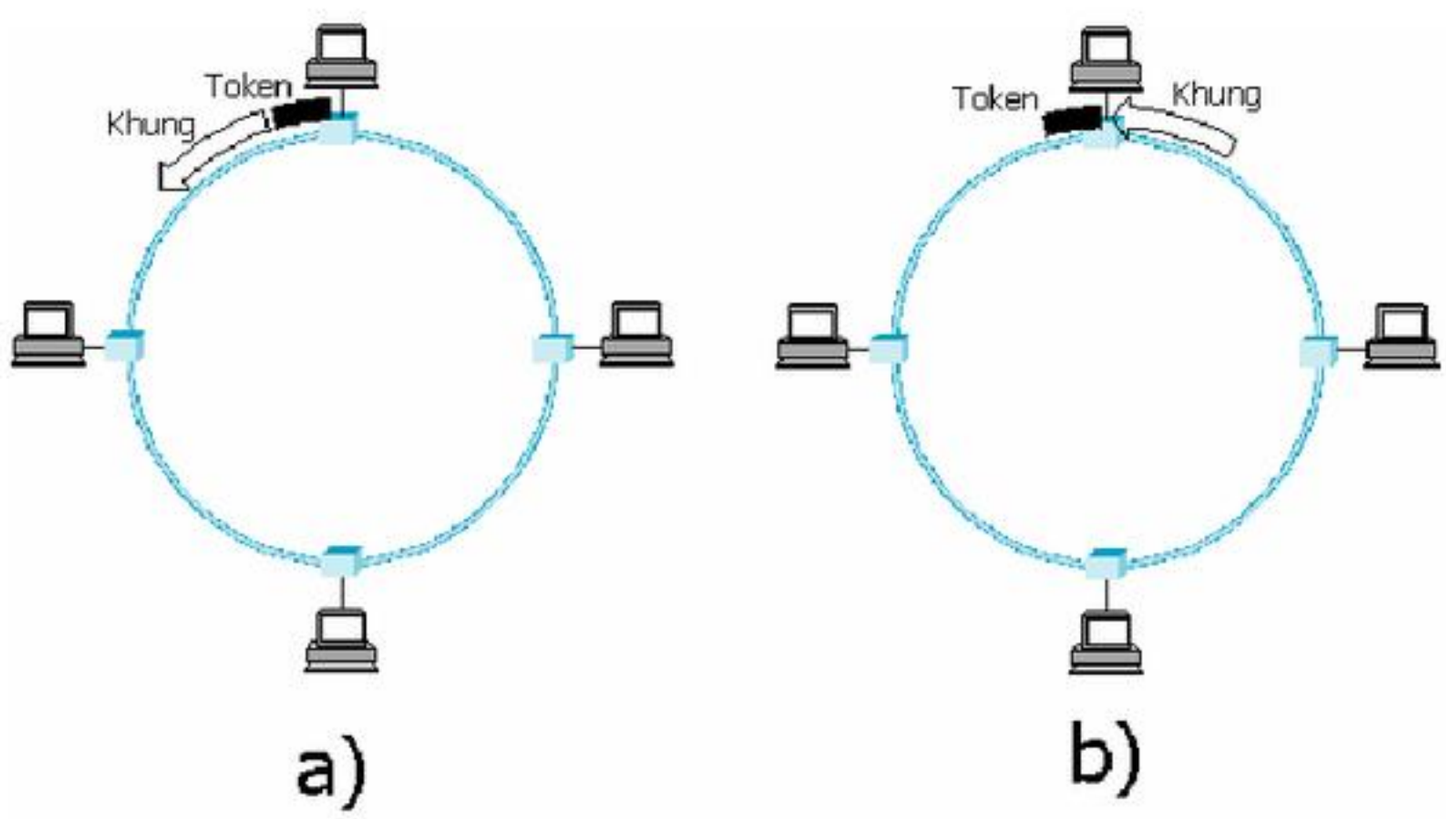
Từ thời gian giữ thẻ bài, chúng ta lại nghĩ ra một số đo quan trọng khác: Thời gian xoay vòng của thẻ bài – TRT (Token rotation time), nghĩa là lượng thời gian bỏ ra để thẻ bài đi hết đúng một vòng. Dễ nhận thấy rằng:

$$TRT \leq \text{Số nút hoạt động} \times THT + \text{Độ trễ của vòng}$$

Với "Độ trễ của vòng" là tổng thời gian để thẻ bài đi hết một vòng khi trong vòng không có trạm nào cần truyền dữ liệu, "Số nút hoạt động" ám chỉ số trạm có dữ liệu cần truyền.

Giao thức 802.5 cung cấp một phương thức truyền dữ liệu tin cậy bằng cách sử dụng hai bit A và C ở đuôi của khung dữ liệu. Hai bit bày ban đầu nhận giá trị 0. Khi một trạm nhận ra nó là đích đến của một khung dữ liệu, nó sẽ đặt bit A trong khung này lên. Khi trạm chép khung vào trong bộ nhớ đệm của nó, nó sẽ đặt bit C lên. Khi trạm gởi thấy khung của nó quay lại với bit A vẫn là 0, nó biết là trạm đích bị hư hỏng hoặc không có mặt. Nếu bit A là 1, nhưng bit C là 0, điều này ám chỉ trạm đích có mặt nhưng vì lý do nào đó trạm đích không thể nhận khung (ví dụ như thiếu bộ đệm chẳng hạn). Vì thế khung này có thể sẽ được truyền lại sau đó với hy vọng là trạm đích có thể tiếp nhận nó.

Chi tiết cuối cùng cần phải xem xét là: chính xác khi nào thì trạm sẽ nhả thẻ bài ra? Có hai đề nghị: a) nhả thẻ bài ra ngay sau khi trạm vừa truyền khung xong (RAT); b) nhả thẻ bài ra ngay sau khi trạm nhận lại khung vừa phát ra (RAR).



H5.16 Nhả token: a)RAT b)RAR

Quản lý hoạt động của mạng Token Ring

Cần thiết phải đề cử ra một trạm làm nhiệm vụ quản lý mạng token ring gọi là monitor. Công việc của monitor là đảm bảo sức khỏe cho toàn bộ vòng. Bất kỳ trạm nào cũng có thể trở thành monitor. Thủ tục bầu chọn monitor diễn ra khi vòng vừa được tạo ra hoặc khi monitor của vòng bị sự cố. Một monitor mạnh khỏe sẽ định kỳ thông báo sự hiện diện của nó cho toàn vòng biết bằng một thông điệp đặc biệt. Nếu một trạm không nhận được thông báo hiện diện của monitor trong một khoảng thời gian nào đó, nó sẽ coi như monitor bị hỏng và sẽ cố trở thành monitor mới.

Khi một trạm quyết định rằng cần phải có một monitor mới, nó sẽ gởi một thông điệp thỉnh cầu, thông báo ý định trở thành monitor của mình. Nếu thông điệp này chạy một vòng và về lại được trạm, trạm sẽ cho rằng mọi người đồng ý vị trí monitor của nó. Còn nếu đồng thời có nhiều trạm cùng gởi thông điệp thỉnh cầu, chúng sẽ phải áp dụng một luật lựa chọn nào đó, chẳng hạn như "ai có địa chỉ cao nhất sẽ thắng cử".

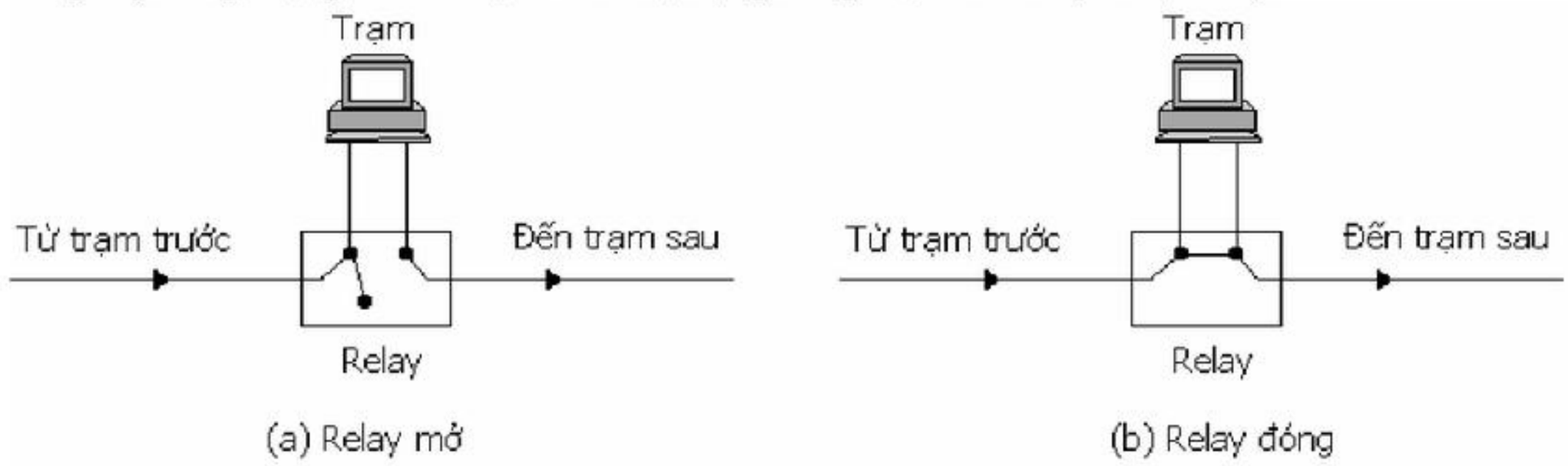
Nhiệm vụ đáng chú ý của monitor là phải đảm bảo rằng luôn luôn có sự hiện diện của thẻ bài ở đâu đó trên vòng, có thể là đang di chuyển hay đang bị giữ bởi một trạm nào đó. Rõ ràng là thẻ bài có thể bị biến mất vì lý do nào đó chẳng hạn như lỗi bit, trạm đang giữ nó bị hư hỏng. Để phát hiện ra việc thẻ bài bị mất, khi thẻ bài chạy ngang qua monitor, nó sẽ bật một bộ đếm thời gian để tính giờ. Bộ đếm này có giá trị tối đa là:

Số lượng trạm × THT + Độ trễ của vòng

Trong đó "Số lượng trạm" là số các trạm làm việc đang hiện diện trên vòng, "độ trễ của vòng" là tổng thời gian lan truyền tín hiệu trên vòng. Nếu bộ đếm đạt đến giá trị tối đa mà monitor vẫn không thấy thẻ bài chạy qua nó nữa thì nó sẽ tạo ra thẻ bài mới.

Monitor cũng phải kiểm tra xem có khung nào bị hỏng hoặc vô thừa nhận hay không. Một khung nếu có lỗi checksum hoặc khuôn dạng không hợp lệ sẽ chạy một cách vô định trên vòng. Monitor sẽ thu khung này lại trước khi chèn lại thẻ bài vào vòng. Một khung vô thừa nhận là khung mà đã được chèn thành công vào vòng, nhưng cha của nó bị chết, nghĩa là trạm gởi nó chỉ gởi nó lên vòng, nhưng chưa kịp thu nó lại thì đã bị chết (down). Những khung như vậy sẽ bị phát hiện bằng cách thêm vào một bit điều khiển gọi là monitor bit. Khi được phát lần đầu tiên, monitor bit trên khung sẽ nhận giá trị 0. Khi khung đi ngang qua monitor, monitor sẽ đặt monitor bit lên 1. Nếu monitor thấy khung này lại chạy qua nó với monitor bit là 1, nó sẽ rút khung này ra khỏi vòng.

Một chức năng quản lý vòng khác là phát hiện ra một trạm bị chết. Nếu một trạm trong vòng bị chết, nó sẽ làm đứt vòng. Để tránh tình trạng này người ta thêm vào trạm một rờ-le điện tử (relay). Khi trạm còn mạnh khỏe, rờ-le sẽ mở và trạm được nối với vành, khi trạm bị chết và ngưng không cung cấp năng lượng cho rờ-le, rờ-le sẽ tự động đóng mạch và bỏ qua trạm này.



H5.17 Sử dụng relay để tránh đứt vòng

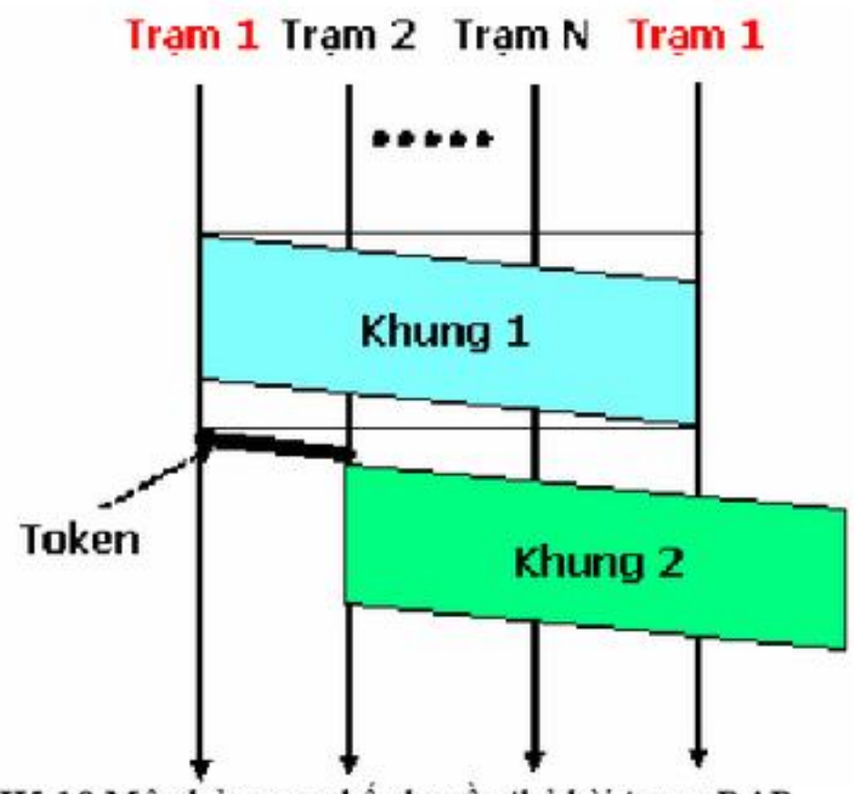
Khi monitor nghi ngờ một trạm bị chết, nó sẽ gởi đến trạm đó một khung đặc biệt gọi là khung beacon. Nếu không nhận được trả lời thích đáng, monitor sẽ coi trạm đó đã chết.

ĐÁNH GIÁ VỀ MẠNG TOKEN RING:

Ta sẽ khảo sát hai kiểu chuyển thẻ bài: Release After Reception (RAR) và Release After Transmisions (RAT).

RAR: Nhả thẻ bài sau khi nhận lại dữ liệu

Sau khi một trạm phát đi khung dữ liệu của nó, trạm sẽ chờ đến khi khung này quay trở lại mới chuyền thẻ bài cho trạm kế tiếp. Mạng IEEE 802.5 Token Ring (16Mbps) sử dụng cơ chế này.



H5.18 Mô phỏng cơ chế chuyền thẻ bài trong RAR

Ta gọi hiệu suất truyền khung là η_{RAR}. Mạng kiểu RAR sẽ đạt được hiệu suất tối đa nếu ***một trạm phát liên tục***.

Đặt:

T_{prop} là thời gian lan truyền tín hiệu giữa hai đầu mút xa nhau nhất trên đường truyền tải.

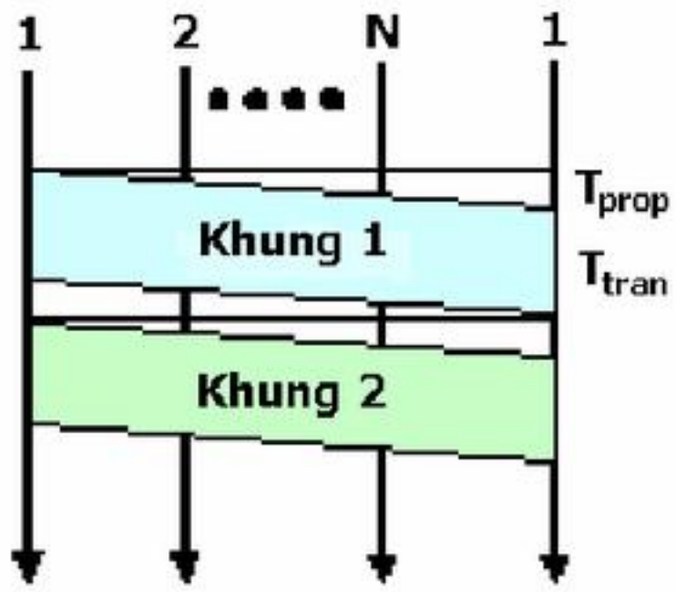
T_{tran} là thời gian để phát hết một khung dữ liệu lên đường truyền.

P là kích thước của khung dữ liệu, ví dụ 1000 bits.

C là dung lượng của đường truyền, ví dụ 10 Mbps.

Có $T_{tran} = \frac{P}{C}$

Sau đây là biểu đồ mô phỏng mối liên quan giữa thời gian phát khung và thời gian truyền tín hiệu:



H5.19 Thời gian lan truyền và phát khung với RAR

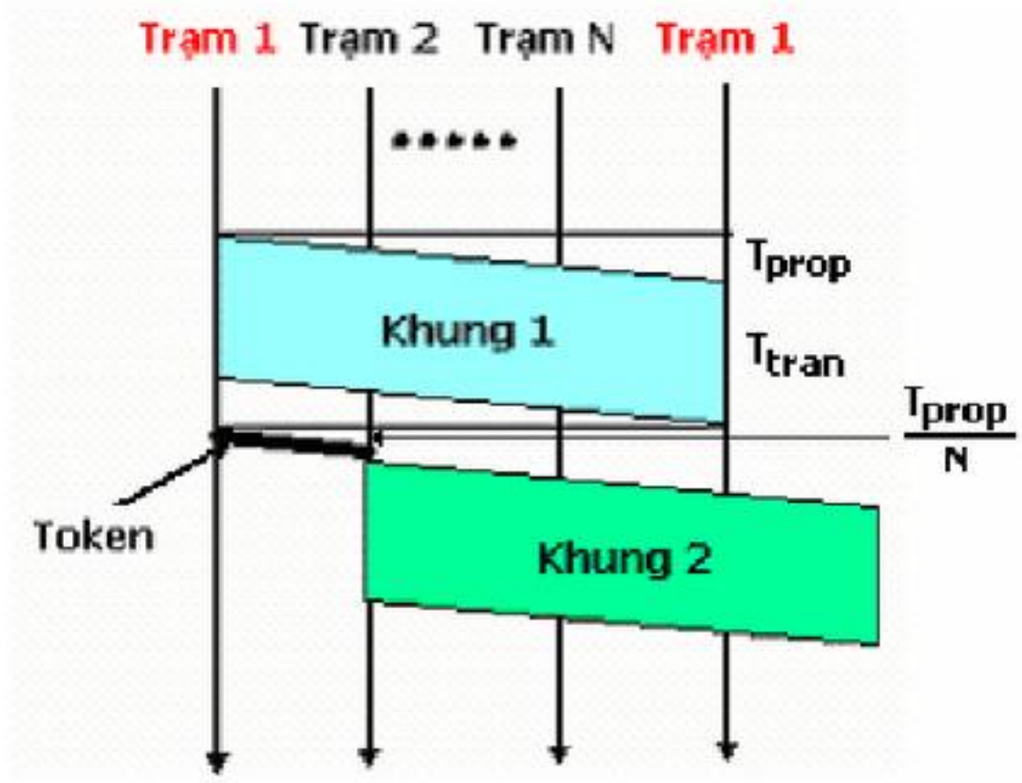
Có thể thấy:

$$\eta_{RAR} = \frac{1}{1+a}$$

Với

$$a = \frac{T_{prop}}{T_{tran}}$$

Trong trường hợp một trạm ***luôn phải nhường token*** ngay sau khi token đi hết một vòng và quay trở lại nó thì hiện trạng phân bổ thời gian được vẽ ra dưới đây:

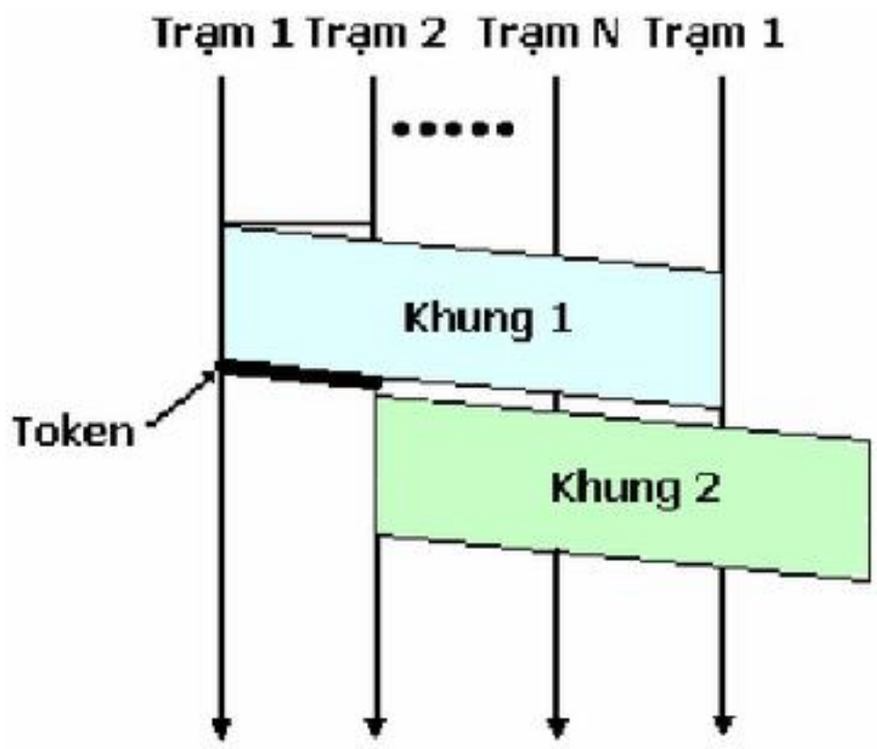


H5.20 Các khoảng thời gian một trạm phải trải qua khi gởi dữ liệu và chuyển thẻ bài

Giả sử token có kích thước quá nhỏ để coi như thời gian phát nó bằng không, mạng có N trạm làm việc và khoảng cách giữa các trạm là bằng nhau. Vì vậy, thời gian lan truyền tín hiệu từ một trạm đến một trạm liền kề nó là T_{prop}/N.
Có thể nhận thấy thời gian để token chuyển từ một trạm sang một trạm kề nó là T_{prop}/N.
Vì vậy:

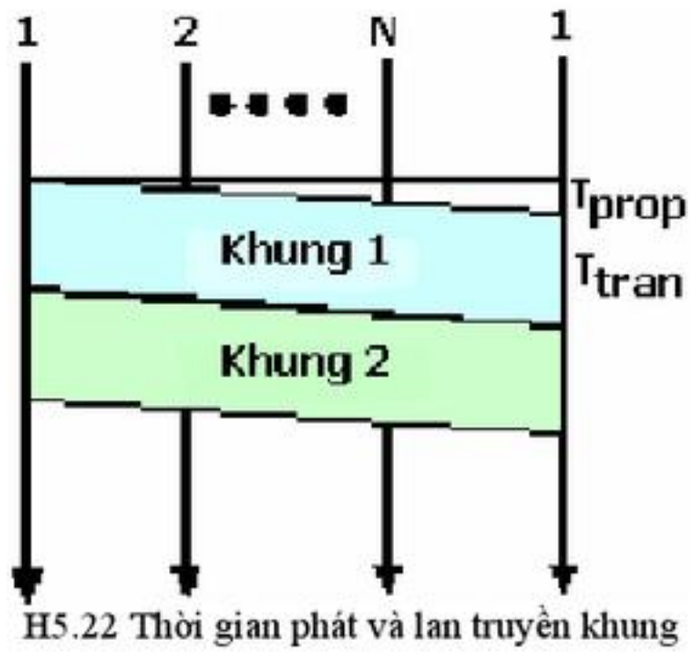
$$\eta_{RAR} = \frac{T_{tran}}{T_{prop} + T_{tran} + \frac{T_{prop}}{N}} = \frac{1}{1 + \left(\frac{N+1}{N}\right)\frac{T_{prop}}{T_{tran}}} \approx \frac{1}{1+a}$$

RAT: **Nhả thẻ bài ngay sau khi truyền dữ liệu**
Với kỹ thuật RAT, trạm sẽ chuyển thẻ bài điều khiển cho trạm kế tiếp ngay sau khi nó vừa phát song khung dữ liệu. Ví dụ mạng FDDI (Fiber Distributed Data Interface - 100Mbps) sử dụng kỹ thuật này.



H5.21 Mô phỏng kỹ thuật RAT

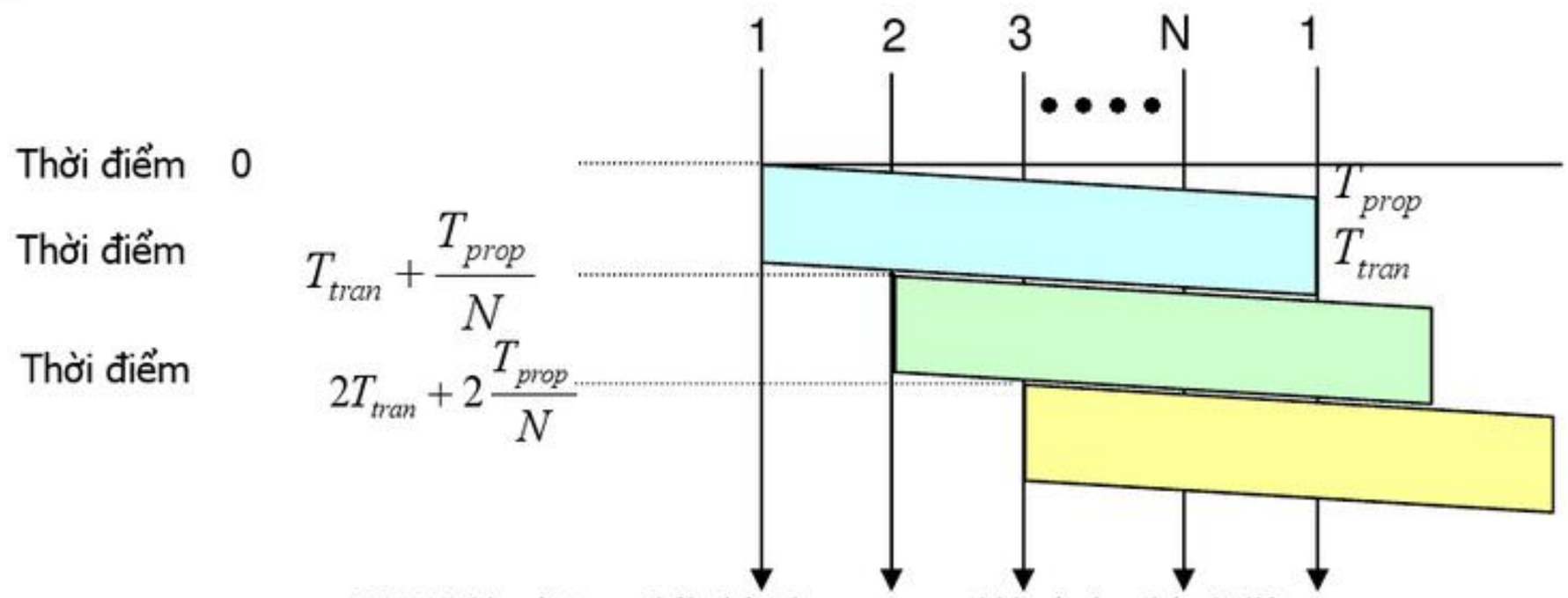
Gọi hiệu suất truyền khung là η_{RAT}. Với kỹ thuật RAT, hiệu suất đạt tối đa khi một trạm ***liên tục truyền khung***. Sau đây là biểu đồ thời gian mô phỏng:



H5.22 Thời gian phát và lan truyền khung

Có thể thấy khi một trạm phát khung liên tục thì $\eta_{RAT} = 1$

Tuy nhiên khi một trạm buộc ***phải nhả token ngay sau khi nó vừa phát dữ liệu xong***, thì biểu đồ thời gian có khác:



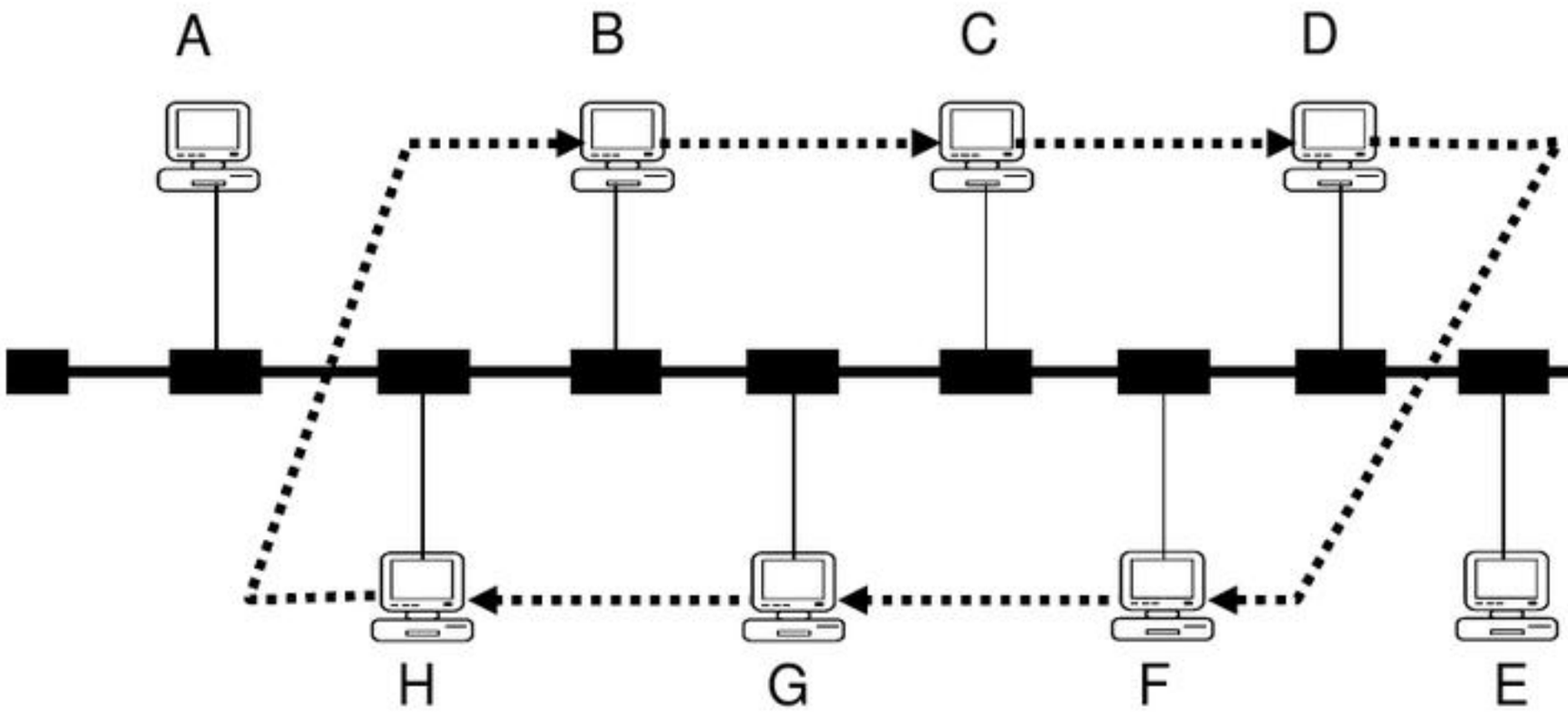
H5.23 Khi một trạm phải nhả token ngay sau khi nó vừa phát dữ liệu

Ta tính lại hiệu suất như sau:

$$\eta_{RAT} = \frac{T_{tran}}{T_{tran} + \frac{T_{prop}}{N}} = \frac{1}{1 + \frac{1}{N}\frac{T_{prop}}{T_{tran}}} = \frac{1}{1 + \frac{a}{N}}$$

Với $a = \frac{T_{prop}}{T_{tran}}$

5.3.3.3 Ví dụ về phương pháp chuyền thẻ bài: Token Bus



H5.24 Sơ đồ mạng Token Bus

Kỹ thuật Token Bus về bản chất là sử dụng mạng hình bus. Tuy nhiên người ta muốn thiết lập một vòng ảo trên đó để nó hoạt động giống như Token Ring. Nguyên tắc hoạt động như sau: trạm có nhu cầu truyền dữ liệu thì sẽ tham gia vào vòng ảo, ngược lại thì sẽ nằm ngoài và chỉ nghe thôi!

Giải thuật bổ sung một trạm vào vòng:

- Mỗi trạm trong vòng có trách nhiệm định kỳ tạo điều kiện cho các trạm khác tham gia vào vòng.
- Trước khi chuyển thẻ bài đi, trạm sẽ gởi thông báo "tìm trạm đứng sau" (có địa chỉ giữa nó và trạm đứng liền kề hiện tại).
- Nếu sau một thời gian xác định mà vẫn không có yêu cầu gia nhập nào, trạm sẽ chuyển thẻ bài đến trạm kế tiếp như thường lệ.
- Nếu có yêu cầu gia nhập vòng, thì trạm sẽ ghi nhận trạm mới yêu cầu là trạm kế tiếp của nó và sẽ chuyển thẻ bài tới trạm kế mới này.

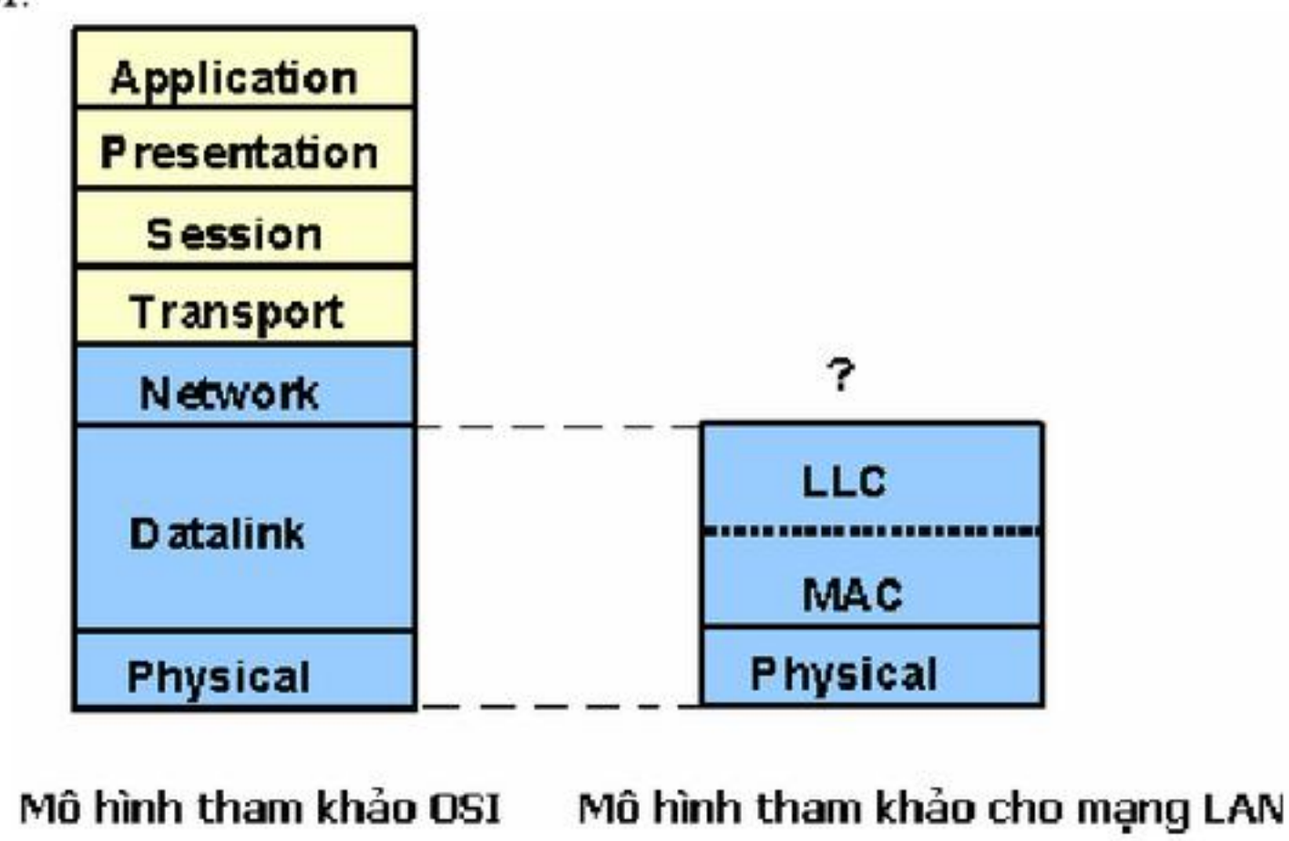
Giải thuật rút lui ra khỏi vòng:

- Khi muốn rút ra khỏi vòng, trạm sẽ chờ đến khi nó có token, sau đó sẽ gởi yêu cầu "nối trạm đứng sau" tới trạm đứng trước nó, yêu cầu trạm đứng trước nối trực tiếp với trạm đứng liền sau nó.

Ngoài ra còn phải quan tâm đến tình trạng mất thẻ bài, các trạm thành viên trong vòng bị hư hỏng.

5.4 Chuẩn hóa mạng cục bộ

Ngoài mô hình OSI dùng cho việc chuẩn hóa các mạng nói chung, việc chuẩn hóa mạng cục bộ cũng đã được thực hiện trong một khoảng thời gian dài. Do đặc trưng riêng, việc chuẩn hóa mạng cục bộ chỉ được thực hiện trên hai tầng thấp nhất, tương ứng với tầng vật lý và liên kết dữ liệu trong mô hình OSI.



H5.25 Mô hình phân tầng của mạng cục bộ

Trong LAN, tầng liên kết dữ liệu được chia làm hai tầng con: LLC (Logical Link Layer) và MAC. MAC quản lý việc truy cập đường truyền, trong khi LLC đảm bảo tính độc lập của việc quản lý các liên kết dữ liệu với đường truyền vật lý và phương pháp truy cập đường truyền MAC.

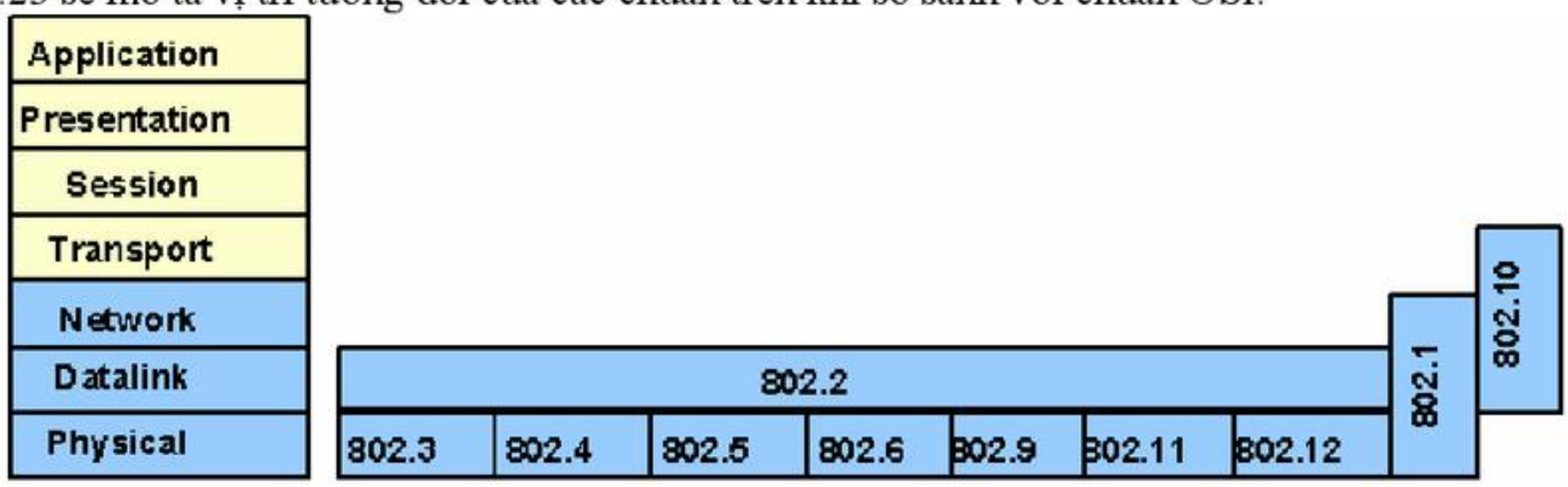
IEEE (Institute of Electrical and Electronic Engineers) là tổ chức đi tiên phong trong lĩnh vực chuẩn hóa mạng cục bộ với dự án IEEE 802 nổi tiếng bắt đầu được triển khai từ năm 1980 và kết quả là hàng loạt chuẩn thuộc họ IEEE 802.x ra đời, tạo nền tảng quan trọng cho việc thiết kế và cài đặt mạng nội bộ trong thời gian qua. Vị trí của họ chuẩn này càng cao hơn khi ISO đã xem xét và tiếp nhận chúng thành chuẩn quốc tế mang tên 8802.x.

Đến nay họ IEEE 802.x bao gồm các chuẩn sau:

IEEE 802.1 : High Level Interface
IEEE 802.2 : Logical Link Control (LLC)
IEEE 802.3: CSMA/CD

IEEE 802.4: Token bus
IEEE 802.5: Token ring
IEEE 802.6: MAN
IEEE 802.7: Broadband Technical Advisory Group
IEEE 802.8: Fiber Technical Advisory Group
IEEE 802.9: Intergrated Data and Voice Network
IEEE 802.10: Standard for Interoperable LAN security
IEEE 802.11: Wireless LAN
IEEE 802.12: 100VG – AnyLAN

H5.23 sẽ mô tả vị trí tương đối của các chuẩn trên khi so sánh với chuẩn OSI:



H5.26 Quan hệ giữa các chuẩn IEEE và mô hình OSI

- IEEE 802.1 là chuẩn đặc tả kiến trúc mạng, nối kết giữa các mạng và việc quản trị mạng đối với mạng cục bộ.
- IEEE 802.2 là chuẩn đặc tả tầng LLC (dịch vụ, giao thức) của mạng cục bộ.
  Có 3 kiểu giao thức LLC chính được định nghĩa:
  LLC type 1: Là giao thức kiểu không liên kết, không báo nhận.
  LLC type 2: Là giao thức kiểu có liên kết.
  LLC type 3: Là giao thức dạng không liên kết, có báo nhận.
  Các giao thức này được xây dựng dựa theo phương thức cân bằng của giao thức HDLC và có các khuôn dạng dữ liệu và các chức năng tương tự, đặc biệt là trong trường hợp LLC-type 2.
- IEEE 802.3: Là chuẩn đặc tả một mạng cục bộ dựa trên mạng Ethernet nổi tiếng do Digital, Intel và Xerox hợp tác phát triển từ năm 1990. IEEE 802.3 bao gồm cả tầng vật lý và tầng con MAC với các đặc tả sau:
  - Đặc tả dịch vụ MAC.
  - Giao thức MAC.
  - Đặc tả vật lý độc lập với đường truyền.
  - Đặc tả vật lý phụ thuộc vào đường truyền.

  Phần cốt lõi của IEEE 802.3 là giao thức MAC dựa trên phương pháp CSMA/CD đã trình bày ở phần trước.
- IEEE 802.4 là chuẩn đặc tả mạng cục bộ với hình trạng bus sử dụng thẻ bài để điều khiển truy cập đường truyền.
  IEEE 802.4 cũng bao gồm cả tầng vật lý và tầng con MAC với các đặc tả sau:
  - Đặc tả dịch vụ MAC.
  - Giao thức MAC.
  - Đặc tả dịch vụ tầng vật lý.
  - Đặc tả thực thể tầng vật lý.
  - Đặt tả đường truyền.
- IEEE 802.5 là chuẩn đặc tả mạng cục bộ với hình trạng vòng sử dụng thẻ bài để điều khiển truy cập đường truyền.
  IEEE 802.5 cũng bao gồm cả tầng vật lý và tầng con MAC với các đặc tả sau:
  - Đặc tả dịch vụ MAC.

  - Giao thức MAC.
  - Đặc tả thực thể tầng vật lý.
  - Đặc tả nối trạm.
- IEEE 802.6 là chuẩn đặc tả một mạng tốc độ cao nối kết nhiều LAN thuộc các khu vực khác nhau của một đô thị. Mạng này sử dụng cáp quang với hình trạng dạng bus kép (dual-bus), vì thế còn được gọi là DQDB (Distributed Queue Dual Bus). Lưu thông trên mỗi bus là một chiều và khi cả cặp bus cùng hoạt động sẽ tạo thành một cấu hình chịu lỗi. Phương pháp điều khiển truy cập dựa theo một giải thuật xếp hàng phân tán có tên là QPDS (Queued-Packet, Distributed-Switch).
- IEEE 802.9 là chuẩn đặc tả một mạng tích hợp dữ liệu và tiếng nói bao gồm 1 kênh dị bộ 10 Mbps cùng với 95 kênh 64 Kbps. Giải thông tổng cộng 16 Mpbs. Chuẩn này được thiết kế cho các môi trường có lưu lượng lưu thông lớn và cấp bách.
- IEEE 802.10 là chuẩn đặc tả về an toàn thông tin trong các mạng cục bộ có khả năng liên tác (interoperable).
- IEEE 802.11 là chuẩn đặc tả mạng LAN không dây (Wireless LAN). Xu hướng chọn phương pháp truy cập CSMA được khẳng định.
- IEEE 802.12 là chuẩn đặc tả mạng cục bộ dựa trên công nghệ được đề xuất bởi AT&T, IBM và HP, gọi là 100 VG – AnyLAN. Mạng này sử dụng hình trạng mạng hình sao và một phương pháp truy cập đường truyền có điều khiển tranh chấp. Khi có nhu cầu truyền dữ liệu, trạm sẽ gởi yêu cầu đến hub và trạm chỉ có thể truyền dữ liệu khi được hub cho phép. Chuẩn này nhằm cung cấp một mạng tốc độ cao (100 Mbps và có thể lớn hơn) có thể hoạt động trong các môi trường hỗn hợp Ethernet và Token Ring, bởi thế nó chấp nhận cả hai dạng khung. 100 VG – AnyLAN là đối thủ cạnh tranh đáng gờm của 100BASE-T (Fast Ethernet) nhờ một số tính năng trội hơn, chẳng hạn về khoảng cách đi cáp tối đa cho phép…

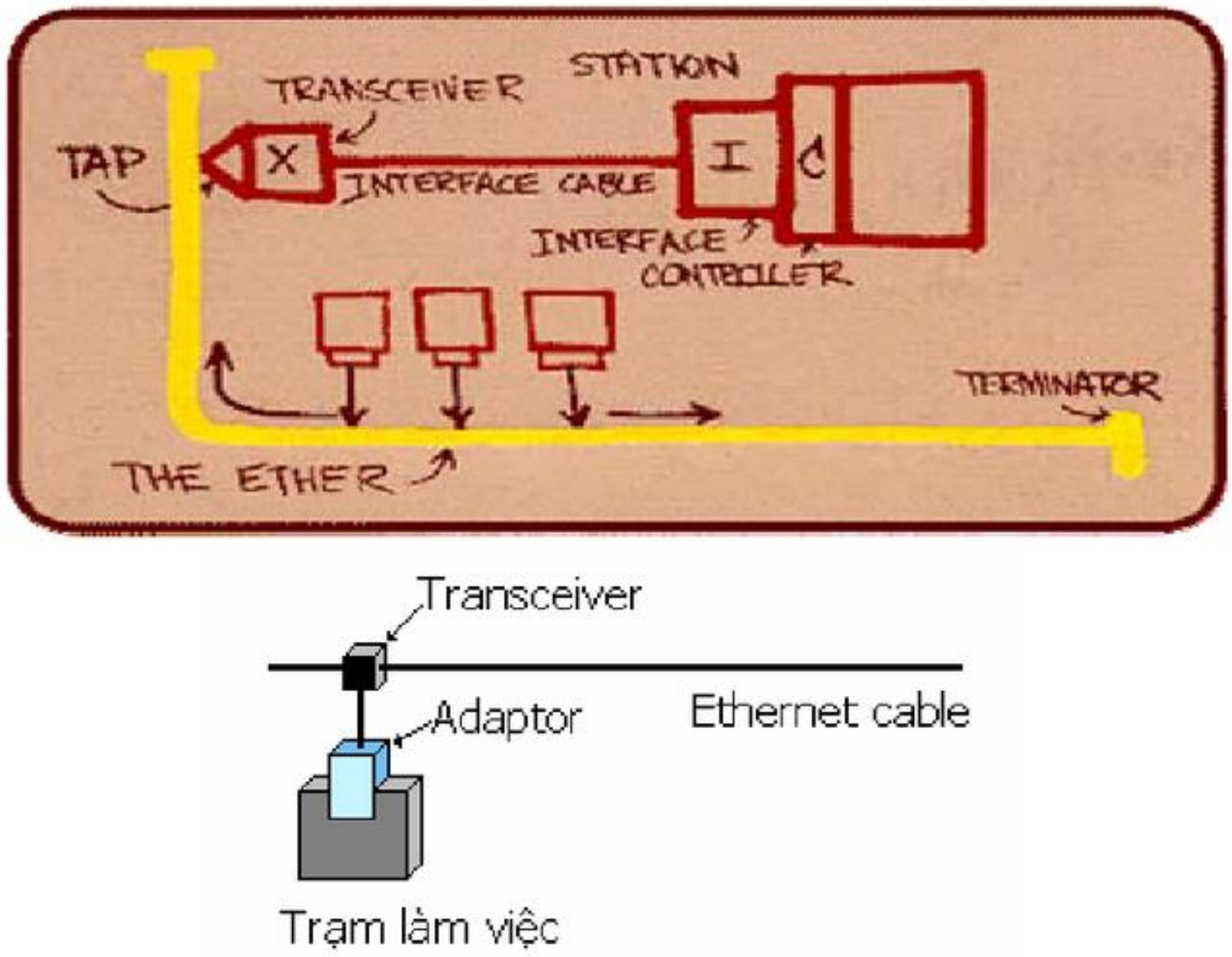
5.5 Giới thiệu một số công nghệ mạng LAN

5.5.1 Ethernet (802.3)

Ethernet đã dễ dàng trở thành công nghệ mạng LAN thành công nhất trong suốt 20 năm qua. Được phát triển vào giữa thập kỷ 1970s bởi các nhà nghiên cứu tại Xerox Palo Atlto Research Center (PARC), Ethernet là một ví dụ thực tiễn của loại mạng cục bộ sử dụng giao thức CSMA/CD.

5.5.2 Tổng quan

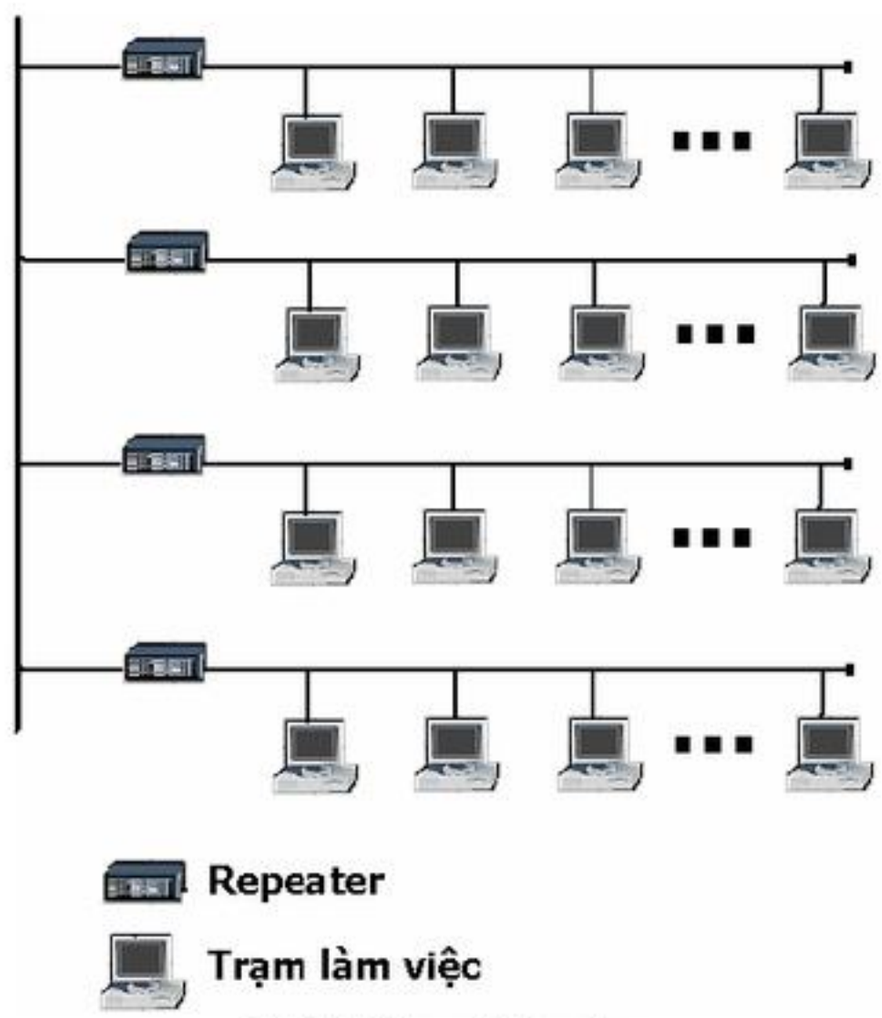
Khởi thủy, một phân đoạn mạng của Ethernet (Ethernet segment) được cài đặt trên một sợi cable đồng trục dài tối đa 500 m. Các trạm nối vào Ethernet segment bằng cách "mắc dây" (tab) nối vào nó. Các điểm đấu nối phải cách nhau ít nhất 2,5 m. T*ransceiver,* một thiết bị nhỏ được gắn trực tiếp vào điểm đấu nối, làm nhiệm vụ nghe ngóng khi đường truyền rỗi để đưa tín hiệu ra đó khi trạm phát tín hiệu. Transceiver cũng làm nhiệm vụ nhận tín hiệu đến. Đến lượt transceiver lại được nối đến card mạng Ethernet, được gắn trong máy trạm. Tất cả những chi tiết luận lý làm nên giao thức Ethernet được cài đặt trong card mạng này.



H5.27 Bức phác họa Ethernet của Bob Metcalfe, người sáng lập ra Ethernet (Xerox PARC - 1972) *Và mô tả chi tiết transceiver + adaptor*

Các segment có thể được nối với nhau bởi các repeater. Một repeater là một thiết bị dùng để chuyển tiếp tín hiệu số. Tuy nhiên, không được có hơn 4 repeater được đặt giữa hai trạm, có nghĩa là một mạng Ethernet nguyên thủy chỉ kéo dài tối đa là 2500 m.

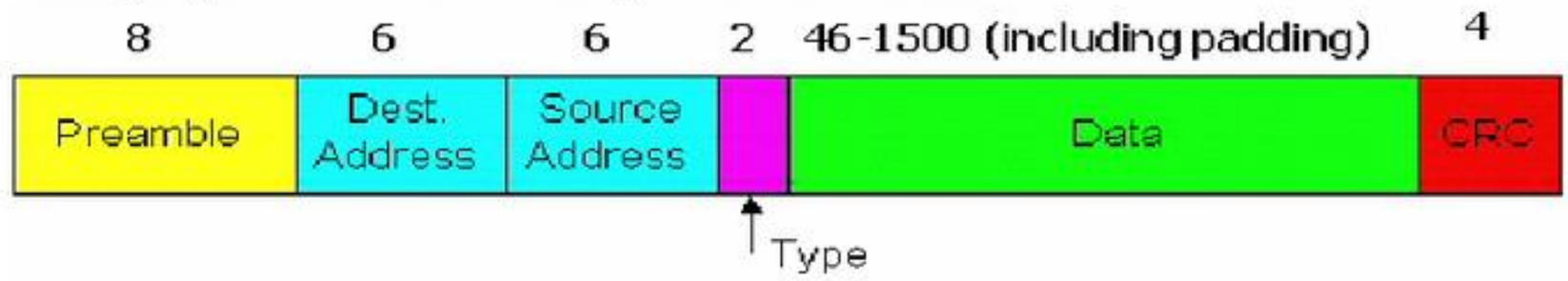
Bất kỳ tín hiệu nào được phát ra Ethernet sẽ được truyền quảng bá ra toàn mạng, repeater có nhiệm vụ chuyển tín hiệu từ trong segment ra ngoài, và nhận tín hiệu từ ngoài phát quảng bá vào trong segment.



H5.28 Ethernet Repeater

5.5.2.1 Khuôn dạng khung thông tin của Ethernet

Bên gởi sẽ bao gói gói tin IP thành khung Ethernet như sau:



H5.29 Khuôn dạng khung thông tin Ethernet

- Preamble: dài 7 bytes với mẫu 10101010 theo sau bởi 1 byte với mẫu 10101011, được sử dụng để đồng bộ hóa tốc độ đồng hồ giữa bên gởi và bên nhận.
- Source and dest. addresses: Địa chỉ nguồn và đích, gồm 6 bytes. Khung được nhận bởi tất cả các trạm trong LAN. Khung bị xóa nếu dest. address không trùng với địa chỉ MAC của bất kỳ trạm nào hoặc không phải thuộc dạng multicast.
- Type: chỉ ra giao thức được sử dụng ở tầng cao hơn, thường là IP, nhưng các giao thức khác vẫn được hỗ trợ - ví dụ: Novell IPX và AppleTalk.
- CRC: Phần kiểm tra lỗi. Lỗi được kiểm tra tại trạm đích. Nếu khung có lỗi, nó sẽ bị xóa.

5.5.2.2 Địa chỉ Ethernet

Mỗi host trong một mạng Ethernet (thật ra là tất cả các host trên thế giới) có một địa chỉ Ethernet duy nhất. Mô tả một cách kỹ thuật, địa chỉ được gắn vào card mạng chứ không phải máy tính; nó được ghi vào ROM trên card mạng. Các địa chỉ Ethernet thường được in theo thể thức mà con người có thể đọc được: một dãy gồm 6 bytes được viết dưới dạng thập lục phân, cách nhau bởi dấu hai chấm. Ví dụ **8:0:2b:e4:b1:2** là cách biểu diễn dễ đọc của địa chỉ Ethernet sau

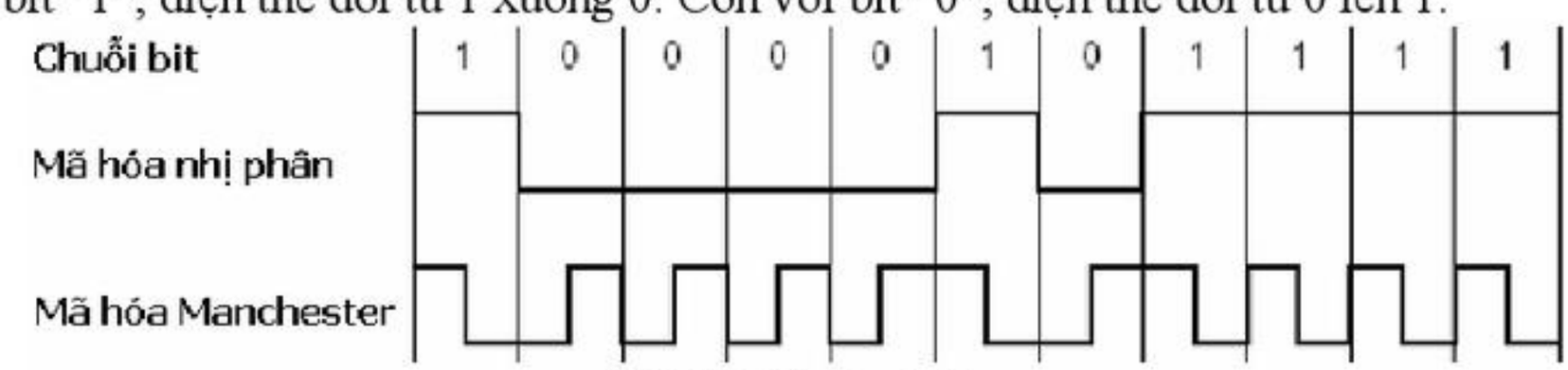
00001000 00000000 00101011 11100100 10110001 00000010

Để đảm bảo rằng mọi card mạng được gán một địa chỉ duy nhất, mỗi nhà sản xuất thiết bị Ethernet được cấp cho một phần đầu địa chỉ (prefix) khác nhau. Ví dụ Advanced Micro Devices đã được cấp phần đầu dài 24 bit x08002 (hay 8:0:2). Nhà sản xuất này sau đó phải đảm bảo phần đuôi (suffix) của các địa chỉ mà họ sản xuất ra là duy nhất.

Mỗi khung được phát ra Ethernet sẽ được nhận bởi tất cả các card mạng có nối với đường truyền. Mỗi card mạng sẽ so sánh địa chỉ đích trong khung với địa chỉ của nó, và chỉ cho vào máy tính những khung nào trùng địa chỉ. Địa chỉ duy nhất như vậy gọi là địa chỉ *unicast*. Ngoài ra còn có loại địa chỉ *broadcast* là loại địa chỉ quảng bá, có tất cả các bit đều mang giá trị 1. Mọi card mạng đều cho phép các khung thông tin có địa chỉ đích là *broadcast* đi đến host của nó. Cũng có một loại địa chỉ khác gọi là *multicast*, trong đó chỉ một vài bit đầu được đặt là 1. Một host có thể lập trình điều khiển card mạng của nó chấp nhận một số lớp địa chỉ *multicast*. Địa chỉ *multicast* được dùng để gởi thông điệp đến một tập con (subset) các host trong mạng Ethernet.

5.5.2.3 Cách thức mã hóa tín hiệu

Để chuyển đổi dữ liệu bit sang tín hiệu truyền trên đường truyền, Ethernet dùng kiểu mã hóa Manchester. Trong sơ đồ mã hóa Manchester, một bit sẽ được mã hóa bằng một sự thay đổi điện thế. Với bit "1", điện thế đổi từ 1 xuống 0. Còn với bit "0", điện thế đổi từ 0 lên 1.



H5.30 Mã hóa Machester

5.5.2.4 Giải thuật truy cập đường truyền

Ethernet sử dụng giải thuật CSMA/CS+Exponential backoff, được trình bày cụ thể như sau:

```
Nhận một gói tin từ tầng cao hơn;
K := 0; n :=0; // K: thời gian chờ đợi ngẫu nhiên; n: số vụ đụng độ đã gặp phải
repeat:
chờ trong khoảng thời gian K*512 bit-time;
        while (đường truyền bận) wait;
        chờ tiếp 96 bit-time sau khi nhận thấy không có tín hiệu trên đường truyền;
        truyền khung và chú ý phát hiện đụng độ;
        if (có đụng độ)
        {   ngừng truyền và phát tiếp một dãy nhồi 48-bit;
            n ++;
            m:= min(n, 10);
            chọn K ngẫu nhiên từ tập hợp {0, 1, 2, ..., 2^m-1}.
            if (n < 16) goto repeat;
            else bỏ việc truyền;
        }
```

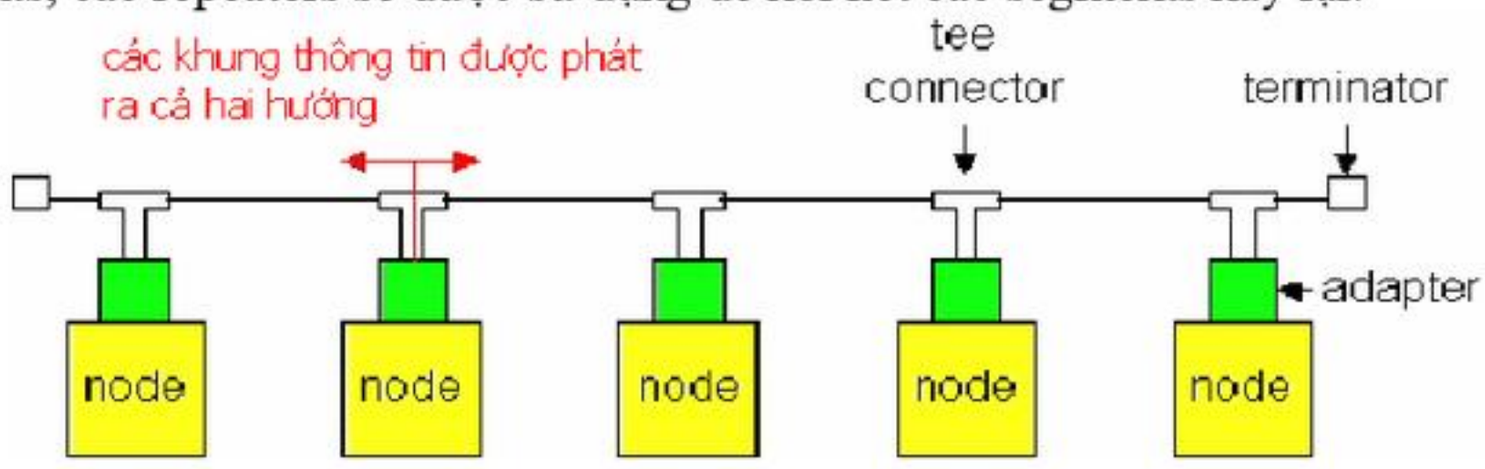
5.5.2.5 Các công nghệ Ethernet

5.5.2.5.1 10-BASE-2

Giải thích các ký hiệu:

- 10: 10 Mbps
- 2: chiều dài cable tối đa là 200 m
- Base: Baseband, Broad: Broadband.

