

**TS. NGÔ QUỲNH THU**

**ĐÁNH GIÁ  
HIỆU NĂNG MẠNG**

**NHÀ XUẤT BẢN BÁCH KHOA – HÀ NỘI**

Bản quyền thuộc về trường Đại học Bách Khoa Hà Nội.

Mọi hình thức xuất bản, sao chép mà không có sự cho phép bằng văn bản của trường là vi phạm pháp luật.

---

**Mã số: 58 – 2013/CXB/92 – 01/BKHN**

**Biên mục trên xuất bản phẩm của Thư viện Quốc gia Việt Nam**

Ngô Quỳnh Thu

Đánh giá hiệu năng mạng / Ngô Quỳnh Thu. - H. : Bách khoa Hà Nội, 2013. - 172tr. ; 24cm

Thư mục: tr. 171

ISBN 9786049113420

1. Mạng máy tính 2. Hiệu năng  
004.6 - dc14

BKB0065p-CIP

## LỜI NÓI ĐẦU

Đánh giá hiệu năng mạng máy tính nói riêng và các hệ thống truyền thông nói chung luôn là vấn đề thời sự, thu hút được sự quan tâm của những người làm việc trong lĩnh vực mạng. Những phương pháp phân tích và đánh giá hiệu năng mạng giúp con người tiến gần hơn nữa tới các ứng dụng thực tế cũng như khả năng nâng cao được hiệu năng cho các hệ thống mạng và truyền thông hiện tại.

Cuốn sách ***Đánh giá hiệu năng mạng*** được tác giả biên soạn theo nội dung giảng dạy môn học Đánh giá hiệu năng mạng dành cho sinh viên năm thứ 5 hệ đại học chính quy Trường Đại học Bách Khoa Hà Nội.

Sách được bố cục thành sáu chương, với các nội dung như sau:

Chương 1 giới thiệu những khái niệm cơ bản nhất của đánh giá hiệu năng mạng.

Chương 2 đề cập đến các kiến thức cơ bản về xác suất thống kê và các tiến trình ngẫu nhiên.

Hai chương 3, 4 trình bày các khái niệm về hàng đợi, mạng hàng đợi và cách sử dụng chúng để đánh giá hiệu năng của các hệ thống trong thực tế.

Chương 5 đưa ra các khái niệm về chất lượng dịch vụ và một số mô hình cung cấp chất lượng dịch vụ.

Chương 6 trình bày các kỹ thuật mô phỏng cùng với các công cụ được sử dụng rộng rãi hiện nay.

Trong quá trình biên soạn sách, tác giả đã nhận được sự giúp đỡ rất nhiệt tình và nhiều ý kiến đóng góp bổ ích từ PGS. TS. Đặng Văn Chuyết, PGS. TS. Nguyễn Linh Giang, TS. Nguyễn Kim Khánh, PGS. TS. Nguyễn Hữu Thanh và toàn thể các thầy cô trong Bộ môn Truyền thông và Mạng máy tính,

Viện Công nghệ Thông tin và Truyền thông, Trường Đại học Bách Khoa Hà Nội. Tác giả xin chân thành cảm ơn.

Tuy nhiên, do cuốn sách được xuất bản lần đầu nên chắc chắn sẽ không tránh khỏi những thiếu sót. Chúng tôi rất mong nhận được những ý kiến đóng góp của bạn đọc và đồng nghiệp để nội dung sách được hoàn thiện hơn. Mọi ý kiến đóng góp xin gửi về Bộ môn Truyền thông và Mạng máy tính, Viện Công nghệ Thông tin và Truyền thông, Trường Đại học Bách Khoa Hà Nội, số 1 Đại Cồ Việt, quận Hai Bà Trưng, Hà Nội.

Xin trân trọng giới thiệu cùng bạn đọc.

**TÁC GIẢ**

# MỤC LỤC

LỜI NÓI ĐẦU .....	3
<b>CHƯƠNG 1. GIỚI THIỆU.....</b>	<b>9</b>
1.1. Mục đích của việc mô hình hóa và đánh giá hiệu năng .....	9
1.2. Phân loại các phương pháp mô hình hóa.....	13
1.3. Các tham số sử dụng khi đánh giá hiệu năng .....	15
1.4. Các công cụ đánh giá hiệu năng .....	17
Tài liệu tham khảo.....	18
<b>CHƯƠNG 2. CÁC TIẾN TRÌNH NGẪU NHIÊN .....</b>	<b>19</b>
2.1. Xác suất và sự kiện (Probability and Event) .....	19
2.1.1. <i>Phép thử và sự kiện ngẫu nhiên</i> .....	19
2.1.2. <i>Định nghĩa xác suất</i> .....	20
2.2. Biến ngẫu nhiên .....	21
2.2.1. <i>Khái niệm biến ngẫu nhiên</i> .....	21
2.2.2. <i>Các hàm phân phối xác suất và bảng phân phối xác suất</i> .....	22
2.2.3. <i>Các số đặc trưng của biến ngẫu nhiên</i> .....	25
2.3. Các mô hình phân bố xác suất cơ bản .....	29
2.3.1. <i>Phân bố Bernoulli</i> .....	29
2.3.2. <i>Phân bố nhị thức</i> .....	30
2.3.3. <i>Phân bố đều (Uniform Distribution)</i> .....	30
2.3.4. <i>Phân bố chuẩn (Gaussian Distribution)</i> .....	31
2.3.5. <i>Phân bố mũ (Exponential Distribution)</i> .....	32
2.3.6. <i>Phân bố Poisson (Poisson Distribution)</i> .....	33
2.3.7. <i>Phân bố Gamma (Gamma Distribution)</i> .....	33

2.3.8. Phân phối $\chi^2$ với $n$ bậc tự do.....	33
2.3.9. Phân bố Beta .....	34
2.4. Tiến trình ngẫu nhiên ( <i>Stochastic Process</i> ) .....	35
2.4.1. Định nghĩa tiến trình ngẫu nhiên .....	35
2.4.2. Phân loại các tiến trình ngẫu nhiên .....	35
2.4.3. Các tiến trình ngẫu nhiên thường gặp.....	36
Bài tập chương 2 .....	43
Tài liệu tham khảo.....	44
<b>CHƯƠNG 3. HỆ THỐNG HÀNG ĐỢI .....</b>	<b>45</b>
3.1. Giới thiệu .....	45
3.2. Mô hình hàng đợi – ký hiệu Kendall .....	46
3.2.1. Mô hình hàng đợi đơn .....	46
3.2.2. Ký hiệu Kendall.....	48
3.2.3. Các tham số quan trọng để đánh giá đặc tính của hệ thống hàng đợi.....	51
3.2.4. Hệ thống đóng .....	51
3.2.5. Định lý Little.....	53
3.2.6. Một số đặc tính khác của hệ thống đóng, hoạt động ở trạng thái ổn định .....	56
3.3. Các mô hình hàng đợi .....	58
3.3.1. Tiến trình sinh tử.....	58
3.3.2. Hệ thống hàng đợi M/M/1/0.....	59
3.3.3. Hệ thống Hàng đợi M/M/1 .....	62
3.3.4. Hàng đợi M/M/1/K.....	66
3.3.5. Hàng đợi M/M/m .....	70
3.3.6. So sánh các hệ thống hàng đợi.....	72

Bài tập chương 3.....	74
Tài liệu tham khảo.....	76
<b>CHƯƠNG 4. HỆ THỐNG MẠNG HÀNG ĐỢI .....</b>	<b>78</b>
4.1. Mạng hàng đợi.....	78
4.2. Hệ thống mạng nội tiếp.....	81
4.3. Hệ thống mạng Jackson mở .....	83
4.4. Mạng Jackson đóng.....	87
Bài tập chương 4 .....	90
Tài liệu tham khảo.....	92
<b>CHƯƠNG 5. CHẤT LƯỢNG DỊCH VỤ (QOS).....</b>	<b>93</b>
5.1. Tại sao phải cung cấp chất lượng dịch vụ cho mạng Internet ....	93
5.2. Một số mô hình cung cấp chất lượng dịch vụ .....	98
5.2.1. Cấu trúc Best-Effort (BE) .....	98
5.2.2. Cấu trúc dịch vụ tích hợp (Integrated Services Architecture – IntServ) .....	98
5.2.3. Cấu trúc dịch vụ phân biệt (Differentiated Services – DiffServ) .....	104
5.2.4. MPLS (Multiprotocol Label Switching) .....	117
5.2.5. Kỹ thuật lưu lượng (Traffic Engineering) và các phương pháp định tuyến có điều kiện (Constrained Based Routing) .....	120
Bài tập chương 5 .....	127
Tài liệu tham khảo .....	128
<b>CHƯƠNG 6. MÔ PHỎNG .....</b>	<b>131</b>
6.1. Các kỹ thuật mô phỏng .....	131
6.1.1. Thực hiện mô phỏng theo hướng sự kiện.....	137
6.1.2. Bộ phát số ngẫu nhiên (Random Number Generation RNG)..	138
6.2. Đánh giá thống kê kết quả mô phỏng .....	143

6.2.1. Các kết quả thu được .....	144
6.2.2. Giá trị trung bình và khoảng tin cậy ( <i>Confidence Intervals</i> )..	146
6.3. Giới thiệu một số công cụ mô phỏng .....	151
6.3.1. <i>OPNET</i> .....	152
6.3.2. <i>NS2</i> .....	153
6.3.3. <i>NS3</i> .....	154
6.3.4. <i>OMNeT++</i> .....	155
6.4. <i>NS2</i> VÀ <i>OMNeT++</i> .....	156
6.4.1. <i>OMNeT++</i> .....	156
6.4.2. <i>NS2</i> .....	161
Bài tập chương 6 .....	168
Tài liệu tham khảo.....	171



# Chương 1

## GIỚI THIỆU

### 1.1. MỤC ĐÍCH CỦA VIỆC MÔ HÌNH HÓA VÀ ĐÁNH GIÁ HIỆU NĂNG

Mục đích của các phương pháp đánh giá hiệu năng là để dự đoán hoạt động của hệ thống. Khi xây dựng một hệ thống mới hoặc sửa chữa nâng cấp một hệ thống cũ, người ta phải sử dụng các phương pháp đánh giá hiệu năng để dự đoán được ảnh hưởng của nó đến hiệu năng của hệ thống đó.

Khía cạnh quan trọng nhất của đánh giá hiệu năng là đo đạc và theo dõi hiệu năng của hệ thống. Bằng việc quan sát theo dõi hiệu năng của hệ thống, ta có thể biết được hoạt động của hệ thống đó như thế nào, hoặc khi nào hệ thống quá tải. Điều kiện tiên quyết của việc đo đạc hiệu năng là hệ thống đó phải có các thông số có thể đo được. Một khía cạnh quan trọng khác nữa của việc đánh giá hiệu năng là trong quá trình đo đạc thông thường người ta sẽ làm hệ thống thay đổi chút ít, như khi thêm một vài thông số hoặc nhấn vào dữ liệu chẳng hạn. Điều này cũng có thể dẫn tới làm thay đổi hiệu năng của hệ thống đó.

Để có thể hiểu thêm về bài toán đánh giá hiệu năng, chúng ta cùng xem xét ví dụ sau. Giả sử hệ thống cần phải được đánh giá hiệu năng là một mạng Ethernet, được trang bị  $N$  máy tính. Người ta tiến hành đo tổng lưu lượng đầu vào của mạng Ethernet này và kết quả đo được là  $\lambda$  Mbit/s. Thiết bị truyền dẫn của mạng có dung lượng là  $C$  (Mbit/s). Để có thể đánh giá được hiệu năng của mạng Ethernet này, thì một ví dụ điển hình nhất lúc này là người ta phải tính toán được hiệu suất hoạt động của thiết bị truyền dẫn

của mạng và **trễ trung bình của một gói tin** khi gói đó được truyền từ nguồn tới đích. Lưu ý rằng ở đây **hiệu suất hoạt động** của thiết bị truyền dẫn của mạng Ethernet này được tính bằng **% của lưu dung lượng  $C$  và trễ trung bình của một gói tin khi được truyền từ nguồn tới đích được tính bằng  $s$ .**

Để đánh giá hiệu năng, người ta chia làm ba phương pháp theo dõi như sau:

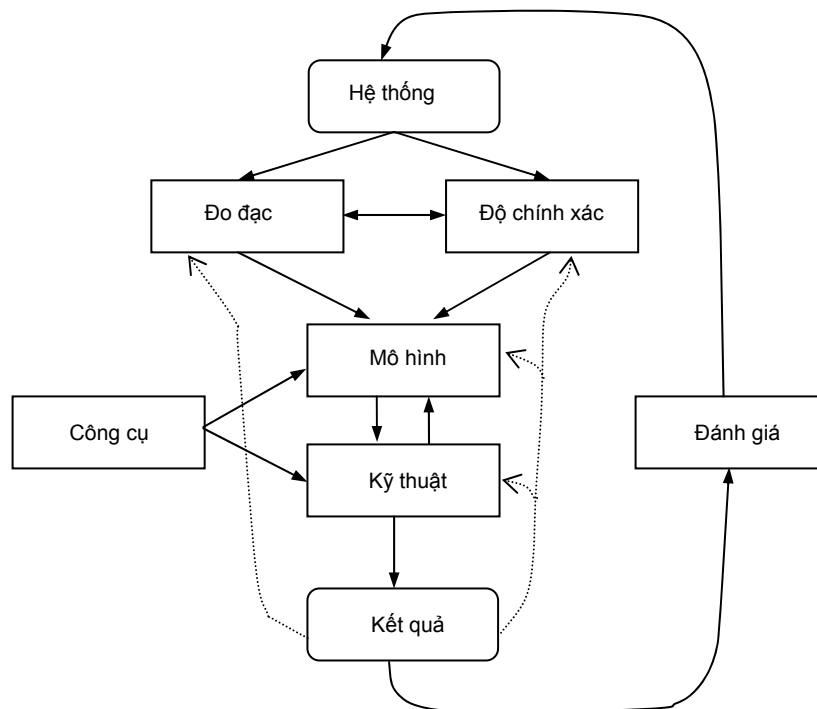
- **Theo dõi bằng phần mềm** (*software monitoring*): là trường hợp các chức năng của quá trình theo dõi được cài đặt bằng một phần mềm.
- **Theo dõi bằng phần cứng** (*hardware monitoring*): là trường hợp phải có thêm một vài thiết bị phụ trợ để có thể theo dõi hệ thống.
- **Theo dõi kiểu lai** (*hybrid monitoring*): là tổng hợp cả hai loại trên.

Trong cả ba trường hợp trên, người ta thấy rằng việc theo dõi đều cần phải có thêm các thiết bị đặc biệt và dù là phần cứng hay phần mềm thì cũng thường rất đắt tiền.

Chính vì lý do này, người ta đã thay thế các phương pháp trên bằng một giải pháp: **đánh giá hiệu năng dựa trên mô hình** (*model-based performance evaluation*), hay còn gọi là phương pháp mô hình hóa. Việc mô hình hóa ở đây có thể hiểu là người ta sẽ tiến hành mô tả các thành phần của hệ thống **một cách trừu tượng** (về mặt toán học chẳng hạn), đồng thời cũng phải mô tả tương tác của các thành phần của hệ thống cũng như tương tác với môi trường bên ngoài. Môi trường bên ngoài ở đây có thể hiểu là con người hoặc các hệ thống khác và thường được gọi là mô hình tải (*workload model*).

Cụ thể hơn nữa, có thể định nghĩa mô hình hay quá trình mô hình hóa như sau:

Mô hình (*model*) thông thường hay quá trình mô hình hóa (*modelling technique*), thực chất chính là **quá trình làm đơn giản hóa một hệ thống thực**. Quá trình này có thể được thực hiện bằng cách miêu tả các hoạt động và trạng thái của hệ thống đó có kèm theo một số điều kiện, như là các điều kiện khởi đầu và điều kiện bờ cho trước.



**Hình 1.1. Chu kỳ đánh giá hiệu năng dựa theo mô hình**

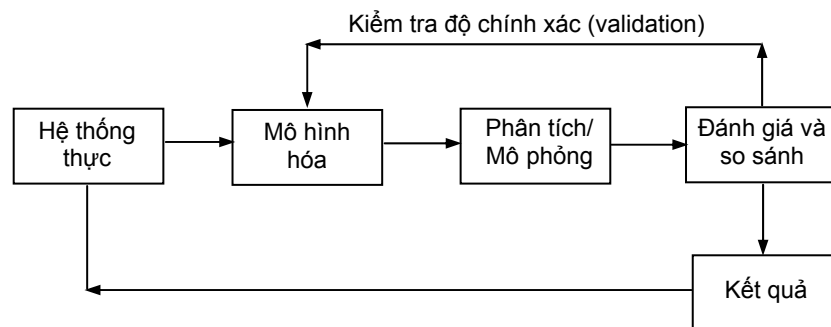
Đối với các hệ thống máy tính và truyền thông, để có thể mô hình hóa, trước hết người ta phải xác định các thành phần của hệ thống đó. Tuy nhiên đây là một nhiệm vụ rất khó khăn và phải đối mặt với nhiều thách thức. Trên thực tế, không một cuốn sách nào có thể hướng dẫn chúng ta được cách xây dựng mô hình đánh giá hiệu năng chính xác cho tất cả các hệ thống máy tính và truyền thông cả mà chỉ có thể có một vài hướng dẫn tổng quan mà thôi. Ví dụ, việc lựa chọn một mô hình cụ thể sẽ phụ thuộc rất nhiều vào thông số cần đo đạc là gì. Sau khi trả lời được câu hỏi này, câu hỏi tiếp theo là chúng ta muốn đo thông số cụ thể đó như thế nào? Giá trị trung bình, phương sai hay phân bố của thông số đó? Tùy thuộc vào từng thông số khác nhau mà người ta sẽ lựa chọn một mô hình cụ thể. Ví dụ đối với một hệ thống máy tính đa nhiệm (*multiprogrammed computer system*), để có thể đo đạc được thời gian trả lời trung bình, người ta phải sử dụng một mô hình khác với mô hình khi đo xác suất để thời gian trả lời lớn hơn một mức

ngưỡng nào đó. Trong trường hợp các thông số đo đặc chung chung và không có yêu cầu nào cụ thể, thì mô hình có thể rất đơn giản. Nhưng trong trường hợp phải đo đặc các thông số chính xác, mô hình sẽ phải đưa vào rất nhiều thành phần của hệ thống và sẽ trở nên rất phức tạp. Ngoài ra cũng cần phải nhấn mạnh thêm rằng, trong rất nhiều trường hợp, khi đánh giá hiệu năng theo phương pháp mô hình, có rất nhiều thông số và khía cạnh của hệ thống được đo đặc nhưng có thể sẽ không chính xác. Để giải quyết vấn đề này, người ta thường đưa ra một mô hình trừu tượng đơn giản chứ không lựa chọn một mô hình quá phức tạp mà không cung cấp được thông số đầu vào.

Sau khi đã xây dựng được mô hình, bước tiếp theo là phân tích mô hình đó. Quá trình phân tích có thể sử dụng nhiều kỹ thuật khác nhau. Các hệ thống truyền thông và máy tính trong thực tế thường quá phức tạp để có thể được mô hình hóa và phân tích bằng các phép tính đơn giản. Trong rất nhiều trường hợp, việc xây dựng mô hình và phân tích nó có thể được thực hiện nhờ sự trợ giúp của một số phần mềm.

Sau khi phân tích xong, các thông số đầu ra cũng cần phải được xem xét một cách chi tiết. Trước hết, các thông số đầu ra cần thỏa mãn yêu cầu ban đầu. Nếu các thông số đầu ra không thỏa mãn yêu cầu ban đầu, người ta sẽ phải lựa chọn một thông số khác nhưng với độ chính xác khác. Một giải pháp nữa là có thể sử dụng kỹ thuật phân tích khác hoặc có thể thay đổi hẳn mô hình ban đầu. Việc đánh giá này có thể dẫn đến việc xem xét chỉ một bộ phận riêng biệt của hệ thống.

Điểm cuối cùng cần phải nhấn mạnh là việc đánh giá hiệu năng của hệ thống cũng cần phải gắn liền với việc theo dõi hiệu năng của hệ thống đó, đặc biệt khi hệ thống có sự thay đổi. Trong trường hợp này, người ta có thể đo đặc một sự kiện cụ thể của hệ thống để xác định một vài thông số cụ thể. Các thông số này có thể được đưa vào một mô hình để mô hình hóa hệ thống đó. Sau đó sử dụng mô hình này, các thông số thiết lập hệ thống sẽ được ước lượng, qua đó có thể kết luận rằng hệ thống thật sẽ thích nghi như thế nào khi có sự thay đổi.



**Hình 1.2. Các bước cơ bản trong mô hình hóa và đặc tính đánh giá của hệ thống**

## 1.2. PHÂN LOẠI CÁC PHƯƠNG PHÁP MÔ HÌNH HÓA

Có rất nhiều các phương pháp mô hình hóa khác nhau và qua đó cũng có nhiều cách phân loại khác nhau. Phương pháp phân loại phổ biến nhất hay được sử dụng là **phương pháp phân tích toán học** (*analytical techniques*) và **phương pháp mô phỏng** (*simulative techniques*).

Phương pháp phân tích toán học được thực hiện trong trường hợp nếu như mô hình hệ thống thỏa mãn một số điều kiện nào đó và người ta có thể tính toán trực tiếp các thông số quan trọng của mô hình đó. Phương pháp phân tích này thường rất tiện lợi. Tuy nhiên trong thực tế không phải hệ thống nào cũng có thể thỏa mãn được tất cả các điều kiện nào đó. **Các bước do phương pháp phân tích toán học** được thực hiện như sau:

- Mô tả hình thức một hệ thống.
- Định nghĩa các mối quan hệ trong hệ thống, mô tả chúng bằng các công thức, mô hình toán học.
- Tính toán dựa vào mô hình toán học vừa lập.

Nhược điểm lớn nhất của phương pháp phân tích toán học là **không phải lúc nào cũng có các hệ thống có thể thỏa mãn tất cả các điều kiện cho trước**. Trong nhiều trường hợp khác, không thể sử dụng phương pháp phân

tích toán học để đánh giá hiệu năng của hệ thống. Lúc này người ta có thể sử dụng **phương pháp mô phỏng**. Đối với phương pháp này, người ta sẽ **lựa chọn một chương trình mô phỏng thích hợp** và **sử dụng chương trình mô phỏng này để mô tả lại các hoạt động của hệ thống**.

Sau đó, người ta có thể tiến hành thu thập dữ liệu đo đạc được sau một thời gian chạy chương trình mô phỏng này và tiến hành đánh giá dựa trên các dữ liệu đó. Một cách nôm na, người ta có thể nói rằng mô phỏng chính là việc **miêu tả một quá trình xảy ra trong thực tế** thông qua các chương trình máy tính. Để có thể đánh giá đặc tính hoạt động của một hệ thống từ các kết quả thí nghiệm thu được thông qua quá trình mô phỏng, người ta phải dựa trên **cơ sở của xác suất thống kê**. Các kết quả thí nghiệm thu được dựa trên cơ sở xác suất thống kê có thể kể ra là **giá trị kỳ vọng, phương sai, các phân bố ngẫu nhiên** của kết quả...

**Ưu điểm lớn nhất** của mô phỏng là việc xây dựng, phân tích và đánh giá dễ dàng hơn so với phương pháp toán học. Trong nhiều trường hợp, mô phỏng là phương pháp khả thi nhất về mặt thời gian và tài chính (không phải mua một hệ thống thực) để khảo sát và đánh giá đặc tính một hệ thống.

Ngoài ưu điểm lớn nhất kể trên thì phương pháp phân tích bằng mô phỏng cũng có những **nhược điểm** của nó. Thứ nhất là khi mô phỏng các hệ thống phức tạp thì **thời gian chạy chương trình khá lớn**. Vì vậy người ta cũng phải thực hiện việc đơn giản hóa đi khá nhiều các hệ thống thật khi chuyển sang mô hình mô phỏng. Nhược điểm thứ hai là **độ chính xác kết quả của mô hình mô phỏng tương đối khó kiểm chứng**. Một trong các phương pháp để kiểm tra độ chính xác của mô hình là chạy chương trình mô phỏng với các tham số đầu vào mà giá trị đầu ra đã được biết trước, sau đó so sánh các kết quả mô phỏng với kết quả đo đạc được trong thực tế.

Ngoài hai phương pháp phân tích toán học và phương pháp mô phỏng kể trên, còn phải kể đến một phương pháp nữa được thực hiện theo kiểu lai (*hybrid technique*). Kiểu lai ở đây có thể hiểu là **vừa theo kiểu toán học vừa**

**theo kiểu mô phỏng**. Ngoài ra, phương pháp phân tích theo kiểu lai này có thể thực hiện trên từng phần của hệ thống hoặc trên toàn bộ hệ thống.

Trong cả ba phương pháp phân tích trên (toán học, mô phỏng và lai), mỗi loại đều có các ưu nhược điểm riêng của nó. Ví dụ như phương pháp phân tích toán học thường không tốn kém và cho phép chúng ta có thể hiểu và đánh giá một cách rõ ràng các đặc tính của hệ thống. Tuy nhiên trong thực tế các hệ thống đều rất phức tạp và hiếm khi có thể mô hình hóa chính xác được hệ thống bằng các phương pháp phân tích toán học. Vì vậy, để có thể tiết kiệm chi phí mà vẫn đánh giá được hệ thống, người ta đưa vào một phương pháp nữa, gọi là các **phương pháp phân tích toán học xấp xỉ** (*approximate technique*). Nói cách khác, phương pháp xấp xỉ được sử dụng khi phương pháp phân tích toán học quá phức tạp để có thể mô tả một hệ thống. Cũng chính vì chỉ là xấp xỉ, nên lúc này các thông số đầu ra nhận được sẽ thường không chính xác.

Một phương pháp cuối cùng phải kể đến, đó là **phương pháp đo đạc** (*measurement method*). Đo đạc được sử dụng để đo các thông số cần thí nghiệm trong các hệ thống thực (như thông lượng, băng thông, trễ, tuyến đường mà một gói IP đi qua...). **Đo đạc được sử dụng để hỗ trợ cho phương pháp phân tích toán học và phương pháp mô phỏng**. Các tham số đầu vào sử dụng trong các mô hình toán học và mô phỏng đều được đo đạc trong thực tế. Mặt khác, đo đạc cũng được sử dụng để kiểm tra độ chính xác của một mô hình so với hệ thống thực tế.

### **1.3. CÁC THAM SỐ SỬ DỤNG KHI ĐÁNH GIÁ HIỆU NĂNG**

Như đã nhấn mạnh ở trên, các hệ thống truyền thông và máy tính thường rất phức tạp và khi sử dụng một mô hình để đánh giá hiệu năng của nó, rất khó để có thể xác định các thông số đầu vào một cách chính xác. Nói cách khác, sẽ có những yếu tố ngẫu nhiên xảy ra khi mô hình hóa các hệ thống đó.

Chính vì các yếu tố ngẫu nhiên này, người ta có thể đánh giá hiệu năng của hệ thống bằng cách thực hiện rất nhiều lần với các tham số khác nhau hay bằng cách phân tích độ nhạy của các tham số này. Cả hai giải pháp đều có thể giúp chúng ta biểu diễn được bằng đồ thị hiệu năng của hệ thống và biểu diễn nó bằng một vài tham số đầu ra, ứng với các tham số đầu vào cho trước.

Trong nhiều trường hợp khác nữa, các yếu tố ngẫu nhiên không nằm trong hệ thống mà nằm ở các thông số đầu vào của nó (hay còn được gọi là tải của hệ thống). Tải đầu vào của hệ thống này phụ thuộc vào rất nhiều tham số và thường không thể miêu tả được một cách chính xác vì chúng ta thường không biết được trong tương lai người dùng sẽ yêu cầu hệ thống phải làm gì cho họ. Chính vì vậy, các yếu tố ngẫu nhiên này thường dẫn đến sự xuất hiện của các biến ngẫu nhiên trong mô hình.

Người ta sẽ phải tiến hành lựa chọn các tham số cụ thể khi đánh giá hiệu năng hay đặc tính hoạt động của hệ thống. Có thể phân loại các tham số này như sau:

- Các tham số thời gian có thể có là:
  - + Thời gian hệ thống thực hiện một yêu cầu.
  - + Thời gian đáp ứng của hệ thống.
  - + Thời gian một yêu cầu lưu lại trong hệ thống...
- Các tham số không gian có thể có là:
  - + Độ lớn của hàng đợi hệ thống (độ lớn bộ đệm ...).
  - + Yêu cầu về bộ nhớ cho một chương trình...
- Các tham số khác:
  - + Thông lượng.
  - + Giá thành ...



## 1.4. CÁC CÔNG CỤ ĐÁNH GIÁ HIỆU NĂNG

Trong thực tế, bài toán phân tích và đánh giá hiệu năng các hệ thống truyền thông và mạng máy tính là bài toán rất hay gặp, nên đã có rất nhiều các công cụ được tạo ra để phục vụ cho mục đích này.

Loại công cụ thứ nhất **sử dụng phương pháp đo đạc từ các hệ thống thực**. Nhiệm vụ quan trọng nhất của các loại công cụ này là phải phân tích các số liệu đo được, chủ yếu bằng các máy đo (phần cứng) hoặc bằng các phần mềm đo đạc. Các công cụ đo này sẽ sử dụng nhiều biện pháp khác nhau để đo tải đầu vào, sau đó đo đạc và biểu diễn các thông số ở đầu ra. Chúng có thể là những thiết bị từ rất đơn giản cho đến phức tạp và phức tạp nhất có thể là các bộ phân tích giao thức ở các lớp khác nhau của mô hình OSI.

Ngoài các công cụ đo đạc trực tiếp từ các hệ thống thực, người ta còn có thể sử dụng **các công cụ toán học, hoặc trong trường hợp phức tạp hơn, sử dụng các phần mềm mô phỏng** để đánh giá hiệu năng của các hệ thống truyền thông và mạng máy tính.

## TÀI LIỆU THAM KHẢO

1. Arnold O. Allen. *Probability, Statistics and Queueing Theory with Computer Science Applications*. Academic Press, Boston et al, 2<sup>nd</sup> edition, 1990.
2. Donald Gross and Carl M. Morris. *Fundamentals of Queueing Theory*. John Wiley and Sons, New York, 2<sup>nd</sup> edition, 1985.
3. Randolph Nelson. *Probability, Stochastic Process and Queueing Theory*. Springer, Heidelberg New York, 1995.
4. Kishor S. Trivedi. *Probability, Statistics with Reliability, Queueing and Computer Science Applications*. Prentice–Hall, Englewood Cliffs, NJ, 1982.

## Chương 2

# CÁC TIẾN TRÌNH NGẪU NHIÊN

### 2.1. XÁC SUẤT VÀ SỰ KIỆN (PROBABILITY AND EVENT)

Trong mục này, các khái niệm cơ bản trong môn xác suất thống kê sẽ được trình bày.

#### 2.1.1. Phép thử và sự kiện ngẫu nhiên

Khái niệm thường gặp trong lý thuyết xác suất là sự kiện. Sự kiện được hiểu như là một sự việc, một hiện tượng nào đó của cuộc sống tự nhiên và xã hội. Khi thực hiện một tập hợp điều kiện xác định, gọi tắt là bộ điều kiện, gọi là một phép thử, có thể có nhiều kết cục khác nhau.

Ví dụ như khi gieo một con xúc xắc đồng chất trên một mặt phẳng chính là một phép thử. Phép thử này có 6 kết cục là: xuất hiện mặt 1, mặt 2..., mặt 6 chấm. Mỗi kết cục này cùng với các kết quả phức tạp hơn như xuất hiện mặt có số chấm chẵn, mặt có số chấm bội 3, đều có thể coi là các sự kiện.

Hoặc khi một bà mẹ sinh con thì có thể hiểu như là một phép thử ngẫu nhiên mà kết quả có thể là trai hoặc gái.

Một ví dụ nữa là khi kiểm tra một học sinh về một môn học nào đó cũng được xem như tiến hành một phép thử với kết quả có thể là đạt hay không đạt.

Người ta có thể định nghĩa phép thử và sự kiện ngẫu nhiên như sau:

Một phép thử ngẫu nhiên (*random experiment*) là sự thực hiện một nhóm các điều kiện xác định và có thể được lặp lại nhiều lần. Kết quả của nó không đoán định được trước.

Chú ý rằng tuy kết quả của phép thử ngẫu nhiên không thể tiên đoán trước, nhưng các kết quả này phải nằm trong một tập giá trị xác định.

Như vậy, nếu gọi  $e$  là kết quả của phép thử ngẫu nhiên,  $\Omega$  là tập hợp các giá trị của phép thử, ta có:

$$e \in \Omega \quad (2.1)$$

Sự kiện ngẫu nhiên (*random event*) là kết quả của một phép thử ngẫu nhiên. Kết quả này tuy không đoán trước được nhưng nằm trong một tập các kết quả đã được xác định.

### 2.1.2. Định nghĩa xác suất

Trong mục này, chúng ta làm việc với các phép thử có kết cục đồng khả năng. Khái niệm đồng khả năng đóng vai trò chủ đạo và khó có thể định nghĩa một cách hình thức. Giả sử trong tổng số  $n$  kết cục đồng khả năng của một phép thử, có đúng  $m$  kết cục thuận lợi cho việc xuất hiện  $A$ . Khi đó xác suất xuất hiện  $A$  sẽ là  $m/n$ .

*Ví dụ 1:* Gieo một lần một con xúc xắc cân đối và đồng chất. Gọi  $A$  là sự kiện mặt trên con xúc xắc có 1 chấm. Vì con xúc xắc cân đối và đồng chất nên khả năng xuất hiện các mặt 1 chấm, 2 chấm..., 6 chấm là như nhau. Vậy số khả năng có thể là 6, nhưng số khả năng thuận lợi cho  $A$  chỉ là 1. Khi đó xác suất xuất hiện  $A$  sẽ là  $1/6$ .

Khái niệm xác suất được định nghĩa như sau:

*Xác suất được định nghĩa là tần suất quan hệ (relative frequency) của việc xuất hiện một sự kiện ngẫu nhiên.*

Nếu gọi  $A$  là sự kiện ngẫu nhiên; sự kiện  $A$  xuất hiện  $n_A$  lần trong một số lượng lớn các sự kiện  $n$ , xác suất của sự kiện  $A$  sẽ được định nghĩa như sau:

$$P(A) \equiv \frac{n_A}{n} \quad (2.2)$$

Dựa vào sự kiện này, có thể suy ra một số tính chất của xác suất như sau:

*Tính chất 1:* Có thể thấy nếu  $A$  không bao giờ xuất hiện thì  $P(A) = 0$ . Mặt khác, nếu  $A$  luôn xuất hiện,  $P(A) = 1$ . Do đó:

$$0 \leq P(A) \leq 1$$

*Tính chất 2:* Nếu gọi  $P(A \cup B)$  là xác suất xuất hiện của sự kiện  $A$  hoặc sự kiện  $B$ . Ta có, trong trường hợp biến  $A$  và  $B$  tương quan:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

Trong đó:  $P(A \cap B)$  là xác suất xuất hiện sự kiện  $A$  và sự kiện  $B$ . Trong trường hợp  $A$  và  $B$  độc lập:

$$P(A \cup B) = P(A) + P(B)$$

*Tính chất 3:* Nếu gọi  $P(A/B)$  là xác suất có điều kiện ( $A$  xuất hiện với điều kiện  $B$  xuất hiện). Trong trường hợp  $A$  và  $B$  tương quan:

$$P(A \cap B) = P(A) \cdot P(B/A) = P(B) \cdot P(A/B)$$

Công thức trên được gọi là công thức Bayes. Mặt khác, nếu  $A$  và  $B$  là hai biến độc lập:

$$P(A \cap B) = P(A) \cdot P(B)$$

## 2.2. BIẾN NGẪU NHIÊN

### 2.2.1. Khái niệm biến ngẫu nhiên

Khái niệm biến ngẫu nhiên là một khái niệm rất quen thuộc trong toán học.

Một cách đơn giản nhất, khái niệm biến ngẫu nhiên được coi như là một đại lượng phụ thuộc vào kết cục của một phép thử ngẫu nhiên nào đó.

*Ví dụ 2:* Vẫn tiếp tục xem xét trò chơi tung một con xúc xắc. Nếu ta gọi biến ngẫu nhiên là số chấm xuất hiện, rõ ràng nó là kết quả của một phép thử ngẫu nhiên và nhận các giá trị nguyên từ 1 đến 6. Trò chơi tung xúc xắc này tính điểm với luật chơi sau:

**Bảng 2.1. Điểm tương ứng với kết quả tung xúc xắc**

<i>Kết quả tung</i>	<i>Điểm</i>
1	200
2	500
3	600
4	1000
5	1200
6	1500

Trong bảng trên, kết quả tung chính là một sự kiện ngẫu nhiên (kết quả của phép thử), điểm là biến ngẫu nhiên tương ứng với kết quả đó.