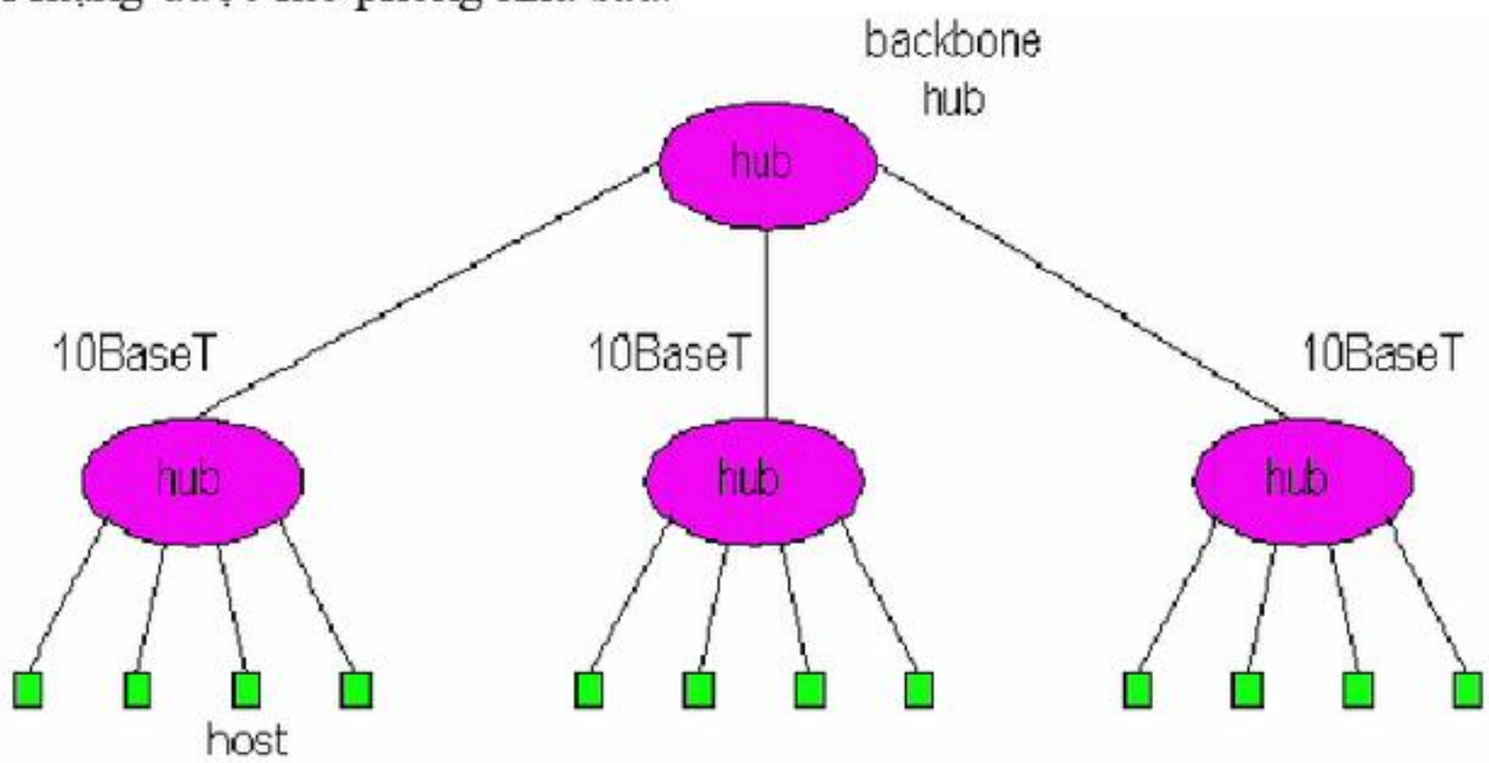
Mạng Ethernet 10BASE2 sử dụng cáp đồng trục gầy, hình thái bus. Trong trường hợp mạng có nhiều segments, các repeaters sẽ được sử dụng để nối kết các segments này lại.



Hình 2.31 Mô hình mạng 10BASE2

5.5.2.5.2 10-BASE-T và 100-BASE-T

Mạng đạt tốc độ 10/100 Mbps, về sau được gọi là "fast Ethernet". Chữ T viết tắt cho Twisted Pair: cáp xoắn đôi.
Cách thức nối mạng được mô phỏng như sau:



H5.32 Mô hình mạng 10BaseT

Các HUB được nối tới các SWITCH bằng cáp xoắn đôi. Với cách thức đấu nối như vậy, mạng được gọi là "hình sao". Cơ chế CSMA/CD được cài đặt tại HUB

5.5.2.5.3 GIGABIT ETHERNET

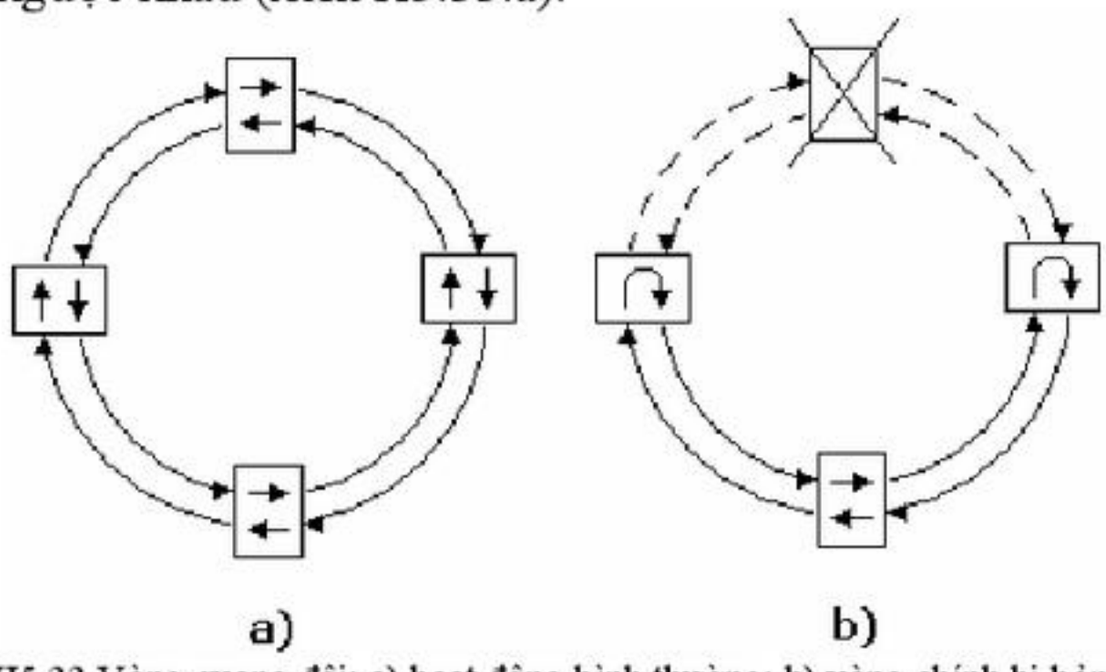
Gbit Ethernet sử dụng khuôn dạng khung chuẩn của Ethernet. Nó cho phép mạng hoạt động trên cả hai kiểu nối kết điểm điểm và kênh quảng bá chia sẻ.
Trong kiểu nối kết điểm-điểm, Gbit Ethernet sử dụng các switches thay cho các hub. Đường truyền được sử dụng theo kiểu hai chiều đồng thời với tốc độ 1 Gbps.
Trong kiểu kênh quảng bá chia sẻ, Gbit Ethernet sử dụng các hubs làm kênh quảng bá và áp dụng giải thuật CSMA/CD để các trạm chia sẻ việc sử dụng các hubs này.

5.5.3 FDDI (Fiber Distributed Data Interface)

Theo nhiều phương diện, FDDI tương tự như 802.5 và IBM Token Ring. Tuy nhiên, cũng có những khác biệt đáng kể, một số phát sinh vì lý do FDDI chạy trên cáp quang, một số khác phát sinh do có nhiều đổi mới sau khi người ta phát minh ra IBM Token Ring. Chúng ta sẽ thảo luận về những khác biệt quan trọng trong phần dưới đây.

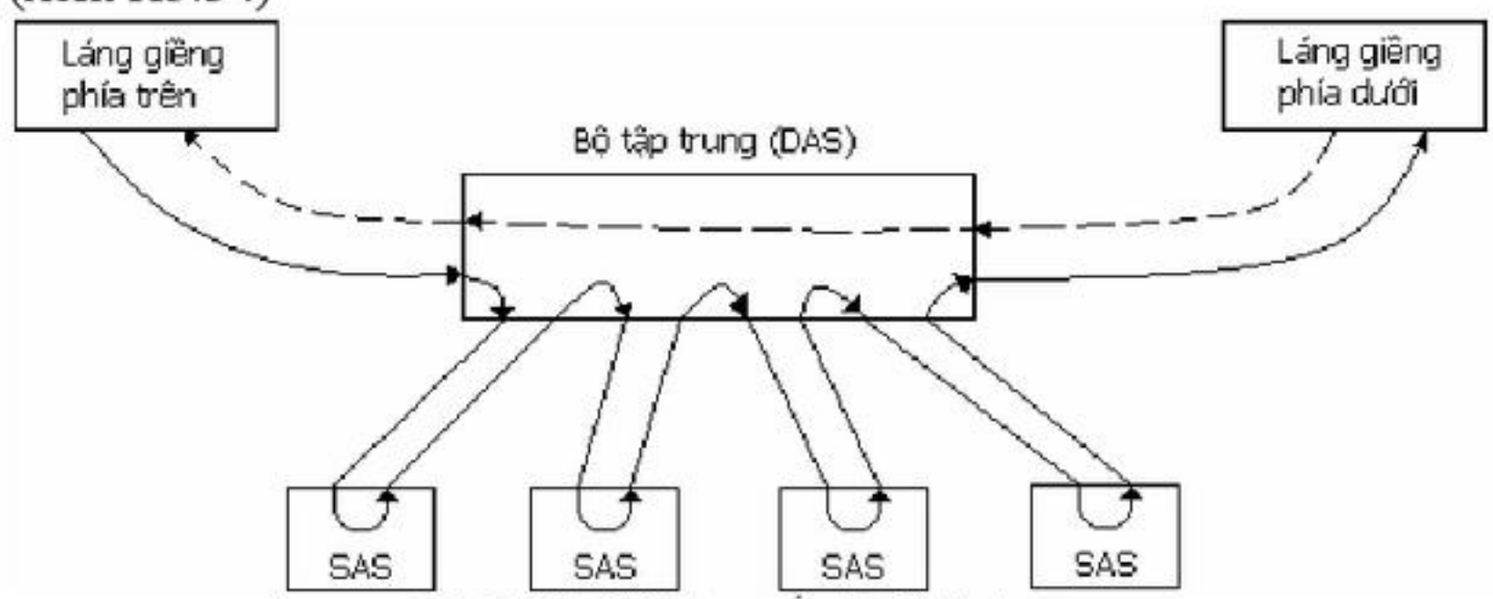
5.5.3.1 Các tính chất vật lý

Không giống như mạng 802.5, một mạng FDDI bao gồm một vòng đôi = hai vòng độc lập truyền dữ liệu theo hai chiều ngược nhau (xem H5.33.a).



H5.33 Vòng quang đôi: a) hoạt động bình thường; b) vòng chính bị hỏng

Vòng phụ không được sử dụng trong khi hệ thống hoạt động bình thường, nó chỉ vào cuộc khi vòng chính bị sự cố (xem H5.33.b). Nghĩa là vòng chính sẽ quành lại vòng phụ để tạo ra một vòng hoàn chỉnh, và chính điều này giúp cho FDDI có khả năng chịu lỗi khi một cộng cáp bị đứt hay một trạm trong vòng bị hỏng.
Do phải chịu phí tổn khi cấu hình theo kiểu vòng đôi, nên FDDI còn cho phép một trạm chọn nối vào chỉ một vòng đơn thôi. Những trạm như vậy gọi là những "trạm nối đơn" (single attachment station – SAS). Những trạm nối cả vào hai vòng dĩ nhiên sẽ được gọi là những "trạm đấu đôi" (dual attachment station – DAS). Một bộ tập trung (concentrator) sẽ được sử dụng để nối các SAS vào vòng đôi (xem H5.34)



H5.34 Các SAS được nối vào bộ tập trung

Nếu một SAS bị hỏng hóc, bộ tập trung sẽ phát hiện ra tình trạng này và sử dụng cơ chế bỏ qua tín hiệu quang (obtical bypass) để cô lập SAS bị hỏng, vì thế giữ cho vòng được thông suốt.
Trong FDDI, bộ đệm của giao diện mạng có thể có kích thước khác nhau tại những trạm khác nhau, mặc dù kích thước của nó không bao giờ nhỏ hơn 9 bit và lớn hơn 80 bit. Một trạm cũng có thể bắt đầu phát các bit trong bộ đệm đi trước khi bộ đệm của nó bị đầy. Dĩ nhiên là tổng thời gian để một thẻ bài di chuyển hết một vòng là một hàm của kích thước của các bộ đệm này. Ví dụ, FDDI là mạng tốc độ 100 Mbps, nó có thời gian xử lý 1 bit là 10 ns. Nếu mỗi trạm cài đặt buffer dài 10 bit và chờ cho đến khi buffer bị đầy một nửa mới bắt đầu truyền, thì mỗi trạm tạo ra thời gian trì hoãn là 5×10 ns = 50 ns đối với tổng thời gian xoay vòng mạng.
FDDI còn có các tính chất vật lý khác. Chẳng hạn, một mạng đơn có giới hạn chuẩn là có tối đa 500 trạm làm việc, với khoảng cách xa nhất giữa một cặp trạm bất kỳ là 2 km. Nhưng trên hết, mạng lại bị giới hạn với kích thước tổng cộng là 200 km cáp quang. Do tính chất là vòng đôi, nên tổng kích thước cáp quang nối tất cả các trạm là 100 km. Ngoài ra, mặc dù ký tự "F" ám chỉ cáp quang, nhưng chuẩn FDDI đã được định nghĩa để có thể chạy trên một số thiết bị tải khác, bao gồm cả cáp đồng trục và cáp xoắn đôi. Tuy nhiên cũng nên cẩn thận khi xét đến tổng kích thước mà vòng bao phủ. Như chúng ta sẽ thấy dưới đây, lượng thời gian bỏ ra để cho thẻ bài đi hết một vòng mạng sẽ đóng vai trò quan trọng trong giải thuật điều khiển truy cập.
FDDI sử dụng phương pháp mã hóa 4B/5B. Do FDDI chuẩn mạng phổ biến đầu tiên sử dụng cáp quang, nên các con chíp mã hóa dạng 4B/5B chạy trên tốc độ của FDDI có rất nhiều ngoài thị trường.

5.5.3.2 Giải thuật "Thẻ bài được định thời" – Timed Token

Các luật qui định thời gian giữ thẻ bài trong FDDI phức tạp hơn trong 802.5 một ít. THT cho mỗi trạm được tính như trong phần trình bày về Token Ring, và được tinh chỉnh để có được một giá trị hợp lý. Ngoài ra, để đảm bảo cho mỗi trạm có cơ hội truyền trong một khoảng thời gian cụ thể nào đó, nghĩa là để đặt một cận trên cho giá trị TRT mà mọi trạm đều thấy được, chúng ta định nghĩa thông số "đích thời gian xoay vòng của thẻ bài" (target token rotation time – TTRT). (Việc các trạm thống nhất với nhau về thời gian TTRT như thế nào sẽ được trình bày trong phần phía dưới). Mỗi trạm đo thời gian giữa hai lần liên tiếp thẻ bài đến nó, chúng ta gọi thời gian này là TRT đo được của trạm (measured TRT). Nếu thời gian TRT đo được của trạm lớn hơn thời gian TTRT được dàn xếp trước đó thì thẻ bài bị trễ, vì thế trạm sẽ không được truyền dữ liệu nữa. Ngược lại, thẻ bài đến sớm và trạm sẽ có quyền giữ thẻ bài trong khoảng thời gian (TTRT-TRT).

Tuy nhiên, có vấn đề phát sinh như sau: Nếu một nút có quá nhiều dữ liệu cần phải gởi có cơ hội giữ thẻ bài, nó sẽ tận dụng hết thời gian giữ thẻ bài được phép. Vì thế nút kế sau nó sẽ tính toán và thấy thời gian TTRT và TRT là bằng nhau, nghĩa là nút kế sau này sẽ không có quyền truyền dữ liệu nữa. Để tính đến khả năng này, FDDI định nghĩa hai lớp giao thông trên mạng: đồng bộ và dị bộ. Khi một trạm có được thẻ bài, nó luôn được phép gởi dữ liệu dạng đồng bộ mà không cần phải quan tâm là thẻ bài tới sớm hay trễ. Ngược lại, trạm có thể gởi dữ liệu dạng dị bộ chỉ khi thẻ bài tới sớm.

Chú ý rằng các khái niệm đồng bộ và dị bộ ở đây có thể gây hiểu lầm. Bằng cách dùng khái niệm đồng bộ, FDDI ám chỉ rằng giao thông trên mạng là có nhạy cảm với độ trễ thông tin. Ví dụ, người ta sẽ muốn gởi âm thanh hay video trên mạng FDDI theo kiểu đồng bộ. Ngược lại, những ứng dụng nào quan tâm đến thông lượng của đường truyền thì sẽ thích kiểu truyền dị bộ hơn. Ví dụ, ứng dụng truyền file trên mạng sẽ muốn sử dụng kiểu lưu thông dị bộ trên FDDI.

Lại thêm vấn đề phát sinh: Do kiểu lưu thông dạng đồng bộ không có quan tâm đến việc thẻ bài đến sớm hay muộn, nên có khả năng nếu trạm co-dãn thời gian gởi dữ liệu đồng bộ thì thông số TTRT không còn ý nghĩa gì nữa! Để giải quyết vấn đề này, ta qui định: tổng thời gian gởi dữ liệu đồng bộ trong một vòng xoay của thẻ bài không được vượt quá TTRT.

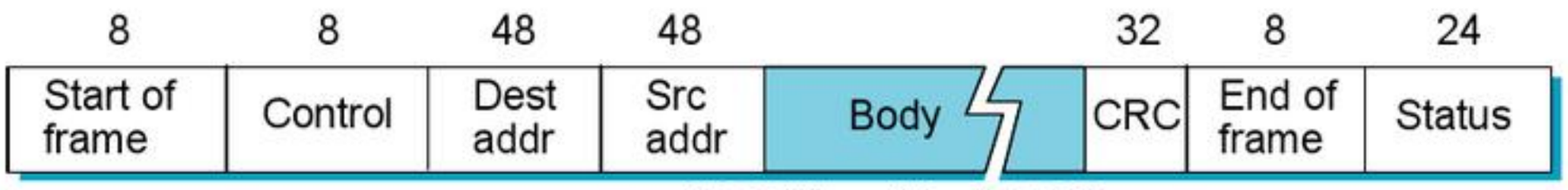
5.5.3.3 Quản lý thẻ bài

Các cơ chế mà FDDI dùng để đảm bảo luôn có một thẻ bài hợp lệ chạy trên vòng cũng khác so với 802.5, do chúng dính với quá trình thiết đặt TTRT. Trước tiên, tất cả các trạm trong vòng luôn quan sát để đảm bảo là thẻ bài không bị mất. Để ý rằng một trạm trên vòng luôn có thể thấy được thời gian truyền hợp lệ của khung hay là thẻ bài. Thời gian rỗi tối đa giữa những lần truyền hợp lệ mà mỗi trạm nên biết qua là bằng độ trễ của vòng cộng với thời gian bỏ ra để truyền một khung đầy. (Trên một vòng có kích thước tối đa, thời gian tối đa để truyền một khung đầy là nhỏ hơn 2.5 ms). Do đó, mỗi trạm sẽ đặt bộ đếm lên, bộ đếm này sẽ hết hạn trong thời gian 2.5 ms. Nếu bộ đếm hết hạn, trạm sẽ nghi ngờ là có vấn đề không ổn và sẽ phát khung "thỉnh cầu". Ngược lại khi bộ đếm chưa hết hạn mà trạm thấy được một sự truyền hợp lệ, nó sẽ đặt lại giá trị tối đa của bộ đếm là 2.5 ms.

Khung thỉnh cầu trong FDDI khác với trong 802.5 do nó có chứa giá trị TTRT mà trạm mời chào (bid for the TTRT), nghĩa là: đó là thời gian xoay vòng của thẻ bài mà trạm cảm thấy đủ để các ứng dụng chạy trên nó thỏa mãn các ràng buộc về thời gian. Một trạm có thể gởi khung thỉnh cầu trong khi không có thẻ bài trong tay, và nó thường làm vậy khi có nghi ngờ mất thẻ bài hoặc khi mới tham gia vào vòng lần đầu tiên. Nếu khung thỉnh cầu đi được hết một vòng và quay lại, trạm phát sẽ xóa nó đi và hiểu rằng giá trị TTRT được mời chào trong đó là giá trị nhỏ nhất. Bây giờ có thể coi là trạm đang nắm thẻ bài, nghĩa là nó có trách nhiệm chèn vào mạng một thẻ bài hợp lệ, và có thể tiếp tục với giải thuật dùng thẻ bài thông thường.

Khi một nút nhận được một khung thỉnh cầu, nó kiểm tra xem giá trị TTRT được mời chào trong đó có nhỏ hơn giá trị TTRT của chính nó không. Nếu nhỏ hơn, nó sẽ đặt lại giá trị TTRT của nó thành giá trị TTRT được mời chào. Nếu lớn hơn, trạm sẽ xóa khung thỉnh cầu này đi, chèn vào vòng một khung thỉnh cầu mới chứa giá trị TTRT được mời chào bằng giá trị TTRT của nó. Trường hợp cuối cùng, nếu giá trị TTRT được mời chào bằng với giá trị TTRT của trạm, trạm sẽ so sánh địa chỉ của trạm gởi khung thỉnh cầu và địa chỉ của nó, địa chỉ nào lớn hơn sẽ thắng – nghĩa là bên thắng sẽ sở hữu khung thỉnh cầu. Do đó, nếu một khung thỉnh cầu đi hết một vòng và trở về trạm gởi, trạm gởi sẽ biết rằng nó là người mời chào giá trị TTRT duy nhất và nó có thể tạo ra một thẻ bài mới một cách an toàn. Tại thời điểm đó, tất cả các trạm bấy giờ đều đồng ý rằng giá trị TTRT là giá trị đủ nhỏ để làm cho tất cả chúng cảm thấy hạnh phúc!

5.5.4 Khuôn dạng của khung thông tin



H5.35 Khung thông tin FDDI

Do FDDI sử dụng cơ chế mã hóa 4B/5B, nên nó dùng các ký tự điều khiển 4B/5B. Ngoài ra trong phần Control có tồn tại một bit để phân biệt kiểu lưu thông là dị bộ hay đồng bộ.

5.5.5 Mạng không dây (802.11)

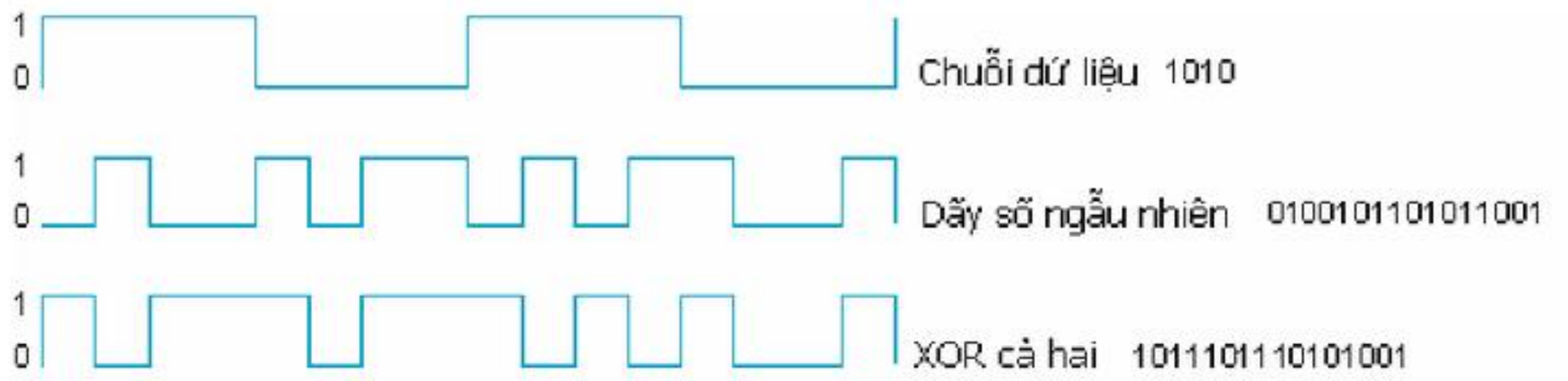
Mạng không dây là kỹ thuật nối mạng đang phát triển nhanh hiện nay. Như chúng ta đã biết, khả năng để xây dựng mạng không dây là hầu như không có giới hạn, trải dài từ cách sử dụng tín hiệu hồng ngoại để dựng mạng nội bộ trong phạm vi một tòa nhà cho đến việc kiến thiết mạng toàn cầu từ một lưới các vệ tinh quĩ đạo thấp. Phần này sẽ có cái nhìn gần hơn vào một kỹ thuật cụ thể xoay quanh chuẩn 802.11 đang nổi hiện nay. Giống như những người anh em Ethernet và Token Ring, 802.11 được thiết kế để hoạt động trong một phạm vi địa lý hẹp (các ngôi nhà, các tòa nhà văn phòng, các khu đại học) và thách thức quan trọng nó đặt ra là phải trù tính đến việc truy xuất đến phương tiện truyền thông chia sẻ - trong trường hợp này là các tín hiệu lan truyền trong không gian. 802.11 hỗ trợ thêm một số tính năng như các dịch vụ có giới hạn về thời gian, quản lý năng lượng và các cơ chế an toàn, nhưng chúng ta chỉ tập trung vào thảo luận về chức năng cơ bản của nó thôi.

5.5.5.1 Các tính chất vật lý

802.11 được thiết kế để chạy trên ba phương tiện vật lý khác nhau - hai dựa trên sóng radio phổ rộng và một dựa trên tia hồng ngoại được khuếch tán. Phiên bản chạy trên sóng radio hiện đang chạy với tốc độ 11 Mbps, nhưng có thể sớm đạt được tốc độ 54 Mbps.

Ý tưởng đằng sau khái niệm phổ tần rộng là nhằm trải rộng tín hiệu lên trên một băng tần rộng hơn so với bình thường, vì thế có thể giảm thiểu tác động của sự giao thoa tín hiệu với các thiết bị khác. Ví dụ, kỹ thuật "*nhảy tần số" (frequency hopping)* là một kỹ thuật sử dụng phổ tần rộng, nó xoay quanh việc gởi tín hiệu qua một dãy tần số ngẫu nhiên; nghĩa là lần đầu sẽ gởi trên một tần số, lần hai gởi trên tần số khác, lần thứ ba và vân vân. Dãy tần số này không thật sự là ngẫu nhiên mà được tính toán một cách có giải thuật bởi một bộ sinh số ngẫu nhiên. Bên nhận sẽ dùng cùng một giải thuật như bên gởi và do đó có thể nhảy qua các tần số khác nhau đồng bộ với bên gởi để nhận chính xác khung thông tin.

Một kỹ thuật sử dụng phổ tần rộng khác, được gọi là *"dãy trực tiếp" (direct sequence)*, cũng đạt được cùng một hiệu quả bằng cách thể hiện một bit trong khung thành nhiều bit trong tín hiệu truyền đi. Với mỗi bit bên gởi muốn truyền đi, nó thực ra sẽ gởi một chuỗi bit là kết quả của phép toán exclusive-OR của bit đó với một chuỗi n bit ngẫu nhiên. Cũng như trong frequency hopping, chuỗi n bit này được sinh ra bởi một bộ sinh số ngẫu nhiên và được hiểu bởi cả hai bên gởi và nhận. Các giá trị được truyền đi, được gọi là "mã cắt lát" n bit (n-bit chipping code), sẽ rải tín hiệu trên một dãi tần rộng hơn gấp n lần so với dãi tần mà thông thường khung cần để được truyền đi. H5.35 là một ví dụ về mã cắt lát 4 bit.



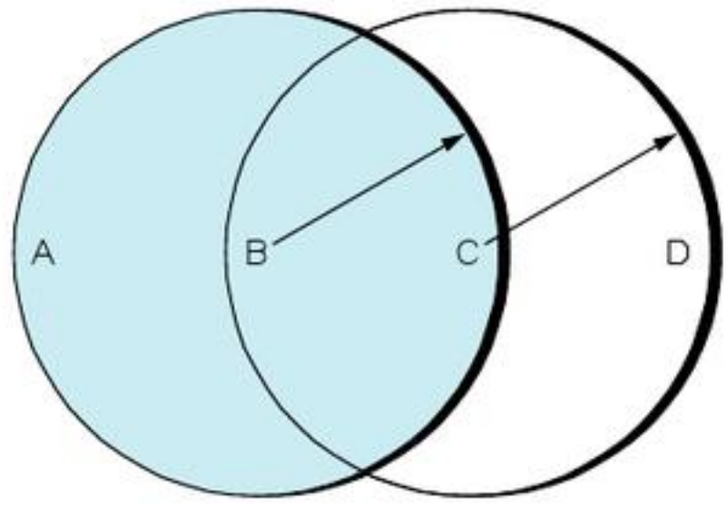
H5.36 Ví dụ về mã cắt lát 4 bit.

802.11 định nghĩa một lớp vật lý sử dụng cơ chế "nhảy tần số" (trên 79 dải tần có độ rộng 1-Mhz), lớp vật lý thứ hai sử dụng "dãy trực tiếp" (sử dụng dãy cắt lát 11 bit). Cả hai chuẩn đều chạy trên sóng điện từ băng tần 2.4 GHz. Trong cả hai trường hợp, việc trải rộng phổ truyền đều có điểm

thú vị là làm cho tín hiệu giống như nhiễu đối với bất kỳ bên nhận nào không biết chuỗi ngẫu nhiên giả.
Chuẩn vật lý thứ ba của 802.11 dựa trên tín hiệu hồng ngoại. Tín hiệu lưu thông sẽ được khuếch tán, nghĩa là bên gởi và bên nhận không cần phải hướng vào nhau và không cần tầm nhìn rõ ràng. Kỹ thuật này có phạm vi hoạt động khoảng 10 m và bị giới hạn chỉ trong các tòa nhà mà thôi.

5.5.5.2 Tránh đụng độ (Collision Avoidance)

Thoạt nhìn, có vẻ như giao thức không dây cũng tuân thủ cùng giải thuật như ở Ethernet: đợi cho đến khi đường truyền rỗi thì mới truyền, lùi lại nếu có đụng độ. 802.11 cũng làm tương tự như vậy. Tuy nhiên, vấn đề phức tạp hơn trong hệ thống mạng không dây, bởi vì không phải tất cả các nút đều có thể thấy nhau!



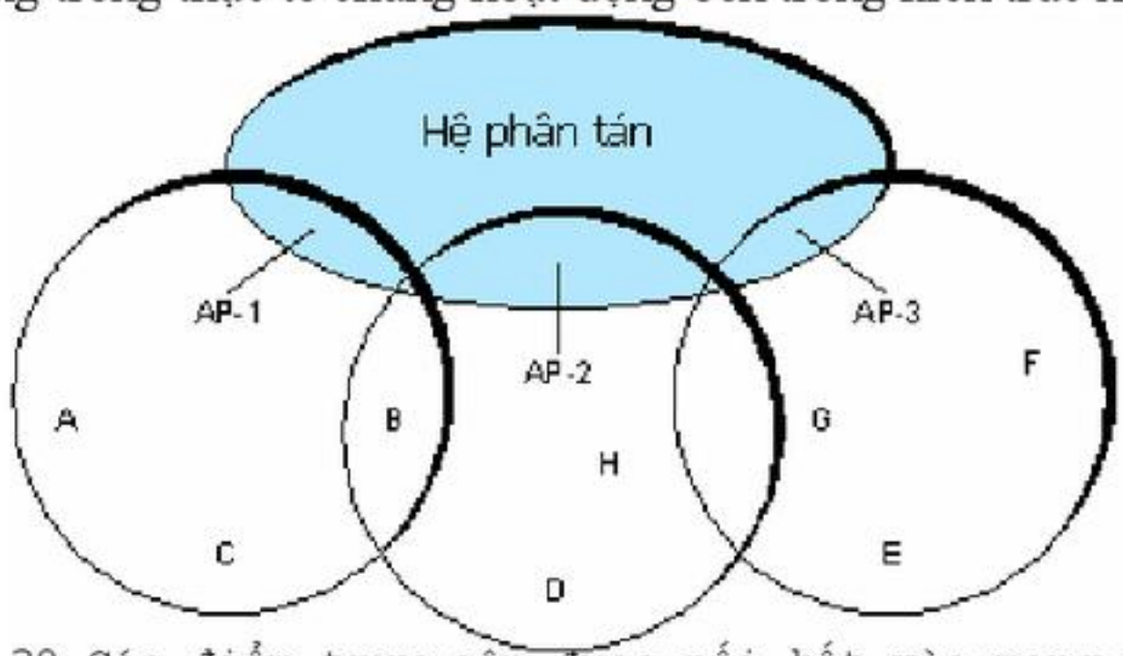
H5.37 Ví dụ về mạng không dây

Xét tình huống được chỉ ra trong H5.37, mỗi trạm chỉ có thể gởi và nhận tín hiệu đến hai nút kề trái và phải của nó. Ví dụ nút B có thể trao đổi các khung dữ liệu với nút A và C nhưng không thể phát tín hiệu đến D được, trong khi C chỉ có thể trao đổi với B và D mà không thể với A. (Tầm phủ sóng của A và D không được vẽ trong hình). Giả sử cả A và C cùng muốn nói giao tiếp với B và do đó chúng cùng gởi khung đến B. A và C không biết gì về nhau do sóng không thể đi xa như vậy. Hai khung này sẽ bị đụng độ tại B, nhưng không như trong Ethernet, không ai trong A và C hay biết gì về sự đụng độ này. A và C được gọi là các nút ẩn (hiden nodes) đối với nhau.
Một vấn đề liên quan nữa, gọi là vấn đề "nút bị phơi trần" (exposed node problem), phát sinh trong tính huống sau: Giả sử B đang gởi cho A, nút C nhận biết được tình huống này vì nó nghe được việc truyền của B. Và sẽ là sai lầm nếu C kết luận là nó không thể truyền đến bất kỳ nút nào khác do nó nghe thấy B đang truyền. Ví dụ, giả sử C muốn truyền cho D thì việc này không gây ảnh hưởng gì vì việc truyền từ C đến D không can dự vào khả năng nhận tín hiệu của A từ B. (Chỉ gây ảnh hưởng khi A truyền cho B, nhưng trong trường hợp này B đang truyền).
802.11 giải quyết hai vấn đề trên bằng một giải thuật gọi là "Đa truy cập với cơ chế tránh đụng độ" – Multiple Access with Collision Avoidance (MACA). Ý tưởng là cho hai bên gởi và nhận trao đổi những khung điều khiển với nhau trước khi thật sự truyền dữ liệu. Việc trao đổi này dụng ý là để thông báo cho các trạm lân cận rằng một phiên truyền dữ liệu sắp xảy ra. Cụ thể giải thuật như sau: Bên gởi gởi một khung "yêu cầu gởi" – Request To Send (RTS) đến bên nhận; khung RTS có chứa một trường, trong đó chỉ ra bên gởi muốn giữ kênh truyền bao lâu (nghĩa là nó chỉ ra chiều dài của khung cần gởi). Bên nhận sẽ trả lời bằng khung "sẵn sàng nhận" – Clear To Send (CTS); khung này sẽ lặp lại trường chiều dài khung như phía bên gởi. Bất kỳ trạm nào thấy khung CTS cũng hiểu được rằng trạm bên nhận ở gần nó, cho nên nó sẽ không thể gởi khung trong khoảng thời gian được chỉ ra trong CTS. Còn những trạm thấy khung RTS nhưng không thấy khung CTS đâu thì chắc chắn là không ở gần bên nhận, do đó chúng có quyền tự do gởi khung đi.
Có thêm hai chi tiết nữa để hoàn thiện bức tranh về Wireless LAN. Thứ nhất, bên nhận gởi một khung báo nhận (ACK) cho bên gởi sau khi đã nhận thành công một khung dữ liệu. Tất cả các nút khác phải chờ khung ACK này đi qua trước khi thử truyền khung của mình. (Khung ACK này không có trong phiên bản MACA gốc, thay vào đó nó được đề nghị trong một phiên bản mở rộng được gọi là MACW: MACA for Wireless LANs). Thứ hai, trường hợp có hơn hai trạm phát hiện kênh truyền rỗi, chúng sẽ gởi khung RTS cùng lúc, dẫn đến các khung này bị va chạm với nhau. 802.11 không có hỗ trợ cơ chế phát hiện đụng độ, thay vào đó, các trạm nhận ra sự đụng độ khi

chúng không nhận được các khung CTS sau một khoảng thời gian nào đó. Nếu thế, mỗi trạm sẽ chờ đợi sau một khoảng thời gian ngẫu nhiên nào đó trước khi thử gởi khung RTS lần nữa. Khoảng thời gian mà một trạm chờ để thử lại được định nghĩa giống như thuật toán back-off trong Ethernet.

5.5.5.3 Hệ thống phân tán

Như những gì đã trình bày, 802.11 có thể thích hợp với cấu hình mạng dạng ad-hoc, trong đó các nút có thể hoặc không thể giao tiếp với những nút còn lại, tùy thuộc vào khoảng cách giữa chúng là gần hay xa. Ngoài ra, do một trong những thuận lợi của mạng không dây là các nút có thể tự do di chuyển do chúng không bị trói buộc bởi hệ thống dây nối, tập các nút có thể thấy nhau trực tiếp là thay đổi theo thời gian. Để giúp giải quyết vấn đề di động và nối kết một phần này, 802.11 định nghĩa thêm một kiến trúc trên một tập các nút. Các nút có quyền tự do giao tiếp trực tiếp với nhau như đã trình bày, nhưng trong thực tế chúng hoạt động bên trong kiến trúc này.



H5.38 Các điểm truy cập được nối kết vào mạng phân tán

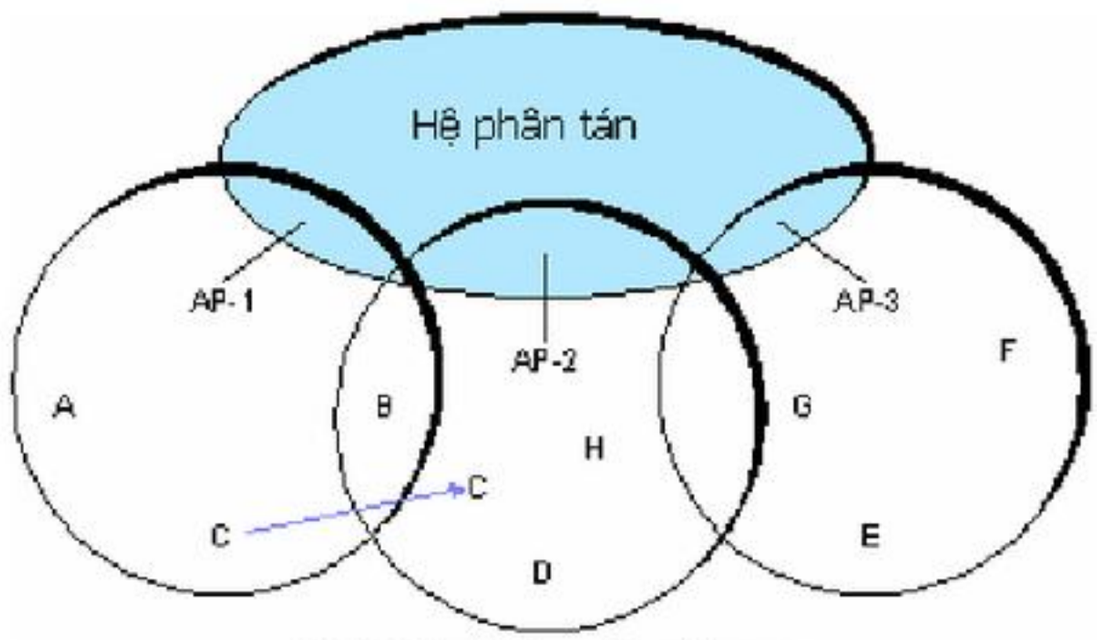
Thay vì tất cả các nút được tạo ra như nhau, một số nút được phép đi lang thang (đó là máy laptop của bạn) và một số được nối kết tới một hạ tầng mạng có nối dây. Các trạm nối kết có dây được gọi là các điểm truy cập - "access point" (AP) và chúng được nối với nhau bằng cái gọi là hệ thống phân tán. H5.38 mô phỏng một hệ thống phân tán nối kết ba điểm truy cập, mỗi điểm truy cập phục vụ các nút di động trong khu vực mình phụ trách. Mỗi khu vực này giống như một cell trong hệ thống điện thoại di động, với AP hoạt động giống như Base station.

Mặc dù hai nút có thể giao tiếp trực tiếp với nhau nếu chúng ở trong tầm với của nhau, tuy nhiên ý tưởng ở trong kiến trúc này là: mỗi nút sẽ kết hợp với một điểm truy cập của nó. Ví dụ, để nút A giao tiếp được với nút E, đầu tiên A gởi khung của nó đến điểm truy cập AP-1, AP-1 sẽ gởi khung qua hệ thống phân tán đến AP-3, rồi AP-3 sẽ phân phát khung đến E. Chỉ ra AP-1 làm cách nào để chuyển khung đến AP-3 là nằm ngoài phạm vi của 802.11, một giao thức cầu nối (sẽ được nghiên cứu ở tầng mạng) có thể được sử dụng để làm nhiệm vụ này. Những gì 802.11 có thể làm là xác định cách thức các nút chọn ra AP của chúng, và thú vị hơn nữa là làm sao giao thức này hoạt động được trong tình cảnh các nút di chuyển từ cell này đến cell khác.

Kỹ thuật để chọn ra một AP được gọi là kỹ thuật "quét" (scanning) và nó xoay quanh bốn bước sau:

1. Nút gởi một khung **Probe** (thăm dò).
2. Tất cả điểm truy cập (AP) trong tầm với của nút sẽ trả lời bằng một khung **Probe Response** (trả lời thăm dò).
3. Nút sẽ chọn một trong các điểm truy cập trên và gởi đến điểm truy cập đó một khung **Association Request** (yêu cầu gia nhập).
4. Điểm truy cập trả lời bằng một khung **Association Reponse** (trả lời cho yêu cầu gia nhập).

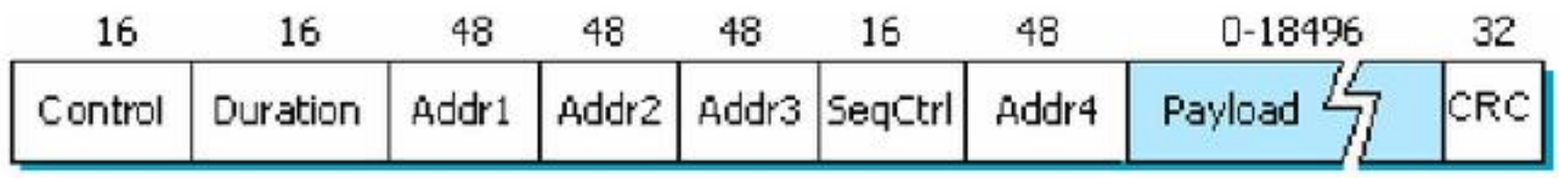
Một nút tiến hành giao thức này khi nó lần đầu tham gia vào mạng hoặc nó không hài lòng với AP hiện tại. Điều này có thể xảy ra, ví dụ như vì tín hiệu từ AP hiện tại yếu dần do nút càng di chuyển xa AP. Mỗi khi nút kiếm được AP mới, AP mới sẽ nhắc nhở AP cũ về sự thay đổi này.



H5.39 Sự di động của nút mạng

Lấy ví dụ như trong H5.38, khi mà nút C di chuyển từ cell được phục vụ bởi AP-1 sang cell được phục vụ bởi AP-2. Khi C di chuyển, nó gởi khung **Probe** đến AP-2 và được AP-2 trả lời bằng khung **Probe Response.** Tại thời điểm đó, C thích AP-2 hơn AP-2, do đó nó gia nhập vào điểm truy cập này.
Cơ chế vừa được mô tả được gọi là cơ chế "quét chủ động" – active scanning, do nút chủ động dò tìm điểm truy cập. Các AP cũng thường xuyên gởi khung **Beacon** (đèn hiệu) để quảng cáo cho khả năng của mình, bao gồm tốc độ truyền được điểm truy cập hỗ trợ. Cơ chế này được gọi là "quét thụ động" – passive scanning, và một nút có thể chuyển sang điểm truy cập này đơn giản bằng cách gởi môt khung **Association Request** ngược lại AP.

5.5.5.4 Khuôn dạng khung



H5.40 Khuôn dạng khung 802.11

Hầu hết các thông tin trong khung 802.11, như được vẽ trong H5.40, là đều như chúng ta mong muốn. Khung bao gồm địa chỉ nguồn và đích, mỗi cái dài 48 bits; dữ liệu tối đa 2312 bytes; và phần kiểm tra lỗi CRC 32 bits. Trường Control chứa ba trường con: Trường Type dài 6 bits – dùng để chỉ ra khung là khung dữ liêu, hay là khung RTS, hay là CTS, hoặc cũng được sử dụng bởi giải thuật quét; một cặp trường mỗi trường dài 1 bit gọi là ToDS và FromDS, sẽ được giải thích ngay sau đây.
Điều kỳ dị trong khung 802.11 là nó chứa đến bốn địa chỉ. Cách thức thông dịch các địa chỉ này lại phụ thuộc vào việc thiết đặt các bít ToDS và FromDS trong trường Control. Điều này là để tính đến khả năng khung phải được chuyển tiếp qua hệ thống phân tán, có nghĩa là bên gởi nguyên thủy không nhất thiết phải là nút phát khung gần đây nhất. Cũng tương tự đối với địa chỉ bên nhận. Trong trường hợp đơn giản nhất, khi một nút gởi khung trực tiếp đến nút kia, cả hai bit DS đều có giá trị 0, Addr1 chứa địa chỉ nút đích, Addr2 chứa địa chỉ nút nguồn. Trong trường hợp phức tạp nhất, cả hai bit DS mang giá trị 1, chỉ ra rằng khung thông tin đi từ nút không dây qua hệ thống phân tán và rồi từ hệ thống phân tán đến nút không dây bên đích. Với cả hai bit DS được đặt, Addr1 định danh đích cuối cùng, Addr2 định danh nút gởi tạm thời (là nút đã chuyển khung từ hệ thống phân tán đến đích cuối cùng), Addr3 định danh nút đích tạm thời (là nút đã nhận khung từ nút không dây và trung chuyển khung xuyên qua hệ thống phân tán), Addr4 định danh nút gởi nguyên thủy. Trong ví dụ H5.37, Addr1=E, Addr2=AP-3, Addr3=AP-1, Addr4=A.

Chương 6: Tầng mạng (Network Layer)

Mục đích

Chương này nhằm giới thiệu cho người đọc những nội dung sau:

- Vai trò của router trong việc xây dựng các liên mạng có phạm vi rộng và không đồng nhất về chuẩn của các mạng cục bộ thành phần
- Các dịch vụ mà tầng mạng phải cung cấp cho tầng vận chuyển
- Cơ chế hoạt động của router
- Các vấn đề liên quan đến giải thuật chọn đường cho các router
- Giới thiệu về bộ giao thức liên mạng IP

Yêu cầu

Sau khi học xong chương này, người đọc phải có được những khả năng sau:

- Mô tả được sơ đồ tổng quát của một liên mạng ở tầng 3 và vai trò của router trong liên mạng này
- Trình bày được các dịch vụ mà tầng mạng phải cung cấp cho tầng vận chuyển
- Giải thích cơ chế truyền tải thông tin theo kỹ thuật truyền tải lưu và chuyển tiếp của các router
- Giải thích được ý nghĩa của bảng chọn đường trong router
- Phân biệt được các loại giải thuật chọn đường khác nhau
- Cài đặt được các giải thuật chọn đường Dijkstra, Ford-Fulkerson, Distance Vector, Link state
- Nêu lên được các phương pháp để chống tắc nghẽn trên mạng diện rộng
- Biết cách thiết lập sơ đồ đánh địa chỉ IP cho mạng
- Thực hiện được việc phân mạng con theo những yêu cầu khác nhau theo cả hai phương pháp : Phân lớp hoàn toàn và Vạch đường liên miền không phân lớp
- Xây dựng được bảng chọn đường thủ công cho các router trong mạng IP
- Nêu lên được ý nghĩa của các giao thức ARP, RARP và ICMP trong bộ giao thức IP

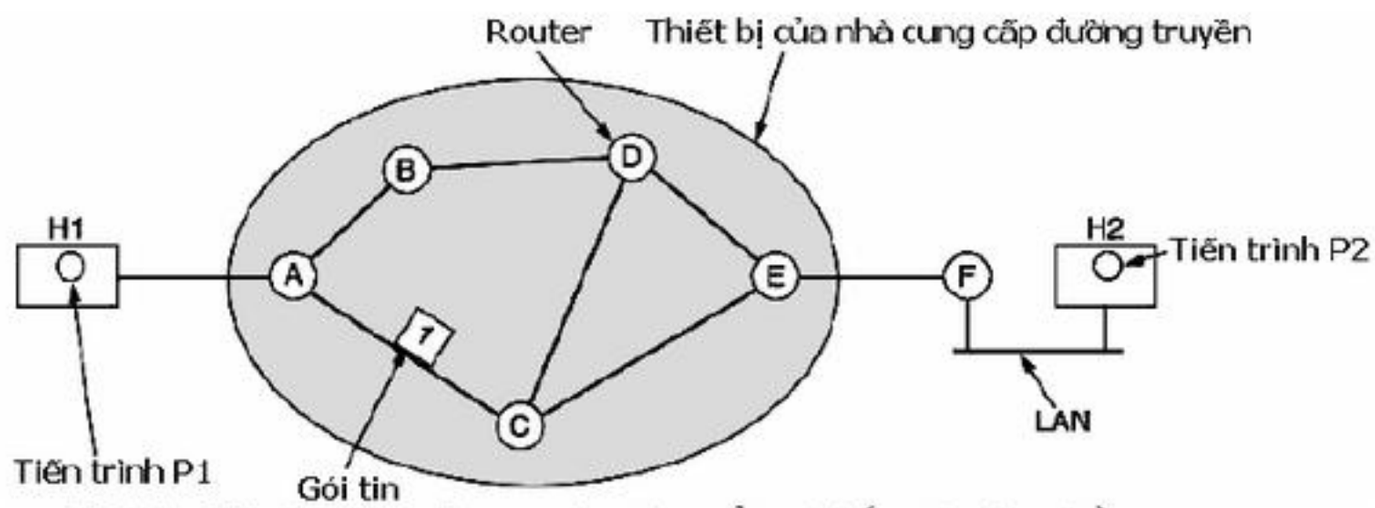
6.1 Giới thiệu

Chúng ta đã xem xét cách thức xây dựng và vận hành của các mạng đơn lẻ sử dụng các nối kết điểm điểm, các đường truyền chia sẻ và các bộ hoán chuyển (switch). Vấn đề phát sinh là có nhiều người muốn xây dựng hệ thống mạng riêng của họ theo nhiều kỹ thuật khác nhau nhưng lại muốn giao tiếp với nhau mà không quan tâm rằng họ đang hoạt động trên các hệ thống không đồng nhất. Chương này sẽ trình bày về cách thức để nối kết những mạng không đồng nhất lại với nhau.
Có hai vấn đề quan trọng cần phải quan tâm khi nối kết các mạng: tính không đồng nhất (heterogeneity) và phạm vi (scale) khác nhau của chúng. Giải thích một cách đơn giản, tính không đồng nhất là khi người dùng trên hai mạng khác kiểu nhau muốn giao tiếp với nhau. Phức tạp hơn một chút, ta có thể thấy việc nối kết các host trên các mạng khác nhau có thể sẽ đòi hỏi việc duyệt qua nhiều mạng trung gian, mà các mạng trung gian này lại có thể có kiểu khác nhau. Chúng có thể là mạng Ethernet, Token Ring hay mạng dạng điểm nối điểm, hoặc nhiều kiểu mạng hoán chuyển (switch) khác nhau, và chúng lại sử dụng các phương thức đánh địa chỉ riêng, các phương pháp truy cập đường truyền riêng và cả mô hình dịch vụ riêng nữa. Thách thức đối với vấn đề không đồng nhất là làm sao cung cấp cho người dùng một dịch vụ nối kết host-host dễ hiểu xuyên qua mớ hỗn độn các mạng không đồng nhất. Để hiểu về vấn đề phạm vi mạng, ta lấy một ví dụ có giá trị là sự phát triển của mạng Internet, mạng có tốc độ phát triển gần gấp đôi sau mỗi năm trong vòng 20 năm qua. Kiểu phát triển chóng mặt này buộc chúng ta phải đối mặt với nhiều thách thức. Một trong số đó là việc vạch đường: Làm sao để tìm ra một đường đi hữu hiệu xuyên qua một mạng gồm cả triệu nút mạng? Thêm một vấn đề có liên quan đến vạch đường là phương pháp đánh địa chỉ, là cách gán cho mỗi nút trên mạng một định danh duy nhất.
Tầng mạng có nhiệm vụ đưa các gói tin từ máy gởi qua các chặn đường để đến được máy nhận. Để đến được đích đến, gói tin có thể phải đi từng bước một qua nhiều router trung gian. Điều này thì trái ngược với tầng liên kết dữ liệu vốn chỉ chịu trách nhiệm truyền tải các khung đi từ đầu này đến đầu kia của một kênh truyền vật lý.
Để thực hiện được nhiệm vụ này, tầng mạng phải biết được hình trạng của mạng đường trục (subnet) và chọn đường thích hợp để cho gói tin đi. Nó phải chú ý đến việc chọn đường sao cho tránh được tình trạng tắc nghẽn trên một số đường truyền và router trong khi số khác thì đang rãnh rỗi.

6.2 Các vấn đề liên quan đến việc thiết kế tầng mạng

6.2.1 Kỹ thuật hoán chuyển lưu và chuyển tiếp (Store-and-Forward Switching)

Xét một liên mạng như hình dưới đây



H6.1 Kỹ thuật lưu và chuyển tiếp trên tầng mạng

Trong đó các router nằm trong hình oval được nối lại với nhau bằng các đường truyền theo kiểu điểm nối điểm được gọi là các thiết bị của nhà cung cấp đường truyền (Carrier's equipment). Các thiết bị nằm bên ngoài hình oval được gọi là các thiết bị của khách hàng (Customer's Equipment).

Máy tính H1 được nối trực tiếp vào router A của nhà cung cấp đường truyền bằng một đường nối kết thường trực (lease line). Máy H2 nối kết vào một mạng LAN cục bộ. Trong mạng LAN có router F thuộc sở hữu của khách hàng. F được nối với router E của nhà cung cấp cũng bằng một đường nối kết thường trực.
Cho dù cách thức nối kết vào mạng của các máy tính có thể khác nhau như trường hợp máy H1 và H2, nhưng cách thức các gói tin của chúng được truyền đi đều giống nhau. Một máy tính có một gói tin cần truyền đi sẽ gởi gói tin đến router gần nó nhất, có thể là router trên LAN của nó hoặc router của nhà cung cấp đường truyền. Gói tin được lưu lại ở đó và được kiểm tra lỗi. Kế đến gói tin sẽ được chuyển đến một router kế tiếp trên đường đi đến đích của gói tin. Và cứ tiếp tục như thế cho đến khi đến được máy nhận gói tin. Đây chính là kỹ thuật lưu và chuyển tiếp.

6.2.2 Các dịch vụ cung cấp cho tầng vận chuyển

Các dịch vụ tầng mạng cung cấp cho tầng vận chuyển cần được thiết kế hướng đến các mục tiêu sau:

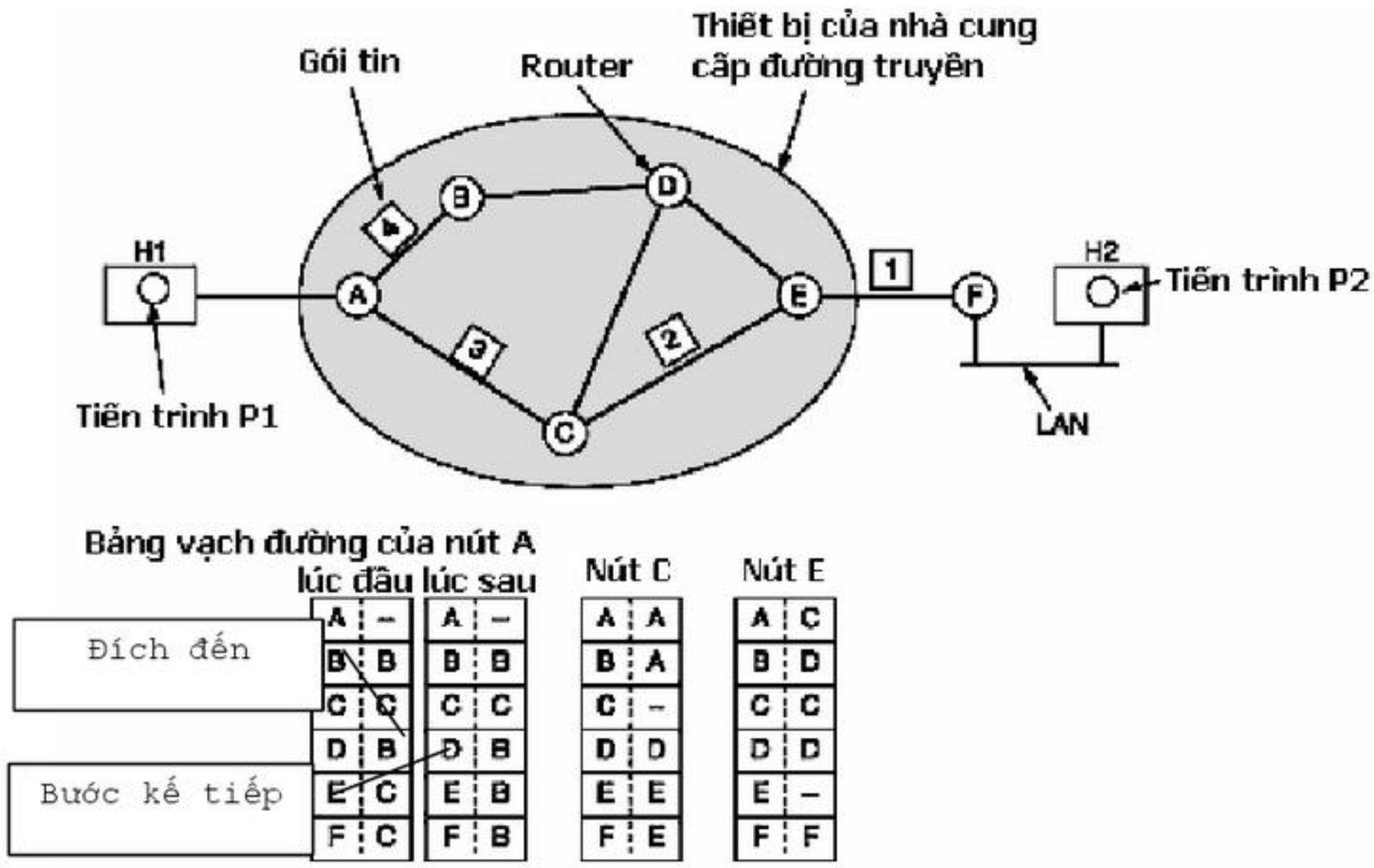
Các dịch vụ này cần nên độc lập với kỹ thuật của các router.

1. Tầng vận chuyển cần được độc lập với số lượng, kiểu và hình trạng của các router hiện hành.
2. Địa chỉ mạng cung cấp cho tầng vận chuyển phải có sơ đồ đánh số nhất quán cho dù chúng là LAN hay WAN.

Tầng mạng cung cấp hai dịch vụ chính là Dịch vụ không nối kết (Connectionless Service) và Dịch vụ định hướng nối kết (Connection – Oriented Service).
Trong dịch vụ không nối kết, các gói tin được đưa vào subnet một cách riêng lẽ và được vạch đường một cách độc lập nhau. Không cần thiết phải thiết lập nối kết trước khi truyền tin. Các gói tin trong trường hợp này được gọi là thư tín (Datagram) và subnet được gọi là Datagram Subnet. Ngược lại trong dịch vụ định hướng nối kết, một đường nối kết giữa bên gởi và bên nhận phải được thiết lập trước khi các gói tin có thể được gởi đi. Nối kết này được gọi là mạch ảo (Virtual Circuit) tương tự như mạch vật lý được nối kết trong hệ thống điện thoại và subnet trong trường hợp này được gọi là virtual circuit subnet.

6.2.2.1 Cài đặt dịch vụ không nối kết (Implementation of Connectionless Service)

Xét hệ thống mạng như hình H6.2. Giả sử rằng quá trình P1 có nhiều thông điệp cần gởi cho quá trình P2. Khi đó P1 sẽ gởi các thông điệp này cho tầng vận chuyển và yêu cầu tầng vận chuyển truyền sang quá trình P2 trên máy tính H2. Tầng vận chuyển sẽ gắn thêm tiêu đề (header) của nó vào thông điệp và chuyển các thông điệp xuống tầng mạng.



H6.2 Hoạt động của Datagram subnet

Giả sử rằng thông điệp gởi đi thì lớn gấp 4 lần kích thước tối đa của một gói tin, vì thế tầng mạng phải chia thông điệp ra thành 4 gói tin 1,2,3 và 4, và lần lượt gởi từng gói một đến router A bằng một giao thức điểm nối điểm như PPP chẳng hạn.

Mỗi router có một bảng thông tin cục bộ chỉ ra nơi nào có thể gởi các gói tin để có thể đến được những đích đến khác nhau trên mạng. Mỗi mục từ của bảng chứa 2 thông quan trọng nhất đó là Đích đến (Destination) và ngỏ ra kế tiếp (Next Hop) cần phải chuyển gói tin đến để có thể đến được đích đến này. Ta gọi là bảng chọn đường (Routing Table).

Ví dụ

- Lúc khởi đầu, router A có bảng chọn đường như hình H6.2 (lúc đầu). Khi gói tin 1,2 và 3 đến router A, nó được lưu tạm thời để kiểm tra lỗi. Sau đó chúng được chuyển tiếp sang router C vì theo thông tin trong bảng chọn đường của A. Gói tin 1 sau đó tiếp tục được chuyển đến E và kế đến là F. Sau đó nó được gói lại trong một khung của tầng liên kết dữ liệu và được chuyển đến máy H2 bởi mạng LAN. Các gói tin 2 và 3 cũng có cùng đường đi tương tự.
- Sau đó, do một số sự cố về đường truyền, router A cập nhật lại bảng chọn đường của mình như hình H6.2(lúc sau). Khi đó gói tin số 4 đến router A, nó sẽ chuyển gói tin này sang B để có thể đi được đến H2.

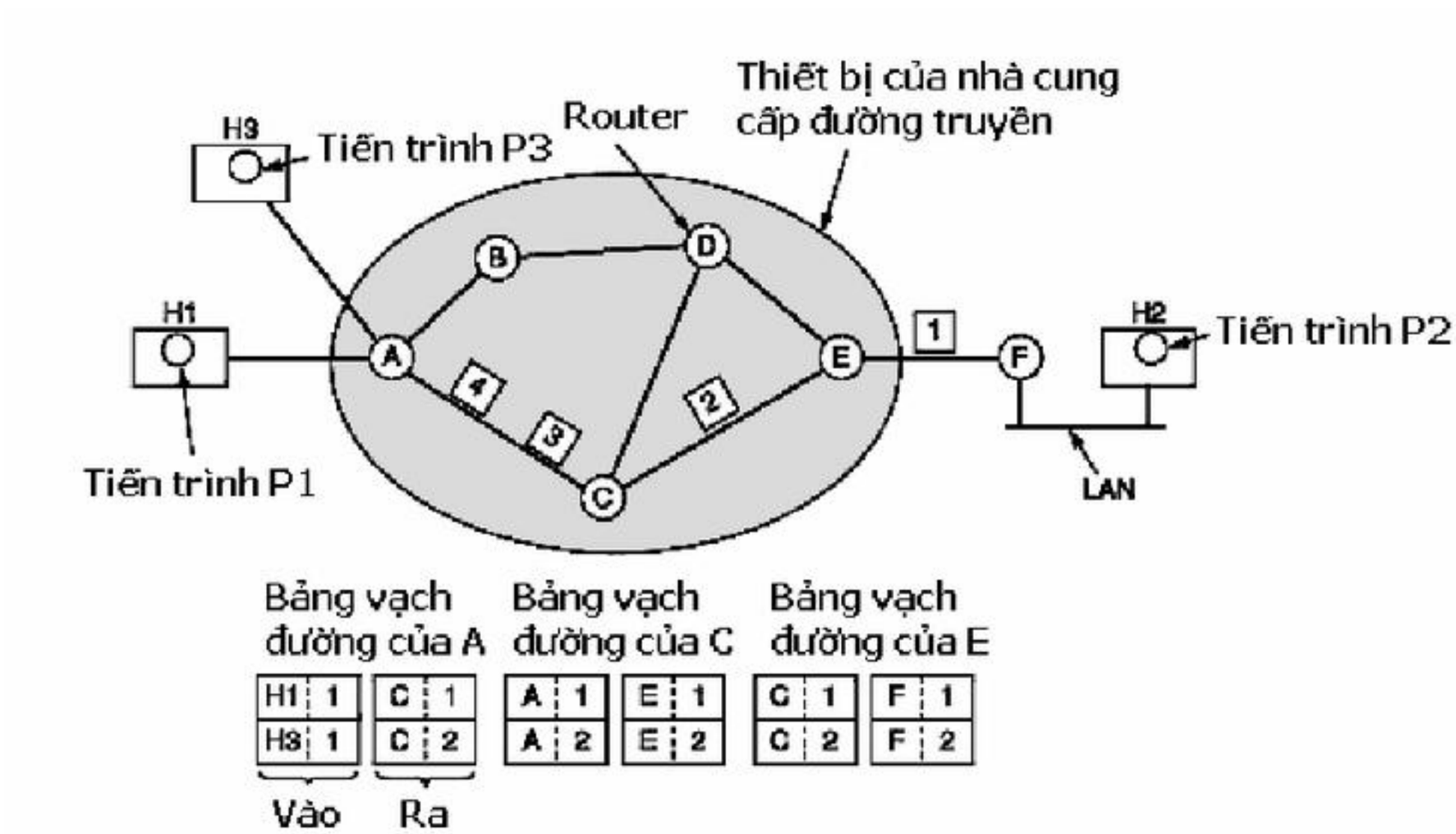
Giải thuật chịu trách nhiệm quản lý thông tin trong bảng chọn đường cũng như thực hiện các quyết định về chọn đường được gọi là **Giải thuật chọn đường** (Routing algorithm).

6.2.2.2 Cài đặt dịch vụ định hướng nối kết (Connection – Oriented Service)

Đối với dịch vụ nối kết định hướng chúng ta cần một mạch ảo trên subnet. Mục đích của việc sử dụng mạch ảo là để tránh phải thực hiện việc chọn lại đường đi mới cho mỗi gói tin gởi đến cùng một đích.

Khi một nối kết được thực hiện, một đường đi từ máy tính gởi đến máy tính nhận được chọn như là một phần của giai đoạn thiết lập nối kết (Connection setup) và được lưu trong bảng chọn đường của các router nằm trên đường đi. Khi nối kết kết thúc, mạch ảo bị xóa.

Với dịch vụ định hướng nối kết, mỗi gói tin có mang một số định dạng để xác định mạch ảo mà nó thuộc về.



H6.3 Hoạt động của Datagram subnet

Như hình H6.3, máy tính H1 thực hiện một nối kết với máy tính H2 qua nối kết số 1. Nối kết này được ghi nhận trong mục từ đầu tiên trong bảng chọn đường của các router.
Dòng đầu tiên trong bảng chọn đường của router A nói rằng: những gói tin mang số nhận dạng nối kết số 1 đến từ máy H1 phải được gởi sang router C với số nhận dạng nối kết là 1. Tương tự, cho các mục từ đầu tiên của router C và E.
Điều gì xảy ra nếu máy tính H3 muốn nối kết với máy tính H2. Nó chọn số nhận dạng nối kết là 1, vì đây là nối kết đầu tiên đối với H3, và yêu cầu subnet thiết lập mạch ảo. Điều này đã làm cho các router phải thêm vào mục từ số 2 trong bảng chọn đường. Đối với router A, số nhận dạng nối kết với H3 là 1, trùng với nối kết với H1, không làm router A lẫn lộn vì A có thêm thông tin máy gởi là H1 hay H3. Tuy nhiên, đối với các router C, E và F thì không thể phân biệt được đâu là nối kết của H1 và đâu là nối kết của H3 nếu sử dụng số nhận dạng nối kết là 1 cho cả 2 nối kết. Chính vì thế A đã gán một số nhận dạng khác, là số 2, cho các gói tin gởi đến C có nguồn gốc từ H3.

6.2.2.3 So sánh giữa Datagram subnet và Virtual-Circuit subnet

Bảng sau so sánh điểm mạnh và điểm yếu của 2 loại dịch vụ không nối kết và định hướng nối kết:

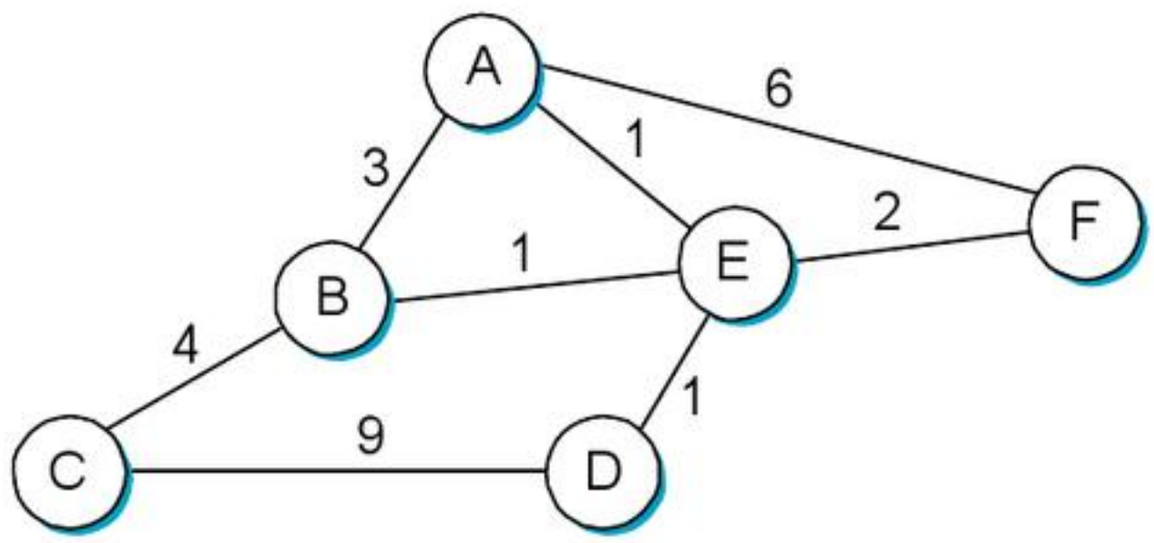
Vấn đề	Datagram Subnet	Circuit Subnet
Thiết lập nối kết	Không cần	Cần thiết
Đánh địa chỉ	Mỗi gói tin chứa đầy đủ địa chỉ gởi và nhận	Mỗi gói tin chỉ chứa số nhận dạng nối kết có kích thước nhỏ.
Thông tin trạng thái	Router không cần phải lưu giữ thông tin trạng thái của các nối kết	Mỗi nối kết phải được lưu lại trong bảng chọn đường của router.
Chọn đường	Mỗi gói tin có đường đi khác nhau	Đường đi được chọn khi mạch ảo được thiết lập, sau đó tất cả các gói tin đều đi trên đường này.
Ảnh hưởng khi router bị hỏng	Không bị ảnh hưởng, ngoại trừ gói tin đang trên đường truyền bị hỏng	Tất cả các mạch ảo đi qua router bị hỏng đều bị kết thúc
Chất lượng dịch vụ	Khó đảm bảo	Có thể thực hiện dễ dàng nếu có đủ tài nguyên gán trước cho từng nối kết

Điều khiển tắc nghẽn	Khó điều khiển	Có thể thực hiện dễ dàng nếu có đủ tài nguyên gán trước cho từng nối kết

6.3 Giải thuật chọn đường

6.3.1 Giới thiệu

Vạch đường về bản chất là một bài toán trong lý thuyết đồ thị. Hình 6.4 thể hiện một đồ thị biểu diễn cho một mạng.



H6.4 Mạng được biểu diễn như một đồ thị

Các nút trong đồ thị (được đánh dấu từ A đến F) có thể là các host, switch, router hoặc là các mạng con. Ở đây chúng ta tập trung vào một trường hợp các nút là các router. Các cạnh của đồ thị tương ứng với các đường nối kết mạng. Mỗi cạnh có một chi phí đính kèm, là thông số chỉ ra cái giá phải trả khi lưu thông trên nối kết mạng đó.

Vấn đề cơ bản của việc vạch đường là tìm ra đường đi có chi phí thấp nhất giữa hai nút mạng bất kỳ, trong đó chi phí của đường đi được tính bằng tổng chi phí khi đi qua tất cả các cạnh làm thành đường đi đó. Nếu không có một đường đi giữa hai nút, thì độ dài đường đi giữa chúng được xem như bằng vô cùng.

6.3.2 Mục tiêu của giải thuật chọn đường

- Xác định đường đi nhanh chóng, chính xác.
- Khả năng thích nghi được với những thay đổi về hình trạng mạng.
- Khả năng thích nghi được với những thay đổi về tải đường truyền.
- Khả năng tránh được các nối kết bị tắt nghẽn tạm thời
- Chi phí tính toán để tìm ra được đường đi phải thấp

6.3.3 Phân loại giải thuật chọn đường

Giải thuật chọn đường có thể được phân thành những loại sau:

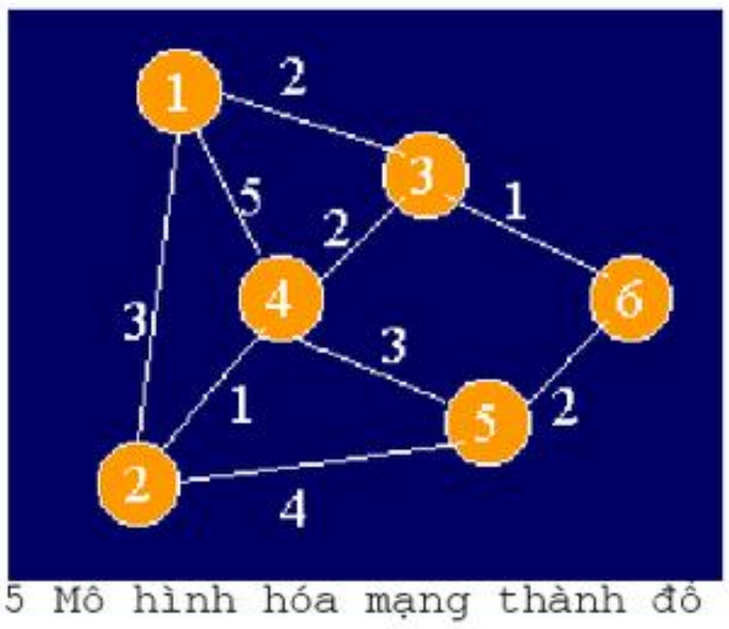
- Chọn đường tập trung (Centralized routing): Trong mạng có một Trung tâm điều khiển mạng (Network Control Center) chịu trách nhiệm tính toán và cập nhật thông tin về đường đi đến tất cả các điểm khác nhau trên toàn mạng cho tất cả các router.
- Chọn đường phân tán (Distributed routing): Trong hệ thống này, mỗi router phải tự tính toán tìm kiếm thông tin về các đường đi đến những điểm khác nhau trên mạng. Để làm được điều này, các router cần phải trao đổi thông tin quan lại với nhau.
- Chọn đường tĩnh (Static routing): Trong giải thuật này, các router không thể tự cập nhật thông tin về đường đi khi hình trạng mạng thay đổi. Thông thường nhà quản mạng sẽ là người cập nhật thông tin về đường đi cho router.
- Chọn đường động (Dynamic routing): Trong giải thuật này, các router sẽ tự động cập nhật lại thông tin về đường đi khi hình trạng mạng bị thay đổi.

6.3.4 Các giải thuật tìm đường đi tối ưu

Đường đi tối ưu từ A đến B là đường đi "ngắn" nhất trong số các đường đi có thể. Tuy nhiên khái niệm "ngắn" nhất có thể được hiểu theo nhiều ý nghĩa khác nhau tùy thuộc vào đơn vị dùng để đo chiều dài đường đi. Đối với các router, các đại lượng sau có thể được sử dụng để đo độ dài đường đi:

- Số lượng các router trung gian phải đi qua (HOP)
- Độ trì quản trung bình của các gói tin
- Cước phí truyền tin

Mỗi giải thuật chọn đường trước tiên phải chọn cho mình đơn vị đo chiều dài đường đi.
Để xác định được đường đi tối ưu, các giải thuật chọn đường sử dụng phương pháp đồ thị để tính toán. Trước tiên, nó mô hình hóa hình trạng mạng thành một đồ thị có các đặc điểm như sau:



H6.5 Mô hình hóa mạng thành đồ thị

- Nút là các router.
- Cạnh nối liền 2 nút là đường truyền nối hai router.
- Trên mỗi cạnh có giá đó là chiều dài đường đi giữa 2 router thông qua đường truyền nối hai router .
- Chiều dài đường đi từ nút A đến nút B là tổng tất cả các giá của các cạnh nằm trên đường đi. Nếu không có đường đi giữa 2 router thì xem như giá là vô cùng.

Trên đồ thị này sẽ thực hiện việc tính toán tìm đường đi ngắn nhất.

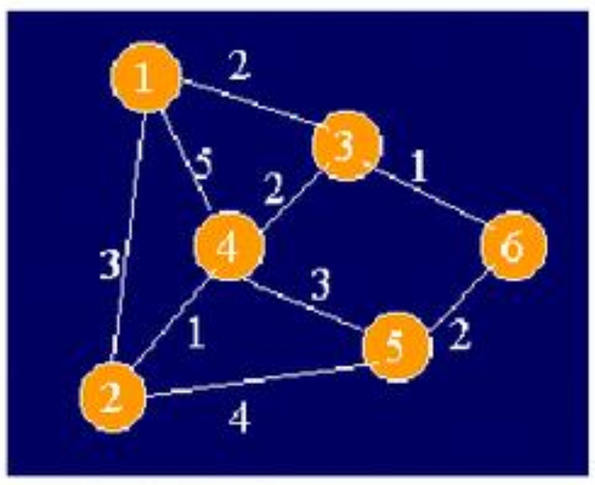
6.3.4.1 Giải thuật tìm đường đi ngắn nhất Dijkstra

Mục đích là để tìm đường đi ngắn nhất từ một nút cho trước trên đồ thị đến các nút còn lại trên mạng.
Giải thuật được mô tả như sau:

- Gọi:
  - S là nút nguồn cho trước
  - N: là tập hợp tất cả các nút đã xác định được đường đi ngắn nhất từ S.
  - Di: là độ dài đường đi ngắn nhất từ nút nguồn S đến nút i.
  - l_{ij}: là giá của cạnh nối trực tiếp nút i với nút j, sẽ là ∞ nếu không có cạnh nối trực tiếp giữa i và j.
  - Pj là nút cha của của nút j.
- Bước 1: Khởi tạo
  - $N=\{S\}$; $D_s=0$;
  - Với $\forall i \neq S$: $D_i=l_{si}$, $P_i=S$
- Bước 2: Tìm nút gần nhất kế tiếp
  - Tìm nút **i** $\notin$ N thoả Di= min (Dj) với j $\notin$ N
  - Thêm nút i vào N.
  - Nếu N chứa tất cả các nút của đồ thị thì dừng. Ngược lại sang Bước 3
  - Bước 3: Tính lại giá đường đi nhỏ nhất
    - Với mỗi nút j $\notin$ N: Tính lại $Dj= \min\{ Dj, D\mathbf{i}+ l_{ij}\}$; Pj=i;
    - Trở lại Bước 2

Ví dụ: Cho mạng có hình trạng như đồ thị hình H6.6:
Tìm đường đi ngắn nhất từ nút 1 đến các nút còn lại.
Áp dụng giải thuật ta có:

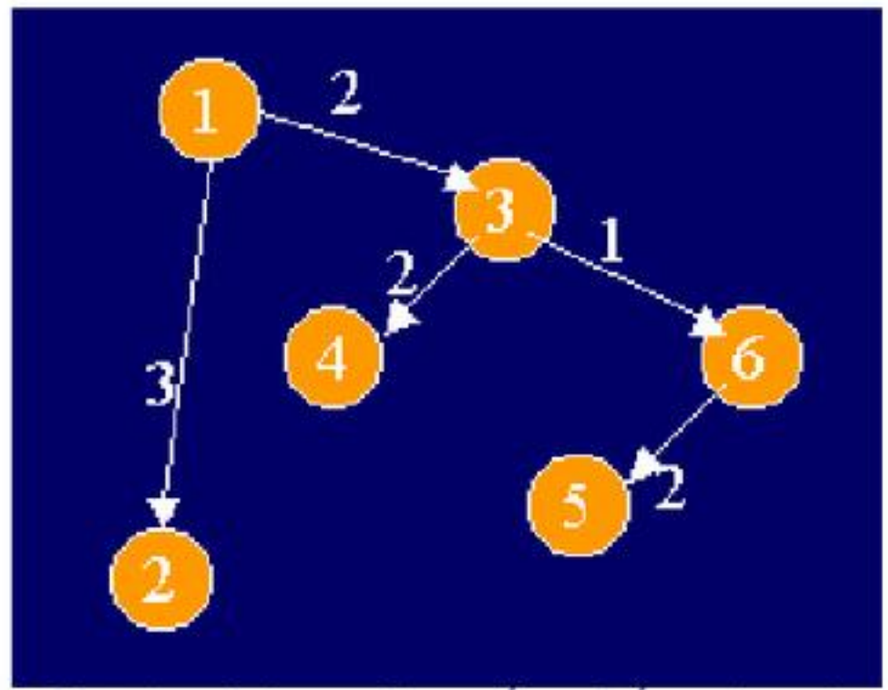
- S=1
- Các bước thực hiện được mô tả như sau:



H6.6 Hình trạng mạn

Lần lặp	N	D_2	D_3	D_4	D_5	D_6	P_2	P_3	P_4	P_5	P_6
Khởi tạo	{1}	3	2	5	∞	∞	1	1	1	1	1
1	{1,3}	3	**2**	4	∞	3	1	1	3	1	3
2	{1,3,2}	**3**		4	7	3	1		3	2	3
3	{1,3,2,6}			4	5	**3**			3	6	3
4	{1,3,2,6,4}			**4**	5				3	6	
5	{1,3,2,6,4,5}				**5**					6	

Từ kết quả trên ta vẽ được cây có đường đi ngắn nhất từ nút số 1 đến các nút còn lại như hình H6.7. Từ cây đường đi ngắn nhất này, ta xác định được rằng: để đi đến các router router 4, 5, 6 , bước kế tiếp router 1 cần gởi gói tin đến là router số 3 (next hop).
Chú ý, đường ngắn nhất này chỉ đúng theo hướng từ nút số 1 về các nút còn lại và chỉ đúng cho nút số 1 mà thôi.
Thông thường giải thuật Dijkstra được sử dụng theo mô hình chọn đường tập trung. Trong đó, Trung tâm điều khiển mạng sẽ tìm cây đường đi ngắn nhất cho từng router trên mạng và từ đó xây dựng bảng chọn đường tối ưu cho tất cả các router.



H6.7 Cây đường đi ngắn nhất từ nút 1

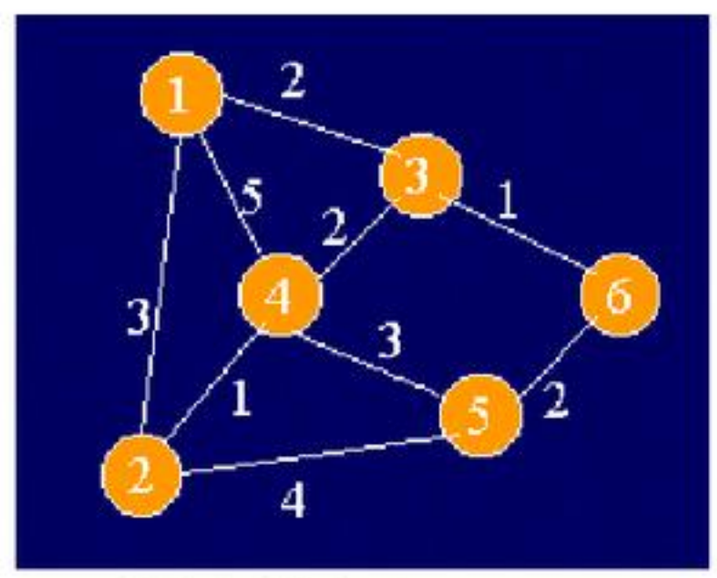
6.3.4.2 Giải thuật chọn đường tối ưu Ford-Fulkerson

Mục đích của giải thuật này là để tìm đường đi ngắn nhất từ tất cả các nút đến một nút đích cho trước trên mạng.
Giải thuật được mô tả như sau:

- Gọi
  - d là nút đích cho trước
  - D_i là chiều dài đường đi ngắn nhất từ nút i đến nút d.
  - C_i là nút con của nút i
- Bước 1: Khởi tạo:
  - Gán $D_d = 0$;
  - Với $\forall i \neq d$: gán $D_i = \infty$; $C_i = -1$;
- Bước 2: Cập nhật giá đường đi ngắn nhất từ nút i đến nút d
  - $D_i = \min\{ l_{ij} + D_j\}$ với $\forall j \neq i \Rightarrow C_i = j$;
  - Lặp lại cho đến khi không còn D_i nào bị thay đổi giá trị

Ví dụ, cho sơ đồ mạng có hình trạng như đồ thị hình H6.8. Hãy tìm đường đi ngắn nhất từ nút khác trên đồ thị đến nút 6.
Áp dụng giải thuật ta có:

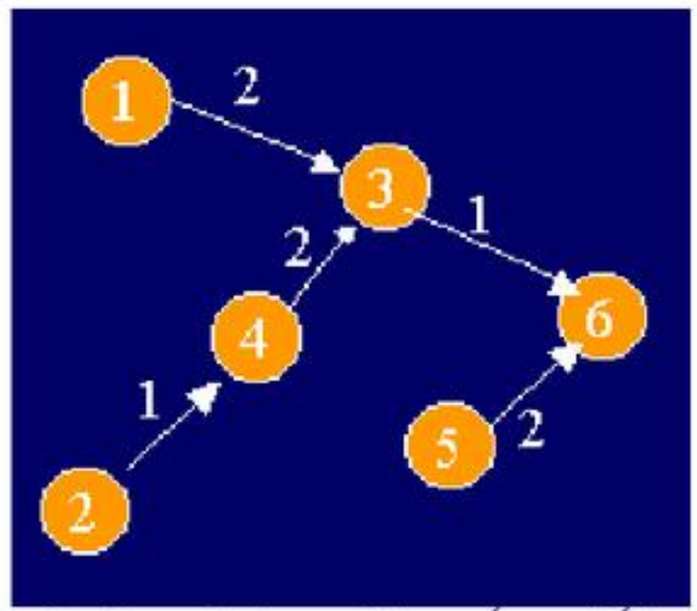
- d=6
- Các bước thực hiện được mô tả như sau:



H6.8 Hình trạng mạng

Lần lặp	D_1	D_2	D_3	D_4	D_5	C_1	C_2	C_3	C_4	C_5
Khởi tạo	∞	∞	∞	∞	∞	-1	-1	-1	-1	-1
1	∞	∞	**1**	**3**	**2**	-1	-1	**6**	**3**	**6**
2	**3**	**4**	1	3	2	**3**	**4**	6	3	6
3	3	4	1	3	2	3	4	6	3	6

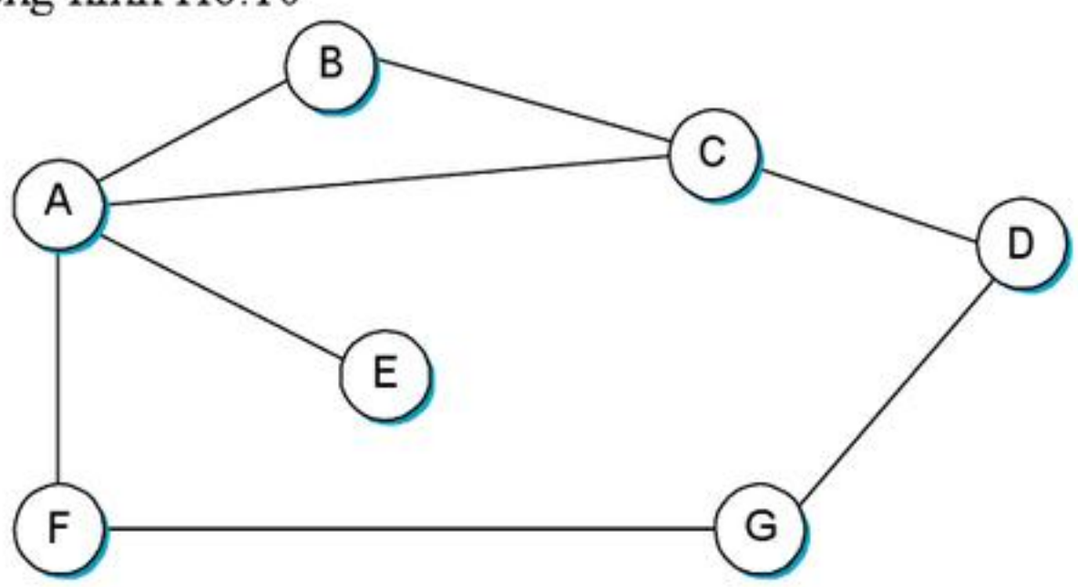
Từ kết quả trên ta vẽ lại được cây đường đi ngắn nhất từ các nút khác nhau về nút đích số 6 như H6.9. Cây này cho phép xác định đường đi tối ưu từ những nút khác nhau về nút số 6. Chẳng hạn tại nút 1, để đi về nút số 6 thì bước kế tiếp phải đi là nút số 3. Tương tự, tại nút 2, để đi về nút số 6 thì bước kế tiếp phải đi là nút số 4.
Giải thuật này được sử dụng theo mô hình phân tán. Ở đó mỗi router sẽ tự tính toán, tìm cây có đường đi ngắn nhất từ các nút khác về nó. Từ đó suy ra đường đi tối ưu cho các nút khác đến nó và gởi các đường đi này đến từng nút trên mạng.



H6.9 Cây đường đi ngắn nhất về nút 6

6.3.5 Giải pháp vạch đường Vector Khoảng cách (Distance Vector)

Ý tưởng của Distance-Vector như sau: Mỗi nút thiết lập một mảng một chiều (vector) chứa khoảng cách (chi phí) từ nó đến tất cả các nút còn lại và sau đó phát vector này đến tất cả các nút láng giềng của nó. Giả thiết đầu tiên trong Distance-Vector là: mỗi nút phải biết được chi phí của các đường nối từ nó đến tất cả các nút láng giềng có đường nối kết trực tiếp với nó. Một nối kết bị đứt (down) sẽ được gán cho chi phí có giá trị vô cùng.
Để xem xét giải thuật vạch đường Distance-Vector hoạt động như thế nào, cách dễ nhất là xem xét đồ thị được cho như trong hình H6.10



Hình 6.10 Một mạng làm ví dụ trong giải thuật Distance-Vector

Trong ví dụ này, chi phí trên mỗi đường nối đều được đặt là 1. Chúng ta có thể biểu diễn sự hiểu biết của các nút về khoảng cách từ chúng đến các nút khác như trong bảng H6.11

Thông tin được lưu tại các nút	Khoảng cách đến nút						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

H6.11 Các khoảng cách ban đầu được lưu tại mỗi nút

Chúng ta có thể xem mỗi một hàng trong bảng 6.11 như là một danh sách các khoảng cách từ một nút đến tất cả các nút khác. Khởi đầu, mỗi nút đặt giá trị 1 cho đường nối kết đến các nút láng giềng kề nó, ∞ cho các đường nối đến tất cả các nút còn lại. Do đó, lúc đầu A tin rằng nó có thể tìm đến B qua một bước nhảy (hop) và rằng nó không thể đi đến D được. Bảng vạch đường lưu tại A thể hiện những niềm tin mà A có được, ngoài ra còn lưu thêm nút kế tiếp mà A cần phải đi ra để đến một nút nào đó. Khởi đầu, bảng vạch đường của nút A trông giống như trong bảng 6.12

Đích (Destination)	**Chi phí (Cost)**	**Nút kế tiếp (Next Hop)**
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

H6.12 Bảng vạch đường khởi đầu tại nút A

Bước kế tiếp trong giải thuật vạch đường Distance-Vector là: mỗi nút sẽ gởi một thông điệp đến các láng giềng liền kề nó, trong thông điệp đó chứa danh sách các khoảng cách mà cá nhân nút tính được. Ví dụ, nút F bảo nút A rằng F có thể đi đến nút G với chi phí là 1; A cũng biết được rằng nó có thể đến F với chi phí là 1, vì thế A cộng các chi phí lại thành chi phí đi đến G là 2 thông qua F. Tổng chi phí là 2 này nhỏ hơn chi phí vô cùng lúc đầu, do đó A ghi lại nó có thể đi đến G thông qua F với chi phí là 2. Tương tự, A học được từ C rằng, nó có thể đi đến D thông qua C với chi phí là 2, và chi phí này nhỏ hơn chi phí cũ là vô cùng. Cùng lúc A cũng học từ C rằng, nó có thể đi đến B thông qua C với chi phí là 2, nhưng chi phí này lại lớn hơn chi phí cũ là 1, vì thế thông tin mới này bị bỏ qua.
Tại thời điểm này, A có thể cập nhật lại bảng chọn đường của nó với chi phí và nút ra kế tiếp để có thể đi đến tất cả các nút khác trong mạng. Kết quả được cho trong bảng H6.13

Đích (Destination)	Chi phí (Cost)	Nút kế tiếp (Next Hop)
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

H.6.13 Bảng vạch đường cuối cùng tại nút A

Nếu không có sự thay đổi về hình trạng mạng nào, chỉ cần vài cuộc trao đổi thông tin vạch đường giữa các nút trong mạng thì mọi nút đều có được thông tin vạch đường hoàn hảo. Quá trình đem thông tin vạch đường nhất quán đến mọi nút trong mạng được gọi là sự hội tụ (convergence). Hình 6.14 chỉ ra thông tin về chi phí cuối cùng từ một nút đến các nút khác trong mạng khi quá trình vạch đường đã hội tụ.

Thông tin được lưu tại các nút	Khoảng cách đến nút						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

H6.14 Các khoảng cách cuối cùng được lưu tại mỗi nút

Nét đẹp của loại giải thuật phân tán như trên nằm ở chỗ nó cho phép tất cả các nút đạt được thông tin vạch đường mà không cần phải có sự hiện diện của bộ điều khiển trung tâm nào.
Còn có vài chi tiết làm cho giải thuật Distance-Vector hoàn hảo hơn. Thứ nhất, chú ý rằng có hai tình huống khác nhau mà tại đó một nút quyết định gởi thông tin vạch đường của mình cho các nút láng giềng kề bên. Tình huống đầu tiên là sự cập nhật theo chu kỳ (periodic update). Trong tình huống này, mỗi nút tự động gởi bản cập nhật thường xuyên, ngay cả khi không có thay đổi gì trong đó cả. Điều này giúp cho các nút khác biết được nút hiện tại đang còn sống. Vả lại việc cập nhật thường xuyên còn giúp cho các nút láng giềng có thể có được thông tin vạch đường nhanh chóng trong trường hợp thông tin của chúng bị mất. Tần số phát thông tin vạch đường đi có thể khác nhau tùy vào giải thuật, chúng thường có giá trị từ vài giây đến vài phút. Tình huống thứ hai gọi là sự cập nhật do bị kích hoạt (triggered update). Tình huống này xảy ra mỗi khi có sự thay đổi thông tin trong bảng vạch đường của nút. Nghĩa là mỗi khi bảng vạch đường có sự thay đổi, nút sẽ gởi bản cập nhật này cho các láng giềng của mình.
Bây giờ ta xem xét điều gì xảy ra khi một đường nối kết hay một nút bị hỏng. Những nút đầu tiên phát hiện ra điều này sẽ gởi thông tin vạch đường mới cho láng giềng của chúng ngay, và thông thường hệ thống sẽ ổn định với tình trạng mới một cách nhanh chóng. Còn đối với câu hỏi làm sao nút phát hiện ra sự cố, có nhiều câu trả lời khác nhau. Cách tiếp cận thứ nhất là: một nút liên tục kiểm tra đường nối tới các nút láng giềng khác bằng cách gởi các gói tin điều khiển tới chúng và kiểm tra xem nó có nhận được các gói tin trả lời hay không. Cách tiếp cận khác là: một nút phát hiện ra một đường nối kết (hay nút ở đầu kia của đường nối) gặp sự cố khi nó không nhận được thông tin vạch đường một cách định kỳ từ láng giềng của nó.
Ví dụ, xem xét điều gì sẽ xảy ra khi F phát hiện ra đường nối từ nó đến G bị hỏng. Đầu tiên, F đặt chi phí của đường nối từ nó đến A thành vô cùng và gởi thông tin này đến A. Do A đã biết là cần 2 bước để đi từ nó đến G thông qua F, A sẽ đặt lại chi phí từ nó đến G là vô cùng. Tuy nhiên, với bản cập nhật kế tiếp từ C, A phát hiện ra rằng có một đường đi dài 2 hops từ C đến G, do đó nó sẽ cập nhật lại đường đi từ nó đến G dài 3 hops thông qua C. Và khi A quảng cáo thông tin này cho F, F lại cập nhật lại đường đi dài 4 hops đến G thông qua A.
Không may là: một số tình huống phát sinh lỗi khác lại làm cho mạng mất ổn định nghiêm trọng. Giả sử nối kết từ A đến E bị đứt. Trong những chu kỳ cập nhật sau, A thông báo đường đi từ nó đến E dài vô cùng, nhưng B và C lại quảng cáo đường đi từ chúng đến E dài 2 hops. Nếu các bản cập nhật được định thời để phát cùng lúc, B sẽ sửa lại độ dài đường đi từ nó đến E là 3 thông qua C, C sửa lại độ dài đường đi từ nó đến E là 3 thông qua B. Sau đó A lại nghe B và C quảng cáo độ dài đường đi từ chúng đến E là 3 và giả sử A chọn B là nút kế tiếp để đi đến E, nó sẽ cập nhật lại độ dài đoạn đường là 4. Đến chu kỳ kế tiếp, B nghe C nói độ dài từ C đến E là 3 nên cập nhật lại độ dài đường đi từ B đến E là 4 thông qua C, C thì làm ngược lại: sửa lại con đường từ nó đến E là 4 thông qua B. Rồi lại đến lượt A nghe B sửa lại độ dài từ A đến E là 5 thông qua B. Sự thể sẽ tiếp diễn cho đến khi các độ dài tăng đến một số có thể coi là vô cùng. Nhưng tại thời điểm này, không nút nào biết là E không thể đến được, và các bảng vạch đường trong mạng luôn không ổn định. Tình huống này được gọi là vấn đề "*đếm tới vô cùng*" (count-to-infinity problem).

Có vài giải pháp giải quyết một phần vấn đề "*đếm tới vô cùng*". Giải pháp thứ nhất là dùng một số khá nhỏ để coi như gần bằng vô cùng. Ví dụ như chúng ra có thể quyết định số lượng bước nhảy (hop) tối đa để đi qua một mạng là không quá 16, và do đó ta chọn 16 là số gần bằng vô cùng. Con số này ít ra cũng giới hạn được thời gian mà các nút có thể phải bỏ ra để đếm tới vô cùng. Tuy nhiên giải pháp này có thể gặp vấn đề nếu một số nút mạng được chia tách và mạng có thể cần nhiều hơn 16 bước nhảy để duyệt hết nó.
Một kỹ thuật khác dùng để cải thiện thời gian dùng để ổn định hóa mạng được gọi là kỹ thật "*chia tầm nhìn*" (split horizon). Ý tưởng là: khi một nút gởi một bảng cập nhật vạch đường cho các láng giềng của nó, nó sẽ không gởi những thông tin vạch đường mà nó đã nhận từ một nút láng giềng ngược lại chính nút láng giềng đó. Ví dụ như nếu B có một đường đi (E, 2, A) trong bảng vạch đương của nó, B chắc hẳn phải biết rằng nó học con đường này từ A, vì thế mỗi khi B gởi thông tin cập nhật cho A nó sẽ không gởi con đường (E, 2) trong đó. Tuy nhiên giải pháp này chỉ tốt khi nó xoay quanh 2 nút mà thôi.

6.3.6 Giải pháp chọn đường "Trạng thái nối kết" (Link State)

Vạch đường kiểu Link-state là một ví dụ thứ hai trong họ giao thức vạch đường bên trong một miền. Giả thiết đầu tiên trong Link-state cũng khá giống trong Distance-vector: Mỗi nút được giả định có khả năng tìm ra trạng thái của đường nối nó đến các nút láng giềng và chi phí trên mỗi đường nối đó. Nhắc lại lần nữa: chúng ta muốn cung cấp cho mỗi nút đủ thông tin để cho phép nó tìm ra đường đi có chi phí thấp nhất đến bất kỳ đích nào. Ý tưởng nền tảng đằng sau các giao thức kiểu Link-state là rất đơn giản: Mọi nút đều biết đường đi đến các nút láng giềng kề bên chúng và nếu chúng ta đảm bảo rằng tổng các kiến thức này được phân phối cho mọi nút thì mỗi nút sẽ có đủ hiểu biết về mạng để dựng lên một bản đồ hoàn chỉnh của mạng. Giải thuật Link-state dựa trên hai kỹ thuật: sự phân phối một cách tin cậy thông tin về trạng thái các đường nối kết; và sự tính toán các đường đi từ kiến thức tổng hợp về trạng thái các đường nối kết.

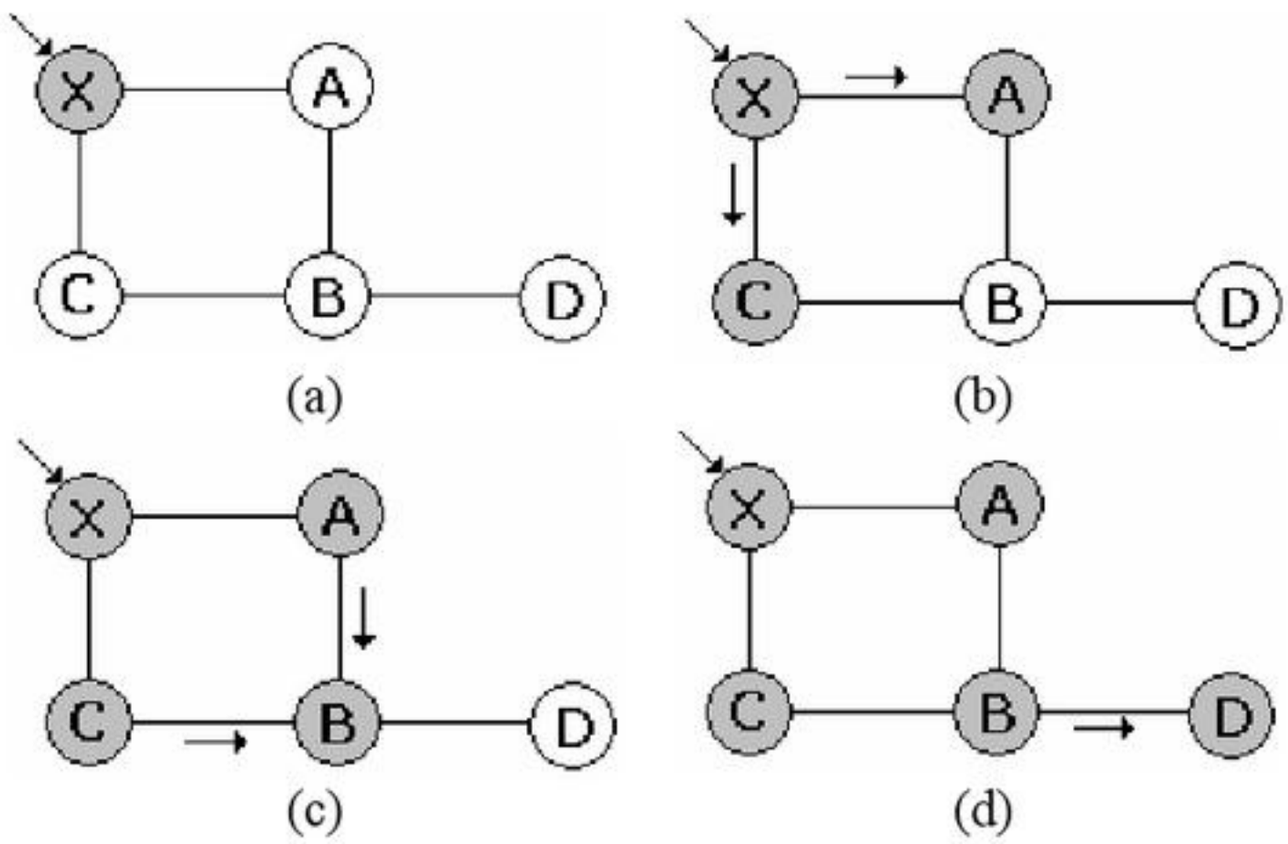
6.3.6.1 Làm ngập một cách tin cậy (Reliable Flooding)

"Làm ngập" là quá trình thực hiện cam kết: "tất cả các nút tham gia vào giao thức vạch đường đều nhận được thông tin về trạng thái nối kết từ tất cả các nút khác". Như khái niệm "làm ngập" ám chỉ, ý tưởng cơ sở của Link-state là cho một nút phát thông tin về trạng thái nối kết của nó với mọi nút láng giềng liền kề, đến lượt mỗi nút nhận được thông tin trên lại chuyển phát thông tin đó ra các nút láng giềng của nó. Tiến trình này cứ tiếp diễn cho đến khi thông tin đến được mọi nút trong mạng.
Cụ thể hơn, mỗi nút tạo ra gói tin cập nhật, còn được gọi là gói tin trạng thái nối kết (link-state packet – LSP), chứa những thông tin sau:

- ID của nút đã tạo ra LSP
- Một danh sách các nút láng giềng có đường nối trực tiếp tới nút đó, cùng với chi phí của đường nối đến mỗi nút.
- Một số thứ tự
- Thời gian sống (time to live) của gói tin này

Hai mục đầu là cần thiết cho việc tính toán chọn đường; hai mục sau cùng được sử dụng để giúp cho quá trình làm ngập thật chắc. Tính tin cậy ở đây đòi hỏi việc đảm bảo các nút trong mạng có được thông tin có phiên bản mới nhất, do có nhiều LSP trái ngược nhau từ một nút được phát lên mạng. Đảm bảo việc làm ngập có thể tin cậy được là một việc khó (Ví dụ, một phiên bản cũ của giao thức vạch đường link-state trong ARPANET đã làm cho mạng này bị tê liệt vào năm 1981).
Việc làm ngập được thực hiện như sau: Đầu tiên, việc truyền các LSP giữa các nút kề nhau được bảo đảm tính tin cậy bằng cách sử dụng cơ chế báo nhận (acknowledgement) và làm lại khi bị lỗi (retransmission) giống như ở tầng liên kết dữ liệu. Tuy nhiên, cần thực hiện thêm một số bước để đảm bảo việc phát một LSP từ một nút đến tất cả các nút khác trong mạng là đáng tin cậy.
Giả sử nút X nhận được một phiên bản LSP có nguồn gốc từ nút Y nào đó. Chú ý rằng nút Y có thể là bất kỳ router nào ở trong cùng một miền với X. X kiểm tra xem nó đã có bất kỳ phiên bản LSP nào từ Y không. Nếu không, nó sẽ lưu LSP này. Nếu có, X sẽ so sánh hai số thứ tự trong hai

LSP. Nếu LSP mới đến có số thứ tự lớn hơn số thứ tự của LSP có sẵn, X cho rằng LSP mới đến là mới hơn, và do đó X sẽ thay LSP cũ bằng phiên bản mới này. Ngược lại, với một số thứ tự nhỏ hơn (hoặc bằng), LSP mới đến sẽ bị coi là cũ hơn cái đang có sẵn (hoặc ít ra là không mới hơn), và vì thế nó sẽ bị bỏ qua, không cần phải làm gì thêm. Nếu LSP mới đến là cái mới hơn, nút X sẽ gởi một phiên bản của LSP này ra tất cả các nút láng giềng liền kề nó ngoại trừ nút láng giềng vừa gởi cho nó phiên bản LSP mới vừa đề cập. Đến phiên các nút láng giềng của X lại xoay qua phát tán LSP mới này sang các nút láng giềng khác. Việc "LSP không được gởi ngược lại nút vừa phát ra nó" sẽ giúp dẫn đến điểm dừng của quá trình phát tán LSP này. Sự phát tán dây chuyền có điểm dừng này sẽ đảm bảo cho việc đem phiên bản LSP mới nhất đến tất cả các nút trong mạng.
Hình H6.15 thể hiện một LSP được dùng làm ngập một mạng nhỏ. Hình (a) thể hiện X nhận được một LSP mới; (b) X đẩy LSP mới ra A và C; (c) A và C đẩy LSP qua B; (d) B đẩy LSP qua D và quá trình làm ngập kết thúc.



H6.15 Việc làm ngập mạng với các gói tin LSP

Cũng giống như trong giải thuật Distance-Vector, sẽ có hai tình huống mà một nút quyết định gởi LSP đến các nút láng giềng: gởi một cách định kỳ hoặc gởi do bị kích hoạt.
Một trong những ưu tiên hàng đầu của cơ chế làm ngập (flooding) là phải đảm bảo đem thông tin mới nhất đến mọi nút trong mạng càng nhanh càng tốt và các thông tin cũ phải được rút ra không cho lưu thông trên mạng nữa. Thêm nữa, rất là lý tưởng nếu ta có thể giảm thiểu lượng thông tin vạch đường lưu chuyển trên mạng – một kiểu phí tổn theo cách nhìn của nhiều người.
Một phương pháp cần thiết để giảm thiểu phí tổn dành cho việc vạch đường là tránh gởi các LSP trừ trường hợp hết sức cần thiết. Điều này có thể thực hiện được bằng cách sử dụng các bộ định thời (timer) có giá trị rất lớn – thường là kéo dài hàng giờ - dùng để định kỳ phát các LSP.
Còn để đảm bảo rằng thông tin cũ phải được thay thế bởi thông tin mới, các LSP sẽ mang số thứ tự. Mỗi khi một nút phát LSP mới, nó sẽ tăng số thứ tự lên 1. Không giống như hầu hết các giao thức khác, số thứ tự trong LSP sẽ không được đếm xoay vòng (modulo), vì thế trường chứa số này phải đủ lớn (ví dụ như 64 bit). Nếu một nút bị chết (down) và sau đó được khởi động lại, nó sẽ khởi động trường số thứ tự lại bằng 0. Nếu một nút bị chết quá lâu, tất cả các LSP của nút đó sẽ bị mãn kỳ (timed out); ngoài ra, nếu cuối cùng nút này lại nhận được LSP của chính nó với số thứ tự lớn hơn bản gốc, nút có thể lấy số lớn hơn làm số thứ tự mới.
Các LSP cũng mang theo thời gian sống của nó (Time to live - TTL). Điều này dùng để đảm bảo các LSP cũ rút cuộc cũng bị xóa khỏi mạng. Một nút luôn luôn giảm trường TTL của một LSP mới đến nó đi 1 trước khi chuyển LSP này ra các nút láng giềng. Khi trường TTL còn 0, nút phát lại LSP với TTL = 0, điều đó sẽ được thông dịch bởi tất cả các nút trong mạng như là tín hiệu cho phép xóa LSP đó.

6.3.6.2 Tính toán chọn đường trong Link State

Khi một nút có một phiên bản LSP từ tất cả các nút khác trong mạng, nó đã có thể tính toán ra bản đồ hoàn chỉnh cho hình thái của mạng, và từ bản đồ này nút có thể quyết định con đường tốt nhất đến tất cả các nút còn lại trong mạng. Giải pháp chọn đường chính là giải thuật tìm đường đi ngắn nhất Dijkstra.

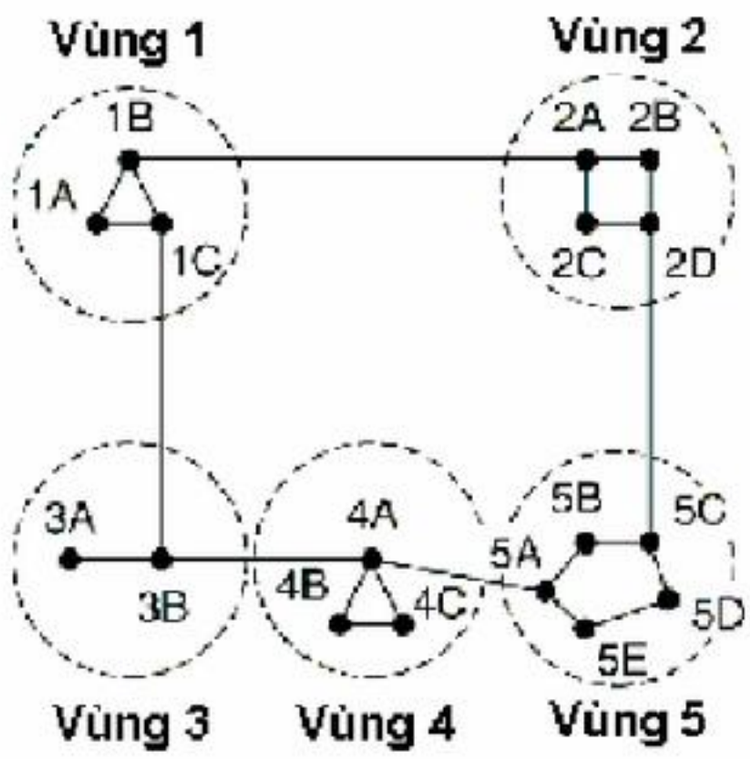
6.3.7 Vạch đường phân cấp (Hierarchical Routing)

Khi mạng tăng kích thước, kích thước bảng vạch đường của các router tăng theo. Không chỉ bộ nhớ của router bị tiêu hao quá nhiều cho việc trữ các bảng vạch đường, mà CPU còn phải tốn nhiều thời gian để quét bộ nhớ và cũng cần nhiều băng thông hơn để truyền những thông tin chọn đường này. Rồi cũng sẽ đến lúc mạng máy tính phát triển đến mức không một router nào có đủ khả năng trữ một đầu mục thông tin về một router khác, vì thế việc vạch đường phải phát triển theo đường hướng khác: vạch đường phân cấp.

Khi việc vạch đường phân cấp được áp dụng, các router được chia thành những vùng (domain). Trong mỗi vùng, mỗi router biết cách vạch đường cho các gói tin đi đến được mọi đích trong nội vùng của nó, nhưng không biết gì về cấu trúc bên trong của các vùng khác. Khi nhiều vùng được nối kết với nhau, đương nhiên mỗi vùng được công nhận tính độc lập để giải phóng các router trong các vùng đó khỏi việc phải tìm hiểu hình trạng của các vùng khác.

Với những mạng thật lớn, kiến trúc phân cấp hai mức có thể sẽ không đủ; có thể cần phải nhóm các vùng lại thành liên vùng, nhóm các liên vùng thành khu vực...

Hình H6.16 thể hiện một mạng được vạch đường phân cấp gồm hai mức có năm vùng. Bảng vạch đường đầy đủ của router A gồm có 17 mục từ như trong hình H6.16(b). Khi vạch đường được thực hiện theo kiểu phân cấp, bảng vạch đường của A giống như bảng H6.16(c), có mọi mục từ chỉ đến các router cục bộ giống như trước, tuy nhiên các mục từ chỉ đến các vùng khác lại được cô đặc lại thành một router. Do tỉ lệ các router trong các vùng tăng, vì thế cách làm này giúp rút ngắn bảng vạch đường.



(a)

Bảng vạch đường đầy đủ của nút 1A

Đích	Lối ra	Chi phí
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Bảng vạch đường phân cấp của host 1A

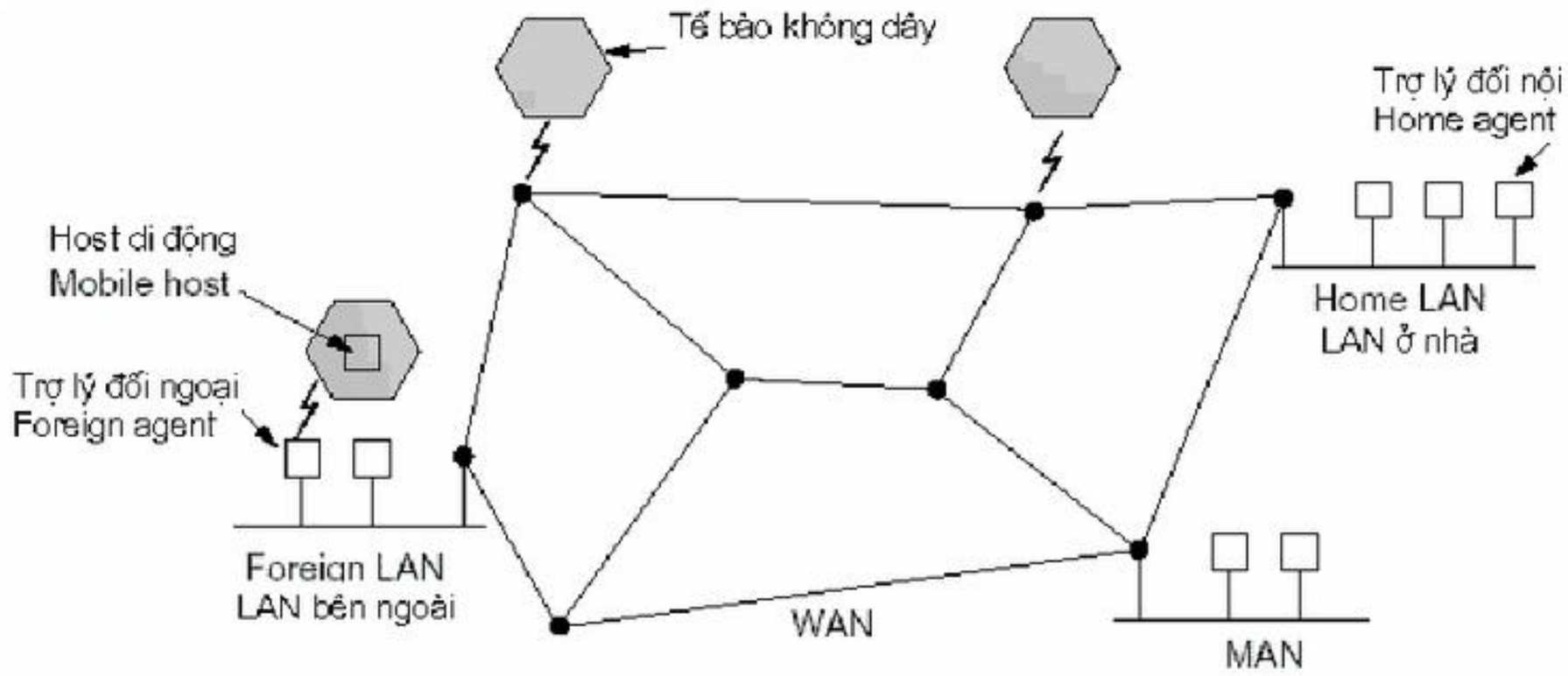
Đích	Lối ra	Chi phí
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

H6.16 Vạch đường phân cấp

6.3.8 Vạch đường trong mạng di động

Ngày nay, hàng triệu người đang sở hữu máy tính xách tay, và thông thường họ muốn đọc email cũng như truy xuất các hệ thống tập tin cho dù họ đang ở bất kỳ nơi nào trên thế giới. Việc sử dụng các host di động này dẫn đến một vấn đề phức tạp mới: để vạch đường cho gói tin đến host di động, trước tiên phải tìm ra nó đã. Chủ đề về tích hợp các host di động lại thành một mạng là tương đối mới, tuy vậy trong phần này chúng ta sẽ phác thảo ra một số vấn đề phát sinh và chỉ ra các giải pháp khả thi.



H6.17 Mô hình mạng có hệ thống không dây

Mô hình mạng mà các nhà thiết kế thường sử dụng được chỉ ra trong hình H6.17. Ở đây chúng ta có một mạng WAN bao gồm vài router và host. Mạng WAN được dùng để nối kết các mạng LAN, MAN, các tế bào mạng không dây (Wireless cell).

Các host không bao giờ di chuyển được gọi là cố định, chúng được nối vào mạng bởi các đường cáp đồng hoặc quang. Ngược lại, chúng ta sẽ phân biệt hai loại host khác: loại di cư (migratory host) và loại lang thang (roaming host). Loại host di cư về bản chất là host cố định, nhưng chúng thỉnh thoảng lại chuyển từ địa bàn (site) này đến địa bàn mạng kia, và chúng chỉ có thể sử dụng mạng mới khi được nối kết vật lý vào đấy. Loại host lang thang thực chất vừa chạy vừa tính toán, nó muốn duy trì các nối kết mạng ngay cả khi đang di chuyển. Chúng ta sẽ sử dụng thuật ngữ "host di động" để ám chỉ hai loại di động vừa nói đến, tức là chúng đã đi khỏi nhưng lại muốn duy trì liên lạc về nhà.

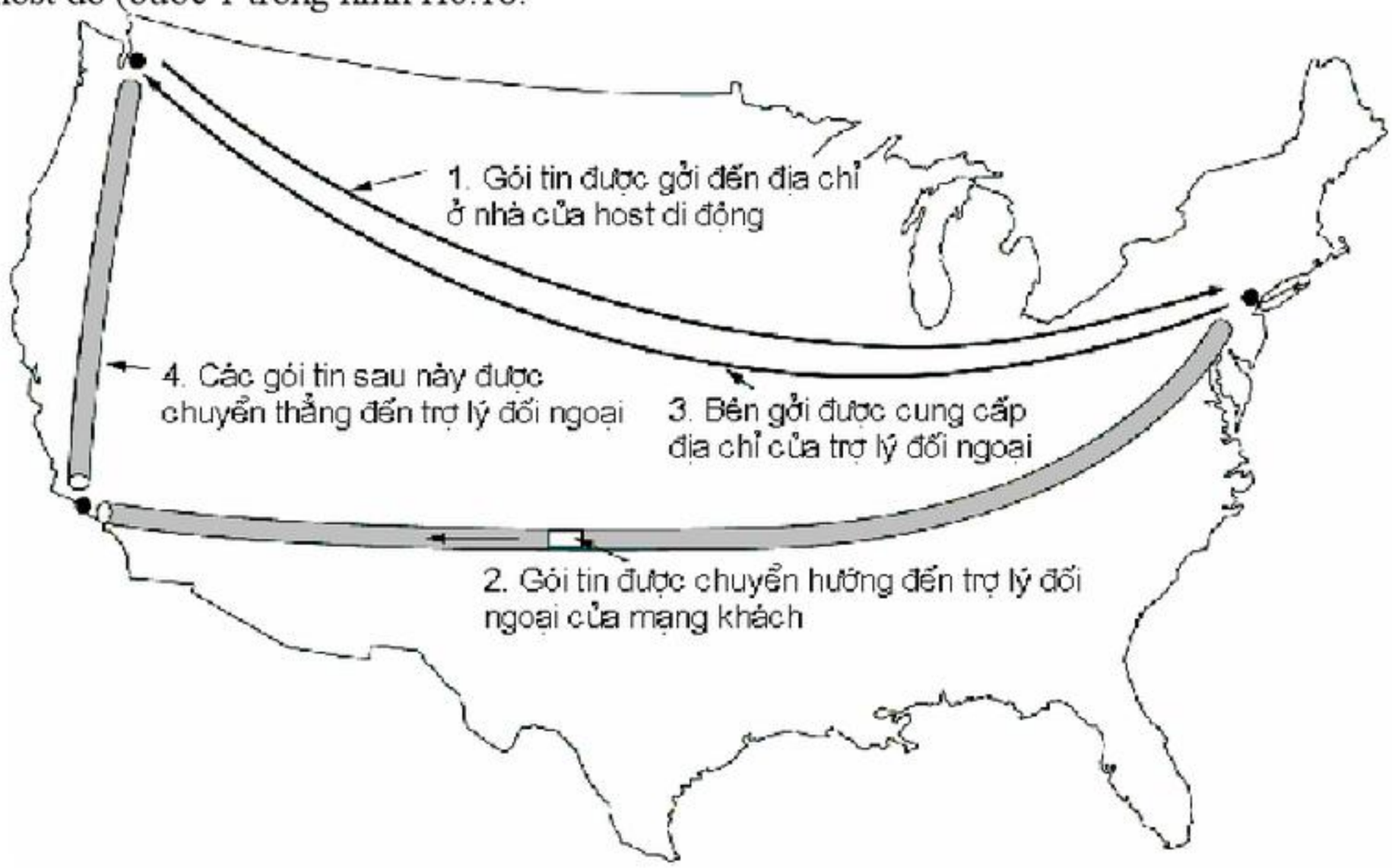
Tất cả các host được giả sử đều có vị trí mạng nhà (home location) và vị trí này không bao giờ thay đổi. Các host cũng có địa chỉ lâu dài tại nhà (home address) và địa chỉ này có thể được dùng để xác định vị trí mạng nhà của nó, cũng giống như số điện thoại 84-071-831301 chỉ ra số đó ở Việt Nam (mã 084), thành phố Cần Thơ (mã 071). Mục tiêu của việc vạch đường trong hệ thống có các host di động là phải đảm bảo có thể gởi được gói tin đến host di động sử dụng địa chỉ tại nhà của nó và làm cho các gói tin đến được host di động một cách hiệu quả cho dù host này có ở đâu đi nữa.

Trong mô hình ở hình H6.17, WAN được chia thành các đơn vị nhỏ, ở đây chúng ta gọi là khu vực (area), thường là LAN hoặc tế bào mạng không dây. Mỗi khu vực có một hoặc nhiều trợ lý đối ngoại (foreign agent - FA) – đó là những tiến trình làm nhiệm vụ theo dõi các host khách đang viếng thăm khu vực của mình. Thêm vào đó, mỗi khu vực còn có một trợ lý đối nội (home agent - HA), làm nhiệm vụ theo dõi những host có nhà nằm trong khu vực nhưng hiện đang viếng thăm khu vực khác.