Một ví dụ nữa là nghiên cứu biến ngẫu nhiên là nhiệt độ của một phản ứng hóa học trong một khoảng thời gian nào đó. Rõ ràng nhiệt độ đó nhận giá trị trong một khoảng xác định.

Như vậy, nếu gọi  $X$  là một biến ngẫu nhiên phụ thuộc vào sự kiện ngẫu nhiên  $e$ ,  $\Omega$  là tập hợp các giá trị của phép thử, ta có:

$$X \equiv X(e), e \in \Omega \quad (2.3)$$

Có hai loại biến ngẫu nhiên là biến ngẫu nhiên rời rạc và biến ngẫu nhiên liên tục. Hai biến này được định nghĩa như sau:

– **Biến ngẫu nhiên rời rạc** (*discrete random variable*) là biến mà tập giá trị của nó là **một tập hữu hạn hoặc vô hạn đếm được các phần tử**. Nói cách khác, miền giá trị của một biến ngẫu nhiên rời rạc là một dãy số và dãy số này có thể hữu hạn hoặc vô hạn.

Ví dụ như điểm thi của một sinh viên (từ 1 đến 10) hay số cuộc gọi của một tổng đài trong một đơn vị thời gian.

– **Biến ngẫu nhiên liên tục** (*continous random variable*) là biến ngẫu nhiên mà **tập giá trị của nó lấp kín một khoảng trên trục số**. Nói cách khác, số phần tử của tập giá trị là vô hạn, không đếm được theo lý thuyết số.

Ví dụ như huyết áp của một bệnh nhân hay tuổi thọ của một loại bóng đèn điện tử.

### 2.2.2. Các hàm phân phối xác suất và bảng phân phối xác suất

Việc xác định một biến ngẫu nhiên bằng tập các giá trị của nó rõ ràng là chưa đủ. Bước tiếp theo là xác định xác suất của từng giá trị. Khi đó chúng ta sẽ có các hàm phân phối xác suất và bảng phân phối xác suất.

Đối với biến ngẫu nhiên rời rạc, mỗi giá trị của nó được gán với một xác suất đặc trưng cho khả năng biến ngẫu nhiên nhận giá trị đó  $p_i = P(X = x_i)$ . Tập hợp các giá trị xác suất tương ứng với các biến ngẫu nhiên khác nhau tạo nên bảng phân phối xác suất của biến ngẫu nhiên đó.

Ví dụ 3: Bảng phân phối xác suất của biến ngẫu nhiên nhận các giá trị của mặt xúc xắc từ 1 đến 6 sẽ là:

**Bảng 2.2. Bảng phân phối xác suất của biến ngẫu nhiên nhận các giá trị của mặt xúc xắc**

X	P(X)
1	1/6
2	1/6
3	1/6
4	1/6
5	1/6
6	1/6

Trong ví dụ trên, biến ngẫu nhiên  $X$  là một biến rời rạc và tương ứng với các giá trị rời rạc đó là các xác suất để  $X$  nhận giá trị rời rạc đó. Chú ý rằng bảng phân phối xác suất này có giá trị đều trên miền từ 1 đến 6. Tuy nhiên trong trường hợp  $X$  là một biến ngẫu nhiên tùy ý hay liên tục thì bảng phân phối xác suất này chưa đủ tổng quát. Lúc này người ta đưa thêm khái niệm các hàm phân phối xác suất như sau:

#### **Hàm phân phối xác suất**

Hàm phân phối xác suất của biến ngẫu nhiên  $X$ , ký hiệu  $F(x)$ , được xác định như sau:

$$F(x) = P(X < x), x \in R \quad (2.4)$$

Từ định nghĩa trên,  $F(x)$  phản ánh **độ tập trung xác suất ở bên phải của số thực  $x$** . Trong trường hợp biến ngẫu nhiên rời rạc, người ta nhận được hàm phân phối tích lũy hay xác suất tích lũy. Đồ thị của hàm phân phối xác suất này là hàm bậc thang. Nếu  $X$  có bao nhiêu giá trị thì  $F(x)$  có bấy nhiêu điểm gián đoạn.

Chúng ta có thể xác định dễ dàng hàm phân phối xác suất của biến ngẫu nhiên  $X$  trong ví dụ 3 như sau:

$$F(x) = \begin{cases} 0 & (x < 1) \\ 1/6 & (1 \leq x < 2) \\ 2/6 & (2 \leq x < 3) \\ 3/6 & (3 \leq x < 4) \\ 4/6 & (4 \leq x < 5) \\ 5/6 & (5 \leq x < 6) \\ 1 & (6 \leq x) \end{cases}$$

Hàm phân phối xác suất có vai trò quan trọng khi nghiên cứu các biến ngẫu nhiên liên tục. Nếu ta biết được hàm phân phối xác suất có nghĩa là xác định được hoàn toàn biến ngẫu nhiên. Tuy nhiên, trong thực tế cũng phải thấy rằng việc tìm được  $F(x)$  là rất khó, nếu không muốn nói là hầu như không thể làm được.

Hàm phân phối xác suất có một số tính chất như sau:

*Tính chất 1:*  $0 \leq F(x) \leq 1$ .

*Tính chất 2:*  $F(x)$  là một hàm đơn điệu tăng, tức là khi  $x_1 < x_2$  thì  $F(x_1) < F(x_2)$ .

*Tính chất 3:*  $P(\alpha \leq X \leq \beta) = F(\beta) - F(\alpha)$ .

### Hàm mật độ xác suất

Hàm phân phối xác suất  $F(x)$  còn một hạn chế là không cho biết rõ phân phối xác suất ở lân cận một điểm nào đó trên trục số. Vì vậy đối với các biến ngẫu nhiên liên tục, có  $F(x)$  khả vi, người ta đưa ra khái niệm hàm mật độ phân phối xác suất như sau:

Hàm **mật độ xác suất** của biến ngẫu nhiên  $X$ , ký hiệu là  $f(x)$ , có hàm phân phối xác suất  $F(x)$  khả vi (trừ ở một số hữu hạn điểm gián đoạn bị chặn), được xác định bằng:

$$F(x) = \int_{-\infty}^{+\infty} f(t) dt \quad (2.5)$$

Hàm mật độ xác suất có hai tính chất cơ bản như sau:

*Tính chất 1:* Hàm mật độ xác suất của tổng hai biến ngẫu nhiên độc lập sẽ là tích chập của hai hàm mật độ xác suất thành phần.



Tức là:

$$f_{X_i+X_j}(x) = \int_{-\infty}^{\infty} f_{X_i}(y)f_{X_j}(x-y)dy$$

Trong đó:  $X_i$  và  $X_j$  là hai biến ngẫu nhiên. Hàm  $f_{X_i+X_j}(x)$ ,  $f_{X_i}(x)$  và  $f_{X_j}(x)$  là các hàm mật độ xác suất tương ứng.

*Tính chất 2:*  $f(x) \geq 0 \forall x$ .

*Tính chất 3:*  $\int_{-\infty}^{+\infty} f(x)dx = 1$ .

### 2.2.3. Các số đặc trưng của biến ngẫu nhiên

Các hàm phân phối xác suất và hàm mật độ xác suất sẽ cho chúng ta thông tin đầy đủ nhất về biến ngẫu nhiên. Tuy nhiên, trong thực tế không phải lúc nào ta cũng có thể xác định được hàm phân phối xác suất và hàm mật độ xác suất này. Từ đó dẫn đến việc tìm một vài đặc trưng quan trọng của các biến ngẫu nhiên, đó là kỳ vọng, phương sai, độ lệch chuẩn, một và trung vị.

#### Kỳ vọng của biến ngẫu nhiên

Kỳ vọng của biến ngẫu nhiên  $X$  ký hiệu là  $E(X)$ , được xác định như sau:

– Nếu  $X$  là biến rời rạc có xác suất  $p(x_i) = p_i, i = 1, 2, \dots$  thì:

$$E(X) = \sum_{\forall i} x_i p_i \quad (2.6)$$

– Nếu  $X$  là biến liên tục có hàm mật độ  $f(x)$ ,  $x \in R$  thì:

$$E(X) = \int_{-\infty}^{+\infty} xf(x)dx \quad (2.7)$$

Từ đó ta thấy kỳ vọng chính là tổng có trọng số của tất cả các giá trị của  $X$ , hay còn gọi là trị trung bình của biến ngẫu nhiên (phân biệt với trung bình cộng của các giá trị). Trên thực tế, nếu quan sát các giá trị của  $X$  nhiều lần và lấy trung bình cộng, thì khi số quan sát càng lớn, số trung bình sẽ càng gần tới kỳ vọng  $E(X)$ . Vì vậy kỳ vọng được gọi là trị trung bình của biến  $X$  mà không sợ nhầm lẫn.

*Ví dụ 4:* Nếu  $X$  là số chấm xuất hiện khi gieo một con xúc xắc thì:

$$E(X) = 1/6 (1 + 2 + 3 + 4 + 5 + 6) = 3,5$$

Như vậy trong trường hợp xác suất được phân phối đều trên tập giá trị, kỳ vọng chính là trung bình cộng của các giá trị ấy. Giá trị  $E(X)$  đạt được là 3,5 còn có nghĩa là nếu gieo nhiều lần, số chấm trung bình thu được sẽ là 3,5.

Một số tính chất của kỳ vọng:

*Tính chất 1:*  $E(c) = c$  với  $c$  là hằng số.

*Tính chất 2:*  $E(cX) = cE(X)$ .

*Tính chất 3:*  $E(X + Y) = E(X) + E(Y)$ .

*Tính chất 4:*  $E(XY) = E(X) \cdot E(Y)$  nếu  $X$  và  $Y$  độc lập với nhau.

Để có thể hiểu được khái niệm kỳ vọng, chúng ta xét một vài ví dụ sau:

*Ví dụ 5:* Giả sử phải tìm kỳ vọng của biến ngẫu nhiên  $X$  khi biết:

$$F_X(x) = \begin{cases} 0 & x \leq 2 \\ a(x-2)^2 & 2 < x \leq 4 \\ 1 & x > 4 \end{cases}$$

Do  $X$  liên tục,  $F_X(x)$  liên tục, nên tại  $x = 4$ ,  $a(4-2)^2 = 1 \Rightarrow a = \frac{1}{4}$ .

Từ đây ta xác định được hàm mật độ xác suất:

$$f_X(x) = \begin{cases} \frac{1}{2} \cdot (x-2) & x \in [2, 4] \\ 0 & x \notin [2, 4] \end{cases}$$

Kỳ vọng của biến ngẫu nhiên  $X$  được xác định:

$$E(X) = \int_{-\infty}^{+\infty} x f_X(x) dx = \int_2^4 \frac{x}{2} \cdot (x-2) dx = \left[ \frac{x^3}{6} - \frac{x^2}{2} \right]_2^4 = \frac{10}{3}$$

*Ví dụ 6:* Tìm kỳ vọng của biến ngẫu nhiên  $X$  khi biết bảng phân phối:

$X$	0	1	2	3
$p(x)$	0,064	0,288	0,432	0,216

Giải:

$$E(X) = \sum_{\forall i} x_i p(x_i) = \sum_{i=0}^3 x_i p(x_i) = 0 * 0,064 + 1 * 0,288 + 2 * 0,432 + 3 * 0,216 = 1,8$$

Ví dụ 7: Gieo đồng thời hai con xúc xắc. Tìm tổng số chấm trung bình.

Lúc này gọi  $X_i$  là số chấm xuất hiện của con xúc xắc thứ  $i$  ( $i=1,2$ ), dễ thấy từ ví dụ 4 ta có  $E(X_1) = E(X_2) = 3,5$ . Mặt khác tổng số chấm của hai con xúc xắc sẽ là  $X_1 + X_2$ . Dựa vào tính chất của kỳ vọng ta có  $E(X_1 + X_2) = 3,5 + 3,5 = 7$ .

### Phương sai của biến ngẫu nhiên

Phương sai của biến ngẫu nhiên  $X$ , ký hiệu là  $V(X)$ , được định nghĩa như sau:

$$V(X) = E[(X - E(X))^2] \quad (2.8)$$

Từ định nghĩa trên ta thấy  $X - E(X)$  chính là độ lệch của biến  $X$  so với trung bình của nó, từ đó có thể thấy phương sai chính là trung bình của bình phương độ lệch đó. Vậy phương sai đặc trưng cho độ phân tán của biến ngẫu nhiên quanh trị trung bình của biến đó. Cũng theo ý nghĩa này, phương sai càng lớn thì độ bất định của biến tương ứng càng lớn.

Trong tính toán, phụ thuộc vào  $X$  là rời rạc hay liên tục, ta có hai công thức tính phương sai như sau:

$$V(X) = \sum_{\forall i} (x_i - E(X))^2 p_i \quad (2.9)$$

$$V(X) = \int_{-\infty}^{+\infty} (x - E(X))^2 f(x) dx \quad (2.10)$$

Tuy nhiên việc tính theo công thức trên khá phức tạp. Để đơn giản hơn, ta có thể sử dụng các tính chất của kỳ vọng để biến đổi định nghĩa của phương sai về dạng tương đương như sau:

$$V(X) = E(X^2) - (E(X))^2$$

Khi đó các phương trình (2.9) và (2.10) sẽ trở thành:

$$V(X) = \sum_{\forall i} x_i^2 p_i - \left( \sum_{\forall i} x_i p_i \right)^2$$

và: 
$$V(X) = \int_{-\infty}^{+\infty} x^2 f(x) dx - \left( \int_{-\infty}^{+\infty} x f(x) dx \right)^2$$

Ví dụ 8: Tìm phương sai của biến ngẫu nhiên  $X$  khi biết bảng phân phối:

x	0	1	2	3
$p(x)$	0,064	0,288	0,432	0,216

Giải:

$$V(X) = E(X^2) - (EX)^2$$

Theo ví dụ 6,  $E(X) = 1,8$

$$E(X^2) = 0^2 * 0,064 + 1^2 * 0,288 + 2^2 * 0,432 + 3^2 * 0,216 = 3,96$$

$$V(X) = E(X^2) - (EX)^2 = 3,96 - 3,24 = 0,72$$

Ví dụ 9: Tìm phương sai của biến ngẫu nhiên  $X$  nếu hàm mật độ xác suất của biến ngẫu nhiên  $X$  tuân theo quy luật phân bố hàm số mũ:

$$f(x) = \begin{cases} 0 & x \leq 0 \\ \lambda \cdot e^{-\lambda x} & x > 0; \quad \lambda > 0 \end{cases}$$

Để tính phương sai của  $X$ , chúng ta có:

$$E(X) = \int_{-\infty}^{+\infty} x f_X(x) dx = \int_0^{\infty} x \cdot \lambda \cdot e^{-\lambda x} dx = \lambda \cdot \frac{1}{\lambda^2} = \frac{1}{\lambda};$$

$$E(X^2) = \int_{-\infty}^{+\infty} x^2 f_X(x) dx = \int_0^{\infty} x^2 \cdot \lambda \cdot e^{-\lambda x} dx = \frac{2}{\lambda^2};$$

$$V(X) = E(X^2) - (EX)^2 = \frac{1}{\lambda^2}.$$

### **Độ lệch chuẩn của biến ngẫu nhiên**

Chúng ta thấy rằng đại lượng  $V(X)$  luôn là một số không âm. Từ định nghĩa ta cũng thấy rằng, về mặt vật lý,  $V(X)$  không cùng thứ nguyên (cùng đơn vị đo) đối với  $X$ . Vì vậy người ta đưa vào khái niệm độ lệch chuẩn của biến ngẫu nhiên.

Độ lệch chuẩn của biến ngẫu nhiên  $X$ , ký hiệu là  $\delta(X)$ , được định nghĩa như sau:

$$\delta(X) = \sqrt{V(X)} \quad (2.11)$$

Độ lệch chuẩn được dùng thường xuyên hơn phương sai do có cùng đơn vị đo với chính biến  $X$ .

**Phương sai và độ lệch chuẩn có một số tính chất như sau:**

*Tính chất 1:*  $Vc = 0$  với  $c$  là hằng số;

*Tính chất 2:*  $V(cX) = c^2 V(X)$ ;  $\delta(cX) = |c| \delta(X)$ ;

*Tính chất 3:* Nếu  $X$  và  $Y$  là độc lập thì:

$$V(X+Y) = V(X) + V(Y); \delta(X+Y) = \sqrt{\delta^2(X) + \delta^2(Y)}.$$

### Hệ số thay đổi của biến ngẫu nhiên

Hệ số thay đổi của biến ngẫu nhiên  $X$ , ký hiệu là  $c(X)$  được xác định bởi:

$$c(X) = \frac{\sigma(X)}{E(X)} = \frac{\sqrt{V(X)}}{E(X)}$$

### Mốt

Mốt là giá trị của biến ngẫu nhiên  $X$  có khả năng xuất hiện lớn nhất trong một lần cận nào đó của nó. Như vậy đối với biến rời rạc, mốt là giá trị của  $X$  ứng với xác suất lớn nhất, còn đối với biến liên tục, mốt là giá trị làm hàm mật độ đạt cực đại. Như vậy mốt chỉ có thể là cực đại địa phương và một biến ngẫu nhiên có thể có một mốt hoặc nhiều mốt.

### Trung vị

Trung vị là giá trị của biến ngẫu nhiên  $X$  chia phân phối thành hai phần có xác suất giống nhau.

## 2.3. CÁC MÔ HÌNH PHÂN BỐ XÁC SUẤT CƠ BẢN

### 2.3.1. Phân bố Bernoulli

Phân bố Bernoulli là phân bố ngẫu nhiên của biến rời rạc. Biến ngẫu nhiên này nhận hai giá trị là 1 với xác suất  $p$  và 0 với xác suất  $(1 - p)$ . Hàm xác suất của phân bố xác suất của nó có dạng:

$$p(x) = p^x(1-p)^{1-x} \text{ với } x = 0 \text{ hoặc } 1 \quad (2.12)$$

Như vậy, giá trị kỳ vọng và phương sai của phân bố Bernoulli được tính như sau:

$$E(X) = p$$

$$V(X) = p(1-p)$$

Ta thấy mọi phép thử chỉ có hai kết cục đều có thể mô hình hóa bằng phân phối này. Trong thực tế, phân phối Bernoulli ít được sử dụng do nó quá đơn giản. Tuy nhiên nó được dùng làm cơ sở để tìm luật phân phối của các biến ngẫu nhiên khác.

### 2.3.2. Phân bố nhị thức

Đây là một phân phối rất hay dùng trong thống kê hiện đại. Phân bố nhị thức được định nghĩa như sau:

Một biến ngẫu nhiên  $X$  có phân bố nhị thức bậc  $n$ , nếu nó có những giá trị  $0, 1, 2, \dots, n$  với xác suất:

$$p(x) = C_n^x p^x q^{n-x} \quad (2.13)$$

Trong đó:  $0 < p < 1$ ;  $p + q = 1$  và  $x = 0, 1, 2, \dots, n$ .

Kỳ vọng, phương sai và hệ số thay đổi của biến ngẫu nhiên  $X$  tuân theo phân bố nhị thức được xác định:

$$E(X) = n \cdot p;$$

$$V(X) = n \cdot p \cdot q = n \cdot p \cdot (1-p);$$

$$c(X) = \frac{\sigma(X)}{E(X)} = \sqrt{\frac{(1-p)}{n \cdot p}}.$$

Cần phải nhắc lại các điều kiện để có phân phối nhị thức là các phép thử giống nhau và độc lập. Trong mỗi phép thử chỉ có hai kết cục là có và không.

### 2.3.3. Phân bố đều (Uniform distribution)

Một biến ngẫu nhiên  $X$  tuân theo phân bố đều nếu nó có hàm mật độ xác suất là hằng số trong khoảng  $[a, b]$ :

$$f(x) = \begin{cases} \frac{1}{b-a} & x \in [a, b] \\ 0 & x \notin [a, b] \end{cases} \quad (2.14)$$

Nếu  $a = 0$ ,  $b = 1$ , chúng ta có **phân bố đều đơn vị** (*unit uniform distribution*):

$$f(x) = \begin{cases} 1 & x \in [0, 1] \\ 0 & x \notin [0, 1] \end{cases}$$

Tương ứng, chúng ta có hàm phân phối xác suất:

$$F(x) = \frac{x-a}{b-a}; \quad a \leq x \leq b$$

Kỳ vọng, phương sai và hệ số thay đổi của biến ngẫu nhiên  $X$  tuân theo phân bố đều được xác định:

$$E(X) = \frac{a+b}{2};$$

$$V(X) = \frac{(b-a)^2}{12};$$

$$c(X) = \frac{\sigma(X)}{E(X)} = \sqrt{\frac{(b-a)^2}{3(a+b)^2}}.$$

#### 2.3.4. **Phân bố chuẩn** (Gaussian distribution)

Đây là phân phối liên tục quan trọng và có ứng dụng rộng rãi nhất. Một biến ngẫu nhiên  $X$  tuân theo phân bố chuẩn (hay còn được gọi là phân bố Gauss) nếu hàm mật độ xác suất có dạng:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-a)^2}{2\sigma^2}} \quad (2.15)$$

Phân bố chuẩn này còn được ký hiệu là  $N(a, \sigma^2)$ . Kỳ vọng, phương sai và hệ số thay đổi của biến ngẫu nhiên  $X$  tuân theo phân bố chuẩn được xác định:

$$E(X) = \mu;$$

$$V(X) = \sigma^2;$$

$$c(X) = \frac{\sigma(X)}{E(X)} = \frac{\sigma}{\mu}.$$

Để thấy hai tham số  $a$  và  $\mu^2$  là hai số đặc trưng quan trọng  $E(X)$  và  $V(X)$  của biến  $X$ .

### 2.3.5. Phân bố mũ (*Exponential distribution*)

Phân bố mũ là một phân bố ngẫu nhiên của một biến ngẫu nhiên liên tục với hàm mật độ xác suất:

$$f(x) = \lambda e^{-\lambda x} \quad (2.16)$$

Ngoài ra, hàm phân bố xác suất, kỳ vọng, phương sai và hệ số thay đổi của của biến ngẫu nhiên  $x$  tuân theo phân bố mũ được tính như sau:

$$F(x) = 1 - e^{-\lambda x};$$

$$E(X) = \frac{1}{\lambda};$$

$$V(X) = \frac{1}{\lambda^2};$$

$$c(X) = \frac{\sigma(X)}{E(X)} = 1.$$

Trong đó:  $\lambda > 0$  là tham số tốc độ (*rate parameter*).

Phân bố hàm số mũ khá quan trọng và thường được dùng phổ biến trong lý thuyết hàng đợi để mô tả các phân bố ngẫu nhiên xảy ra trong các hệ thống thông tin, ví dụ như sự phân bố của khoảng thời gian đến giữa các cuộc gọi cũng như thời gian phục vụ của hệ thống tổng đài điện thoại.

Một tính chất quan trọng của biến ngẫu nhiên  $X$  tuân theo phân bố hàm số mũ là thuộc tính không nhớ (*memoryless*):

$$P(X \leq u + t \mid X > u) = 1 - e^{-\lambda t} = P(X \leq t)$$

Phương trình trên cho thấy hàm phân bố xác suất trong một khoảng  $[u; u + t]$  không phụ thuộc vào giá trị khởi đầu  $u$  mà chỉ phụ thuộc vào độ rộng của khoảng giá trị  $t$ .



### 2.3.6. Phân bố Poisson (Poisson distribution)

Một biến ngẫu nhiên rời rạc  $X$  có phân bố Poisson với tham số  $\lambda$  ( $\lambda > 0$ ) nếu nó lấy những giá trị  $0, 1, 2, \dots, n$  với xác suất:

$$p(x) = \frac{\lambda^x e^{-\lambda}}{x!} \text{ với } x = 0, 1, \dots \quad (2.17)$$

Kỳ vọng, phương sai và hệ số thay đổi của biến ngẫu nhiên  $X$  tuân theo phân bố Poisson được xác định:

$$E(X) = \lambda$$

$$V(X) = \lambda$$

$$c(X) = \frac{\sigma(X)}{E(X)} = \sqrt{\frac{1}{\lambda}}$$

### 2.3.7. Phân bố Gamma (Gamma distribution)

Phân bố Gamma là phân bố ngẫu nhiên của một biến  $x$  liên tục với hàm mật độ xác suất:

$$f(x) = \frac{\lambda}{\Gamma(x)} e^{-\lambda x} x^{x-1} \text{ với } r > 0, \lambda > 0, x > 0 \quad (2.18)$$

với  $\Gamma(x)$  là hàm xác định bởi:

$$\Gamma(x) = \int_0^{\infty} t^{x-1} e^{-t} dt \text{ với } x > 0$$

Ngoài ra, kỳ vọng và phương sai của phân bố Gamma như sau:

$$E(X) = \frac{r}{\lambda}; \quad V(X) = \frac{r}{\lambda^2}$$

Trong đó:  $\lambda$  và  $r$  là hai tham số của phân bố Gamma.  $\lambda > 0$  được gọi là tham số tỷ lệ (*scale parameter*),  $n$  nguyên dương được gọi là tham số hình dạng (*shape parameter*).

### 2.3.8. Phân phối $\chi^2$ với $n$ bậc tự do và phân bố Student

Phân bố này ký hiệu là  $\chi^2(n)$  có hàm mật độ xác suất như sau:

$$f(x) = \frac{x^{\frac{n}{2}-1} e^{-\frac{x}{2}}}{2^{\frac{n}{2}} \Gamma\left(\frac{n}{2}\right)} \text{ với } x > 0, n > 0$$

Trong đó hàm Gamma đã được định nghĩa ở trên.

Hàm phân phối  $\chi^2(n)$  là cơ sở để tính toán hàm phân phối Student với  $n$  bậc tự do, ký hiệu là  $t(n)$  như sau:

Cho  $X$  và  $Y$  là hai biến ngẫu nhiên độc lập tuân theo luật phân bố  $N(0,1)$  và  $\chi^2(n)$  tương ứng. Khi đó biến:

$$T_n = \frac{X}{\sqrt{Y/n}}$$

sẽ có phân bố  $t(n)$ . Hàm mật độ xác suất của phân bố này có dạng như sau:

$$\frac{1}{\sqrt{\pi n}} \frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)} \left(1 + \frac{x^2}{n}\right)^{-\frac{n+1}{2}}$$

Các tham số đặc trưng của phân bố  $t(n)$  là:

$$E(T_n) = 0, \quad V(T_n) = \frac{n}{n-2}$$

### 2.3.9. Phân bố Beta

Phân bố Beta là phân bố được sử dụng để mô tả các biến ngẫu nhiên bị chặn, ví dụ như giữa hai mức 0 và 1. Phân bố Beta có hai tham số là  $\alpha$  và  $\lambda$  và có hàm mật độ xác suất được xác định bởi hàm sau:

$$f(x) = \frac{\Gamma(\alpha + \lambda)}{\Gamma(\alpha)\Gamma(\lambda)} x^{\alpha-1} (1-x)^{\lambda-1} \quad (2.19)$$

Phân bố này được ký hiệu là  $\beta(\alpha, \lambda)$ . Các số đặc trưng của phân bố Beta là:

$$E(X) = \frac{\alpha}{\alpha + \lambda}$$

$$V(X) = \frac{\alpha\lambda}{(\alpha + \lambda)^2(\alpha + \lambda + 1)}$$

## 2.4. TIẾN TRÌNH NGẪU NHIÊN (STOCHASTIC PROCESS)

### 2.4.1. Định nghĩa tiến trình ngẫu nhiên

Trong các hệ thống truyền thông nói chung, các tín hiệu của nó như tín hiệu thoại, tín hiệu truyền hình, dữ liệu máy tính số và tín hiệu nhiễu thường là các tín hiệu ngẫu nhiên. Có thể kể ra hai đặc tính của các tín hiệu ngẫu nhiên này như sau:

- Các tín hiệu này đều là những hàm biến đổi theo thời gian được xác định trong một số khoảng thời gian quan trắc.
- Các tín hiệu này là ngẫu nhiên và không thể mô tả được chính xác dạng sóng của tín hiệu quan sát được.

Người ta định nghĩa tiến trình ngẫu nhiên như sau:

Giả sử ta có  $X$  là một biến ngẫu nhiên thay đổi theo thời gian. Tập hợp các biến ngẫu nhiên  $\{X(t)\}$  thay đổi theo thời gian  $t$  ( $t \geq 0$ ) được gọi là một tiến trình ngẫu nhiên. Kết quả là 1 hàm thời gian

Ta có thể viết:  $X \in \{X(t)\}$ . Tiến trình ngẫu nhiên này có kết quả phụ thuộc vào kết quả của phép thử ngẫu nhiên  $e$  và phụ thuộc cả vào thời gian  $t$ .

### 2.4.2. Phân loại các tiến trình ngẫu nhiên

Giữa biến ngẫu nhiên và tiến trình ngẫu nhiên có một sự khác nhau cơ bản. Sự khác nhau cơ bản này được thể hiện như sau: với biến ngẫu nhiên, kết quả phép thử của thí nghiệm ngẫu nhiên được ánh xạ thành một giá trị số, còn với tiến trình ngẫu nhiên, kết quả của phép thử được ánh xạ thành một hàm thời gian.

Để có thể phân loại các tiến trình ngẫu nhiên, người ta sẽ căn cứ vào tính liên tục hoặc rời rạc của không gian mẫu  $S$  và  $t$ . Như vậy sẽ có thể được phân chia thành bốn loại khác nhau:

**Chuỗi ngẫu nhiên rời rạc (Discrete Random Sequence):** nếu cả  $S$  và  $t$  là những giá trị rời rạc. Ví dụ nếu  $X_n$  đại diện cho kết quả thứ  $n$  của việc tung một con xúc xắc thì  $\{X_n, n \geq 1\}$  là một chuỗi ngẫu nhiên rời rạc vì  $t = \{1, 2, 3, \dots\}$  và  $S = \{1, 2, 3, 4, 5, 6\}$ .

**Chuỗi ngẫu nhiên liên tục (Continuous Random Sequence):** nếu  $t$  là rời rạc còn  $S$  là liên tục. Ví dụ nếu  $X_n$  đại diện cho nhiệt độ tại giờ thứ  $n$  trong ngày thì  $\{X_n, 1 \leq n \leq 24\}$  là một chuỗi ngẫu nhiên liên tục vì nhiệt độ có thể là bất kỳ giá trị nào trong một khoảng giá trị xác định và do vậy liên tục.

**Tiến trình ngẫu nhiên rời rạc: (Discrete Random Process)** nếu  $t$  là liên tục và  $S$  là rời rạc. Ví dụ nếu  $X(t)$  đại diện cho số cuộc gọi điện thoại nhận được trong khoảng thời gian  $(0, t)$  thì  $\{X(t)\}$  là tiến trình ngẫu nhiên rời rạc vì  $S = \{1, 2, 3, 4, 5, 6\}$ .

**Tiến trình ngẫu nhiên liên tục (Continuous Random Process):** nếu cả  $t$  và  $S$  đều là liên tục. Ví dụ nếu  $X(t)$  đại diện cho nhiệt độ cao nhất tại một địa điểm nào đó trong khoảng thời gian  $(0, t)$ , thì  $\{X(t)\}$  là tiến trình ngẫu nhiên liên tục.

Nói một cách tổng quát, từ rời rạc hoặc liên tục liên quan đến  $S$ , còn chuỗi hoặc tiến trình liên quan đến  $t$ .

Ngoài ra, các tiến trình ngẫu nhiên còn được phân biệt thành: tiến trình ngẫu nhiên xác định và tiến trình ngẫu nhiên không xác định:

- **Tiến trình ngẫu nhiên không xác định:** Nếu giá trị của bất kỳ hàm mẫu nào không được dự đoán chính xác từ các giá trị quan trắc được trong quá khứ thì tiến trình ngẫu nhiên đó được gọi là không xác định.
- **Tiến trình ngẫu nhiên xác định:** Một tiến trình ngẫu nhiên được gọi là xác định nếu giá trị tương lai của bất kỳ một hàm mẫu nào cũng đều có thể dự đoán được từ các giá trị trong quá khứ. Ví dụ:  $X(t) = A \cdot \cos(\omega_0 t + \phi)$ .

### 2.4.3. Các tiến trình ngẫu nhiên thường gặp

#### 2.4.3.1. Tiến trình đếm (Counting Process)

Tiến trình đếm được định nghĩa như sau:

Một tiến trình ngẫu nhiên  $\{N(t), t \geq 0\}$  được gọi là tiến trình đếm nếu  $N(t)$  đại diện cho tổng số các “sự kiện” xuất hiện cho tới thời điểm  $t$ . Nói cách khác, một tiến trình đếm phải thỏa mãn các điều kiện sau:

Điều kiện 1:  $N(0) = 0$ ;

Điều kiện 2:  $N(t) \geq 0$ ;

Điều kiện 3:  $N(t)$  là số nguyên;

Điều kiện 4: Nếu  $s < t$  thì  $N(s) \leq N(t)$ ;

Điều kiện 5: Đối với  $s < t$  thì  $N(t) - N(s)$  chính là số sự kiện xuất hiện trong khoảng thời gian  $(s, t)$ .

Ví dụ 10: Tại một ngã tư, khi tiến hành đếm số phương tiện giao thông đi qua và giả sử ở thời điểm bắt đầu đếm  $t = 0$  không có một phương tiện nào hay  $N(t) = 0$ . Khi đó số phương tiện đếm được luôn luôn là một số nguyên thỏa mãn điều kiện 4 và 5, tức là  $N(t) - N(s)$  chính là số phương tiện xuất hiện trong khoảng thời gian từ  $s$  đến  $t$ .

#### 2.4.3.2. Tiến trình tăng trưởng độc lập (Independent Increment Process)

Tiến trình tăng trưởng độc lập được định nghĩa như sau:

Một tiến trình ngẫu nhiên  $\{X(t), t \geq 0\}$  được gọi là tiến trình tăng trưởng độc lập (Independent Increment Process) khi các sự kiện xảy ra trong các khoảng thời gian khác nhau là độc lập với nhau.

Xét các khoảng thời gian  $(s_1, t_1); (s_2, t_2); \dots; (s_n, t_n)$ . Nếu các sự kiện xảy ra trong các khoảng thời gian này:  $[X(t_1) - X(s_1)]; [X(t_2) - X(s_2)]; \dots; [X(t_n) - X(s_n)]$  là độc lập với nhau thì tiến trình này được gọi là tiến trình tăng trưởng độc lập.

#### 2.4.3.3. Tiến trình tăng trưởng dừng (Stationary Increment Process)

Tiến trình tăng trưởng dừng được định nghĩa như sau:

Một tiến trình ngẫu nhiên  $\{X(t), t \geq 0\}$  được gọi là tăng trưởng dừng (Stationary Increment Process) khi sự phân bố của sự kiện  $\{X(t+h) - X(t)\}$  trong khoảng thời gian  $\{t, t+h\}$  giống như sự phân bố các sự kiện  $\{X(h) - X(0)\}$  trong khoảng thời gian  $\{0, h\}$ .

Nói cách khác, tiến trình đếm là tăng trưởng dừng nếu như việc phân bố số các sự kiện xuất hiện trong bất kỳ khoảng thời gian nào chỉ phụ thuộc vào độ dài của khoảng thời gian quan trắc mà không phụ thuộc vào thời điểm bắt đầu tiến hành quan trắc.

#### 2.4.3.4. Tiến trình **Bernoulli** (Bernoulli Process)

Giả sử  $X$  là một biến ngẫu nhiên tuân theo **phân bố Bernoulli**. Khi đó tập hợp các giá trị của biến ngẫu nhiên  $X$  này được gọi là tiến trình Bernoulli.

#### 2.4.3.5. Tiến trình **nhị phân** (Binomial Process)

Dãy  $\{S_n; n = 0, 1, 2, \dots\}$  được gọi là tiến trình nhị phân khi thỏa mãn điều kiện sau:

$$\begin{cases} S_0 = X_0 \\ S_1 = X_0 + X_1 \\ \dots \\ S_n = X_0 + X_1 + \dots + X_n \end{cases}$$

Trong đó  $X_i$  với  $i = 0, 1, 2, \dots, n$  là biến Bernoulli.