Khi một host đi vào một khu vực mới (có thể là host này muốn thường trú trong mạng LAN mới hoặc chỉ đi ngang cell này thôi), nó phải đăng ký với trợ lý đối ngoại ở đó. Thủ tục đăng ký diễn ra như sau:

1. Theo chu kỳ, mỗi trợ lý đối ngoại sẽ phát ra những thông điệp thông báo sự hiện diện của nó cùng với địa chỉ. Một host mới tới sẽ chờ lắng nghe thông báo này. Nếu host cảm thấy nó đã chờ lâu nhưng không nhận được thông báo, host có thể tự phát câu hỏi: Có bất kỳ trợ lý đối ngoại nào ở đây không?
2. Host di động đăng ký với trợ lý đối ngoại, cung cấp thông tin về địa chỉ ở nhà, địa chỉ MAC và một số thông tin về an ninh khác.
3. Trợ lý đối ngoại liên hệ với trợ lý đối nội ở nhà của host đó và nói: Một host của ông đang ở đây. Thông điệp mà trợ lý đối ngoại gởi cho trợ lý đối nội bên kia chứa địa chỉ mạng của trợ lý đối ngoại đó. Thông điệp này còn chứa thông tin an ninh dùng để thuyết phục trợ lý đối nội bên kia rằng host di động của nó thực sự đang ở đó.
4. Trợ lý đối nội bên kia kiểm tra thông tin an ninh, trong đó có một tem thời gian, để chứng tỏ được rằng tem này vừa được tạo ra trong vòng vài giây. Và nếu kết quả kiểm tra là tốt đẹp, nó sẽ bảo trợ lý đối ngoại bên kia tiến hành làm việc.
5. Khi trợ lý đối ngoại nhận được sự chấp thuận của trợ lý đối nội bên kia, nó tạo ra một đầu mục trong các bảng quản lý và thông báo cho host di động rằng: Bạn đã đăng ký thành công.

Lý tưởng nhất là khi một host di động rời khỏi một cell, nó phải thông báo với trợ lý đối ngoại ở đó để xóa đăng ký. Nhưng đa phần người sử dụng thường tắt máy ngay khi sử dụng xong.
Khi một gói tin được gởi đến một host di động, đầu tiên gói tin đó được gởi đến mạng LAN nhà của host đó (bước 1 trong hình H6.18.

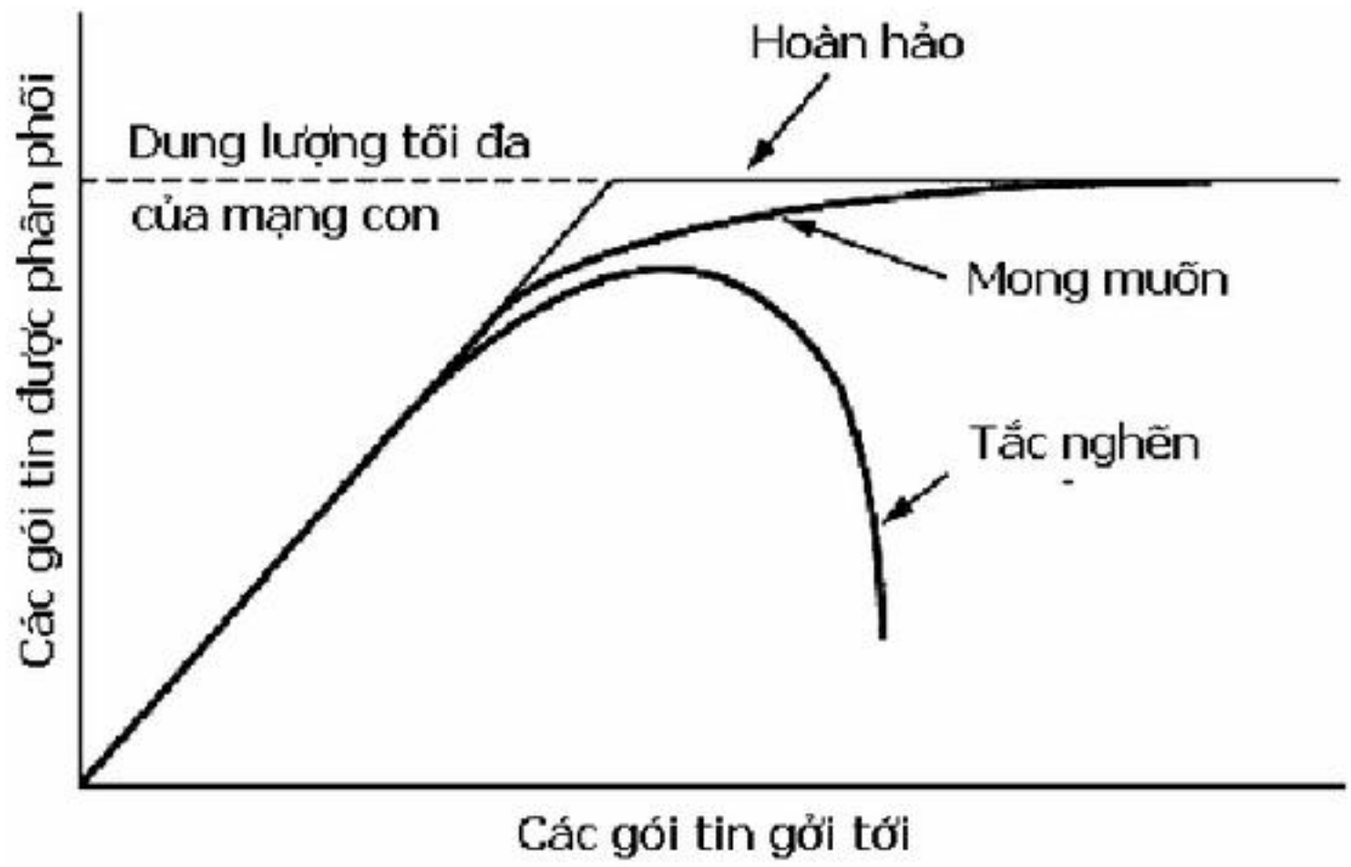


H6.18 Vạch đường trong mạng di động

Bên gởi, ví dụ đang ở Tiền Giang, gởi gói tin đến mạng nhà của host di động ở Cần Thơ. Giả sử host di động đang ở Đồng Tháp. Trợ lý đối nội ở Cần Thơ tìm ra được địa chỉ tạm thời của host di động, đóng gói gói tin đó và chuyển cho trợ lý đối ngoại của mạng ở Đồng Tháp (bước 2). Đến phiên trợ lý đối ngoại ở Đồng Tháp mở gói gói tin đó và phát cho host di động thông tin dưới dạng khung thông tin ở mức liên kết dữ liệu.
Sau đó trợ lý đối ngoại ở Đồng Tháp sẽ bảo bên gởi ở Tiền Giang hãy đóng gói và gởi gói tin trực tiếp đến Đồng Tháp (bước 3). Từ đó trở về sau, nhưng gói tin mà bên gởi muốn gởi cho host di động được gởi trực tiếp đến trợ lý đối ngoại tại Đồng Tháp, rồi được trợ lý đối ngoại phát trực tiếp đến host (bước 4).

6.4 Các giải thuật chống tắc nghẽn

Khi có quá nhiều gói tin hiện diện trong một mạng con (hoặc một phần của nó), hiệu năng hoạt động của hệ thống bị giảm. Tình trạng này được gọi là "tắc nghẽn".



H6.19 Mô tả tắc nghẽn

Hình H6.19 mô tả lại hiện tượng tắc nghẽn. Khi số lượng gói tin chạy trong mạng con nằm dưới ngưỡng cho phép, chúng đều được phân phối đến đích (ngoại trừ những gói tin bị lỗi), và số lượng gói tin được phân phối tỉ lệ thuận với số lượng gói tin được phát ra lúc đầu. Tuy nhiên, khi mật độ giao thông tăng quá cao, các router không còn đáp ứng kịp nữa và chúng dần dần đánh mất một số gói tin. Điều này có xu hướng làm cho vấn đề tắc nghẽn nghiêm trọng thêm. Khi mà giao thông cực cao, hiệu năng hệ thống sụp đổ hoàn toàn và hầu như không gói tin nào được phân phát đến đích.

Có vài yếu tố góp phần gây ra tắc nghẽn. Nếu đột nhiên nhiều luồng mang các gói tin đến một nút tại nhiều ngõ vào, và tất cả các gói tin này đều cần một ngõ ra, thì một hàng đợi sẽ xuất hiện. Nếu không đủ bộ nhớ để lưu các gói tin trên hàng đợi này, một số gói tin sẽ bị mất. Tăng thêm bộ nhớ chỉ giúp không mất gói tin trong hàng đợi, nhưng Nagle (1987) đã chỉ ra rằng: nếu một router có bộ nhớ vô hạn, sự tắc nghẽn lại càng tồi tệ hơn! Lý do là khi mà gói tin đến được đầu của hàng đợi thì nó đã bị mãn kỳ (timed out), và do đó sẽ có nhiều phiên bản trùng với gói tin đó được bên gởi gởi đến router, làm tăng thêm tải của mọi hướng đi đến đích của gói tin.

Các bộ xử lý chậm cũng có thể gây ra tắc nghẽn. Nếu CPU của router xử lý các gói tin trung chuyển qua nó chậm, hàng đợi cũng sẽ phát sinh, cho dù dung lượng các đường nối vào và ra đều vượt yêu cầu.

Tóm lại, đường truyền băng thông thấp có thể gây ra tắc nghẽn. Nâng cấp đường truyền nhưng năng lực xử lý của bộ xử lý tại router yếu cũng gây ra tắc nghẽn. Thành thử, nâng cấp một phần mà không phải là toàn bộ hệ thống chỉ đẩy sự tắc nghẽn từ nơi này đến nơi khác mà thôi. Vấn đề phát sinh từ sự bất cân đối giữa các bộ phận của hệ thống, và nó chỉ qua đi khi mà các bộ phận này được giữ cân bằng với nhau.

6.4.1 Các nguyên tắc chung để điều khiển tắc nghẽn

Nhiều bài toán trong các hệ thống phức tạp, ví dụ như trong mạng máy tính, có thể được xem xét theo quan điểm của lý thuyết điều khiển (control theory). Cách tiếp cận này dẫn đến việc chia các giải pháp thành hai loại: vòng đóng và vòng mở (closed loop and open loop). Các giải pháp dạng vòng đóng cố gắng giải quyết vấn đề tắc nghẽn bằng cách đưa ra thiết kế tốt cho mạng, thực chất là để đảm bảo tắt nghẽn sẽ không xảy ra. Một khi mạng được khởi động và chạy, sẽ không có việc sửa chữa giữa kỳ.

Các công cụ thực hiện việc điều khiển kiểu vòng mở bao gồm việc quyết định khi nào nên chấp nhận luồng giao thông mới, quyết định khi nào thì bỏ qua các gói tin và bỏ qua gói nào. Tất cả các công cụ trên đều có đặc điểm chung là chúng đưa ra các quyết định mà không quan tâm đến trạng thái hiện hành của mạng.

Ngược lại, các giải pháp kiểu vòng đóng dựa trên quan niệm về chu trình phản hồi thông tin. Cách tiếp cận này bao gồm 3 phần:

1. Giám sát hệ thống để phát hiện nơi nào và khi nào xảy ra tắc nghẽn.
2. Chuyển thông tin đến những nơi cần có những hành động ứng phó.
3. Điều chỉnh lại hoạt động của hệ thống để khắc phục sự cố.

Nhiều kiểu đo lường có thể được sử dụng để giám sát một mạng con để phát hiện ra tắc nghẽn ở đó. Các kiểu đo lường thường dùng nhất là tỉ lệ các gói tin bị bỏ qua do thiếu không gian trữ đệm, chiều dài trung bình của các hàng đợi, số lượng các gói tin bị mãn kỳ và được tái truyền, thời gian trì hoãn gói tin trung bình. Trong mọi tình huống, các số đo tăng đồng nghĩa với việc tăng tắc nghẽn.

Bước thứ hai trong chu trình phản hồi là chuyển thông tin về tắc nghẽn từ điểm được phát hiện bị tắc nghẽn đến điểm có trách nhiệm xử lý tình huống đó. Cách dễ nhất là để cho router phát hiện ra tắc nghẽn phát thông báo đến nút nguồn vừa gởi thông tin đến làm tắc hệ thống. Dĩ nhiên, thông báo này làm cho tắc nghẽn tăng thêm tạm thời.

Một cách thông báo tắc nghẽn khác là: Người ta dành riêng một bit hoặc một trường trong gói tin để trong trường hợp có tắc nghẽn, router có thể bật bit hoặc trường này lên và gởi nó đến mọi ngõ ra nhằm thông báo cho các láng giềng của nó biết.

Hoặc cũng có thể dùng cách phản hồi sau: Cho các host hoặc router thường xuyên gởi các gói tin thăm dò ra ngoài để hỏi thẳng về tình hình tắc nghẽn. Thông tin này có thể được sử dụng để chuyển hướng vạch đường vòng qua khu vực bị tắc nghẽn. Ví dụ thực tế: Một số đài phát thanh thường phái một số máy bay trực thăng bay vòng quanh thành phố để báo cáo lại những trục đường bị tắc, từ đó thông báo đến thính giả giúp họ chuyển hướng lái xe tránh những điểm nóng.

Sự hiện diện của tắc nghẽn đồng nghĩa với việc: tài nguyên của hệ thống không đủ để tải gánh nặng thông tin truyền qua. Vì thế ta nghĩ ra hai giải pháp: tăng tài nguyên hoặc giảm tải. Ví dụ, một mạng con có thể bắt đầu sử dụng các đường điện thoại quay số để tạm thời tăng băng thông giữa một số điểm nào đó. Trong các hệ thống vệ tinh, việc tăng công suất truyền đồng nghĩa với việc cung cấp băng thông lớn hơn. Chia tách lưu lượng thông tin cho chúng chạy trên nhiều đường đi khác nhau cũng có thể giúp tăng băng thông. Cuối cùng, các router dự phòng (thường để dự phòng tình huống các router chính bị sự cố) có thể được mang ra chạy trực tuyến để tăng dung lượng truyền tải của hệ thống khi tắc nghẽn nghiêm trọng xảy ra.

Tuy nhiên, đôi khi ta không thể tăng tài nguyên của hệ thống lên nữa, hoặc tài nguyên đã tăng tối đa. Cách thức duy nhất để chống lại tắc nghẽn là giảm tải. Có nhiều cách giảm tải, ví dụ: từ chối phục vụ một số người dùng, giảm cấp dịch vụ đối với vài hoặc tất cả người dùng, và buộc người dùng cung cấp lịch trình phát ra yêu cầu của họ.

6.4.2 Các biện pháp phòng ngừa tắc nghẽn

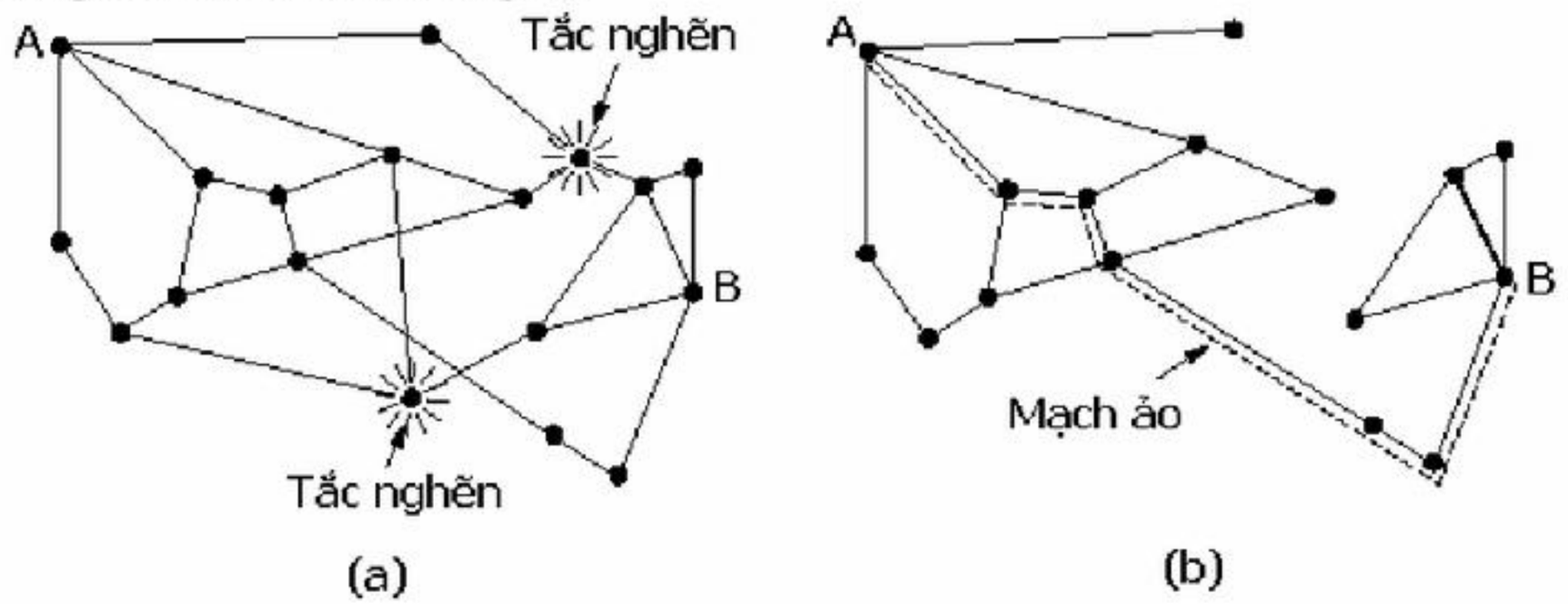
Tại tầng mạng, việc chọn sử dụng mạch ảo hay datagram sẽ tác động đến tắc nghẽn do nhiều giải thuật điều khiển tắc nghẽn chỉ chạy trên mạch ảo. Giải pháp "lập hàng đợi cho các gói tin và phục vụ chúng" liên quan đến việc một router có một hàng đợi cho mỗi ngõ vào, một hàng đợi cho mỗi ngõ ra hay cả hai. Nó cũng liên quan đến trình tự xử lý các gói tin trong hàng đợi (round-robin hay dựa trên sự ưu tiên). Chính sách hủy bỏ gói tin sẽ chỉ ra gói tin nào cần bị hủy bỏ khi không còn không gian chứa. Một chính sách tốt có thể giúp làm giảm tắc nghẽn, ngược lại có thể làm tắc nghẽn trầm trọng thêm.

Một giải thuật vạch đường tốt có thể giúp tránh được tắc nghẽn bằng cách trải đều giao thông trên tất cả đường nối, trong khi một giải thuật tồi chỉ đơn giản gởi quá nhiều thông tin lên một đường tải đã quá tải rồi. Cuối cùng, việc quản lý thời gian sống của gói tin sẽ phải đưa ra quyết định là một gói tin có thể sống bao lâu trong hàng đợi trước khi bị hủy bỏ. Thời gian sống quá dài sẽ làm

trì trệ công việc rất lâu. Nhưng nếu thời gian sống quá ngắn, các gói tin thỉnh thoảng sẽ bị mãn kỳ (timed-out) trước khi chúng đến được đích, vì thế dẫn đến việc tái truyền.

6.4.3 Điều khiển tắc nghẽn trong các mạng con dạng mạch ảo

Một giải pháp đơn giản là điều khiển cấp phép (admission control). Ý tưởng như sau: một khi có cảnh báo về tắc nghẽn, hệ thống sẽ không thiết lập thêm mạch ảo nào nữa đến khi sự cố qua đi. Vì thế, trong lúc tắc nghẽn xảy ra, những cố gắng thiết lập mạch ảo đều thất bại. Lý do: cho phép nhiều người vào đấy sẽ làm cho vấn đề trở nên trầm trọng hơn.
Cách tiếp cận khác là cho phép tạo ra các mạch ảo mới nhưng cẩn trọng vạch đường cho các mạch ảo mới này đi vòng qua khu vực bị vấn đề tắc nghẽn. Ví dụ, xem xét mạng con như trong hình H6.20, trong đó hai router bị tắc nghẽn.



H6.20 (a) Một mạng con bị tắc nghẽn.
(b) Mạng con được vẽ lại sau khi loại trừ các điểm gây tắc nghẽn.

Giả sử một host được nối với router A muốn thiết lập nối kết tới một host của router B. Thường thì nối kết này sẽ chạy qua một trong hai nút bị tắc nghẽn. Để tránh chuyện này, chúng ta vẽ lại mạng con như trong hình (b), bỏ qua các router bị tắc nghẽn cùng với các đường nối của chúng. Đường chấm chỉ ra một đường đi có thể tránh được tắc nghẽn.
Một chiến lược khác liên quan đến mạch ảo là: host và mạng con thỏa thuận với nhau về việc thiết lập mạch ảo. Thỏa thuận này thường bao gồm dung lượng và đường đi của thông tin, chất lượng dịch vụ được yêu cầu và các thông số khác. Để đảm bảo thực hiện được thỏa thuận, mạng con sẽ dành riêng tài nguyên trên suốt con đường mạch ảo đi qua. Các tài nguyên này bao gồm không gian bảng vạch đường và buffer trên các router, cùng với băng thông trên các đường nối. Trong tình huống này, tắc nghẽn hầu như không xảy ra trên một mạch ảo mới bởi vì tất cả tài nguyên cần thiết đã được đảm bảo sẵn dùng.
Kiểu dành riêng tài nguyên này có thể được thực hiện toàn thời gian như là một phương thức hoạt động chuẩn, hoặc chỉ được thực hiện khi tắc nghẽn xảy ra. Nếu được thực hiện toàn thời gian sẽ có hạn chế là lãng phí tài nguyên. Nếu đường truyền 6 Mbps được tận hiến cho 6 mạch ảo, mỗi mạch ảo tiêu tốn 1 Mbps, thì đường truyền này luôn được đánh dấu là đầy, cho dù hiếm có khi nào 6 mạch ảo con của nó truyền hết công suất tại cùng thời điểm.

6.4.4 Điều khiển tắc nghẽn trong mạng con dạng Datagram

Trong mạng dạng Datagram, mỗi router có thể dễ dàng kiểm soát hiệu năng của các đường ra và các tài nguyên khác. Ví dụ, nó có thể gán cho mỗi đường nối một biến thực u, với giá trị từ 0.0 đến 1.0, dùng phản ánh hiệu năng gần đây của đường nối đó. Để duy trì độ chính xác tốt cho u, một mẫu hiệu năng tức thời f của đường nối sẽ được lấy thường xuyên, và u sẽ được cập nhật như sau

$$u_{mới} = a\, n_{cũ} + (1 - a)\, f$$

trong đó hằng số a quyết định router quên đi lịch sử gần đây nhanh như thế nào.

Khi *u* vượt qua ngưỡng, đường ra rơi vào trạng thái "cảnh báo". Mỗi gói tin mới tới sẽ được giữ lại và chờ kiểm tra xem đường ra có ở trạng thái cảnh báo không. Nếu có, một số hành động sẽ được thực hiện, và chúng ta sẽ thảo luận ngay sau đây.

6.4.4.1 Các gói tin chặn (Choke Packets)

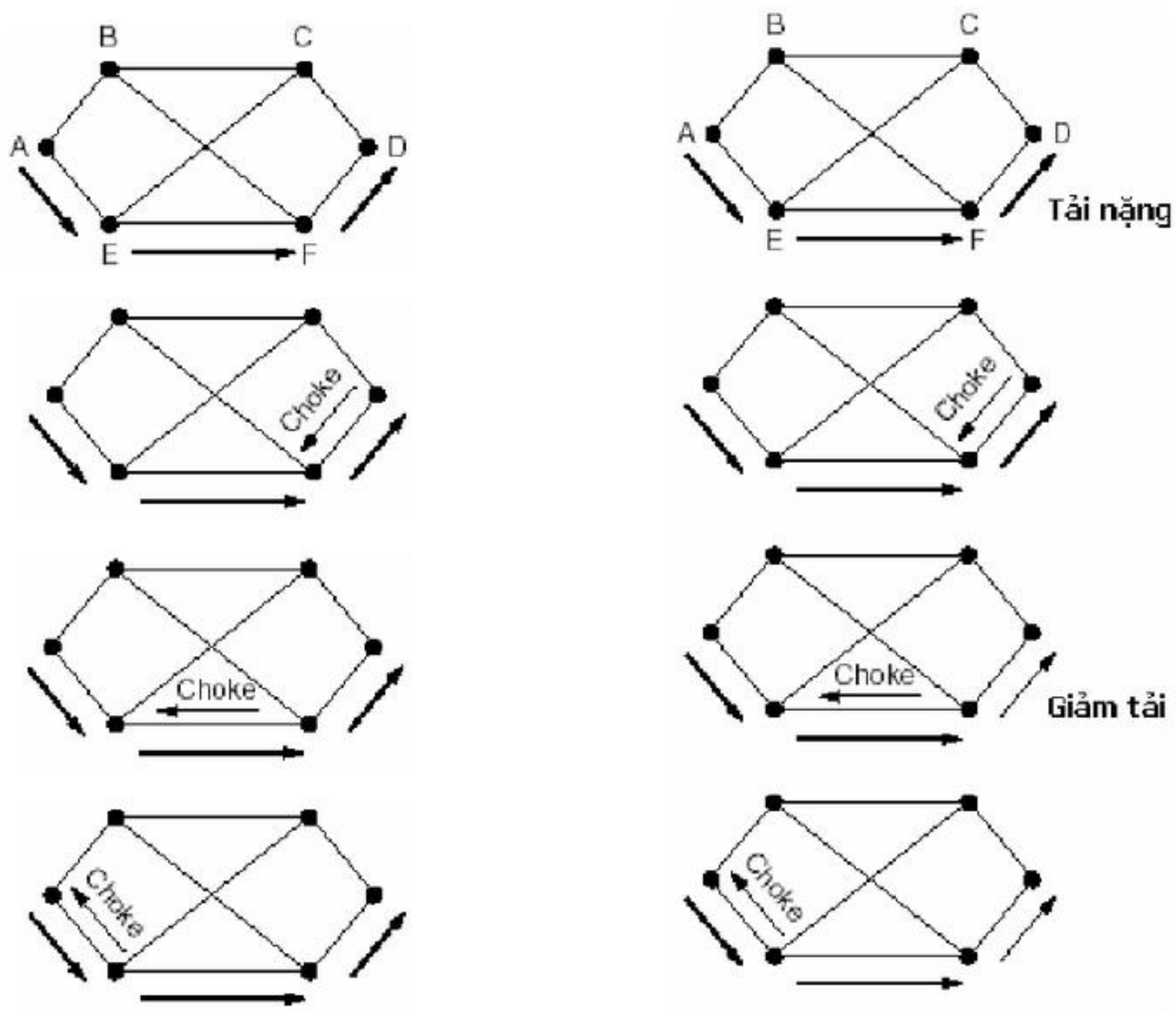
Khi một gói tin đến router và ngõ ra của nó đang ở trong trạng thái báo động, router sẽ gởi một gói tin chặn ngược về nút nguồn đã gởi gói tin đó. Gói tin gặp tắc nghẽn như đã nói sẽ được đánh dấu để nó không làm phát sinh các gói tin chặn khác nữa. Khi gói tin chặn đến được nút nguồn, nút nguồn sẽ giảm lưu lượng thông tin đến điểm bị nghẽn đi X phần trăm. Do có thể còn vài gói tin đang trên đường đi đến đích bị nghẽn, sau này nút nguồn nên bỏ qua các gói tin chặn phát ra tiếp từ đích đó.

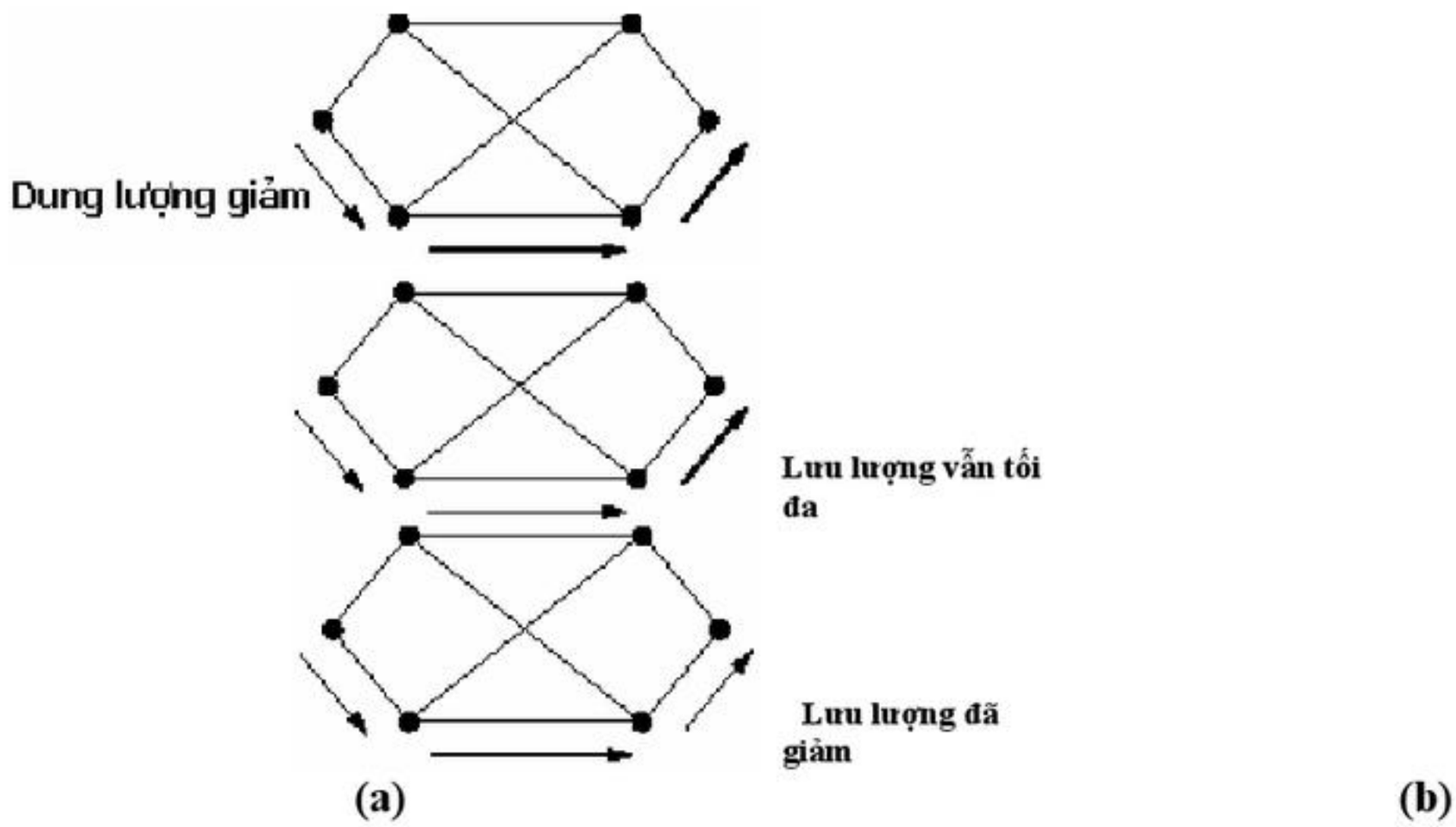
Sau giai đoạn trên, nút nguồn bỏ thêm một khoảng thời gian để lắng nghe thêm các gói tin chặn khác. Nếu chúng còn tới, đường nối vẫn bị nghẽn, nút nguồn tiếp tục giảm dung lượng truyền. Nếu không còn gói tin chặn nào chạy ngược về nút nguồn trong thời gian lắng nghe, nó có thể từng bước tăng lưu lượng truyền lên.

6.4.4.2 Gởi các gói chặn từng bước một (Hop-by-Hop Choke Packets)

Ở tốc độ cao hoặc qua khoảng cách xa, việc gởi gói tin chặn ngược về nút nguồn là không hiệu quả, bởi vì phản ứng của nút nguồn sẽ chậm.

Một cách tiếp cận khác là làm cho gói tin chặn có tác dụng tại mọi nút trung gian mà nó đi qua. Hãy xem hình ví dụ 5.18(b).





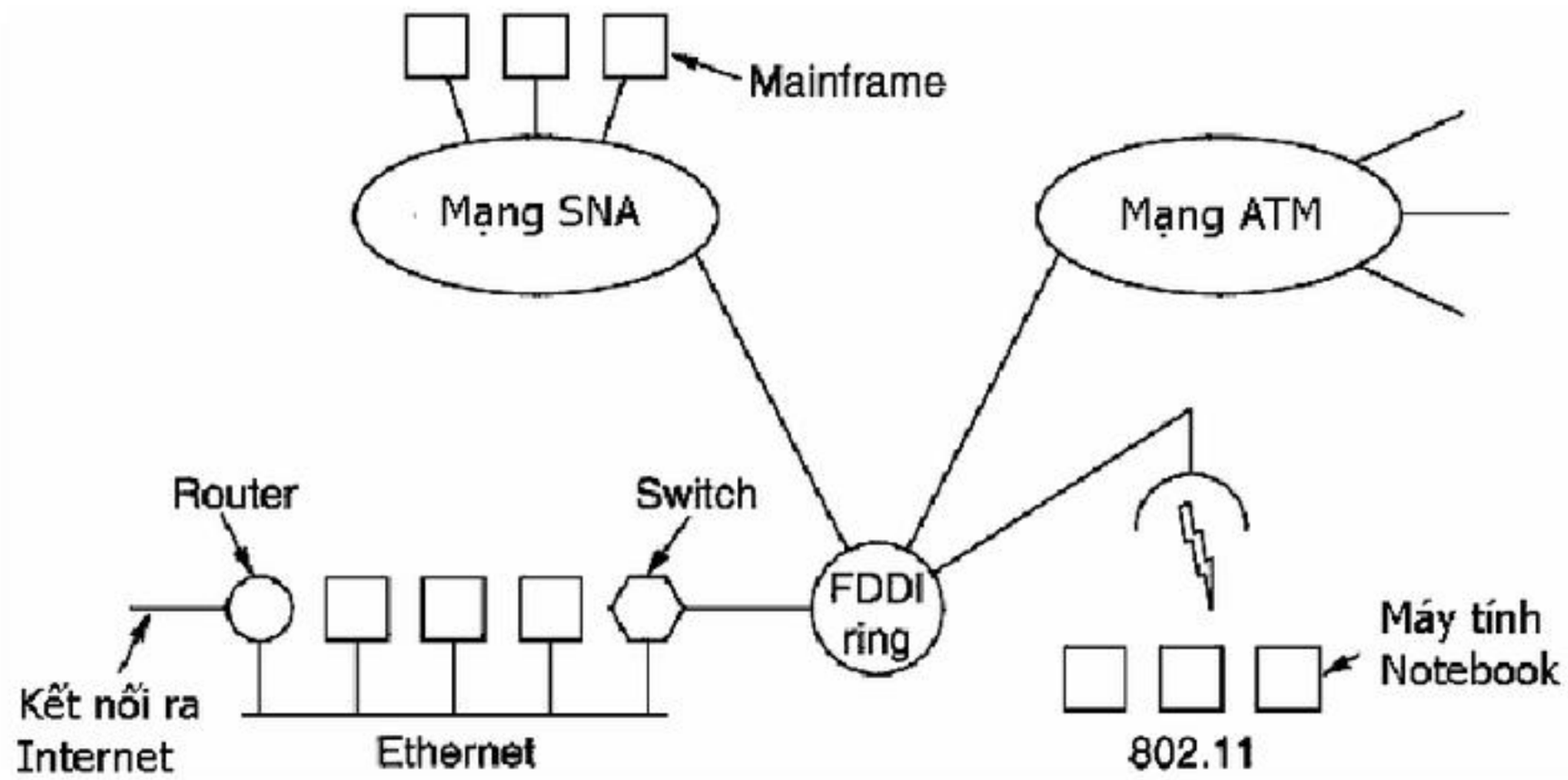
H6.21 (a) Một gói tin chặn chỉ tác động lên nút nguồn.
(b) Một gói tin chặn tác động lên mọi nút mà nó đi qua

Ở trong hình 5.18(b), ngay khi gói tin chặn vừa đến F, F liền giảm lưu lượng truyền đến D. Tương tự, khi gói tin chặn đến E, E sẽ giảm lưu lượng truyền đến F. Cuối cùng gói tin chặn đến A và lưu lượng được giảm suốt tuyến đường từ A đến D.
Hiệu quả của sơ đồ chặn từng bước một là có thể giải phóng điểm bị nghẽn nhanh chóng. Tuy nhiên cái giá phải trả là nó tiêu tốn băng thông hướng lên cho gói tin chặn. Nhưng cái lợi cuối cùng là ở chỗ, giải pháp này bóp chết tắc nghẽn ngay trong trứng nước.

6.5 Liên mạng

Đến thời điểm này, chúng ta đều ngầm định rằng chúng ta đang làm việc trên một mạng đơn đồng nhất với mọi máy tính chạy cùng một giao thức trong cùng một tầng. Không may là sự ngầm hiểu này hơi quá lạc quan. Đã và đang tồn tại nhiều loại mạng khác nhau bao gồm LAN, WAN, MAN. Nhiều giao thức khác nhau đang được sử dụng rộng rãi trên nhiều tầng mạng khác nhau. Trong phần này, chúng ta sẽ có cái nhìn cẩn trọng hơn về các vấn đề phát sinh khi hai hoặc nhiều mạng được nối kết với nhau thành một liên mạng (internet).
Các mạng máy tính đã đa dạng và sẽ vẫn đa dạng, và có nhiều lý do lý giải cho nhận định này. Trước tiên, cơ sở để cài đặt các mạng là khác nhau. Gần như tất cả các máy PC đều cài đặt TCP/IP. Nhiều công ty lớn sử dụng máy mainframe của IBM sử dụng mạng SNA. Một số lượng lớn các công ty điện thoại đang điều hành các mạng ATM. Một số mạng LAN dùng cho các máy tính PC vẫn còn sử dụng Novell IPX hoặc AppleTalk. Cuối cùng, mạng không dây là một lĩnh vực đang phát triển rộng với nhiều giao thức hoạt động trong đó. Chiều hướng sử dụng mạng phức tạp này sẽ còn tiếp diễn nhiều năm nữa với nhiều lý do về tính kế thừa, kỹ thuật mới, và thực tế là không phải nhà sản xuất nào cũng thích thú với việc giúp cho khách hàng của họ dễ dàng chuyển đổi sang hệ thống của nhà sản xuất khác.
Thứ hai, do máy tính và thiết bị mạng ngày càng rẻ, cho nên cấp có thẩm quyền quyết định mua sắm mạng máy tính ngày càng xuống thấp trong cơ cấu các công ty, tổ chức. Nhiều công ty đưa ra chính sách: dự trù mua sắm trên 1 triệu USD do cấp quản lý cao nhất quyết định, mua sắm trên 100.000 USD do cấp trung quyết định, dưới 100.000 USD thì cấp trưởng bộ phận có toàn quyền quyết định. Vì thế, ví dụ, bộ phận kỹ thuật thì có thể cài đặt các máy trạm Unix chạy TCP/IP, còn bộ phận tiếp thị có quyền cài các máy Mac với giao thức AppleTalk.
Thứ ba, các mạng khác khau sử dụng các công nghệ hoàn toàn khác nhau. Vì thế sẽ không mấy ngạc nhiên nếu thấy một sản phẩm phần cứng mới thì cũng xuất hiện phần mềm mới đi kèm. Ví

dụ, một gia đình trung bình hiện nay trang bị mạng giống như một văn phòng trung bình ngày xưa: đầy các máy tính không thể nói chuyện với nhau. Nhưng ở tương lai không xa, đấy sẽ là nơi có đầy đủ điện thoại, TV, máy tính và các dụng cụ khác, tất cả được nối kết với nhau và có thể được điều khiển từ xa. Kỹ thuật mới này chắc chắn sẽ sinh ra một kiểu mạng mới với các giao thức mới.



H6.22 Một liên mạng

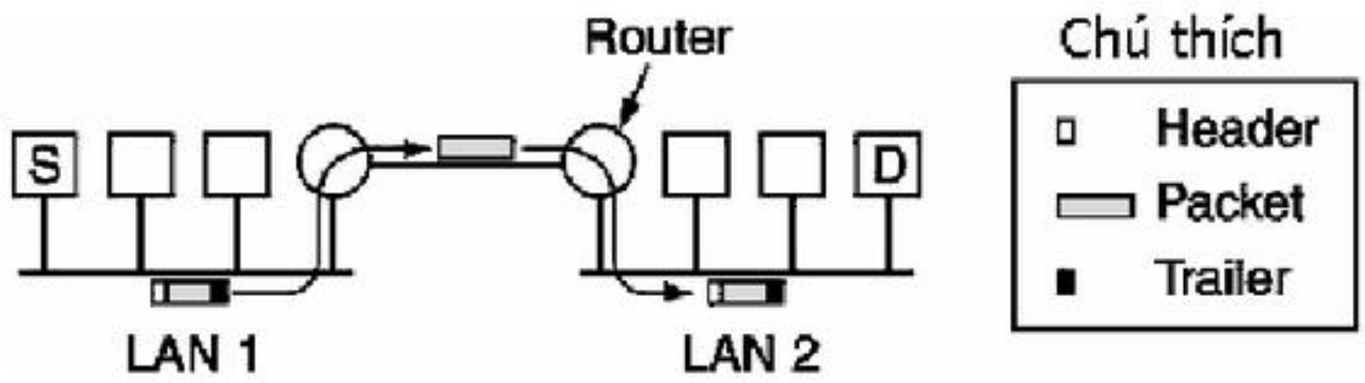
Để lấy ví dụ về cách thức các mạng khác nhau được nối kết với nhau như thế nào, hãy xem xét hình H6.22. Ở đây, ta có một mạng tổ hợp với nhiều địa bàn khác nhau, được kết dính với nhau bởi một mạng WAN/ATM. Tại một địa bàn, một back-bone FDDI được dùng để nối kết một mạng Ethernet, một mạng không dây 802.11 và một trung tâm dữ liệu dùng mạng SNA. Mục tiêu của nối kết liên mạng là cho phép người dùng trên một mạng con có thể liên lạc được với người dùng trên các mạng con khác. Để làm được việc này, ta phải đảm bảo gởi cho được gói tin từ mạng con này đến bất kỳ mạng con khác. Do các mạng con khác nhau về nhiều lĩnh vực, cho nên không dễ để truyền một gói tin từ nơi này đến nơi kia.

6.5.1 Các mạng con được nối kết với nhau ra sao?

Các mạng có thể được nối liên thông bằng nhiều kiểu thiết bị khác nhau:

- Ở tầng vật lý: Các mạng có thể được nối kết bằng các repeater hoặc hub, những thiết bị chỉ đơn thuần làm nhiệm vụ di chuyển các bit từ mạng này sang mạng kia.
- Ở tầng LKDL: Người ta dùng các cầu nối (bridges) hoặc switches. Chúng có thể nhận các khung, phân tích địa chỉ MAC và cuối cùng chuyển khung sang mạng khác trong khi song song đó, chúng vừa làm nhiệm vụ giám sát quá trình chuyển đổi giao thức, ví dụ như từ Ethernet sang FDDI hoặc 802.11.
- Ở tầng mạng: Người ta dùng các router để nối kết các mạng với nhau. Nếu hai mạng có tầng mạng khác nhau, router có thể chuyển đổi khuôn dạng gói tin, quản lý nhiều giao thức khác nhau trên các mạng khác nhau.
- Ở tầng vận chuyển: Người ta dùng các gateway vận chuyển, thiết bị có thể làm giao diện giữa hai đầu nối kết mức vận chuyển. Ví dụ gateway có thể làm giao diện trao đổi giữa hai nối kết TCP và NSA.
- Ở tầng ứng dụng: Các gateway ứng dụng sẽ làm nhiệm vụ chuyển đổi ngữ cảnh của các thông điệp. Ví dụ như gateway giữa hệ thống email Internet và X.400 sẽ làm nhiệm vụ chuyển đổi nhiều trường trong header của email.

Trong chương này, chúng ta chỉ quan tâm đến việc nối kết liên mạng ở tầng mạng dùng các router. Phương thức hoạt động của router được chỉ ra trong hình H5.23.

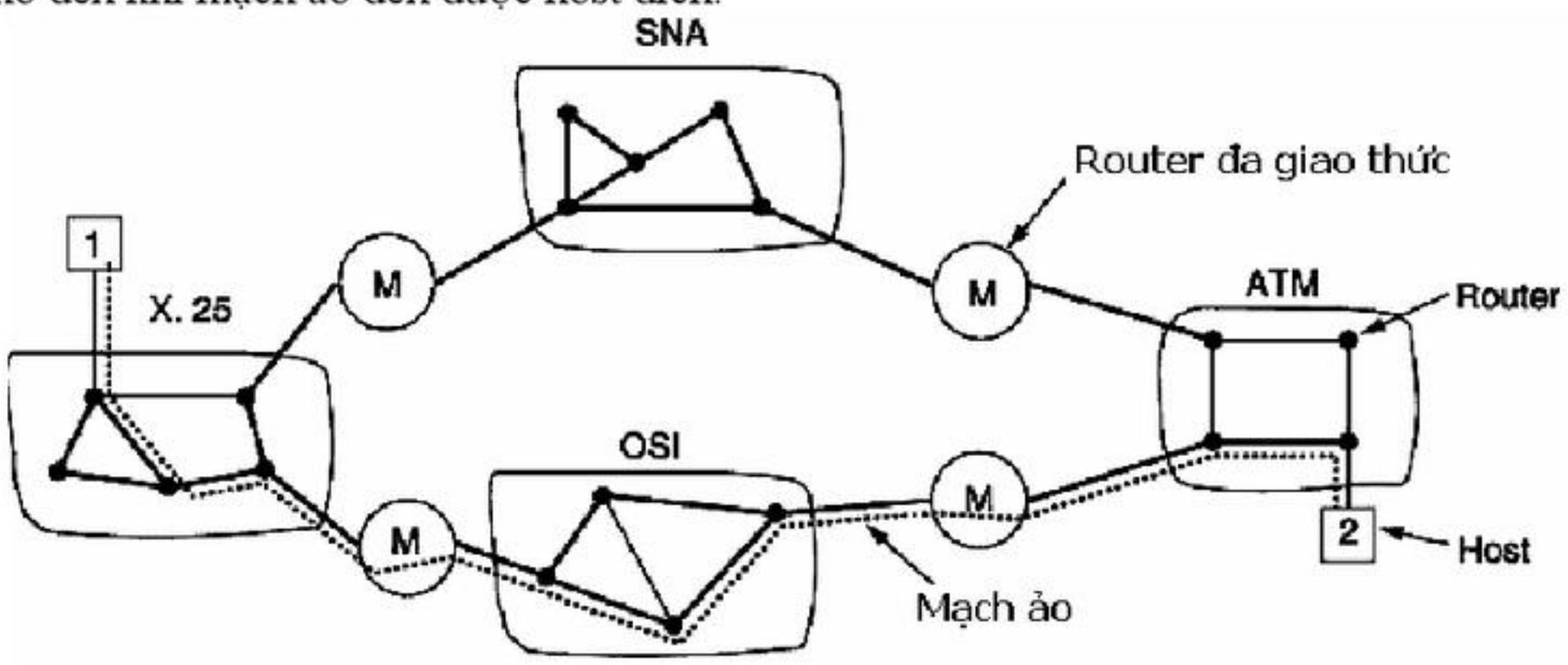


H6.23. Hai mạng Ethernet được nối kết bằng các routers

Ở đây, hai router được nối với nhau bằng đường nối điểm-điểm, có thể là đường leased-line dài hàng trăm km. Máy S muốn gởi cho máy D một gói tin, do đó nó đóng gói gói tin này thành một khung và gởi lên đường truyền. Khung đến được router của LAN1, router này liền bóc vỏ khung, lấy gói tin ra. Gói tin này sẽ được phân tích để tìm ra địa chỉ mạng (IP) của máy đích, địa chỉ này sẽ được tham khảo trong bảng vạch đường của router LAN1. Dựa trên địa chỉ này, router LAN1 quyết định chuyển gói sang router LAN2 bằng cách đóng thành khung gởi cho router LAN2.

6.5.2 Nối kết các mạng con dạng mạch ảo

Có hai kiểu liên mạng: dạng mạch ảo và datagram. Trong quá khứ, hầu hết các mạng công cộng là hướng nối kết (các mạng Frame Relay, SNA, ATM cũng vậy). Rồi với sự chấp nhận rộng rãi của công chúng đối với mạng Internet, mạng dạng datagram lên ngôi. Tuy nhiên sẽ là không chính xác khi nói mạng datagram là mãi mãi. Với sự phát triển quan trọng của các mạng đa phương tiện, có vẻ như liên lạc hướng nối kết đang trở lại ở dạng này hay dạng khác, do dễ đảm bảo chất lượng dịch vụ hơn. Vì thế cũng nên dành một chút thời gian để nói về mạng dạng mạch ảo.
Trong liên mạng dạng mạch ảo như trong hình H6.24, một nối kết tới một host ở mạng xa được thực hiện giống như truyền thống. Mạng con thấy rằng đích đến ở xa và nó sẽ phác thảo một mạch ảo đến router gần mạng đích nhất. Rồi nó tạo một mạch ảo từ router đây đến một gateway của nó (thực chất là router đa giao thức). Gateway này ghi lại thông tin về mạch ảo này trong bảng vạch đường và tiếp tục xây dựng một mạch ảo khác từ nó đến mạng con kế tiếp. Quá trình này cứ tiếp diễn cho đến khi mạch ảo đến được host đích.

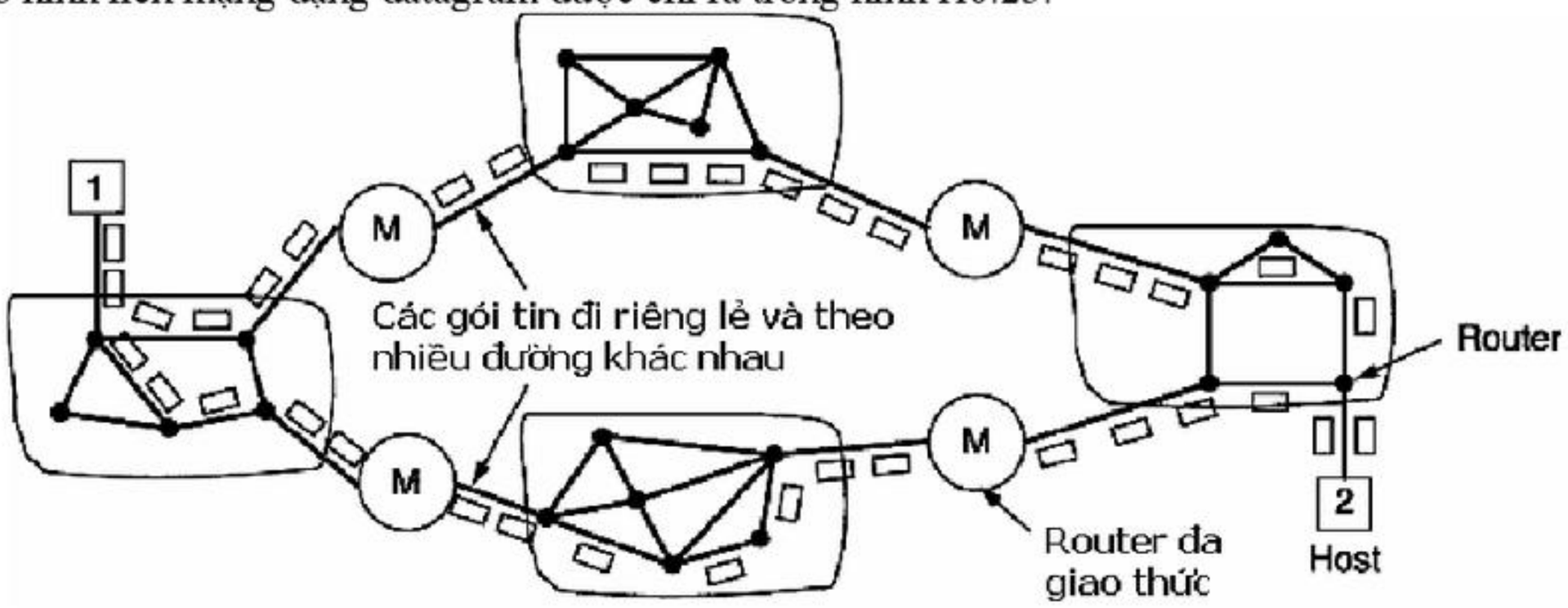


H6.24 Liên mạng dạng mạch ảo

Mỗi khi có một gói tin trung chuyển qua, một gateway lưu gói tin này lại, chuyển đổi khuôn dạng gói tin này và số hiệu mạch ảo khi cần thiết, rồi chuyển nó qua gateway tiếp theo trên con đường mà mạch ảo chỉ ra.
Đặc điểm quan trọng của cách làm này là: một dãy các mạch ảo được thiết lập từ host nguồn, qua một hoặc nhiều gateway rồi mới đến đích. Mỗi gateway duy trì các bảng lưu lại những mạch ảo nào đi qua nó, chúng sẽ được vạch đường ra đâu và số mạch ảo mới là gì.

6.5.3 Nối kết các mạng con dạng datagram

Mô hình liên mạng dạng datagram được chỉ ra trong hình H6.25.



Hình 6.25 Liên mạng dạng datagram

Trong mô hình này, dịch vụ duy nhất mà tầng mạng cung cấp cho tầng vận chuyển là khả năng đẩy gói tin vào mạng con và hy vọng nó đến đích. Mô hình này không đòi hỏi mọi gói tin phải đi đến đích trên cùng một con đường. Trong hình trên, các gói tin đi từ host 1 đến host 2 theo nhiều đường khác nhau. Quyết định vạch đường được đưa ra riêng lẻ cho từng gói tin, tùy thuộc vào lưu lượng thông tin tại thời điểm gói tin được gởi. Chiến lược này cho phép lựa chọn nhiều đường đi cho các gói tin và như vậy sẽ giúp đạt được nhiều băng thông hơn dạng mạch ảo. Mặt trái của nó là: không có sự đảm bảo gói tin có thể đến đích được.

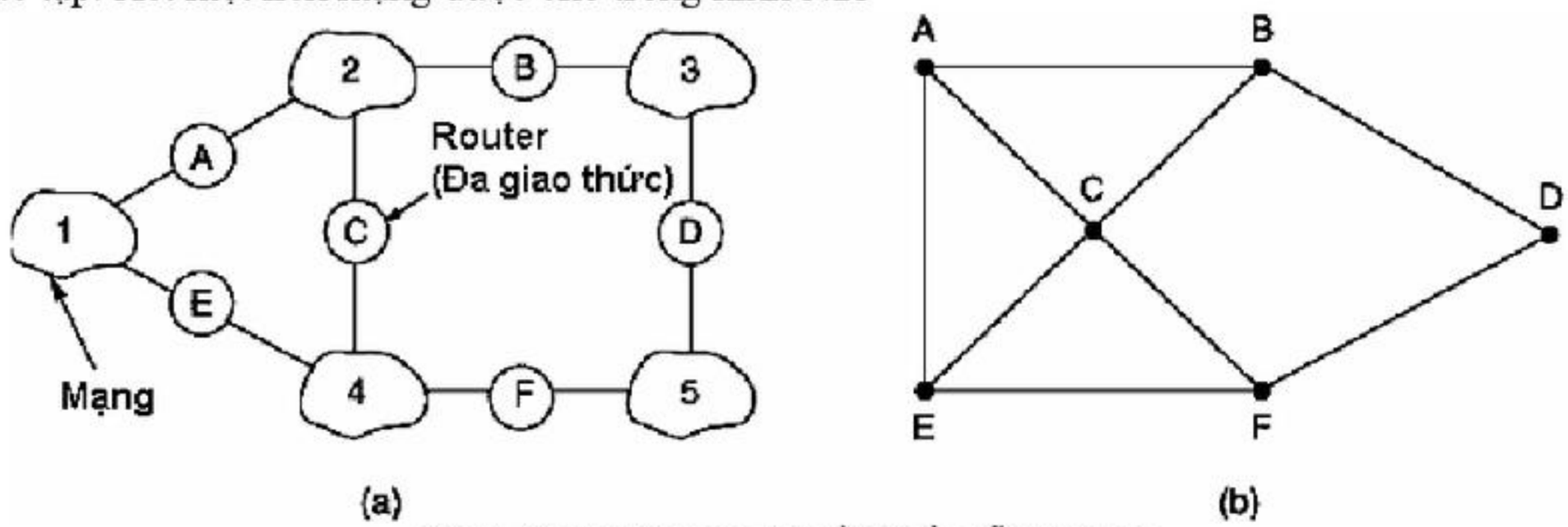
Mô hình như trong hình H6.25 thật ra không đơn giản như ta thấy. Thứ nhất, nếu mỗi mạng con có một tầng mạng riêng thì một gói tin từ một mạng không thể trung chuyển qua mạng khác được. Người ta có thể nghĩ là sẽ có các router đa giao thức làm nhiệm vụ chuyển đổi khuôn dạng gói tin từ dạng này sang dạng kia. Nhưng trừ khi dạng thức của hai gói tin có mối liên hệ gần với nhau và có cùng trường thông tin, không thì việc chuyển đổi khuôn dạng thường kết thúc thất bại. Với lý do này, giải pháp chuyển đổi khuôn dạng thường ít khi được chọn.

Thứ hai, nghiêm trọng hơn, là vấn đề về cơ chế định địa chỉ. Thử tưởng tượng một trường hợp đơn giản: một host trên Internet đang cố gởi một gói tin IP đến một host trong một mạng SNA láng giềng. Địa chỉ IP và SNA là khác nhau. Người ta sẽ cần một cơ chế ánh xạ địa chỉ giữa IP và SNA theo hai chiều. Thêm nữa, quan niệm như thế nào là địa chỉ trong hai mạng trên là hoàn toàn khác nhau. Trong IP, một host (thực ra là một card mạng) có một địa chỉ. Trong SNA, các thực thể khác host (ví dụ thiết bị phần cứng) cũng có thể có địa chỉ. Trong trường hợp tốt nhất, người ta có thể duy trì một cơ sở dữ liệu để ánh xạ mọi thứ này sang mọi thứ kia và ngược lại, và cơ sở dữ liệu này có thể mở rộng được, nhưng nó lại thường là ngọn nguồn của lắm rắc rối.

Một giải pháp khác là xây dựng gói tin internet có hiệu lực toàn cầu, và tất cả các router đều có thể hiểu được nó. Gói tin IP chính là sản phẩm của ý tưởng này. Tuy rằng việc kêu gọi mọi người chấp nhận một khuôn dạng duy nhất là khó do nhiều công ty có tham vọng xây dựng khuôn dạng độc quyền của mình, gói tin IP tự nó vẫn phát triển rộng rãi và được coi là chuẩn toàn cầu.

6.5.4 Vạch đường trong liên mạng

Vạch đường trong liên mạng cũng tương tự như trong mạng con đơn, nhưng thêm vào một chút phức tạp. Xét một liên mạng được cho trong hình 5.23



H6.26. (a) Một liên mạng. (b) Đồ thị biểu diễn liên mạng

Có sáu mạng con được nối kết với nhau bởi sáu router. Tạo ra mô hình đồ thị trong tình huống này là phức tạp do mỗi router đều có thể truy cập trực tiếp (gởi gói tin) đến router khác. Ví dụ, B trong hình 6.2H6(a) có thể nối kết trực tiếp tới A qua mạng 2, C qua mạng 2, và D qua mạng 3. Điều đó dẫn đến đồ thị như trong hình H6.26(b).

Một khi đồ thị đã được dựng lên, các giải thuật vạch đường đã biết, như Distance-Vector hoặc Link-State, có thể được áp dụng bởi các router đa giao thức. Như vậy sẽ dẫn đến giải thuật vạch đường hai mức: trong nội bộ một mạng con – vạch đường nội hạt (interior gateway protocol) và giữa các mạng con với nhau – vạch đường liên mạng (exterior gateway protocol). Thực tế, do các mạng con là độc lập với nhau, chúng có thể sử dụng các giải thuật khác nhau. Mỗi mạng con được gọi là một hệ thống tự trị (Autonomous System – AS).

Một gói tin liên mạng khởi đầu bằng địa chỉ LAN cục bộ của nó, được phát tới router đa giao thức nội bộ, tại đó, mã lệnh ở lớp mạng sẽ quyết định chuyển gói tin đó qua router nào kế tiếp. Nếu router đó có thể đối thoại được bằng giao thức của mạng nội bộ, thì gói tin sẽ được chuyển trực tiếp. Ngược lại, gói tin sẽ được đóng khung bằng giao thức của router đó trước khi chuyển tiếp. Tiến trình này cứ tiếp diễn cho đến khi gói tin đến được đích.

Một trong những điểm khác nhau giữa vạch đường liên mạng và vạch đường nội hạt là vạch đường liên mạng đòi hỏi việc đi qua nhiều lãnh địa quốc tế. Nhiều luật khác nhau được đặt ra. Chẳng hạn luật của Canada nói rằng: dữ liệu từ Canada gởi đến một nơi của Canada thì không được quá cảnh ra bên ngoài. Nghĩa là lưu thông từ Windsor, Ontario đến Vancouver không được quá cảnh sang Detroit của Mỹ, trong khi con đường ngang qua Detroit là con đường ngắn nhất và rẻ nhất.

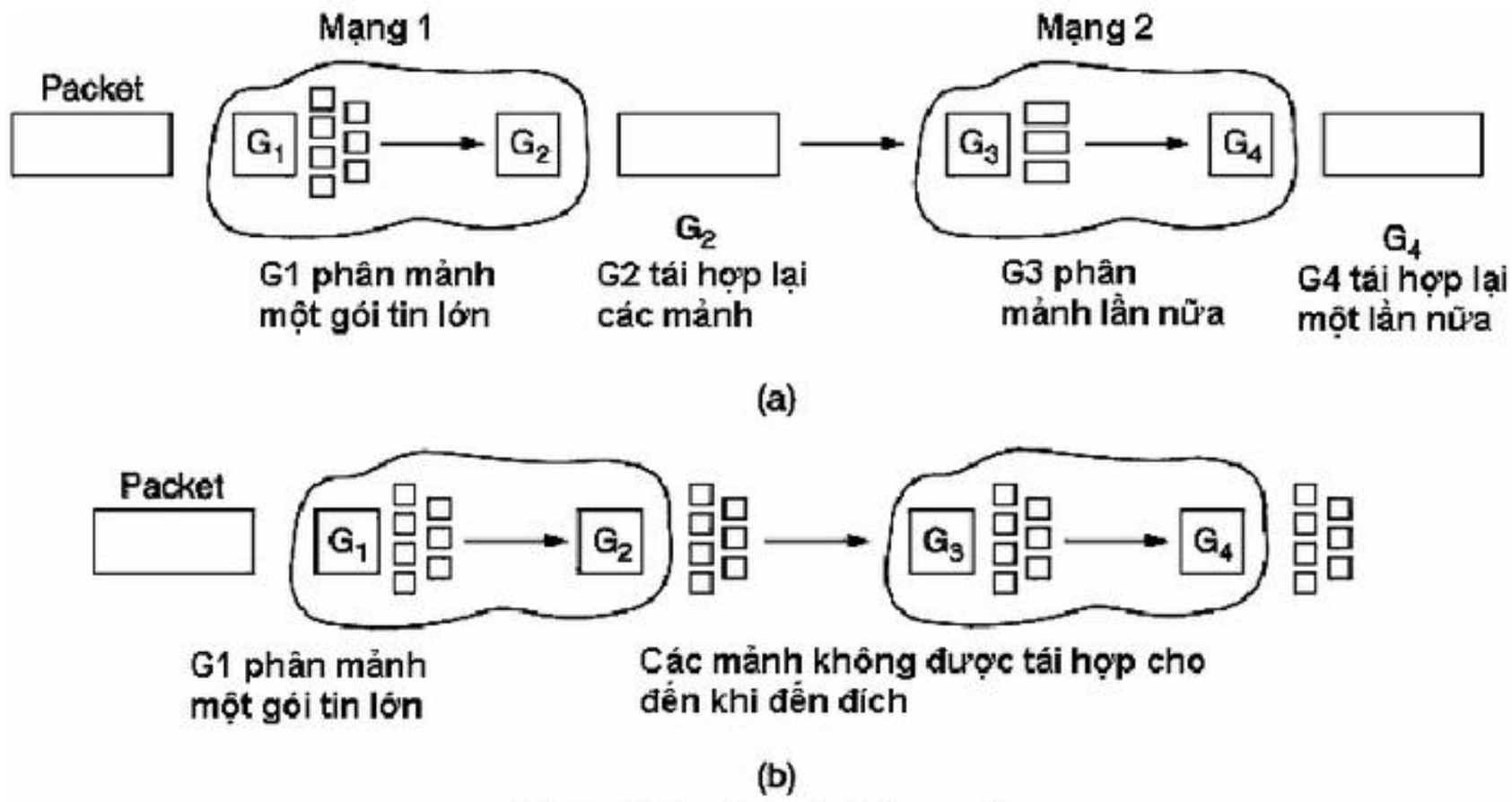
Điểm khác nhau nữa giữa vạch đường nội hạt và liên mạng là chi phí. Trong cùng một mạng, một giải thuật tính cước phí duy nhất được áp dụng. Tuy nhiên, nhiều mạng khác nhau sẽ có cơ chế quản lý cước phí khác nhau.

6.5.5 Phân mảnh và tái hợp

Mỗi mạng sẽ qui định kích cỡ tối đa của các datagram chạy trong nó. Sự giới hạn này xuất phát từ nhiều lý do:

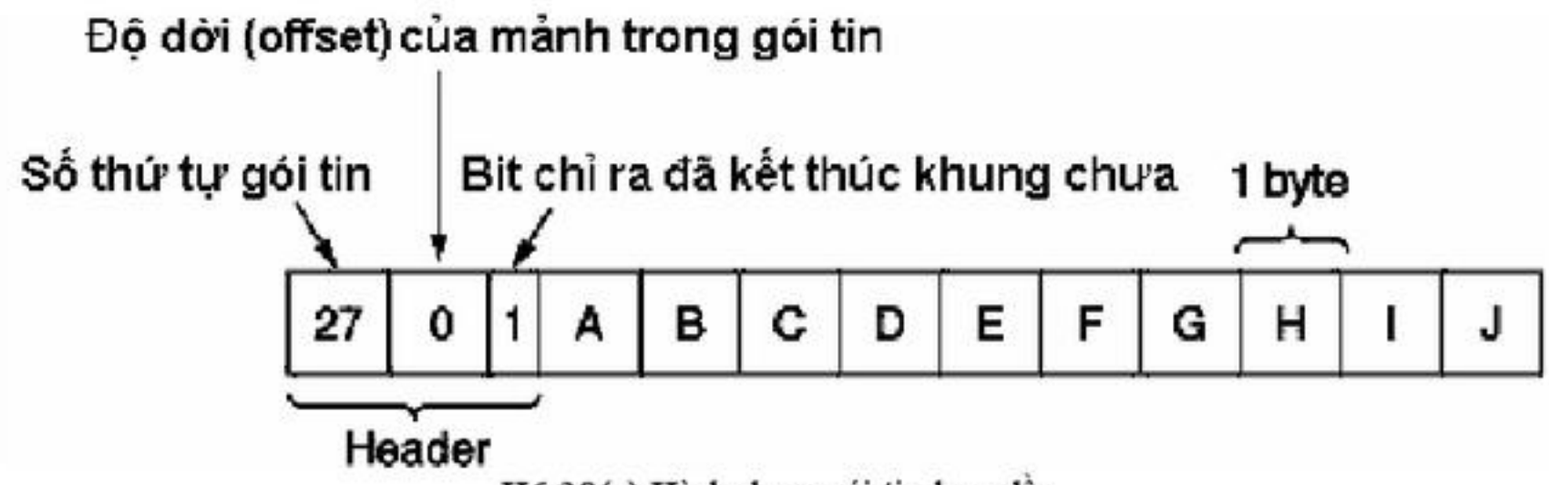
- Phần cứng: ví dụ như kích cỡ giới hạn của khung Ethernet.
- Hệ điều hành: ví dụ như tất cả các buffer đều có kích thước 512 bytes.
- Giao thức: số lượng các bits trong trường chỉ chiều dài của gói tin.
- Tương thích với một chuẩn quốc gia hay quốc tế nào đó.
- Mong muốn giảm thiểu tác động của việc truyền lại do lỗi gây ra.
- Mong muốn ngăn chặn một gói tin chiếm đường truyền quá lâu.

Và kết quả là các nhân viên thiết kế mạng không được tự do chọn kích thước gói tin tối đa như ý thích của họ. Kích thước dữ liệu tối đa của gói tin thay đổi từ 45 bytes (ATM cell) đến 65515 (gói tin IP).
Vấn đề xuất hiện rõ ràng khi môt gói tin lớn muốn đi ngang một mạng con có kích thước gói tin tối đa quá nhỏ. Một giải pháp là làm cho vấn đề này không xảy ra. Nói cách khác, liên mạng nên sử dụng một giải thuật vạch đường có thể tránh được việc gởi gói tin qua các mạng không có khả năng tiếp nhận. Tuy nhiên giải pháp này thực ra không giải quyết được vấn đề. Nếu mạng đích không đủ khả năng tiếp nhận gói tin thì sao?
Giải pháp duy nhất là cho phép các gateway chia nhỏ gói tin thành nhiều mảnh (fragment), gởi các mảnh này đi như là một gói tin độc lập. Tuy nhiên việc tái hợp các mảnh con này lại là việc khó.



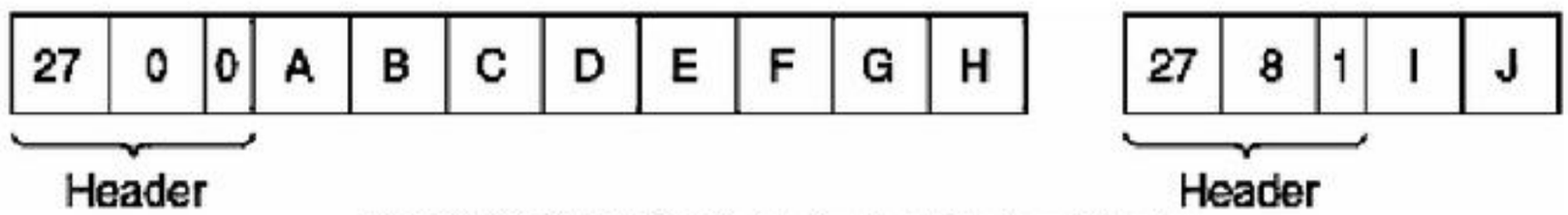
H6.27. (a) Sự phân mảnh trong suốt.
(b) Sự phân mảnh không trong suốt

Có hai chiến lược tái hợp các mảnh lại thành gói tin gốc: trong suốt và không trong suốt.
Trong chiến lược phân mảnh trong suốt, khi một gói tin lớn đi qua một mạng con và mạng con này quyết định phải phân mảnh gói tin, một gateway của mạng con này sẽ làm nhiệm vụ phân mảnh gói tin lớn đó. Khi các mảnh đi hết qua mạng con, phải có một gateway khác đứng ra tập hợp lại chúng, tái tạo lại gói tin ban đầu và chuyển tiếp đến mạng con kế tiếp. Ví dụ trong hình H6.27, ở ngõ vào Mạng 1, gói tin lớn được phân mảnh bởi gateway G1, sau khi các mảnh đi qua hết Mạng 1, gateway G2 sẽ tập hợp chúng lại và tái tạo thành gói tin ban đầu.
Chiến lược phân mảnh trong suốt rất trực quan, tuy nhiên có nhiều vấn đề phải bàn. Thứ nhất, gateway ở đầu ra phải biết khi nào nó đã thu lượm lại hết các phân mảnh. Thứ hai, làm sao để mọi phân mảnh phải đi ra cùng một gateway. Thứ ba, chi phí bỏ ra để phân mảnh và tái hợp gói tin lớn khi nó đi qua hàng loạt các mạng con.
Với chiến lược phân mảnh không trong suốt, các mạng con trung gian có thể phân mảnh gói tin lớn, nhưng không có nhiệm vụ tái hợp lại nó. Việc tái hợp chỉ được thực hiện tại đích đến của gói tin này.
Chiến lượt phân mảnh không trong suốt đòi hỏi mọi host trên mạng đều có khả năng tái hợp thông tin. Ngoài ra nó còn làm phát sinh chi phí cho các header của các mảnh con. Tuy nhiên cái lợi đạt được là: do chiến lược này có quyền chọn lựa nhiều gateway ở đầu ra của mạng con, cho nên hiệu suất vạch đường và truyền gói tin tăng lên nhiều lần.
Từ đây, phát sinh nhu cầu về một cách thức đánh số các mảnh sao cho quá trình tái hợp được dễ dàng. Một cách đánh số thông dụng nhất là cách đánh số của internet.
Ví dụ một gói tin có 10 bytes, số thứ tự của gói tin là 27 sẽ được biểu diễn như sau:



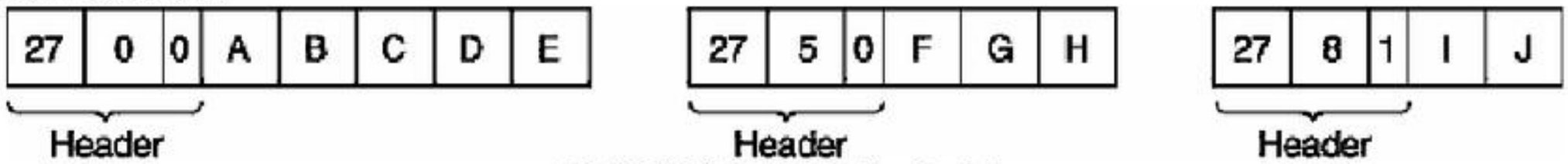
H6.28(a) Hình dạng gói tin ban đầu

Offset của mảnh là 0 vì đây chính là mảnh đầu tiên hay duy nhất trong gói tin. Bit kết thúc khung là 1 nghĩa là đã hết gói tin, là 0 nghĩa là còn mảnh nằm sau.
Bây giờ gói tin này đi qua một mạng con có giới hạn kích thước gói tin tối đa là 8 bytes, nó sẽ bị phân làm hai mảnh:



H6.28(b) Gói bị chia thành hai mảnh 8 bytes và 2 bytes

Mảnh đầu tiên có offset trong gói tin là 0, bit kết thúc là 0 (còn mảnh thứ hai). Mảnh thứ hai có offset trong gói tin là 8 (nó bắt đầu ở vị trí thứ 8), và là mảnh cuối cùng.
Nếu hai mảnh trên lại đi ngang qua gateway có giới hạn gói tin là 5 bytes, thì chúng sẽ bị phân mảnh như sau:

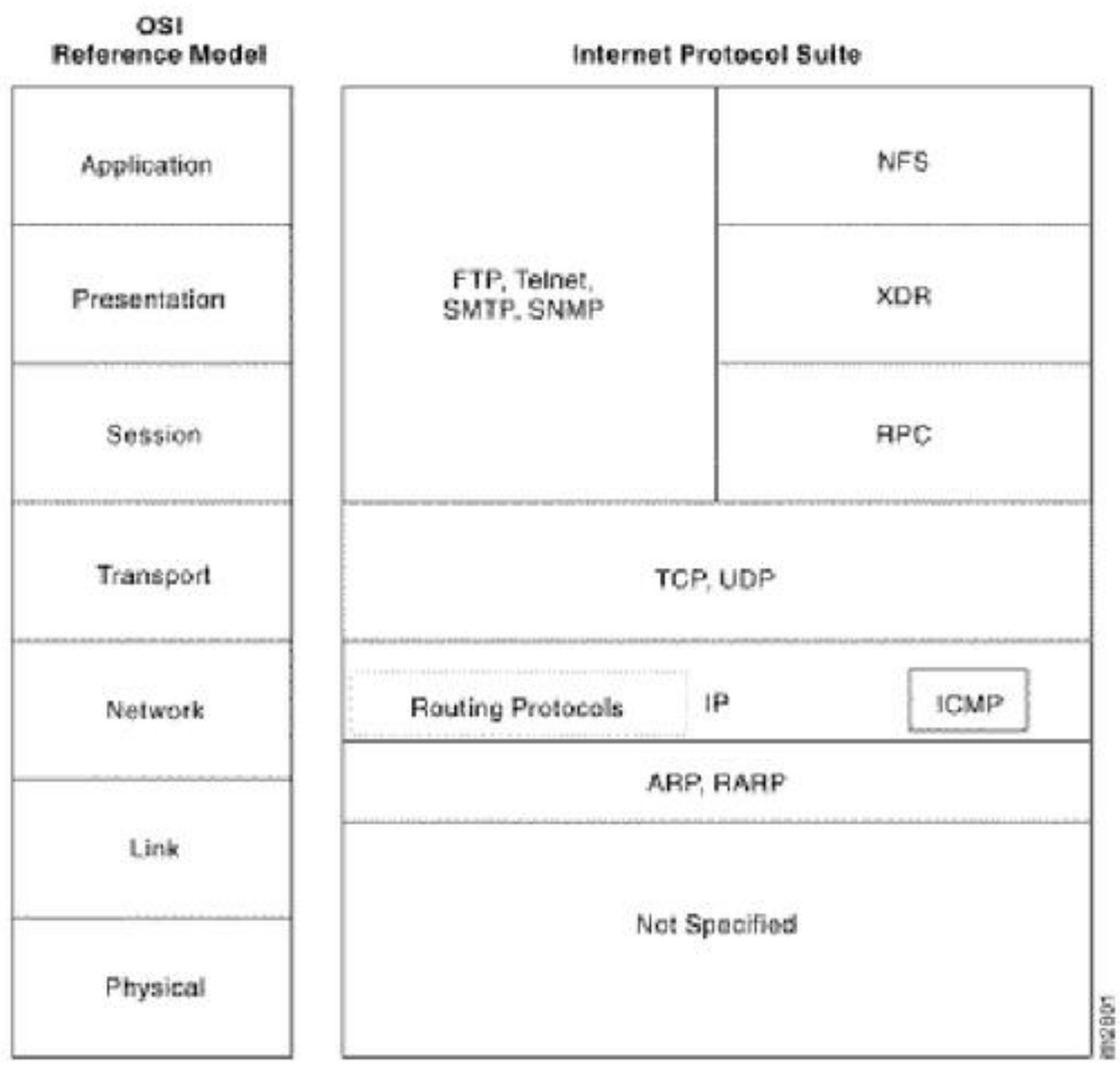


H6.28(c) Gói tin bị phân làm 3 mảnh.

6.6 Bộ giao thức liên mạng (IPs - Internet Protocols)

6.6.1 Giới thiệu

Các giao thức liên mạng là bộ giao thức cho các hệ thống mở nổi tiếng nhất trên thế giới bởi vì chúng có thể được sử dụng để giao tiếp qua bất kỳ các liên mạng nào cũng như thích hợp cho các giao tiếp trong mạng LAN và mạng WAN. Các giao thức liên mạng bao gồm một bộ các giao thức truyền thông, trong đó nổi tiếng nhất là Giao thức điều khiển truyền tải (TCP - Transmission Control Protocol) và Giao thức liên mạng (IP – Internet Protocol) hoạt động ở tầng 4 và tầng 3 trên mô hình OSI. Ngoài hai giao thức này, bộ giao thức IP còn đặc tả nhiều giao thức cho tầng ứng dụng, ví dụ như giao thức cho dịch vụ thư điện tử, giao thức mô phỏng thiết bị đầu cuối và giao thức truyền tải tập tin.
Bộ giao thức liên mạng lần đầu tiên được phát triển vào giữa những năm của thập niên 70 khi Văn phòng các dự án nghiên cứu chuyên sâu của bộ quốc phòng Mỹ (DARPA-Defense Advanced Research Projects Agency) quan tâm đến việc xây dựng một mạng chuyển mạch gói (packet-switched network) cho phép việc trao đổi thông tin giữa các hệ thống máy tính khác nhau của các viện nghiên cứu trở nên dễ dàng hơn. Với ý tưởng nối các hệ thống máy tính không đồng nhất lại với nhau, DARPA đã cấp kinh phí nghiên cứu cho đại học Stanford, Bolt, Beranek, and Newman (BBN) về vấn đề này. Kết quả của những nỗ lực phát triển của dự án này là bộ giao thức Liên mạng đã được hoàn thành vào những năm cuối của thập niên bảy mươi.
Sau đó TCP/IP được tích hợp vào hệ điều hành UNIX phiên bản BSD (Berkeley Software Distribution) trở thành nền tảng cho mạng Internet và dịch vụ WWW (World Wide Web).



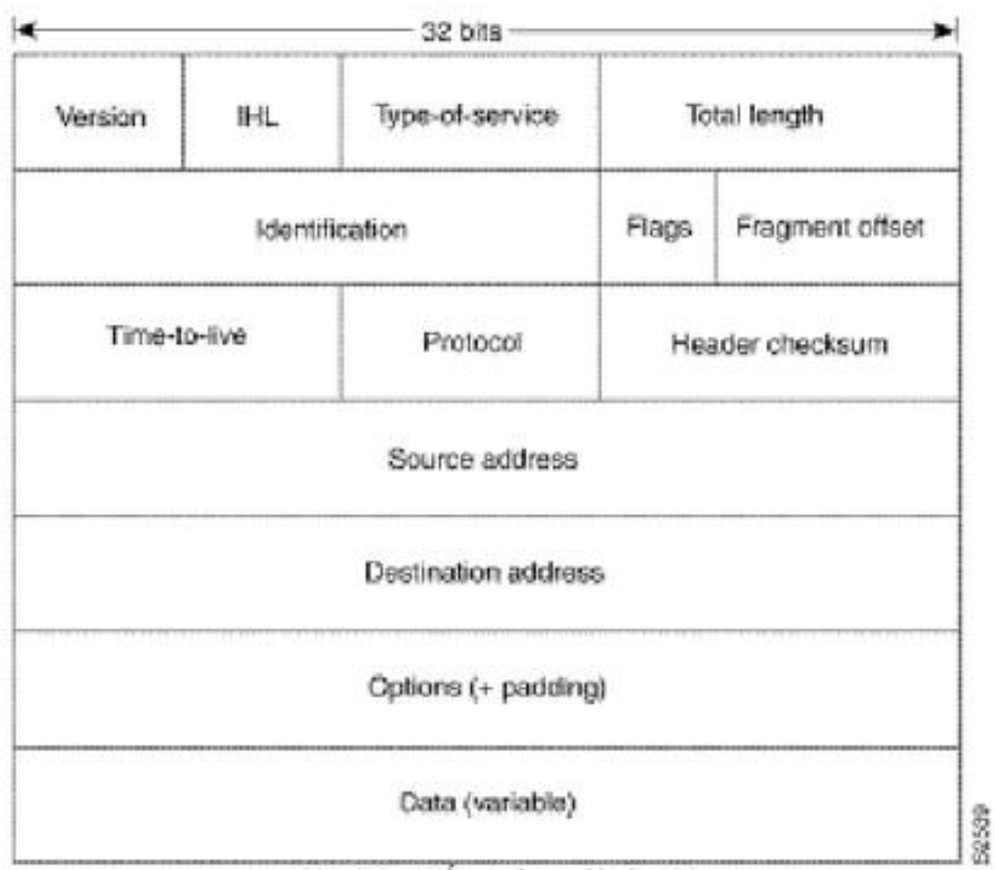
H6.29 Kiến trúc của mạng TCP/IP so với mô hình OSI

6.6.2 Giao thức liên mạng IP (Internet Protocol)

Giao thức liên mạng, thường gọi là giao thức IP (Internet Protocol) là một giao thức mạng hoạt động ở tầng 3 của mô hình OSI, nó qui định cách thức định địa chỉ các máy tính và cách thức chuyển tải các gói tin qua một liên mạng. IP được đặc tả trong bảng báo cáo kỹ thuật có tên RFC (Request For Comments) mã số 791 và là giao thức chủ yếu trong Bộ giao thức liên mạng. Cùng với giao thức TCP, IP trở thành trái tim của bộ giao thức Internet. IP có hai chức năng chính : cung cấp dịch vụ truyền tải dạng không nối kết để chuyển tải các gói tin qua một liên mạng ; và phân mãnh cũng như tập hợp lại các gói tin để hỗ trợ cho tầng liên kết dữ liệu với kích thước đơn vị truyền dữ liệu là khác nhau.

6.6.2.1 Định dạng gói tin IP (IP Packet Format)

Hình sau mô tả cấu trúc của một gói tin IP



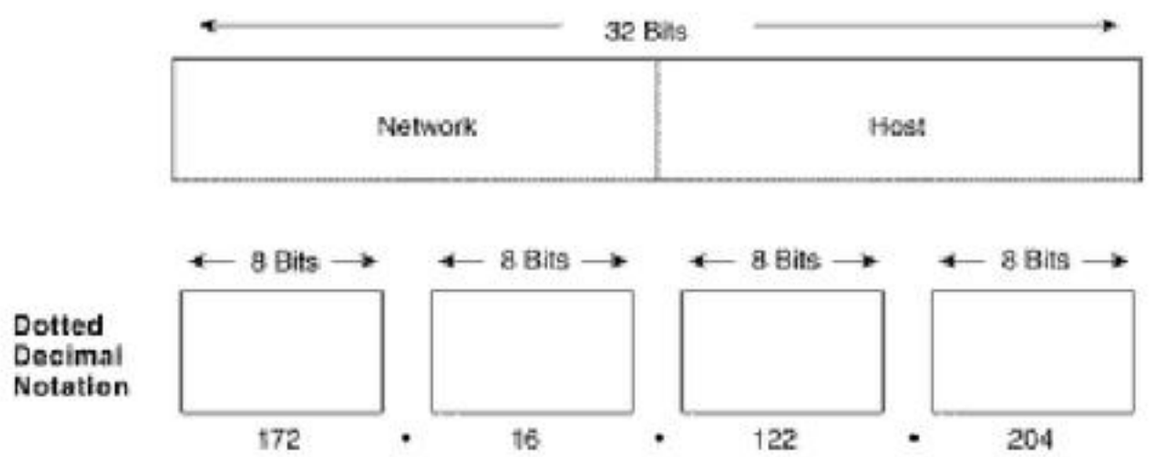
H6.30 Cấu trúc gói tin IP

Ý nghĩa của các trường được mô tả như sau:

- *Version (Phiên bản):* Xác định phiên bản của giao thức đang được sử dụng.
- *IP Header Length (Chiều dài của phần tiêu đề* : Xác định chiều dài của phần tiêu đề của gói tin, tính bằng đơn vị là từ - 32 bits (32-bit word).
- *Type-of-Service (Kiểu của dịch vụ :* Đặc tả mức độ quan trọng mà giao thức phía trên muốn xử lý gói tin.
- *Total Length (Tổng chiều dài gói tin):* Đặc tả chiều dài, tính bằng byte, của cả gói tin IP, bao gồm cả phần dữ liệu và tiêu đề.
- *Identification (Số nhận dạng):* Số nguyên nhận dạng gói tin dữ liệu hiện hành. Trường này được sử dụng để ráp lại các phân đoạn của gói tin.
- *Flags (Cờ hiệu):* Gồm 3 bít, bit có trọng số nhỏ để xác định gói tin có bị phân đoạn hay không. Bit thứ 2 xác định có phải đây là phân đoạn cuối cùng của gói tin hay không. Bit có trọng số lớn nhất chưa sử dụng.
- *Fragment Offset (Vị trí của phân đoạn):* Biểu thị vị trí của phân đoạn dữ liệu so với vị trí bắt đầu của gói dữ liệu gốc, nó cho phép máy nhận xây dựng lại gói tin ban đầu.
- *Time-to-Live (Thời gian sống của gói tin)*: Lưu giữ bộ đếm thời gian, giá trị sẽ được giảm dần đến khi nó có giá trị là 0 thì gói tin sẽ bị xóa. Điều này giúp ngăn ngừa tình trạng gói tin được truyền đi lòng vòng không bao giờ đến được đích.
- *Protocol(Giao thức):* Biểu hiện giao thức ở tầng trên sẽ nhận gói tin khi nó đã được giao thức IP xử lý.
- *Header Checksum (Tổng kiểm tra lỗi tiêu đề)*: kiểm tra tính toàn vẹn của phần tiêu đề.
- *Source Addres :* Địa của máy gởi gói tin.
- *Destination Address*: Địa chỉ của máy nhận gói tin.
- *Options*: Tùy chọn cho phép để hỗ trợ một số vấn đề, chẳng hạn vấn đề bảo mật.
- *Data*: Chứa dữ liệu của tầng trên gởi xuống cần truyền đi.

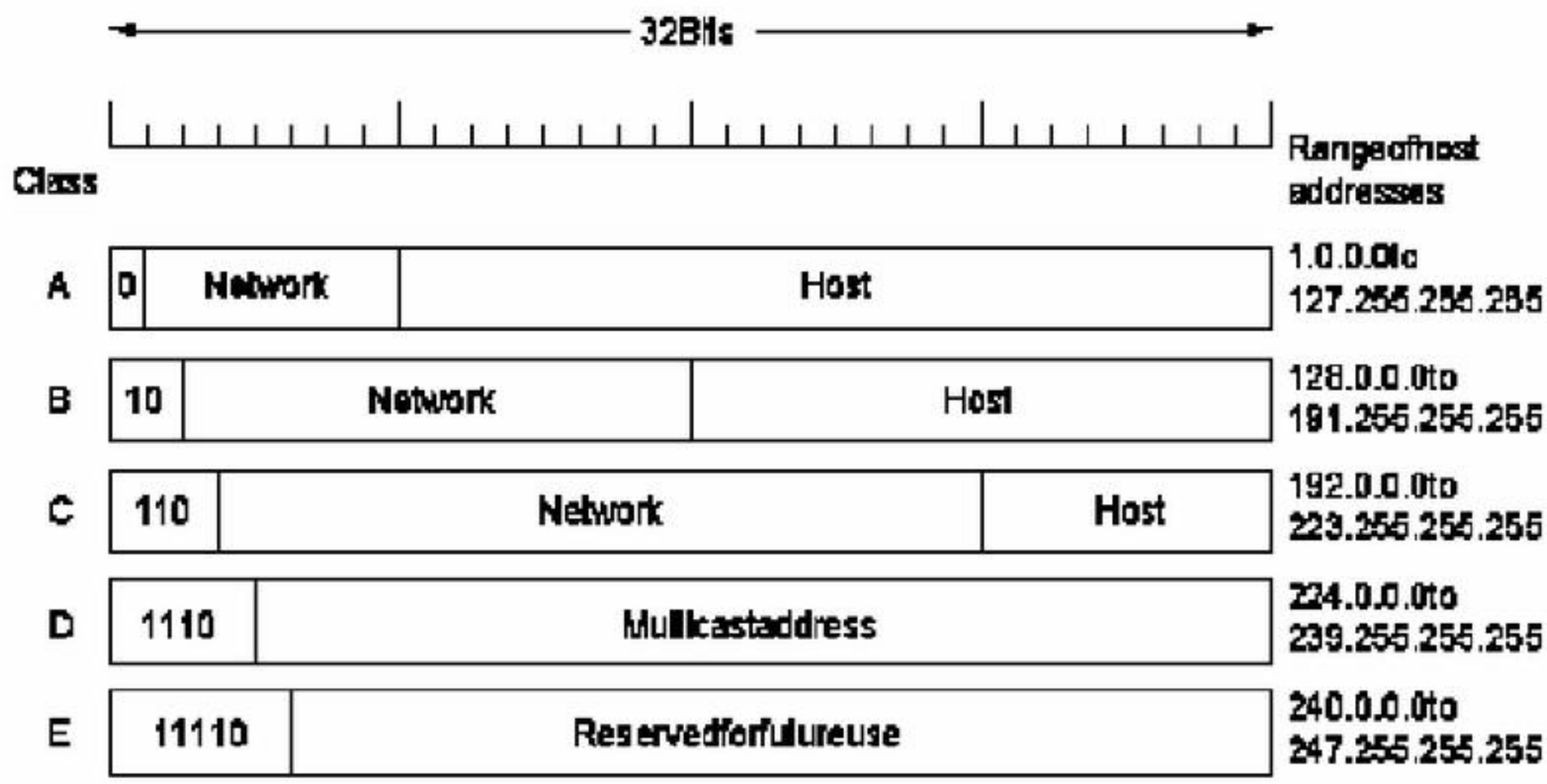
6.6.3 Cấu trúc địa chỉ IP

Mỗi máy tính trên mạng TCP/IP phải được gán một địa chỉ luận lý có chiều dài 32 bits, gọi là địa chỉ IP.



H6.31 Cấu trúc địa chỉ IP

32 bits của địa chỉ IP được chia thành 2 phần : Phần nhận dạng mạng (network id) và phần nhận dạng máy tính (Host id). Phần nhận dạng mạng được dùng để nhận dạng một mạng và phải được gán bởi Trung tâm thông tin mạng Internet (InterNIC - Internet Network Information Center) nếu muốn nối kết vào mạng Internet. Phần nhận dạng máy tính dùng để nhận dạng một máy tính trong một mạng.



H6.32 Phân lớp địa chỉ IP

Để dễ dàng cho việc đọc và hiểu bởi con người, 32 bits của địa chỉ IP được nhóm lại thành 4 bytes và được phân cách nhau bởi 3 dấu chấm (.). Giá trị của mỗi bytes được viết lại dưới dạng thập phân, với giá trị hợp lệ nằm trong khoản từ 0 đến 255.

Câu hỏi được đặt ra là bao nhiêu bits dành cho phần nhận dạng mạng và bao nhiêu bits dành cho phần nhận dạng máy tính. Người ta phân các địa chỉ ra thành 5 lớp : A, B, C, D và E. Trong đó, chỉ có lớp A, B và C được dùng cho các mục đích thương mại. Các bits có trọng số cao nhất chỉ định lớp mạng của địa chỉ. Hình sau mô tả cách phân chia lớp cho các địa chỉ IP.

Thông tin chi tiết về các lớp được mô tả như bảng sau :

Lớp	Dạng	Mục đích	Các bits cao nhất	Khoản địa chỉ	Số bít phần nhận dạng mạng / Số bít phần nhận dạng máy tính	Tổng số máy tính trong một mạng
A	N.H.H.H	Cho một số ít các tổ chức lớn	0	1.0.0.0 đến 126.0.0.0	7/24	16.777. 214 (2^{24} - 2)
B	N.N.H.H	Cho các tổ chức có kích thước trung bình	10	128.1.0.0 đến 191.254.0.0	14/16	65. 543 (2^{16} - 2)
C	N.N.N.H	Cho các tổ chức có kích thước nhỏ	110	192.0.1.0 đến 223.255.254.0	21/8	254 (2^{8} - 2)
D		Truyền nhóm	1110	224.0.0.0 đến 239.255.255.255		
E		Dành cho thí nghiệm	1111	240.0.0.0 đến 254.255.255.255		

Ghi chí H : Host, N=Network

6.6.4 ột số địa chỉ IP đặc biệt

- Địa chỉ mạng (Network Address): là địa chỉ IP mà giá trị của tất cả các bits ở phần nhận dạng máy tính đều là 0, được sử dụng để xác định một mạng.
  - Ví dụ : 10.0.0.0; 172.18.0.0 ; 192.1.1.0
- Địa chỉ quảng bá (Broadcast Address) : Là địa chỉ IP mà giá trị của tất cả các bits ở phần nhận dạng máy tính đều là 1, được sử dụng để chỉ tất cả các máy tính trong mạng.
  - Ví dụ : 10.255.255.255, 172.18.255.255, 192.1.1.255
- Mặt nạ mạng chuẩn (Netmask) : Là địa chỉ IP mà giá trị của các bits ở phần nhận dạng mạng đều là 1, các bits ở phần nhận dạng máy tính đều là 0. Như vậy ta có 3 mặt nạ mạng tương ứng cho 3 lớp mạng A, B và C là :
  - Mặt nạ mạng lớp A : 255.0.0.0
  - Mặt nạ mạng lớp B : 255.255.0.0
  - Mặt nạ mạng lớp C : 255.255.255.0

  Ta gọi chúng là các mặt nạ mạng mặc định (Default Netmask)

Lưu ý : Địa chỉ mạng, địa chỉ quảng bá, mặt nạ mạng không được dùng để đặt địa chỉ cho các máy tính

- Địa chỉ mạng 127.0.0.0 là địa chỉ được dành riêng để đặt trong phạm vi một máy tính. Nó chỉ có giá trị cục bộ (trong phạm vi một máy tính). Thông thường khi cài đặt giao thức IP thì máy tính sẽ được gián địa chỉ 127.0.0.1. Địa chỉ này thông thường để kiểm tra xem giao thức IP trên máy hiện tại có hoạt động không.
- Địa chỉ dành riêng cho mạng cục bộ không nối kết trực tiếp Internet : Các mạng cục bộ không nối kết trực tiếp vào mạng Internet có thể sử dụng các địa chỉ mạng sau để đánh địa chỉ cho các máy tính trong mạng của mình :
  - Lớp A : 10.0.0.0
  - Lớp B : 172.16.0.0 đến 172.32.0.0
  - Lớp C : 192.168.0.0

6.6.5 Ý nghĩa của Netmask

Với một địa chỉ IP và một Netmask cho trước, ta có thể dùng phép toán AND BIT để tính ra được địa chỉ mạng mà địa chỉ IP này thuộc về. Công thức như sau :
Network Address = IP Address & Netmask
Ví dụ : Cho địa chỉ IP = 198.53.147.45 và Netmask = 255.255.255.0. Ta thực hiện phép toán AND BIT (&) hai địa chỉ trên:

	Biểu diễn thập phân	Biểu diễn nhị phân
IP Address	198.53.147.45	11000110 00110101 10010011 00101101
Netmask	255.255.255.0	11111111 11111111 11111111 00000000
Network Address	198.53.147.0	11000110 00110101 10010011 00000000

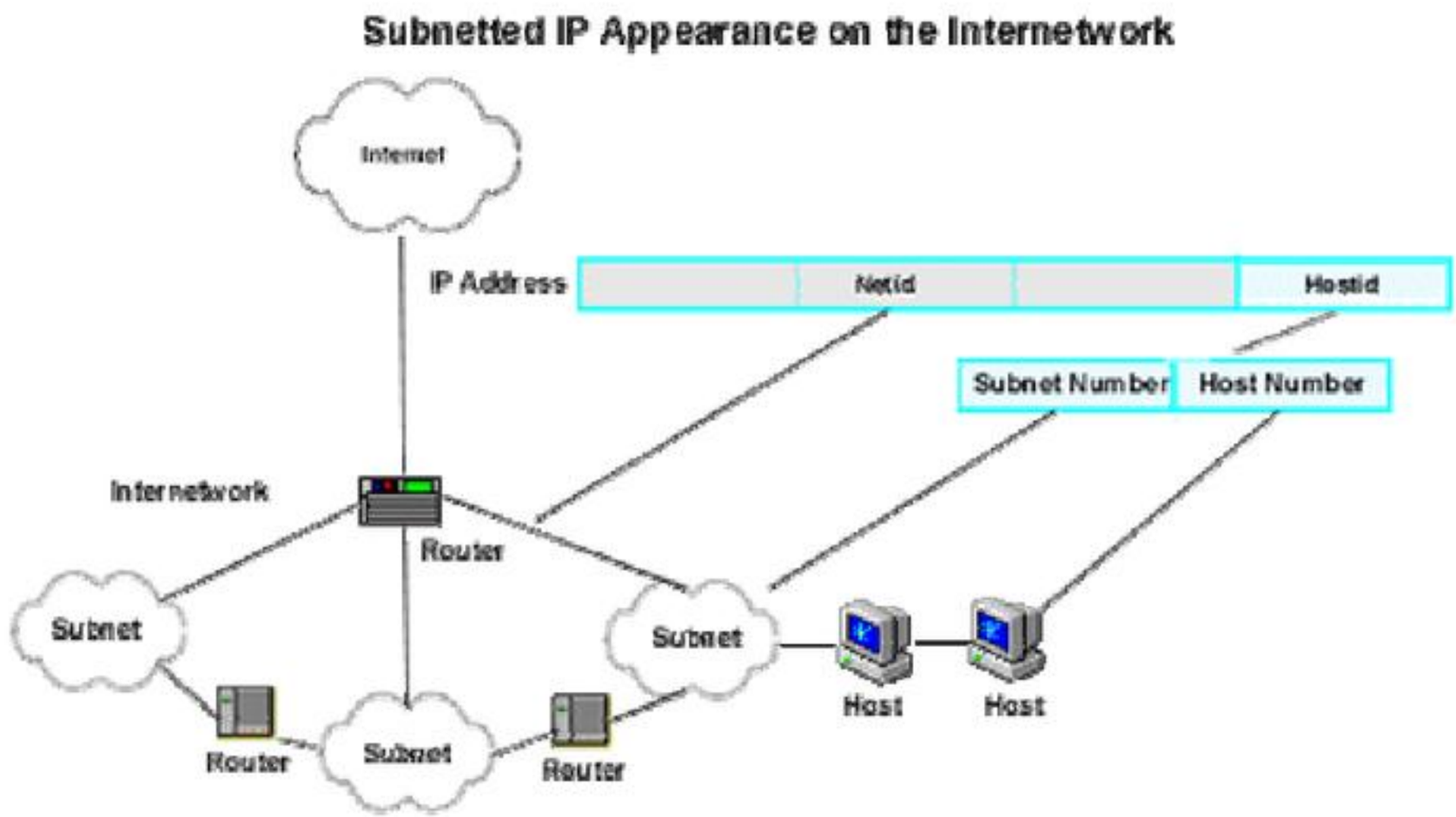
6.6.6 Phân mạng con (Subnetting)

6.6.6.1 Giới thiệu

Phân mạng con là một kỹ thuật cho phép nhà quản trị mạng chia một mạng thành những mạng con nhỏ, nhờ đó có được các tiện lợi sau :

- Đơn giản hóa việc quản trị : Với sự trợ giúp của các router, các mạng có thể được chia ra thành nhiều mạng con (subnet) mà chúng có thể được quản lý như những mạng độc lập và hiệu quả hơn.

- Có thể thay đổi cấu trúc bên trong của mạng mà không làm ảnh hưởng đến các mạng bên ngoài. Một tổ chức có thể tiếp tục sử dụng các địa chỉ IP đã được cấp mà không cần phải lấy thêm khối địa chỉ mới.
- Tăng cường tính bảo mật của hệ thống : Phân mạng con sẽ cho phép một tổ chức phân tách mạng bên trong của họ thành một liên mạng nhưng các mạng bên ngoài vẫn thấy đó là một mạng duy nhất.
- Cô lập các luồng giao thông trên mạng : Với sự trợ giúp của các router, giao thông trên mạng có thể được giữ ở mức thấp nhất có thể.



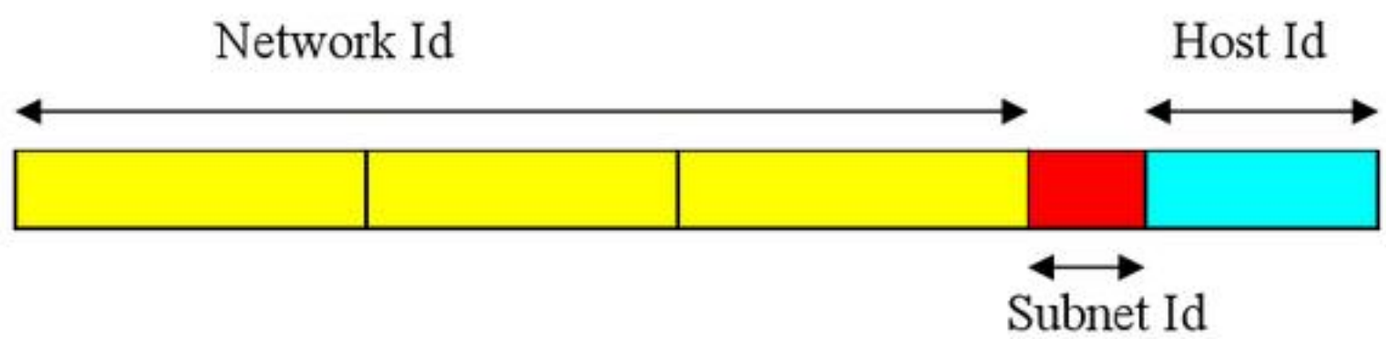
H6.33 Địa chỉ mạng con đối với thế giới bên ngoài

Hình trên mô tả một địa chỉ IP đã được phân mạng con xuất hiện với thế giới Internet bên ngoài và với mạng Intranet bên trong. Internet chỉ đọc phần nhận dạng mạng và các router trên Internet chỉ quan tâm tới việc vạch đường cho các gói tin đến được router giao tiếp giữa mạng Intranet bên trong và mạng Internet bên ngoài. Thông thường ta gọi router này là cửa khẩu của mạng (Gateway). Khi một gói tin IP từ mạng bên ngoài đến router cửa khẩu, nó sẽ đọc phần nhận dạng máy tính để xác định xem gói tin này thuộc về mạng con nào và sẽ chuyển gói tin đến mạng con đó, nơi mà gói tin sẽ được phân phát đến máy tính nhận.

6.6.6.2 Phương pháp phân mạng con

Nguyên tắc chung để thực hiện phân mạng con là :

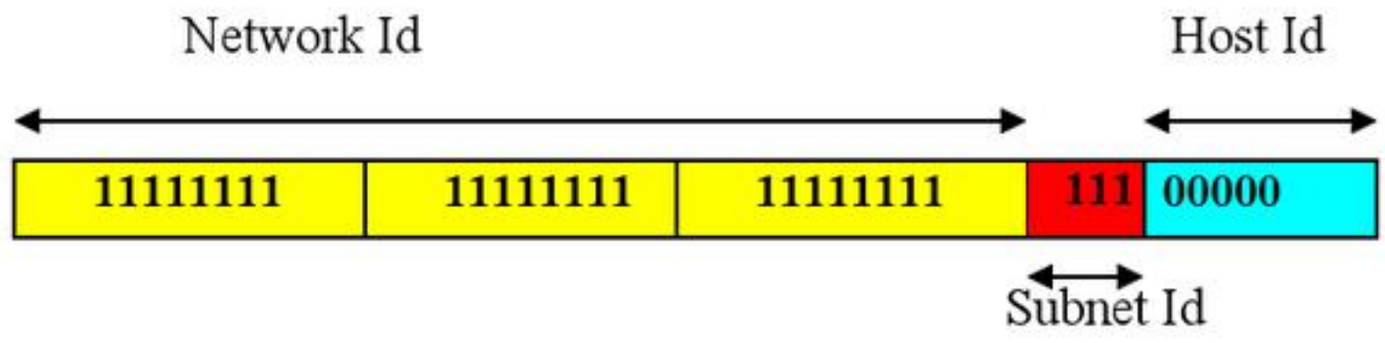
- Phần nhận dạng mạng (Network Id) của địa chỉ mạng ban đầu được giữ nguyên.
- Phần nhận dạng máy tính của địa chỉ mạng ban đầu được chia thành 2 phần : Phần nhận dạng mạng con (Subnet Id) và Phần nhận dạng máy tính trong mạng con (Host Id).



H6.33 Cấu trúc địa chỉ IP khi phân mạng con

Để phân mạng con, người ta phải xác định mặt nạ mạng con (subnetmask). Mặt nạ mạng con là một địa chỉ IP mà giá trị các bit ở phần nhận dạng mạng (Network Id) và Phần nhận dạng mạng

con (Subnet Id) đều là 1 trong khi giá trị của các bits ở Phần nhận dạng máy tính (Host Id) đều là 0. Hình H6.34 mô tả mặt nạ phân mạng con cho một mạng ở lớp C.



H6.34 Mặt nạ mạng con khi phân mạng con

Khi có được mặt nạ mạng con, ta có thể xác định địa chỉ mạng con (Subnetwork Address) mà một địa chỉ IP được tính bằng công thức sau :

Subnetwork Address = IP & Subnetmask

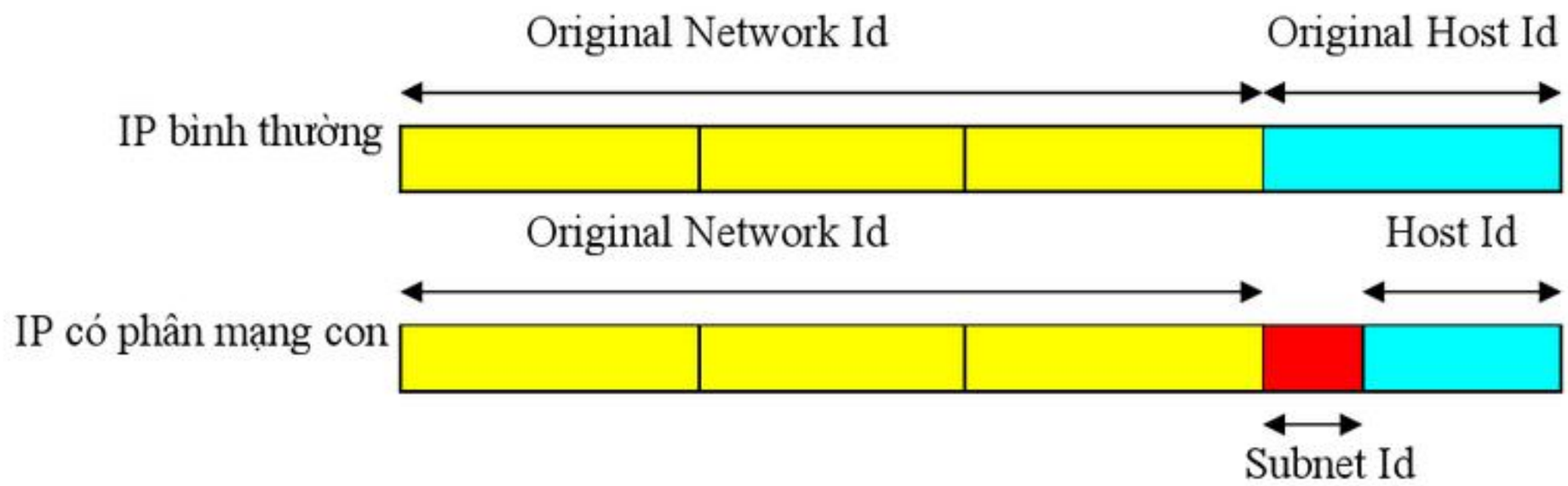
Có hai chuẩn để thực hiện phân mạng con là : Chuẩn phân lớp hoàn toàn (Classfull standard) và chuẩn Vạch đường liên miền không phân lớp CIDR (Classless Inter-Domain Routing). Thực tế, CIDR chỉ mới được đa số các nhà sản xuất thiết bị và hệ điều hành mạng hỗ trợ nhưng vẫn chưa hoàn toàn chuẩn hóa.

6.6.6.2.1 Phương pháp phân lớp hoàn toàn (Classfull Standard)

Chuẩn này qui định địa chỉ IP khi phân mạng con sẽ gồm 3 phần :

- Phần nhận dạng mạng của địa chỉ ban đầu (Network Id):
- Phần nhận dạng mạng con (Subnet Id) : Được hình thành từ một số bits có trọng số cao trong phần nhận dạng máy tính (Host Id) của địa chỉ ban đầu
- Và cuối cùng là phần nhận dạng máy tính trong mạng con (Host Id) bao gồm các bit còn lại.

Ví dụ : Hình sau mô tả cấu trúc địa chỉ IP lớp C khi thực hiện phân mạng con



H6.35 Địa chỉ IP phân mạng con theo chuẩn Phân lớp hoàn toàn

Số lượng bits thuộc phần nhận dạng mạng con xác định số lượng mạng con. Giả sử phần nhận dạng mạng con chiếm 4 bits. Như vậy, về mặt lý thuyết ta có thể phân ra thành 2^4=16 mạng con. Tuy nhiên phần nhận dạng mạng con gồm toàn bit 0 hoặc bit 1 không được dùng để đánh địa chỉ cho mạng con vì nó trùng với địa chỉ mạng và địa chỉ quảng bá của mạng ban đầu.

Ví dụ : Cho địa chỉ mạng lớp C : 192.168.1.0 với mặt nạ mạng mặc định là 255.255.255.0.

Xét trường hợp phân mạng con cho mạng trên sử dụng 2 bits để làm phần nhận dạng mạng con. Mặt nạ mạng trong trường hợp này là 255.255.255.192. Khi đó ta có các địa chỉ mạng con như sau :

Địa chỉ IP	Biểu diễn dạng thập phân	Biểu diễn dạng nhị phân			
Mạng ban đầu	192.168.1.0	1100 0000	1010 1000	0000 0001	0000 0000
Mạng con 1	192.168.1.0	1100 0000	1010 1000	0000 0001	**00**00 0000
Mạng con 2	192.168.1.64	1100 0000	1010 1000	0010 0001	**01**00 0000
Mạng con 3	192.168.1.128	1100 0000	1010 1000	0000 0001	**10**00 0000
Mạng con 4	192.168.1.192	1100 0000	1010 1000	0000 0001	**11**00 0000

Ta nhận thấy rằng:

- Địa chỉ mạng con thứ nhất 192.168.1.0 trùng với địa chỉ mạng ban đầu.
- Địa chỉ mạng con thứ tư 192.168.1.192 có địa chỉ quảng bá trùng với địa chỉ quảng bá của mạng ban đầu .

Chính vì thế mà hai địa chỉ này (có phần nhận dạng mạng con toàn bit 0 hoặc toàn bit 1) không được dùng để đánh địa chỉ cho mạng con.

Nói tóm lại, với **n** bits làm phần nhận dạng mạng con ta chỉ có thể phân ra được 2^n-2 mạng con mà thôi. Mỗi mạng con cũng có địa chỉ quảng bá. Đó là địa chỉ mà các bits ở phần nhận dạng máy tính đều có giá trị là 1.

Ví dụ :

Địa chỉ IP	Biểu diễn dạng thập phân	Biểu diễn dạng nhị phân			
Mạng con 1	192.168.1.64	1100 0000	1010 1000	0010 0001	**01**00 0000
Địa chỉ quảng bá	192.168.1.127	1100 0000	1010 1000	0010 0001	**01**11 1111
Mạng con 2	192.168.1.128	1100 0000	1010 1000	0000 0001	**10**00 0000
Địa chỉ quảng bá	192.168.1.191	1100 0000	1010 1000	0000 0001	**10**11 1111

Như vậy qui trình phân mạng con có thể được tóm tắt như sau :

- Xác định số lượng mạng con cần phân, giả sử là **N**.
- Biểu diễn (**N+1**) thành số nhị phân. số lượng bit cần thiết để biểu diễn (**N+1**) chính là số lượng bits cần dành cho phần nhận dạng mạng con. Ví dụ N=6, khi đó biểu diễn của (6+1) dưới dạng nhị phân là 111. Như vậy cần dùng 3 bits để làm phần nhận dạng mạng con
- Tạo mặt nạ mạng con
- Liệt kê tất cả các địa chỉ mạng con có thể, trừ hai địa chỉ mà ở đó phần nhận dạng mạng con toàn các bits 0 và các bit 1.
- Chọn ra N địa chỉ mạng con từ danh sách các mạng con đã liệt kê.

6.6.6.2.2 Phương pháp Vạch đường liên miền không phân lớp CIDR (Classless Inter-Domain Routing)

CIDR là một sơ đồ đánh địa chỉ mới cho mạng Internet hiệu quả hơn nhiều so với sơ đồ đánh địa chỉ cũ theo kiểu phân lớp A, B và C.

CIDR ra đời để giải quyết hai vấn đề bức xúc đối với mạng Internet là :

- Thiếu địa chỉ IP
- Vượt quá khả năng chứa đựng của các bảng chọn đường.

6.6.6.3 Vấn đề thiếu địa chỉ IP

Với sơ đồ đánh địa chỉ truyền thống, các địa chỉ được phân ra thành các lớp A, B và C. Mỗi địa chỉ có 2 phần, phần nhận dạng mạng và phần nhận dạng máy tính. Khi đó số lượng mạng và số máy tính tối đa cho từng mạng được thống kê như bảng sau :

Lớp mạng	Số lượng mạng	Số máy tính tối đa trong mạng
A	126	16.777.214
B	65.000	65.534
C	Hơn 2 triệu	254

Bởi vì các địa chỉ của mạng Internet thường được gán theo kích thước này dẫn đến tình trạng lãng phí. Trường hợp bạn cần 100 địa chỉ, Bạn sẽ được cấp một địa chỉ lớp C. Như vậy còn 154 địa chỉ không được sử dụng. Chính điều này dẫn đến trình trạng thiếu địa chỉ IP cho mạng Internet. Theo thống kê, chỉ có khoảng 3% số địa chỉ đã được cấp phát được sử dụng đến. Chính vì thế sơ đồ đánh địa chỉ mới CIDR ra đời để khắc phục tình trạng trên.

6.6.6.4 Vấn đề vượt quá khả năng chứa đựng của các bảng chọn đường

Khi số lượng mạng trên mạng Internet tăng cũng đồng nghĩa với việc tăng số lượng router trên mạng. Trong những năm gần đây, người ta dự đoán rằng các router đường trục của mạng Internet đang nhanh chóng tiến đến mức ngưỡng tối đa số lượng router mà nó có thể chấp nhận được. Thậm chí với những công nghệ hiện đại dùng để sản xuất các router thì về mặt lý thuyết kích thước tối đa của một bảng chọn đường cũng chỉ đến 60.000 mục từ (đường đi). Nếu không có những cải tiến thì các router đường trục sẽ đạt đến con số này và như thế không thể mở rộng mạng Internet hơn nữa.

Để giải quyết hai vấn đề trên, cộng đồng Internet đã đưa ra các giải pháp sau :

- Sửa đổi lại cấu trúc cấp phát địa chỉ IP để tăng hiệu quả
- Kết hợp việc chọn đường có cấu trúc để giảm tối đa số lượng các mục từ trong bảng chọn đường.

6.6.6.5 Sửa đổi lại cấu trúc cấp phát địa chỉ IP

CIDR được sử dụng để thay thế cho sơ đồ cấp phát cũ với việc qui định các lớp A, B, C. Thay vì phần nhận dạng mạng được giới hạn với 8, 16 hoặc 24 bits, CIDR sử dụng phần nhận dạng mạng có tính tổng quát từ 13 đến 27 bits. Chính vì thế các khối địa chỉ có thể được gán cho mạng nhỏ nhất với 32 máy tính đến mạng lớn nhất hơn 500.000 máy tính. Điều này đáp ứng gần đúng yêu cầu đánh địa chỉ của các tổ chức khác nhau.

6.6.6.6 Địa chỉ CIDR

Một địa chỉ theo cấu trúc CIDR, gọi tắt tắt địa chỉ CIDR, bao gồm 32 bits của địa chỉ IP chuẩn cùng với một thông tin bổ sung về số lượng các bit được sử dụng cho phần nhận dạng mạng.

Ví dụ : Với địa chỉ CIDR 206.13.01.48/25 thì chuỗi số "/25" chỉ ra rằng 25 bits đầu tiên trong địa chỉ IP được dùng để nhận dạng duy nhất một mạng, số bits còn lại dùng để nhận dạng một máy tính trong mạng.

Bảng sau so sánh giữa sơ đồ đánh địa chỉ theo kiểu CIDR và sơ đồ đánh địa chỉ theo chuẩn phân lớp hoàn toàn.

Số bits nhận dạng mạng trong địa chỉ CIDR	Lớp tương ứng trong chuẩn phân lớp hoàn toàn	Số lượng máy tính trong mạng
/27	1/8 lớp C	32
/26	¼ lớp C	64
/25	1/2 lớp C	128
/24	1 lớp C	256
/23	2 lớp C	512
/22	4 lớp C	1.024

/21	8 lớp C	2.048
/20	16 lớp C	4.096
/19	32 lớp C	8.192
/18	64 lớp C	16.384
/17	128 lớp C	32.768
/16	256 lớp C (= 1 lớp B)	65.536
/15	512 lớp C	131.072
/14	1,024 lớp C	262.144
/13	2,048 lớp C	524.288

Kết hợp việc chọn đường có cấu trúc để giảm tối đa số lượng các mục từ trong bảng chọn đường.

Sơ đồ đánh địa chỉ theo theo CIDR cũng cho phép kết hợp các đường đi, ở đó mục từ trong bảng chọn đường ở mức cao có thể đại diện cho nhiều router ở mức thấp hơn trong các bảng chọn đường tổng thể.

Sơ đồ này giống như hệ thống mạng điện thoại ở đó mạng được thiết lập theo kiến trúc phân cấp. Một router ở mức cao (quốc gia), chỉ quan tâm đến mã quốc gia trong số điện thoại, sau đó nó sẽ vạch đường cho cuộc gọi đến router đường trục phụ trách mạng quốc gia tương ứng với mã quốc gia đó. Router nhận được cuộc gọi nhìn vào phần đầu của số điện thoại, mã tỉnh, để vạch đường cho cuộc gọi đến một mạng con tương ứng với mã tỉnh đó, và cứ như thế. Trong sơ đồ này, các router đường trục chỉ lưu giữ thông tin về mã quốc gia cho mỗi mục từ trong bảng chọn đường của mình, mỗi mục từ như thế đại diện cho một số khổng lồ các số điện thoại riêng lẽ chứ không phải là một số điện thoại cụ thể.

Thông thường, các khối địa chỉ lớn được cấp cho các nhà cung cấp dịch vụ Internet (IP- Internet Service Providers) lớn, sau đó họ lại cấp lại các phần trong khối địa chỉ của họ cho các khách hàng của mình.

Hiện tại, mạng Internet sử dụng cả hai sơ đồ cấp phát địa chỉ Classfull standard và CIDR. Hầu hết các router mới đều hỗ trợ CIDR và những nhà quản lý Internet thì khuyến khích người dùng cài đặt sơ đồ đánh địa chỉ CIDR.

Tham khảo thêm về CIDR ở địa chỉ http://www.rfc-editor.org/rfcsearch.html với các RFC liên quan sau:

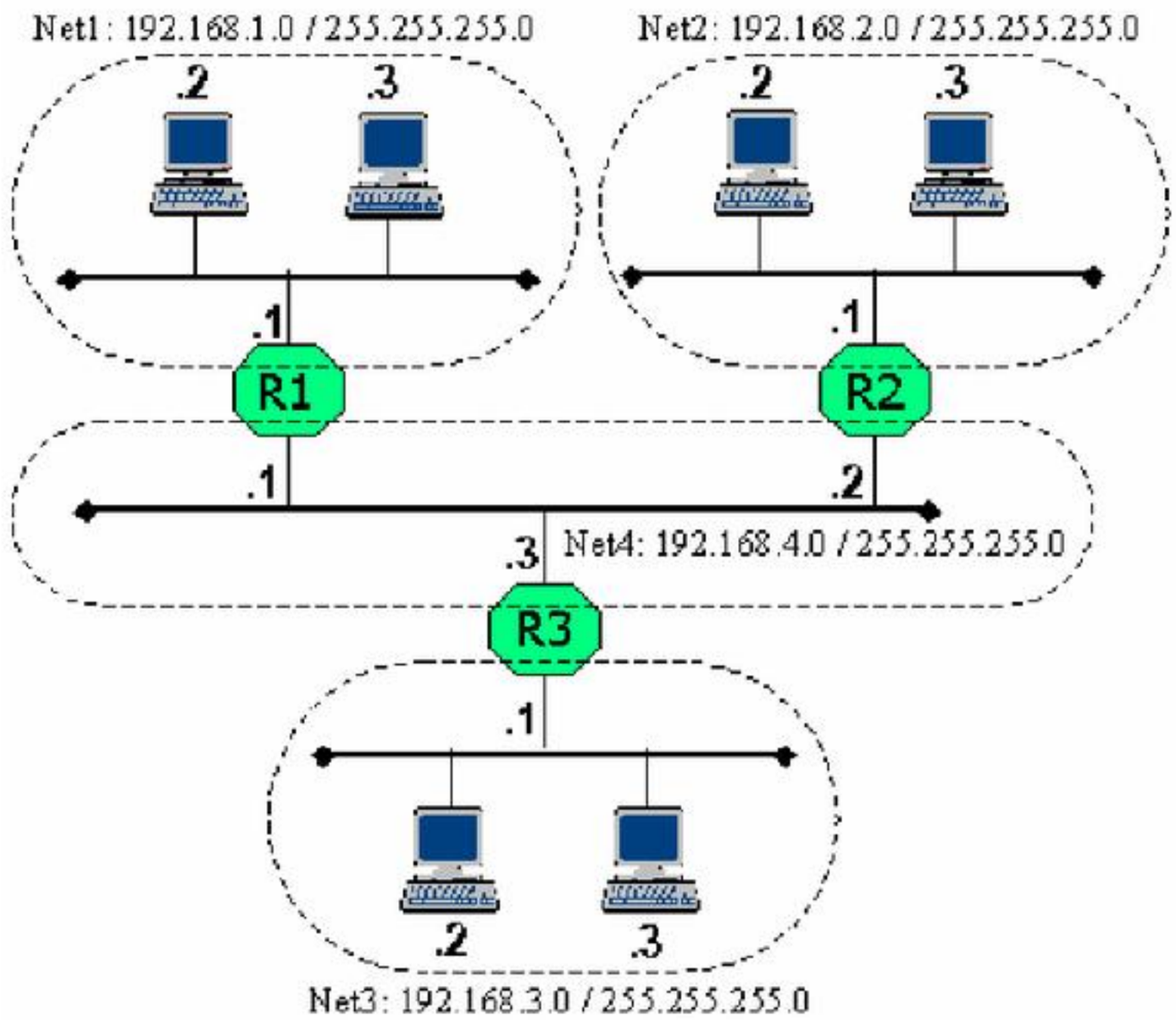
- RFC 1517: Applicability Statement for the Implementation of CIDR
- RFC 1518: An Architecture for IP Address Allocation with CIDR
- RFC 1519: CIDR: An Address Assignment and Aggregation Strategy
- RFC 1520: Exchanging Routing Information Across Provider Boundaries in the CIDR Environment

6.6.7 Vạch đường trong giao thức IP

Cho ba mạng Net1, Net2 và Net3 nối lại với nhau nhờ 3 router R1, R2 và R3. Mạng Net4 nối các router lại với nhau. Công việc đầu tiên trong thiết kế mạng liên mạng IP là chọn địa chỉ mạng cho các nhánh mạng. Trong trường hợp này ta chọn mạng lớp C cho 4 mạng như bảng sau:

Mạng	Địa chỉ mạng	Mặt nạ mạng
Net1	192.168.1.0	255.255.255.0
Net2	192.168.2.0	255.255.255.0
Net3	192.168.3.0	255.255.255.0
Net4	192.168.4.0	255.255.255.0

Kế tiếp, gán địa chỉ cho từng máy tính trong mạng. Ví dụ trong mạng Net1, các máy tính được gán địa chỉ là 192.168.1.2 (Ký hiệu **.2** là cách viết tắt của địa chỉ IP để mô tả Phần nhận dạng máy tính) và 192.168.1.3. Mỗi router có hai giao diện tham gia vào hai mạng khác nhau. Ví dụ, giao diện tham gia vào mạng NET1 của router R1 có địa chỉ là 192.168.1.1 và giao diện tham gia vào mạng NET4 có địa chỉ là 192.168.4.1.



H6.36 Ví dụ về liên mạng một liên mạng sử dụng giao thức IP

Để máy tính của các mạng có thể giao tiếp được với nhau, cần phải có thông tin về đường đi. Bảng chọn đường của router có thể tạo ra thủ công hoặc tự động. Đối với mạng nhỏ, nhà quản trị mạng sẽ nạp đường đi cho các router thông qua các lệnh được cung cấp bởi hệ điều hành của router. Bảng chọn đường trong giao thức IP có 4 thông tin quan trọng là :

- Địa chỉ mạng đích
- Mặt nạ mạng đích
- Router kế tiếp sẽ nhận gói tin (Next Hop)
- Giao diện chuyển gói tin đi

Trong ví dụ trên, các router sẽ có bảng chọn đường như sau :

R1-Routing table

Network/Netmask	NextHop	Interface
192.168.1.0/255.255.255.0	local	local
192.168.2.0/255.255.255.0	192.168.4.2	192.168.4.1
192.168.3.0/255.255.255.0	192.168.4.3	192.168.4.1
192.168.4.0/255.255.255.0	local	local

R2-Routing table

Network/Netmask	NextHop	Interface
192.168.1.0/255.255.255.0	192.168.4.1	192.168.4.2
192.168.2.0/255.255.255.0	local	local
192.168.3.0/255.255.255.0	192.168.4.3	192.168.4.2
192.168.4.0/255.255.255.0	local	local

R3-Routing table

Network/Netmask	NextHop	Interface
192.168.1.0/255.255.255.0	192.168.4.1	192.168.4.3
192.168.2.0/255.255.255.0	192.168.4.2	192.168.4.3
192.168.3.0/255.255.255.0	local	local
192.168.4.0/255.255.255.0	local	local

Các máy tính cũng có bảng chọn đường. Dưới đây là bảng chọn đường của máy tính có địa chỉ 192.168.3.3 :

192.168.3.3 - Routing table

Network/Netmask	NextHop	Interface
192.168.3.0/255.255.255.0	local	local
default	192.168.3.1	local

Mạng đích default ý nói rằng ngoài những đường đi đến các mạng đã liệt kê phía trên, các đường đi còn lại thì gởi cho NextHop của mạng default này. Như vậy, để gởi gói tin cho bất kỳ một máy tính nào nằm bên ngoài mạng 192.168.3.0 thì máy tính 192.168.3.3 sẽ chuyển gói tin cho router 3 ở địa chỉ 192.168.3.1.