#### 2.4.3.6. Tiến trình **Poisson** (Poisson Process)

Để có thể hiểu được tiến trình Poisson, trước hết chúng ta tìm hiểu khái niệm hàm nhỏ  $o(h)$ . Hàm này được định nghĩa như sau:

Hàm  $f$  được gọi là **hàm nhỏ  $o(h)$**  nếu nó thỏa mãn điều kiện sau:

$$\lim_{h \rightarrow 0} \frac{f(h)}{h} = 0$$

Sau đây chúng ta sẽ xem xét một số tính chất của hàm nhỏ này:

**Tính chất 1:** Hàm  $f(x) = x$  không phải là hàm  $o(h)$  vì:

$$\lim_{h \rightarrow 0} \frac{f(h)}{h} = \lim_{h \rightarrow 0} \frac{h}{h} = 1 \neq 0$$

**Tính chất 2:** Hàm  $f(x) = x^2$  là hàm  $o(h)$  vì:

$$\lim_{h \rightarrow 0} \frac{f(h)}{h} = \lim_{h \rightarrow 0} \frac{h^2}{h} = \lim_{h \rightarrow 0} h = 0$$

**Tính chất 3:** Hàm  $f(x) = x^r$  với  $r > 1$  là hàm  $o(h)$  vì:

$$\lim_{h \rightarrow 0} \frac{f(h)}{h} = \lim_{h \rightarrow 0} h^{r-1} = 0$$

**Tính chất 4:** Khi  $f$  và  $g$  là hàm  $o(h)$  thì  $f+g$  cũng là hàm  $o(h)$ , vì:

$$\lim_{h \rightarrow 0} \frac{f(h)+g(h)}{h} = \lim_{h \rightarrow 0} \frac{f(h)}{h} + \lim_{h \rightarrow 0} \frac{g(h)}{h} = 0$$

**Tính chất 5:** Khi  $f$  là hàm  $o(h)$  và  $c$  là hằng số, thì  $cf$  cũng là hàm  $o(h)$ , vì:

$$\lim_{h \rightarrow 0} \frac{cf(h)}{h} = c \lim_{h \rightarrow 0} \frac{f(h)}{h} = c \cdot 0 = 0$$

Một **tiến trình đếm**  $\{N(t), t \geq 0\}$  được gọi là **tiến trình Poisson** với tham số  $\lambda > 0$  khi các điều kiện sau được thỏa mãn:

**Điều kiện 1:** Các sự kiện xảy ra trong các khoảng thời gian khác nhau là độc lập với nhau.

**Điều kiện 2:**  $N(t)$  là tiến trình **tăng trưởng dừng**.

**Điều kiện 3:** Xác suất để có đúng một sự kiện xảy ra trong một khoảng thời gian  $h$  bất kỳ là  **$\lambda h + o(h)$** . Có nghĩa là:

$$p(N(h)=1) = \lambda h + o(h)$$

**Điều kiện 4:** Xác suất để có hai sự kiện trở lên xảy ra trong một khoảng thời gian  $h$  bất kỳ là  **$o(h)$** . Có nghĩa là:

$$p(N(h) \geq 2) = o(h)$$

Từ các điều kiện 3 và 4 người ta nhận thấy rằng **xác suất để không có sự kiện nào xảy ra** trong khoảng thời gian  $h$  là:  $1 - \lambda h - 2 * o(h) = 1 - \lambda h + o(h)$ .

**khi  $h \rightarrow 0$**

Từ định nghĩa trên của tiến trình Poisson, chúng ta có định lý sau:

**Định lý 2.1:** Cho  $\{N(t), t \geq 0\}$  là một tiến trình Poisson với **tốc độ**  $\lambda > 0$ . Lúc này biến ngẫu nhiên  $N(t)$  trong đó  $N(t)$  là số sự kiện xảy ra trong khoảng thời gian  $t$ , sẽ tuân theo luật phân bố Poisson với tham số  $\lambda t$ , có nghĩa là:

$$p(N(t)=n) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} \text{ với } n = 0, 1, 2, \dots$$

Định lý này được chứng minh như sau. Theo định nghĩa của tiến trình Poisson, số sự kiện xảy ra trong khoảng thời gian  $t$  sẽ **không phụ thuộc** vào **điểm bắt đầu** của khoảng thời gian đó mà **chỉ phụ thuộc vào độ**

dài của  $t$ . Vì vậy chúng ta có thể coi  $N(t)$  là số sự kiện xảy ra trong khoảng thời gian từ 0 đến  $t$ . Tiếp theo người ta gọi xác suất để có  $n$  sự kiện xảy ra trong khoảng thời gian  $t$  là:

$$p_n(t) = p\{N(t) = n\} \text{ với } n = 0, 1, 2, \dots$$

Trước hết ta tính  $p_0(t)$  là xác suất để không có sự kiện nào xảy ra trong khoảng thời gian từ 0 đến  $t$ . Xét khoảng thời gian từ  $[0, t, t+h]$ . Một cách hiển nhiên, ta thấy rằng xác suất để không có sự kiện nào xảy ra trong khoảng từ  $[0, t+h]$  sẽ là xác suất để không có sự kiện nào xảy ra trong khoảng  $[0, t]$  và không có sự kiện nào xảy ra trong khoảng thời gian từ  $[t, t+h]$ . Do đó chúng ta có phương trình sau:

vì là tiến trình tăng trưởng độc lập nên nhân đc

$$p_0(t+h) = p_0(t)p_0(t, t+h) = p_0(t)p_0(h) = p_0(t)[1 - \lambda h + o(h)]$$

Từ phương trình trên chúng ta có:

$$\frac{p_0(t+h) - p_0(t)}{h} = -\lambda p_0(t) + \frac{o(h)}{h}$$

Khi cho  $h \rightarrow 0$ , chúng ta nhận được:

$$\frac{dp_0(t)}{dt} = -\lambda p_0(t)$$

Mặt khác chúng ta có  $p_0(0) = p(N(0) = 0) = 1$ .

Từ hai phương trình trên, chúng ta nhận được:  $p_0(t) = e^{-\lambda t}$ .

Tiếp theo chúng ta sẽ tính  $p_1(t)$  là xác suất để có một sự kiện xảy ra trong khoảng thời gian  $[0, t]$ . Xét khoảng thời gian từ  $[0, t, t+h]$ . Để có một sự kiện xảy ra trong khoảng thời gian này thì có hai khả năng sau:

- Có một sự kiện xảy ra trong khoảng thời gian từ  $[0, t)$  và không có sự kiện nào xảy ra trong khoảng thời gian từ  $[t, t+h]$ ;
- Không có sự kiện nào xảy ra trong khoảng thời gian từ  $[0, t)$  và một sự kiện xảy ra trong khoảng thời gian  $[t, t+h]$ .



Khi đó ta có thể viết: ko sự kiện nào xảy ra  
trong khoảng từ  $(t \rightarrow t+h)$

$$p_1(t+h) = p_1(t)(1 - \lambda h + o(h)) + p_0(t)(\lambda h + o(h))$$

Tức là: 1 sự kiện xảy ra  
trong khoảng  $0 \rightarrow t$

$$\frac{p_1(t+h) - p_1(t)}{h} = -\lambda p_1(t) + \lambda p_0(t) + \frac{o(h)}{h}$$

Khi cho  $h \rightarrow 0$ , chúng ta nhận được:

$$\frac{dp_1(t)}{dt} = -\lambda p_1(t) + \lambda p_0(t)$$

Tiếp theo chúng ta sẽ xét trường hợp  $n \geq 2$ . Để có  $n$  sự kiện xảy ra trong khoảng thời gian  $[0, t, t+h]$  thì:

– Có  $n$  sự kiện trong khoảng  $(0, t)$  và không có sự kiện nào trong khoảng thời gian  $[t, t+h]$ ;

– Có  $n-1$  sự kiện xảy ra trong khoảng  $(0, t)$  và có một sự kiện xảy ra trong khoảng  $[t, t+h]$ ;

– Có  $n-k$  sự kiện xảy ra trong khoảng  $(0, t)$  và  $k$  sự kiện xảy ra trong khoảng  $[t, t+h]$  khi  $k \geq 2$ .

Tương tự như tính toán ở trên, chúng ta nhận được:

$$p_n(t+h) = p_n(t)(1 - \lambda h + o(h)) + p_{n-1}(t)(\lambda h + o(h)) + p_{n-k}(t)o(h)$$

Do đó:

$$\frac{p_n(t+h) - p_n(t)}{h} = -\lambda p_n(t) + \lambda p_{n-1}(t) + \frac{o(h)}{h}$$

Khi cho  $h \rightarrow 0$ , chúng ta nhận được:

$$\frac{dp_n(t)}{dt} = -\lambda p_n(t) + \lambda p_{n-1}(t) \quad \text{giải phương trình vi phân}$$

Sau khi giải các phương trình vi phân ở trên với các điều kiện bờ cho trước, người ta nhận được:

$$p_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} \quad \text{với } n = 0, 1, 2, \dots \quad \text{công thức tổng quan}$$

**Định lý 2.2:** Gọi  $\{N(t); t \geq 0\}$  là một tiến trình Poisson với tham số  $\lambda$ . Gọi  $0 < t_1 < t_2 < t_3 \dots$  là thời điểm xảy ra các sự kiện. Định nghĩa dãy  $T_n$  Tn: khoảng thời gian giữa các thời điểm xảy ra sự kiện như sau:  $T_1 = t_1, T_2 = t_2 - t_1, \dots, T_k = t_k - t_{k-1}$ . Khi đó dãy  $T_n$  cũng là dãy độc lập, tuân theo phân bố mũ với tham số  $\lambda$ . Tức là:

$$F(T_n) = p(T_n \leq s) = 1 - e^{-\lambda s} \text{ với } s \geq 0$$

Trong đó:  $p(T_n \leq s)$  là xác suất để  $T_i$  nhỏ hơn giá trị  $s$  cho trước.

Định lý này có thể được chứng minh một cách đơn giản như sau. Vì tiến trình Poisson là tiến trình tăng trưởng độc lập nên dãy  $T_1, T_2, \dots$  cũng là các biến ngẫu nhiên độc lập với nhau.

Để tính giá trị  $p(T_n \leq s)$  chúng ta sẽ tính  $p(T_n \geq s)$ .

Để thời điểm xảy ra sự kiện  $n$  là  $T_n \geq s$ , chúng ta sẽ có:  $T_n = t_n - t_{n-1}$

$$p(T_n > s) = p(N(t_{n-1} + s) - N(t_{n-1}) = 0) = p(N(s) = 0) = p_0(s) = e^{-\lambda s}$$

Từ đó chúng ta sẽ có:

$$p(T_n \leq s) = 1 - e^{-\lambda s} \text{ với } s \geq 0$$

xác suất sự kiện thứ  $n$  xảy ra ngoài thời gian  $s$  chính là xác suất để không có sự kiện nào xảy ra trong khoảng thời gian  $s$

## BÀI TẬP CHƯƠNG 2

1. Tung đồng thời hai đồng xu. Tính xác suất  $p$  để nhận được kết quả một mặt sấp và một mặt ngửa?
2. Kết quả thi môn Đánh giá hiệu năng Mạng và Kỹ thuật truyền thông nhận được như sau:
  - 5% số sinh viên trượt môn Đánh giá hiệu năng Mạng
  - 7% số sinh viên trượt môn Kỹ thuật truyền thông.
  - 4% số sinh viên trượt cả hai môn.Giả sử chọn ngẫu nhiên một sinh viên. Hãy tính xác suất để:
  - a. Sinh viên đó trượt môn Đánh giá hiệu năng Mạng, khi biết rằng sinh viên đó trượt môn Kỹ thuật truyền thông?
  - b. Sinh viên đó trượt môn Kỹ thuật truyền thông, khi biết rằng sinh viên đó trượt môn Đánh giá hiệu năng Mạng?
  - c. Sinh viên đó trượt môn Đánh giá hiệu năng mạng hoặc môn Kỹ thuật truyền thông?
3. Cho tập hợp dãy 10, 20, 40, 80, 90. Hãy tính giá trị trung bình và phương sai của dãy này?
4. Một gói có độ dài  $n$  bit truyền qua một kênh truyền có nhiễu. Xác suất  $p$  để một bit bị lỗi là  $p = 0,01$ . Xác suất có lỗi của các bit không phụ thuộc vào nhau. Hãy tính:
  - a. Xác suất để một gói có  $k$  bit lỗi với  $0 < k < n$ ?
  - b. Xác suất để gói có lỗi phụ thuộc vào  $n$ ?
  - c. Trung bình phải truyền một gói bao nhiêu lần để gói này không có lỗi?

## TÀI LIỆU THAM KHẢO

1. M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions*. Dover Publications, 1965.
2. R. N. Bhattacharya and E.C. Waymire. *Stochastic Process with Applications*. Wiley, New York, 1990.
3. S. Kotz and N.L. Johnson. *Encyclopedia of Statistical Sciences, Vol. 8*, 836ff, John Wiley, New York, 1988.
4. S. M. Ross. *A First Course in Probability*. Macmillan, 4th edition, 1994.
5. H. Richter. *Wahrscheinlichkeitstheorie*. Springer Verlag, 2. Auflage, 1966.
6. William Feller. *An Introduction to Probability Theory and Its Applications—Volume I*. John Wiley, New York, third edition, 1968.

chuyển mạch gói (mạng internet) hiệu suất cao hơn chuyển mạch kênh (của điện thoại di động)

chuyển mạch gói : hiệu suất cao , dễ sử dụng, dễ phát triển => đc ưa chuộng hơn

càng đơn giản càng hiệu quả càng tốt

## Chương 3

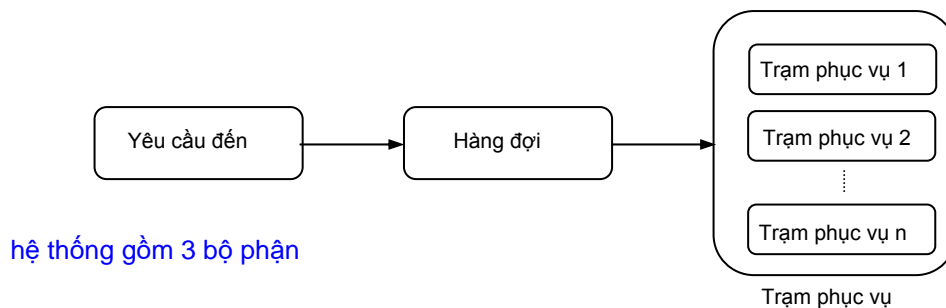
# HỆ THỐNG HÀNG ĐỢI

### 3.1. GIỚI THIỆU

Trong phần trên, chúng ta đã khảo sát các tiến trình ngẫu nhiên thông dụng. Đây chính là các công cụ toán học thường được sử dụng trong hệ thống hàng đợi được đề cập trong chương này.

Có thể nói rằng mô hình hàng đợi là một công cụ không thể thiếu và có ứng dụng rất rộng rãi trong nhiều lĩnh vực của cuộc sống, đặc biệt trong lĩnh vực công nghệ thông tin và truyền thông. **Hàng đợi** là thành phần không thể thiếu khi mô hình hóa hệ thống sử dụng phương pháp toán học và là công cụ hữu hiệu để đánh giá hiệu năng hoạt động của các hệ thống phục vụ nói chung cũng như của hệ thống máy tính và thông tin nói riêng.

**Hệ thống phục vụ** là hệ thống được sử dụng để phục vụ các yêu cầu nhất định hoặc các khách hàng thông qua các điểm cung cấp dịch vụ. Các yêu cầu hoặc khách hàng này sẽ rời hệ thống sau khi dịch vụ đã được cung cấp.



**Hình 3.1. Mô hình chung của một hệ thống hàng đợi**

Một cách đơn giản nhất, mô hình hàng đợi sẽ có **trung tâm phục vụ** và **tập hợp các khách hàng**. Xuất phát từ tập hợp này, các khách hàng sẽ tiến vào trung tâm phục vụ và sau khi được phục vụ xong sẽ đi ra khỏi mô hình

VD về hệ thống hàng đợi : đi siêu thị có quầy tính tiền => hệ thống các hàng đợi 45

mỗi 1 bàn tính tiền là 1 hàng đợi nhỏ

VD khác : xếp hàng lên máy bay

sau khi qua hàng đợi => đc phục vụ

hàng đợi này. Thông thường khả năng xử lý của các trung tâm phục vụ là có hạn, do đó sẽ xảy ra trường hợp các khách hàng muốn được phục vụ phải xếp hàng để chờ đến lượt. Nói cách khác, một mô hình hàng đợi sẽ bao gồm ba thành phần chính: **tập hợp khách hàng, trung tâm phục vụ và hàng đợi.**

Một ví dụ dễ nhận thấy nhất của mô hình hàng đợi là các trung tâm check-in tại các sân bay. Trước khi được phép lên máy bay, khách hàng phải tiến hành xếp hàng chờ đợi vì thông thường chỉ có một trạm soát trước cửa mỗi máy bay và rõ ràng khả năng xử lý của trạm soát này là có hạn. Mỗi khi có một khách hàng mới đến, họ sẽ phải tiếp tục xếp vào vị trí cuối cùng của hàng đợi. Đây chính là một ví dụ của hàng đợi theo kiểu First In First Out (FIFO).

Trong các hệ thống máy tính và truyền thông, có thể kể ra rất nhiều các ví dụ của mô hình hàng đợi khác nhau. Trong mạng điện thoại chẳng hạn, các hệ thống điện thoại sẽ phải đáp ứng số lượng lớn khách hàng quay số để kết nối đến một trong những đường ra hữu hạn của tổng đài. Trong mạng máy tính: khi gói tin được chuyển từ nguồn tới đích và đi qua một số lượng các nút trung gian. Hệ thống hàng đợi **xuất hiện tại mỗi nút mạng ở quá trình lưu tạm thông tin tại bộ đệm.** Trong các hệ thống máy tính: khi các công việc tính toán và tuyển làm việc của hệ thống yêu cầu dịch vụ từ bộ xử lý trung tâm và từ các nguồn khác.

Tất cả các hệ thống trong ví dụ trên đều được mô tả bằng khái niệm hàng đợi.

Lý thuyết hàng đợi sẽ sử dụng các công cụ toán học để trả lời các câu hỏi như: **tính toán thời gian chờ trung bình trong hàng đợi** và trong mô hình hàng đợi, **thời gian đáp ứng trung bình của toàn bộ mô hình, mức độ sử dụng của trung tâm phục vụ khách hàng...** Thông thường các thông số này được tính toán trong bối cảnh ngẫu nhiên, tức là việc đến của các khách hàng được coi là ngẫu nhiên.

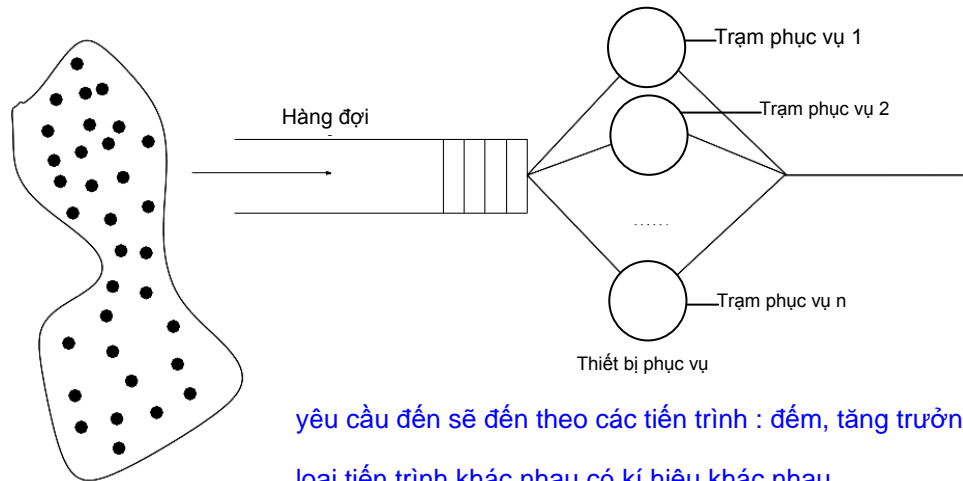
với 1 hệ thống hàng đợi thường sử dụng kí hiệu kendall để biểu diễn, để mô hình hóa hệ thống hàng đợi

## **3.2. MÔ HÌNH HÀNG ĐỢI – KÝ HIỆU KENDALL**

### **3.2.1. Mô hình hàng đợi đơn**

Mô hình hàng đợi được miêu tả một cách đơn giản như hình vẽ 3.2. Các khách hàng sẽ đến mô hình hàng đợi này theo một cách ngẫu nhiên.

Trung tâm phục vụ có thể có một hoặc nhiều trạm phục vụ và mỗi trạm phục vụ chỉ có thể phục vụ được một người tại một thời điểm. Khoảng thời gian mà mỗi khách hàng phải chờ đợi trong trung tâm phục vụ cũng là một biến ngẫu nhiên.



**Hình 3.2. Mô hình hàng đợi đơn**

Nói cách khác, người ta giả thiết một hàng đợi đơn giản như sau:

**Tiến trình tới** (*arrival process*): được đặc trưng bởi **tốc độ yêu cầu** (hoặc sự kiện)  $\lambda$  tới hệ thống (yêu cầu/đơn vị thời gian) và **tính chất ngẫu nhiên của tiến trình tới**.

Ví dụ các gói tin được gửi tới đầu vào của một bộ định tuyến là một tiến trình tới, được đặc trưng bởi **số gói/s** và phân bố ngẫu nhiên của các gói, tức là phân bố ngẫu nhiên của khoảng thời gian giữa hai gói liên tiếp, ví dụ như **phân bố mũ**.

**Tiến trình phục vụ** (*service process*): **Tiến trình phục vụ** gắn liền với các trạm phục vụ (hay server) trong một hệ thống phục vụ. Mỗi trạm phục vụ bao giờ cũng được đặc trưng bởi **tốc độ phục vụ trung bình  $\mu$**  (số yêu cầu/một đơn vị thời gian) và **tính chất ngẫu nhiên của tiến trình phục vụ**.

Ví dụ các gói tin được gửi ra đầu ra của một bộ định tuyến được coi như một tiến trình phục vụ và được đặc trưng bởi số gói được phục vụ trong 1 s.

Thông thường các gói tin trong Internet có độ dài thay đổi, vì vậy thời gian phục vụ một gói tin trung bình (thời gian để gửi gói tin đó từ bit đầu tiên đến bit cuối cùng lên đường truyền) sẽ có tính chất ngẫu nhiên tuân theo một phân bố nào đó.

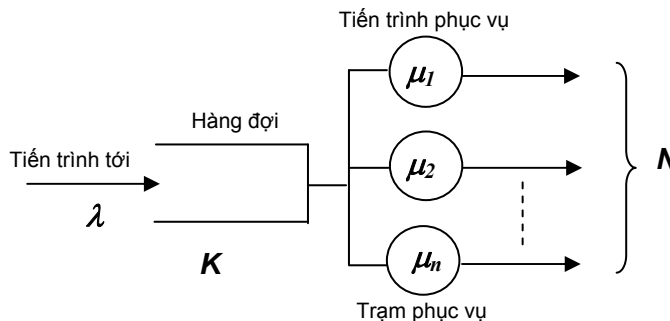
– **Số trạm phục vụ (server)**: Một hệ thống phục vụ có thể có nhiều trạm phục vụ. Trong trường hợp này hệ thống phục vụ có thể phục vụ một lúc nhiều yêu cầu.

– **Quy tắc phục vụ (service strategy)**: thể hiện nguyên tắc và trình tự phục vụ các yêu cầu khi chúng đi vào hệ thống phục vụ.

Ví dụ: Trong nguyên tắc FIFO, yêu cầu nào vào hệ thống trước thì được phục vụ trước. Trong nguyên tắc LIFO, yêu cầu nào vào hệ thống sau cùng lại được phục vụ đầu tiên.

– **Hàng đợi (queue)**: Khi một yêu cầu đi vào một hệ thống phục vụ, nếu tất cả các trạm phục vụ đều bận thì yêu cầu đó sẽ phải đợi trong hàng đợi. Hàng đợi được đặc trưng bởi độ lớn  $K$ .

Hình 3.3 minh họa một hệ thống hàng đợi với  $N$  trạm phục vụ, độ lớn hàng đợi  $K$ , tiến trình tới với tham số  $\lambda$  và  $N$  tiến trình phục vụ với tham số  $\mu_1, \mu_2, \dots, \mu_n$ .



**Hình 3.3. Hệ thống hàng đợi với  $N$  trạm phục vụ, hàng đợi có độ lớn  $K$**

### 3.2.2. Ký hiệu Kendall

Ký hiệu Kendall được sử dụng để mô tả hệ thống hàng đợi đơn, do David George Kendall phát triển. Hệ thống hàng đợi được mô tả bằng một chuỗi ký tự như sau:

$$A/B/m/N-S$$

48

tiến trình tới tuân theo kí tự khác nhau thì khác nhau

chữ cái thứ 1: tiến trình tới , tuân theo phân bố gì , ...

chữ cái thứ 2 : tiến trình phục vụ, tuân theo phân bố j, có thể là poison, mũ, ngẫu nhiên,...

chữ cái thứ 3 : cho biết hệ thống hàng đợi có bao nhiêu trạm phục vụ

chữ cái thứ 4 : kích thước của hàng đợi là bao nhiêu, nếu bỏ qua thì nó là vô cùng. nếu là n thì khi anh n+1 đến sẽ ko có chỗ để chờ



Trong đó:

- $A$ : thể hiện phân bố của tiến trình tới;
- $B$ : thể hiện phân bố của tiến trình phục vụ.

Đối với  $A$  và  $B$ , người ta sử dụng các chữ viết tắt để thể hiện các phân bố khác nhau như sau:

- $M$  (Markov): là ký hiệu của phân bố mũ với  $A(t)$ : Hàm phân bố xác suất,  $a(t)$ : hàm mật độ  $A(t) = 1 - e^{-\lambda t}$  và  $a(t) = \lambda e^{-\lambda t}$  với  $\lambda > 0$ . nên quan tâm đến  $M$ , nó tuân theo phân bố poisson (người đến ở tứ rời rạc)
- $D$  (Deterministic): là ký hiệu của phân bố xác định, trong đó các giá trị của phân bố này là không đổi.
- $E_k$  (Erlang- $k$ ): là phân bố Erlang với  $k$  giai đoạn ( $k \geq 1$ ). Đối với phân bố Erlang, chúng ta có:

$$A(t) = 1 - e^{-k\mu t} \sum_{j=0}^{k-1} \frac{(k\mu t)^j}{j!} \text{ với } \mu > 0$$

Trong đó tham số  $\mu > 0$ . Phân bố này đặc biệt thích hợp trong trường hợp phải mô hình hóa các cuộc gọi trong mạng điện thoại đến các trung tâm xử lý tại các tổng đài.

- $H_k$  (Hyper- $k$ ): là phân phối siêu mũ (hyperexponential) với  $k$  giai đoạn. Tiến trình đến của phân bố này được xác định như sau:

$$A(t) = \sum_{j=1}^k q_j (1 - e^{-\mu_j t})$$

Trong đó  $\mu_i > 0, q_i > 0, i \in \{1..k\}$  có kèm theo điều kiện  $\sum_{j=1}^k q_j = 1$ .

- $G$  (General): luật phân bố chung.

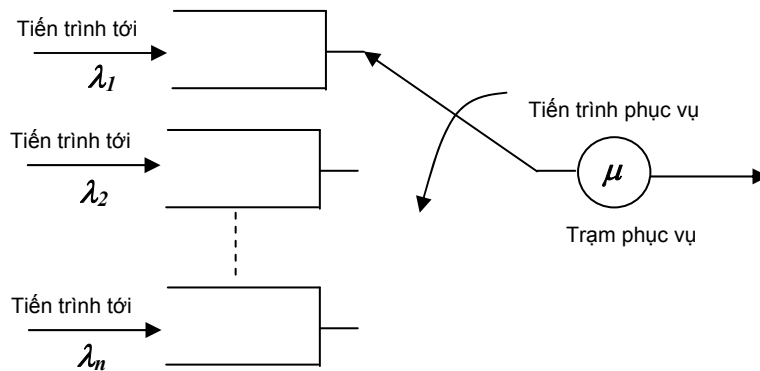
Các ký hiệu tiếp theo có ý nghĩa sau:

- $m$ : số trạm phục vụ.
- $N$ : là kích thước tối đa của hàng đợi trong hệ thống hàng đợi. Nếu  $N$  bị bỏ qua tức là  $N$  có kích thước vô hạn.
- $S$  (service discipline): thể hiện cách các trung tâm phục vụ khách hàng như thế nào. Khi  $S$  bị bỏ qua tức là hệ thống hàng đợi này phục vụ

### quan tâm đến FIFO và LIFO

theo kiểu FIFO. Có năm cách hay gặp nhất là FIFO, LIFO, RR, SIRO và PQ. Ý nghĩa của các cách này như sau:

- **FIFO** (First In First Out): Yêu cầu nào vào hệ thống trước được phục vụ trước.
- **LIFO** (Last In First Out): Yêu cầu nào vào hệ thống sau cùng được phục vụ đầu tiên.
- **RR** (Round Robin): Hệ thống RR có nhiều hàng đợi cho nhiều tiến trình tới khác nhau. Các yêu cầu của các tiến trình tới được phục vụ theo thứ tự lần lượt. Xem hình 3.4.



**Hình 3.4. Round robin**

- **SIRO** (Service In Random Order): Các yêu cầu được phục vụ theo trình tự ngẫu nhiên, không phụ thuộc vào trình tự tới hệ thống.
- **PQ** (Priority Queue): Hàng đợi có ưu tiên. Giống như RR, hệ thống này có nhiều hàng đợi cho các tiến trình tới khác nhau. Tuy nhiên thứ tự phục vụ các yêu cầu trong các hàng đợi tuân theo trình tự ưu tiên. Trạm phục vụ sẽ phục vụ các yêu cầu trong hàng đợi có mức ưu tiên cao nhất. Khi hàng đợi này rỗng, trạm phục vụ sẽ chuyển xuống phục vụ hàng đợi có yêu cầu ở mức ưu tiên thấp hơn. Khi một hàng đợi bất kỳ ở mức ưu tiên cao hơn có yêu cầu, trạm phục vụ sẽ chuyển sang phục vụ yêu cầu ở hàng đợi có mức ưu tiên cao.

Ví dụ như khi xét hệ thống hàng đợi **M/M/1** là hệ thống có một trạm phục vụ, hàng đợi của hệ thống này có kích thước vô hạn, tiến trình đến và tiến trình

phục vụ của hệ thống hàng đợi này tuân theo luật phân bố mũ với các tham số  $\lambda$  và  $\mu$ .

Ví dụ tiếp theo, chúng ta xét hệ thống hàng đợi  $M/M/2/K$  là hệ thống có tiến trình tới theo phân bố mũ. Tiến trình phục vụ theo phân bố mũ. Hệ thống có hai trạm phục vụ. Hàng đợi có độ lớn là  $K$ . Quy tắc phục vụ: FIFO (ngầm định).

### 3.2.3. Các tham số quan trọng để đánh giá đặc tính của hệ thống hàng đợi

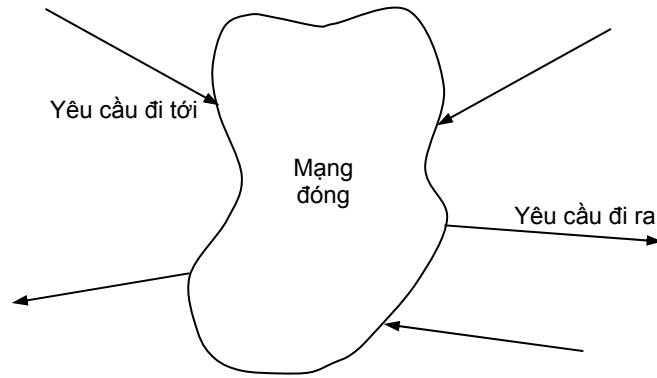
Khi đánh giá đặc tính của hệ thống hàng đợi, người ta thường hay chú ý đến các tham số sau:

- Số yêu cầu trung bình nằm trong hệ thống  $N$ .
- Số yêu cầu trung bình nằm trong hàng đợi  $N_q$ .
- Số yêu cầu trung bình được xử lý ở trạm phục vụ  $N_s$ . Mỗi quan hệ và ý nghĩa của ba tham số trên sẽ được đề cập kỹ hơn ở các phần tiếp theo.
- Thời gian trung bình một yêu cầu lưu lại trong hệ thống  $T$ .
- Thời gian trung bình một yêu cầu phải chờ trong hàng đợi trước khi được phục vụ  $T_q$ .
- Thời gian phục vụ trung bình một yêu cầu tại trạm phục vụ  $T_s$ .
- Xác suất để thời gian một yêu cầu phải đợi trong hàng đợi  $t_q$  nhỏ hơn một khoảng thời gian  $t$  cho trước  $p(t_q < t_s)$ .
- Xác suất để hệ thống có  $i$  yêu cầu  $p_i$ .
- Xác suất để một yêu cầu bị từ chối  $e$ , tức xác suất yêu cầu không được phép đi vào hệ thống do hàng đợi bị đầy.

### 3.2.4. Hệ thống đóng

Xét một mạng hàng đợi ở trạng thái tĩnh. Trạng thái tĩnh này chỉ đạt được khi hệ thống đã hoạt động được sau một thời gian khá dài và trở nên ổn định, nghĩa là phân bố của khách hàng trong hệ thống sẽ không thay đổi. Lúc này ta có một mạng đóng.

Hay nói cách khác, khi các yêu cầu trong một mạng không được tự tạo ra cũng không bị hủy và tính chất của nó vẫn giữ nguyên, thì lúc đó ta có một mạng đóng.



**Hình 3.5. Mạng đóng**

Từ nhận xét trên, người ta thấy rằng khi hệ thống đạt đến trạng thái đóng thì lúc này nó sẽ không tự tạo ra yêu cầu và cũng không hủy yêu cầu. Mặt khác yêu cầu chỉ có thể đi vào hệ thống, lưu trữ trong hệ thống và sau đó đi ra. Nếu tổng yêu cầu trung bình đi vào mạng lớn hơn tổng yêu cầu đi ra khỏi mạng thì tổng số yêu cầu nằm trong mạng sẽ không ngừng tăng lên. Mặt khác, nếu tốc độ tới nhỏ hơn tốc độ đi ra thì tổng yêu cầu trong mạng sẽ giảm về 0. Tại thời điểm đó, tốc độ tới và tốc độ ra sẽ cân bằng nhau.

Qua đó người ta có thể kết luận rằng: khi một hệ thống ở trạng thái ổn định thì tổng tốc độ tới trung bình và tổng tốc độ ra trung bình bằng nhau. Một tính chất quan trọng của hệ thống ở trạng thái ổn định là tính chất thống kê của mạng tại một thời điểm cho trước sẽ tương đương với tính chất thống kê của mạng tại một thời điểm bất kỳ.

Thông thường khi xác định đặc tính của một hệ thống hàng đợi ở trạng thái ổn định, người ta quan tâm đến các tham số sau:

- $N(t)$ : số yêu cầu nằm trong hệ thống tại thời điểm  $t$ ;
- $N_q(t)$ : số yêu cầu nằm trong hàng đợi tại thời điểm  $t$ ;
- $N_s(t)$ : số yêu cầu đang được phục vụ;
- $T$ : thời gian tổng cộng một yêu cầu nằm trong hệ thống hàng đợi;

$T_q$

- $T_s$ : thời gian một yêu cầu phải nằm chờ trong hàng đợi trước khi đến lượt nó được phục vụ;
- $T_s$ : thời gian một yêu cầu được phục vụ bởi trạm phục vụ.

Ta có thể dễ dàng suy luận:

$$N(t) = N_s(t) + N_q(t) \quad (3.1)$$

$$T = T_q + T_s \quad (3.2)$$

Mặt khác, nếu gọi:

- $\lambda$  là tốc độ trung bình của yêu cầu đi tới hệ thống (số yêu cầu/đơn vị thời gian);
- $\mu$  là tốc độ phục vụ trung bình của trạm phục vụ (số yêu cầu/đơn vị thời gian);
- $c$  là số trạm phục vụ của hệ thống hàng đợi.

Ta định nghĩa **tải của hệ thống** bằng biểu thức:

$$\rho = \frac{\lambda}{c\mu} \quad (3.3)$$

$c\mu$  -> số yêu cầu / đơn vị thời gian của toàn hệ thống -> tốc độ phục vụ trung bình của toàn hệ thống hàng đợi

$\rho$  còn được gọi là **mật độ lưu lượng** của hệ thống (*traffic intensity*) và có ý nghĩa là phần thời gian mà người quan sát bên ngoài bắt gặp tối thiểu một trạm phục vụ ở trạng thái bận (đang phục vụ yêu cầu).

Ngoài ra, có thể hiểu  $c\mu$  là tốc độ phục vụ trung bình của toàn hệ thống hàng đợi. Từ định nghĩa chúng ta thấy rằng:

$$0 \leq \rho \leq 1 \quad (3.4)$$

### 3.2.5. Định lý Little

Định lý Little là định lý có thể áp dụng cho các hàng đợi dạng  $G/G/1$ . Định lý này cho phép thiết lập mối quan hệ giữa số khách hàng đến trung bình của hệ thống, tốc độ đến trung bình và thời gian lưu lại hệ thống trung bình (thời gian đáp ứng trung bình là khoảng thời gian từ lúc yêu cầu bắt đầu đi vào hệ thống cho đến lúc đã được phục vụ xong và đi ra khỏi hệ thống).