6.6.7.1 Đường đi của gói tin

Để hiểu rõ có chế hoạt động của giao thức IP, ta hãy xét hai trường hợp gởi gói tin : Trường hợp máy tính gởi và nhận nằm trong cùng một mạng và trường hợp máy tính gởi và máy tính nhận nằm trên hai mạng khác nhau.

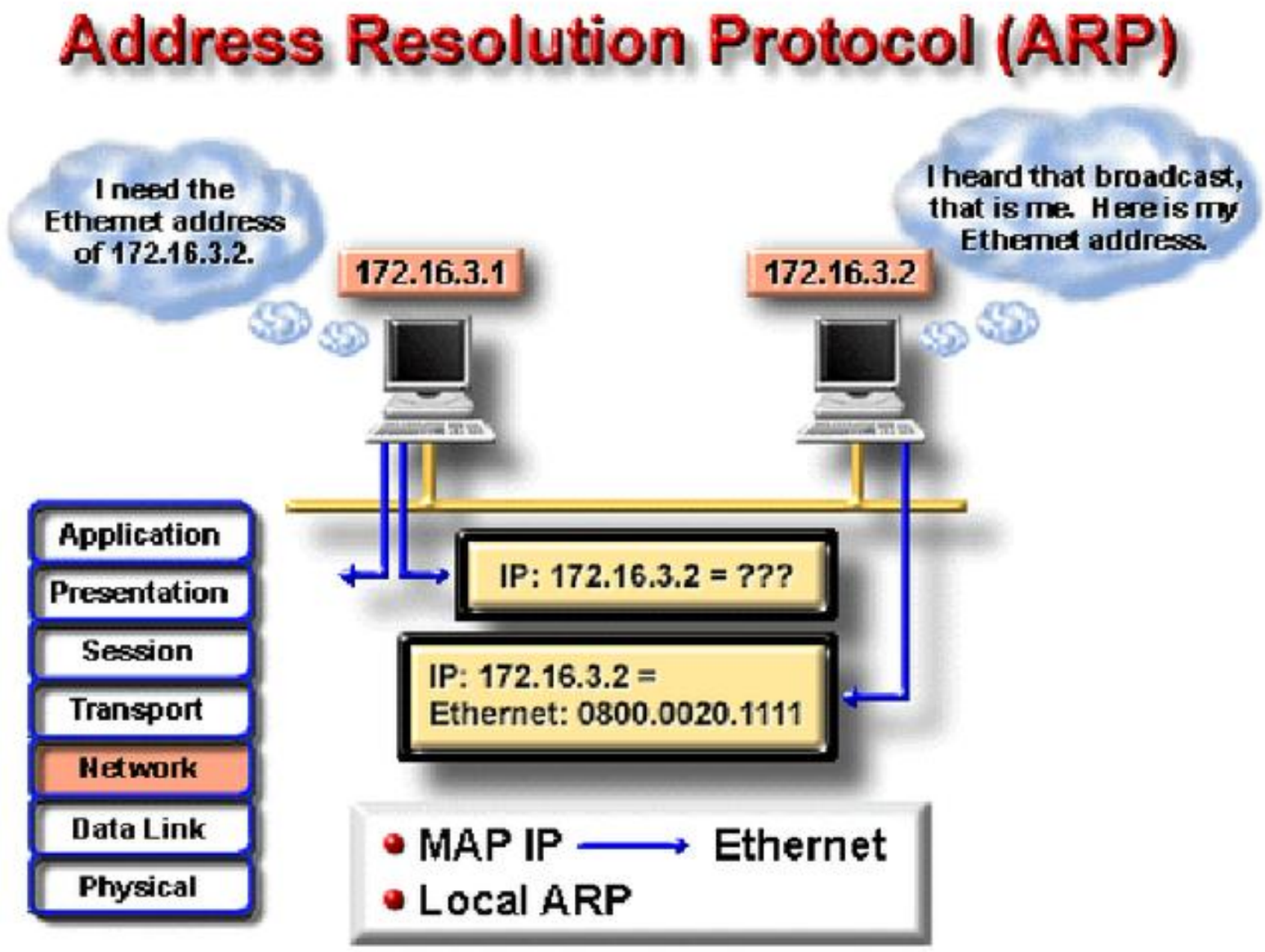
Giả sử máy tính có địa chỉ 192.168.3.3 gởi một gói tin cho máy tính 192.168.3.2. Tầng hai của máy gởi sẽ đặt gói tin vào một khung với địa chỉ nhận là địa chỉ vật lý của máy nhận và gởi khung lên đường truyền NET3, trên đó máy tính nhận sẽ nhận được gói tin.

Bây giờ ta xét trường hợp máy tính có địa chỉ 192.168.3.3 trên mạng NET3 gởi gói tin cho máy tính có địa chỉ 192.168.1.2 trên mạng Net1. Theo như bảng chọn đường của máy gởi, các gói tin có địa chỉ nằm ngoài mạng 192.168.3.0 sẽ được chuyển đến router R3 (địa chỉ 192.168.3.1). Chính vì thế, máy tính gởi sẽ đặt gói tin vào một khung với địa chỉ nhận là địa chỉ vật lý của giao diện 192.168.3.1 và đưa lên đường truyền NET3. Nhận được gói tin, R3 phân tích địa chỉ IP của máy nhận để xác định đích đến của gói tin . Bảng chọn đường cho thấy, với đích đến là mạng 192.168.1.0 thì cần phải chuyển gói tin cho router R1 ở địa chỉ 192.168.4.1 thông qua giao diện 192.168.4.3. Vì thế R3 đặt gói tin vào một khung với địa chỉ nhận là địa chỉ vật lý của giao diện 192.168.4.1 của router R1 và đưa lên đường truyền NET4. Tương tự, R1 sẽ chuyển gói tin cho máy nhận 192.168.1.2 bằng một khung trên đường truyền NET1.

Ta nhận thấy rằng, để đi đến được máy nhận, gói tin được chuyển đi bởi nhiều khung khác nhau. Mỗi khung sẽ có địa chỉ nhận khác nhau, tuy nhiên địa chỉ của gói tin thì luôn luôn không đổi.

6.6.7.2 Giao thức phân giải địa chỉ (Address Resolution Protocol)

Nếu một máy tính muốn truyền một gói tin IP nó cần đặt gói tin này vào trong một khung trên đường truyền vật lý mà nó đang nối kết vào. Để có thể truyền thành công khung, máy tính gởi cần thiết phải biết được địa chỉ vật lý (MAC) của máy tính nhận. Điều này có thể thực hiện được bằng cách sử dụng một bảng để ánh xạ các địa chỉ IP về địa chỉ vật lý. Giao thức IP sử dụng giao thức ARP (Address Resolution Protocol) để thực hiện ánh xạ từ một địa chỉ IP về một địa chỉ MAC.



H6.37 Giao thức ARP

Một máy tính xác định địa chỉ vật lý của nó vào lúc khởi động bằng cách đọc thiết bị phần cứng và xác định địa chỉ IP của nó bằng cách đọc tập tin cấu hình, sau đó lưu thông tin về mối tương ứng giữa địa chỉ IP và MAC của nó vào trong vùng nhớ tạm (ARP cache). Khi nhận được một địa chỉ IP mà ARP không thể tìm ra được địa chỉ vật lý tương ứng dựa vào vùng nhớ tạm hiện tại, nó sẽ thực hiện một khung quảng bá có định dạng như sau :

Tổng quát	Các trường	Kích thức (byte)	Các giá trị
Ethernet Header	Ethernet Destination Address	6	Địa chỉ máy nhận, trong trường hợp này là một địa chỉ quảng bá
	Ethernet Source Address	6	Địa chỉ của máy gởi thông điệp
	Frame Type	2	Kiểu khung, có giá trị là 0x0806 khi ARP yêu cầu và 0x8035 khi ARP trả lời
ARP request/reply	Hardware Type	2	Giá trị là 1 cho mạng Ethernet
	Protocol Type	2	Có giá trị là 0x0800 cho địa chỉ IP

	Hardware Address Size in bytes	1	Chiều dài của địa chỉ vật lý, có giá trị là 6 cho mạng Ethernet
	Protocol Address Size in bytes	1	Chiều dài địa chỉ của giao thức, có giá trị là 4 cho giao thức IP
	Operation	2	Là 1 nếu là khung yêu cầu, là 2 nếu là khung trả lời
	Sender Ethernet Address	6	-
	Sender IP Address	4	-
	Destination Ethernet Address	6	Không sử dụng đến trong yêu cầu của ARP
	Destination IP Address	4	-

Nhờ vào việc gởi các yêu cầu này, một máy tính có thể bổ sung thông tin cho vùng cache của giao thức ARP, nhờ đó cập nhật kịp thời mọi sự thay đổi của sơ đồ mạng. Thông thường thời gian quá hạn (Time-out) cho một thông tin trong vùng cache là 20 phút. Một yêu cầu ARP cho một máy tính không tồn tại trên nhánh mạng được lặp lại một vài lần xác định nào đó.
Nếu một máy tính được nối kết vào nhiều hơn một mạng bằng giao diện mạng, khi đó sẽ tồn tại những vùng cache ARP riêng cho từng giao diện mạng.
Lưu ý, ARP trên một máy tính chỉ thực hiện việc xác địa chỉ vật lý cho các địa chỉ cùng địa chỉ mạng / mạng con với nó mà thôi. Đối với các gói tin gởi cho các máy tính có địa chỉ IP không cùng một mạng / mạng con với máy gởi sẽ được chuyển hướng cho một router nằm cùng mạng với máy gởi để chuyển đi tiếp.
Vì các yêu cầu ARP được quảng bá rộng rãi, cho nên bất kỳ một máy tính nào đang duy trì một vùng cache đều có thể theo dõi tất cả các yêu cầu được quảng bá này để lấy thông tin về địa chỉ vật lý và địa chỉ IP của máy gởi yêu cầu và bổ sung vào vùng cache của nó khi cần thiết. Khi một máy tính khởi động, nó gởi một yêu cầu ARP (có thể cho chính nó) như để thông báo với các máy tính khác về sự xuất hiện của nó trong mạng cục bộ.
Có thể gán nhiều hơn một địa chỉ IP cho một địa chỉ vật lý. Chú ý rằng, định dạng của yêu cầu ARP thì được thiết kế để có thể hỗ trợ được cho các giao thức khác ngoài IP và Ethernet.

6.6.7.3 Giao thức phân giải địa chỉ ngược RARP (Reverse Address Resolution Protocol)

Ngày nay, các trạm làm việc không đĩa cứng (Diskless workstation) được sử dụng rộng rãi. Mỗi máy tính chỉ cần bộ xử lý và bộ nhớ, tất cả không gian lưu trữ được cung cấp từ một máy chủ (Server) sử dụng một hệ thống tập tin mạng theo một chuẩn nào đó. Do không có các tập tin cấu hình, tiến trình khởi động của các máy tính này thường sử dụng một giao thức truyền tải tập tin rất đơn giản như TFTP. Tuy nhiên, trước khi có thể nối kết đến được server, các trạm làm việc cần phải biết được địa chỉ IP của nó. Giao thức RARP được dùng trong trường hợp này. RARP sử dụng cùng định dạng yêu cầu của ARP nhưng trường Operation có giá trị là 3 cho yêu cầu và 4 cho trả lời. Trên server duy trì một bảng mô tả mối tương quan giữa địa chỉ vật lý và địa chỉ IP của các máy trạm. Khi nhận được yêu cầu RARP, server tìm trong bảng địa chỉ và trả về địa chỉ IP tương ứng cho máy trạm đã gởi yêu cầu.

6.6.7.4 Giao thức thông điệp điều khiển Internet ICMP (Internet Control Message Protocol)

Giao thức ICMP được cài đặt trong hầu hết tất cả các máy tính TCP/IP. Các thông điệp của giao thức được gởi đi trong các gói tin IP và được dùng để gởi đi các báo lỗi hay các thông tin điều khiển.

ICMP tạo ra nhiều loại thông điệp hữu ích như :

- Đích đến không tới được (Destination Unreachable),
- Thăm hỏi và trả lời (Echo Request and Reply),
- Chuyển hướng (Redirect),
- Vượt quá thời gian (Time Exceeded),
- Quảng bá bộ chọn đường (Router Advertisement)
- Cô lập bộ chọn đường (Router Solicitation)
- ....

Nếu một thông điệp không thể phân phát được thì nó sẽ không được gởi lại. Điều này để tránh tình trạng di chuyển không bao giờ dừng của các thông điệp ICMP.

Nếu một thông điệp « Đích đến không tới được » được gởi đi bởi một router, điều đó có nghĩa rằng router không thể gởi gói tin đến đích được. Khi đó router sẽ xóa gói tin ra khỏi hàng đợi của nó. Có hai nguyên nhân làm cho một gói tin không thể đi đến nơi được. Phần lớn là máy gởi mô tả một địa chỉ nhận mà nó không tồn tại trên thực tế. Trường hợp ít hơn là router không biết đường đi đến nơi nhận gói tin.

Thông điệp Đích đến không tới được được chia thành bốn loại cơ bản là :

- Mạng không đến được (Network unreachable) : Có nghĩa là có sự cố trong vấn đề vạch đường hoặc địa chỉ nhận của gói tin.
- Máy tính không đến được (Host unreachable) : Thông thường dùng để chỉ trục trặc trong vấn đề phân phát, như là sai mặt nạ mạng con chẳng hạn.
- Giao thức không đến được (Protocol unreachable) : Máy nhận không hỗ trợ giao thức ở tầng cao hơn như gói tin đã mô tả.
- Cổng không đến được (Port unreachable) : Socket của giao thức TCP hay cổng không tồn tại.

Một thông điệp « Thăm hỏi và trả lời » được tạo ra bởi lệnh ping từ một máy tính để kiểm tra tính liên thông trên liên mạng. Nếu có một thông điệp trả lời, điều đó biểu hiện rằng giữa máy gởi và máy nhận có thể giao tiếp được với nhau.

Một thông điệp « Chuyển hướng » được gởi bởi một router đến máy đã gởi gói tin để khuyến cáo về một đường đi tốt hơn. Router hiện tại vẫn chuyển tiếp gói tin mà nó nhận được. Thông điệp chuyển hướng giữ cho bảng chọn đường của các máy tính được nhỏ bởi vì chúng chỉ cần chứa địa chỉ của một router mà thôi, thậm chí router đó cung cấp đường đi không phải là tốt nhất. Đôi khi sau khi nhận được thông điệp chuyển hướng, thiết bị gởi vẫn sử dụng đường đi cũ.

Một thông điệp « Vượt quá thời hạn » được gởi bởi một router nếu thời gian sống (Time–to-live) của gói tin, tính bằng số router hay giây, có giá trị là 0. Thời gian sống của gói tin giúp phòng ngừa trường hợp gói tin được gởi đi lòng vòng trên mạng và không bao giờ đến nơi nhận. Router sẽ bỏ đi các gói tin đã hết thời gian sống.

Chương 7: TẦNG VẬN CHUYỂN

Mục đích

Chương này nhằm giới thiệu với người đọc những nội dung sau:

- Vai trò của tầng vận chuyển và các chức năng mà tầng vận chuyển cung cấp cho tầng ứng dụng
- Ý nghĩa và cơ chế thiết lập nối kết và giải phóng nối kết cho các nối kết điểm – điểm
- Chi tiết về hay giao thức TCP và UDP thuộc tầng vận chuyển

Yêu cầu

Sau khi học xong chương này, người học phải có được những khả năng sau:

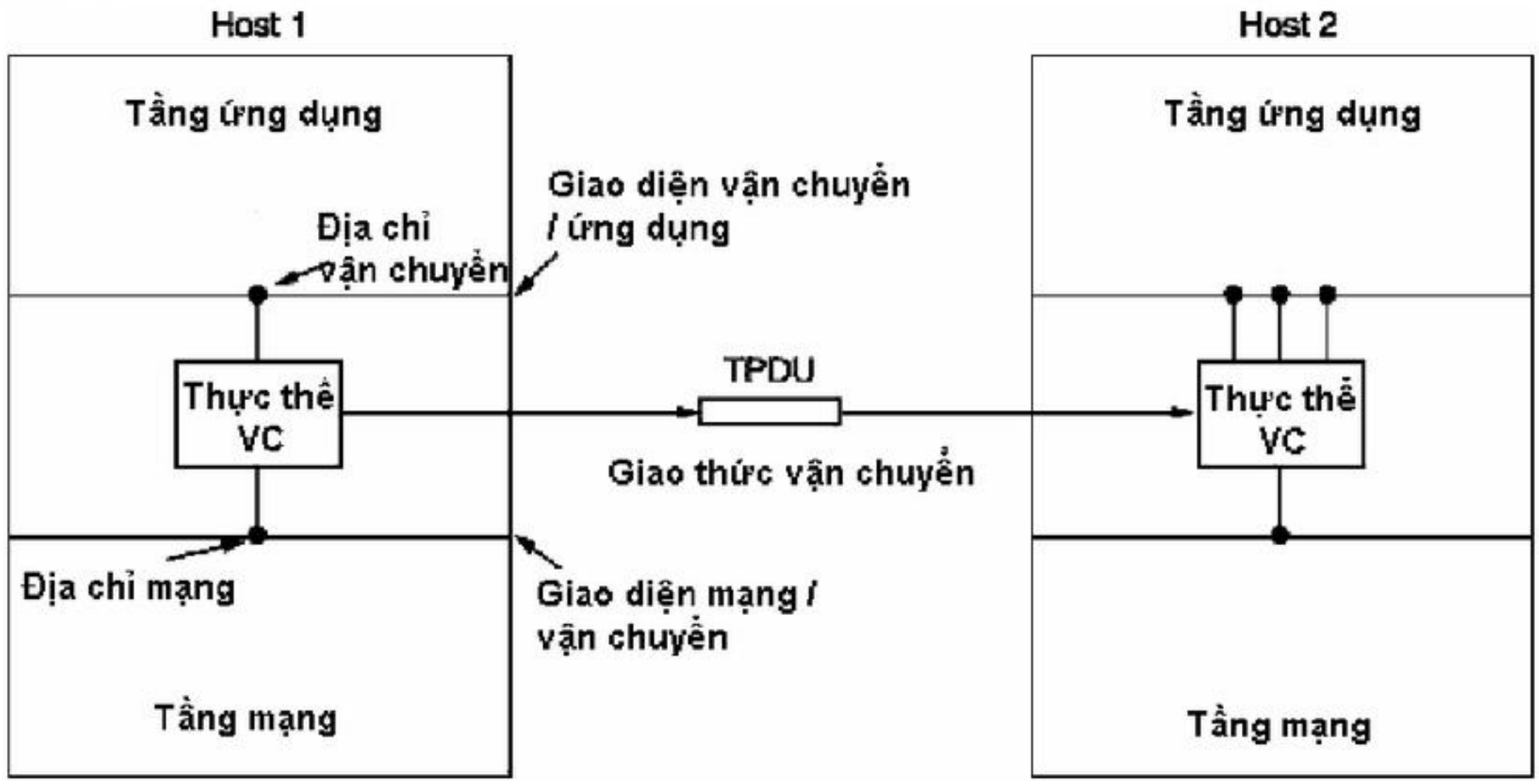
- Biện luận được sự cần thiết của tầng vận chuyển trong một liên mạng
- Giải thích được cơ chế thiết lập và xóa nối kết các cuộc giao tiếp điểm-điểm của tầng vận chuyển
- Trình bày được nguyên tắc hoạt động của hai giao thức TCP và UDP của mạng Internet

7.1 Dịch vụ của tầng vận chuyển

Trong khi tầng mạng đảm bảo việc chuyển gói tin từ một host đến một host khác, tầng vận chuyển lại làm trung gian giữa tầng mạng và các ứng dụng mạng – nó chuyển thông tin giữa các tiến trình chạy trên các host khác nhau. Phần sau sẽ thảo luận về các dịch vụ và kiểu dịch vụ mà tầng vận chuyển cung cấp cho tầng ứng dụng.

7.1.1 Các dịch vụ cung cấp cho tầng ứng dụng

Mục tiêu quan trọng của tầng vận chuyển là cung cấp dịch vụ vận chuyển gói tin hiệu quả, tin cậy và tiết kiệm chi phí cho người dùng của nó, ở đây là các tiến trình chạy ở tầng ứng dụng.
Phần cứng/mềm nằm trong lớp vận chuyển và hoạt động ở đó được gọi là thực thể vận chuyển. Thực thể vận chuyển có thể nằm ở nhân của hệ điều hành, trong một tiến trình người dùng riêng biệt, trong một gói thư viện liên quan đến các ứng dụng mạng, hoặc thậm chí được gói gọn trong card mạng.
Mối quan hệ logic giữa tầng mạng, tầng vận chuyển và tầng ứng dụng được thể hiện trong hình sau H7.1.



H7.1 Các tầng mạng, vận chuyển và ứng dụng

Có hai kiểu dịch vụ vận chuyển: có nối kết và không nối kết. Và tầng vận chuyển cũng phải cung cấp các tham số để người dùng chỉ định loại dịch vụ họ mong muốn.
Loại dịch vụ vận chuyển có nối kết hoạt động giống như dịch vụ có nối kết của tầng mạng. Nghĩa là nó có 3 kỳ: thiết lập nối kết, truyền dữ liệu và hủy nối kết. Loại dịch vụ không nối kết cũng giống như ở tầng mạng, chỉ đơn giản đẩy gói tin ra mạng và hy vọng nó đến đích.
Từ đây phát sinh câu hỏi: Hai tầng vận chuyển và mạng hoạt động giống nhau, sao không nhập lại làm một? Câu trả lời rất dễ gây tranh cãi: Mã lệnh vận chuyển nằm hoàn toàn trong máy tính của người dùng, nhưng lớp mạng hầu hết chạy trên các router. Nếu nhập cả hai vào một lớp mạng, giả sử lớp mạng cung cấp dịch vụ không thỏa đáng thì sao? Nếu lớp mạng thường xuyên làm mất gói tin thì sao? Nếu các router bị chết thường xuyên thì sao?
Vấn đề phát sinh ở chỗ, người dùng không có quyền điều khiển thực sự lên lớp mạng, do đó họ không thể giải quyết vấn đề dịch vụ không tốt bằng cách chọn các đường đi khác, hay áp đặt thêm nhiều giải pháp điều khiển lỗi lên lớp liên kết dữ liệu. Khả năng duy nhất có thể được là đặt trên lớp mạng một lớp khác làm nhiệm vụ cải thiện chất lượng dịch vụ. Nếu, trong một mạng con dạng hướng nối kết, một thực thể vận chuyển được thông báo giữa lúc truyền dữ liệu rằng kết nối mạng đã bị gãy, nó có thể thiết lập một kết nối mạng khác đến bên đối thoại bên kia, rồi gởi đi câu hỏi

rằng dữ liệu nào đã đến, cái nào chưa và cuối cùng khởi động lại từ điểm bị bỏ dở dang. Dữ liệu bị mất hoặc bị hư hỏng sẽ được phục hồi bởi lớp vận chuyển, do đó việc chuyển dữ liệu an toàn hơn. Như thường lệ, tại lớp vận chuyển, người ta thiết kế các hàm dịch vụ cơ sở để triệu gọi các dịch vụ vận chuyển và các hàm này là đơn giản, duy nhất và độc lập với các hàm cơ sở ở tầng mạng. Nhờ vào sự độc lập này, sự phức tạp ở mức mạng bị che đi, các nhà lập trình ứng dụng có thể viết mã lệnh dựa vào một tập hợp chuẩn các hàm cơ sở mức vận chuyển và cho chương trình của họ chạy trên nhiều loại mạng mà không bị đau đầu bởi các vấn đề về giao diện các mạng con khác nhau và việc truyền tải không tin cậy.

7.1.2 Các hàm dịch vụ cơ sở

Các hàm dịch vụ cơ sở ở lớp vận chuyển được chia thành hai nhóm theo phương thức hoạt động: có nối kết và không nối kết.

7.1.3 Các hàm dịch vụ hướng nối kết

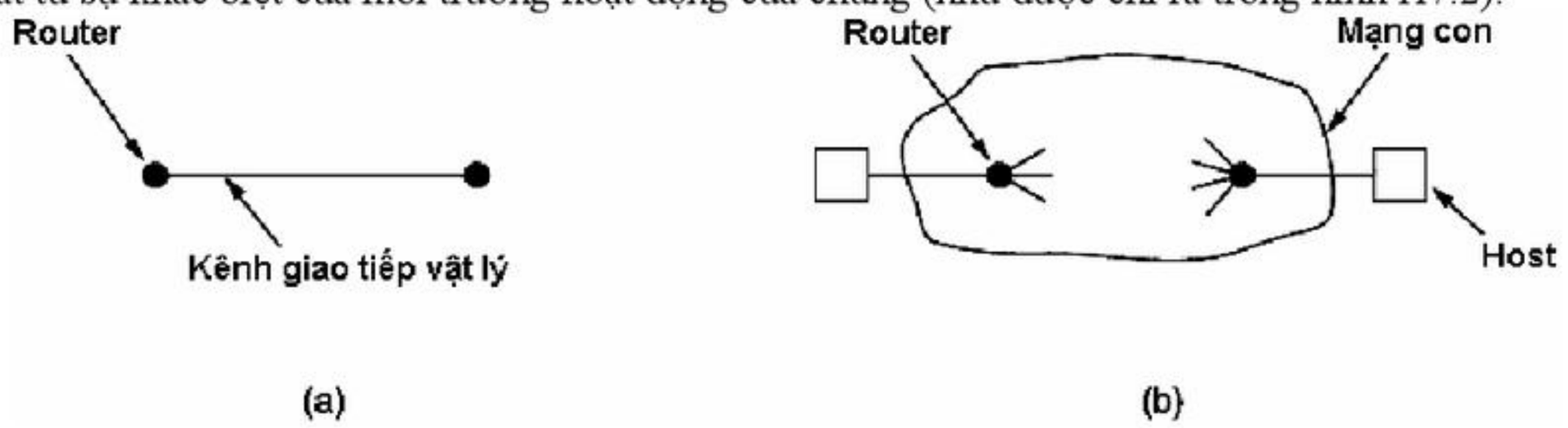
Hàm	Gói tin gởi đi	Ý nghĩa
LISTEN	Không có	Nghẽn cho đến khi tiến trình nào đó nối kết tới
CONNECT	Yêu cầu kết nối (Connection Request)	Chủ động yêu cầu thiết lập nối kết đến tiến trình khác
SEND	Dữ liệu (Data)	Gởi thông tin đi
RECEIVE	Không có	Nghẽn cho đến khi một gói tin đến và nhận nó
DISCONNECT	Yêu cầu hủy kết nối (Disconnection Request)	Muốn hủy kết nối với bên đối tác

7.1.4 Các hàm dịch vụ dạng không nối kết

Hàm	Gói tin gởi đi	Ý nghĩa
SEND	Dữ liệu (Data)	Gởi thông tin đi
RECEIVE	Không có	Nghẽn cho đến khi một gói tin đến và nhận nó

7.2 Các yếu tố cấu thành giao thức vận chuyển

Cũng giống như giao thức ở tầng liên kết dữ liệu, giao thức vận chuyển phải đối phó với các vấn đề về điều khiển lỗi, đánh số thứ tự gói tin và điều khiển luồng dữ liệu.
Tuy nhiên, giao thức trên hai tầng có nhiều điểm khác biệt quan trọng. Những khác biệt này xuất phát từ sự khác biệt của môi trường hoạt động của chúng (như được chỉ ra trong hình H7.2).



H7.2. (a) Môi trường của lớp liên kết dữ liệu.
(b) Môi trường của lớp vận chuyển

Tại lớp liên kết dữ liệu, hai router giao tiếp với nhau qua một kênh truyền vật lý, trong khi tại lớp vận chuyển, kênh truyền này được thay bằng cả một mạng con. Sự khác nhau này sẽ dẫn đến nhiều hệ lụy mà những người thiết kế giao thức vận chuyển phải đau đầu giải quyết: định địa chỉ các tiến trình trên các host khác nhau như thế nào, xử lý như thế nào đối với những trường hợp mất gói tin trong quá trình trao đổi hoặc gói tin đi chậm dẫn đến mãn kỳ và gởi thêm một gói tin bị trùng lắp, đồng bộ hóa hai tiến trình đang trao đổi dữ liệu như thế nào khi mà chúng đang ở rất xa nhau.

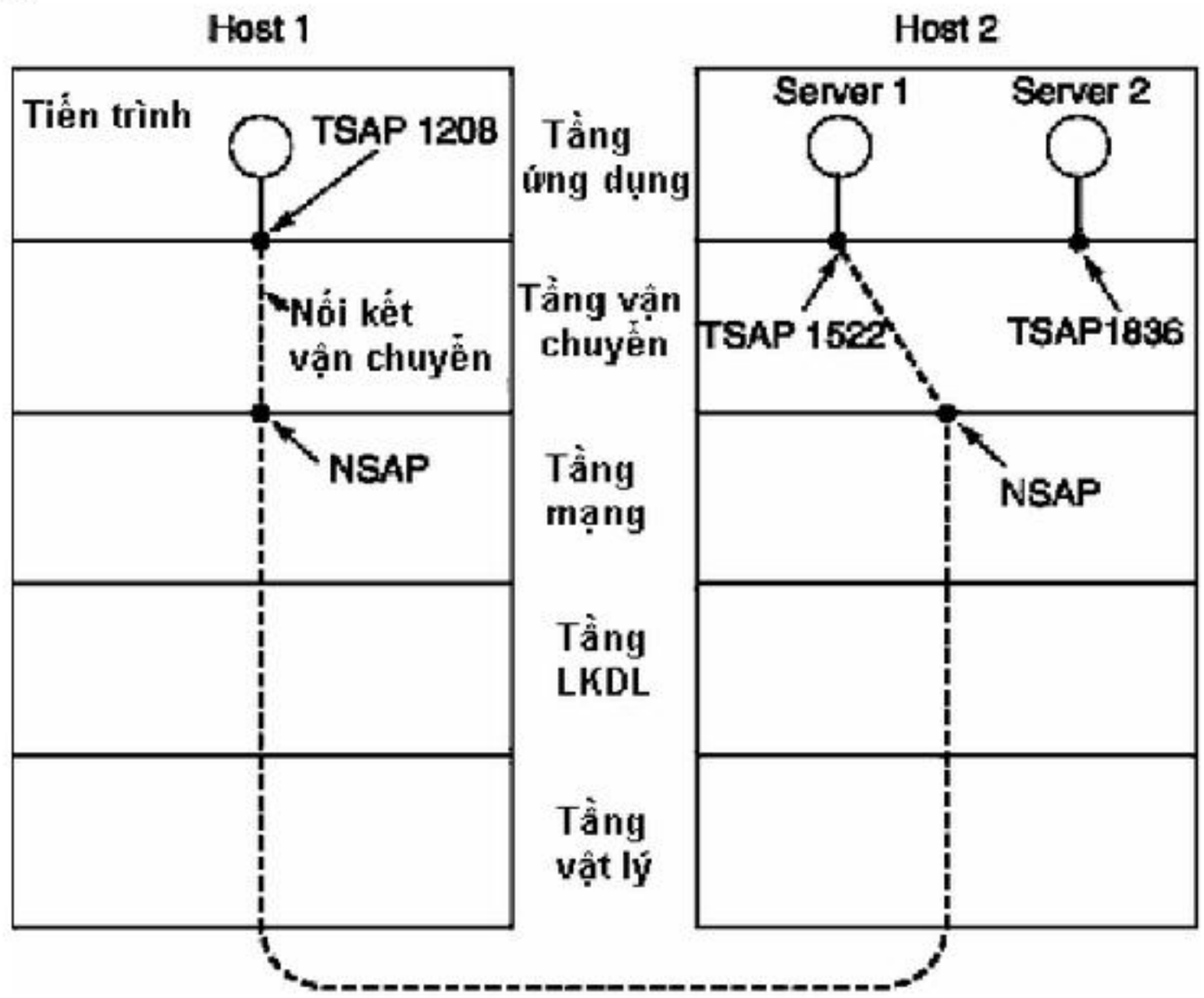
7.2.1 Định địa chỉ

Khi một tiến trình mong muốn thiết lập nối kết với một tiến trình khác từ xa, nó phải chỉ ra rằng nó muốn kết nối với tiến trình nào. (Vận chuyển hướng không nối kết cũng gặp vấn đề tương tự: thông điệp sẽ gởi đến ai?). Một phương pháp định địa chỉ ở tầng vận chuyển của Internet là dùng số hiệu cổng (port), còn ở trong mạng ATM là AAL-SAP. Chúng ta sẽ dùng từ chung nhất để định địa chỉ tiến trình là TSAP (Transport Service Access Point). Tương tự, địa chỉ trong tầng mạng được gọi là NSAP.

Hình H7.3 mô phỏng mối quan hệ giữa NSAP, TSAP và kết nối vận chuyển. Các tiến trình ứng dụng, cả client và server đều phải gắn vào một TSAP và thiết lập nối kết đến TSAP khác. Và kết nối này chạy qua cả hai TSAP. Mục tiêu của việc sử dụng các TSAP là vì trong một số mạng, mỗi máy tính chỉ có một NSAP, do đó cần phải có cách phân biệt nhiều điểm cuối mức vận chuyển khi chúng đang chia sẻ một NSAP.

Ví dụ, dàn cảnh một cuộc kết nối mức vận chuyển có thể diễn ra như sau:

1. Một server phục vụ thông tin về thời gian trên host 2 gắn nó vào TSAP 1522 để chờ một cuộc gọi đến.
2. Một tiến trình ứng dụng chạy trên host 1 muốn biết giờ hiện tại, vì thế nó đưa ra một yêu cầu nối kết chỉ ra TSAP 1208 là cổng nguồn và TSAP 1522 là cổng đích. Hành động này dẫn đến một kết nối vận chuyển được thiết lập giữa hai tiến trình client và server trên hai host 1 và 2.



H 6.3. TSAP, NSAP và kết nối vận chuyển

3. Tiến trình client gởi một yêu cầu đến server để hỏi về thời gian.
4. Server trả lời thời gian hiện tại cho client.
5. Kết nối vận chuyển cuối cùng được giải phóng.

7.2.2 Thiết lập nối kết

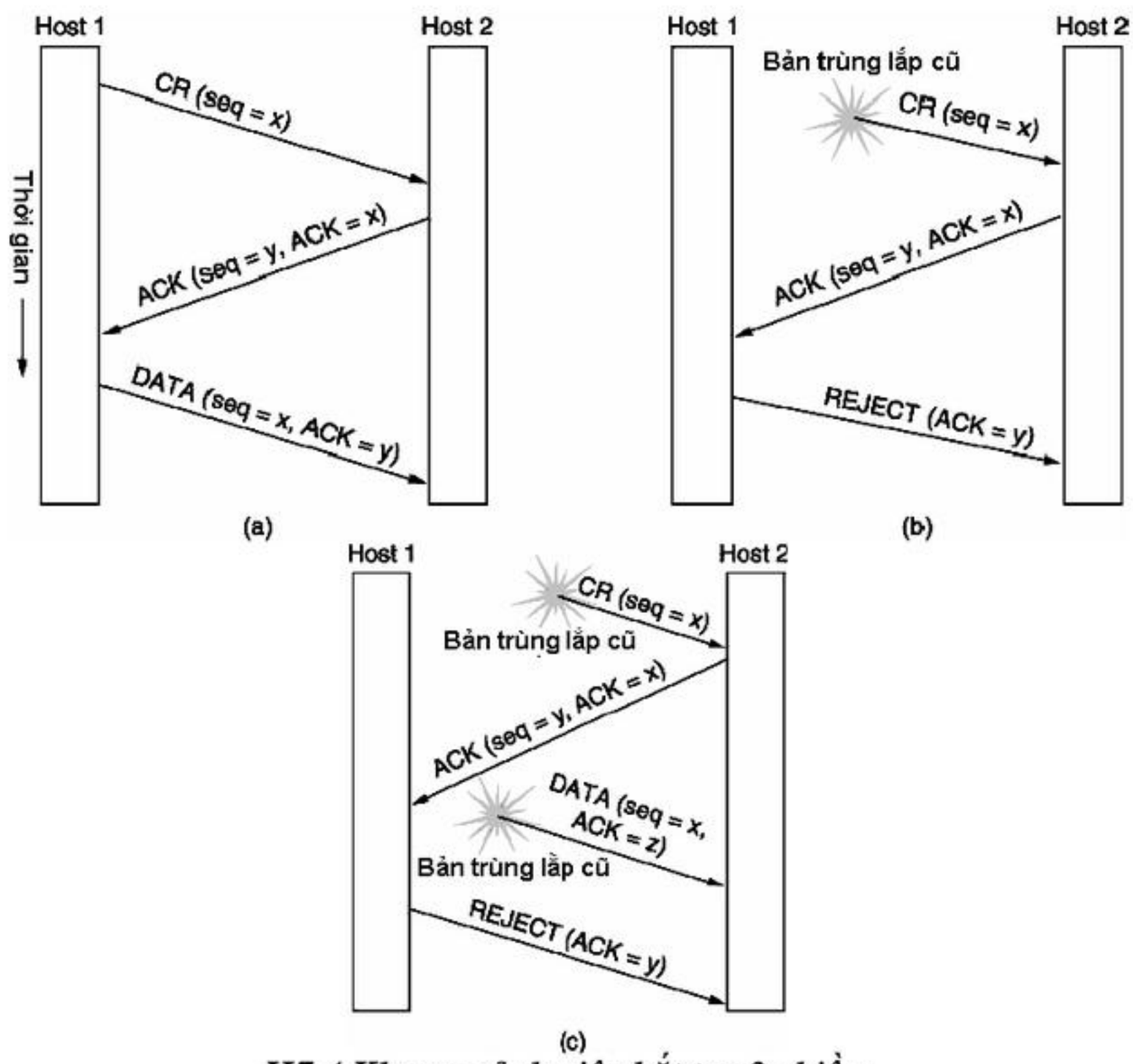
Việc thiết lập nối kết nghe có vẻ dễ dàng, nhưng khi thực hiện có thể sẽ gặp nhiều rắc rối. Thoạt nhìn, một phiên thiết lập nối kết sẽ diễn ra như sau: một bên sẽ gởi TPDU yêu cầu nối kết (Connection Request – CR) đến bên kia, bên kia sẽ gởi một TPDU trả lời chấp nhận nối kết (Connection Accepted – CA).

Vấn đề phát sinh khi mạng làm mất, tồn trữ quá lâu hay làm trùng lắp các gói tin do hai thực thể vận chuyển trao đổi qua lại với nhau. Ví dụ một tình huống như sau: tiến trình 1 gởi yêu cầu kết nối đến tiến trình 2, yêu cầu này bị các mạng con trung gian trì hoãn do tắc nghẽn. Mãn kỳ, tiến trình 1 gởi lại yêu cầu nối kết, vừa lúc đó yêu cầu nối kết bị trì hoãn cũng đến tiến trình 2.

Giải thuật thiết lập nối kết phổ biến nhất là giải thuật bắt tay 3 chiều (three-way hand-shake). Xin xem các tình huống được mô phỏng trong Hình H7.4. Giả sử yêu cầu nối kết phát sinh ở host 1. Host 1 chọn một số thứ tự là ***x*** và đính kèm số đó trong TPDU CR (***CR (seq=x)***) gởi đến host 2. Host 2 báo nhận ACK (***ACK (seq = y, ACK = x)***) và thông báo số thứ tự khởi đầu của nó là ***y***. Cuối cùng host 1 báo nhận cho host 2 nó đã biết số thứ tự khởi đầu của host 2 là ***y*** bằng TPDU dữ liệu đầu tiên gởi đến host 2 (***DATA (seq=x, ACK=y)***).

Bây giờ xét đến tình huống TPDU CR bị trùng lắp. Khi TPDU CR thứ hai đến host 2, host 2 liền trả lời ACK vì tưởng rằng host 1 muốn thiết lập nối kết khác. Khi host 1 từ chối cố gắng thiết lập nối kết của host 2, host 2 hiểu rằng nó đã bị lừa bởi CR bị trùng lắp và sẽ từ bỏ nối kết đó.

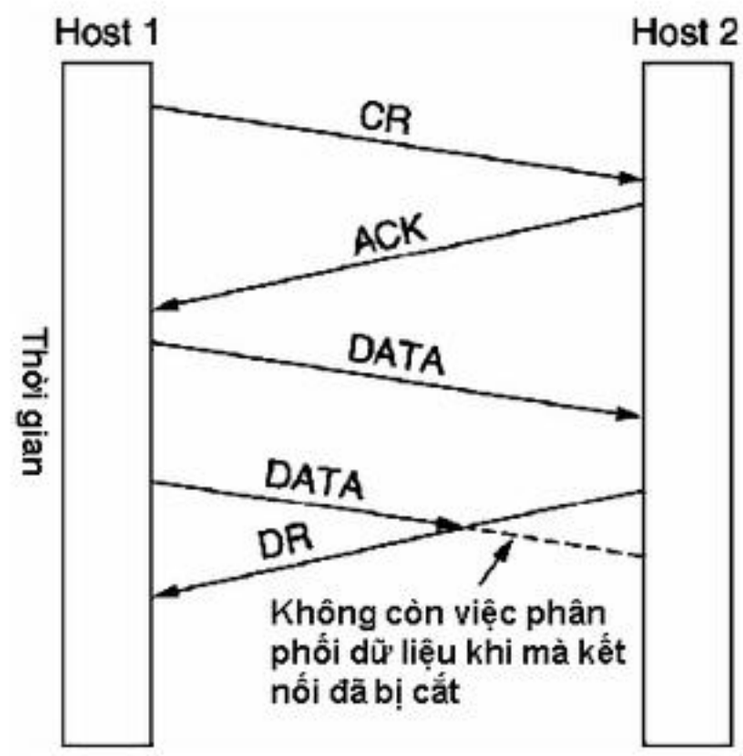
Trường hợp xấu nhất là cả hai TPDU CR và ACK của host 1 đều bị trùng lắp. Như trong ví dụ (b), host 2 nhận được một CR trễ và trả lời cho yêu cầu đó với số thứ tự khởi đầu ***y***. Giả sử, không may trong trả lời cho yêu cầu CR trước đó, host 2 thông báo số thứ tự khởi đầu của nó là ***z***. Báo nhận ở chiều thứ ba của host 1 lại bị trễ. Khi host 1 nhận được báo nhận ***ACK (seq=y, ACK=x)***, nó nhận ra rằng thông báo ***DATA (seq=x, ACK=z)*** bị trễ, do đó nó từ bỏ nối kết này.



H7.4 Khung cảnh việc bắt tay 3 chiều.
(a) Hoạt động bình thường.
(b) Bản CR bị trùng lắp.
(c) Cả CR và ACK đều bị trùng lắp

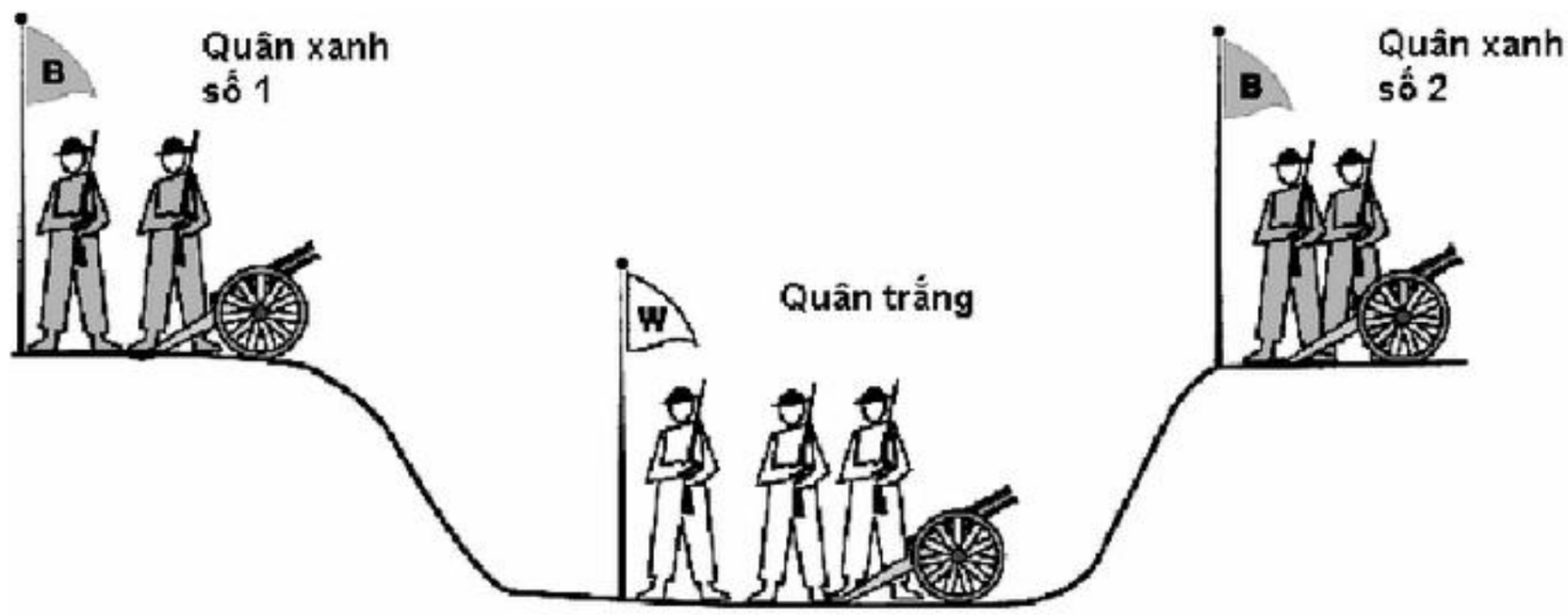
7.2.3 Giải phóng nối kết

Việc giải phóng nối kết đơn giản hơn thiết lập nối kết. Tuy nhiên, người ta sẽ còn gặp nhiều khó khăn không ngờ tới. Bây giờ chúng ta sẽ đề nghị hai kiểu giải phóng nối kết: dị bộ và đồng bộ. Kiểu dị bộ hoạt động như sau: khi một bên cắt nối kết, kết nối sẽ bị hủy bỏ (giống như trong hệ thống điện thoại). Kiểu đồng bộ làm việc theo phương thức ngược lại: khi cả hai đồng ý hủy bỏ nối kết, nối kết mới thực sự được hủy.
Giải phóng nối kết kiểu dị bộ là thô lỗ và có thể dẫn đến mất dữ liệu. Ví dụ tình huống trong Hình H7.5. Sau khi nối kết thành công, host 1 gởi một gói dữ liệu đến đúng host 2. Sau đó host 1 gởi tiếp một gói dữ liệu khác. Không may, host 2 gởi đi một yêu cầu cắt nối kết (DISCONNECT) trước khi gói dữ liệu thứ hai đến. Kết quả là kết nối được giải phóng và dữ liệu bị mất.



H7.5 Sự cắt kết nối một cách thô lỗ sẽ dẫn đến mất dữ liệu

Rõ ràng, chúng ta cần một giải pháp hữu hiệu hơn để tránh mất dữ liệu. Một giải pháp là sử dụng việc giải phóng nối kết đồng bộ, trong đó, mỗi host đều có trách nhiệm trong việc giải phóng nối kết. Một nút phải tiếp tục nhận dữ liệu sau khi đã gởi đi yêu cầu giải phóng nối kết (DISCONNECT REQUEST – CR) đến bên đối tác, cho đến khi nhận được chấp thuận hủy bỏ nối kết của bên đối tác đó. Người ta có thể hình dung giao thức như sau: đầu tiên host 1 nói: "Tôi xong rồi, anh xong chưa?". Nếu host 2 trả lời: "Tôi cũng xong, tạm biệt" thì kết nối coi như được giải phóng an toàn.
Tuy nhiên, giải pháp trên không phải lúc nào cũng chạy đúng. Có một bài toán nổi tiếng dùng để mô tả vấn đề, được gọi là bài toán "**hai sứ quân**" (Two army problem).



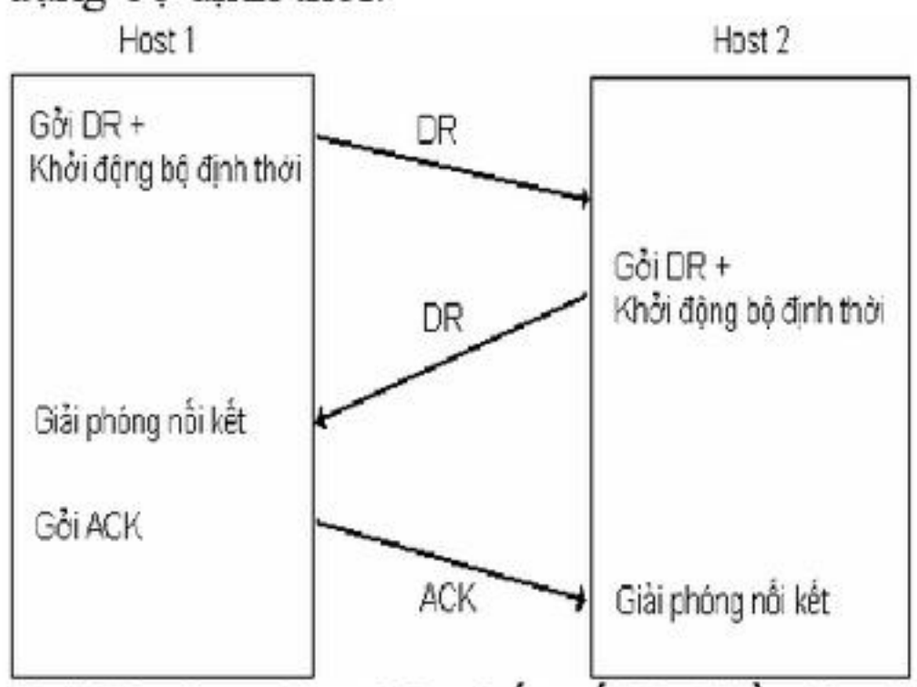
H7.6 Bài toán hai sứ quân

Có hai sứ quân đang dàn trận đánh nhau. Quân trắng dàn quân dưới thung lũng, quân xanh chia thành hai cánh quân chiếm lĩnh hai đỉnh đồi áng ngữ hai bên thung lũng đó. Chỉ huy của hai cánh quân xanh muốn thông báo và nhất trí với nhau về thời điểm cùng tấn công quân trắng. Do quân

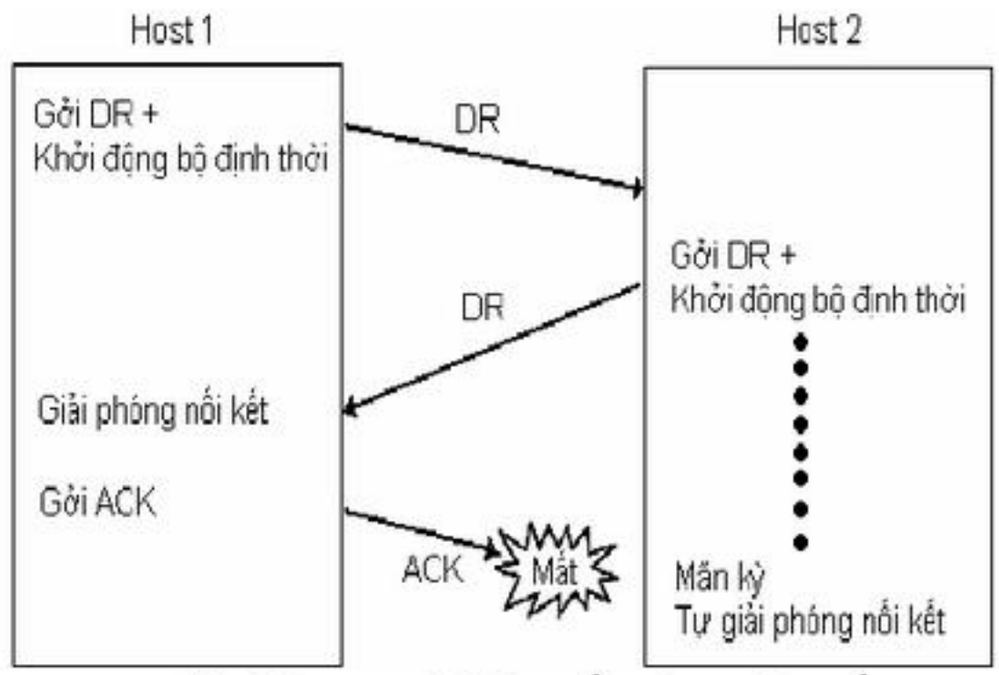
số hai cánh quân xanh cộng lại mới đủ sức thắng quân trắng, một cánh quân xanh tấn công riêng lẻ sẽ bị quân trắng tiêu diệt.
Hai cánh quân xanh muốn đồng bộ hóa cuộc tấn công của họ bằng cánh gởi các thông điệp qua lại. Nhưng những thông điệp đó phải chạy ngang qua thung lũng và có khả năng bị quân trắng phá hỏng. Câu hỏi ở đây là có giao thức nào đảm bảo sự thắng lợi của quân xanh hay không?
Giả sử chỉ huy cánh quân xanh số 1 gởi thông điệp đến chỉ huy cánh quân xanh số 2: "Tôi dự định tấn công vào lúc hoàng hôn ngày 14 tháng 12 năm 2004, có được không?". May mắn thay, chỉ huy cánh quân xanh số 2 nhận được thông điệp và trả lời "Đồng ý". Vậy cuộc tấn công có chắc xảy ra không? Không chắc, bởi vì chỉ huy cánh quân xanh số 2 không chắc câu trả lời của anh ta đến được chỉ huy của cánh quân số 1.
Bây giờ ta cải tiến giao thức thêm một bước: cho nó trở thành giao thức ba chiều: Bên cánh quân số 1 gởi bản hiệp đồng tấn công cho bên cánh quân số 2, bên cánh quân số 2 trả lời đồng ý, bên cánh quân 1 thông báo cho bên 2 nó đã biết được sự đồng ý của bên 2. Thế nhưng nếu thông báo cuối cùng của bên 1 bị mất thì sao? Bên 2 cũng sẽ không tấn công!
Nếu ta cố cải tiến thành giao thức n chiều đi nữa thì việc hiệp đồng vẫn thất bại nếu thông báo cuối cùng bị mất.
Ta có thể thấy mối tương đồng giữa bài toán hai sứ quân và giải pháp giải phóng nối kết. Thay vì hợp đồng tấn công, hai bên hợp đồng hủy nối kết!
Giải pháp cuối cùng là hai bên sử dụng phương pháp hủy nối kết ba chiều cùng với bộ định thời:

- Bên phát động việc hủy nối kết sẽ bật bộ định thời cho mỗi yêu cầu giải phóng nối kết của nó, nếu yêu cầu giải phóng nối kết bị mãn kỳ mà chưa nhận được trả lời của bên đối tác, nó sẽ gởi lại yêu cầu một lần nữa. Nếu yêu cầu hủy nối kết bị mãn kỳ liên tục N lần, bên phát động sẽ tự ý hủy bỏ nối kết đó.
- Bên đối tác khi nhận được yêu cầu hủy nối kết từ phía phát động, sẽ trả lời chấp thuận và cũng bật bộ định thời. Nếu mãn kỳ mà trả lời chấp thuận của nó không có báo trả từ phía phát động, bên đối tác sẽ tự hủy nối kết.

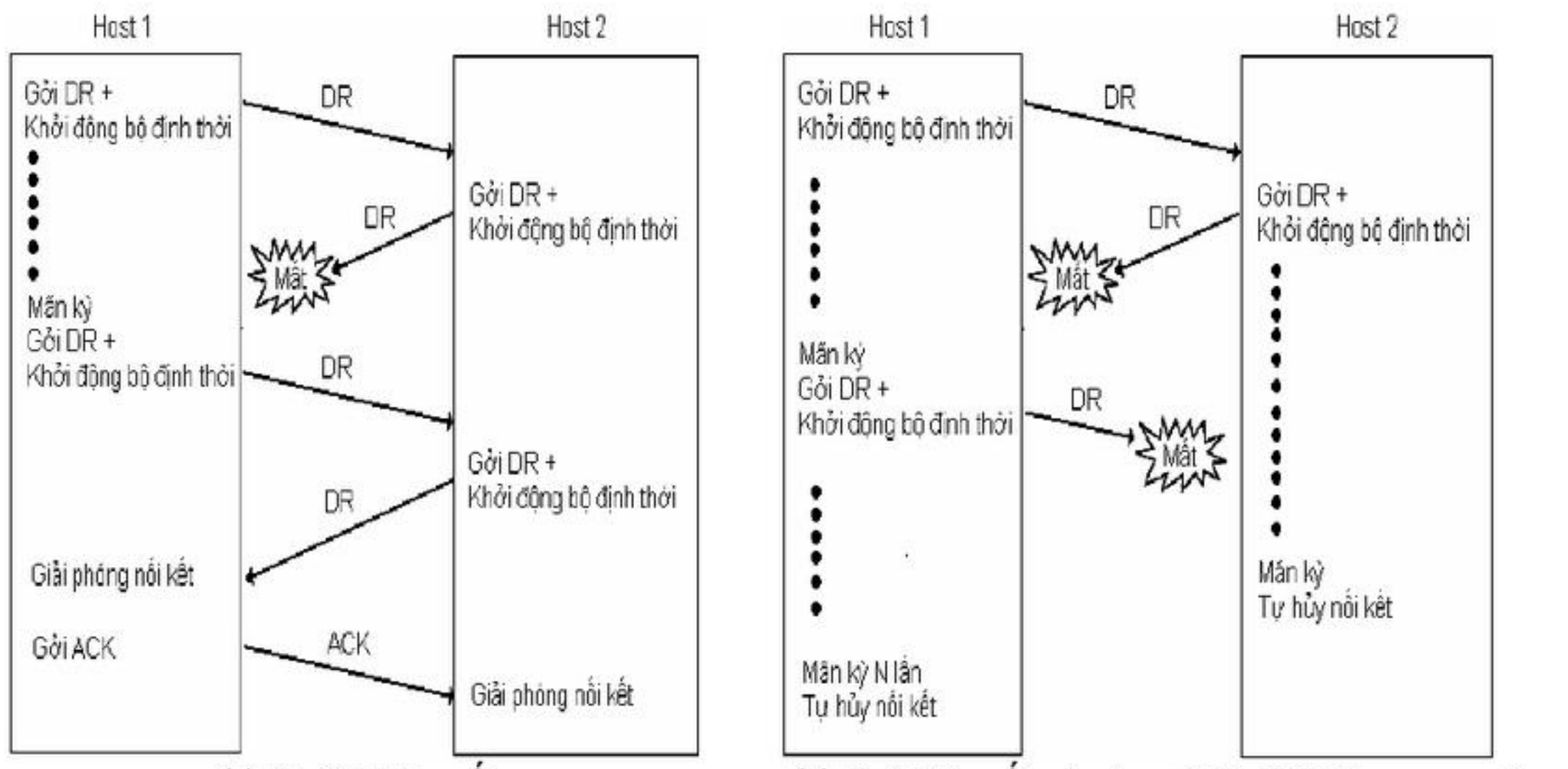
Hình H7.7 sẽ mô phỏng một số tình huống phát sinh trong quá trình hủy nối kết 3 chiều có sử dụng bộ định thời.



(a) Trường hợp hủy kết nối 3 chiều bình thường



(b) Khung ACK cuối cùng bị mất



(c) Trả lời bị mất *(d) Trả lời mất và các gói tin DR theo sau cũng bị mất*

H7.7 Một số tình huống hủy nối kết theo phương pháp 3 chiều

7.2.4 Điều khiển thông lượng

Điều khiển thông lượng trong tầng vận chuyển về cơ bản là giống giao thức cửa sổ trượt trong tầng liên kết dữ liệu, nhưng kích thước cửa sổ của bên gởi và bên nhận là khác nhau. Mỗi host có thể có quá nhiều kết nối tại một thời điểm, vì thế nó không chắc là có thể đảm bảo cung cấp đủ số lượng buffer cho mỗi kết nối nhằm thực hiện đúng giao thức cửa sổ trượt. Cần phải có sơ đồ cung cấp buffer động.

Trước tiên, bên gởi phải gởi đến bên nhận một yêu cầu dành riêng số lượng buffer để chứa các gói bên gởi gởi đến. Bên nhận cũng phải trả lời cho bên gởi số lượng buffer tối đa mà nó có thể cung cấp. Mỗi khi báo nhận ACK cho một gói tin có số thứ tự SEQ_NUM, bên nhận cũng phải gởi kèm theo thông báo cho bên gởi biết là lượng buffer còn lại là bao nhiêu để bên gởi không làm ngập bên nhận.

Ví dụ sau sẽ mô phỏng một tình huống trao đổi thông tin giữa hai máy A và B.

	A	Thông điệp	B	Giải thích
1	→	<yêu cầu 8 buffers>	→	A muốn B cung cấp 8 buffers
2	←	<ack = 0, buf = 4>	←	B chỉ cấp cho A 4 buffers thôi
3	→	<seq = 0, data = m0>	→	A còn lại 3 buffers
4	→	<seq = 1, data = m1>	→	A còn lại 2 buffers
5	→	<seq = 2, data = m2>	…	Thông điệp bị mất, nhưng A nghĩ nó còn 1 buffer
6	←	<ack = 1, buf = 3>	←	B báo nhận cho thông điệp 0 và 1, còn 3 buffers
7	→	<seq = 3, data = m3>	→	A còn lại 1 buffer
8	→	<seq = 4, data = m4>	→	A không còn buffer nào và phải dừng
9	→	<seq = 2, data = m2>	→	Thông điệp thứ 2 của A mãn kỳ và được truyền lại
10	←	<ack = 4, buf = 0>	←	Mọi thứ đã được báo nhận, nhưng A vẫn nghẽn

11	←	<ack = 4, buf = 1>	←	A có thể gởi 1 gói tin thứ 5
12	←	<ack = 4, buf = 2>	←	B có thêm 1 buffer nữa
13	→	<seq = 5, data = m5>	→	A còn lại 1 buffer
14	→	<seq = 6, data = m6>	→	A nghẽn một lần nữa
15	←	<ack = 6, buf = 0>	←	A vẫn còn nghẽn
16	…	<ack = 6, buf = 4>	←	Khả năng dẫn đến deadlock

H7.8 Ví dụ một phiên giao dịch giữa hai thực thể tầng vận chuyển

Một vấn đề tiềm tàng trong sơ đồ dùng buffer động là cơ chế hoạt động của nó có thể dẫn đến deadlock. Ví dụ trong hàng 16, nếu báo nhận <ack = 6, buf = 4> của bên B bị mất, cả hai bên A và B đều rơi vào trạng thái deadlock. Để tránh tình trạng này, nên cho các host định kỳ gởi các báo nhận và trạng thái buffer lên mọi kết nối vận chuyển của chúng.

7.3 Tầng vận chuyển trong mạng Internet

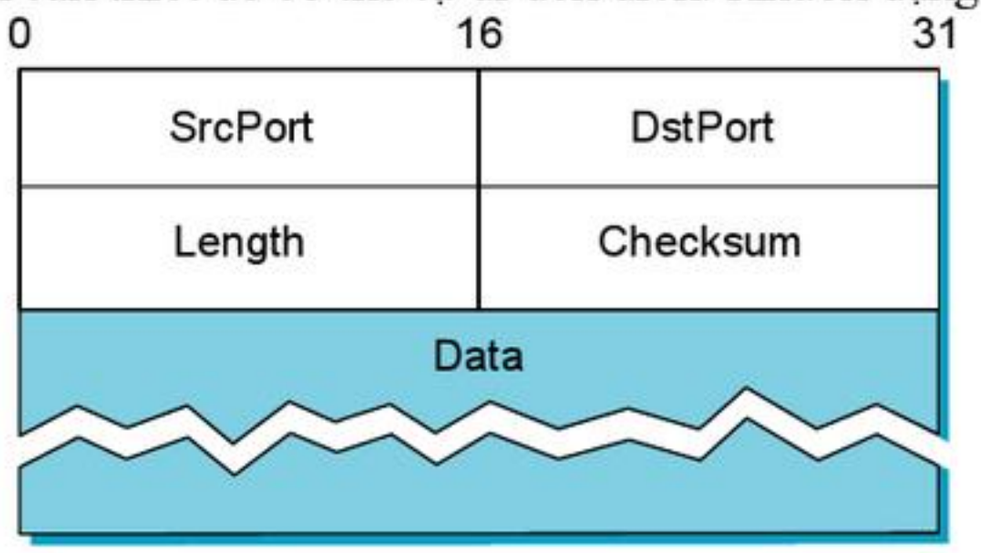
Trong Internet, tầng vận chuyển được thiết kế ra với ý đồ thực hiện các nhiệm vụ sau:

- Đảm bảo việc phân phối thông điệp qua mạng.
- Phân phối các thông điệp theo thứ tự mà chúng được gởi.
- Không làm trùng lắp thông điệp.
- Hỗ trợ những thông điệp có kích thước lớn.
- Hỗ trợ cơ chế đồng bộ hóa.
- Hỗ trợ việc liên lạc của nhiều tiến trình trên mỗi host.