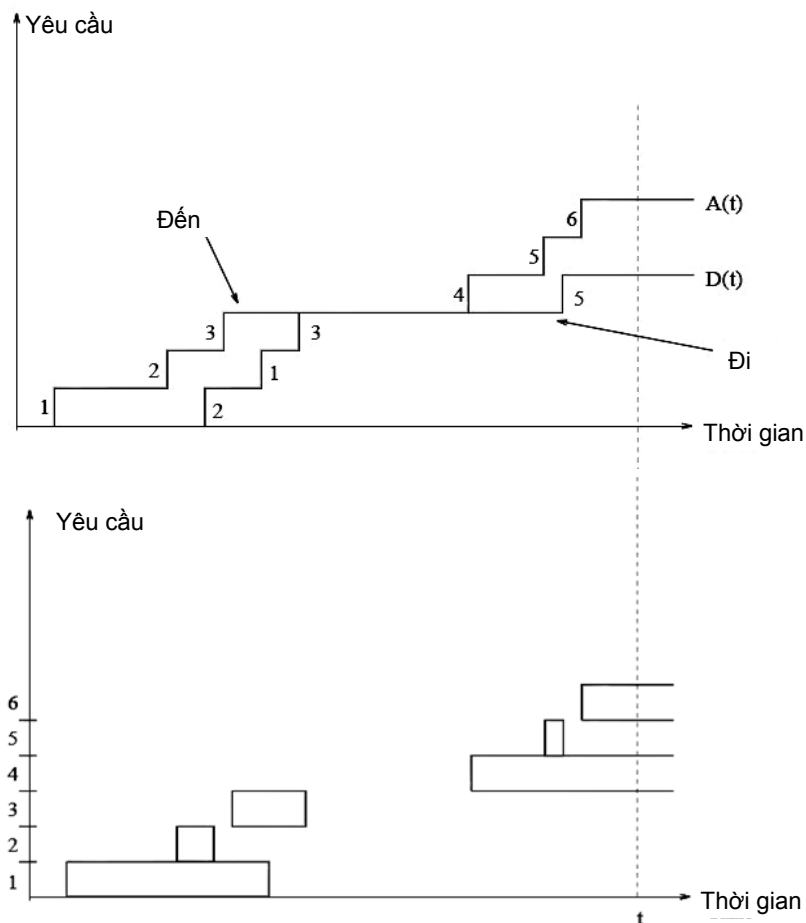
Xét một mạng đóng bất kỳ với các tham số sau:

- $N(t)$  : số yêu cầu nằm trong hệ thống tại thời điểm  $t$ ;
- $A(t)$  : số yêu cầu đi đến hệ thống trong khoảng thời gian từ  $[0, t]$ ;
- $D(t)$  : số yêu cầu rời khỏi hệ thống trong khoảng thời gian từ  $[0, t]$ ;
- $T_i$  : thời gian lưu lại hệ thống của khách hàng thứ  $i$ .

Vậy hiển nhiên ta sẽ có:

$$N(t) = A(t) - D(t)$$

Hình 3.6 dưới đây thể hiện sự biến thiên của các đường  $A(t)$  và  $D(t)$ . Lưu ý rằng thời điểm các yêu cầu rời khỏi hệ thống không nhất thiết phải giống với thời điểm các yêu cầu đến hệ thống.



**Hình 3.6. Chứng minh định lý Little**

Số yêu cầu đến hệ thống trung bình trong khoảng thời gian  $[0, t]$  sẽ là:

$$\bar{A}(t) = \frac{A(t)}{t}$$

Chúng ta giả sử rằng giới hạn sau tồn tại và là một số hữu hạn, tức là:

$$\lambda = \lim_{t \rightarrow \infty} \bar{A}(t)$$

thì lúc này giá trị  $\lambda$  chính là tốc độ đến trung bình. Ngoài ra, thời gian lưu lại trung bình của các yêu cầu trong hệ thống sẽ là:

$$\bar{N}(t) = \frac{1}{t} \int_0^t N(u) du$$

và chúng ta cũng giả sử rằng giới hạn sau tồn tại và là một số hữu hạn, tức là:

$$\bar{N} = \lim_{t \rightarrow \infty} \bar{N}(t)$$

Tương tự như vậy, chúng ta định nghĩa thời gian đáp ứng yêu cầu trung bình sẽ là:

$$\bar{T}(t) = \frac{1}{A(t)} \sum_{i=1}^{A(t)} T_i$$

Hình vẽ 3.6 chỉ ra mối quan hệ giữa số yêu cầu đến và số yêu cầu đi trung bình trên cùng một đồ thị. Bởi vì ta luôn có  $A(t) \geq D(t)$  nên ta luôn có  $N(t) \geq 0$  và phần diện tích nằm giữa hai đường sẽ được tính như sau:

$$F(t) = \int_0^t (A(u) - D(u)) du = \int_0^t N(u) du$$

Dựa trên hình 3.6, chúng ta thấy rằng  $F(t)$  thể hiện tổng số thời gian lưu lại của các yêu cầu trong hệ thống đến thời điểm  $t$  ( $\sum_{i=1}^{A(t)} T_i$ ) trừ đi tổng thời gian đáp ứng của các yêu cầu trong hệ thống cũng đến thời điểm  $t_1 > t$  đó:

$$F(t) = \sum_{i=1}^{A(t)} T_i - E(t)$$

Với giả thiết rằng  $E(t)$  tương đối nhỏ.

Từ đó chúng ta có:

$$\int_0^t N(u) du = \sum_{i=1}^{A(t)} T_i - E(t)$$

Phương trình trên có thể viết lại thành:

$$\frac{1}{t} \int_0^t N(u) du = \frac{A(t)}{t} \frac{1}{A(t)} \sum_{i=1}^{A(t)} T_i - \frac{E(t)}{t}$$

Lưu ý rằng  $t \rightarrow \infty$ , chúng ta nhận được công thức của định lý Little như sau vì  $\lim_{t \rightarrow \infty} E(t) = 0$ :

$$N = \lambda T$$

Định lý Little được phát biểu như sau:

Số yêu cầu trung bình nằm trong hệ thống bằng tích của tốc độ tới trung bình với thời gian lưu lại hệ thống trung bình của yêu cầu:

$$N = \lambda T \quad (3.5)$$

Định lý Little đúng cho một mạng đóng bất kỳ, không phụ thuộc vào các tham số hàng đợi, tiến trình đến và tiến trình phục vụ.

### 3.2.6. Một số đặc tính khác của hệ thống đóng, hoạt động ở trạng thái ổn định

Xét một hệ thống hàng đợi được đặc trưng bởi các tham số sau:

- Tiến trình tới với tốc độ trung bình  $\lambda$ .
- $c$  trạm phục vụ với tốc độ phục vụ trung bình của từng trạm là  $\mu$ .
- Hàng đợi có độ lớn vô tận (yêu cầu luôn được chấp nhận).

Nếu xem xét hệ thống hàng đợi như một mạng đóng bao gồm hàng đợi và các trạm phục vụ thì theo định lý Little ta có:

$$N = \lambda T$$



Mặt khác, nếu xem xét riêng khối hàng đợi của hệ thống hàng đợi trên, căn cứ vào định nghĩa, ta thấy đây vẫn là một mạng đóng, vì vậy có thể áp dụng định lý Little cho riêng khối hàng đợi như sau:

$$N_q = \lambda T_q$$

Mặt khác:  $T = T_q + T_s$

với :  $T_s = \frac{1}{c\mu}$  thì ta có:

$$\frac{N}{\lambda} = \frac{1}{c\mu} + \frac{N_q}{\lambda}$$

Hay:

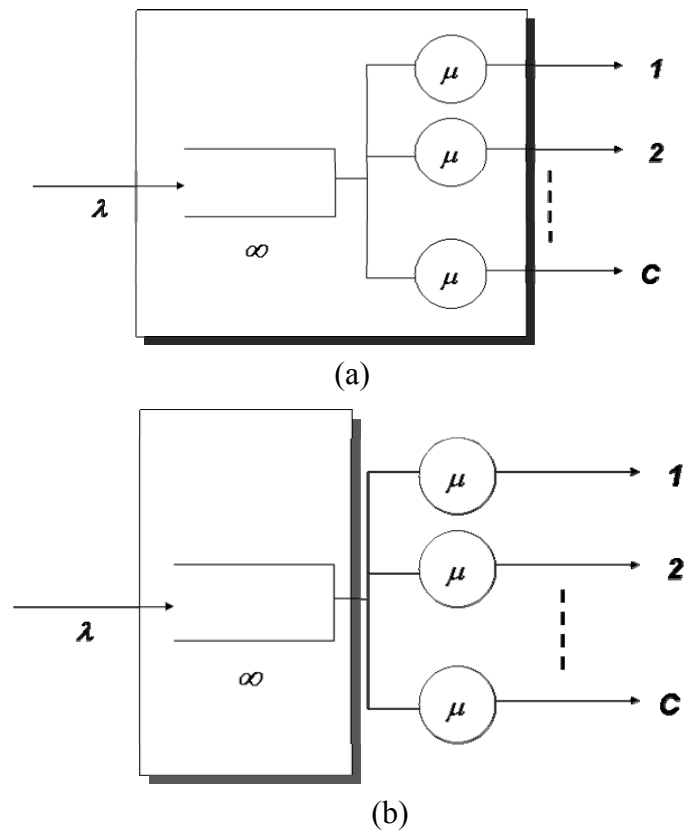
$$N = \frac{\lambda}{c\mu} + N_q$$

Ngoài ra ta có  $\rho = \frac{\lambda}{c\mu}$  nên phương trình trên trở thành:

$$N = \rho + N_q$$

Như vậy ta có thể nói rằng  $\rho$  chính là số yêu cầu trung bình mà hệ thống phục vụ phải xử lý tại các trạm phục vụ.

Chúng ta thấy rằng định lý Little và các đặc tính của hệ thống đóng mà chúng ta xét đến thời điểm này không phụ thuộc vào các đặc điểm riêng biệt của từng hệ thống hàng đợi như tiến trình đến, tiến trình phục vụ, số trạm phục vụ... Tuy nhiên nếu muốn khảo sát các đặc tính khác của hệ thống hàng đợi, chúng ta cần phải biết các đặc tính thống kê của luồng lưu lượng đầu vào cũng như tiến trình xử lý yêu cầu.



**Hình 3.7. Hệ thống hàng đợi được quan sát với hai ranh giới khác nhau**

(a) Hàng đợi và các trạm phục vụ; (b) Hàng đợi.

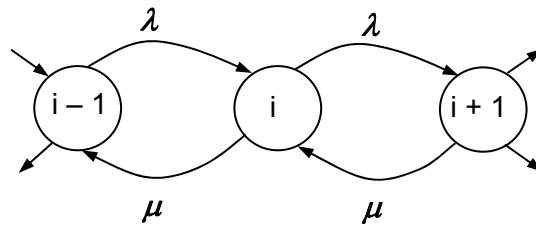
### 3.3. CÁC MÔ HÌNH HÀNG ĐỢI

Kể từ phần này, chúng ta lần lượt khảo sát đặc tính hoạt động của các hệ thống hàng đợi từ đơn giản đến phức tạp, với các đặc tính thống kê cho trước của tiến trình đến và tiến trình phục vụ. Trước khi đi vào phân tích các hệ thống hàng đợi khác nhau, trước hết chúng ta tìm hiểu khái niệm tiến trình sinh tử (*birth-death process*).

#### 3.3.1. Tiến trình sinh tử

Ta gọi một hệ thống hàng đợi **đang ở trạng thái  $i$**  nếu trong hệ thống **đang có  $i$  yêu cầu**. Người ta có nhận xét sau:

Nếu hệ thống hàng đợi có tiến trình tới và tiến trình phục vụ đều tuân theo phân bố mũ thì hệ thống sẽ có **tính chất cân bằng xác suất**, nghĩa là xác suất hệ thống chuyển từ trạng thái  $i$  sang trạng thái  $(i + 1)$  hoặc  $(i - 1)$  sẽ bằng tổng xác suất hệ thống chuyển từ trạng thái  $(i + 1)$  hoặc  $(i - 1)$  về trạng thái  $i$ .



**Hình 3.8. Cân bằng xác suất tại trạng thái  $i$ . Tiến trình sinh – tử**

Nhận xét trên có thể được biểu diễn như sau:

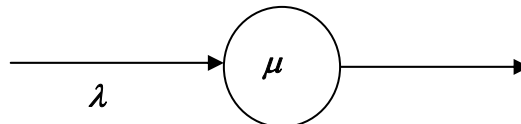
$$(\lambda + \mu)p_i = \lambda p_{i-1} + \mu p_{i+1} \quad \text{lamda: tốc độ tới trung bình} \quad (3.6)$$

Muy: tốc độ phục vụ trung bình của 1 trạm

Trong đó  $p_j$  là xác suất để hệ thống có  $j$  yêu cầu.

### 3.3.2. Hệ thống hàng đợi **$M/M/1/0$**

Ở trên, chúng ta đã nhận xét rằng điều kiện để khảo sát đặc tính một hệ thống hàng đợi là chúng phải hoạt động ở chế độ tĩnh (hay ổn định), nghĩa là các tính chất thống kê của hệ thống không phụ thuộc vào thời gian.



**Hình 3.9. Hàng đợi  $M/M/1/0$**

Hệ thống hàng đợi  $M/M/1/0$  được đặc trưng bởi:

- Số yêu cầu đến tới tuân theo tiến trình Poisson với tham số là  $\lambda$ .
- Thời gian phục vụ tuân theo phân bố mũ với tham số là  $\mu$ .
- **Một trạm phục vụ.**
- **Không có hàng đợi, như vậy hệ thống có thể chứa tối đa một yêu cầu.**
- Phương thức phục vụ là FIFO.

Tiến trình tới là tiến trình poisson tham số lamda  
tiến trình phục vụ là tiến trình poisson tham số muy

*a. Tính xác suất trong hệ thống*

Gọi  $N(t)$  là số yêu cầu trong hệ thống tại thời điểm  $t$ . Do chỉ có hai khả năng là hệ thống không có yêu cầu, hoặc trạm phục vụ đang xử lý một yêu cầu nên  $N(t) \in \{0,1\}$ .

Gọi:

–  $p_n^i(t_1, t_2)$  là xác suất để trong khoảng thời gian  $(t_1, t_2)$  có  $n$  yêu cầu đi vào hệ thống.

–  $p_k^o(t_1, t_2)$  là xác suất để trong khoảng thời gian  $(t_1, t_2)$  có  $k$  yêu cầu đi ra khỏi hệ thống.

Nhiệm vụ đầu tiên đặt ra là phải tính xác suất  $p\{N(t)=0\} \equiv p_0(t)$  và  $p\{N(t)=1\} \equiv p_1(t)$ . Muốn vậy ta xét khoảng thời gian  $(0, t+dt)$  như hình vẽ sau:



**Hình 3.10. Khoảng thời gian xét**

Xét trường hợp  $N(t) = 0$ . Trước hết phải tính  $p\{N(t+dt) = 0\}$ . Ta nhận thấy rằng:

Nếu (Tại thời điểm  $t+dt$  hệ thống không có yêu cầu nào) thì:

[(Tại thời điểm  $t$  hệ thống không có yêu cầu nào) và (Trong khoảng thời gian  $(t, t+dt)$  không có yêu cầu nào đến hệ thống)].

Hoặc:

[(Tại thời điểm  $t$  có một yêu cầu trong hệ thống) và (Trong khoảng thời gian  $(t, t+dt)$  có một yêu cầu rời khỏi hệ thống)].

Hoặc:

[(Tại thời điểm  $t$  hệ thống không có yêu cầu nào) và (Trong khoảng thời gian  $(t, t+dt)$  có một yêu cầu đến hệ thống) và (Trong khoảng thời gian  $(t, t+dt)$  có một yêu cầu rời khỏi hệ thống)].

Chúng ta có thể biểu diễn trường hợp trên bằng biểu thức toán học như sau:

$$p_0(t+dt) = p_0(t)p_0^i(t, t+dt) + p_1(t)p_1^0(t, t+dt) + p_0(t)p_1^i(t, t+dt)p_1^0(t, t+dt)$$

Theo định nghĩa của phân bố Markov (hay Poisson), ta có:

$$p_0^i(t, t+dt) = 1 - \lambda dt$$

$$p_1^0(t, t+dt) = \mu dt$$

$$p_1^i(t, t+dt) = \lambda dt$$

Thay vào phương trình trên, chúng ta có:

$$p_0(t+dt) = p_0(t)(1 - \lambda dt) + p_1(t)\mu dt + p_0(t)\lambda dt\mu dt$$

Số hạng thứ ba của vế phải trong phương trình trên bằng 0 do  $(dt)^2$  tiến tới 0 nhanh hơn  $dt$ . Do đó ta có:

$$p_0(t+dt) = p_0(t)(1 - \lambda dt) + p_1(t)\mu dt$$

tức là:

$$\frac{p_0(t+dt) - p_0(t)}{dt} = -p_0(t)\lambda + p_1(t)\mu$$

Cuối cùng có:

$$\frac{dp_0(t)}{dt} = -p_0(t)\lambda + p_1(t)\mu$$

Nhận xét rằng hệ thống đang xét là **một hệ thống dừng**, không phụ thuộc vào thời gian nên:

$$\frac{dp_0(t)}{dt} = 0$$

Do đó chúng ta nhận được:

$$p_0\lambda = p_1\mu$$

Mặt khác:

$$p_0 + p_1 = 1$$

Giải hệ hai phương trình trên, ta có:

$$\begin{cases} p_0 = \frac{\mu}{\mu + \lambda} \\ p_1 = \frac{\lambda}{\mu + \lambda} \end{cases}$$

Phương trình  $p_0\lambda = p_1\mu$  có thể nhận được một cách rất dễ dàng từ tiến trình sinh tử. Vì hệ thống chỉ có hai trạng thái, nên dựa vào phương trình cân bằng xác suất với hai trạng thái 0 và 1, chúng ta có:

$$p_0\lambda = p_1\mu$$

p0: xác suất để hệ thống có 0 yêu cầu  
p1: xác suất để hệ thống có 1 yêu cầu

b. Tính các **tham số hiệu năng** của hệ thống

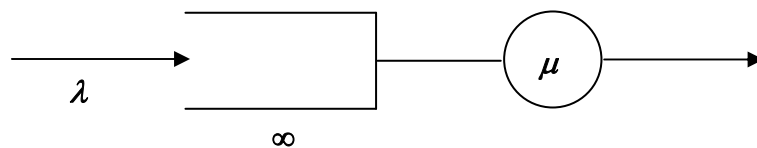
Từ phương trình trên, ta nhận thấy rằng **xác suất  $p_B$  để một yêu cầu bị từ chối** chính là **xác suất  $p_1$  để yêu cầu đi vào hệ thống nhận thấy trạm phục vụ đang bận.**

$$p_B = \frac{\lambda}{\mu + \lambda}$$

Kỳ vọng của số yêu cầu trong một hệ thống sẽ là:

$$E(N) = \sum_{n=0}^1 np_n = p_1 = \frac{\lambda}{\mu + \lambda}$$

### 3.3.3. Hệ thống hàng đợi **M/M/1** (M/M/1/vô cùng)



**Hình 3.11. Hàng đợi M/M/1**

Hệ thống hàng đợi **M/M/1** được đặc trưng bởi:

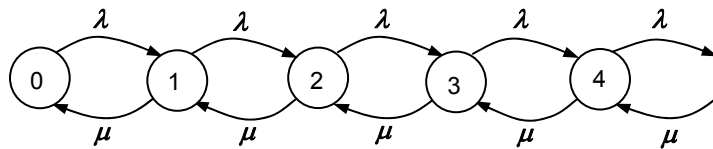
– Số yêu cầu đến tuân theo tiến trình Poisson với tham số là  $\lambda$ .

FIFO

- Thời gian phục vụ tuân theo phân bố mũ với tham số là  $\mu$ .
- Một trạm phục vụ.
- Hàng đợi có độ lớn là vô hạn, tức là không xảy ra trường hợp dịch vụ bị từ chối.

a. Tính xác suất trong hệ thống

Đối với hệ thống hàng đợi này, chúng ta sẽ đánh giá các thông số của nó dựa vào tiến trình sinh tử. Tại bất cứ thời điểm nào, cũng chỉ có nhiều nhất một sự kiện xảy ra (một yêu cầu đến hệ thống hàng đợi hoặc một yêu cầu rời khỏi hệ thống hàng đợi khi đã phục vụ xong). Điều này làm hệ thống hàng đợi  $M/M/1$  trở nên đơn giản và được biểu diễn như sau:



Hình 3.12. Sơ đồ chuyển đổi trạng thái của hệ thống  $M/M/1$

Giả sử hệ thống đã đạt đến trạng thái tĩnh và gọi xác suất để hệ thống ở trạng thái  $k$  là  $p_k$ . Xác suất này được xác định như sau:

$$p_k = \lim_{t \rightarrow \infty} p_k(t)$$

Trong đó  $p_k(t)$  là xác suất để có  $k$  yêu cầu trong hệ thống tại thời điểm  $t$ . Lưu ý rằng khi hệ thống đạt đến trạng thái ổn định, xác suất  $p_k$  sẽ không phụ thuộc vào thời gian. Chúng ta tập trung vào trạng thái  $k$  của hệ thống hàng đợi và thấy rằng để đạt đến trạng thái  $k$ , hệ thống có thể chuyển từ trạng thái  $(k-1)$  hoặc trạng thái  $(k+1)$ . Từ nhận xét của tiến trình sinh tử, hệ thống sẽ có tính chất cân bằng xác suất, nghĩa là xác suất hệ thống chuyển từ trạng thái  $i$  sang trạng thái  $(i+1)$  hoặc  $(i-1)$  sẽ bằng tổng xác suất hệ thống chuyển từ trạng thái  $(i+1)$  hoặc  $(i-1)$  về trạng thái  $i$ . Từ đó ta sẽ có phương trình sau:

$$\frac{dp_k(t)}{dt} = (\lambda p_{k-1}(t) + \mu p_{k+1}(t)) - (\lambda p_k(t) + \mu p_k(t))$$

Khi  $t$  dần đến vô cùng, chúng ta nhận được phương trình sau:

$$\frac{dp_k(t)}{dt} = 0$$

Với các giá trị  $k$  khác nhau, chúng ta sẽ nhận được hệ phương trình sau:

$$\lambda p_0 = \mu p_1$$

$$\lambda p_0 + \mu p_2 = (\lambda + \mu) p_1$$

$$\lambda p_{k-1} + \mu p_{k+1} = (\lambda + \mu) p_k$$

Sau khi giải hệ phương trình này, chúng ta nhận được kết quả sau:

$$p_k = \left(\frac{\lambda}{\mu}\right)^k p_0$$

Ngoài ra, tổng các xác suất  $p_k$  tồn tại ở các trạng thái sẽ là 1. Do đó chúng ta nhận được:

$$\sum_{k=0}^{\infty} p_k = 1$$

Vì vậy chúng ta có:

$$1 = p_0 + \sum_{k=1}^{\infty} p_k = p_0 + \sum_{k=1}^{\infty} p_0 \left(\frac{\lambda}{\mu}\right)^k = p_0 \left( \sum_{k=0}^{\infty} \left(\frac{\lambda}{\mu}\right)^k \right) = p_0 \frac{1}{1 - \frac{\lambda}{\mu}}$$

$$p_0 = 1 - \frac{\lambda}{\mu} = 1 - \rho \quad \text{Rô} = \text{lamda} / \text{muy}$$

Tóm lại, với hệ thống hàng đợi  $M/M/1$ , chúng ta nhận được:

$$p_0 = 1 - \frac{\lambda}{\mu} = 1 - \rho$$

$$p_k = \left(\frac{\lambda}{\mu}\right)^k p_0 = \rho^k (1 - \rho)$$



b. **Tính toán các tham số hiệu năng của hàng đợi M/M/1**

Đầu tiên chúng ta phải tính kỳ vọng của số yêu cầu trong hệ thống:

$$E(N) = \sum_{k=0}^{\infty} k p_k = (1-\rho) \sum_{k=0}^{\infty} k \rho^k = \frac{\rho}{1-\rho}$$

Như vậy:

$$\text{Kì vọng} \quad E(N) = \frac{\rho}{1-\rho}$$

Cũng có thể tính được phương sai của  $N$  như sau:

$$V(N) = E(N^2) - E(N)^2 = (1-\rho) \sum_n n^2 \rho^n - \left( \frac{\rho}{1-\rho} \right)^2 = \frac{\rho}{(1-\rho)^2}$$

Tiếp theo, để tính **kỳ vọng của số yêu cầu nằm trong hàng đợi**, chúng ta để ý rằng nếu có  $k$  yêu cầu nằm trong hệ thống thì sẽ có  $(k-1)$  yêu cầu nằm trong hàng đợi và một yêu cầu đang được phục vụ bởi trạm phục vụ ( $k \geq 1$ ), do đó:

$$E(N_q) = \sum_{k=1}^{\infty} (k-1) p_k = \sum_{k=1}^{\infty} k p_k - \sum_{k=1}^{\infty} p_k = \sum_{k=0}^{\infty} k p_k - \left( \sum_{k=0}^{\infty} p_k - p_0 \right) = N - (1-p_0)$$

Từ phương trình trên ta có:

$$\text{kì vọng của số yêu cầu trong hàng đợi} \quad E(N_q) = \frac{\rho^2}{1-\rho}$$

Tương tự như trên, chúng ta tính được:

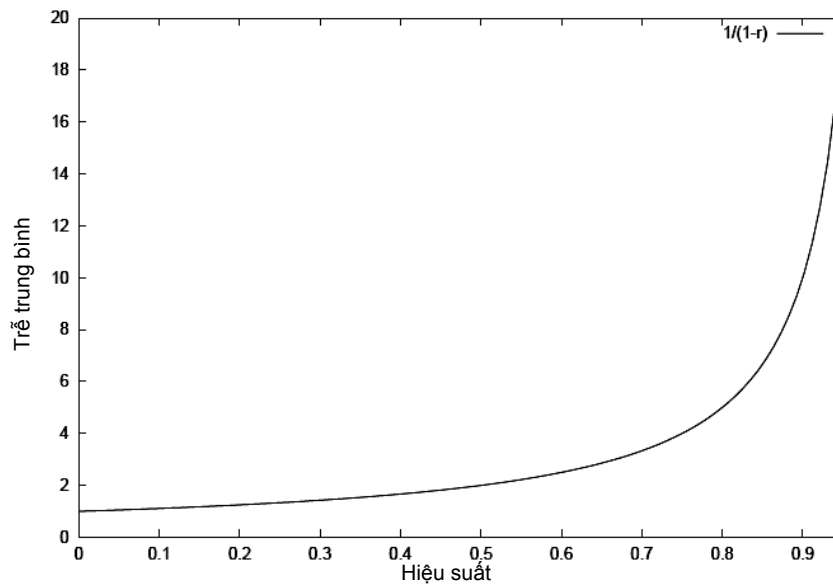
$$V(N_q) = \frac{\rho^2}{(1-\rho)^2} (1 + \rho - \rho^2)$$

Tính **thời gian đáp ứng trung bình**:

Thời gian đáp ứng trung bình  $T$  là thời gian trung bình các yêu cầu trong hệ thống, tức là cả phần thời gian trong hàng đợi và phần thời gian được phục vụ. Theo định lý Little Law chúng ta có:

$$\bar{T} = \frac{\bar{N}}{\lambda} = \frac{1/\mu}{1-\rho} = \frac{1}{\mu - \lambda}$$

Trong trường hợp  $\mu = 1$  thì thời gian đáp ứng trung bình (hay trễ trung bình) của một yêu cầu trong hệ thống được biểu diễn bằng hình vẽ sau.

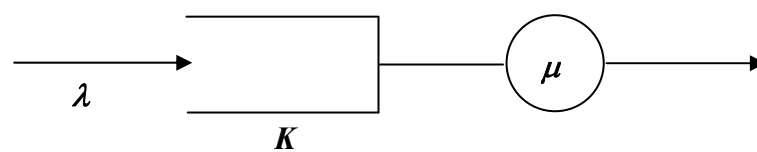


**Hình 3.13. Trễ trung bình**

Trong hàng đợi này, có một thông số quan trọng nữa cần phải quan tâm, đó là xác suất để hệ thống có nhiều hơn  $k$  yêu cầu được tính như sau:

$$p(N > k) = 1 - p(N \leq k) = 1 - \sum_{v=0}^k p_v = 1 - p_0 \frac{1 - \rho^{k+1}}{1 - \rho} = \rho^{k+1}$$

### 3.3.4. Hàng đợi $M/M/1/K$



**Hình 3.14. Hệ thống  $M/M/1/K$**

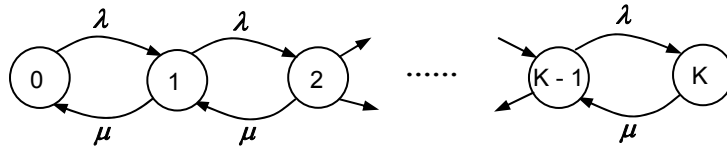
Hệ thống hàng đợi  $M/M/1/K$  được đặc trưng bởi:

- Số yêu cầu đến tuân theo tiến trình Poisson với tham số là  $\lambda$ .
- Thời gian phục vụ tuân theo phân bố mũ với tham số là  $\mu$ .
- Một trạm phục vụ.

– Hàng đợi có độ lớn  $K$  phần tử. Như vậy nếu một yêu cầu đi vào hàng đợi mà trong hàng đợi đã có  $K$  yêu cầu khác thì yêu cầu mới đến sẽ bị từ chối.

a. Tính xác suất có  $n$  yêu cầu trong hệ thống

Đầu tiên chúng ta biểu diễn hệ thống hàng đợi  $M/M/1/K$  bằng tiến trình sinh – tử:



**Hình 3.15. Hệ thống hàng đợi  $M/M/1/K$**

Hệ phương trình cân bằng của hệ thống này được biểu diễn như sau:

$$\begin{cases} \lambda p_0 = \mu p_1 \\ \lambda p_{n-1} + \mu p_{n+1} = (\lambda + \mu) p_n; \quad n \in \{1, \dots, K-1\} \\ \lambda p_{K-1} = \mu p_K; \quad n = K \end{cases}$$

Từ hệ phương trình trên, ta được:

$$p_n = \rho^n p_0; n \in \{0, K\}$$

Rô = lamda / mu

Mặt khác ta cũng có:

$$\sum_{n=0}^K p_n = 1$$

Từ đó tính ra được:

$$p_0 = \frac{1}{\sum_{n=0}^K \rho^n} = \frac{1-\rho}{1-\rho^{K+1}}$$

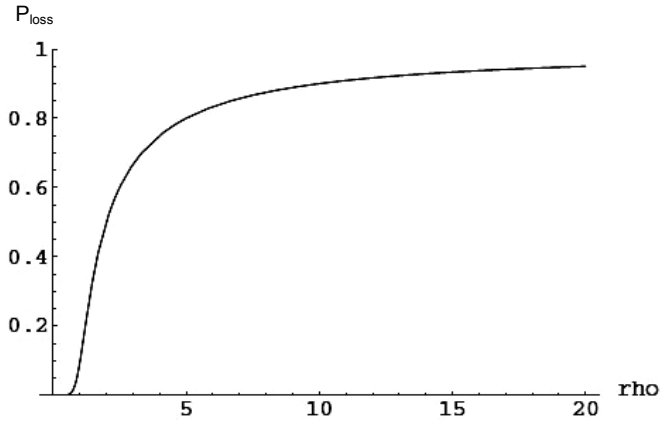
$$p_n = \frac{1-\rho}{1-\rho^{K+1}} \rho^n$$

b. Tính toán các tham số hiệu năng của hệ thống  $M/M/1/K$

Tính xác suất từ chối dịch vụ (xác suất mất yêu cầu).

Chúng ta nhận xét rằng một yêu cầu bị từ chối khi nó đến hàng đợi và thấy có  $K$  yêu cầu đang ở trong hàng đợi. Gọi **xác suất để một yêu cầu bị từ chối dịch vụ là  $P_{loss}$** , ta nhận thấy rằng:

$$P_{loss} = p_K = \frac{1-\rho}{1-\rho^{K+1}} \times \rho^K; n \in \{0, \dots, K\}$$



**Hình 3.16. Xác suất yêu cầu từ chối dịch vụ của hàng đợi M/M/1/20**

Tính kỳ vọng của số yêu cầu trong hệ thống:

Ta có:

$$E(N) = \sum_{n=1}^K n p_n = \frac{1-\rho}{1-\rho^{K+1}} \sum_{n=1}^K n \rho^n$$

Để tính được phương trình trên, chúng ta chú ý đến phương trình sau:

$$f(\rho) = \sum_{n=0}^K \rho^n = \frac{1-\rho^{K+1}}{1-\rho}$$

Lấy vi phân phương trình trên:

$$f'(\rho) = \sum_{n=1}^K n \rho^{n-1} = \frac{1-(K+1)\rho^K + K\rho^{K+1}}{(1-\rho)^2}$$

Thay vào ta có:

$$E(N) = \frac{\rho}{1-\rho} \frac{1-(K+1)\rho^K + K\rho^{K+1}}{1-\rho^{K+1}} = \frac{\rho}{1-\rho} - (K+1) \frac{\rho^{K+1}}{1-\rho^{K+1}}$$

Ngoài ra chúng ta cũng tính được:

$$V(N) = \frac{\rho}{(1-\rho)^2} - (K+1)^2 \frac{\rho^{K+1}}{(1-\rho^{K+1})^2}$$

Tính kỳ vọng của số yêu cầu trong hàng đợi:

$$E(N_q) = \sum_{n=1}^K (n-1)p_n = \sum_{n=1}^K np_n - \sum_{n=1}^K p_n = E(N) - (1-p_0)$$

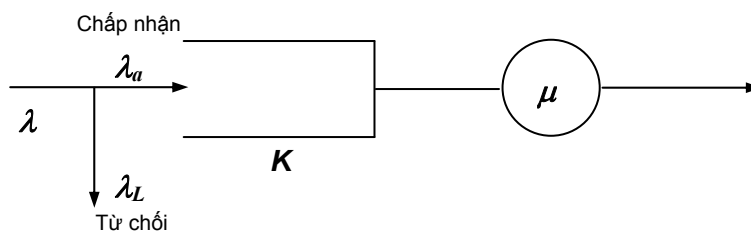
Cuối cùng ta có:

$$E(N_q) = \frac{1}{1-\rho} + \frac{\rho - (K+1)\rho^{K+1}}{1-\rho^{K+1}}$$

Nhận xét: Nếu gọi  $\lambda_a \equiv \lambda(1-p_K)$  là tốc độ yêu cầu đi vào hàng đợi thì ta tuân theo định lý Little:

$$T_q = \frac{N_q}{\lambda_a} \quad \text{thời gian trung bình 1 yêu cầu trong hàng đợi}$$

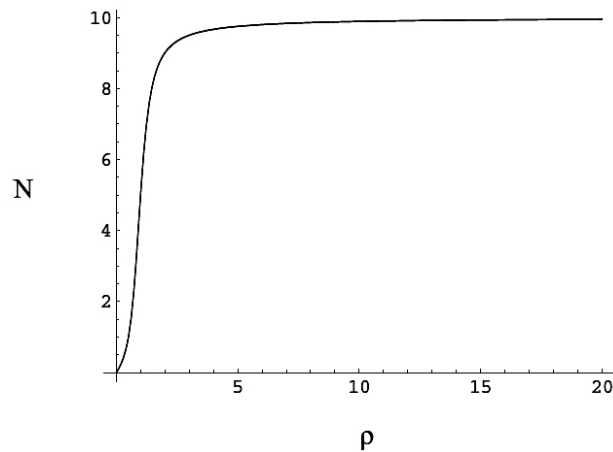
Nếu quan sát hàng đợi  $M/M/1/K$  trên quan điểm một hệ thống đóng, chúng ta để ý rằng tại đầu vào hệ thống, dòng yêu cầu đầu vào với tốc độ trung bình  $\lambda$  sẽ được chia thành hai nhánh: nhánh yêu cầu đi vào hàng đợi với tốc độ trung bình  $\lambda_a = \lambda(1-p_K)$  và nhánh các yêu cầu bị từ chối với tốc độ trung bình  $\lambda_L = \lambda p_K$ .