Tầng vận chuyển trong Internet cũng hỗ trợ hai phương thức hoạt động không nối kết và có nối kết với hai giao thức liên lạc tương ứng là UDP và TCP.

7.3.1 Giao thức UDP (User Datagram Protocol)

UDP là dịch vụ truyền dữ liệu dạng không nối kết. Không có thiết lập nối kết giữa hai bên truyền nhận, do đó gói tin UDP (segment) có thể xuất hiện tại nút đích bất kỳ lúc nào. Các segment UDP tự thân chứa mọi thông tin cần thiết để có thể tự đi đến đích. Khuôn dạng của chúng như sau:



H7.9 Khuôn dạng của một segment UDP

Giải thích:

- SrcPort: Địa chỉ cổng nguồn, là số hiệu của tiến trình gởi gói tin đi.
- DstPort: Địa chỉ cổng đích, là số hiệu của tiến trình sẽ nhận gói tin.
- Length: Tổng chiều dài của segment, tính luôn cả phần header.
- Checksum: Là phần kiểm tra lỗi. UDP sẽ tính toán phần kiểm tra lỗi tổng hợp trên phần header, phần dữ liệu và cả phần header ảo. Phần header ảo chứa 3 trường trong IP header: địa chỉ IP nguồn, địa chỉ IP đích, và trường chiều dài của UDP. Phương thức tính toán như sau:

```
u_short
cksum(u_short *buf, int count)
{
  register u_long sum = 0;
  while (count--)
  {
    sum += *buf++;
    if (sum & 0xFFFF0000)
    {
      /* bit carry xuất hiện, vì thế gấp và cộng dồn nó lại */
      sum &= 0xFFFF;
      sum++;
    }
  }
  return ~(sum & 0xFFFF);
}
```

Xem thông điệp là một chuỗi các số nguyên 16 bits. Cộng dồn các số nguyên này từng bit một. Kết quả cộng dồn cuối cùng chính là phần kiểm tra lỗi.

- Data: Phần dữ liệu hai bên gởi cho nhau.

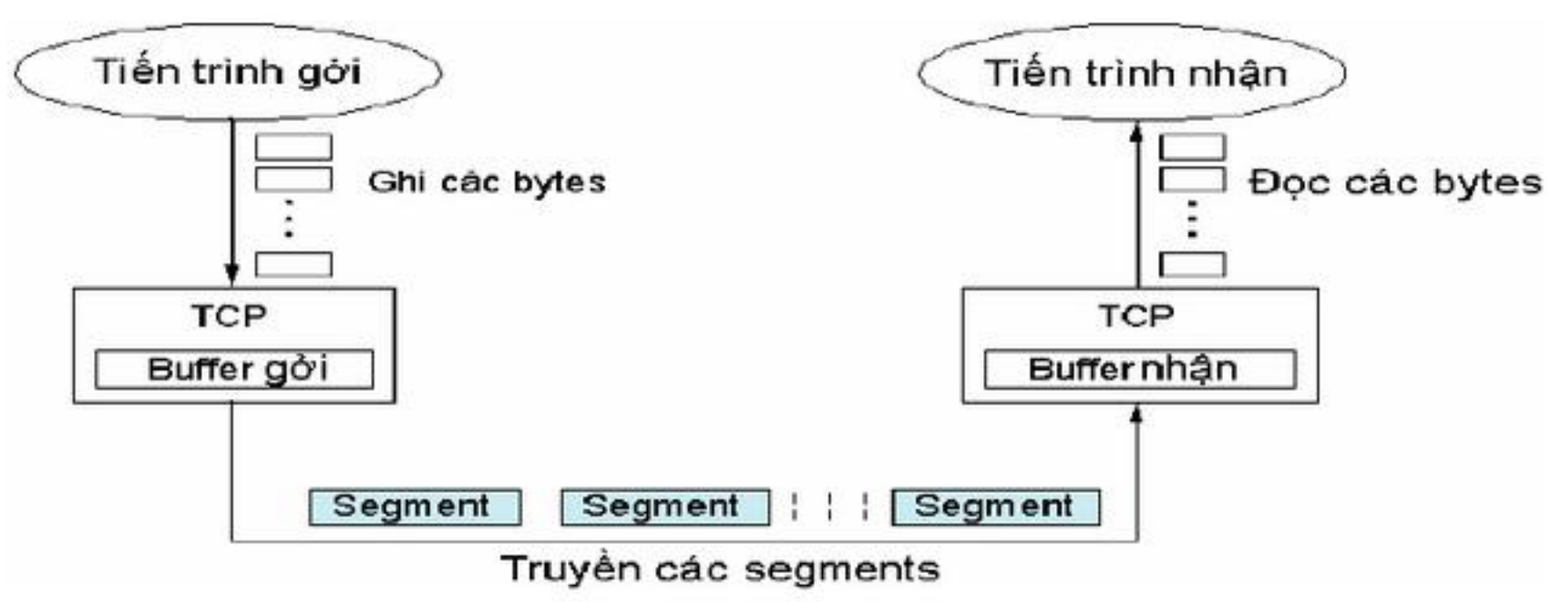
UDP hoạt động không tin cậy cho lắm, vì: Không có báo nhận dữ liệu từ trạm đích; không có cơ chế để phát hiện mất gói tin hoặc các gói tin đến không theo thứ tự; không có cơ chế tự động gởi lại những gói tin bị mất; không có cơ chế điều khiển luồng dữ liệu, và do đó có thể bên gởi sẽ làm ngập bên nhận.

7.3.2 Giao thức TCP (Transmission Control Protocol)

Ngược với giao thức UDP, TCP là giao thức vận chuyển tinh vi hơn, dùng để cung cấp dịch vụ vận chuyển tin cậy, hướng nối kết theo kiểu truyền thông tin bằng cách phân luồng các bytes. TCP là giao thức truyền hai hướng đồng thời, nghĩa là mỗi một nối kết hỗ trợ hai luồng bytes chạy theo hai hướng. Nó cũng bao gồm một cơ chế điều khiển thông lượng cho mỗi luồng bytes này, để cho phép bên nhận giới hạn lượng dữ liệu mà bên gởi có thể truyền tại một thời điểm nào đó. TCP cũng hỗ trợ cơ chế đa hợp, cho phép nhiều tiến trình trên một máy tính có thể đồng thời thực hiện đối thoại với đối tác của chúng.

7.3.2.1 Hai đầu mút truyền dữ liệu với nhau như thế nào?

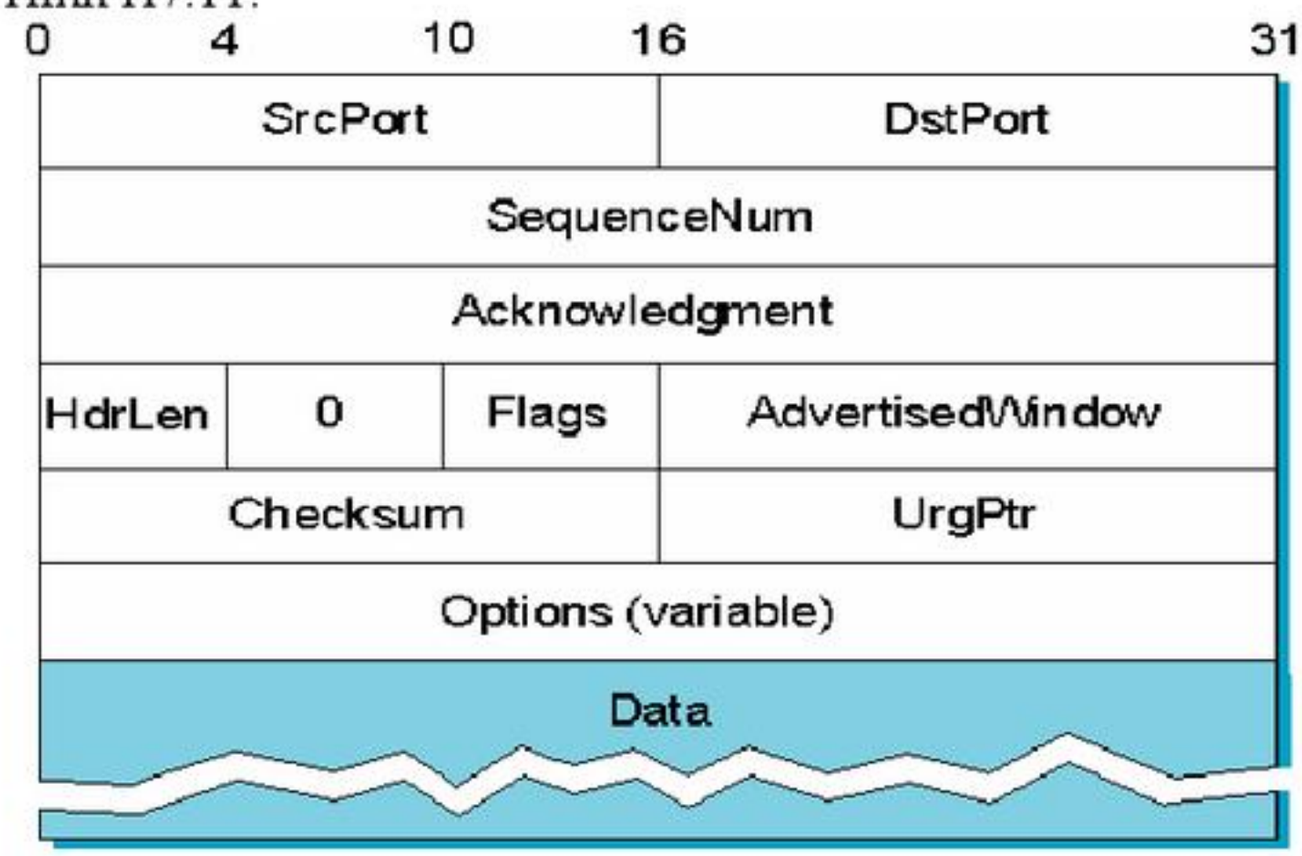
TCP là giao thức hướng byte, nghĩa là bên gởi ghi các bytes lên nối kết TCP, bên nhận đọc các bytes từ nối kết TCP đó. Mặc dù TCP mô tả dịch vụ mà nó cung cấp cho tầng ứng dụng là theo kiểu "luồng các bytes", nhưng tự thân TCP không truyền từng byte một qua mạng Internet. Thay vào đó, thực thể TCP trên máy nguồn trữ tạm đủ số bytes phát ra từ tiến trình gởi để tạo nên một gói tin có kích thước hợp lý rồi mới gởi gói tin đó đến thực thể TCP ngang hàng bên máy đích. Thực thể TCP bên máy đích sẽ bóc các bytes dữ liệu trong gói tin ra và đặt chúng vào buffer của nó. Tiến trình bên nhận từ đó có thể đọc các bytes từ buffer này tùy thích. Quá trình truyền nhận trên được mô phỏng trong Hình H7.10.



H7.10. Cách thức TCP quản lý luồng các bytes

7.3.2.2 Khuôn dạng TCP Segment

Các gói tin được trao đổi bởi hai thực thể TCP trong Hình H7.10 được gọi là các ***segment*** (đoạn), do mỗi gói tin mang theo một đoạn của cả một luồng các bytes. Mỗi segment có một header như được chỉ ra trong Hình H7.11.

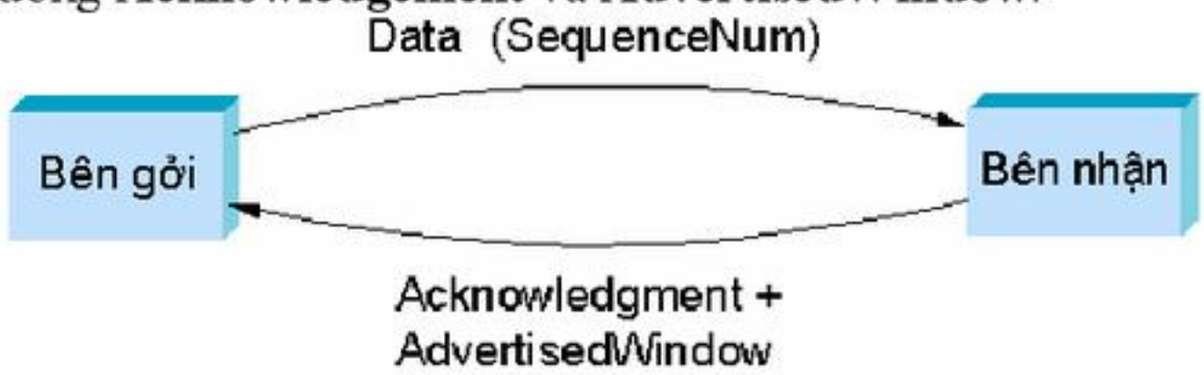


H.11 Khuôn dạng TCP header

Giải thích:

- Các trường **SrcPort** và **DstPort** chỉ ra địa chỉ cổng nguồn và đích, giống như trong UDP. Hai trường này cùng với hai địa chỉ IP nguồn và đích sẽ được kết hợp với nhau để định danh duy nhất một kết nối TCP. Nghĩa là một kết nối TCP sẽ được định danh bởi một bộ 4 trường

  (Cổng nguồn, Địa chỉ IP nguồn, Cổng đích, Địa chỉ IP đích)
- Các trường **Acknowledgement**, **SequenceNum** và **AdvertisedWindow** tất cả được sử dụng trong giải thuật cửa sổ trượt của TCP. Bởi vì TCP là giao thức hướng byte, nên mỗi byte của dữ liệu sẽ có một số thứ tự; trường **SequenceNum** chứa số thứ tự của byte đầu tiên của một dãy các bytes chứa trong một segment. Các trường **Acknowledgement** và **AdvertisedWindow** được dùng để thông báo tiến độ nhận các bytes trong luồng dữ liệu và khả năng tiếp nhận chúng. Để đơn giản hóa vấn đề, chúng ta bỏ qua sự thật là dữ liệu có thể chạy theo hai chiều, ở đây ta đưa ra sơ đồ như sau: bên gởi sẽ gởi một segment dữ liệu, byte dữ liệu đầu tiên trong segment đó sẽ có số thứ tự là **SequenceNum**; bên nhận sẽ báo nhận bằng các trường **Acknowledgement** và **AdvertisedWindow**.



H7.12 Gởi dữ liệu và báo nhận

  Cách thức hai bên sử dụng các trường trên như thế nào sẽ được trình bày trong phần điều khiển luồng dữ liệu.
- Trường **Flags** dài 6 bits được sử dụng để chứa thông tin điều khiển giữa hai bên sử dụng giao thức TCP. Một bit trong trường này là một cờ, cụ thể như sau: **SYN, FIN, RESET, PUSH, URG, ACK.** Hai cờ **SYN** và **FIN** được dùng để thiết lập và giải phóng nối kết. Cờ **ACK** được đặt mỗi khi trường **Acknowledgement** là hợp lệ. Cờ **URG** được dùng để đánh dấu segment này chứa dữ liệu khẩn cấp. Khi cờ này được đặt, trường **UrgPtr** sẽ chỉ ra nơi bắt đầu của dữ liệu không khẩn cấp (dữ liệu khẩn cấp luôn nằm ở đầu của phần dữ liệu).

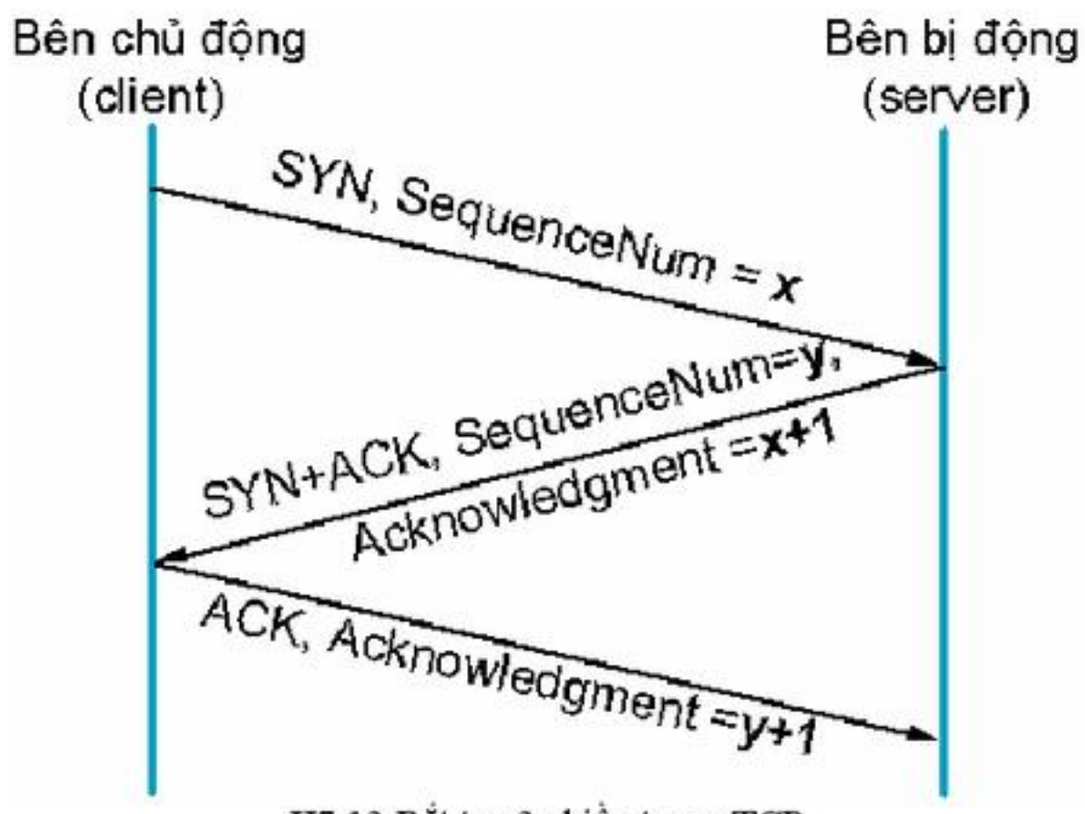
Cờ **PUSH** báo hiệu cho bên nhận rằng bên gởi đã dùng thao tác PUSH, tức là bên gởi đã không chờ nhận đủ các bytes để lấp đầy một segment, trong buffer gởi dù có bao nhiêu bytes dữ liệu cũng được bên gởi đóng vào segment và gởi đi. Cuối cùng, cờ **RESET** được dùng để thông báo rằng bên nhận đã bị rối (ví dụ như nó đã nhận một segment mà đáng lẽ ra không phải là segment đó), vì thế nó muốn hủy bỏ nối kết.

- Trường **Checksum** được sử dụng chính xác giống như trong giao thức UDP.
- Do header của TCP có độ dài thay đổi, nên trường **HdrLen** sẽ chỉ ra độ dài cụ thể của phần header này.

7.3.2.3 Bắt tay trong TCP

TCP sử dụng giao thức bắt tay 3 chiều.

- **Bước 1**: Cient (bên chủ động) gởi đến server một segment yêu cầu nối kết, trong đó chứa số thứ tự khởi đầu mà nó sẽ dùng (**Flags = SYN, SequenceNum = x**).
- **Bước 2**: Server trả lời cho client bằng một segment, trong đó báo nhận rằng nó sẵn sàng nhận các byte dữ liệu bắt đầu từ số thứ tự x+1 (**Flags = ACK, Ack = x + 1**) và cũng báo rằng số thứ tự khởi đầu của bên server là y (**Flags = SYN, SequenceNum = y**).
- **Bước 3**: Cuối cùng client báo cho server biết, nó đã biết số thứ tự khởi đầu của server là y (**Flags = ACK, Ack = y+1**).
-

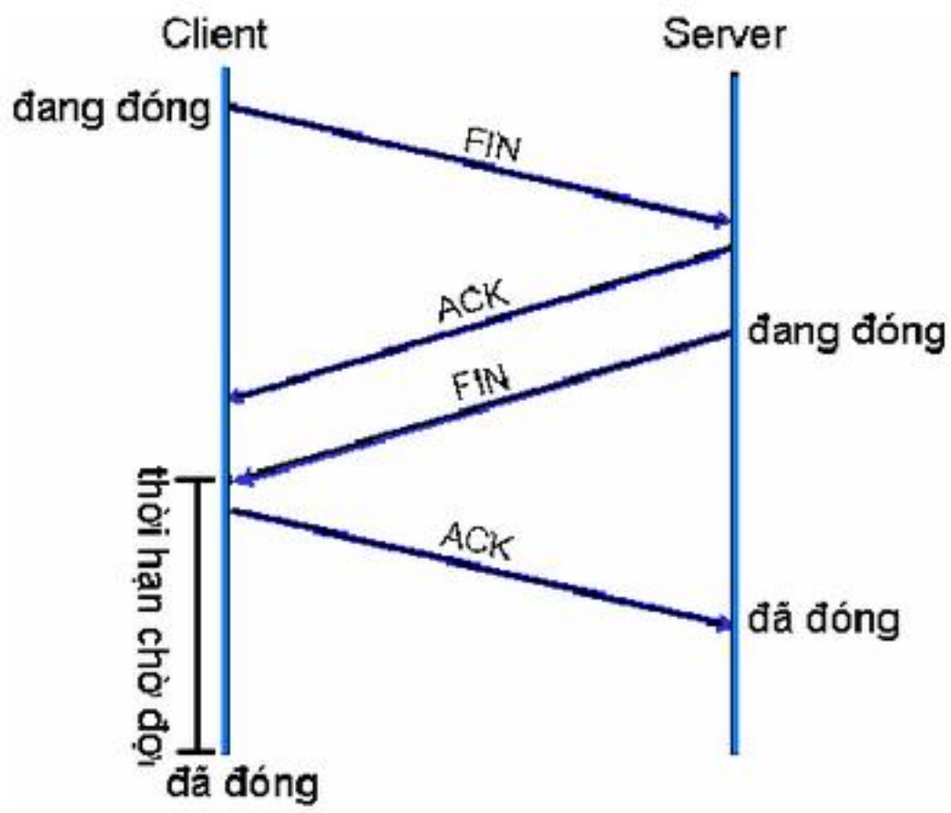


H7.12 Bắt tay 3 chiều trong TCP

7.3.2.4 Hủy bắt tay trong TCP

Việc hủy bắt tay trong TCP được thực hiện qua 4 bước:

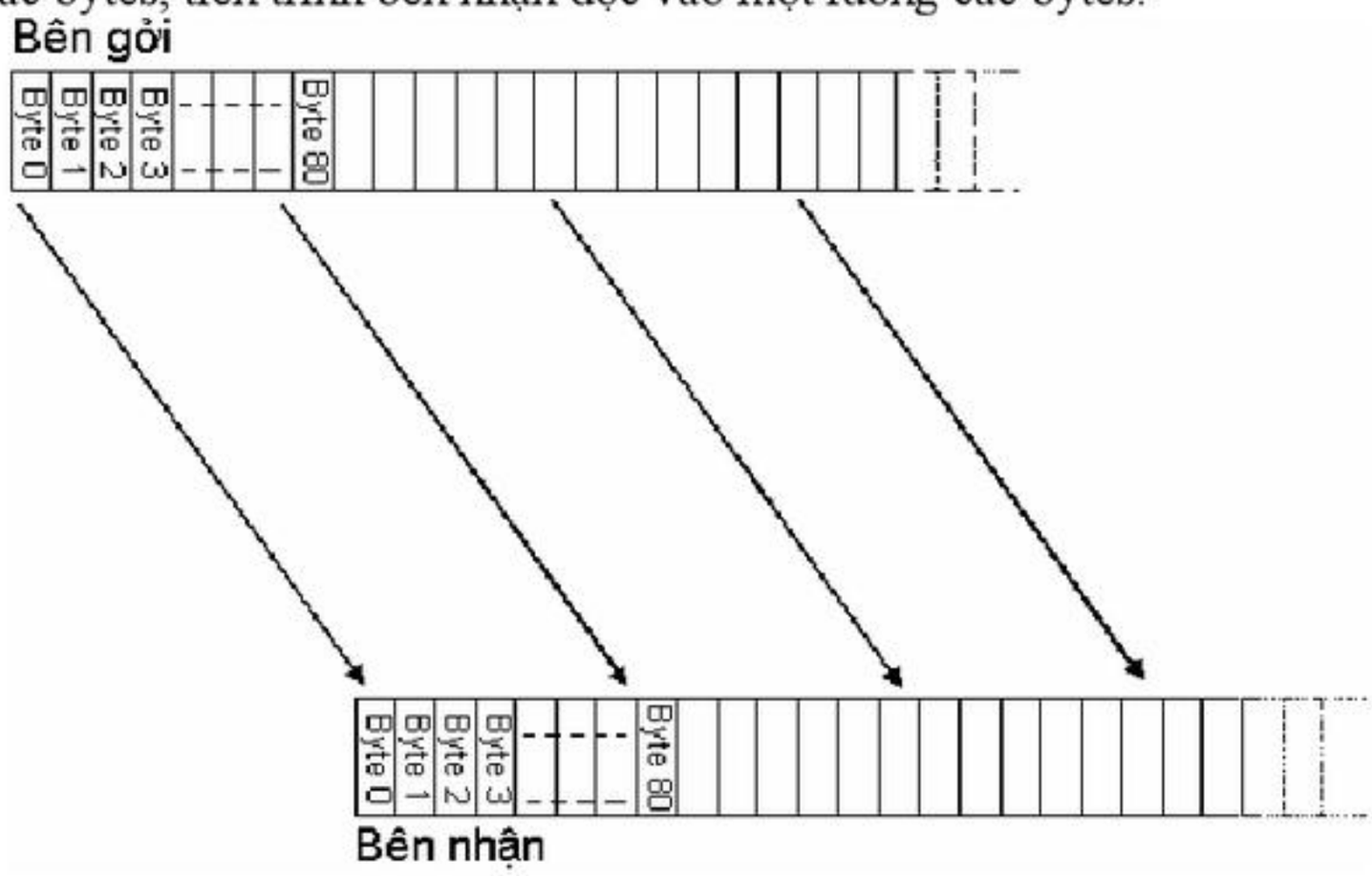
- **Bước 1**: Cient (bên chủ động) gởi đến server một segment yêu cầu hủy nối kết (**Flags = FIN**).
- **Bước 2**: Server nhận được một segment FIN, sẽ trả lời bằng một segment ACK. Sau khi đã hoàn tất hết mọi thứ để đóng nối kết, server sẽ gởi cho client tiếp một segment FIN.
- **Bước 3:** Client nhận được FIN sẽ trả lời ACK sau đó nó sẽ chuyển sang trạng thái chờ đợi có định hạn. Trong thời gian chờ đợi này, client sẽ trả lời ACK cho mọi khung FIN. Hết thời gian chờ đợi, client sẽ thật sự đóng nối kết.
- **Bước 4**: Server khi nhận được ACK sẽ thật sự đóng nối kết.



H7.13 Hủy nối kết trong TCP

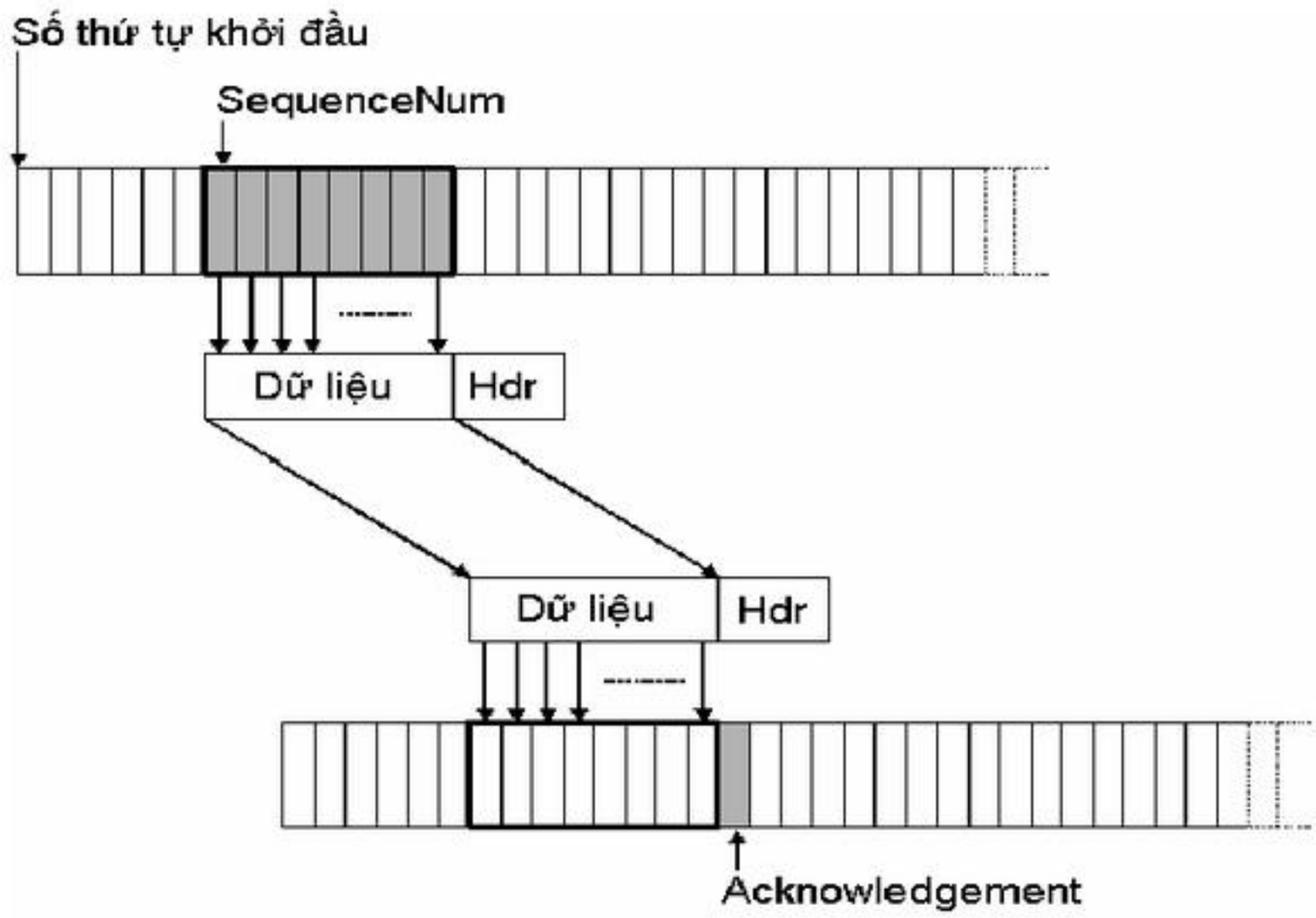
7.3.2.5 Điều khiển thông lượng trong TCP

TCP dùng phương pháp điều khiển thông lượng "cửa sổ trượt với kích thước cửa sổ động". Nhắc lại rằng TCP là giao thức hướng bytes. Ta có thể tưởng tượng hình ảnh sau: tiến trình bên gởi ghi ra một luồng các bytes, tiến trình bên nhận đọc vào một luồng các bytes.



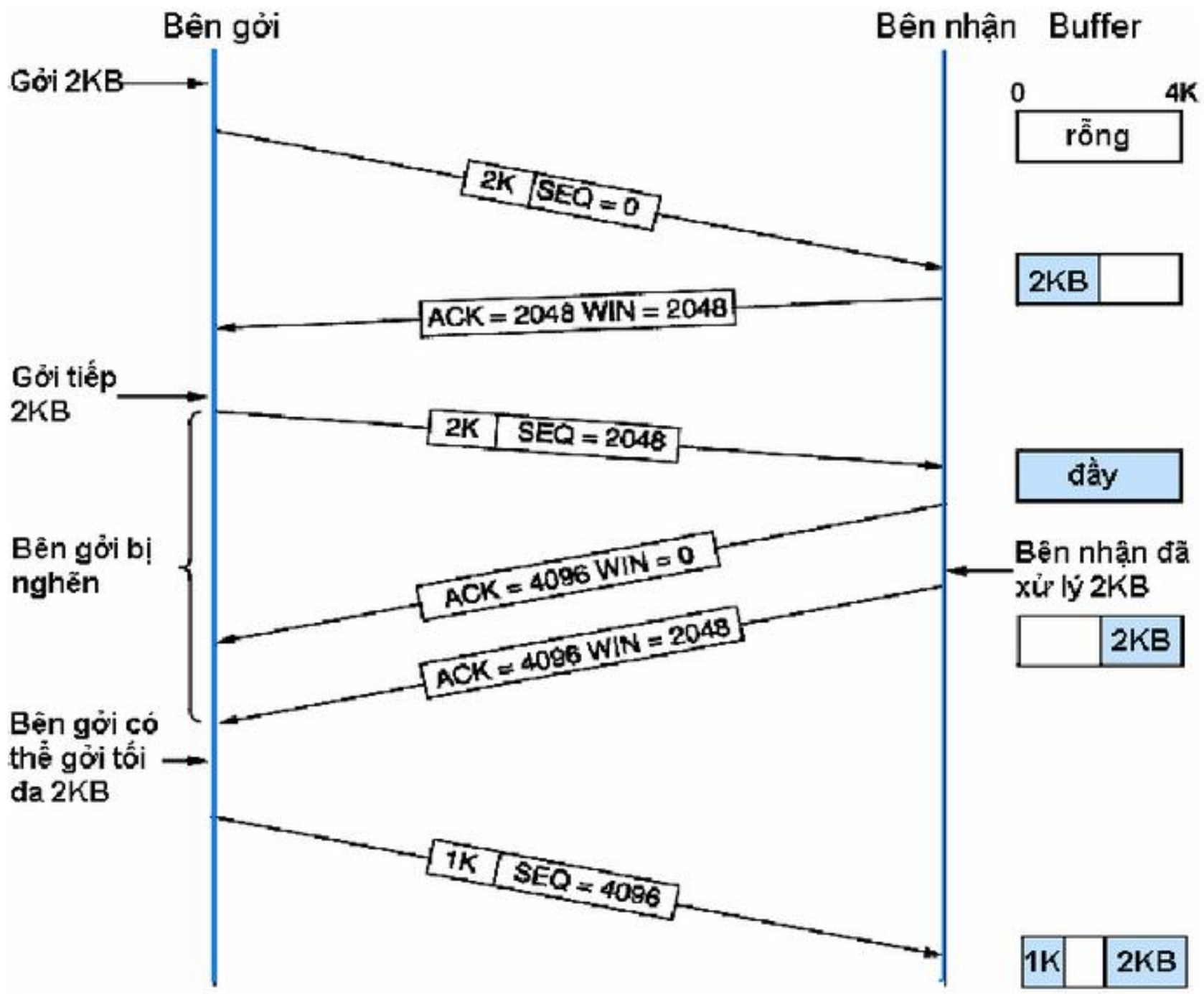
H7.14 Truyền nhận theo luồng bytes

Như đã nói, TCP không truyền nhận dữ liệu cho ứng dụng từng byte một mà nó trữ tạm các bytes trong buffer đến khi đủ đóng gói thành một segment thì mới truyền đi.



H7.15 Các luồng bytes được phân đoạn như thế nào

Byte đầu tiên của mỗi luồng bytes sẽ được đánh số bằng "số thứ tự khởi đầu", và số này được dàn xếp trong giai đoạn bắt tay 3 chiều. Trường **SequenceNum** trong TCP segment chứa số thứ tự của byte đầu tiên nằm trong segment đó. Cũng như trong giao thức cửa sổ trượt, khi bên nhận nhận được **n** bytes trong một segment, bắt đầu từ byte thứ **SequenceNum**, nó sẽ báo nhận tốt **n** bytes này và chờ nhận tiếp từ byte thứ **Acknowledgement** (Acknowledgement = SequenceNum + n).



H7.16 Ví dụ về điều khiển thông lượng trong TCP

Do kích thước cửa sổ là động, nên trong mỗi khung báo nhận của mình, bên nhận đính kèm theo thông báo về kích thước cửa sổ sẵn dùng của nó (lượng buffer còn trống), đó chính là trường **AdvertisedWindow** trong TCP segment. Lần sau, bên gởi sẽ không được gởi lượng bytes vượt quá **AdvertisedWindow**.

Trong ví dụ trên, lúc đầu bên nhận có kích thước buffer là 4 KB. Bên gởi đặt số thứ tự khởi đầu là 0, sau đó truyền 2 KB. Buffer bên nhận còn lại 2 KB rỗng, do đó nó báo nhận "**Acknowledgement = 2048, AdvertisedWindow = 2048**". Bên gởi gởi tiếp 2 KB, khi đó buffer bên nhận bị đầy, nó liền báo nhận "**Acknowledgement = 4096, AdvertisedWindow = 0**". Không còn buffer nhận, nên bên gởi sẽ tạm thời bị nghẽn. Sau khi bên nhận xử lý xong 2 KB, nó liền báo "**Acknowledgement = 4096, AdvertisedWindow = 2048**". Lúc này bên gởi có thể gởi tiếp tối đa là 2 KB, nhưng nó chỉ còn 1 KB dữ liệu để gởi mà thôi.

Chương 8: CÁC ỨNG DỤNG MẠNG

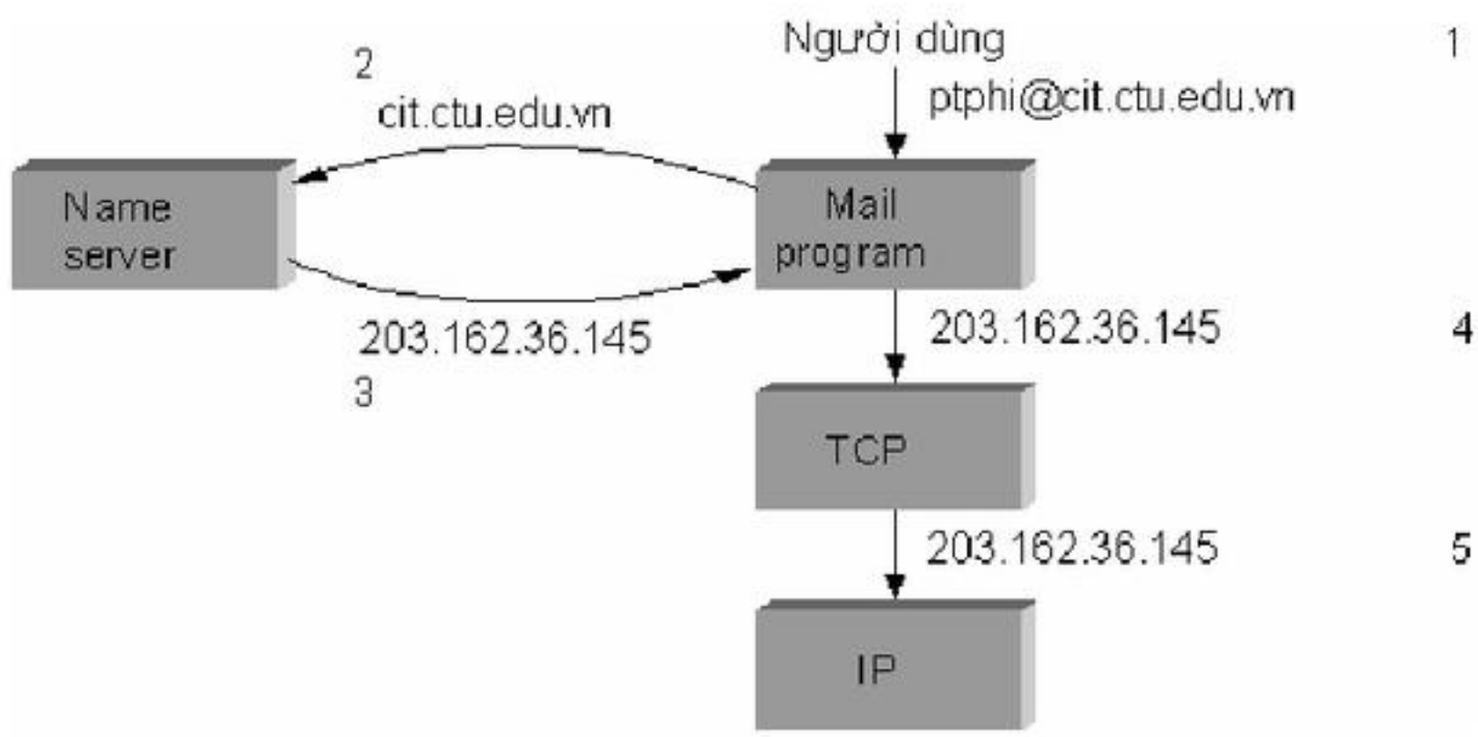
Chương này sẽ tìm hiểu một số ứng dụng mạng phổ biến hiện nay, chủ yếu tập trung vào giao thức hoạt động của chúng.
Ví dụ đầu tiên được xem xét là dịch vụ tên phân tán, dịch vụ đầu tiên được cài đặt trong một mạng máy tính. Về thực chất dịch vụ tên là cái mà các ứng dụng khác phải phụ thuộc vào. Một server phục vụ tên thường được sử dụng bởi các ứng dụng khác hơn là bởi con người.
Sau đó, các ứng dụng mạng truyền thống và phổ biến sẽ được giới thiệu, bao gồm các dịch vụ MAIL, WEB và FTP.
Cũng cần nói trước rằng, những dịch vụ mạng vừa nói sẽ dựa trên hai giao thức vận chuyển đã được đề cập trong chương 6 là TCP và UDP.

8.1 Dịch vụ tên (DNS)

Cho đến bây giờ, chúng ta vẫn dùng địa chỉ để định danh các host. Trong khi rất thuận tiện cho việc xử lý của các router, các địa chỉ số không thân thiện với người dùng lắm. Vì lý do này, các host thường được gán cho một cái tên thân thiện và dịch vụ tên được sử dụng để ánh xạ từ cái tên thân thiện với người dùng này sang địa chỉ số vốn rất thân thiện với các router. Dịch vụ như vậy thường là ứng dụng đầu tiên được cài đặt trong một mạng máy tính do nó cho phép các ứng dụng khác tự do định danh các host bằng tên thay vì bằng địa chỉ. Dịch vụ tên thường được gọi là phần trung gian (middleware) vì nó lấp đầy khoảng cách giữa các ứng dụng khác và lớp mạng phía dưới.
Tên host và địa chỉ host khác nhau ở hai điểm quan trọng. Thứ nhất, tên host thường có độ dài thay đổi và dễ gợi nhớ, vì thế nó giúp người dùng dễ nhớ hơn. Thứ hai, tên thường không chứa thông tin gì để giúp mạng định vị (chuyển các gói tin đến) host. Địa chỉ, ngược lại, lại hàm chứa thông tin vạch đường trong đó.
Trước khi đi vào chi tiết cách thức đặt tên cho các host trong mạng như thế nào, chúng ta đi định nghĩa một số thuật ngữ trước:

- ***Không gian tên*** (name space) định nghĩa tập các tên có thể có. Một không gian tên có thể là phẳng (flat) – một tên không thể được chia thành các thành phần nhỏ hơn, hoặc phân cấp.
- Hệ thống tên duy trì một ***tập các ánh xạ*** (collection of bindings) từ tên sang giá trị. Giá trị có thể là bất cứ thứ gì chúng ta muốn hệ thống tên trả về khi ta cấp cho nó một tên để ánh xạ; trong nhiều trường hợp giá trị chính là địa chỉ host.
- Một ***cơ chế phân giải*** (resolution mechanism) là một thủ tục mà khi được gọi với tham số là một tên, sẽ trả về một giá trị tương ứng.
- Một ***server tên*** (name server) là một kết quả cài đặt cụ thể của một cơ chế phân giải luôn sẵn dùng trên mạng và có thể được truy vấn bằng cách gởi đến nó một thông điệp.

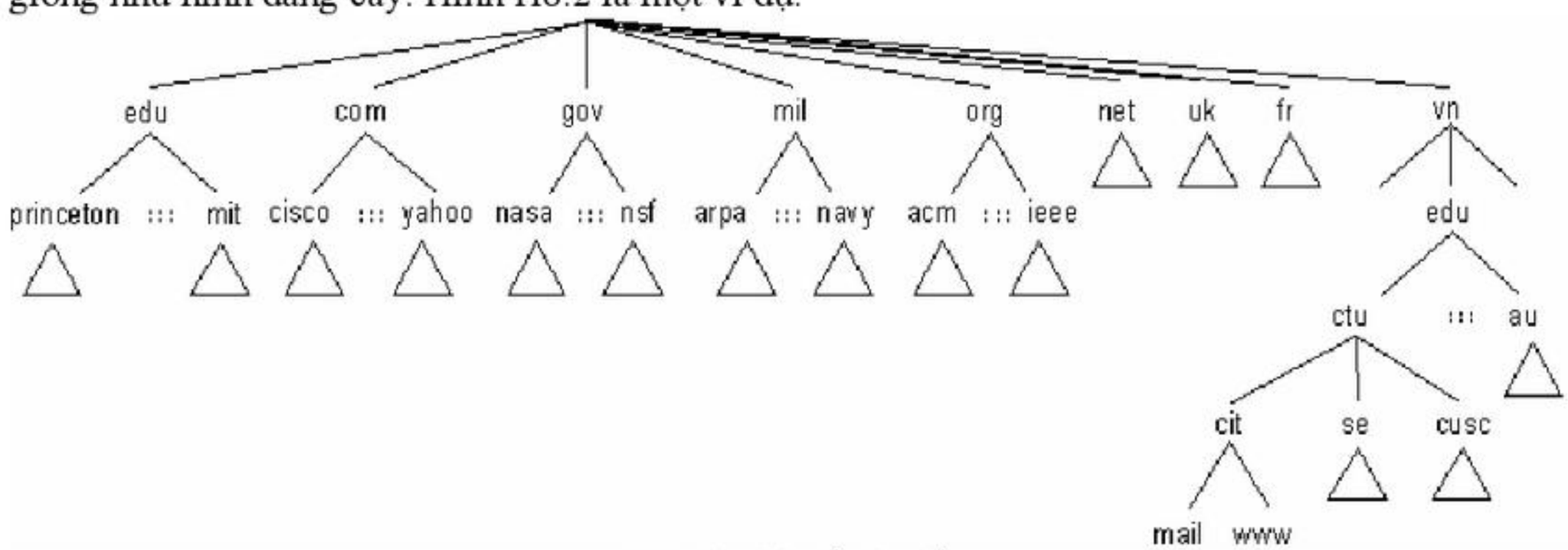
Mạng Internet đã có sẵn một hệ thống đặt tên được phát triển tốt, gọi là ***hệ thống tên miền*** (domain name system – DNS). Vì thế chúng ta sẽ dùng DNS làm cơ sở để thảo luận về vấn đề đặt tên cho các host.
Khi người dùng đưa một tên host đến một ứng dụng (có thể tên host đó là một phần của một tên hỗn hợp như địa chỉ email chẳng hạn), ứng dụng này sẽ liên hệ với hệ thống tên để dịch tên host sang địa chỉ host. Sau đó ứng dụng liền tạo một nối kết đến host đó thông qua giao thức TCP chẳng hạn. Hiện trạng được mô tả trong hình H8.1.



H8.1 Tên máy được dịch sang địa chỉ, các số từ 1-5 thể hiện trình tự các bước xử lý

8.1.1 Miền phân cấp

DNS cài đặt không gian tên phân cấp dùng cho các đối tượng trên Internet. Các tên DNS được xử lý từ phải sang trái, sử dụng các dấu chấm (.) làm ký tự ngăn cách. (Mặc dù các tên DNS được xử lý từ phải qua trái, người dùng thường đọc chúng từ trái sang phải). Ví dụ tên miền của một host là ***mail.cit.ctu.edu.vn***. Chú ý rằng các tên miền được sử dụng để đặt tên các đối tượng trên Internet, không phải chỉ được dùng để đặt tên máy. Ta có thể mường tượng cấu trúc phân cấp của DNS giống như hình dáng cây. Hình H8.2 là một ví dụ.

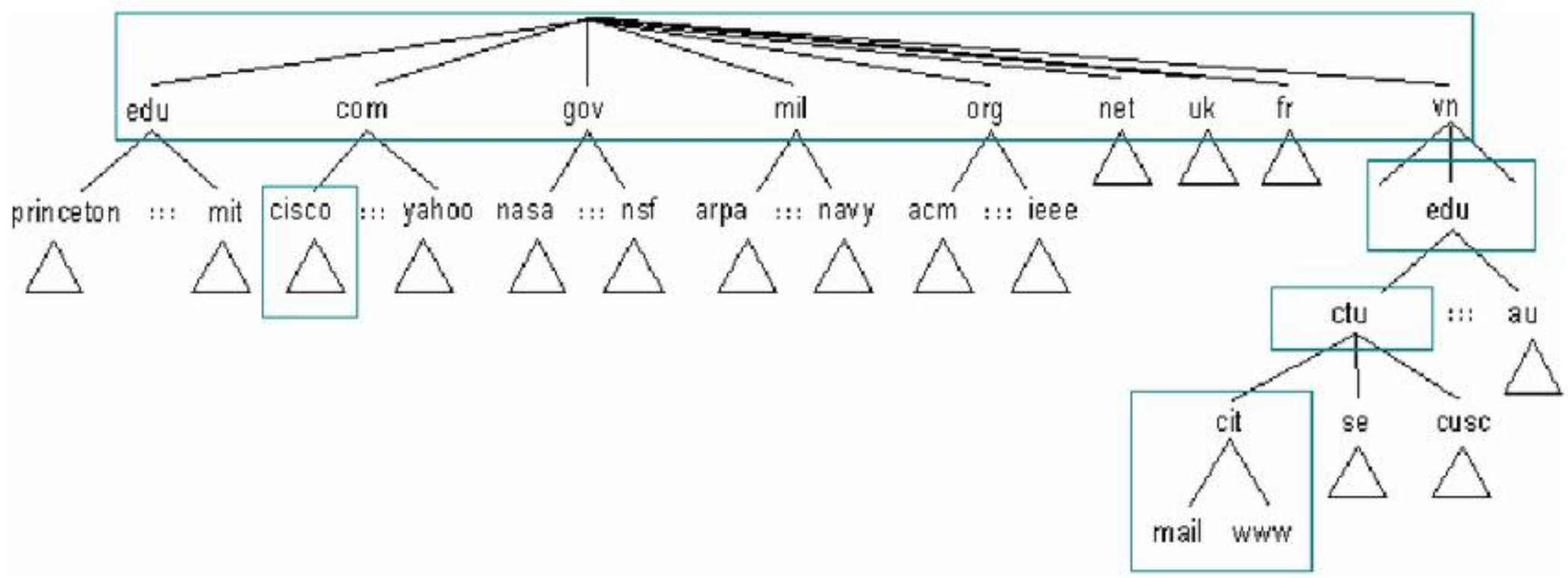


H8.2 Cây phân cấp tên miền

Có thể thấy rằng, cây phân cấp không quá rộng ở mức đầu tiên. Mỗi quốc gia có một tên miền, ngoài ra còn có 6 miền lớn khác gồm: **edu**, **com**, **gov**, **mil**, **org** và **net**. Sáu miền lớn này nằm ở Mỹ. Những tên miền không chỉ ra tên nước một cách tường minh thì mặc nhiên là nằm ở Mỹ.

8.1.2 Các server phục vụ tên

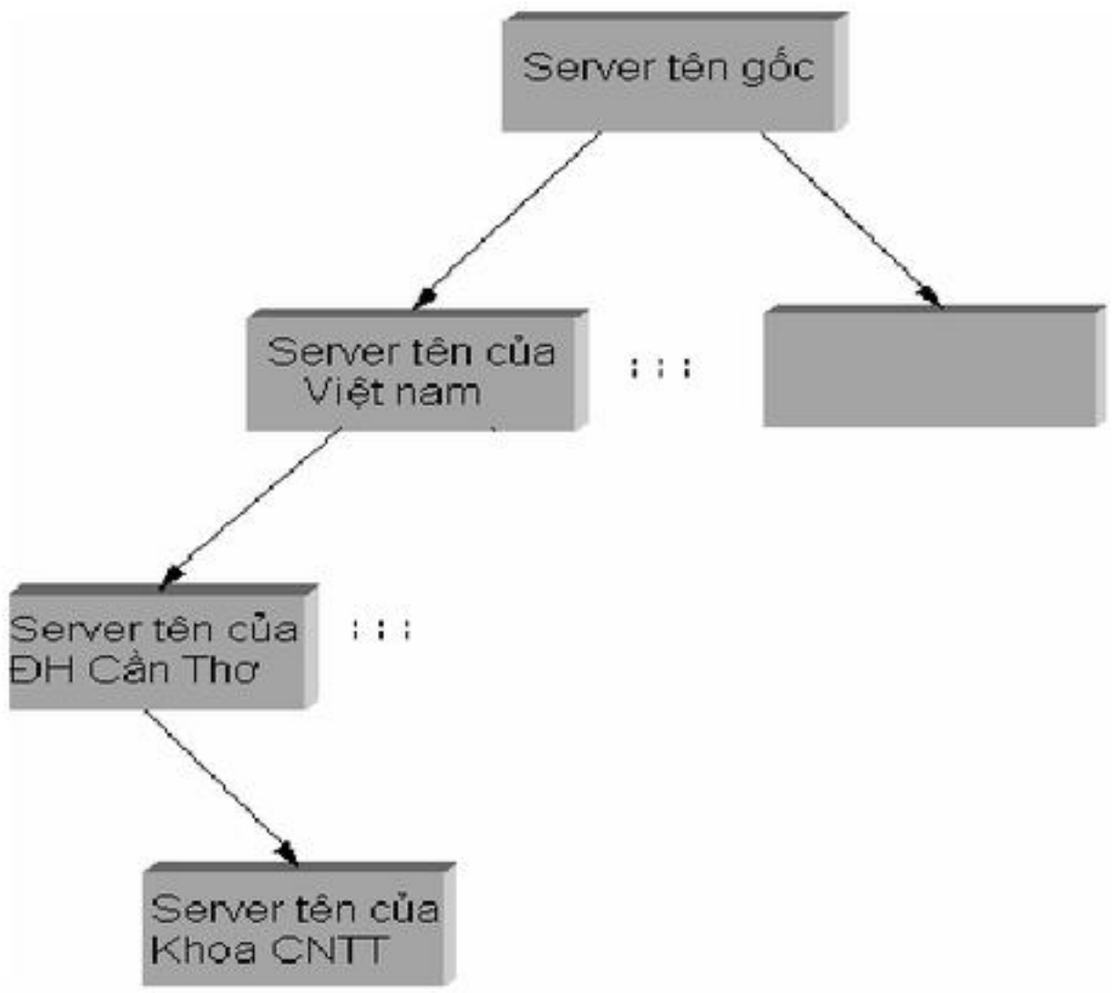
Một cấu trúc tên miền phân cấp hoàn chỉnh chỉ tồn tại trong ý niệm. Vậy thì trong thực tế cấu trúc phân cấp này được cài đặt như thế nào? Bước đầu tiên là chia cấu trúc này thành các cây con gọi là các ***vùng*** (zone). Ví dụ, hình H8.3 chỉ ra cách thức cấu trúc phân cấp trong hình H8.2 được chia thành các vùng như thế nào.



H8.3 Cấu trúc miền phân cấp được chia thành các vùng

Mỗi một vùng có thể được xem là đơn vị quản lý một bộ phận của toàn hệ thống phân cấp. Ví dụ, vùng cao nhất của hệ thống phân cấp được quản lý bởi NIC (Network Information Center), vùng **ctu** được quản lý bởi Trường Đại Học Cần Thơ.

Một vùng luôn có mối liên hệ đến các đơn vị cài đặt cơ bản trong DNS - các server tên. Thông tin chứa trong một vùng được thiết lập tại hai hoặc nhiều server tên. Mỗi server tên có thể truy xuất được qua mạng Internet. Client gởi yêu cầu đến server tên, server tên sẽ trả lời cho yêu cầu đó. Câu trả lời đôi khi chứa thông tin cuối cùng mà client cần, đôi khi lại chứa chỉ điểm đến một server tên khác mà client nên gởi câu hỏi đến đó. Vì thế, theo cách nhìn thiên về cài đặt, người ta có thể nghĩ về DNS được cài đặt bằng cấu trúc phân cấp các server tên hơn là bằng cấu trúc phân cấp các miền.



H8.4 Cấu trúc phân cấp của các server tên

Để ý rằng mỗi vùng được cài đặt trong hai hoặc nhiều server tên với lý do dự phòng; nghĩa là nếu một server bị chết sẽ còn các server khác thay thế. Mặt khác, một server tên cũng có thể được dùng để cài đặt nhiều hơn một vùng.

Mỗi server tên quản lý thông tin về một vùng dưới dạng một tập các mẫu tin tài nguyên (resource record). Mỗi mẫu tin tài nguyên là một ánh xạ từ tên sang giá trị (name to value binding), cụ thể hơn là một mẫu tin gồm 5 trường:

(Tên, Giá trị, Kiểu, Lớp, TTL)

Các trường **Tên** và **Giá trị** là những gì chúng ta muốn có, ngoài tra trường **Kiểu** chỉ ra cách thức mà **Giá trị** được thông dịch. Chẳng hạn, trường **Kiểu = A** chỉ ra rằng **Giá trị** là một địa chỉ IP. Vì thế các mẫu tin kiểu **A** sẽ cài đặt kiểu ánh xạ từ tên miền sang địa chỉ IP. Ví dụ như mẫu tin:
(ns.ctu.edu.vn, 203.162.41.166, A, IN)
chỉ ra rằng địa chỉ IP của host có tên ns.ctu.edu.vn là 203.162.41.166.
Ngoài ra còn có những kiểu khác:

- **NS:** Trường **Giá trị** chỉ ra tên miền của máy tính đang chạy dịch vụ tên, và dịch vụ đó có khả năng thông dịch các tên trong một miền cụ thể.
  Ví dụ mẫu tin:
  (ctu.edu.vn, ns.ctu.edu.vn, NS, IN)
  chỉ ra rằng server tên của miền ctu.edu.vn có tên là ns.ctu.edu.vn.
- **CNAME:** Trường **Giá trị** chỉ ra một cái tên giả của một host nào đó. Kiểu này được dùng để đặt thêm bí danh cho các host trong miền.
- **MX:** Trường **Giá trị** chỉ ra tên miền của host đang chạy chương trình mail server mà server đó có khả năng tiếp nhận những thông điệp thuộc một miền cụ thể.

Ví dụ mẫu tin
(ctu.edu.vn, mail.ctu.edu.vn, MX, IN)
chỉ ra rằng host có tên mail.ctu.edu.vn là mail server của miền ctu.edu.vn.

Trường **Lớp** được sử dụng nhằm cho phép thêm vào những thực thể mạng không do NIC quản lý. Ngày nay, lớp được sử dụng rộng rãi nhất là loại được Internet sử dụng; nó được ký hiệu là **IN**.
Cuối cùng trường **TTL** chỉ ra mẫu tin tài nguyên này sẽ hợp lệ trong bao lâu. Trường này được sử dụng bởi những server đang trữ tạm các mẫu tin của server khác; khi trường **TTL** hết hạn, các mẫu tin chứa trường **TTL** hết hạn đó sẽ bị các server xóa khỏi cache của mình.
Để hiểu rõ hơn cách thức các mẫu tin tài nguyên được thể hiện trong cấu trúc phân cấp, hãy xem ví dụ được vẽ trong hình H8.3. Để đơn giản hóa vấn đề, chúng ta bỏ qua trường **TTL** và cung cấp thông tin tương ứng cho một server tên làm nhiệm vụ quản lý cho một vùng.
Đầu tiên, server tên gốc (root name server) sẽ chứa một mẫu tin **NS** cho mỗi server cấp hai. Nó cũng chứa một mẫu tin **A** để thông dịch từ một tên server cấp hai sang địa chỉ IP của nó. Khi được ghép với nhau, hai mẫu tin này cài đặt một cách hiệu quả một con trỏ từ server gốc đến mỗi server cấp hai của nó.

(edu.vn, dns1.vnnic.net.vn, NS, IN);*thông tin về miền con edu.vn lưu ở máy dns1.vnnic.net.vn*
(dns1.vnnic.net.vn, 203.162.57.105, A, IN);*máy dns1.vnnic.net.vn có địa chỉ 203.162.57.105*
(cisco.com, ns1.cisco.com, NS, IN)

Kế tiếp, miền edu.vn có một server tên hiện hữu tại máy dns1.vnnic.net.vn và server này lại chứa các mẫu tin sau:

```
(ctu.edu.vn, ns.ctu.edu.vn, NS, IN)
(ns.ctu.edu.vn, 203.162.41.166, A, IN)
 ▪
 ▪
 ▪
```

Cuối cùng server ns.ctu.edu.vn lại chứa thông tin về các máy tính của trường Đại Học Cần Thơ cũng như các miền con của Trường Đại Học Cần Thơ

```
(cit.ctu.edu.vn, ns.cit.ctu.edu.vn, NS, IN)
(ns.cit.ctu.edu.vn, 203.162.36.144, A, IN)
(ctu.edu.vn, mail.ctu.edu.vn, MX, IN)
(mail.ctu.edu.vn, 203.162.139.21, A, IN)
(www.ctu.edu.vn, mail.ctu.edu.vn, CNAME, IN)
 ▪
 ▪
 ▪
```

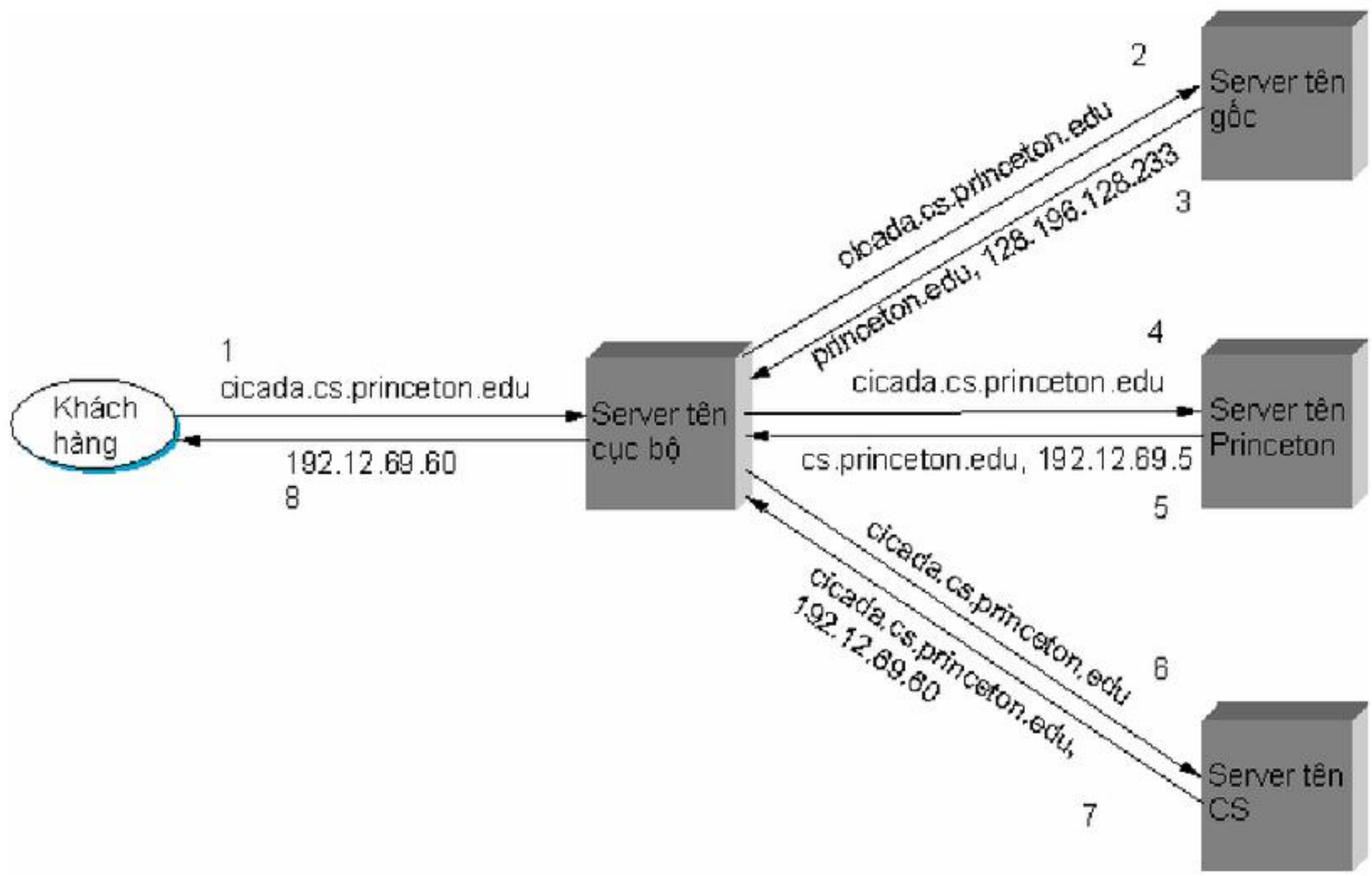
Chú ý rằng trên lý thuyết các mẫu tin có thể được dùng để định nghĩa bất kỳ kiểu đối tượng nào, DNS lại thường được sử dụng để định danh các host và site. DNS không được dùng để định danh cá nhân con người hoặc các đối tượng khác như tập tin hay thư mục, việc định danh này được thực hiện trong các hệ thống phục vụ tên khác. Ví dụ X.500 là hệ thống định danh của ISO được dùng để định danh con người bằng cách cung cấp thông tin về tên, chức vụ, số điện thoại, địa chỉ, và vân vân. X.500 đã chứng tỏ là quá phức tạp nên không được hỗ trợ bởi các search engine nổi tiếng hiện nay. Tuy nhiên nó lại là nguồn gốc phát sinh ra chuẩn LDAP (Lightweight Directory Access Protocol). LDAP vốn là thành phần con của X.500 được thiết kế để làm phần front-end cho X.500. Ngày nay LDAP đang trở nên phổ biến nhất là ở cấp độ công ty, tổ chức lớn, đóng vai trò là hệ thống học và quản lý thông tin về người dùng của nó.