**Hình 3.17. Hệ thống  $M/M/1/K$  trên quan điểm hệ thống đóng**

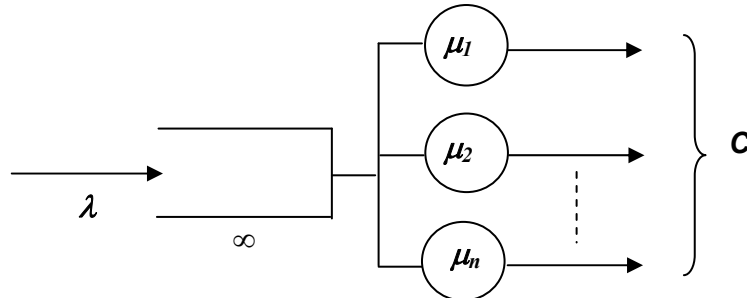
Cũng tính toán tương tự như trên, ta có thời gian trung bình  $T$  một yêu cầu lưu lại hệ thống:

$$T = \frac{1}{\lambda} * \frac{1}{1 - p_K} * E(N) = \frac{E(N)}{\lambda_a}$$



Hình 3.18. Số yêu cầu trung bình trong hệ thống M/M/1/10

### 3.3.5. Hàng đợi M/M/m

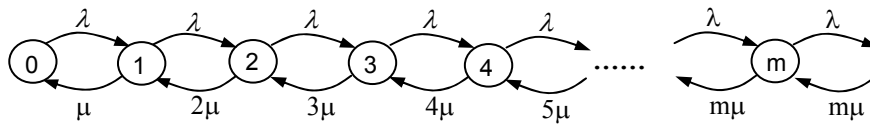


Hình 3.19. Hàng đợi M/M/m

Hệ thống hàng đợi M/M/m có các đặc điểm sau:

- Số yêu cầu tới tuân theo tiến trình Poisson với tham số là  $\lambda$ .
- Có  $m$  trạm phục vụ.
- Thời gian phục vụ của một trạm  $i$  bất kỳ tuân theo phân bố mũ với tham số là  $\mu_i$ .
- Hàng đợi có chiều dài vô tận.

a. Tính xác suất có  $n$  yêu cầu trong hệ thống



**Hình 3.20. Chuyển đổi trạng thái của hàng đợi M/M/m**

Để tính toán các xác suất của hệ thống  $M/M/m$ , người ta cũng thực hiện tương tự như các hàng đợi trước dựa vào tiến trình sinh tử. Khi đó chúng ta có:

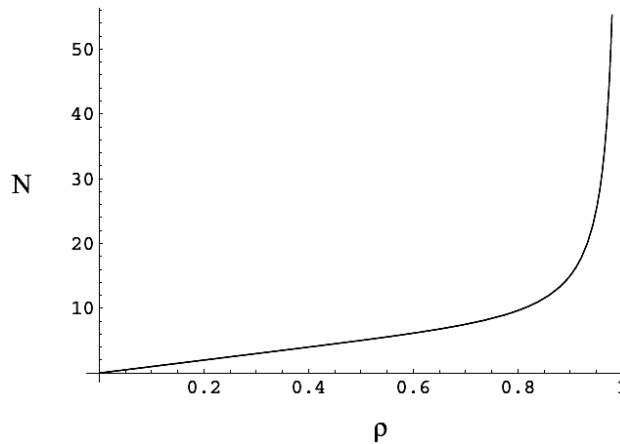
$$p_0 = \left[ \sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!} + \left( \frac{(m\rho)^m}{m!} \right) \left( \frac{1}{1-\rho} \right) \right]^{-1} \quad \text{Rô} = \text{lamda} / \text{muy}$$

$$p_k = \begin{cases} p_0 \frac{(m\rho)^k}{k!} & : k \leq m \\ p_0 \frac{\rho^k m^m}{m!} & : k \geq m \end{cases}$$

b. Tính toán các thông số hiệu năng của hệ thống

Kỳ vọng của số yêu cầu trong hệ thống được tính toán như sau:

$$E(N) = \sum_{k=0}^{\infty} k p_k = m\rho + \rho \frac{(m\rho)^m}{m!} \frac{p_0}{(1-\rho)^2}$$



**Hình 3.21. Số yêu cầu trong hệ thống của hàng đợi M/M/10**

Tiếp theo, chúng ta tính toán xác suất một yêu cầu đến hệ thống phải chờ trong hàng đợi vì không còn trạm phục vụ nào rỗi cả. Xác suất này thường được sử dụng trong mạng điện thoại và chính là xác suất khi một cuộc gọi đến tổng đài nhưng không còn đường trung kế nào rỗi cả, với giả sử các tiến trình đến của các cuộc gọi và tiến trình phục vụ của tổng đài tuân theo phân bố mũ. Lúc đó xác suất này được tính toán như sau:

$$\Pr[\text{queueing}] = \sum_{k=m}^{\infty} p_k = \sum_{k=m}^{\infty} p_0 \frac{(m\rho)^k}{m!} \frac{1}{m^{k-m}} = \frac{\left(\frac{(m\rho)^m}{m!}\right) \left(\frac{1}{1-\rho}\right)}{\sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!} + \left(\frac{(m\rho)^m}{m!}\right) \left(\frac{1}{1-\rho}\right)}$$

Công thức này thường được gọi là công thức **Erlang – C**, ký hiệu là **C(m,ρ)**.

### 3.3.6. So sánh các hệ thống hàng đợi

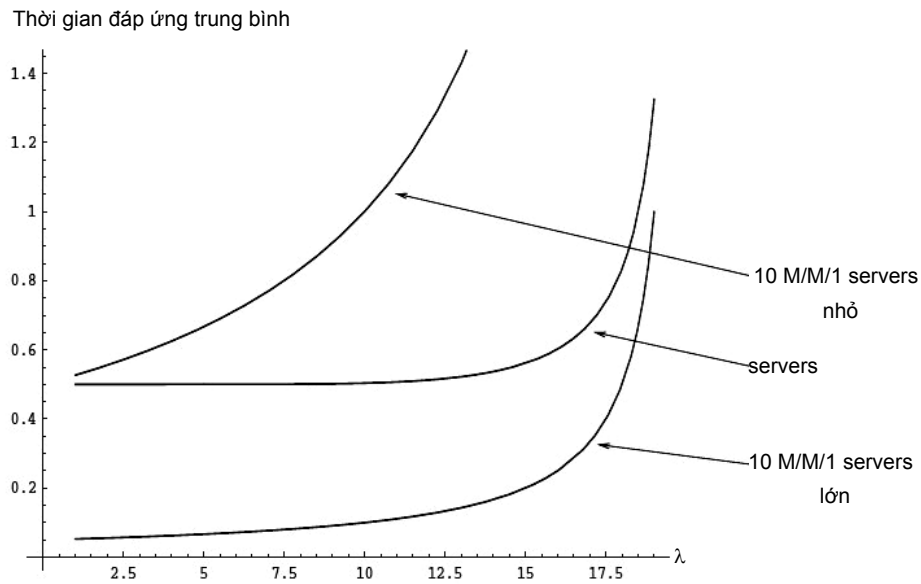
Trong phần này, người ta sẽ tiến hành so sánh ba hệ thống hàng đợi khác nhau trên phương diện thời gian đáp ứng trung bình (hay trễ trung bình) của hệ thống so với tải. Giả thiết rằng có hệ thống hàng đợi  $M/M/1$  với một trạm phục vụ có tốc độ là  $m\mu$ , một hệ thống hàng đợi  $M/M/m$  với mỗi trạm phục vụ có tốc độ là  $\mu$  và một hệ thống hàng đợi gồm  $m$  hàng đợi theo kiểu  $M/M/1$  hoạt động độc lập với nhau với tốc độ  $\mu$ .

Việc so sánh các hàng đợi này có thể thích hợp với bối cảnh sau trong thực tế: một máy tính với bộ xử lý theo kiểu  $X$  và một tập hợp các người sử dụng đang phải chờ đợi bên ngoài. Người ta thấy rằng cần thiết phải mở rộng máy tính này và để mở rộng thì có ba khả năng như sau:

- Mua  $m - 1$  bộ xử lý dạng  $X$  và lắp ráp vào một máy tính duy nhất, tạo nên một máy tính đa xử lý.
- Mua một bộ xử lý mới kiểu  $Y$  có tốc độ nhanh hơn bộ xử lý  $X$  là  $n$  lần và thay vào bộ xử lý cũ.
- Cung cấp cho mỗi người sử dụng một thiết bị riêng biệt, mỗi thiết bị này được trang bị một bộ xử lý kiểu  $X$ .

Đồ thị sau biểu diễn thời gian đáp ứng trung bình của ba hệ thống này trong trường hợp  $n = 10$ .






**Hình 3.22. Thời gian đáp ứng trung bình**

Dựa vào đồ thị trên, chúng ta thấy rằng giải pháp tốt nhất là giải pháp thứ hai vì nó tạo ra thời gian đáp ứng nhỏ nhất, tiếp theo là giải pháp thứ nhất và cuối cùng là giải pháp thứ ba. Giải pháp thứ nhất tương đương với hệ thống hàng đợi  $M/M/m$  với mỗi trạm phục vụ có tốc độ  $\mu$  và tốc độ đến của toàn hệ thống là  $\lambda$ . Giải pháp thứ hai tương đương với hệ thống  $M/M/I$  với tốc độ đến là  $\lambda$  và tốc độ của trạm phục vụ là  $\mu\lambda$ . Giải pháp cuối cùng tương ứng với hệ thống hàng đợi  $M/M/I$  với tốc độ đến là  $\lambda/\mu$  và tốc độ của trạm phục vụ là  $\mu$ . Đồ thị trên tương ứng với  $m = 10$  và  $\mu = 2$ .

## BÀI TẬP CHƯƠNG 3

1. Cho một hệ thống hàng đợi  $M/M/1$  với tốc độ đến  $\lambda = 1$  và tốc độ phục vụ  $\mu = 4$ .
  - a. Hãy tính xác suất  $p_0$  để hệ thống không có yêu cầu nào? 
  - b. Tính xác suất để hệ thống có từ 1 đến 10 yêu cầu?
2. Cho một hệ thống hàng đợi  $M/M/1$ , trong đó tốc độ phục vụ là không đổi, tuy nhiên tốc độ đến của  $k$  yêu cầu trong hệ thống là hàm của  $k$  như sau:  $\lambda/(k+1)$ .
  - a. Hãy vẽ sơ đồ trạng thái của hệ thống?
  - b. Tình xác suất  $p_k$  để có  $k$  yêu cầu trong hệ thống như là một hàm của  $\lambda, \mu, p_0$ ?
  - c. Tính  $P_0$ ?
3. Cho một hàng đợi  $M/M/1$  với  $\rho = \lambda/\mu = 0,4$ . Hãy xác định  $m$  sao cho xác suất để có  $m$  yêu cầu trong hệ thống là nhỏ hơn 1%?
4. Cho một hàng đợi  $M/M/1/N$ . Hãy tính xác suất để yêu cầu đến rời khỏi hệ thống với  $N = 5$  và:

là tính Ploss

- a.  $\rho = \lambda/\mu = 1$ .
- b.  $\rho = \lambda/\mu = 0,75$ . lamda
5. Cho một hệ thống hàng đợi  $M/M/1/6$  có **tốc độ đến là 20** yêu cầu trên một đơn vị thời gian và **tốc độ phục vụ là 10** yêu cầu trên một đơn vị muy thời gian. Tính số yêu cầu trung bình trong hệ thống?
6. Cho một hệ thống máy tính có thể mô tả bằng hàng đợi  $M/M/1$  với tốc độ đến  $\lambda$  và tốc độ phục vụ là  $\mu$ . Có ba trường hợp xảy ra như sau:

M/M/1/vô cùng

- Thay thế trạm phục vụ có tốc độ phục vụ mới gấp hai lần.
  - Thêm một trạm phục vụ nữa có cùng tốc độ hoạt động song song, tức là lúc này hệ thống là  $M/M/2$ .
  - Thêm một hệ thống máy tính mới đặt song song với hệ thống cũ, tức là lúc này tốc độ đến giảm đi một nửa.
- a. Hãy so sánh hiệu năng của ba trường hợp trên.
  - b. Giải pháp nào dễ thực hiện và có giá thành nhỏ nhất trong thực tế?
7. Hãy giải thích tại sao tốc độ đến của hệ thống hàng đợi không nên quá cao?
8. Sự không ổn định của các hàng đợi có kích thước hữu hạn là gì?

## TÀI LIỆU THAM KHẢO

1. D. Gross, C. M. Harris. *Fundamentals of Queueing Theory*. Wiley, Second Edition, New York, 1985.
2. L. Kleinrock. *Queueing Systems, Vol. 1: Theory*. Wiley, New York, 1975. *Vol. 2: Computer Applications*. Wiley, New York, 1976.
3. D.V. Lindley. *The Theory of Queues with a Single Server*. Proc. Camb Phil. Soc. 48, 277–289, 1952.
4. T. L. Saaty. *Elements of Queueing Theory with Applications*. Dover, New York, 1961.
5. H. M. Wagner. *Principles of Operation Research*. Prentice Hall International Editors, London, 1972.
6. Gunter Bolch, Stefan Greiner, Hermann de Meer, and Kishov S. Trivedi. *Queueing Networks and Markov Chain– Modelling and Performance Evaluation with Computer Science Applications*. John Wiley and Sons, New York, 1998.
7. Gunter Bolch. *Leistungsbewertung von Rechensystemen– mittels analytischer Warteschlangenmodelle*. B. G. Teubner, Stuttgart, 1989.
8. Boudewijn R. Haverkort. *Performance of Computer Communication Systems–A Model Based Approach*. John Wiley and Sons, Chichester/New York, 1998.
9. Randolph Nelson. *Probability, Stochastic Processes, and Queueing Theory – The Mathematics of Computer Performance Modeling*. Springer Verlag, New York, 1995.

10. Thomas G. Robertazzi. *Computer Networks and Systems – Queueing Theory and Performance Evaluation*. Springer Verlag, New York, 1994.
11. Rolf Schassberger. *Warteschlangen*. Springer Verlag, Wien, 1973.
12. William J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, Princeton, New Jersey, 1994.
13. Phuoc Tran-Gia. *Analytische Leistungsbewertung verteilter Systeme–eine Einfuhrung*. Springer Verlag, Berlin, 1996.

## Chương 4

# HỆ THỐNG MẠNG HÀNG ĐỢI

### 4.1. MẠNG HÀNG ĐỢI

Trong thực tế, nhiều hệ thống không được mô hình hóa bằng các hệ thống hàng đợi đơn mà phải được mô hình hóa bằng một tập hợp gồm nhiều hệ thống hàng đợi. Một ví dụ cụ thể nhất chính là mạng Internet, nơi mà các bộ định tuyến và các tuyến truyền dẫn có thể được mô hình hóa bằng các hệ thống hàng đợi đơn và mạng sẽ là một tập hợp của nhiều hàng đợi đơn. Mỗi khi một gói được phục vụ xong tại một bộ định tuyến (được mô hình hóa bằng một hàng đợi đơn) thì nó sẽ ngay lập tức đến một hàng đợi đơn khác và cứ như thế cho đến khi gói đến được đích.

Do đó, để có thể đánh giá hiệu năng của các hệ thống này, người ta phải phân tích mô hình hệ thống mạng hàng đợi.

Mạng hàng đợi có thể coi là **một tập hợp của nhiều nút, mỗi nút có thể coi như một hệ thống hàng đợi đơn**, mỗi hàng đợi đơn này có thể có một hay nhiều trạm phục vụ. Các yêu cầu đi vào hệ thống mạng hàng đợi ở một số nút xác định và đi ra ở một số nút khác.

Trong trường hợp tổng quát nhất, các yêu cầu sẽ đến hệ thống và được phục vụ ở một trạm nào đó. Sau khi được phục vụ xong thì các yêu cầu này có thể chuyển sang nút khác để được phục vụ tiếp, hoặc cũng có thể quay lại chính nút vừa rồi để được phục vụ lại, hoặc rời khỏi hệ thống mạng hàng đợi này.

Chúng ta sẽ xem xét các hệ thống mạng hàng đợi thỏa mãn điều kiện sau:

- Tiến trình đến nút  $i$  từ bên ngoài tuân theo luật phân bố Poisson với tốc độ đến là  $\gamma_i$ .

- Tại mỗi nút  $i$ , tốc độ phục vụ của các trạm không phụ thuộc nhau và tuân theo phân bố mũ với tốc độ là  $\mu_i$ .

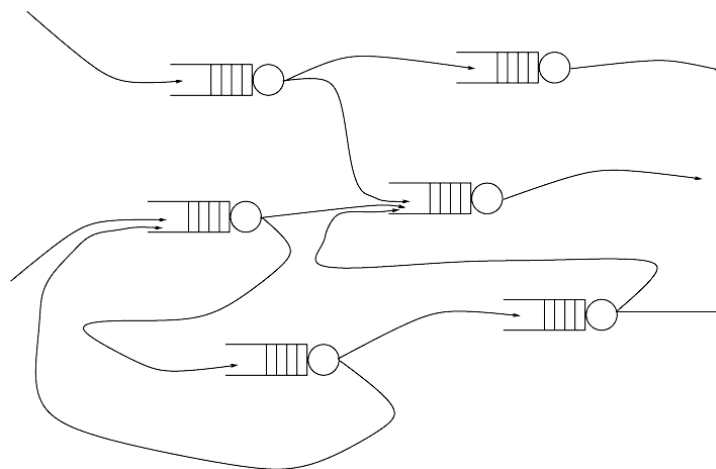
- Xác suất để một yêu cầu sau khi được phục vụ xong tại nút  $i$  và chuyển sang nút  $j$  là  $r_{ij}$ . Xác suất  $r_{ij}$  này không phụ thuộc vào trạng thái của hệ thống mạng hàng đợi. Trong đó  $i = 1, 2, \dots, k, j = 0, 1, \dots, k$  và  $r_{i0}$  là xác suất để một yêu cầu sau khi được phục vụ tại nút  $i$  rời khỏi hệ thống.

Hệ thống mạng hàng đợi, khi thỏa mãn các điều kiện trên, được gọi là mạng Jackson. Trong phần tiếp theo, chúng ta sẽ xem xét các thông số hiệu năng của hệ thống mạng hàng đợi này khi đạt đến trạng thái tĩnh.

Trong trường hợp hệ thống mạng hàng đợi có các giá trị  $\gamma_i = 0$  và  $r_{i0} = 0$ , nghĩa là hệ thống mạng hàng đợi này không có yêu cầu nào đi vào và cũng không có yêu cầu nào đi ra khỏi hệ thống, thì hệ thống này được gọi là mạng Jackson đóng.

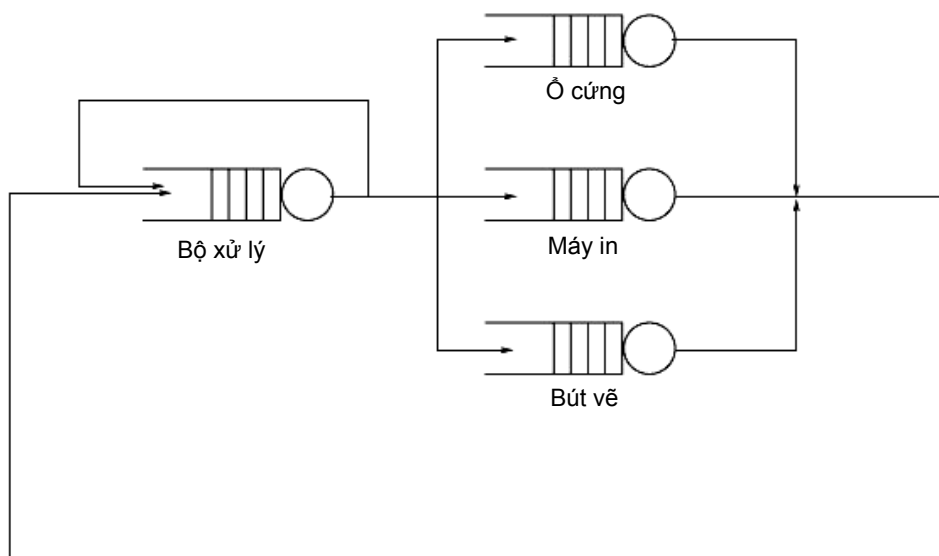
Ngược lại, mạng Jackson mở là mạng có các giá trị  $\gamma_i \neq 0$  và  $r_{i0} \neq 0$ , nghĩa là có thể có các yêu cầu đi vào hệ thống và rời khỏi hệ thống.

Nói cách khác, người ta phân biệt các mạng hàng đợi thành hai loại: mạng hàng đợi mở và mạng hàng đợi đóng. Trong mạng hàng đợi mở, các yêu cầu hay các khách hàng có thể đến từ phía ngoài hệ thống và rời khỏi hệ thống. Trong mạng hàng đợi đóng, số yêu cầu hay số khách hàng là cố định và không có yêu cầu nào rời khỏi hệ thống. Có thể tham khảo mạng hàng đợi đóng và mở ở hai hình vẽ sau. Trong hình vẽ thể hiện mạng hàng đợi mở, các yêu cầu có thể đến hệ thống ở bất kỳ hàng đợi đơn nào và cũng có thể rời khỏi hệ thống ở bất kỳ hàng đợi nào. Hệ thống có thể chứa các vòng lặp hoặc các ngã rẽ.



**Hình 4.1. Mạng hàng đợi mở**

Đối với mạng hàng đợi đóng, có thể xem xét một hệ thống máy tính. Trong hệ thống máy tính này, số lượng các nhiệm vụ phải thực hiện (hay các yêu cầu) là cố định. Mỗi một nhiệm vụ hay một yêu cầu phải được tính toán, truy cập vào ổ cứng, sau đó lại sử dụng bộ xử lý.



**Hình 4.2. Ví dụ mạng hàng đợi đóng**

Sau đây, chúng ta sẽ chỉ xem xét trường hợp của mạng hàng đợi mà các yêu cầu hoặc các khách hàng có các đặc tính giống nhau.



## 4.2. HỆ THỐNG MẠNG NỐI TIẾP

Một hệ thống mạng Jackson mở nếu thỏa mãn các điều kiện sau:

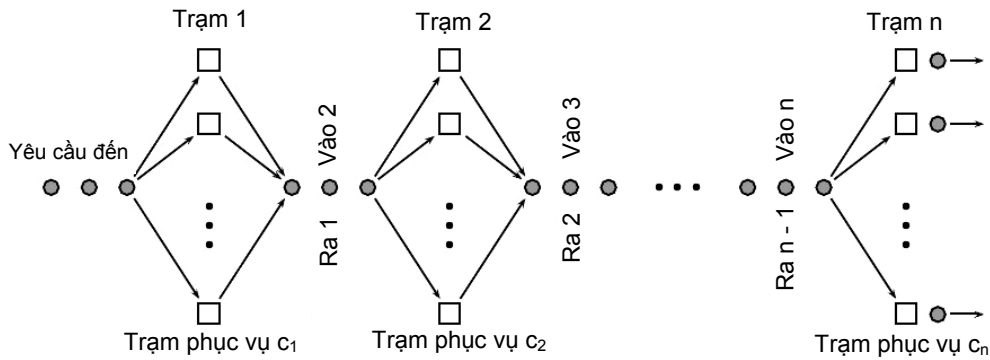
$$\gamma_i = \begin{cases} \lambda(i=1) \\ 0(i \neq 1) \end{cases}$$

và điều kiện:

$$r_{ij} = \begin{cases} 1(j = i+1, 1 \leq i \leq k-1) \\ 1(i = k, j = 0) \\ 0 \end{cases}$$

xác suất để 1 yêu cầu sau khi được phục vụ xong ở nút i thì đến nút j

được gọi là một mạng nối tiếp. Trong mạng nối tiếp này, các nút sẽ sắp xếp thành một hàng, trong đó các yêu cầu được phục vụ xong ở nút này sẽ được chuyển sang thành yêu cầu vào ở các nút tiếp theo. Cụ thể được minh họa trong hình vẽ sau:



**Hình 4.3. Hệ thống mạng nối tiếp**

Ở đây, chúng ta xem xét một dãy các nút với các hàng đợi có kích thước vô hạn. Chúng ta cũng giả thiết rằng các yêu cầu đến hệ thống sẽ tuân theo phân bố Poisson với thông số  $\lambda$ . Thời gian phục vụ của các trạm phục vụ của nút mạng thứ  $i$  tuân theo luật phân bố mũ với tham số  $\mu_i$ . Ngoài ra do các hàng đợi không có giới hạn về mặt kích thước, nên mỗi nút mạng có thể được xử lý một cách riêng rẽ so với các nút mạng khác.

Nút mạng thứ nhất có thể được coi như một hệ thống hàng đợi theo mô hình  $M/M/c_1/\infty$ . Để xác định được phân bố thời gian của tiến trình

đến của một nút mạng, chúng ta phải xác định phân bố thời gian phục vụ của nút mạng trước đó. Ngoài ra, với hệ thống mạng hàng đợi theo kiểu nối tiếp, phân bố thời gian của tiến trình đến tại một nút mạng chính là phân bố thời gian của tiến trình phục vụ của nút mạng trước đó.

**Định lý 4.1:** Cho một hệ thống hàng đợi đơn theo kiểu  $M/M/c$  ở trạng thái tĩnh. Thời gian đến của các yêu cầu đến hệ thống hàng đợi đơn này tuân theo phân bố mũ với tham số  $\lambda$ . Khi đó khoảng thời gian giữa hai thời điểm phục vụ xong liên tiếp nhau cũng tuân theo phân bố mũ với tham số  $\lambda$ .

Chứng minh: Gọi  $p_n$ , với  $n=0,1,2,\dots$  là xác suất để hệ thống có  $n$  yêu cầu:

$$p_n = p(N=n) \text{ với } n=0,1,2,\dots$$

Nếu gọi  $T$  là biến ngẫu nhiên, biểu thị khoảng cách giữa hai thời điểm phục vụ xong nối tiếp nhau, tức là:

$$F_n(t) = p(N(t)=n \text{ \& } T > t)$$

Trong đó  $F_n(t)$  là luật phân bố, để tại thời điểm  $t$  có  $n$  yêu cầu sau khi yêu cầu cuối cùng được phục vụ xong và  $t$  nhỏ hơn thời gian phục vụ, có nghĩa là yêu cầu tiếp theo vẫn chưa được phục vụ. Qua đó chúng ta tính được:

$$F_T(t) = p(T \leq t) = 1 - \sum_{n=0}^{\infty} F_n(t)$$

và:

$$\sum_{n=0}^{\infty} F_n(t) = p(T > t)$$

Bây giờ chúng ta sẽ có:

$$F_n(t+dt) = (1-\lambda dt)(1-c\mu dt)F_n(t) + \lambda dt(1-c\mu dt)F_{n-1}(t) \text{ với } c \leq n$$

$$F_n(t+dt) = (1-\lambda dt)(1-n\mu dt)F_n(t) + \lambda dt(1-n\mu dt)F_{n-1}(t) \text{ với } 1 \leq n \leq c$$

$$F_0(t+dt) = (1-\lambda dt)F_0(t)$$

Bởi vậy chúng ta có khi  $dt \rightarrow 0$ :

$$\frac{dF_n(t)}{dt} = -(\lambda + c\mu)F_n(t) + \lambda F_{n-1}(t)$$

$$\frac{dF_n(t)}{dt} = -(\lambda + n\mu)F_n(t) + \lambda F_{n-1}(t)$$

$$\frac{dF_0(t)}{dt} = -\lambda F_0(t)$$

Với các điều kiện bờ như sau:

$$F_n(0) = p(N(0) = n \& T > 0) = p(N(0) = n) = p_n$$

Chúng ta sẽ nhận được:

$$F_n(t) = p_n e^{-\lambda t}$$

Áp dụng các tính toán với hàng đợi  $M / M / c$ :

$$p_{n+1} = \begin{cases} \frac{1}{n+1} \frac{\lambda}{\mu} p_n & (1 \leq n \leq c) \\ \frac{1}{c} \frac{\lambda}{\mu} p_n & (c < n) \end{cases}$$

Do đó chúng ta nhận được:

$$F_T(t) = 1 - \sum_{n=0}^{\infty} p_n e^{-\lambda t} = 1 - e^{-\lambda t} \sum_{n=0}^{\infty} p_n = 1 - e^{-\lambda t}$$

Đây là điều cần phải chứng minh của định lý 4.1.

### 4.3. HỆ THỐNG MẠNG JACKSON MỞ

Giả sử chúng ta có một mạng hàng đợi gồm  $k$  nút. Các yêu cầu có thể từ bên ngoài đến bất cứ một nút nào của hệ thống mạng hàng đợi này theo tiến trình Poisson. Chúng ta gọi tốc độ đến nút  $i$  là  $\gamma_i$  (chứ không phải  $\lambda_i$ ). Mỗi nút  $i$  có các trạm phục vụ theo phân bố mũ với tốc độ phục vụ trung bình  $\mu_i$  (tức là tất cả các trạm phục vụ tại nút  $i$  đều có tốc độ phục vụ như nhau). Khi một yêu cầu được phục vụ xong tại nút  $i$ , nó có thể tiếp tục đi

đến nút  $j$  với xác suất  $r_{ij}$  và giả thiết xác suất này không phụ thuộc vào trạng thái của hệ thống. Chúng ta sẽ phải có  $\sum_{j=0}^k r_{ij} = 1$  với  $i = 1, \dots, k$ . Xác suất  $r_{i0}$  sẽ là xác suất để yêu cầu rời khỏi hệ thống sau khi đã được phục vụ ở nút  $i$ . Kích thước hàng đợi ở các nút  $i$  là không bị giới hạn.

Ngoài ra chúng ta thấy rằng số lượng yêu cầu nằm trong hệ thống mạng hàng đợi này sẽ chính bằng tổng số lượng các yêu cầu có mặt tại tất cả các nút của hệ thống. Gọi  $N_i$  là số yêu cầu có tại nút  $i$ . Chúng ta có:

$$w(N_1 = n_1, N_2 = n_2, \dots, N_k = n_k) = p_{n_1, n_2, \dots, n_k}$$

Để có thể xác định được phân bố của biến ngẫu nhiên  $N_i$ , phải mô tả hệ thống này khi nó hoạt động ở trạng thái ổn định. Sau khi hệ thống ở trạng thái ổn định rồi, có thể dựa vào tiến trình sinh tử để xác định phương trình cân bằng xác suất như ở các phần trước.

Để thuận tiện hơn, chúng ta sẽ sử dụng các ký hiệu sau:

Trạng thái  $n_1, n_2, \dots, n_i, \dots, n_j, \dots, n_k$  được kí hiệu bằng  $\bar{n}$ .

Trạng thái  $n_1, n_2, \dots, n_i + 1, \dots, n_j, \dots, n_k$  được kí hiệu bằng  $\bar{n}; i +$ .

Trạng thái  $n_1, n_2, \dots, n_i - 1, \dots, n_j, \dots, n_k$  được kí hiệu bằng  $\bar{n}; i -$ .

Trạng thái  $n_1, n_2, \dots, n_i + 1, \dots, n_j - 1, \dots, n_k$  được kí hiệu bằng  $\bar{n}; i + j -$ .

Chúng ta cũng giả thiết rằng mỗi một nút  $i$  chỉ có một trạm phục vụ, tức là  $c_i = 1$  với  $i = 1, \dots, k$ . Áp dụng phương trình cân bằng xác suất, tổng xác suất chuyển đến trạng thái  $\bar{n}$  sẽ bằng tổng xác suất xuất phát từ trạng thái đó. Chúng ta sẽ nhận được phương trình sau:

$$\sum_{i=1}^k \gamma_i p_{\bar{n}; i-} + \sum_{j=1}^k \sum_{i=1}^k \mu_i r_{ij} p_{\bar{n}; i+j-} + \sum_{i=1}^k \mu_i r_{i0} p_{\bar{n}; i+} = \sum_{i=1}^k \mu_i (1 - r_{ii}) p_{\bar{n}} + \sum_{i=1}^k \gamma_i p_{\bar{n}} \quad (4.1)$$

Chú ý rằng:

$$\sum_{i=1}^k \sum_{j=0}^k r_{ij} \mu_i p_{\bar{n}} = \sum_{i=1}^k \mu_i p_{\bar{n}} \sum_{j=0}^k r_{ij} = \sum_{i=1}^k \mu_i p_{\bar{n}} (1 - r_{ii})$$

Người ta đã chỉ ra rằng, kết quả của phương trình trên có thể được biểu diễn dưới dạng như sau:

**Định lý 4.2:** Cho một mạng Jackson. Gọi  $\lambda_i$  là tốc độ tổng cộng đến nút  $i$  (bao gồm các yêu cầu đến từ bên ngoài và các yêu cầu đến sau khi đã phục vụ xong từ các nút khác). Khi đạt đến trạng thái cân bằng, phương trình cân bằng tại mỗi nút sẽ có:

$$\lambda_i = \gamma_i + \sum_{j=1}^k r_{ji} \lambda_j \text{ với } i=1, \dots, k \quad (4.2)$$

Giả thiết tiếp theo là  $\rho_i = \lambda_i / \mu_i$  với  $i=1, \dots, k$ . Từ đó kết quả của (4.1) trở thành:

$$p_n = p_{n_1, n_2, \dots, n_k} = (1 - \rho_1) \rho_1^{n_1} (1 - \rho_2) \rho_2^{n_2} \dots (1 - \rho_k) \rho_k^{n_k} = \prod_{i=1}^k (1 - \rho_i) \rho_i^{n_i}$$

Chúng minh:

Trước hết chúng ta chỉ ra rằng:

$$p_n = C \rho_1^{n_1} \rho_2^{n_2} \dots \rho_k^{n_k}$$

thỏa mãn (4.1). Khi đó  $C$  sẽ được xác định:

$$C = \prod_{i=1}^k (1 - \rho_i)$$

Để chỉ ra rằng phương trình trên có thể thỏa mãn (4.1), chúng ta định nghĩa:

$$p_n = CR^n; R^n = \rho_1^{n_1} \rho_2^{n_2} \dots \rho_k^{n_k}$$

Thay thế vào (4.1):

$$CR^n \sum_{i=1}^k \frac{\gamma_i}{\rho_i} + CR^n \sum_{j=1}^k \sum_{i=1}^k \mu_i r_{ij} \frac{\rho_i}{\rho_j} + CR^n \sum_{i=1}^k \mu_i r_{i0} \rho_i = CR^n \sum_{i=1}^k \mu_i (1 - r_{ii}) + CR^n \sum_{i=1}^k \gamma_i$$

Chia cả hai vế cho  $CR^n$ , chúng ta nhận được:

$$\sum_{i=1}^k \frac{\gamma_i \mu_i}{\lambda_i} + \sum_{i \neq j} \mu_i r_{ij} \frac{\lambda_i \mu_j}{\lambda_j \mu_i} + \sum_i \mu_i r_{ij} \frac{\lambda_i}{\mu_i} = \sum_i (\mu_i - \mu_i r_{ii} + \gamma_i) \quad (4.3)$$

(4.2) lúc này tương đương với:

$$\lambda_j = \gamma_i + \sum_{\substack{i=1 \\ i \neq j}}^k r_{ij} \lambda_i + r_{jj} \lambda_j \text{ với } j=1, \dots, k$$

Hoặc:

$$\sum_{\substack{i=1 \\ i \neq j}}^k r_{ij} \lambda_i = \lambda_j - \gamma_j - r_{jj} \lambda_j$$

Thay thế vào (4.3), chúng ta nhận được:

$$\sum_i \frac{\gamma_i \mu_i}{\lambda_i} + \sum_j \frac{\mu_j}{\lambda_j} (\lambda_j - \gamma_j - r_{jj} \lambda_j) + \sum_i \mu_i r_{i0} \frac{\lambda_i}{\mu_i} = \sum_i (\mu_i - \mu_i r_{ii} + \gamma_i)$$

Phương trình trên có thể rút gọn lại như sau:

$$\sum_i \left( \frac{\gamma_i \mu_i}{\lambda_i} + \frac{\mu_i}{\lambda_i} (\lambda_i - \gamma_i - r_{ii} \lambda_i) + \lambda_i r_{i0} \right) = \sum_i (\mu_i - \mu_i r_{ii} + \gamma_i)$$

và cuối cùng nhận được:

$$\sum_i \lambda_i r_{i0} = \sum_i \gamma_i$$

Trong phương trình trên, vế trái là tổng luồng đi ra khỏi mạng hàng đợi, còn vế phải là tổng luồng đi vào mạng. Khi ở trạng thái tĩnh thì chúng phải bằng nhau.

Để xác định  $C$  chúng ta xuất phát từ:

$$\sum_{n_1=0}^{\infty} \dots \sum_{n_k=0}^{\infty} C \rho_1^{n_1} \dots \rho_k^{n_k} = 1$$

Từ (4.1) chúng ta suy ra được:

$$N_i = \frac{\rho_i}{1 - \rho_i}$$

với  $i=1, 2, \dots, k$ .

Theo định lý Little tại mỗi nút chúng ta nhận được:

$$T_i = \frac{1}{\lambda_i} N_i \text{ với } i=1,2,\dots,k.$$

Với mạng hàng đợi có các nút, không phải chỉ có một trạm phục vụ mà có nhiều trạm phục vụ (nút  $i$  có  $c_i$  trạm, mỗi trạm có tốc độ xử lý trung bình  $\mu_i$ ) và lưu ý rằng  $\rho_i = \lambda_i / \mu_i$  thì:

$$p_n^- = p_{p_m \dots p_{n_k}} = \prod_{i=1}^k \frac{\rho_i^{n_i}}{a_i(n_i)} p_{0i}$$

Trong đó:

$$a_i(n_i) = \begin{cases} n_i! (n_i \leq c_i) \\ c_i^{n_i - c_i} c_i! (n_i > c_i) \end{cases}$$

Và  $p_{0i}$  được xác định bởi:

$$\sum_{n_i=1}^{\infty} \frac{\rho_i^{n_i}}{a_i(n_i)} p_{0i} = 1$$

#### 4.4. MẠNG JACKSON ĐÓNG

Khi chúng ta có  $\gamma_i = 0$  và  $r_{i0} = 0$  với  $i=1,2,\dots,k$  thì sẽ có một mạng Jackson đóng. Tức là lúc này trong mạng hàng đợi, số yêu cầu là hữu hạn và được ký hiệu là  $N$ .

Với  $c_i = 1$ , khi hệ thống ở trạng thái tĩnh, phương trình cân bằng xác suất nhận được sẽ là:

$$\sum_{\substack{i=1 \\ i \neq j}}^k \mu_i r_{ij} p_{n,i+j}^- = \sum_{i=1}^k \mu_i (1 - r_{ii}) p_n^- \quad (4.4)$$

**Định lý 4.3:** Cho một mạng Jackson đóng với  $N$  yêu cầu nằm trong hệ thống. Khi đó phương trình cân bằng của hệ thống tĩnh sẽ được viết như sau:

Kết quả của phương trình (4.4) khi hệ thống ở trạng thái cân bằng sẽ là:

$$p_n^- = C \rho_1^{n_1} \dots \rho_k^{n_k} = C \prod_{i=1}^k \rho_i^{n_i}, \rho_i = \frac{\lambda_i}{\mu_i} \text{ với } i=1,2,\dots,k. \quad (4.5)$$

Trong đó  $C = C(N)$  được xác định như sau:

$$\frac{1}{C(N)} = \sum_{\substack{n_1, \dots, n_k \\ \sum_{i=1}^k n_i = N}} \rho_1^{n_1} \dots \rho_k^{n_k} = \sum_{\substack{n_1, \dots, n_k \\ \sum_{i=1}^k n_i = N}} \prod_{i=1}^k \rho_i^{n_i}$$

Chúng minh:

Tại mỗi nút  $i$ , tổng xác suất chuyển đến đến nút này sẽ bằng tổng xác suất ra khỏi nút này. Chúng ta sẽ có  $\gamma_i = 0$  với  $i = 1, 2, \dots, k$ . Từ phương trình (4.2):

$$\mu_i \rho_i = \sum_{j=1}^k \mu_j r_{ji} \rho_j \text{ với } i = 1, 2, \dots, k. \quad (4.6)$$

Tương tự mạng Jackson mở:

$$p_n = C \rho_1^{n_1} \dots \rho_k^{n_k} = C R^{\bar{n}}$$

Thay thế vào (4.4), và chia cho  $C R^{\bar{n}}$ :

$$\sum_{i \neq j=1}^k \mu_i r_{ij} \rho_i \frac{1}{\rho_j} = \sum_{i=1}^k \mu_i - \sum_{i=1}^k \mu_i r_{ii}$$

Tương đương với:

$$-\sum_{i=1}^k \mu_i r_{ii} + \sum_{j=1}^k \frac{1}{\rho_j} \sum_{i=1}^k \mu_i r_{ij} \rho_i = \sum_{i=1}^k \mu_i - \sum_{i=1}^k \mu_i r_{ii}$$

Khi sử dụng (4.6) và rút gọn, ta được:

$$\sum_{j=1}^k \frac{1}{\rho_j} \mu_j \rho_j = \sum_{i=1}^k \mu_i$$

Xét hai trường hợp đặc biệt sau:

+  $k = 2$  ta có:

$$\frac{1}{C_2(N)} = \sum_{n_1+n_2=N} \rho_1^{n_1} \rho_2^{n_2} = \sum_{n=0}^N \rho_1^{n_1} \rho_2^{N-n_1} = \frac{\rho_2^{N+1} - \rho_1^{N+1}}{\rho_2 - \rho_1}$$



+  $k = 3$  ta có:

$$\begin{aligned}\frac{1}{C_3(N)} &= \sum_{n_1+n_2+n_3=N} \rho_1^{n_1} \rho_2^{n_2} \rho_3^{n_3} = \sum_{n_1=0}^N \sum_{n_2=0}^{N-n_1} \rho_3^{N-n_1-n_2} \rho_1^{n_1} \rho_2^{n_2} = \\ &= \frac{\rho_1^{N+2}}{(\rho_3 - \rho_1)(\rho_2 - \rho_1)} + \frac{\rho_2^{N+2}}{(\rho_1 - \rho_2)(\rho_3 - \rho_2)} + \frac{\rho_3^{N+2}}{(\rho_2 - \rho_3)(\rho_1 - \rho_3)}\end{aligned}$$

Khi  $k \geq 3$  thì theo phương pháp đệ quy:

$$\frac{1}{C_k(N)} = \sum_{n=0}^N \frac{\rho_k^n}{C_{k-1}(N-n)}$$

Cuối cùng là trường hợp ở nút thứ  $i$  có  $c_i$  trạm phục vụ với  $c_i > 1$  và  $i=1, \dots, k$  thì:

$$p_n^- = p_{n_1 \dots n_k} = C(N) \prod_{i=1}^k \frac{\rho_i^{n_i}}{a_i(n_i)}$$

Trong đó  $a_i$  được định nghĩa như trên và  $C(N)$ :

$$\frac{1}{C(N)} = \sum_{\substack{n_1, \dots, n_k \\ \sum_{i=1}^k n_i = N}} \prod_{i=1}^k \frac{\rho_i^{n_i}}{a_i(n_i)}$$

## BÀI TẬP CHƯƠNG 4

1. Cho một hệ thống hàng đợi mở với  $N = 3$  nút, phục vụ theo phương pháp FIFO, thời gian phục vụ tuân theo phân bố mũ với các giá trị trung bình như sau:

$$1/\mu_1 = 0,08s; \quad 1/\mu_2 = 0,06s; \quad 1/\mu_3 = 0,04s$$

Các yêu cầu từ bên ngoài chỉ đi vào nút thứ nhất với phân bố mũ và tốc độ  $\lambda_{01} = 4 \text{ yêu cầu/s}$ . Nút 1 là một hệ thống với hai trạm phục vụ, nút 2 và 3 chỉ có một trạm phục vụ. Các xác suất định tuyến được xác định như sau:

$$p_{11} = 0,2; \quad p_{21} = 1; \quad p_{31} = 0,5;$$

$$p_{12} = 0,4; \quad p_{30} = 0,5;$$

$$p_{13} = 0,4.$$

- a. Hãy vẽ hệ thống hàng đợi này
  - b. Xác định xác suất ở trạng thái ổn định khi  $k = (4,3,2)$ .
  - c. Xác định các thông số hiệu năng của hệ thống.
2. Xác định hiệu suất sử dụng CPU và các tham số hiệu năng khác của một hệ thống máy tính với  $N = 3$  nút và  $K = 4$  yêu cầu. Mỗi nút có một trạm phục vụ và theo kiểu FIFO. Thời gian phục vụ tuân theo phân bố mũ với các thông số trung bình như sau:

$$1/\mu_1 = 2ms, 1/\mu_2 = 5ms, 1/\mu_3 = 5ms$$

và các xác suất định tuyến:

$$p_{11} = 0,3; \quad p_{12} = 0,5; \quad p_{13} = 0,2; \quad p_{21} = p_{31} = 1.$$

3. Xét một hệ thống mạng hàng đợi đóng với  $K = 3$  nút và  $N = 3$  yêu cầu, phục vụ theo phương pháp FIFO. Nút thứ nhất có hai trạm phục vụ và hai nút còn lại chỉ có một trạm phục vụ. Xác suất định tuyến là:

$$p_{11} = 0,6; p_{21} = 0,5; p_{31} = 0,4;$$

$$p_{12} = 0,3; p_{22} = 0; p_{32} = 0,6;$$

$$p_{13} = 0,1; p_{23} = 0,5; p_{33} = 0.$$

Thời gian phục vụ tuân theo phân bố mũ với tốc độ:

$$\mu_1 = 0,4s^{-1}; \mu_2 = 0,6s^{-1}; \mu_3 = 0,3s^{-1}.$$

- a. Vẽ mạng hàng đợi này.
- b. Xác suất để có hai yêu cầu ở nút số 2?
- c. Xác định các thông số hiệu năng?
4. Cho mạng hàng đợi đóng với  $N = 4$  nút và  $K = 3$  yêu cầu, thời gian phục vụ tuân theo phân bố mũ với tốc độ như sau:

$$\mu_1 = 0,72s^{-1}; \mu_2 = 0,64s^{-1}; \mu_3 = 1s^{-1}$$

và xác suất định tuyến là  $p_{31} = 0,4; p_{32} = 0,6; p_{13} = p_{23} = 1$ . Xác định các thông số hiệu năng của hệ thống.

## TÀI LIỆU THAM KHẢO

1. F. Baskett, K. M. Chandy, R. R. Muntz, F.G. Palacios. *Open, Closed and Mixed Networks of Queues with Different Classes of Customers*. JCAM22 (2) 248–260, 1975.
2. P. J. Burke. *The Output of Queueing System*. Operation Research 4, 699–714, 1956.
3. W. J. Gordon, G. –F. Newell. *Closed Queueing Systems with Exponential Servers*, Operation Research 15, 254–265, 1967.
4. J. R. Jackson. *Networks of Waiting Lines*. Operation Research 5, 518–521, 1957.
5. J. R. Jackson, Jobshop-like Queueing Systems. *Management Science* 10, 131–142, 1963.

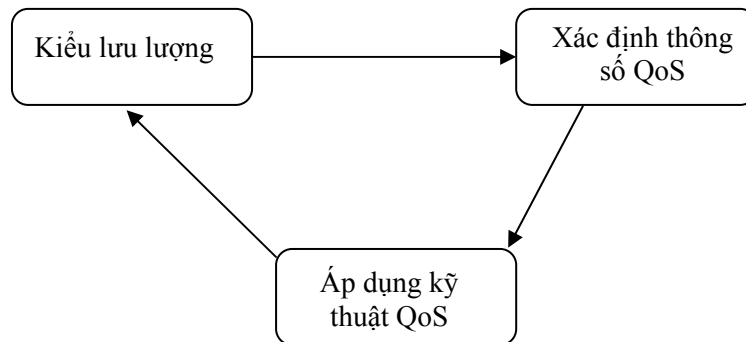
## Chương 5

# CHẤT LƯỢNG DỊCH VỤ (QUALITY OF SERVICE – QoS)

### 5.1. TẠI SAO PHẢI CUNG CẤP CHẤT LƯỢNG DỊCH VỤ QoS CHO MẠNG INTERNET?

Trước đây, khi Internet chủ yếu là truyền dữ liệu thì người ta không cần quan tâm đến việc phân biệt và ưu tiên cho các gói tin bởi vì lúc này băng thông mạng và các tài nguyên khác đủ để cấp cho các ứng dụng trong mạng. Vì vậy các nhà cung cấp dịch vụ Internet sẽ cung cấp cho khách hàng của họ dịch vụ *Best-Effort* (BE), loại dịch vụ mà tất cả các khách hàng đều được đối xử như nhau. Dịch vụ BE này là dịch vụ phổ biến nhất trên mạng Internet hay mạng *IP* nói chung. Các gói thông tin được truyền đi theo nguyên tắc đến trước được phục vụ trước mà không quan tâm đến đặc tính lưu lượng của dịch vụ là gì. Điều này dẫn đến việc rất khó hỗ trợ các dịch vụ đòi hỏi độ trễ thấp như các dịch vụ thời gian thực hay video.

Khác với các dịch vụ kinh điển của Internet như email, chia sẻ file hay *www*, ngày nay đang xuất hiện rất nhiều các loại hình dịch vụ mới khác đòi hỏi chất lượng của Internet phải đáp ứng cao hơn. Người ta phân chia ra hai loại dịch vụ cơ bản là các dịch vụ trong thời gian thực (*realtime service*) và các dịch vụ không phải trong thời gian thực (*non-realtime service*). Các dịch vụ trong thời gian thực có thể là thoại, video hay truyền dữ liệu và mỗi loại dịch vụ này đều có những đòi hỏi nhất định. Ví dụ như với dịch vụ thoại thì trễ của thông tin không được vượt quá một mức ngưỡng nào đó, còn với dịch vụ truyền video thì băng thông cũng không được quá nhỏ.



**Hình 5.1. Mô hình cung cấp chất lượng dịch vụ**

*Quality of Service (QoS)* là thuật ngữ được sử dụng để đo một tập các thuộc tính hoạt động của mạng liên quan tới một dịch vụ cụ thể nào đó. Trong môi trường mạng *IP*, *IPQoS* được xem là hoạt động của các gói tin *IP* chạy qua một hoặc nhiều mạng. Mục tiêu cao nhất của các nhà cung cấp dịch vụ là chuyển tải dịch vụ *IP* hỗ trợ *QoS* từ đầu cuối tới đầu cuối trên mạng *IP* bao gồm: dữ liệu, video, đa phương tiện và thoại... Khi triển khai các giải pháp *IPQoS*, các nhà cung cấp dịch vụ có thể:

- Tăng lợi nhuận: Tăng doanh thu bằng cách thu hút các khách hàng mới và giữ được khách hàng truyền thống. Cung cấp các dịch vụ chất lượng cao trong khi giảm chi phí qua việc sử dụng băng thông hiệu quả.
- Nâng cao sức cạnh tranh bằng cách đưa ra nhiều dịch vụ tốt hơn dịch vụ *BE* và các giải pháp theo yêu cầu riêng của khách hàng.

*QoS* được đặc trưng bởi tập các tham số và thông số như sau:

**Khả năng sẵn có dịch vụ:** là khả năng tin cậy của các kết nối từ người sử dụng tới mạng *IP*.

**Băng thông (*Bandwidth*):** Băng thông được coi là yếu tố quan trọng nhất để có thể đảm bảo được chất lượng dịch vụ và thường được đo bằng đơn vị bit–trên–giây (bps). Băng thông thường được hệ thống cung cấp dưới dạng tốc độ cố định (ví dụ như trong mạng Internet) hoặc tốc độ thay đổi (như mạng Frame Relay). Để có thể giảm được băng thông mà vẫn đảm bảo được chất lượng, người ta có thể sử dụng các biện pháp nén.

**Trễ (Delay):** được hiểu là khoảng thời gian tiêu hao từ lúc lưu lượng của mạng được phát ra ở bên gửi, truyền qua hệ thống đến bên nhận. Trễ xảy ra có thể do rất nhiều yếu tố gây nên, ví dụ do trễ trên đường truyền, trễ tại các bộ đệm của bộ định tuyến, trễ tại bộ đệm phía đầu thu, cụ thể như sau:

– Trễ nối tiếp: là khoảng thời gian cần thiết để một gói có thể được phát ra và truyền trên đường truyền vật lý. Theo định nghĩa này, trễ nối tiếp để có thể phát hết 128000 bit trên đường truyền có tốc độ 64000 bps sẽ là 2s.

– Trễ truyền dẫn: là khoảng thời gian cần thiết để một bit có thể truyền từ điểm đầu đến điểm cuối trên đường truyền vật lý.

– Trễ xử lý: là khoảng thời gian cần thiết để một bộ định tuyến hoặc thiết bị chuyển mạch có thể chuyển gói từ bộ đệm đầu vào sang bộ đệm đầu ra của nó. Thông số này bị ảnh hưởng bởi nhiều yếu tố, như tốc độ chuyển mạch của bộ định tuyến hay switch đó và kích thước của bảng định tuyến.

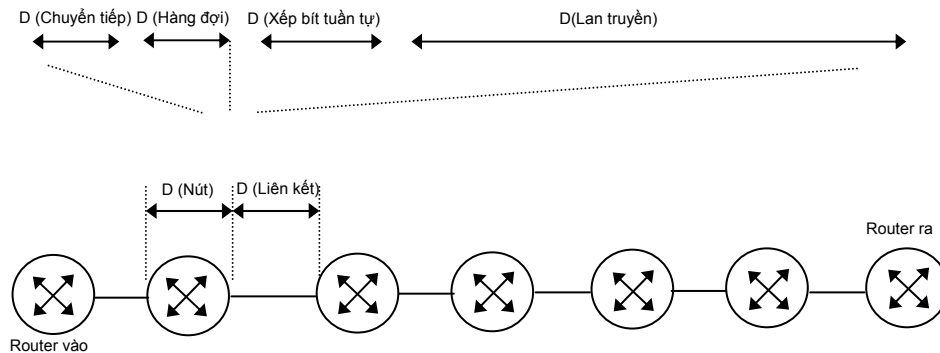
– Trễ hàng đợi: là khoảng thời gian một gói phải chờ trong bộ đệm cho đến khi được chuyển ra trên đường truyền. Nếu kích thước của bộ đệm quá nhỏ, thì sẽ có trường hợp bộ đệm bị đầy, lúc đó phải loại bỏ gói và làm ảnh hưởng tới chất lượng của ứng dụng. Còn khi kích thước của bộ đệm quá lớn thì sẽ có quá nhiều gói phải chờ và làm tăng trễ hàng đợi lên.

**Độ biến thiên trễ (Jitter):** có thể hiểu là sự biến đổi của trễ. Thông số này có sự ảnh hưởng đặc biệt quan trọng đến chất lượng của một số dịch vụ mang tính chất tương tác như thoại chẳng hạn. Có một số phương pháp có thể giảm được biến thiên trễ, đó là bộ đệm thu.

**Tỷ lệ mất gói (Loss ratio):** Tỷ lệ này cho biết có bao nhiêu gói bị mất khi truyền từ nguồn đến đích. Tỷ lệ này cũng đặc biệt ảnh hưởng đến chất lượng của một số dịch vụ, ví dụ như khi truyền thoại qua mạng IP, thông thường người ta đòi hỏi tỷ lệ mất gói không quá 1% và trễ không quá 150 ms.

**Mất trình tự gói (Sequence Error):** Nghẽn trên mạng chuyển mạch gói có thể khiến gói chọn nhiều tuyến khác nhau để đi đến đích. Gói có thể đến đích không đúng theo trình tự đã định trước làm cho tiếng nói bị đứt quãng.

Đây là các thuộc tính quan trọng cần phải được xem xét khi đánh giá chất lượng truyền tin. Đối với các dịch vụ nhạy cảm với thời gian thực, người ta thường đánh giá chất lượng qua hai tham số: trễ và độ biến thiên trễ.



**Hình 5.2. Trễ đầu cuối tới đầu cuối**

Có nhiều nhân tố đóng góp vào trễ khi gói tin được truyền trên mạng. Hình vẽ trên minh họa trễ từ đầu cuối tới đầu cuối là tổng các thành phần trễ: chuyển tiếp, trễ hàng đợi, trễ xếp bit tuần tự, trễ đường truyền tại mỗi nút và liên kết trong mạng.

*IETF* đã định nghĩa *QoS* như sau:

*QoS* là đặc trưng của dịch vụ chuyển gói cung cấp thông qua các tham số như tốc độ đạt được, trễ gói và tỉ lệ mất gói.

Để đạt được các thông số về chất lượng dịch vụ như mong muốn, có hai cách tiếp cận như sau:

- Cung cấp chất lượng dịch vụ dựa theo luồng riêng biệt (*flow-based*).
- Cung cấp chất lượng dịch vụ theo lớp (*class-based*).

Tương ứng với hai phương pháp tiếp cận này, người ta đưa ra hai mô hình cung cấp chất lượng dịch vụ khác nhau. Đó là:

- Mô hình dịch vụ tích hợp (*Intergrated Services*).
- Mô hình dịch vụ phân biệt (*Differentiated Services*).

Ngoài hai mô hình trên, để cung cấp chất lượng dịch vụ cho mạng Internet, còn phải kể đến các kỹ thuật lưu lượng và các phương pháp định tuyến có điều kiện.



Trước khi đi vào tìm hiểu các mô hình này, chúng ta sẽ sử dụng một số định nghĩa trong bảng sau:

**Bảng 5.1. Một số định nghĩa**

<i>Luồng</i>	Là dãy các gói với cùng địa chỉ <i>IP</i> nguồn, địa chỉ cổng nguồn, địa chỉ <i>IP</i> đích, địa chỉ cổng đích và cùng một giao thức.
<i>Service Level Agreement (SLA)</i>	Là thỏa thuận giữa khách hàng và nhà cung cấp dịch vụ. Thỏa thuận này sẽ xác định mức chất lượng dịch vụ mà khách hàng có thể nhận được.
<i>Traffic Profile</i>	Là các đặc tính mô tả tính chất của lưu lượng, ví dụ như tốc độ và trễ.
<i>Differentiated Services Code Point (DSCP)</i>	Là các bit được mã hóa của trường <i>TOS</i> của gói <i>IPv4</i> và trường <i>Traffic Class</i> của gói <i>IPv6</i> .
<i>Per-Hop-Behavior (PHB)</i>	Là các tác động từng chặng được thực hiện tại các bộ định tuyến lên các gói có các bit được thiết lập theo mã <i>DSCP</i> .
<i>Admission Control</i>	Là quá trình xem xét liệu có chấp nhận một yêu cầu về một mức chất lượng dịch vụ nào đó hay không.
<i>Classification</i>	Là quá trình phân loại gói dựa vào thông tin trong phần tiêu đề theo một tập các quy tắc đã biết trước.
<i>Policing</i>	Là quá trình xử lý lưu lượng, ví dụ như loại bỏ các gói vượt quá tốc độ định.
<i>Shaping</i>	Là quá trình nắn gói để lưu lượng tuân theo một tập quy tắc đã được định nghĩa trước.
<i>Scheduling</i>	Là quá trình phân hoạch để lựa chọn gói.
<i>Quản lý hàng đợi</i>	Là quá trình điều khiển độ dài hàng đợi khi cần.

## 5.2. MỘT SỐ MÔ HÌNH CUNG CẤP CHẤT LƯỢNG DỊCH VỤ

Để có thể cung cấp được chất lượng dịch vụ cho các ứng dụng đặc biệt như thoại hoặc ảnh, người ta đã xây dựng hai cấu trúc. Đó là cấu trúc dịch vụ tích hợp (*Integrated Services Architecture*) và cấu trúc dịch vụ phân biệt (*Differentiated Services Architecture*). Trước khi đi vào tìm hiểu hai mô hình cung cấp chất lượng dịch vụ này, chúng ta sẽ tìm hiểu mô hình *Best-Effort* là mô hình đã tồn tại từ trước đến nay của Internet.

### 5.2.1. Cấu trúc Best-Effort (BE)

*BE* là một mô hình dịch vụ đơn và phổ biến trên mạng Internet hay mạng *IP* nói chung, cho phép ứng dụng gửi dữ liệu bất cứ khi nào với bất cứ khối lượng nào nó có thể thực hiện mà không đòi hỏi sự cho phép hoặc thông tin cơ sở mạng, nghĩa là mạng phân phối dữ liệu nếu có thể mà không cần sự đảm bảo về độ tin cậy, độ trễ hoặc khả năng thông mạng.

Dịch vụ *BE* rất phù hợp cho những ứng dụng truyền số liệu như truyền file hoặc email.

### 5.2.2. Cấu trúc dịch vụ tích hợp (*Integrated Services Architecture-IntServ*)

Đứng trước nhu cầu ngày càng tăng trong việc cung cấp dịch vụ thời gian thực (thoại, video) và băng thông cao (đa phương tiện), cấu trúc dịch vụ tích hợp *IntServ* đã ra đời (được định nghĩa trong *RFC 1633*). Đây là sự phát triển của mạng *IP* nhằm đồng thời cung cấp dịch vụ truyền thống *BE* và các dịch vụ trong thời gian thực. Cấu trúc dịch vụ tích hợp này cho phép cung cấp chất lượng dịch vụ một cách chính xác (*hard QoS*) và theo luồng. Việc này được thực hiện bằng cách phải đặt trước băng thông bằng một giao thức báo hiệu đặc biệt, chính là giao thức *RSVP (Resource Reservation Protocol-RFC 2205)*.

Để có thể đáp ứng được các dịch vụ trong thời gian thực, *IntServ* dựa trên giao thức giữ trước tài nguyên *RSVP*. *IntServ* đưa ra nhiều khả năng với các mức điều khiển dịch vụ gói dữ liệu khác nhau cho các ứng dụng

lựa chọn. Ngoài ra, việc hỗ trợ *QoS* trong *IntServ* được thực hiện theo luồng. Để có thể thực hiện được điều này, *IntServ* yêu cầu kiến trúc phức hợp gồm phân loại, xếp hàng và định trình dọc theo một đường truyền bất kỳ từ biên đến biên. *IntServ* được phát triển dựa trên mô hình *BE* của *Internet* nhưng mở rộng cho các ứng dụng tương tác và thời gian thực. *IntServ* hỗ trợ cho hai lớp ứng dụng:

- Các ứng dụng thời gian thực có yêu cầu chặt chẽ về băng thông và trễ, mà người sử dụng không có được ở mạng chỉ hỗ trợ các dịch vụ nỗ lực cao nhất *BE*.

- Các ứng dụng truyền thống mà trong đó người sử dụng không phải quan tâm tới lưu lượng của những người sử dụng khác. Khi đó mạng được xem như một mạng *BE* có mức tải thấp.

*a. Động lực thúc đẩy sự ra đời của mô hình này*

- Dịch vụ *BE* không còn đủ đáp ứng nữa. Ngày càng có thêm nhiều ứng dụng khác nhau, các yêu cầu khác nhau về đặc tính lưu lượng được triển khai, đồng thời người sử dụng cũng yêu cầu chất lượng dịch vụ ngày càng cao hơn. Các ứng dụng đa phương tiện ngày càng xuất hiện nhiều.

- Mạng *IP* phải có khả năng hỗ trợ không chỉ đơn dịch vụ mà còn hỗ trợ đa dịch vụ của nhiều loại lưu lượng khác nhau từ thoại, số liệu đến video. Tối ưu hóa hiệu suất sử dụng mạng và tài nguyên mạng.

- Đảm bảo hiệu quả sử dụng và đầu tư. Tài nguyên mạng sẽ được dự trữ cho lưu lượng có độ ưu tiên cao hơn, phần còn lại sẽ dành cho số liệu *BE*.

- Mô hình *IntServ* cho phép nhà cung cấp mạng tung ra những dịch vụ tốt nhất khác biệt với các đối thủ cạnh tranh khác.

Mô hình *IntServ* được *IETF* đưa ra vào cuối những năm 90 với mục đích hỗ trợ chất lượng dịch vụ từ đầu cuối tới đầu cuối. Các ứng dụng nhận được băng thông đúng yêu cầu và truyền đi trong mạng với độ trễ cho phép. Trên thực tế giao thức *RSVP* là giao thức duy nhất dùng để báo hiệu cho mô hình *IntServ*. Ngoài giao thức báo hiệu, mô hình tích hợp dịch vụ còn định nghĩa thêm một số lớp dịch vụ. Một ứng dụng sẽ xác định đặc tính của luồng lưu lượng mà nó đưa vào mạng, đồng thời xác định một số yêu cầu về mức dịch vụ mạng.

*b. Các mô hình dịch vụ cung cấp bởi IntServ*

Cấu trúc này chỉ định nghĩa hai kiểu dịch vụ là dịch vụ đảm bảo *GS* (*Guaranteed Services*) và dịch vụ tải có điều khiển *CL* (*Controlled Load*).

Dịch vụ *GS* cho phép cung cấp trễ từ điểm đầu đến điểm cuối cho các ứng dụng một cách rất chặt chẽ, do đó thích hợp với các lưu lượng trong thời gian thực hoặc các ứng dụng trong thời gian thực. Trong khi đó, dịch vụ tải có điều khiển *CL* cho phép hỗ trợ trễ end-to-end một cách kém chặt chẽ hơn, vì vậy thích hợp với lưu lượng không phải trong thời gian thực.

Một cách cụ thể hơn nữa, các dịch vụ này được định nghĩa chi tiết như sau:

– Dịch vụ đảm bảo *GS*: Dịch vụ đảm bảo này cung cấp một giới hạn trễ hàng đợi nhất định, do đó dữ liệu sẽ tới đích sau một khoảng thời gian nhất định và sẽ không bị mất do tràn bộ nhớ. Loại hình dịch vụ này sẽ cố gắng xác định trễ trong trường hợp xấu nhất và nó không cố gắng để giới hạn hoặc xác định trễ cực tiểu, trễ trung bình hoặc độ biến thiên trễ. Độ biến thiên trễ, trong trường hợp này, sẽ được loại trừ bởi các bộ đệm *playout* ở phía bên thu. *GS* phù hợp nhất với các ứng dụng thời gian thực và có thể kể đến là: hội nghị truyền hình chất lượng cao, thanh toán tài chính cần độ trễ nhỏ thời gian thực.

– Dịch vụ tải có điều khiển *CL*: Đây là dịch vụ cung cấp chất lượng dịch vụ từ điểm đầu đến điểm cuối cho các ứng dụng mà một ứng dụng gần đúng với những gì nó nhận được từ dịch vụ *BE* trên một mạng tương đương trong điều kiện không tải. Lúc này có thể hiểu điều kiện không tải không có nghĩa là hoàn toàn không có tải, mà là không có tải nặng hoặc không tắc nghẽn. Vì vậy có thể nói loại hình dịch vụ này chỉ cung cấp mang định nghĩa định danh của đường truyền *BE* tương đương trong điều kiện không tắc nghẽn.

Vì vậy khi một luồng yêu cầu dịch vụ *CL* tại một thời điểm, có nghĩa là nó mong muốn được biết dịch vụ *BE* khi đi qua mạng không tải, không quan tâm tới các luồng lưu lượng khác đang được chuyển qua mạng trong cùng thời điểm đó. Dịch vụ *CL* có thể được xem như một mạng *BE* riêng

cho phép các ứng dụng khác nhau chia sẻ cùng một mạng, trong đó mức độ ảnh hưởng lẫn nhau giữa chúng được giảm xuống đáng kể.

*c. Đặt sẵn tài nguyên trong mạng*

Trong cấu trúc dịch vụ tích hợp này, người ta phải sử dụng quá trình đặt sẵn tài nguyên trong mạng theo luồng và yếu tố này là yếu tố quan trọng nhất dẫn đến ưu nhược điểm của nó. Điều này cũng dẫn tới việc mỗi một luồng tạo bởi một ứng dụng, cấu trúc dịch vụ tích hợp phải tiến hành thực hiện rất nhiều quá trình trao đổi để có thể đặt sẵn tài nguyên. Hình vẽ 5.3 mô tả các quá trình phải thực hiện tại một *bộ định tuyến* của cấu trúc này:

- Phân loại luồng (*Flow Classification*): là bộ phận có nhiệm vụ phân chia các gói về các luồng khác nhau.

- Lập lịch (*Scheduling*): Để có thể đảm bảo được về mặt trễ, băng thông..., cần thiết phải có các bộ lập lịch để định thời điểm gói được đưa ra đường truyền vật lý.

- Quản lý hàng đợi (*Queue Management*): là cơ chế phân chia tài nguyên của hàng đợi thành các phần khác nhau cho từng luồng.

- Điều khiển truy cập (*Admission Control*): là cơ chế cho biết các yêu cầu về chất lượng dịch vụ được đặt ra có thể được chấp nhận hay không.

- Đặt trước tài nguyên (*Resource Reservation*): Để có thể đảm bảo được chất lượng của từng luồng riêng biệt, cấu trúc dịch vụ tích hợp tiến hành đặt trước tài nguyên cho từng luồng trên mạng.

Các bộ phận này (phân loại luồng, lập lịch, quản lý bộ đệm, nắn lưu lượng...) là các thành phần quan trọng nhất trong cấu trúc dịch vụ tích hợp.

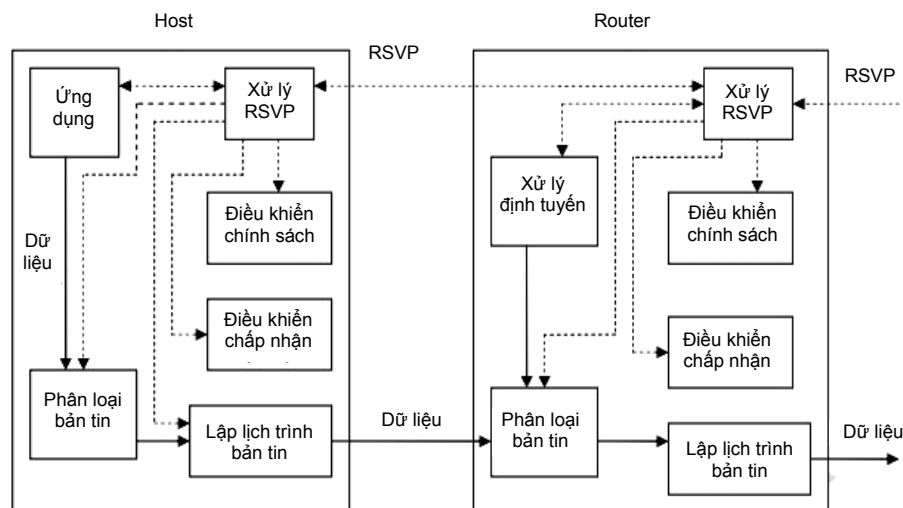
**Giao thức giữ trước tài nguyên RSVP:** là giao thức được sử dụng trong *IntServ* để thực hiện việc giữ trước tài nguyên trong mạng. Chức năng quan trọng nhất của *RSVP* là khi sử dụng nó kết hợp cùng với *IntServ* thì có thể thiết lập và duy trì trước tài nguyên một cách linh hoạt cho các luồng ứng dụng trong mạng.

*RSVP* không phải là giao thức định tuyến mà được thiết kế để hoạt động với giao thức định tuyến đơn hướng và đa hướng hiện tại. Các máy

trạm sẽ sử dụng giao thức này để yêu cầu một mức chất lượng dịch vụ từ mạng cho các dòng truyền tin của các ứng dụng.

Ngoài ra, còn có các trường hợp các bộ định tuyến sử dụng *RSVP* để chuyển các yêu cầu chất lượng dịch vụ tới các nút mạng trên đường truyền dữ liệu, cũng như để thiết lập và duy trì các mức dịch vụ yêu cầu. Dựa vào các yêu cầu của *RSVP*, lúc này tài nguyên sẽ được bảo lưu trên các máy trạm dọc theo đường truyền dữ liệu cũng như để thiết lập và duy trì các mức dịch vụ yêu cầu.

Ngoài ra, về mặt logic, *RSVP* sẽ phân biệt phía gửi và phía nhận. Bởi vì một ứng dụng có thể theo một hoặc hai hướng, nên cả hai hướng này đều phải sử dụng *RSVP* để yêu cầu lưu lượng trên đường truyền. *RSVP* hoạt động ở phía trên của *IPv4* hoặc *IPv6*, ở vị trí của giao thức giao vận trong các tầng giao thức.



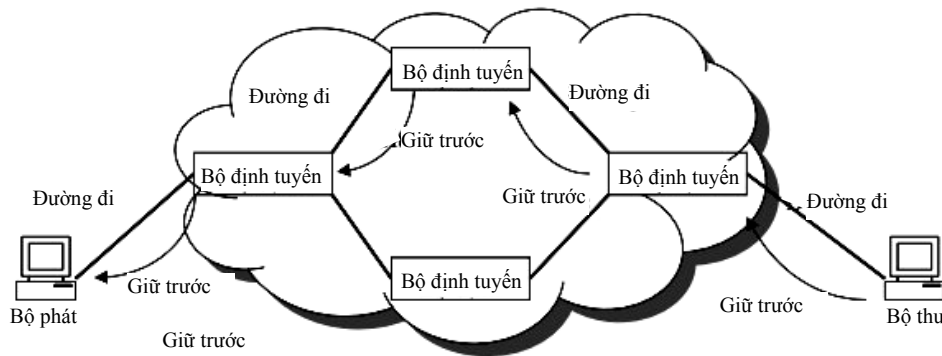
**Hình 5.3. Các chức năng RSVP tại Host và Router**

**Hoạt động của RSVP:** Thủ tục báo hiệu của *RSVP* được mô tả ở hình trên. Trong hình này chúng ta biểu diễn quá trình giữ trước tài nguyên khởi tạo ở bộ thu.