8.1.3 Phương pháp phân tích tên

Với một hệ thống phân cấp các server tên đã trình bày, bây giờ chúng ta đi tìm hiểu cách thức một khách hàng giao tiếp với các server này để phân tích cho được một tên miền thành địa chỉ. Giả sử một khách hàng muốn phân tích tên miền www.ctu.edu.vn, đầu tiên khách hàng này sẽ gởi yêu cầu chứa tên này đến server tên gốc. Server gốc không thể so khớp tên theo yêu cầu với các tên mà nó chứa, liền trả lời cho khách hàng một mẫu tin kiểu NS chứa edu.vn. Server gốc cũng trả về tất cả các mẫu tin có liên quan đến mẫu tin NS vừa nói, trong đó có mẫu tin kiểu A chứa địa chỉ của dns1.vnnic.vnn.vn. Khách hàng chưa có thông tin cuối cùng mà nó muốn, tiếp tục gởi yêu cầu đến server tên tại địa chỉ 203.162.57.105. Server tên thứ hai này lại không thể so khớp tên theo yêu cầu với các tên mà nó chứa, tiếp tục trả lời cho khách hàng một mẫu tin loại NS chứa tên ctu.edu.vn cùng với mẫu tin kiểu A tương ứng với tên server là ns.ctu.edu.vn. Khách hàng lại tiếp tục gởi yêu cầu đến server tên tại địa chỉ 203.162.41.166 và lần này nhận được câu trả lời cuối cùng có kiểu A cho tên www.ctu.edu.vn.
Ví dụ trên chắc chắn sẽ để lại nhiều câu hỏi về quá trình phân giải tên. Câu hỏi thường được đặt ra là: Lúc khởi đầu, làm sao khách hàng có thể định vị được server gốc? Đây là bài toán cơ bản đặt ra cho mọi hệ thống phục vụ tên và câu trả lời là: hệ thống phải tự thân vận động để có được thông tin về các server gốc! Trong tình huống của hệ thống DNS, ánh xạ từ tên sang địa chỉ của một hay nhiều server gốc được phổ biến cho mọi người, nghĩa là ánh xạ đó được loan báo thông qua các phương tiện truyền thông khác nằm ngoài hệ thống tên.
Tuy nhiên, trong thực tế không phải tất cả khách hàng đều biết về các server gốc. Thay vào đó, chương trình khách hàng chạy trên mỗi host trong Internet được khởi động với các địa chỉ lấy từ server tên cục bộ. Ví dụ, tất cả các host trong Khoa Công Nghệ Thông Tin của Trường Đại Học Cần Thơ đều biết server tên nội bộ đang chạy trên máy ns.cit.ctu.edu.vn. Đến lượt server tên cục bộ này lại chứa các mẫu tin tài nguyên cho một hoặc nhiều server gốc của nó, ví dụ:

(. , a.root-servers.net, NS, IN)
(a.root-server.net, 198.41.0.4, A, IN)

Trong ví dụ trên, server tên cục bộ có thông tin về một server tên gốc của nó (chú ý miền gốc được ký hiệu bằng dấu chấm) là a.root-servers.net, địa chỉ IP tương ứng của server gốc này là 198.41.0.4.
Từ đó, việc phân giải một tên miền bắt đầu từ câu truy vấn của khách hàng đến server cục bộ. Nếu server cục bộ không có sẵn câu trả lời, nó sẽ gởi câu hỏi đến server từ xa dùm cho khách hàng. Chuỗi hành động trên có thể được mô tả trong hình H8.5



H8.5 Quá trình phân giải tên trong thực tế, các số 1 đến 8 chỉ ra trình tự thực hiện

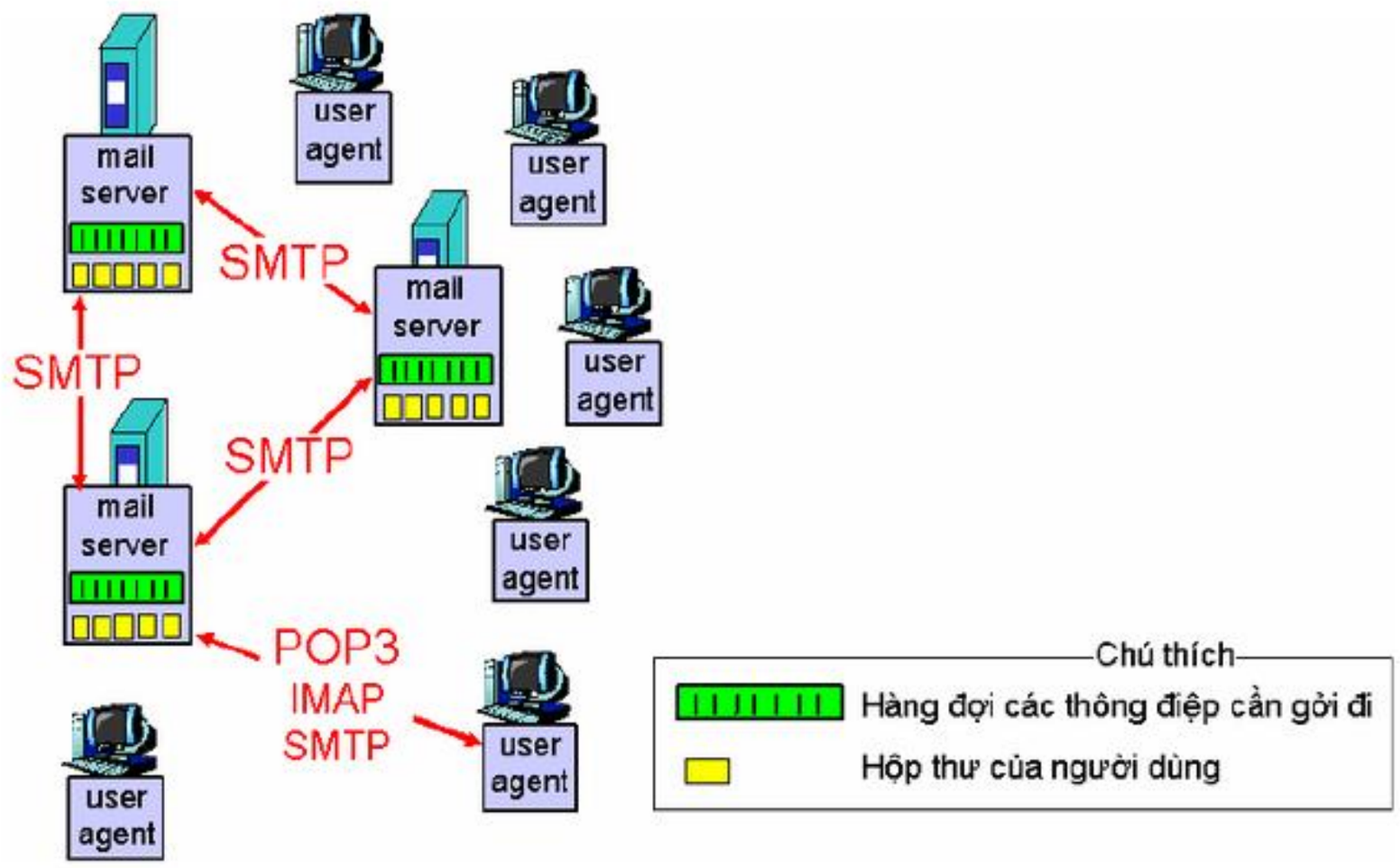
8.2 Electronic Mail (SMTP, MIME, POP3, IMAP)

Email là một trong những ứng dụng mạng lâu đời nhất nhưng lại phổ dụng nhất. Thử nghĩ khi bạn muốn gởi thông điệp đến một người bạn ở đầu kia của thế giới, bạn muốn mang thư chạy bộ qua đó hay chỉ đơn giản lên máy tính gõ ít hàng và nhấn nút **Send**? Thật ra, những bậc tiền bối của mạng ARPANET đã không tiên đoán được email sẽ là ứng dụng then chốt chạy trên mạng này, mục tiêu chính của họ là thiết kế hệ thống cho phép truy cập tài nguyên từ xa. Hệ thống email ra đời không mấy nổi bật, để bây giờ lại được sử dụng hằng ngày bởi hàng triệu người trên thế giới. Mục tiêu của phần này là chỉ ra những nhân vật hoạt động trong hệ thống email, vai trò của họ, giao thức mà họ sử dụng và khuôn dạng thông điệp mà họ trao đổi với nhau.

8.2.1 Các thành phần của hệ thống email

Một hệ thống email thường có 3 thành phần chính: Bộ phận trợ giúp người dùng (User Agent), Mail Server và các giao thức mà các thành phần này dùng để giao tiếp với nhau.
Người ta phân loại các giao thức như sau:

- Giao thức giữa các mail servers bao gồm:
  - SMTP (Simple Mail Transfer Protocol): được các server dùng để chuyển thư qua lại với nhau. Ví dụ nôm na, nó giống như cách thức mà các trạm bưu điện dùng để chuyển các thùng thư của khách hàng cho nhau. Thông tin chi tiết về giao thức này được mô tả trong tài liệu RFC 822.
- Giao thức giữa mail server và user agent bao gồm:
  - POP3 (Post Offic Protocol version 3 [RFC 1939]): được user agent sử dụng để lấy thư về từ hộp thư của nó trên server.
  - SMTP: được user agent sử dụng để gởi thư ra server.
  - IMAP: (Internet Mail Access Protocol [RFC 1730]): Có nhiều tính năng vượt trội hơn POP3. Ngoài ra IMAP còn cho phép gởi mail.



H8.6 Các thành phần của hệ thống email

8.2.2 Khuôn dạng của một email

RFC 822 định nghĩa một email gồm có hai phần: phần tiêu đề (header) và phần thân (body).

H8.7Khuôn dạng của email

Cả hai phần đều được thể hiện dưới dạng ký tự ASCII. Lúc đầu, phần thân được qui định có khuôn dạng văn bản đơn giản. Sau này người ta đề nghị một chuẩn mới gọi là MIME, có thể cho phép phần thân của email chứa bất kỳ loại dữ liệu nào.
Phần tiêu đề bao gồm nhiều dòng thông tin, mỗi dòng kết thúc bằng hai ký tự <CRLF>. Phần tiêu đề được chia khỏi phần thân bởi một hàng rỗng. Mỗi một hàng tiêu đề chứa một cặp "tên" và "giá trị", cách nhau bởi dấu hai chấm (:). Người dùng có thể rất quen với nhiều hàng tiêu đề vì họ thường phải điền thông tin vào đấy. Ví dụ

Tên	Giá trị
From:	Địa chỉ người gởi
To:	Địa chỉ của người nhận
Subject:	Chủ đề thư
Date:	Ngày gởi

RFC 822 được mở rộng năm 1993 (và được cập nhật lại năm 1996) để cho phép email mang được nhiều loại dữ liệu: audio, video, hình ảnh, tài liệu Word, … MIME (Multipurpose Internet Mail Extensions) về cơ bản có ba phần. Phần đầu tiên là tập các dòng header dùng để bổ túc cho phần header cũ của RFC 822. Theo nhiều cách, những dòng header này mô tả dữ liệu chứa trong phần thân. Cụ thể như sau:

Tên	Giá trị
MIME-Version:	Phiên bản MIME đang sử dụng
Content-Description:	Mô tả trong thư đang có dữ liệu gì
Content-Type:	Mô tả kiểu dữ liệu đang nằm trong thư
Content-Transfer-Encoding:	Mô tả cách thức mã hóa dữ liệu trong thư

Phần thứ hai là các định nghĩa cho một tập các kiểu nội dung (và kiểu con nếu có). Ví dụ một số kiểu mà MIME định nghĩa:

Kiểu	Ý nghĩa
image/gif	Ảnh dạng gif
image/jpeg	Ảnh dạng jpeg
text/plain	Văn bản đơn giản
text/richtext	Văn bản mở rộng (có đặt font chữ, được định dạng đậm, nghiêng hoặc gạch dưới …)
application	Dữ liệu trong thư được xuất ra từ một ứng dụng nào đó. Chẳng hạn: *application/postscript*: tài liệu Postscript (.ps) *application/msword*: tài liệu Microsoft Word (.doc)

MIME cũng định nghĩa kiểu **multipart** để chỉ ra cách mà phần thân của thư mang nhiều loại dữ liệu khác nhau như thế nào. Chỉ có một kiểu con của **multipart** là **mixed** với ý nói rằng trong phần thân của thư có nhiều mảnh dữ liệu khác nhau, độc lập với nhau và được sắp xếp theo một trình tự cụ thể. Mỗi mảnh dữ liệu sẽ có phần tiêu đề riêng để mô tả kiểu dữ liệu của mảnh đó. Phần thứ ba mô tả cách thức mã hóa các kiểu dữ liệu nói trên để có thể truyền chúng dưới dạng ASCII. Lý do để mọi bức thư phải chứa các ký tự ASCII là vì để đi được đến đích, bức thư đó có thể phải trung chuyển qua nhiều gateway, mà các gateway này đều coi mọi bức thư dưới dạng ASCII. Nếu trong thư chứa bất kỳ ký tự nào khác ASCII thì thư sẽ bị đứt gãy nội dung. MIME sử dụng phương pháp mã hóa trực tiếp dữ liệu nhị phân thành các ký tự nhị phân, gọi là **base64**. Ý tưởng của **base64** là ánh xạ 3 bytes dữ liệu nhị phân nguyên thủy thành 4 ký tự ASCII. Giải thuật đơn giản như sau: tập hợp 3 bytes dữ liệu nhị phân lại thành 24 bits, sau đó chia 24 bits này thành 4 cụm, một cụm 6 bits. Một cụm 6 bits được ánh xạ vào một trong 64 ký tự ASCII hợp lệ; ví dụ 0 ánh xạ thành A, 1 ánh xạ thành B… Nếu nhìn vào bức thư đã được mã hóa dạng **base64**, người dùng sẽ thấy chỉ có 52 chữ cái cả hoa lẫn thường, 10 chữ số từ 0 đến 9 và các ký tự đặc biệt + và /. Đối với những người dùng chỉ sử dụng trình đọc thư hỗ trợ duy nhất kiểu ký tự thì việc đọc những bức thư có kiểu **base64** sẽ rất là đau khổ. Vì lý do nhân đạo, MIME còn hỗ trợ kiểu mã hóa ký tự thường được gọi là **7-bit**. **7-bit** sẽ giữ nguyên dạng ký tự mà người ta nhập vào.
Tổng hợp lại, ví dụ một bức thư có 2 loại dữ liệu: văn bản thường, một ảnh JPEG, sẽ có hình dáng như sau:

```
From: ptphi@cit.ctu.edu.vn
To: TH27@cit.ctu.edu.vn
Subject: Picture of students.
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="-98766789"

--98766789
Content-Transfer-Encoding: 7bit
Content-Type: text/plain

Hi,
Please find a picture of you.
--98766789
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.........................
......base64 encoded data
--98766789--
```

8.2.3 Chuyển thư

Kế đến, chúng ta sẽ xem xét giao thức SMTP – giao thức được dùng để chuyển thư từ máy này đến máy kia. Để đặt SMTP vào đúng ngữ cảnh, chúng ta nên nhắc lại các nhân vật then chốt trong hệ thống email. Đầu tiên, người dùng tương tác với trình đọc thư (hay còn gọi là user agent) để soạn, lưu, tìm kiếm và đọc thư của họ. Hiện trên thị trường có nhiều phần mềm đọc thư, cũng giống như hiện cũng đang có nhiều loại trình duyệt Web vậy. Thứ hai, có trình xử lý thư (hay còn gọi là mail server) chạy trên một máy nào đó trong mạng nội bộ của người dùng. Có thể xem mail server như một bưu điện: Người dùng trao cho mail server các bức thư mà họ muốn gởi cho người dùng khác, mail server sử dụng giao thức SMTP trên TCP để chuyển bức các thư này đến mail server bên đích. Mail server bên đích nhận các thư đến và đặt chúng vào hộp thư của người dùng bên đích. Do SMTP là giao thức mà rất nhiều người có thể tự cài đặt, vì thế sẽ có rất nhiều sản phẩm mail server hiện có trên thị trường. Sản phẩm mail server thường được sử dụng nhất là **sendmail**, ban đầu được cài đặt trong hệ điều hành Berkeley Unix.

Tất nhiên mail server bên máy gởi có thể kết nối SMTP/TCP trực tiếp tới mail server bên máy nhận, nhưng trong thực tế, một bức thư có thể đi ngang qua vài mail gateways trước khi đến đích. Cũng giống như máy đích, mỗi mail gateway cũng chạy một mail server. Không phải ngẫu nhiên mà các nút chuyển thư trung gian được gọi là mail gateway. Công việc của chúng cũng giống như các IP gateway là lưu tạm và chuyển phát tiếp các bức thư của người dùng. Điểm khác nhau duy nhất giữa chúng là, mail gateway trữ tạm các bức thư trong đĩa, trong khi các IP gateway trữ tạm các gói tin IP trong bộ nhớ.

Bạn có thể đặt câu hỏi: tại sao lại cần đến các mail gateways? Tại sao không dùng phương pháp nối kết SMTP/TCP trực tiếp từ bên gởi sang bên nhận? Lý do thứ nhất, người gởi không muốn kèm trong thư địa chỉ của máy đích. Ví dụ, riêng việc nhập vào trong thư địa chỉ đích **ptphi@cit.ctu.edu.vn** đã mất công rồi, không ai thấy thoải mái khi phải nhập thêm địa chỉ máy đích là **machine-of-phi.cit.ctu.edu.vn**. Thứ hai, không chắc lúc bên gởi thiết lập nối kết đến bên nhận, người dùng bên nhận đã bật sẵn máy! Thành thử chỉ cần địa chỉ thư bên nhận là đủ. Khi bức thư đến được mail gateway của Khoa Công Nghệ Thông Tin – Đại học Cần Thơ, nếu người dùng **ptphi** đang mở máy, mail gateway sẽ chuyển thư cho anh ta ngay, nếu không mail gateway sẽ trữ tạm thư trên đĩa của nó đến khi **ptphi** bật máy lên và kiểm tra thư.

Dù có bao nhiêu mail gateways trung gian trên đường đến đích vẫn không đáng lo lắng, bởi vì mỗi mail gateway trung gian sẽ nỗ lực sử dụng một kết nối SMTP độc lập đến gateway kế tiếp trên đường đi nhằm chuyển thư càng ngày càng đến gần người nhận.
SMTP là một giao thức đơn giản dùng các ký tự ASCII. Sau khi thiết lập nối kết TCP đến cổng 25 của máy đích (được coi là server), máy nguồn (được coi là client) chờ nhận kết quả trả về từ server. Server khởi đầu cuộc đối thoại bằng cách gởi một dòng văn bản đến client thông báo danh tính của nó và khả năng tiếp nhận thư. Nếu server không có khả năng nhận thư tại thời điểm hiện tại, client sẽ hủy bỏ nối kết và thử thiết lập lại nối kết sau.
Nếu server sẵn sàng nhận thư, client sẽ thông báo lá thư đó từ đâu đến và ai sẽ là người nhận. Nếu người nhận đó tồn tại, server sẽ thông báo cho client tiếp tục gởi thư. Sau đó client gởi thư và server báo nhận cho thư đó. Sau khi cả hai bên hoàn tất phiên truyền nhận, kết nối sẽ được đóng lại.
Ví dụ một phiên truyền nhận được cho ngay dưới đây. Những dòng bắt đầu bằng **C:** là của phía client gởi đi; bằng **S:** là các câu trả lời của server.

```
S: 220 ctu.edu.vn
C: HELO cit.ctu.edu.vn
S: 250  ctu.edu.vn says hello to cit.ctu.edu.vn
C: MAIL FROM: <ptphi@cit.ctu.edu.vn>
S: 250 Sender ok
C: RCPT TO: <lhly@yale.edu>
S: 250 Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Subject: It's Xmast!
C: So I hope you a merry Xmas and a happy new year!
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 Bye-Bye
```

a

LỆNH CỦA CLIENT	
Lệnh	**Ý nghĩa**
HELO	Câu chào và xưng danh của client
MAIL FROM	Địa chỉ email của người gởi
RCPT TO	Địa chỉ email của người nhận
DATA	Bắt đầu truyền nội dung của thư
QUIT	Hủy nối kết
TRẢ LỜI CỦA SERVER	
Trả lời	**Ý nghĩa**
250	Yêu cầu hợp lệ
550	Yêu cầu không hợp lệ, không tồn tại hộp thư như client đã chỉ ra.
354	Cho phép bắt đầu nhập thư vào. Kết thúc thư bằng <CRLF>.<CRLF>
221	Server đang đóng kết nối TCP

Vẫn còn nhiều lệnh và mã trả lời chưa được trình bày, xin tham khảo tài liệu RFC 822 để có được đầy đủ thông tin.

8.2.4 Phân phát thư

Như đã trình bày, khi đứng về góc độ người dùng thư, họ sẽ dùng user agent để gởi và nhận thư cho họ. User agent dùng giao thức SMTP để gởi thư đi, dùng giao thức POP3 hoặc IMAP để nhận thư về.

8.2.4.1 POP3

Một phiên làm việc theo giao thức POP3 bắt đầu tại user agent. User agent khởi động một nối kết TCP đến cổng 110 của mail server. Khi kết nối thực hiện xong, phiên làm việc POP3 sẽ trải qua theo thứ tự ba kỳ:

1. Chứng thực.
2. Giao dịch dữ liệu.
3. Cập nhật.

Kỳ chứng thực buộc người dùng thực hiện thủ tục đăng nhập bằng cách nhập vào hai lệnh sau:

Lệnh	Ý nghĩa
USER <*tên người dùng*>	Khai báo tên người dùng.
PASS <*mật khẩu*>	Khai báo mật khẩu.

Báo trả của mail server sẽ là một trong hai câu sau:

Trả lời	Ý nghĩa
+OK <*chú thích*>	Khai báo của người dùng là đúng.
+ERR <*chú thích*>	Khai báo của người dùng là sai và lời giải thích.

Trong kỳ giao dịch, người dùng có thể xem danh sách thư chưa nhận về, nhận thư về và xóa thư trong hộp thư của mình khi cần thiết. Các lệnh mà người dùng thường sử dụng để giao dịch với server là:

Lệnh	Ý nghĩa
LIST [<*số thứ tự thư*>]	Nếu dùng LIST không tham số, server sẽ trả về toàn bộ danh sách các thư chưa nhận. Nếu có tham số là số thứ tự thư cụ thể, server sẽ trả về thông tin của chỉ bức thư đó thôi.
RETR <*số thứ tự thư*>	Tải lá thư có số thứ tự <*số thứ tự thư*> về.
DELE <*số thứ tự thư*>	Xóa lá thứ số <*số thứ tự thư*> khỏi hộp thư.
QUIT	Hoàn tất giai đoạn giao dịch và hủy nối kết TCP

Các trả lời của server có thể là các số liệu mà client yêu cầu hoặc các thông báo +OK, -ERR như trong phần đăng nhập.

Sau đây là dàn cảnh một phiên làm việc ví dụ giữa người dùng **ptphi** khi anh ta đăng nhập và làm việc trên hộp thư của mình tại server có địa chỉ **mail.cit.ctu.edu.vn**.

Client	Server	Giải thích
	+OK POP3 server ready	Server sẵn sàng phục vụ client
USER ptphi		
	+OK	Server xác nhận người dùng hợp lệ
PASS godblessus		
	+OK login successfully	Chứng thực thành công
LIST		ptphi kiểm tra hộp thư

	+OK 1 1024 2 2550	Hộp thư của ptphi còn hai thư chưa nhận về, thư thứ nhất có kích thước 1024 bytes, thư thứ hai có kích thước 2550 bytes
RETR 1		ptphi tải thư thứ nhất về
	+OK	server gởi thư thứ 1 cho ptphi
DELE 1		ptphi xóa thư thứ nhất trong hộp thư
	+OK	server xoá thư thứ 1 thành công
QUIT		ptphi hủy nối kết
	+OK Bye-Bye	server hủy nối kết

8.2.4.2 IMAP

Với những người dùng có một tài khoản email trên một ISP và người dùng này thường truy cập email trên một PC thì giao thức POP3 hoạt động tốt. Tuy nhiên, một sự thật trong ngành công nghệ máy tính, khi một thứ gì đó đã hoạt động tốt, người ta lập tức đòi hỏi thêm nhiều tính năng mới (và tự chuốc lấy nhiều phiền nhiễu). Điều đó cũng xảy ra đối với hệ thống email. Ví dụ, người ta chỉ có một tài khoản email, nhưng họ lại muốn ngồi đâu cũng truy cập được nó. POP3 cũng làm được chuyện này bằng cách đơn giản tải hết các email xuống máy PC mà người dùng này đang ngồi làm việc. Và dĩ nhiên là thư từ của người dùng này nằm rải rác khắp nơi.
Sự bất tiện này khơi mào cho sự ra đời của giao thức phân phối thư mới, IMAP (Internet Message Access Protocol), được định nghĩa trong RFC 2060. Không giống như POP2, IMAP coi các thông điệp mặc nhiên nằm trên server vô hạn và trên nhiều hộp thư. IMAP còn đưa ra cơ chế cho phép đọc các thông điệp hoặc một phần của thông điệp, một tính năng hữu ích khi người dùng kết nối đến server bằng đường truyền tốc độ chậm như điện thoại nhưng lại đọc các email có âm thanh, hình ảnh… Với quan niệm cho rằng người dùng không cần tải thư về lưu trên PC, IMAP cung cấp các cơ chế cho phép tạo, xóa và sửa đổi nhiều hộp thư trên server.
Cung cách làm việc của IMAP cũng giống như POP3, ngoài trừ trong IMAP có rất nhiều lệnh. IMAP server sẽ lắng nghe trên cổng 143. Cũng nên chú ý rằng, không phải mọi ISP đều hỗ trợ cả hai giao thức POP3 và IMAP.
Bảng sau so sánh các tính năng của POP3 và IMAP

Tính năng	**POP3**	**IMAP**
Giao thức được định nghĩa ở đâu?	RFC 1939	RFC 2060
Cổng TCP được dùng	110	143
Email được lưu ở đâu	PC của người dùng	Server
Email được đọc ở đâu	Off-line	On-line
Thời gian nối kết	Ít	Nhiều
Sử dụng tài nguyên của server	Tối thiểu	Nhiều hơn
Nhiều hộp thư	Không	Đúng
Ai lưu phòng hờ các hộp thư	Người dùng	ISP
Tốt cho người dùng di động	Không	Có
Kiểm soát của người dùng đối với việc tải thư về	Ít	Tốt
Tải một phần thư	Không	Có
Quota đĩa có là vấn đề không?	Không	Thỉnh thoảng
Dễ cài đặt	Có	Không
Được hỗ trợ rộng rãi	Có	Đang phát triển

8.3 World Wide Web (HTTP)

Ứng dụng Web đã rất thành công, giúp cho nhiều người có thể truy cập Internet đến nỗi Web được hiểu đồng nghĩa với Internet! Có thể hiểu Web như là một tập các client và server hợp tác với nhau và cùng nói chung một ngôn ngữ: HTTP (Hyper Text Transfer Protocol). Đa phần người dùng tiếp xúc với Web thông qua chương trình client có giao diện đồ họa, hay còn gọi là trình duyệt Web (Web browser). Các trình duyệt Web thường được sử dụng nhất là Netscape Navigator (của Netscape) và Internet Explorer (của Microsoft). Hình H8.8 thể hiện trình duyệt Explorer đang trình bày trang chủ của Khoa Công Nghệ Thông Tin – Đại Học Cần Thơ:

H8.8 Trình duyệt Web Internet Explorer

Bất kỳ trình duyệt Web nào cũng có chức năng cho phép người dùng "mở một URL". Các URL (Uniform Resource Locators) cung cấp thông tin về vị trí của các đối tượng trên Internet; chúng thường trông giống như sau:

http://www.cit.ctu.edu.vn/index.html

Nếu người dùng mở URL trên, trình duyệt Web sẽ thiết lập một kết nối TCP đến Web Server tại địa chỉ **www.cit.ctu.edu.vn** và ngay lập tức tải tập tin **index.html** về và thể hiện nó. Hầu hết các tập tin trên Web chứa văn bản và hình ảnh, một số còn chứa audio và video clips. Chúng còn có thể chứa các liên kết đến các tập tin khác – được gọi là các liên kết siêu văn bản (hypertext links). Khi người dùng yêu cầu trình duyệt Web mở ra một liên kết siêu văn bản (bằng cách trỏ chuột và

click lên liên kết đó), trình duyệt sẽ mở một nối kết mới, tải về và hiển thị một tập tin mới. Vì thế, rất dễ để duyệt từ server này đến server khác trên khắp thế giới để có được hết những thông tin mà người dùng cần.
Khi người dùng chọn xem một trang Web, trình duyệt Web sẽ nạp trang Web đó từ Web server về sử dụng giao thức HTTP chạy trên TCP. Giống như SMTP, HTTP là giao thức hướng ký tự. Về cốt lõi, một thông điệp HTTP có khuôn dạng tổng quát sau:

START_LINE <CRLF>
MESSAGE_HEADER <CRLF>
<CRLF>
MESSAGE_BODY <CRLF>

Hàng đầu tiên chỉ ra đây là thông điệp yêu cầu hay trả lời. Nó sẽ chỉ ra "thủ tục cần được thực hiện từ xa" (trong tình huống là thông điệp yêu cầu) hoặc là "trạng thái trả về" (trong tình huống là thông điệp trả lời). Tập hợp các hàng kế tiếp chỉ ra các tùy chọn hoặc tham số nhằm xác định cụ thể tính chất của yêu cầu hoặc trả lời. Phần MESSAGE_HEADER có thể không có hoặc có một vài hàng tham số và được kết thúc bằng một hàng trống. HTTP định nghĩa nhiều kiểu header, một số liên quan đến các thông điệp yêu cầu, một số liên quan đến các thông điệp trả lời và một số lại liên quan đến phần dữ liệu trong thông điệp. Ở đây chỉ giới thiệu một số kiểu thường dùng. Cuối cùng, sau hàng trống là phần nội dung của thông điệp trả lời (MESSAGE_BODY), phần này thường là rỗng trong thông điệp yêu cầu.

8.3.1 Các thông điệp yêu cầu

Hàng đầu tiên của một thông điệp yêu cầu HTTP sẽ chỉ ra 3 thứ: thao tác cần được thực thi, trang Web mà thao tác đó sẽ áp lên và phiên bản HTTP được sử dụng. Bảng sau sẽ giới thiệu một số thao tác phổ biến.

Hành động	Mô tả
OPTIONS	Yêu cầu thông tin về các tùy chọn hiện có.
GET	Lấy về tài liệu được xác định trong URL
HEAD	Lấy về thông tin thô về tài liệu được xác định trong URL
POST	Cung cấp thông tin cho server
PUT	Tải tài liệu lên server và đặt ở vị trí được xác định trong URL
DELETE	Xóa tài liệu nằm ở vị trí URL trên server
TRACE	Phản hồi lại thông điệp yêu cầu
CONNECT	Được sử dụng bởi các proxy

Hai thao tác thường được sử dụng nhiều nhất là GET (lấy một trang Web về) và HEAD (lấy về thông tin của một trang Web). GET thường được sử dụng khi trình duyệt muốn tải một trang Web về và hiển thị nó cho người dùng. HEAD thường được sử dụng để kiểm tra tính hợp lệ của một liên kết siêu văn bản hoặc để xem một trang nào đó có bị thay đổi gì không kể từ lần tải về trước đó.
Ví dụ, dòng START_LINE

GET http://www.cit.ctu.edu.vn/index.html HTTP/1.1

nói rằng: người dùng muốn tải về trên server www.cit.ctu.edu.vn trang Web có tên index.html và hiển thị nó. Ví dụ trên dùng URL tuyệt đối. Ta cũng có thể sử dụng URL tương đối như sau:

GET /index.html HTTP/1.1
Host: www.cit.ctu.edu.vn

Ở đây, **Host** là một trong các trường trong MESSAGE_HEADER.

8.3.2 Các thông điệp trả lời

Giống như các thông điệp yêu cầu, các thông điệp trả lời bắt đầu bằng một hàng START_LINE. Trong trường hợp này, dòng START_LINE sẽ chỉ ra phiên bản HTTP đang được sử dụng, một mã 3 ký số xác định yêu cầu là thành công hay thất bại và một chuỗi ký tự chỉ ra lý do của câu trả lời này.
Ví dụ, dòng START_LINE
HTTP/1.1 202 Accepted
chỉ ra server đã có thể thõa mãn yêu cầu của người dùng.
Còn dòng
HTTP/1.1 404 Not Found
chỉ ra rằng server đã không thể tìm thấy tài liệu như được yêu cầu.
Có năm loại mã trả lời tổng quát với ký số đầu tiên xác định loại mã.

Mã	Loại	Lý do
1xx	Thông tin	Đã nhận được yêu cầu, đang tiếp tục xử lý
2xx	Thành công	Thao tác đã được tiếp nhận, hiểu được và chấp nhận được
3xx	Chuyển hướng	Cần thực hiện thêm thao tác để hoàn tất yêu cầu được đặt ra
4xx	Lỗi client	Yêu cầu có cú pháp sai hoặc không thể được đáp ứng
5xx	Lỗi server	Server thất bại trong việc đáp ứng một yêu cầu hợp lệ

Cũng giống như các thông điệp yêu cầu, các thông điệp trả lời có thể chứa một hoặc nhiều dòng trong phần MESSAGE_HEADER. Những dòng này cung cấp thêm thông tin cho client. Ví dụ, dòng header **Location** chỉ ra rằng URL được yêu cầu đang có ở vị trí khác. Vì thế, nếu trang Web của Khoa Công Nghệ Thông Tin được di chuyển từ địa chỉ **http://www.cit.ctu.edu.vn/index.html** sang địa chỉ **http://www.ctu.edu.vn/cit/index.html** mà người dùng lại truy cập vào URL cũ, thì Web server sẽ trả lời như sau

HTTP/1.1 301 Moved Permanently
Location: http://www.ctu.edu.vn/cit/index.html

Trong tình huống chung nhất, thông điệp trả lời cũng sẽ mang theo nội dung trang Web được yêu cầu. Trang này là một tài liệu HTML, nhưng vì nó có thể chứa dữ liệu không phải dạng văn bản (ví dụ như ảnh GIF), dữ liệu này có thể được mã hóa theo dạng MIME. Một số hàng trong phần MESSAGE_HEADER cung cấp thêm thông tin về nội dung của trang Web, bao gồm **Content-Length** (số bytes trong phần nội dung), **Expires** (thời điểm mà nội dung trang Web được xem như lỗi thời), và **Last-Modified** (thời điểm được sửa đổi lần cuối cùng).

8.3.3 Các kết nối TCP

Nguyên tắc chung của giao thức HTTP là client nối kết đến cổng TCP số 80 tại server, server luôn lắng nghe trên cổng này để sẵn sàng phục vụ client. Phiên bản đầu tiên (HTTP/1.0) sẽ thiết lập một nối kết riêng cho mỗi hạng mục dữ liệu cần tải về từ server. Không khó để thấy rằng đây là cơ chế không mấy hiệu quả: Các thông điệp dùng để thiết lập và giải phóng nối kết sẽ phải được trao đổi qua lại giữa client và server và khi mà tất cả client muốn lấy thông tin mới nhất của một trang Web, server sẽ bị quá tải.
Cải tiến quan trọng nhất trong phiên bản HTTP/1.1 là nó cho phép các kết nối lâu dài – client và server sẽ trao đổi nhiều thông điệp yêu cầu/trả lời trên cùng một kết nối TCP. Kết nối lâu dài có hai cái lợi. Thứ nhất, nó làm giảm thiểu chi phí cho việc thiết lập/giải phóng nối kết. Thứ hai, do client gởi nhiều thông điệp yêu cầu qua một kết nối TCP, cơ chế điều khiển tắc nghẽn của TCP sẽ hoạt động hiệu quả hơn.
Tuy nhiên, kết nối lâu dài cũng có cái giá phải trả. Vấn đề phát sinh ở chỗ: không ai trong client và server biết được kết nối đó sẽ kéo dài bao lâu. Điều này thực sự gây khó khăn cho phía server bởi vì tại mỗi thời điểm, nó phải đảm bảo duy trì kết nối đến cả ngàn client. Giải pháp cho vấn đề này là: server sẽ mãn kỳ và cắt nối kết nếu nó không nhận được một yêu cầu cụ thể nào từ phía

client trong một khoảng thời gian định trước. Ngoài ra, cả client và server phải theo dõi xem phía bên kia có chủ động cắt nối kết hay không và lấy đó làm cơ sở để tự cắt nối kết của mình. (Nhắc lại rằng, cả hai bên phải cắt nối kết thì nối kết TCP mới thực sự kết thúc).

8.3.4 Trữ đệm

Một trong những lĩnh vực nghiên cứu tích cực nhất hiện nay về Internet là làm sao để trữ tạm các trang Web một cách hiệu quả. Việc trữ tạm mang lại nhiều lợi ích. Từ phía client, việc nạp và hiển thị một trang Web từ bộ đệm gần đấy là nhanh hơn rất nhiều so với từ một server nào đó ở nửa vòng trái đất. Đối với server, có thêm một bộ đệm để can thiệp vào và phục vụ giúp yêu cầu của người dùng sẽ giảm bớt tải trên server.
Việc trữ đệm có thể được cài đặt tại nhiều nơi khác khau. Ví dụ, trình duyệt Web có thể trữ tạm những trang Web mới được nạp về gần đây, để khi người dùng duyệt lại những trang Web đó, trình duyệt sẽ không phải nối kết ra Internet để lấy chúng về mà dùng bản trữ sẵn. Ví dụ khác, một khu vực làm việc (site) có thể đề cử một máy làm nhiệm vụ trữ tạm các trang Web, để những người dùng sau có thể sử dụng các bản trữ sẵn của những người dùng trước. Yêu cầu của hệ thống này là mọi người dùng trong site phải biết địa chỉ của máy tính làm nhiệm vụ bộ trữ tạm, và họ chỉ đơn giản là liên hệ với máy tính này để tải các trang Web về theo yêu cầu. Người ta thường gọi máy tính làm nhiệm vụ trữ tạm các trang Web cho một site là ***proxy***. Vị trí trữ đệm có thể di chuyển gần hơn đến phần lõi của Internet là các ISP. Trong tình huống này, các site nối kết tới ISP thường không hay biết gì về việc trữ tạm ở đây. Khi các yêu cầu HTTP từ các site được chuyển phát đến router của ISP, router liền kiểm tra xem URL được yêu cầu có giống với các URL được trữ sẵn hay không. Nếu có, router sẽ trả lời ngay. Nếu không, router sẽ chuyển yêu cầu đến server thật sự và cũng không quên lưu vào bộ đệm của mình thông điệp trả lời từ phía server đó.
Việc trữ tạm là đơn giản. Tuy nhiên bộ đệm phải đảm bảo những thông tin trữ đệm trong đó không quá cũ. Để làm được việc này, các Web server phải gán "ngày hết hạn" (tức là trường **Expires** trong header) cho mọi trang Web mà nó phục vụ cho client. Nhân đó, các bộ đệm cũng lưu lại thông tin này. Và từ đó, các bộ đệm sẽ không cần phải kiểm tra tính cập nhật của trang Web đó cho đến khi ngày hết hạn đến. Tại thời điểm một trang Web hết hạn, bộ đệm sẽ dùng lệnh HEAD hoặc lệnh GET có điều kiện (GET với trường **If-Modified-Since** trong phần header được đặt) để kiểm tra rằng nó có một phiên bản mới nhất của trang Web kia. Tổng quát hơn, cần phải có "các chỉ thị hướng dẫn" cho việc trữ đệm và các chỉ thị này phải được tuân thủ tại mọi bộ đệm. Các chỉ thị sẽ chỉ ra có nên trữ đệm một tài liệu hay không, trữ nó bao lâu, một tài liệu phải tươi như thế nào và vân vân.

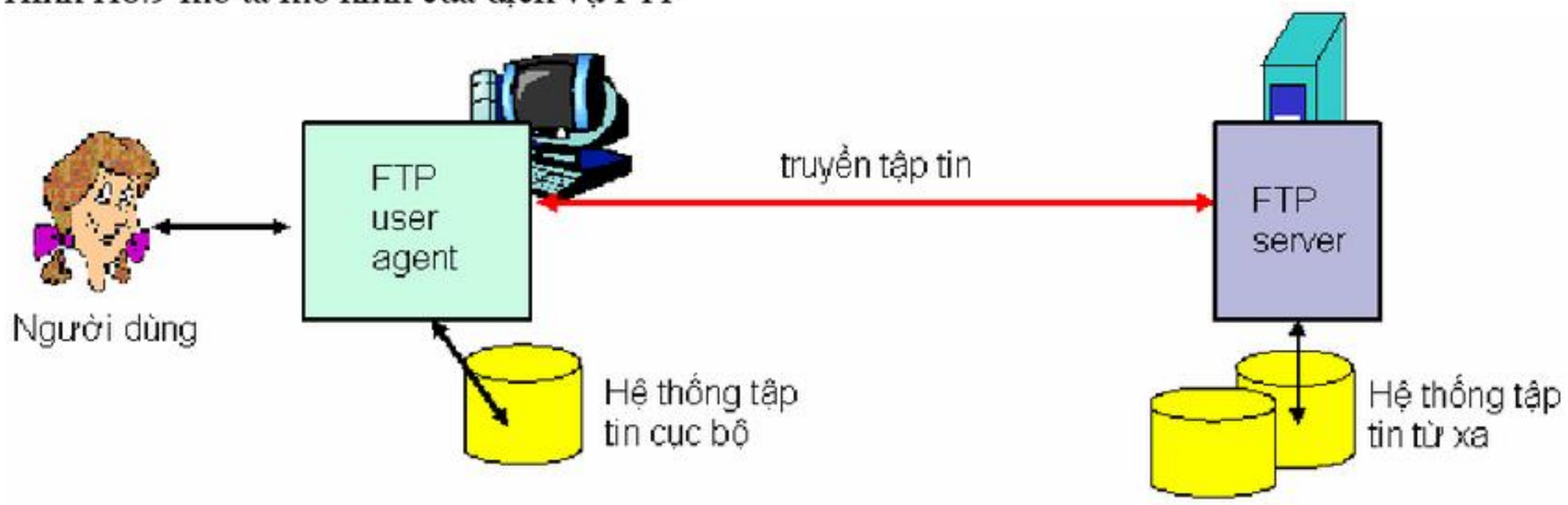
8.4 Truyền tập tin (FTP)

Thông qua dịch vụ FTP, người dùng tại một máy tính có thể đăng nhập và thao tác lên hệ thống tập tin được chia sẻ của một máy tính từ xa.
Mục tiêu của dịch vụ FTP là:

1) Đảm bảo việc chia sẻ tập tin (chương trình máy tính hoặc dữ liệu) trên mạng.
2) Khuyến khích việc sử dụng không trực tiếp (thông qua chương trình) tài nguyên trên các máy tính khác.
3) Người dùng không cần phải quan tâm đến sự khác nhau của các hệ thống tập tin trên mạng.
4) Truyền dữ liệu một cách tin cậy và hiệu quả.

8.4.1 Mô hình dịch vụ FTP

Hình H8.9 mô tả mô hình của dịch vụ FTP

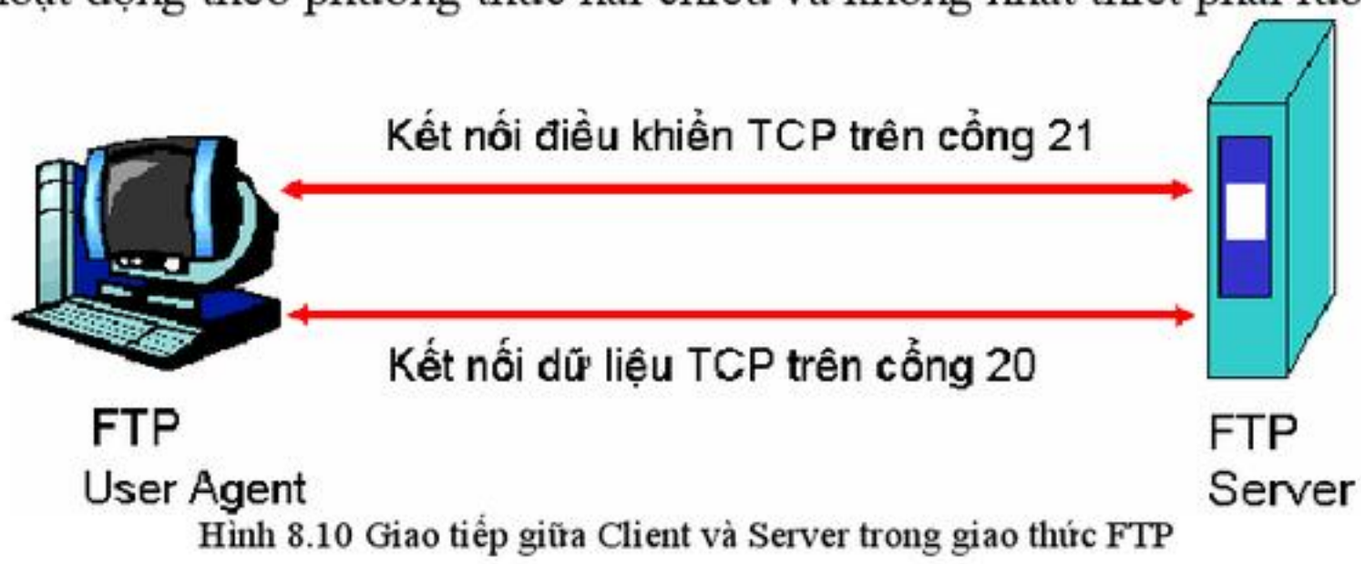


H8.9 Mô hình dịch vụ FTP

Trong hệ thống này, người dùng sẽ ra lệnh cho FTP user agent. User agent sẽ nối kết tới FTP server để dàn xếp thủ tục làm việc, thực thi các tác vụ theo yêu cầu và trả kết quả về cho người dùng.

8.4.2 Giao thức FTP

Đầu tiên, user agent thiết lập một kết nối điều khiển trên cổng 21 tới FTP server. Sau khi đã thỏa thuận các tham số truyền nhận, hai bên sẽ thiết lập một kênh dữ liệu chạy trên cổng 20. Dữ liệu của các tập tin được trao đổi qua lại giữa user agent và server sẽ chạy trên kênh dữ liệu này. Kênh dữ liệu là kênh hoạt động theo phương thức hai chiều và không nhất thiết phải luôn tồn tại.



Hình 8.10 Giao tiếp giữa Client và Server trong giao thức FTP

8.4.3 Các lệnh cơ bản

Sau đây là các lệnh cơ bản mà người dùng có thể sử dụng để thao tác lên hệ thống FTP

Lệnh	Tham số	Ý nghĩa
FTP	host-name	Nối kết đến FTP server có địa chỉ host-name
USER	user-name	Cung cấp tên người dùng cho FTP server để thực hiện quá trình chứng thực
ASCII		Chỉ định kiểu dữ liệu truyền nhận là ký tự
BINARY		Chỉ định kiểu dữ liệu truyền nhận là nhị phân
LS		Xem nội dung thư mục từ xa
CD	remote-dir	Chuyển đến thư mục khác trong hệ thống tập tin từ xa
GET	remote-file local-file	Tải tập tin remote-file trên FTP server về hệ thống tập tin cục bộ và đặt tên là local-file
PUT	local-file remote-file	Nạp tập tin cục bộ local-file lên server và đặt tên là remote-file
MKDIR	dir-name	Tạo một thư mục có tên dir-name trên hệ thống tập tin từ xa.
RMDIR	dir-name	Xóa thư mục có tên dir-name trên hệ thống tập tin từ xa
QUIT		Đóng nối kết FTP và thoát khỏi chương trình FTP

Mục lục

Chương 1 ..... **Error! Bookmark not defined.**
Tổng quan về mạng máy tính 1
1.1 Mạng điện báo 2
1.2 Mạng điện thoại 2
1.3 Mạng hướng đầu cuối 2
1.4 Mạng máy tính 3
1.4.1 Đường biên mạng 3
1.4.1.1 Mô hình khách hàng/người phục vụ (client/server): 4
1.4.1.2 Mô hình ngang cấp (peer-to-peer): 4
1.4.2 Đường trục mạng 4
1.4.2.1 Chuyển mạch (circuit switching) 4
1.4.2.2 Mạng chuyển gói 5
1.4.2.3 So sánh mạng chuyển mạch và mạng chuyển gói 5
1.4.2.4 Mạng truy cập 5
1.4.3 Các lợi ích của mạng máy tính 6
1.4.3.1 Mạng tạo khả năng dùng chung tài nguyên cho các người dùng. 6
1.4.3.2 Mạng cho phép nâng cao độ tin cậy. 6
1.4.3.3 Mạng giúp cho công việc đạt hiệu suất cao hơn. 6
1.4.3.4 Tiết kiệm chi phí. 6
1.4.3.5 Tăng cường tính bảo mật thông tin. 6
1.4.3.6 Việc phát triển mạng máy tính đã tạo ra nhiều ứng dụng mới 6
Chương 2 7
Các thành phần của mạng máy tính 7
2.1 Phần cứng mạng máy tính 8
2.1.1 Phân loại mạng máy tính theo kỹ thuật truyền tin 8
2.1.1.1 Mạng quảng bá 8
2.1.1.2 Mạng điểm nối điểm 8
2.1.2 Phân loại mạng máy tính theo phạm vị địa lý 8
2.1.2.1 Mạng cục bộ 8
2.1.2.1.1 Mạng hình bus 8
2.1.2.1.2 Mạng hình sao 9
2.1.2.1.3 Mạng hình vòng 9
2.1.2.2 Mạng đô thị 9
2.1.2.3 Mạng diện rộng 9
2.1.3 Mạng không dây 10
2.1.3.1 Nối kết hệ thống (System interconnection) 10
2.1.3.2 Mạng cục bộ không dây (Wireless LANs): 10
2.1.3.3 Mạng diện rộng không dây (Wireless WANs): 10
2.1.4 Liên mạng (Internetwork) 10
2.2 Phần mềm mạng 11
2.2.1 Cấu trúc thứ bậc của giao thức 11
2.2.2 Ví dụ về cấu trúc thứ bậc của giao thức 12
2.2.3 Dịch vụ mạng 13
2.2.3.1 Các phép toán của dịch vụ 14
2.2.3.2 Sự khác biệt giữa dịch vụ và giao thức 14
2.3 Mô hình tham khảo OSI 15
Chương 3 18
Tầng vật lý 18

3.1 Giới thiệu 19
3.2 Vấn đề số hóa thông tin 19
3.2.1 Số hóa văn bản 20
3.2.2 Số hóa hình ảnh tĩnh 21
3.2.3 Số hóa âm thanh và phim ảnh 22
3.3 Các loại kênh truyền 22
3.3.1 Kênh truyền hữu tuyến 22
3.3.1.1 Cáp xoắn đôi (Twisted Pair) 23
3.3.1.2 Cáp đồng trục (Coaxial Cable) 23
3.3.1.3 Cáp quang (Fiber Optic) 24
3.3.2 Kênh truyền vô tuyến 25
3.4 Đặc điểm kênh truyền 25
3.4.1.1 Truyền tải tín hiệu sóng dạng hình sin 26
3.4.1.2 Truyền tín hiện bất kỳ 27
3.4.1.3 Băng thông của một kênh truyền (Bandwidth) 27
3.4.1.4 Tần số biến điệu và tốc độ dữ liệu (Baund rate and bit rate) 28
3.4.1.5 Nhiễu và khả năng kênh truyền 29
3.4.1.6 Giao thông (Traffic) 30
3.5 Mã hóa đường truyền (Line Coding) 31
3.5.1 Mã hóa đường truyền bằng tín hiệu số 31
3.5.2 Mã hóa đường truyền bằng tín hiệu tuần tự 32
Chương 4 33
Tầng liên kết dữ liệu 33
(Data link layer) 33
4.1 Chức năng của tầng liên kết dữ liệu 34
4.1.1 Các dịch vụ cơ bản của tầng liên kết dữ liệu 34
4.1.2 Xử lý lỗi 34
4.1.3 Định khung 34
4.1.3.1 Phương pháp đếm ký tự (Character Count) 35
4.1.3.2 Phương pháp sử dụng byte làm cờ và các byte độn (Flag byte with byte stuffing) 35
4.1.3.3 Sử dụng cờ bắt đầu và kết thúc khung cùng với các bit độn (Starting and ending flags with bit stuffing) 35
4.1.4 Điều khiển lỗi (Error Control) 36
4.1.5 Điều khiển luồng (Flow Control) 36
4.2 Vấn đề xử lý lỗi 37
4.2.1 Bộ mã phát hiện lỗi 37
4.2.2 Những bộ mã phát hiện lỗi (Error-Detecting Codes) 37
4.2.2.1 Kiểm tra chẵn lẻ (Parity Check) 37
4.2.2.2 Kiểm tra thêm theo chiều dọc (Longitudinal Redundancy Check or Checksum) 38
4.2.2.3 Kiểm tra phần dư tuần hoàn (Cyclic Redundancy Check) 38
4.3 Một số giao thức điều khiển lỗi (Error Control) 40
4.3.1.1 Giao thức truyền đơn công không ràng buộc (Unrestricted Simplex Protocol) 42
4.3.1.2 Giao thức truyền đơn công dừng và chờ (Simplex Stop-and-wait Protocol) 43
4.3.1.3 Giao thức truyền đơn công cho kênh truyền có nhiễu (Simplex Protocol for Noisy Channel) 44
4.4 Giao thức của sổ trượt (Sliding windows) 45
4.4.1 Vấn đề truyền tải thông tin theo hai chiều (Duplex) 45
4.4.2 Giới thiệu về giao thức cửa sổ trượt 45

4.4.3 Hoạt động của cửa sổ trượt 46
4.4.4 Cài đặt giao thức cửa sổ trượt kích thước 1 bit (A One-Bit Sliding Window Protocol 47
4.4.5 Vấn đề điều khiển lỗi (Error Control) 48
4.4.5.1 Giao thức Go-Back-N 48
4.4.5.2 Giao thức Selective Repeat 51
4.4.5.2.1 Kích thước tối đa của cửa sổ gởi và nhận là bao nhiêu ? 54
4.4.5.2.2 Số lượng buffer để lưu khung là bao nhiêu? 54
4.4.5.2.3 Khi nào gởi báo nhận cho một gói tin? 54
4.4.6 Giao thức HDLC (High-Level Data Link Control) 54
4.4.6.1 Các đặc tính của giao thức HDLC 54
4.4.6.1.1 Ba loại trạm trong HDLC 55
4.4.6.2 Hai cấu hình đường nối kết: 55
4.4.6.3 Có 3 chế độ truyền tải là: 55
4.4.6.4 Cấu trúc khung 55
4.4.6.5 Một vài kịch bản về giao thức HDLC 57
4.4.6.6 Giao thức Điểm nối điểm (PPP- Point-to-Point Protocol) 59
Chương 5 61
MẠNG NỘI BỘ 61
& 61
LỚP CON ĐIỀU KHIỂN TRUY CẬP 61
5.1 Tổng quan về LAN 62
5.2 Hình thái mạng 62
5.2.1 Mạng hình sao 62
5.2.2 Mạng hình vòng 63
5.2.3 Mạng hình bus 63
5.3 Lớp con MAC (Media Access Control Sublayer) 63
5.3.1 Phương pháp chia kênh 64
5.3.1.1 Chia tần số (FDMA – Frequency Division Multiple Access) 64
5.3.1.2 Chia thời gian (TDMA – Time Division Multiple Access) 65
5.3.1.3 Kết hợp giữa FDMA và TDMA 65
5.3.1.4 Phân chia mã (CDMA – Code Division Multiple Access) 65
5.3.2 Phương pháp truy cập đường truyền ngẫu nhiên (Random Access) 68
5.3.2.1 ALOHA 68
5.3.2.2 CSMA – Carrier Sense Multiple Access 69
5.3.3 Phương pháp phân lượt truy cập đường truyền 73
5.3.3.1 Ví dụ về phương pháp thăm dò: Thăm dò phân tán (Distributed Polling) . 73
5.3.3.2 Ví dụ về phương pháp chuyển thẻ bài: Token Ring 74
5.3.3.3 Ví dụ về phương pháp chuyển thẻ bài: Token Bus 79
5.4 Chuẩn hóa mạng cục bộ 80
5.5 Giới thiệu một số công nghệ mạng LAN 82
5.5.1 Ethernet (802.3) 82
5.5.1.1 Tổng quan 82
5.5.1.2 Khuôn dạng khung thông tin của Ethernet 84
5.5.1.3 Địa chỉ Ethernet 84
5.5.1.4 Cách thức mã hóa tín hiệu 84
5.5.1.5 Giải thuật truy cập đường truyền 85
5.5.1.6 Các công nghệ Ethernet 85
5.5.2 FDDI (Fiber Distributed Data Interface) 86
5.5.2.1 Các tính chất vật lý 86
5.5.2.2 Giải thuật "Thẻ bài được định thời" – Timed Token 87
5.5.2.3 Quản lý thẻ bài 88

5.5.3 Mạng không dây (802.11) 89
5.5.3.1 Các tính chất vật lý....89
5.5.3.2 Tránh đụng độ (Collision Avoidance) 90
5.5.3.3 Hệ thống phân tán 91
5.5.3.4 Khuôn dạng khung 92
Chương 6 93
Tầng mạng....93
(Network Layer)....93
6.1 Giới thiệu 94
6.2 Các vấn đề liên quan đến việc thiết kế tầng mạng 94
6.2.1.1 Kỹ thuật hoán chuyển lưu và chuyển tiếp (Store-and-Forward Switching)94
6.2.2 Các dịch vụ cung cấp cho tầng vận chuyển 95
6.2.2.1 Cài đặt dịch vụ không nối kết (Implementation of Connectionless Service) 95
6.2.2.2 Cài đặt dịch vụ định hướng nối kết (Connection – Oriented Service)....96
6.2.2.3 So sánh giữa Datagram subnet và Virtual-Circuit subnet....97
6.3 Giải thuật chọn đường....98
6.3.1 Giới thiệu....98
6.3.2 Mục tiêu của giải thuật chọn đường 98
6.3.3 Phân loại giải thuật chọn đường 98
6.3.4 Các giải thuật tìm đường đi tối ưu....99
6.3.4.1 Giải thuật tìm đường đi ngắn nhất Dijkstra 99
6.3.4.2 Giải thuật chọn đường tối ưu Ford-Fulkerson 100
6.3.5 Giải pháp vạch đường Vector Khoảng cách (Distance Vector)....101
6.3.6 Giải pháp chọn đường "Trạng thái nối kết" (Link State)....104
6.3.6.1 Làm ngập một cách tin cậy (Reliable Flooding)....104
6.3.6.2 Tính toán chọn đường trong Link State 106
6.3.7 Vạch đường phân cấp (Hierarchical Routing) 106
6.3.8 Vạch đường trong mạng di động....107
6.4 Các giải thuật chống tắc nghẽn 109
6.4.1 Các nguyên tắc chung để điều khiển tắc nghẽn 109
6.4.2 Các biện pháp phòng ngừa tắc nghẽn....110
6.4.3 Điều khiển tắc nghẽn trong các mạng con dạng mạch ảo 111
6.4.4 Điều khiển tắc nghẽn trong mạng con dạng Datagram 111
6.5 Liên mạng 113
6.5.1 Các mạng con được nối kết với nhau ra sao?....114
6.5.2 Nối kết các mạng con dạng mạch ảo....115
6.5.3 Nối kết các mạng con dạng datagram 116
6.5.4 Vạch đường trong liên mạng 117
6.5.5 Phân mảnh và tái hợp 117
6.6 Bộ giao thức liên mạng (IPs - Internet Protocols) 119
6.6.1 Giới thiệu....119
6.6.2 Giao thức liên mạng IP (Internet Protocol)....120
6.6.2.1 Định dạng gói tin IP (IP Packet Format)....120
6.6.3 Cấu trúc địa chỉ IP 121
6.6.4 Một số địa chỉ IP đặc biệt....123
6.6.5 Ý nghĩa của Netmask 123
6.6.6 Phân mạng con (Subnetting) 123
6.6.6.1 Giới thiệu 123
6.6.6.2 Phương pháp phân mạng con....124
6.6.6.2.1 Phương pháp phân lớp hoàn toàn (Classfull Standard)....125

6.6.6.2.2 Phương pháp Vạch đường liên miền không phân lớp CIDR (Classless Inter-Domain Routing) 126
6.6.7 Vạch đường trong giao thức IP 128
6.6.7.1 Đường đi của gói tin 130
6.6.7.2 Giao thức phân giải địa chỉ (Address Resolution Protocol) 131
6.6.7.3 Giao thức phân giải địa chỉ ngược RARP (Reverse Address Resolution Protocol) 132
6.6.7.4 Giao thức thông điệp điều khiển Internet ICMP (Internet Control Message Protocol) 132
Chương 7 134
TẦNG VẬN CHUYỂN 134
7.1 Dịch vụ của tầng vận chuyển 135
7.1.1 Các dịch vụ cung cấp cho tầng ứng dụng 135
7.1.2 Các hàm dịch vụ cơ sở 136
7.2 Các yếu tố cấu thành giao thức vận chuyển 136
7.2.1 Định địa chỉ 137
7.2.2 Thiết lập nối kết 138
7.2.3 Giải phóng nối kết 139
7.2.4 Điều khiển thông lượng 141
7.3 Tầng vận chuyển trong mạng Internet 142
7.3.1 Giao thức UDP (User Datagram Protocol) 142
7.3.2 Giao thức TCP (Transmission Control Protocol) 143
Chương 8 149
CÁC ỨNG DỤNG MẠNG 149
8.1 Dịch vụ tên (DNS) 149
8.1.1 Miền phân cấp 150
8.1.2 Các server phục vụ tên 150
8.1.3 Phương pháp phân tích tên 153
8.2 Electronic Mail (SMTP, MIME, POP3, IMAP) 154
8.2.1 Các thành phần của hệ thống email 154
8.2.2 Khuôn dạng của một email 155
8.2.3 Chuyển thư 157
8.2.4 Phân phát thư 159
8.2.4.1 POP3 159
8.2.4.2 IMAP 160
8.3 World Wide Web (HTTP) 161
8.3.1 Các thông điệp yêu cầu 162
8.3.2 Các thông điệp trả lời 163
8.3.3 Các kết nối TCP 163
8.3.4 Trữ đệm 164
8.4 Truyền tập tin (FTP) 164
8.4.1 Mô hình dịch vụ FTP 165
8.4.2 Giao thức FTP 165
8.4.3 Các lệnh cơ bản 165