Trước tiên, bên phát sẽ mô tả lưu lượng bởi đặc tả lưu lượng *Tspec* liên quan đến cận trên và cận dưới của băng thông, trễ. Sau đó, bên phát sẽ gửi một bản tin đường đi *Path* chứa các thông tin này đến bộ thu.

Tại các bộ định tuyến trung gian, các bản tin đường đi này sẽ được chuyển tiếp đến bộ định tuyến tiếp theo bằng các giao thức định tuyến. Ở phía bên thu, sau khi tiếp nhận bản tin đường đi, sẽ gửi bản tin giữ trước mô tả đặc tả yêu cầu *Rspec* và đặc tả lọc *FilterSpec* ngược lại theo đường đã nhận bản tin đường đi. Đặc tả *Rspec* chỉ định loại dịch vụ còn đặc tả lọc mô tả các gói được giữ trước tài nguyên. *Rspec* và *Filter Spec* tạo thành tham số đặc tả luồng cho bộ định tuyến để xác nhận mỗi tài nguyên giữ trước.

Mỗi bộ định tuyến nhận bản tin giữ trước *Resv* sẽ yêu cầu quá trình điều khiển chấp nhận để tiếp nhận hoặc từ chối yêu cầu. Nếu yêu cầu được tiếp nhận, tài nguyên cần thiết như băng thông và bộ đệm được cấp phát cho luồng và đặc tả luồng được lưu trữ trong bộ định tuyến tương ứng. Bộ định tuyến cuối cùng sẽ gửi bản tin xác nhận quay trở lại bộ thu. Nếu bị từ chối, lỗi sẽ được trả về bộ thu.



**Hình 5.4. Hoạt động của RSVP**

**Ưu điểm và nhược điểm của IntServ/RSVP:** Cấu trúc dịch vụ tích hợp *IntServ/RSVP* có những ưu nhược điểm như sau:

Ưu điểm:

– *IntServ* giúp đảm bảo được chất lượng dịch vụ theo luồng. Đây là ưu điểm quan trọng nhất của *IntServ*.

– *IntServ* thực hiện báo hiệu các yêu cầu cho mỗi luồng riêng rẽ, hệ thống mạng, sau đó có thể cung cấp bảo đảm cho các lưu lượng cá biệt.

– *IntServ* báo cho các thiết bị mạng biết các tham số của lưu lượng như địa chỉ *IP* và cổng.

Nhược điểm:

– Số lượng luồng có thể lên đến hàng trăm ngàn luồng trong một thời điểm đối với mạng có lưu lượng cao hoặc các tổ chức doanh nghiệp lớn. Lúc này sẽ dẫn đến hiện tượng băng thông dành cho tín hiệu điều khiển tăng một cách đột biến do băng thông sử dụng để thiết lập kênh *RSVP* tăng lên rất nhiều.

– Nếu có quá nhiều luồng thì khả năng mở rộng là rất khó. Vì *IntServ* hoạt động theo kiểu kết nối trạng thái nên liên tục phải báo hiệu.

– Mặc dù *IntServ* là mô hình đảm bảo chất lượng dịch vụ tuyệt đối từ đầu cuối đến đầu cuối, nhưng nó không linh hoạt và khả năng mở rộng thấp nên thường không được lựa chọn để thực hiện *QoS* trong mạng có quy mô lớn.

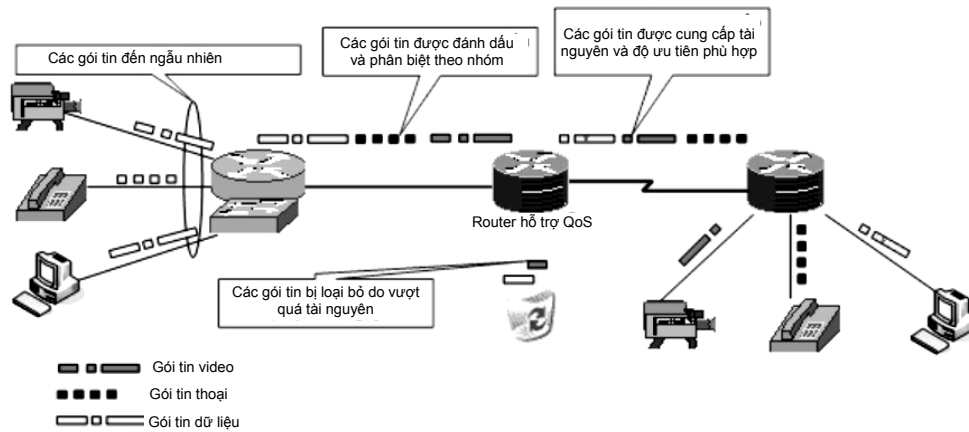
### **5.2.3. Cấu trúc dịch vụ phân biệt (Differentiated Services – DiffServ)**

Như chúng ta đã tìm hiểu ở trên, người ta đưa ra mô hình *IntServ* với mục đích giải quyết được nhiều vấn đề liên quan đến *QoS* trong mạng *IP*. Tuy nhiên sau khi triển khai *IntServ*, người ta thấy rằng nó không đảm bảo được *QoS* xuyên suốt. Điều này xảy ra do nhược điểm lớn nhất của *IntServ* là có quá nhiều luồng nên lượng thông tin điều khiển cho số luồng này cũng rất lớn. Chính vì vậy, người ta đưa ra mô hình *DiffServ* (*RFC 2475*) để khắc phục những hạn chế của mô hình *IntServ*. Khác với *IntServ*, mô hình *DiffServ* này có thể làm việc với số lượng luồng rất lớn.

Đặc điểm quan trọng nhất của mô hình *DiffServ* là khả năng linh hoạt cao và khả năng mở rộng lớn. Nguyên nhân các ưu điểm nổi bật này của *DiffServ* là do thay vì thực hiện chất lượng dịch vụ xuyên suốt và thống nhất trên cả đường truyền như mô hình *IntServ*, mô hình *DiffServ* thực hiện chất lượng dịch vụ riêng rẽ trên từng bộ định tuyến. Với cách thực hiện như vậy, mô hình *DiffServ* không cần phải tiến hành báo hiệu theo từng luồng



nên tiết kiệm băng thông và có khả năng mở rộng lớn, rất phù hợp trong mô hình hệ thống mạng lớn.



**Hình 5.5. Mô hình tổng quát của DiffServ**

#### *a. Nguyên tắc của DiffServ*

– Không giống như *IntServ* làm việc với từng luồng, *DiffServ* định nghĩa ra một số lượng nhỏ các lớp dịch vụ với các mức độ ưu tiên về đặc tính lưu lượng khác nhau.

– *DiffServ* định nghĩa lại các điểm mã dịch vụ phân biệt *DSCP* (*Differentiated Services Code Point*) bằng cách dùng trường *TOS* của trường tiêu đề của gói *IPv4*. Dựa trên các mã *DSCP* này, người ta sẽ định nghĩa ra các tác động từng chặng *Per Hop Behavior (PHB)* khác nhau.

– Tại biên của mạng, người ta sẽ tiến hành phân loại và đánh dấu trường *TOS* của tiêu đề của gói *IP* theo các *DSCP* khác nhau.

– Dựa vào các mã *DSCP* đã được đánh dấu trong phần tiêu đề của gói tin, các bộ định tuyến trong mạng lõi sẽ phục vụ các gói này theo các phương thức *PHB* khác nhau đã được định nghĩa từ trước.

#### *b. So sánh DiffServ với IntServ*

– *DiffServ* không bị mất nhiều băng thông cho phân báo hiệu vì nó không thực hiện báo hiệu cho từng luồng riêng biệt như *IntServ*.

- Các nhà cung cấp dịch vụ sẽ phân phối một số mức dịch vụ khác nhau cho các khách hàng có nhu cầu.

- Hỗ trợ rất tốt dịch vụ mạng riêng ảo *VPN*.

- Việc quản lý tài nguyên trong mô hình *DiffServ* được thực hiện một cách hiệu quả do không dành riêng tài nguyên cho một dịch vụ nào. Các dịch vụ được phân chia theo độ ưu tiên, dịch vụ nào có độ ưu tiên cao hơn sẽ được cung cấp tài nguyên ở chế độ tốt hơn, tài nguyên sẽ được trả về cho hệ thống và được sử dụng bởi các dịch vụ khác nếu không có dịch vụ hoạt động.

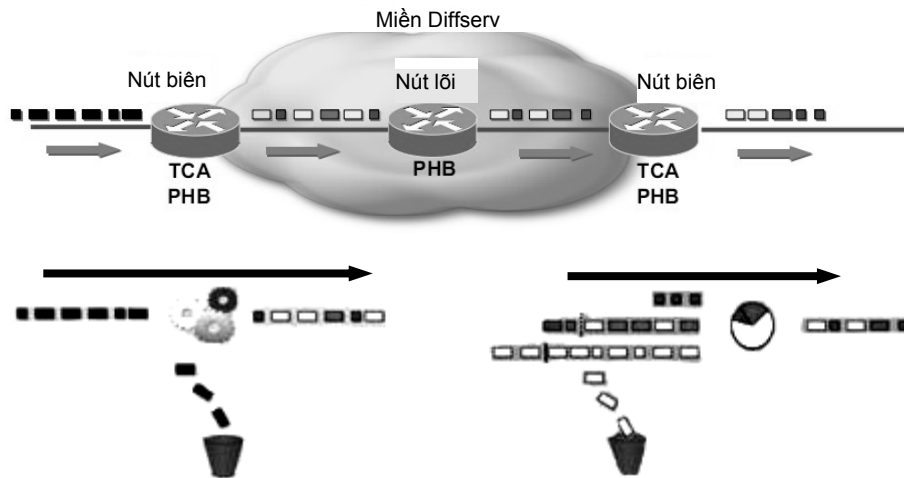
### *c. Nguyên lý hoạt động*

Trong mô hình *DiffServ*, các gói tin đầu tiên được chia ra thành nhiều nhóm (*class*), mỗi nhóm có độ ưu tiên từ thấp đến cao tùy theo đặc điểm của từng dịch vụ đó là gì. Sau đó các thiết bị sẽ tiến hành cung cấp tài nguyên theo từng nhóm. Các gói thuộc nhóm có thứ tự cao hơn sẽ nhận được nhiều tài nguyên ưu tiên hơn. Khi các nhóm có độ ưu tiên cao không dùng tài nguyên nữa thì tài nguyên này sẽ được các nhóm thấp hơn dùng. Tất cả các quá trình này sẽ được thực hiện riêng lẻ trên từng thiết bị.

Tại các bộ định tuyến biên, gói *IP* sẽ được phân loại. Bộ định tuyến biên thực hiện việc phân loại bằng cách mã hóa trường *DSCP*, cùng với một số dữ liệu khác liên quan đến luồng vi mô của gói *IP*, địa chỉ đầu gửi, địa chỉ đầu nhận. Sau đó, các thiết bị định tuyến biên sẽ áp dụng một số giải pháp điều chỉnh tiếp theo cho gói nếu cần thiết.

Các giải pháp được bộ định tuyến biên sử dụng có thể là đánh dấu gói, điều chỉnh gói, bao gồm: loại bỏ gói hoặc làm trễ gói trong một thời gian nhất định. Những tác động liên quan này mang mục đích nắn lại tính chất của luồng lưu lượng cho phù hợp với những tính chất đã được định nghĩa trước.

Khác với tại các bộ định tuyến biên, các bộ định tuyến lõi chỉ có nhiệm vụ kiểm tra chủng loại của gói *IP* và đơn giản chuyển tiếp gói *IP* theo cách chủng loại đó được nhận, bao gồm định tuyến cho gói hoặc sắp xếp gói vào bộ đệm thích hợp nếu cần thiết.



**Hình 5.6. Tác động từng chặng được thực hiện trong miền DiffServ**

*d. Các mức chất lượng dịch vụ cung cấp bởi DiffServ*

*DiffServ* chỉ định nghĩa một số giới hạn các lớp dịch vụ được chỉ ra bởi trường *DSCP*. Thứ hai, các hoạt động phân chia, đánh dấu và sắp xếp phức tạp chỉ cần ở biên của mạng. Nhà cung cấp dịch vụ lấy ra các bộ định tuyến là để thực thi sự phân loại hay còn gọi là *Behavior Aggregate-BA*.

**Dịch vụ có đảm bảo**

Dịch vụ có đảm bảo (*Assured Service*) là loại dịch vụ dành cho các khách hàng cần các đáp ứng tin cậy từ nhà cung cấp dịch vụ của họ, ngay cả khi mạng tắc nghẽn. Các *SLA* của dịch vụ có đảm bảo này sẽ xác định rõ lượng băng thông phân phối cho khách hàng. Các *SLA* cho các dịch vụ có đảm bảo thường là tĩnh, có nghĩa là các khách hàng có thể bắt đầu truyền dữ liệu bất cứ khi nào họ muốn mà không phải ra hiệu cho các nhà cung cấp dịch vụ của họ.

Dịch vụ có đảm bảo này có thể được thực hiện như sau. Trước tiên, tại các bộ định tuyến biên, các gói sẽ được phân loại. Nếu như lưu lượng của dịch vụ có đảm bảo này không vượt quá tốc độ đã được xác định trong *SLA* thì lưu lượng này sẽ được coi như lưu lượng *in profile*, còn nếu ngược lại thì được coi như *out profile*. Sau đó cả hai loại lưu lượng này sẽ được xếp vào

hàng đợi *Assured Queue AQ*. Hàng đợi này sẽ được quản lý bằng phương pháp *RED* hoặc *RIO*.

*RED (Random Early Detection)* là phương pháp quản lý hàng đợi tiên hành loại bỏ gói một cách ngẫu nhiên. Sử dụng *RED* sẽ giúp cho các luồng *TCP* trở nên không đồng bộ với nhau, do đó giúp cho việc tắc nghẽn không bị xảy ra. *RED* là phương pháp quản lý bộ đệm được sử dụng rất rộng rãi trong các bộ định tuyến ngày nay.

*RIO* cũng là phương pháp quản lý hàng đợi, tuy nhiên nó sử dụng hai thuật toán *RED* cho các lưu lượng *in profile* và *out profile*. Hai thuật toán *RED* này chỉ khác nhau ở mức ngưỡng thiết lập. Nếu như độ dài hàng đợi nhỏ hơn mức ngưỡng thứ nhất, thì không gói nào bị loại. Khi hàng đợi có độ dài nằm giữa hai mức ngưỡng, thì chỉ có các gói thuộc lưu lượng *out profile* mới bị loại bỏ một cách ngẫu nhiên. Nếu như kích thước hàng đợi vượt quá mức ngưỡng thứ hai, tức là tắc nghẽn có thể sắp xảy ra, thì các gói thuộc cả lưu lượng *in* và *out profile* sẽ bị loại bỏ một cách ngẫu nhiên nhưng các gói thuộc lưu lượng *out profile* sẽ bị loại bỏ nhiều hơn.

Bằng cách thực hiện như vậy, các gói thuộc lưu lượng *in profile* sẽ nhận được tỉ lệ mất gói thấp ngay cả khi có tắc nghẽn xảy ra và như vậy người sử dụng sẽ có thể biết trước được băng thông mà họ nhận được khi lưu lượng của họ không quá lớn. Khi không có tắc nghẽn xảy ra, các gói *out profile* cũng sẽ được chú ý đến.

### **Dịch vụ ưu tiên**

Dịch vụ ưu tiên (*Premium Service*) là loại dịch vụ đưa ra các độ trễ thấp và độ biến thiên trễ thấp cho các khách hàng có lưu lượng đỉnh cố định. Mỗi khách hàng sẽ có một *SLA* với nhà cung cấp dịch vụ của họ để chỉ rõ tốc độ bit tối đa mong muốn của họ. Khách hàng sẽ có trách nhiệm không vượt quá tốc độ tối đa, ngược lại lưu lượng vượt quá sẽ bị loại bỏ. Nhà cung cấp dịch vụ đảm bảo rằng băng thông đã giao ước sẽ sẵn sàng khi lưu lượng được gửi. Loại dịch vụ này đặc biệt thích hợp cho các ứng dụng *Internet Telephone, video...* Giá thành của nó cao hơn dịch vụ đảm bảo.

Vì giá thành của dịch vụ ưu tiên là cao, nên các nhà cung cấp dịch vụ cho phép sử dụng *SLA* tĩnh và *SLA* động. Các *SLA* động sẽ cho phép

khách hàng yêu cầu dịch vụ ưu tiên khi nào họ muốn dù cho trước đó họ chưa đăng ký dịch vụ này.

Dịch vụ ưu tiên này có thể được thực hiện như sau. Về phía khách hàng, tùy vào từng yêu cầu cụ thể mà sẽ có một số luồng của một số ứng dụng đòi hỏi dịch vụ ưu tiên. Các bộ định tuyến tiếp theo sẽ tiến hành phân loại và nắn chỉnh lưu lượng. Giả sử khi P bit của trường *DSCP* của phần tiêu đề của gói được thiết lập, tức là gói đó thuộc về dịch vụ ưu tiên. Ngược lại thì gói đó sẽ thuộc về dịch vụ đảm bảo hoặc chỉ là *BE*. Tất cả các gói được thiết lập P bit sẽ được xếp vào hàng đợi *Premium Queue (PQ)*, còn các gói còn lại sẽ được xếp vào hàng đợi *AQ* như ở trên. Các gói thuộc hàng đợi *PQ* sẽ được đưa ra đầu ra trước các gói còn lại và lưu lượng của các gói này có thể chiếm đến 100% lưu lượng ở đầu ra. Như vậy thì trễ và độ biến thiên trễ của các gói thuộc dịch vụ ưu tiên này cũng sẽ được đảm bảo rất nhỏ. Tuy nhiên, dịch vụ ưu tiên này cũng không đưa ra được một mức ngưỡng cụ thể cho trễ và độ biến thiên trễ mà nó sẽ cung cấp.

*e. Per Hop Behavior PHB*

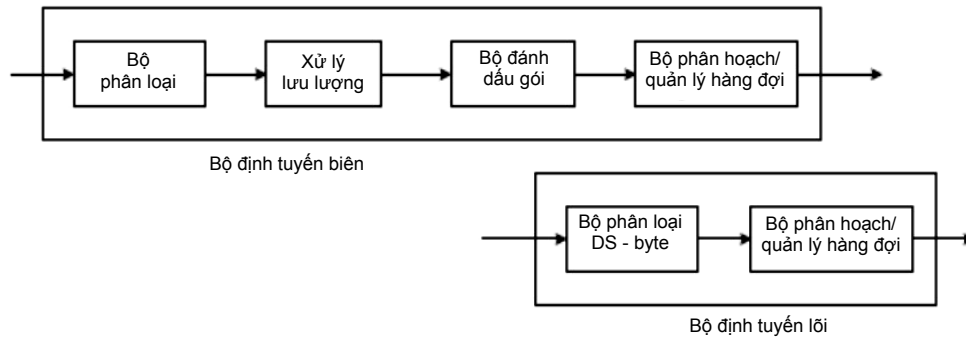
*DiffServ* đưa ra khái niệm tác động từng chặng *PHB* để định nghĩa lưu lượng thuộc về một tập đồng tác động nào đó được xử lý như thế nào tại một nút mạng riêng biệt. Tác động từng chặng được biểu diễn qua giá trị điểm mã *DSCP*. Người ta định nghĩa trường dịch vụ phân biệt thay thế cho trường loại dịch vụ trong *IPv4* và trường loại lưu lượng trong *IPv6*. Sáu bit trong này được dùng làm *DSCP* để chọn tác động từng chặng cho gói tin tại mỗi nút. *DSCP* có 64 giá trị khác nhau, nhưng số tác động từng chặng không bị giới hạn. Trong một vùng mạng, có sự chuyển đổi riêng giữa *DSCP* & *PHB*.

0	1	2	3	4	5	6	7
DSCP						CU	

DSCP: Điểm mã dịch vụ phân biệt.

CU (Currently Unused): Hiện không sử dụng.

**Hình 5.7. Trường dịch vụ phân biệt DSCP**



**Hình 5.8. Mô hình DiffServ tại nút biên và lõi mạng**

Trong đó mô hình bao gồm các phần:

- *DSCP-byte*: Byte xác định *DiffServ* là thành phần loại dịch vụ *ToS* của *IPv4* và trường lưu lượng *IPv6*. Các bit trong byte này thông báo gói tin mong đợi nhận được thuộc loại dịch vụ nào.

- Các bộ định tuyến biên: nằm tại lối vào hay lối ra của mạng cung cấp bởi dịch vụ *DiffServ*.

- + Khối *Multi-byte Classifier*: xử lý phân lớp luồng tin.
- + Khối *Policier*: xử lý luồng tin.
- + Khối *Packet Marker*: đánh dấu các gói tin.
- + Khối *Mngt/Scheduler*: định trình và quản lý hàng đợi.

- Các thiết bị trong mạng lõi.

- Quản lý cường bức: Các công cụ và nhà quản trị mạng giám sát và đo kiểm đảm bảo thỏa thuận lớp dịch vụ giữa mạng và người dùng.

Người ta định nghĩa ra hai nhóm *PHB*: chuyển tiếp nhanh *EF PHB* và chuyển tiếp được đảm bảo *AF PHB*.

#### **Chuyển tiếp nhanh *EF-PHB***

*EF PHB (Expedited Forwarding – PHB)* được thiết kế để cung cấp dịch vụ đầu cuối tới đầu cuối có băng thông được bảo đảm, trễ, tổn thất và độ biến thiên trễ thấp. Trên thực tế, *EF PHB* mô phỏng một đường ảo để hỗ trợ dịch vụ video hoặc thoại.

Bởi vì chỉ có thể kiểm soát trễ hàng đợi trên mạng, nên chỉ có thể tối thiểu được trễ và độ biến thiên trễ thông qua việc giảm thiểu trễ xếp hàng, vì vậy ý tưởng của *EF PHB* là sắp xếp để các gói được đánh dấu vào các hàng

đội rỗng hoặc ngắn. Điều này chỉ có thể đạt được khi tốc độ dịch vụ của gói tin *EF* trên một cổng ra nào đó lớn hơn tốc độ của các gói tin đến cổng đó, độc lập với tải trên các *PHB* khác. Một *EF PHB* yêu cầu mọi bộ định tuyến dọc theo đường truyền luôn phục vụ các gói *EF* nhanh hơn tốc độ ở các hop trước đó. Điều này dẫn đến các yêu cầu sau:

- Kiểm soát tốc độ lưu lượng *EF* ở đầu vào miền *DiffServ*.
- Xác định thời gian phục vụ gói *EF* ở mọi bộ định tuyến lõi. Thời gian phục vụ gói *EF* không bị ảnh hưởng bởi các tải khác.

*EF PHB* có thể được hỗ trợ trên các bộ định tuyến *DiffServ* theo một số cách sau:

- Kiểm soát luồng dữ liệu *EF* tới một giá trị nào đó tại biên của mạng *DiffServ*.
- Đảm bảo băng thông cần thiết dọc theo mạng lõi.
- Đặt các gói tin *EF* vào hàng đợi có độ ưu tiên cao nhất và đảm bảo rằng tốc độ ra ít nhất bằng với tốc độ vào.
- Hạn chế tải cùng *EF* trong mạng lõi để tránh thiếu băng thông cho các loại dịch vụ khác.

Thông thường người ta sử dụng thuật toán loại bỏ gói ngẫu nhiên *RED* khi hỗ trợ *EF PHB* vì phần lớn lưu lượng là *UDP* mà *UDP* không phức tạp bỏ gói tin bằng cách giảm tốc độ truyền. Mặc dù trong thực tế các gói *EF* sẽ được gửi đến một hàng đợi xác định phù hợp cho định trình, nhưng định nghĩa của dịch vụ *EF* cũng chỉ ra rằng hàng đợi nên có kích thước nhỏ.

Vì vậy, *EF PHB* phù hợp cho các dịch vụ yêu cầu trễ thấp, xác suất mất gói thấp, độ biến thiên trễ thấp. *RFC 2598* khuyến nghị *EF* nên được sử dụng để xây dựng đường thuê kênh riêng ảo. Nó tạo ra các dải nhỏ băng thông được bảo vệ khỏi những người sử dụng khác. Do các luồng lưu lượng riêng biệt sử dụng dịch vụ *EF* luôn được kiểm soát chặt chẽ tại đầu vào của DS tên miền, nên xác suất mất gói đối với lưu lượng *EF* là rất thấp.

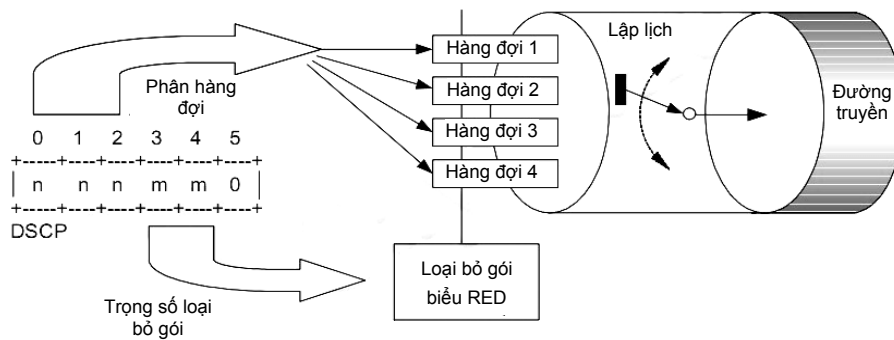
### **Chuyển tiếp đảm bảo AF – PHB (*Assured PHB*)**

Chuẩn *RFC 2597* định nghĩa *AF* là một nhóm *PHB* dành cho các dịch vụ được xác định theo băng thông khả dụng và đặc tính loại bỏ gói. Trong khi *EF* hỗ trợ các dịch vụ với băng thông và đặc tính độ biến thiên trễ cứng thì *AF PHB* cho phép chia sẻ các nguồn tài nguyên mạng linh hoạt

hơn. *AF PHB* hỗ trợ các dịch vụ có lưu lượng bùng nổ trong thời gian ngắn. *RFC 2597* cũng định nghĩa bốn nhóm *AF PHB*, mỗi nhóm hỗ trợ một phương thức *AF* khác nhau. Đối với *AF*, giá trị *DSCP* chỉ thị nó được chia làm hai phần. Phần thứ nhất, lớp dịch vụ cho phép xác định hàng đợi phù hợp với nó. Phần thứ hai, thứ tự trước sau xác định mức độ ưu tiên loại bỏ gói.

Tại một bộ định tuyến của *DiffServ*, mức *AF PHB* phụ thuộc vào:

- Dung lượng tài nguyên ấn định cho loại *AF PHB* mà gói tin thuộc về.
- Tải hiện tại của loại *AF PHB*, trong trường hợp có nghẽn trong loại *AF* đó.
- Giá trị tuần tự bỏ qua của gói tin.



**Hình 5.9. Chuyển tiếp được bảo mã hóa lớp dịch vụ và mức ưu tiên loại bỏ gói**

Hiện tại *RFC 2597* định nghĩa bốn lớp dịch vụ và ba mức ưu tiên như trong bảng sau:

**Bảng 5.2. Các loại AF**

Mức ưu tiên	Loại 1	Loại 2	Loại 3	Loại 4
Thấp	001010	010010	011010	100010
Trung bình	001100	010100	011100	100100
Cao	001110	010110	011110	100110

Người ta không định nghĩa dịch vụ cụ thể mà *AF PHB* được xem như một cơ chế cho phép nhà cung cấp dịch vụ đưa ra các mức phân biệt về độ



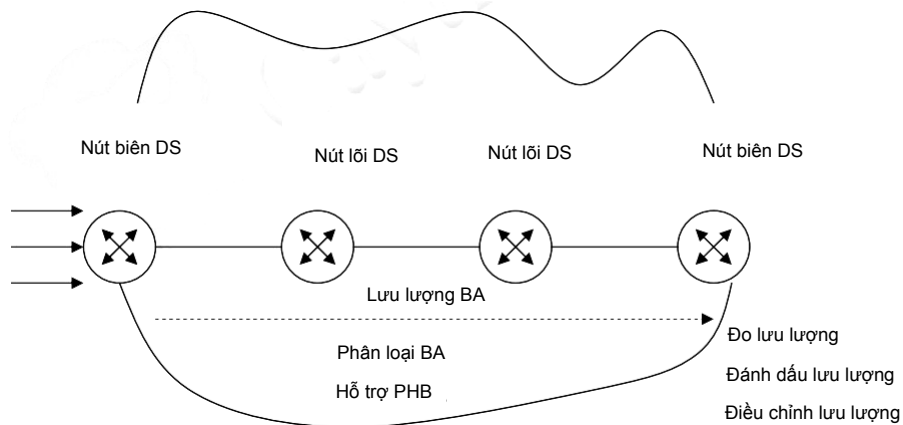
đảm bảo chuyển tiếp cho gói IP. Mạng *DiffServ* sẽ quyết định sự khác nhau giữa các loại *AF*. Các bộ định tuyến hỗ trợ nhóm *AF PHB* bằng cách:

- Kiểm soát luồng dữ liệu *AF* tới một giá trị nào đó tại biên của mạng *DiffServ*.

- Đảm bảo băng thông cần thiết dọc theo mạng lõi.
- Đặt mỗi loại dịch vụ *AF* vào các hàng đợi khác nhau.
- Lựa chọn nguyên tắc lập lịch trình hàng đợi thích hợp.

*f. Vùng mạng dịch vụ phân biệt DiffServ*

Vùng mạng *DiffServ* là một tập liên tiếp các bộ định tuyến hoạt động với cùng một chính sách triển khai dịch vụ. Thông thường một vùng mạng *DiffServ* chỉ do một nhà khai thác mạng quản lý để đảm bảo đủ tài nguyên mạng hỗ trợ các đặc tính lớp dịch vụ (*SLS – Service Level Specification*) và đặc tính kiểm tra lưu lượng (*TCS – Traffic Conditioning Specification*).



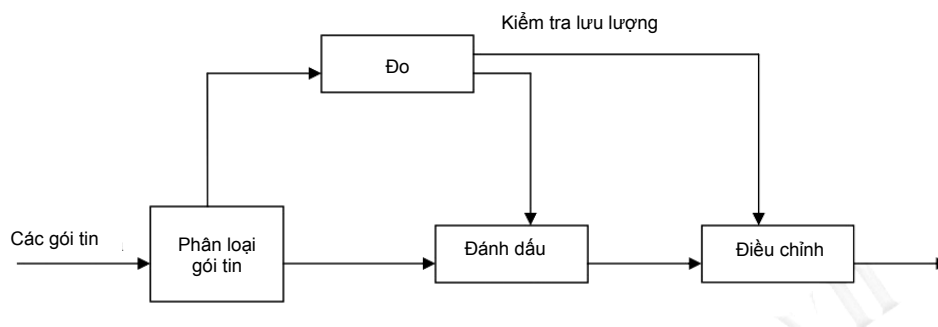
**Hình 5.10. Mạng DiffServ**

Một mạng *DiffServ* sẽ bao gồm các bộ định tuyến biên và lõi:

- Bộ định tuyến biên đặt tại biên của mạng. Các nút biên này đóng vai trò cả nút biên vào và nút biên ra. Khi đóng vai trò nút biên vào, bộ định tuyến biên có chức năng phân loại, đánh dấu và kiểm tra lưu lượng vào. Bộ định tuyến biên phân loại gói tin dựa trên kiểm tra tiền tố gói tin và viết

điểm mã dịch vụ phân biệt *DSCP* để chỉ ra nhóm *PHB* được hỗ trợ trong mạng *DiffServ*. Khi đóng vai trò nút biên ra, bộ định tuyến thực hiện các chức năng kiểm tra lưu lượng chuyển đi tới mạng *DiffServ* khác hoặc mạng không hỗ trợ *DiffServ*.

Nút lõi chọn tác động chuyển tiếp cho mỗi gói tin dựa trên *DSCP* của gói tin và chuyển gói tin tới nút lõi khác hoặc nút biên khác.



**Hình 5.11. Phân loại gói tin và kiểm tra lưu lượng**

g. *Ưu nhược điểm của DiffServ*

Ưu điểm của *DiffServ*:

- *DiffServ* không yêu cầu báo hiệu cho từng luồng, bắt tay khi thiết lập luồng nên không bị mất băng thông cho phần báo hiệu.

- *DiffServ* có thể triển khai diện rộng, vì vậy phù hợp trong mô hình hệ thống mạng lớn.

- *DiffServ* cung cấp nhiều cấp độ về chất lượng dịch vụ trong mạng.

- Dịch vụ ưu tiên có thể áp dụng cho một số luồng riêng biệt cùng một lớp dịch vụ. Điều này cho phép nhà cung cấp dịch vụ dễ dàng phân phối một số mức dịch vụ khác nhau cho các khách hàng.

- *DiffServ* không yêu cầu thay đổi tại các máy chủ hay các ứng dụng để hỗ trợ dịch vụ ưu tiên. Đây là nhiệm vụ của thiết bị biên.

Nhược điểm của *DiffServ*:

- Không đảm bảo hoàn toàn chất lượng dịch vụ.

– *DiffServ* yêu cầu một tập hợp các cơ chế để làm việc và liên quan đến việc truyền tải trong mạng. Vấn đề quản lý trạng thái của một số lượng lớn các thiết bị biên là một vấn đề không nhỏ.

– *DiffServ* không có khả năng cung cấp băng thông và độ trễ đảm bảo như *IntServ*.

– Chính sách khuyến khích khách hàng trên cơ sở giá cước cho dịch vụ cung cấp cũng ảnh hưởng đến giá trị của *DiffServ*.

#### *h. Ví dụ minh họa*

Trong phần này, chúng ta sẽ xem xét một vài ví dụ để hiểu được quá trình thiết lập dịch vụ từ điểm đầu đến điểm cuối của mô hình *DiffServ*.

Trong hình vẽ 5.12, máy *S* của mạng *CN1* muốn sử dụng dịch vụ đảm bảo để gửi dữ liệu đến máy *D* của mạng *CN2*. *CN1* sẽ đàm phán một *SLA* tính với nhà cung cấp dịch vụ *ISPI*. Quá trình thiết lập dịch vụ đảm bảo này được diễn ra như sau:

*Bước 1.* Máy *S* gửi một bản tin *RSVP* đến bộ *Bandwidth Broker (BB)* của *CN1* để yêu cầu dịch vụ đảm bảo.

*Bước 2.* Nếu bộ *Bandwidth Broker* của *CN1* chấp nhận yêu cầu này, nó sẽ điều khiển bộ định tuyến *LRI* thiết lập bit thứ *A* của gói và đồng thời gửi trả lời đồng ý đến máy *S*. Nếu không chấp nhận được yêu cầu này, ngược lại bộ *Bandwidth Broker* của *CN1* sẽ gửi bản tin báo lỗi đến máy *S*.

*Bước 3.* Máy *S* gửi gói đến bộ định tuyến *LRI*.

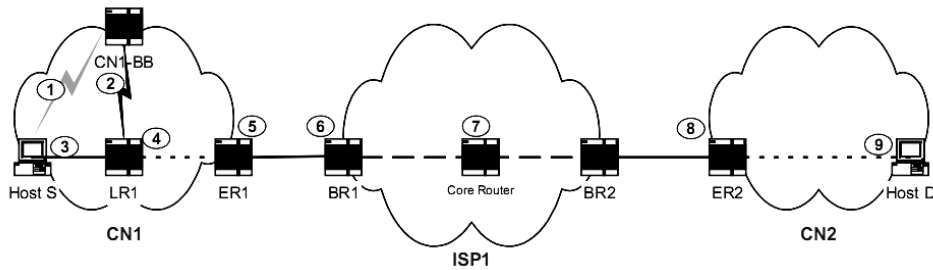
*Bước 4.* *LRI* sẽ thiết lập bit *A* của các gói thuộc dịch vụ đảm bảo.

*Bước 5.* Tất cả các bộ định tuyến từ *LRI* đến *ERI* sẽ tiến hành phân loại gói. Các gói có bit *A* được thiết lập sẽ được coi là *in profile*, ngược lại là *out profile*. Tất cả các gói sẽ được xếp vào hàng đợi *AQ* và hàng đợi này được xử lý bằng phương pháp quản lý *RIO*.

*Bước 6.* *BRI* thực hiện nắn lưu lượng. Từ *BRI* cho đến *ERI* cũng sẽ thực hiện phân loại, sử dụng hàng đợi *AQ* và phương pháp quản lý *RIO*.

*Bước 7.* *ER2* thực hiện các bước giống *BRI*.

*Bước 8.* Gói được gửi đến máy *D*.



**Hình 5.12. Ví dụ minh họa hoạt động của DiffServ**

Tiếp theo là ví dụ thiết lập dịch vụ ưu tiên với *SLA* động. Hình 5.13 minh họa các bước thực hiện của quá trình này.

### **Giai đoạn 1. Báo hiệu**

*Bước 1.* Máy B gửi bản tin *RSVP PATH* đến bộ *Bandwidth Broker* của CN1.

*Bước 2.* Nếu CN1-BB không chấp nhận dịch vụ này, nó sẽ báo lỗi đến máy S.

*Bước 3.* Nếu CN1-BB chấp nhận dịch vụ này, CN1-BB sẽ gửi bản tin *PATH* đến ISP1-BB.

*Bước 4.* Nếu ISP1-BB không chấp nhận đề nghị của CN1-BB, một bản tin lỗi sẽ được gửi đến CN1-BB, sau đó lại được gửi tiếp đến máy S. Nếu ISP1-BB chấp nhận yêu cầu của CN1-BB, ISP1-BB sẽ gửi bản tin *PATH* đến CN2-BB.

*Bước 5.* CN2-BB sẽ xem xét chấp nhận yêu cầu này của ISP1-BB. Nếu không chấp nhận, CN2-BB sẽ gửi bản tin lỗi đến ISP1-BB và sau đó đến máy S. Nếu CN2-BB chấp nhận yêu cầu này, nó sẽ sử dụng *RSVP* để báo cho ER2 biết phân loại và nắn gói. CN2-BB cũng sẽ gửi bản tin *RSVP RESV* đến ISP1-BB.

*Bước 6.* Nếu ISP1-BB nhận được bản tin *RESV*, nó sẽ thiết lập phân loại và nắn gói tại bộ định tuyến BR1 và BR2. Sau đó nó gửi bản tin *RESV* đến CN1-BB.

*Bước 7.* Khi CN1-BB nhận được bản tin *RESV*, nó sẽ thiết lập phân loại và nắn gói tại LR1 và ER1. CN1-BB sẽ gửi bản tin *RESV* đến máy S.

*Bước 8.* Nếu máy S nhận được bản tin *RESV*, nó sẽ bắt đầu truyền dữ liệu.

## Giai đoạn 2. Truyền dữ liệu

*Bước 9.* Máy *S* gửi gói đến *LR1*.

*Bước 10.* *LR1* phân loại gói. *LR1* cũng sẽ thiết lập bit *P* của gói và xếp vào hàng đợi *PQ*.

*Bước 11.* Tất cả các bộ định tuyến giữa *LR1* và *ER1* cũng sẽ tiến hành phân loại và xếp gói vào hàng đợi *PQ*.

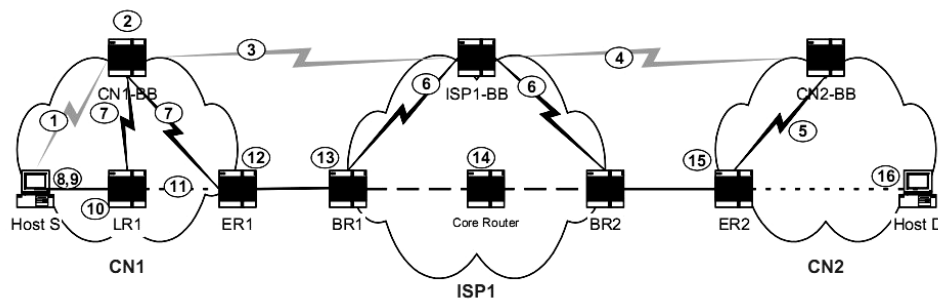
*Bước 12.* *ER1* tiến hành phân loại và nắn gói để đảm bảo rằng tốc độ đỉnh không bị phá vỡ.

*Bước 13.* *BR1* tiến hành phân loại và nắn gói.

*Bước 14.* Các bộ định tuyến giữa *BR1* và *BR2* cũng tiến hành phân loại và nắn gói.

*Bước 15.* *ER2* cũng tiến hành phân loại và nắn gói.

*Bước 16.* Gói được chuyển đến máy *D*.



**Hình 5.13. Ví dụ minh họa hoạt động của DiffServ**

### 5.2.4. MPLS (Multiprotocol Label Switching)

*MPLS* là một giao thức hoạt động ở giữa lớp hai và lớp ba của mô hình *OSI*. Trong giao thức này, mỗi một gói *MPLS* sẽ có một phần tiêu đề. Phần tiêu đề này dài 20 bit, trong đó có 3 bit là định nghĩa lớp dịch vụ *Class of Service CoS*, 1 bit định nghĩa kiểu nhãn và 8 bit của trường *TTL*. Các bộ định tuyến có khả năng làm việc với giao thức *MPLS*, thường được gọi là *Bộ định tuyến Label Switching (LSR)*, sẽ chỉ quan tâm đến các nhãn khi phải chuyển tiếp gói. Do đó *MPLS* có thể hoạt động tương thích với nhiều giao thức khác nhau nên được gọi là giao thức chuyển mạch nhãn.

Để có thể hoạt động được, *MPLS* cần phải thiết lập một tuyến đường dựa trên các nhãn gọi là *Label Switched Paths (LSPs)*. Các bộ định tuyến *LSR* sẽ phải đàm phán với nhau về cách thức xử lý từng loại nhãn tại các bộ định tuyến này sẽ được thực hiện như thế nào. Các tuyến đường *LSP* này có thể được thiết lập giống như các tuyến đường từng chặng tại lớp ba, hoặc các bộ định tuyến có thể định nghĩa một tuyến đường mặc định (*Explicit Route*) *ER* cho các *LSP*. Dựa vào các tuyến đường này, một bảng chuyển tiếp sẽ được tạo ra và giúp các bộ định tuyến biết phải làm gì khi gói đến mang một nhãn xác định.

Tại các bộ định tuyến biên, các gói sẽ được phân loại và sau đó sẽ được chèn thêm phần tiêu đề *MPLS* hay chính là các nhãn. Khi một bộ định tuyến *LSR* nhận được một gói có một nhãn xác định, nó sẽ sử dụng nhãn đó để tìm kiếm trong bảng chuyển tiếp của nó. Quá trình tìm kiếm dựa trên nhãn này diễn ra nhanh hơn nhiều so với quá trình tìm kiếm trong bảng định tuyến của các *IP* bộ định tuyến. Sau đó các gói có nhãn xác định sẽ được xử lý theo một số phương pháp đã được định nghĩa từ trước. Các nhãn của gói ở đầu vào sẽ được thay thế bằng các nhãn khác và được chuyển tới các *LSR* tiếp theo. Trong các mạng *MPLS*, quá trình chuyển tiếp gói, phân loại và chất lượng dịch vụ sẽ được thực hiện dựa trên nhãn và trường *COS* của nhãn. Điều này làm cho các bộ định tuyến *LSR* trở nên đơn giản hơn. Trước khi ra khỏi mạng *MPLS*, nhãn *MPLS* của gói sẽ được loại bỏ khỏi gói này.

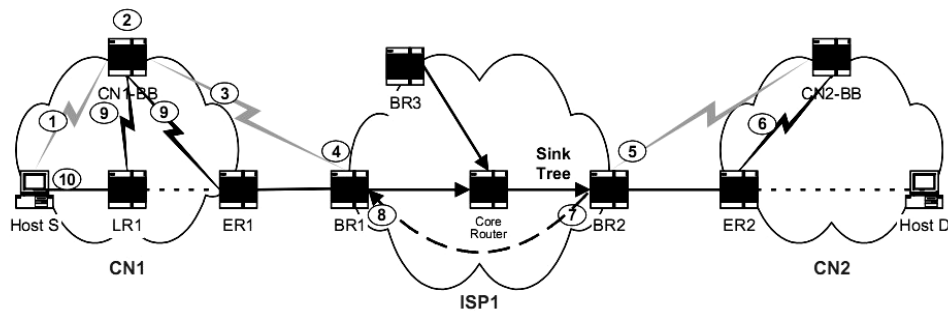
#### **5.2.4.1. Mô hình dịch vụ của MPLS**

*MPLS* là một giao thức có thể được sử dụng cùng với mô hình *DiffServ* để cho phép cung cấp chất lượng dịch vụ. Khi đó, các tuyến *LSP* sẽ được thiết lập trước hết giữa các bộ định tuyến biên ở đầu vào. Với các *LSP* từ *LSR1* đến *LSR2* và *LSP* từ *LSR2* đến *LSR1* là đối xứng nhau. Ngoài ra, với mỗi cặp bộ định tuyến biên vào ra, các tuyến *LSP* sẽ được thiết lập riêng biệt. Như vậy khi có  $C$  loại lưu lượng và  $N$  là số bộ định tuyến biên, thì sẽ có  $C \cdot N \cdot (N - 1) / 2$  tuyến *LSP* được thiết lập. Để có thể giảm thiểu được số *LSP* này, người ta có thể nhóm các tuyến từ tất cả các bộ định tuyến biên đầu vào đến một bộ định tuyến biên ở đầu ra thành một cây, gọi là *Sink Tree*. Số lượng cây này sẽ là  $C \cdot N$ . Các cây (*Sink Tree*) này có thể được sử dụng để truyền lưu lượng thuộc về một lớp dịch vụ và có thể dùng trường

*COS* của nhãn *MPLS* để phân biệt các lớp dịch vụ này với nhau. Trong trường hợp này, số cây sẽ giảm xuống còn là  $N$ . Bằng cách thực hiện như vậy, trong mô hình này, khi số luồng tăng lên thì số luồng tại các *LSP* hoặc các cây cũng tăng lên, nhưng số *LSP* và số các cây thì không tăng. Mô hình này do đó có tính mở rộng cao hơn các mô hình *IntServ*.

Hoạt động của mô hình này tương tự như hoạt động của mô hình *DiffServ* và chỉ khác với mô hình *DiffServ* ở ba điểm sau:

- Tại các bộ định tuyến biên đầu vào, các nhãn *MPLS* sẽ được tạo thêm cho các gói.
  - Tại các bộ định tuyến lõi, các bộ định tuyến lõi sẽ xử lý gói chỉ dựa vào nhãn *MPLS* này và trường *COS*.
  - Tại các bộ định tuyến biên đầu ra, các nhãn *MPLS* sẽ được bóc tách ra.
- Cụ thể chúng ta xem xét ví dụ ở hình 5.14 sau:



**Hình 5.14. Ví dụ minh họa hoạt động của MPLS**

### Giai đoạn báo hiệu

**Bước 1.** Máy *S* gửi bản tin *RSVP PATH* đến *Bandwidth Broker BB* của *CN1*.

**Bước 2.** *CN1-BB* sẽ quyết định có chấp nhận yêu cầu này hay không. Nếu không đồng ý, một bản tin báo lỗi sẽ được gửi quay trở lại máy *S* và quá trình báo hiệu kết thúc.

**Bước 3.** Nếu *CN1-BB* đồng ý chấp nhận yêu cầu, *CN1-BB* sẽ gửi bản tin *PATH* đến *BR1*.

**Bước 4.** *BR1* sẽ kiểm tra xem có đủ tài nguyên để gửi lưu lượng do *CN1* yêu cầu này đến bộ định tuyến biên đầu ra *BR2* hay không. Nếu đủ tài nguyên, *ISP1-BB* sẽ gửi bản tin *PATH* bằng tuyến đường *LSP* đến *BR2*.

Nếu không đủ tài nguyên, yêu cầu sẽ bị từ chối. Một bản tin lỗi sẽ được gửi trả lại *CN1-BB*. Host *S* sẽ được báo về việc từ chối này.

*Bước 5. BR2 gửi bản tin PATH đến CN2-BB.*

*Bước 6. CN2-BB sẽ quyết định xem mạng CN2 có đủ tài nguyên để cung cấp cho lưu lượng này hay không. Nếu không đủ tài nguyên, yêu cầu sẽ bị từ chối và một bản tin lỗi sẽ được gửi trở lại BR2. Máy S sẽ được thông báo lại. Nếu đủ tài nguyên, yêu cầu sẽ được chấp nhận. CN2-BB sẽ sử dụng giao thức LDAP hay RSVP để thiết lập các quy ước phân loại và xử lý gói tại ER2, sau đó gửi bản tin RSVP RESV đến BR2.*

*Bước 7. BR2 thiết lập các yêu cầu nắn gói đối với lưu lượng. Sau đó nó gửi bản tin RESV thông qua tuyến LSP đến BR1.*

*Bước 8. BR1 thiết lập quy tắc phân loại và xử lý gói cho lưu lượng. Sau đó nó gửi bản tin RESV đến CN1-BB.*

*Bước 9. Khi CN1-BB nhận được bản tin RESV, nó sẽ thiết lập quy tắc phân loại và nắn gói tại LRI và ERI. CN1-BB sau đó sẽ gửi bản tin RESV đến máy S.*

*Bước 10. Máy S bắt đầu truyền dữ liệu.*

#### **5.2.5. Kỹ thuật lưu lượng (Traffic Engineering) và các phương pháp định tuyến có điều kiện (Constrained Based Routing)**

Các mô hình cung cấp chất lượng dịch vụ như *IntServ*, *DiffServ* hay *DiffServ/MPLS* là các mô hình cho phép cung cấp các mức chất lượng khác nhau cho các loại lưu lượng khác nhau khi tải của mạng là lớn. Một hướng tiếp cận khác của các việc cung cấp chất lượng dịch vụ là làm thế nào để tránh được tắc nghẽn ngay từ đầu. Một trong những kỹ thuật có thể giải quyết được bài toán này là kỹ thuật lưu lượng (*Traffic Engineering*).

##### **5.2.5.1. Kỹ thuật lưu lượng (Traffic Engineering)**

Tắc nghẽn mạng xảy ra do nguyên nhân thiếu tài nguyên trong mạng hoặc do lưu lượng trong mạng được phân bố không đồng đều. Trong trường hợp thứ nhất, khi các bộ định tuyến và các đường truyền bị quá tải thì giải pháp duy nhất có thể có được lúc này là phải mở rộng tài nguyên của mạng. Trường hợp thứ hai sẽ xảy ra khi một vài phần của mạng bị quá tải,



trong khi các phần khác của mạng tải lại quá thấp. Hiện tượng tải bị phân bố không đồng đều này xảy ra do các giao thức định tuyến như *RIP*, *OSPF* và *IS-IS* đều có một đặc điểm giống nhau là lựa chọn đường đi ngắn nhất để chuyển gói. Khi đó, các bộ định tuyến và các đường truyền nằm trên tuyến đường ngắn nhất giữa hai nút sẽ trở nên tắc nghẽn, trong khi các bộ định tuyến và các đường truyền nằm trên các tuyến đường dài hơn lại bị rỗi. Đối với các mạng đơn giản, các nhà cung cấp dịch vụ hay quản lý mạng có thể thiết lập giá thành của các đường truyền sao cho lưu lượng trên các đường truyền được phân bố đồng đều. Nhưng với các mạng phức tạp hơn, quá trình thiết lập bằng tay này sẽ không thể thực hiện được.

Kỹ thuật lưu lượng chính là quá trình sắp xếp lưu lượng trong mạng sao cho tắc nghẽn xảy ra do phân bố lưu lượng không đồng đều sẽ không xảy ra. Các phương pháp định tuyến có điều kiện là một công cụ quan trọng để có thể thực hiện được kỹ thuật lưu lượng một cách tự động.

#### **5.2.5.2. Định tuyến có điều kiện CBR (Constraint Based Routing)**

Định tuyến có điều kiện là các phương pháp định tuyến thiết lập đường đi dựa vào một số điều kiện cho trước. Các điều kiện cho trước này có thể là các điều kiện *QoS* nào đó, nên định tuyến có điều kiện còn được gọi là định tuyến chất lượng dịch vụ, hay *QoS Routing*. Các phương pháp định tuyến này có mục đích như sau:

- Chọn các tuyến đường thích hợp với các yêu cầu của chất lượng dịch vụ *QoS*.
- Nâng cao hiệu suất sử dụng của mạng.

Khi tìm đường đi, các giao thức định tuyến có điều kiện sẽ xem xét không chỉ cấu hình của mạng, mà còn xem xét cả các đòi hỏi của luồng lưu lượng, tài nguyên còn lại trên đường truyền... Bằng cách này, các giao thức định tuyến có điều kiện có thể tìm được các tuyến đường dài nhưng có tải thấp hơn các tuyến đường tuy ngắn nhưng lại sắp quá tải. Qua đó lưu lượng trong mạng sẽ được phân bố đồng đều hơn.

Để có thể định tuyến theo phương pháp này, các bộ định tuyến sẽ phải trao đổi các thông số về trạng thái của đường truyền và sau đó tính toán đường đi theo các tham số này.

#### *a. Trao đổi thông tin về trạng thái của đường truyền*

Một bộ định tuyến sẽ cần các thông tin về cấu hình và tài nguyên còn lại của mạng để có thể chọn các tuyến đường thỏa mãn điều kiện chất lượng dịch vụ. Thông tin về các tài nguyên còn lại ở đây có thể là băng thông còn lại trên đường truyền, hay kích thước các hàng đợi tại các bộ định tuyến.

Một phương pháp trao đổi thông tin về đường truyền là mở rộng các giao thức trao đổi dựa trên trạng thái đường truyền như các giao thức *OSPF* và *IS-IS*. Bởi vì băng thông còn lại của các đường truyền thường xuyên thay đổi, do đó việc trao đổi thông tin của các đường truyền cần được diễn ra thường xuyên để theo kịp với sự thay đổi của mạng.

Để giảm thiểu được các thông tin điều khiển này, người ta có thể tiến hành trao đổi thông tin của các đường truyền chỉ khi nào có sự thay đổi, hoặc khi có một sự thay đổi vượt quá một mức ngưỡng nào đó, ví dụ 50% hay nhiều hơn 10 *Mbps*.

#### *b. Tìm đường đi*

Bảng định tuyến sẽ cho phép tìm đường đi theo phương pháp định tuyến có điều kiện *CBR* và thông thường độ phức tạp của các thuật toán này phụ thuộc vào tham số lựa chọn tuyến đường đó là gì.

Các tham số hay được sử dụng nhiều nhất là số nút, băng thông, độ tin cậy, trễ và độ biến thiên trễ. Các thuật toán định tuyến sẽ lựa chọn đường đi sao cho nó có thể tối ưu hóa được một hoặc các tham số này.

Các tham số này có thể được chia làm ba loại. Gọi  $d(i, j)$  là các tham số cho đường truyền  $(i, j)$ . Cho bất kỳ tuyến đường  $P=(i, j, k, \dots, l, m)$ , tham số  $d$  là:

– Tính chất cộng nếu  $d(P) = d(i, j) + d(j, k) + \dots + d(l, m)$ ;

– Tính chất nhân nếu  $d(P) = d(i, j) * d(j, k) * \dots * d(l, m)$ ;

– Tính concave nếu  $d(P) = \min(d(i, j), d(j, k), \dots, d(l, m))$ .

Theo định nghĩa này thì trễ, giá thành, số bước có tính chất cộng, độ tin cậy có tính chất nhân, còn băng thông có tính concave.

Sau nhiều nghiên cứu tìm hiểu về các phương pháp định tuyến có điều kiện *CBR*, việc tính toán một đường đi tối ưu với một số điều kiện cho trước là bài toán *NP* khó (*NP-complete*). Điều đó có nghĩa là, các thuật toán

sử dụng trễ, độ biến thiên trễ, số bước, độ tin cậy là các tham số để tối ưu sẽ là bài toán *NP* khó. Còn các thuật toán tính toán trên băng thông sẽ là bài toán đơn giản hơn.

Cụ thể như bài toán định tuyến tìm đường dựa vào băng thông và số bước đơn giản hơn rất nhiều. Các thuật toán *Bellman–Ford* và *Dijkstra* cũng có thể sử dụng được. Ví dụ như khi tìm đường đi ngắn nhất giữa hai nút có băng thông lớn hơn 1 *Mbps*, thì tất cả các đường truyền với băng thông nhỏ hơn 1 *Mbps* có thể bị lược bớt đi. Sau đó người ta có thể áp dụng thuật toán *Bellman–Ford* và *Dijkstra* cho mạng đã bị lược bớt đi các đường truyền này. Độ phức tạp của các thuật toán này khi đó được giảm đi rất nhiều.

Băng thông và số bước thường được sử dụng làm tham số điều kiện hơn là trễ và độ biến thiên trễ, bởi vì:

- Mặc dù các ứng dụng thường rất quan tâm đến thông số trễ và độ biến thiên trễ, nhưng chỉ có một vài ứng dụng là không hoạt động được khi các điều kiện trễ và độ biến thiên trễ không được thỏa mãn. Do đó, không cần thiết phải định tuyến cho các luồng đòi hỏi trễ và độ biến thiên trễ phải nhỏ hơn một mức ngưỡng nào đó. Bên cạnh đó, trễ và độ biến thiên trễ của một luồng có thể được xác định dựa vào băng thông và số chặng của tuyến đường. Trong trường hợp cần thiết, trễ và độ biến thiên trễ có thể được biểu diễn thông qua băng thông và số chặng.

- Rất nhiều các ứng dụng trong thời gian thực đòi hỏi phải cung cấp cho chúng một lượng băng thông xác định. Do đó băng thông là thông số hữu dụng nhất. Số chặng của tuyến đường cũng rất quan trọng vì càng nhiều chặng thì càng tốn nhiều tài nguyên.

Các phương pháp định tuyến *CBR* thường được sử dụng khi cần thiết hoặc một cách tự động cho mỗi lớp dịch vụ nào đó, ví dụ khi nhận được yêu cầu về chất lượng dịch vụ của một luồng hay một lớp dịch vụ, *CBR* sẽ được thực hiện. Trong cả hai trường hợp, một bộ định tuyến sẽ phải tính toán bảng định tuyến của nó thường xuyên hơn vì kể cả khi cấu hình mạng được giữ nguyên, thì bảng định tuyến cũng phải được cập nhật thường xuyên khi có sự thay đổi về băng thông. Ngoài ra, do phải cập nhật thường xuyên nên lưu lượng của các thông tin điều khiển của các giao thức định tuyến *CBR* cũng sẽ cao hơn của các phương pháp định tuyến động thông thường.

Để giảm thiểu được lượng thông tin điều khiển của các giao thức định tuyến theo phương pháp *CBR*, có thể có một số giải pháp sau:

- Giảm tần suất cập nhật thông tin;
- Lựa chọn băng thông và số chặng làm tham số;
- Sử dụng các phương pháp xử lý để loại bỏ các đường truyền không cần thiết trước khi tính toán.

*c. Ưu nhược điểm của định tuyến có điều kiện*

Ưu điểm của các phương pháp định tuyến này là:

- Đáp ứng được các yêu cầu về chất lượng dịch vụ;
- Nâng cao hiệu suất sử dụng của mạng.

Nhược điểm của các phương pháp này là:

- Tăng lượng thông tin điều khiển;
- Tăng kích thước bảng định tuyến;
- Các tuyến đường dài hơn sẽ tiêu tốn nhiều tài nguyên hơn;
- Mất cân bằng trong quá trình định tuyến.

**Cấu trúc bảng định tuyến và kích thước**

Cấu trúc của bảng định tuyến và kích thước của nó phụ thuộc trực tiếp vào bước định tuyến và các tham số định tuyến. Về mặt logic, một bảng định tuyến có thể được coi như một mảng hai chiều. Số hàng của mảng này được xác định bằng bước định tuyến và số cột được xác định bằng tham số định tuyến. Ví dụ như khi định tuyến theo đích với tham số là băng thông và số chặng thì kích thước bảng định tuyến sẽ là mảng có kích thước  $K \times H$  với  $K$  là số đích và  $H$  là số chặng lớn nhất có thể cho phép của một tuyến đường.

Do đó, kích thước bảng định tuyến của các giao thức định tuyến theo kiểu *CBR* có thể sẽ lớn hơn rất nhiều so với kích thước của các giao thức định tuyến thông thường. Điều này sẽ làm cho quá trình tìm kiếm trong bảng định tuyến này trở nên chậm hơn.

Để có thể làm giảm kích thước của bảng định tuyến, có nhiều giải pháp khác nhau như:

- Tăng độ mịn hay bước định tuyến;
- Lượng tử hóa số chặng;

– Chỉ tính toán tuyến đường cho các luồng đòi hỏi chất lượng dịch vụ khi có yêu cầu.

### **Mối quan hệ giữa tiêu thụ tài nguyên và cân bằng tải**

Một giao thức định tuyến theo phương pháp *CBR* có thể lựa chọn các phương pháp sau để tìm đường tới đích:

– Chọn đường theo kiểu rộng nhất – ngắn nhất, tức là tuyến đường có số chặng ít nhất và nếu có nhiều tuyến đường như vậy thì chọn tuyến có băng thông còn lại lớn nhất.

– Chọn đường ngắn nhất – rộng nhất, tức là chọn tuyến đường có băng thông còn lại lớn nhất và nếu có nhiều tuyến đường như vậy thì chọn đường có số chặng ít nhất.

– Chọn đường ngắn nhất theo khoảng cách. Khoảng cách của một đường  $P$  có  $k$  chặng được định nghĩa bởi:

$$dist(P) = \sum_{i=1}^k \frac{1}{r_i}$$

trong đó:  $r_i$  là băng thông của tuyến  $i$ .

Thông thường, lựa chọn tuyến đường không phải là ngắn nhất sẽ làm tiêu tốn nhiều tài nguyên hơn. Khi tải của mạng là lớn, việc tiêu tốn nhiều tài nguyên hơn sẽ dẫn đến tắc nghẽn mạng nhanh hơn. Giữa yếu tố tiêu thụ tài nguyên và cân bằng tải có một mối liên hệ với nhau. Trong ba phương pháp chọn đường trên, phương pháp thứ nhất lựa chọn đường đi ngắn nhất, do đó nó sẽ ưu tiên việc tiêu tốn ít tài nguyên hơn cân bằng tải. Phương pháp thứ hai lựa chọn các tuyến đường có băng thông lớn hơn, do đó ưu tiên yếu tố cân bằng tải hơn. Phương pháp thứ ba là phương pháp trung gian. Nó lựa chọn tuyến đường ngắn nhất khi tải của mạng là cao và lựa chọn tuyến đường có băng thông lớn nhất khi tải của mạng là thấp.

### **Độ tin cậy**

Các phương pháp định tuyến theo kiểu *CBR* phải tính toán và tìm tuyến đường nhiều hơn các phương pháp định tuyến thông thường, do đó nó thường không ổn định bằng các phương pháp định tuyến thông thường.

#### **5.2.5.3. Vai trò của định tuyến có điều kiện trong các mô hình cung cấp QoS**

Trước hết, các giao thức định tuyến theo kiểu *CBR* sẽ lựa chọn tuyến đường tối ưu cho các luồng để đạt được chất lượng dịch vụ. Tuy nhiên, điều này không phải hoàn toàn chắc chắn, do đó các giao thức này không thể thay thế được mô hình *DiffServ*, tuy nhiên nó có thể bổ trợ thêm cho *DiffServ*.

Khi xem xét ảnh hưởng của các giao thức định tuyến theo kiểu *CBR* với *RSVP*, người ta thấy rằng hai giao thức này độc lập với nhau. Với một bộ định tuyến hoạt động theo kiểu các giao thức định tuyến thông thường, khi nhận được bản tin *RSVP PATH*, nó sẽ chuyển tiếp bản tin này đến nút tiếp theo tùy theo phương thức mà giao thức định tuyến thông thường đó làm việc. Chất lượng dịch vụ và tải của mạng không được quan tâm đến khi chuyển tiếp bản tin này. Với một giao thức định tuyến làm việc theo kiểu *CBR*, việc chuyển tiếp bản tin này đến nút tiếp theo sẽ quan tâm đến tải của mạng và chất lượng dịch vụ. Nói cách khác, các giao thức hoạt động theo kiểu *CBR* sẽ xác định đường đi cho các bản tin của *RSVP* nhưng không đặt trước tài nguyên như *RSVP*.

Khi so sánh các giao thức định tuyến theo kiểu *CBR* với *MPLS*, có một sự khác biệt đơn giản là *MPLS* chỉ là một giao thức chuyển tiếp thông thường chứ không phải là một giao thức định tuyến. *CBR* cho phép xác định đường đi giữa hai nút dựa vào các thông tin về tài nguyên và cấu hình. Sử dụng các thông tin về tuyến đường này, *MPLS* sẽ sử dụng các giao thức của nó để xác định các tuyến đường *LSP*. *MPLS* không quan tâm đến việc các tuyến đường này được thiết lập bởi các giao thức định tuyến *CBR* hay bởi các giao thức định tuyến thông thường. Tuy nhiên, khi *MPLS* và *CBR* được sử dụng đồng thời, các yêu cầu về *QoS* sẽ được thiết lập ở cả lớp 3 và lớp 2, do đó sẽ hiệu quả hơn nếu chỉ sử dụng *MPLS* hoặc chỉ sử dụng *CBR*.

## BÀI TẬP CHƯƠNG 5

1. Phân biệt hai khái niệm luồng trong *IntServ* và lớp trong *DiffServ*.
2. Cài đặt công cụ mô phỏng *NS2*. Sử dụng *NS2* để thiết lập ứng dụng thuộc dịch vụ có đảm bảo và dịch vụ ưu tiên của *DiffServ*.
3. Cài đặt công cụ mô phỏng *NS2*. Sử dụng *NS2* để thiết lập các dịch vụ tải có điều khiển và dịch vụ *GS*.
4. Tại sao trong phương pháp định tuyến có điều kiện, các thông số trễ, giá thành, số bước có tính chất cộng, độ tin cậy có tính chất nhân, còn băng thông có tính concave?
5. Hãy phân tích vai trò của giao thức *RSVP* trong hoạt động của *IntServ*, *DiffServ*, *MPLS* và kỹ thuật lưu lượng?

## TÀI LIỆU THAM KHẢO

1. R. Comerford. State of the Internet: *Roundtable 4.0*. IEEE Spectrum, Oct. 1998.
2. D. Ferrari and L. Delgrossi. *Charging For QoS*. IEEE/IFIP IWQOS '98 keynote paper, Napa, California, May 1998.
3. P. Ferguson and G. Huston. *Quality of Service*. John Wiley & Sons, 1998.
4. Braden, R., Clark, D. and Shenker, S. *Integrated Services in the Internet Architecture: an Overview*. Internet RFC 1633, Jun. 1994.
5. R. Jain. Myths about Congestion Management in High Speed Networks. *Internetworking: Research and Experience*, Volume 3, pp. 101–113, 1998.
6. S. Shenker, C. Partridge and R. Guerin. *Specification of Guaranteed Quality of Service*. RFC 2212, Sept. 1997.
7. J. Wroclawski. *Specification of the Controlled-Load Network Element Service*. RFC 2211, Sept. 1997.
8. R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin. *Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification*. RFC 2205, Sept. 1997.
9. D. Ferrari, D. Verma. *A Scheme for Real-Time Channel Establishment in Wide-Area Networks*. IEEE JSAC, vol. 8, no. 3, April, pp. 368–379, 1990.
10. A. Banerjea and B. Mah. *The Real-Time Channel Administration Protocol*. Proceedings of the 2nd International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'91), Springer-Verlag, Heidelberg, Germany, pp. 160–170, 1991.



11. D. Clark. *The Design Philosophy of the DARPA Internet Protocol*. ACM SIGCOMM '88, Aug. 1988.
12. J. Postel. *Service Mappings*. RFC 795, Sept. 1981.
13. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. *An Architecture for Differentiated Services*. RFC 2475, Dec, 1998.
14. K. Nichols, S. Blake, F. Baker and D. Black. *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*. RFC 2474, Dec, 1998.
15. T. Li and Y. Rekhter. *Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)*. RFC 2430, Oct, 1998.
16. B. Braden et al. *Recommendation on Queue Management and Congestion Avoidance in the Internet*. RFC 2309, Apr, 1998.
17. W. Yeong, T. Howes, and S. Kille. *Lightweight Directory Access Protocol*. RFC 1777, Mar, 1995.
18. M. Waldvoe, G. Varghese, J. Turner and B. Plattner. *Scalable High Speed IP Routing Lookups*. ACM SIGCOMM '97, Cannes, France, Sept. <http://www.acm.org/sigcomm/sigcomm97>, 1997.
19. S. Nilsson and G. Karlsson. *Fast Address Lookup for Internet Routers*, ACM SIGCOMM '97, Cannes, France, Sept.. <http://www.acm.org/sigcomm/sigcomm97>, 1997.
20. E. Crawley, R. Nair, B. Jajagopalan and H. Sandick. *A Framework for QoS-based Routing in the Inter-net*. RFC 2386, Aug, 1998.
21. G. Apostolopoulos, R. Guerin, S. Kamat and S. Tripathi. *Quality of Service Based Routing: A Performance Perspective*. ACM SIGCOMM '98, pp. 17–28, Vancouver, Canada, Aug, 1998.
22. Q. Ma. *QoS Routing in the Integrated Services networks*. Ph.D. thesis, CMU-CS-98-138, Jan, 1998.
23. Z. Wang. *Routing and Congestion Control in Datagram Networks*. Ph.D. thesis, Dept. of Computer Sci., University College London, Jan, 1992.
24. Z. Wang and J. Crowcroft. *Quality of Service Routing for Supporting Multimedia Applications*. IEEE JSAC, Sept, 1996.

25. A. Orda. *Routing with End-to-End QoS Guarantees in Broadband Networks*. Technical Report, Technion, I.I.T., Israel.
26. Y. Goto, M. Ohta and K. Araki. *Path QoS Collection for Stable Hop-by-hop QoS Routing*. Proceedings of INET '97, Kuala Lumpur, Malaysia, Jun, 1997.
27. F. Kelly. *Notes on Effective Bandwidths. in Stochastic Networks: Theory and Applications*. pp. 141–168, Oxford University Press, 1996.
28. F. Kelly. *Modelling Communication Networks, Present and Future*. Philosophical Transactions of the Royal Society A354, pp. 437–463, 1996.
29. G. R. Ash, J. S. Chen, A. E. Frey and B. D. Huang. *RealTime Network Routing in a Dynamic Class-of-Service Network*. Proceedings of ITC 13, Copenhagen, Jun, 1991.
30. ATM Forum PNNI subworking group. *Private Network-Network Interface Spec. v1.0 (PNNI 1.0)*. af-pnni-0055.00, Mar, 1996.

## Chương 6

# MÔ PHÒNG

Trong các chương trước, chúng ta đã tìm hiểu các phương pháp đánh giá hiệu năng theo phương pháp phân tích. Tuy nhiên trong thực tế vẫn tồn tại nhiều mô hình không thể đánh giá được bằng các phương pháp phân tích này. Trong trường hợp này, chúng ta có thể sử dụng phương pháp mô phỏng.

### 6.1. CÁC KỸ THUẬT MÔ PHÒNG

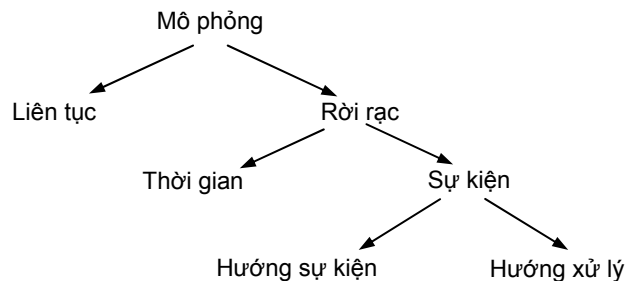
Để phân loại các kỹ thuật mô phỏng, người ta căn cứ vào hai tiêu chí: khoảng không gian trạng thái và sự phát triển theo thời gian của nó. Cụ thể có những kỹ thuật mô phỏng sau:

**Kỹ thuật mô phỏng theo sự kiện liên tục** (*continuous – event simulations*): là kỹ thuật được sử dụng khi các trạng thái của hệ thống biến thiên một cách liên tục theo thời gian. Ví dụ điển hình của trường hợp này là các quá trình vật lý mà có thể được biểu diễn bằng phương trình vi phân với các điều kiện bờ.

**Kỹ thuật mô phỏng theo sự kiện rời rạc** (*discrete–event simulations*): là một phương pháp khác với kỹ thuật mô phỏng theo sự kiện liên tục. Nguyên nhân dẫn đến sự thay đổi trạng thái của hệ thống được gọi là sự kiện (*event*). Khi các sự kiện của hệ thống xảy ra một cách lần lượt, lúc này chúng ta có khái niệm mô phỏng theo sự kiện rời rạc, hay còn gọi là *discrete–event simulations*. Một cách hiển nhiên, kỹ thuật mô phỏng một hệ thống theo sự kiện rời rạc được thực hiện dễ dàng hơn rất nhiều so với mô phỏng một hệ thống theo sự kiện liên tục.

Chính vì vậy, trong chương này, chúng ta sẽ chỉ chú ý đến kỹ thuật mô phỏng theo sự kiện rời rạc.

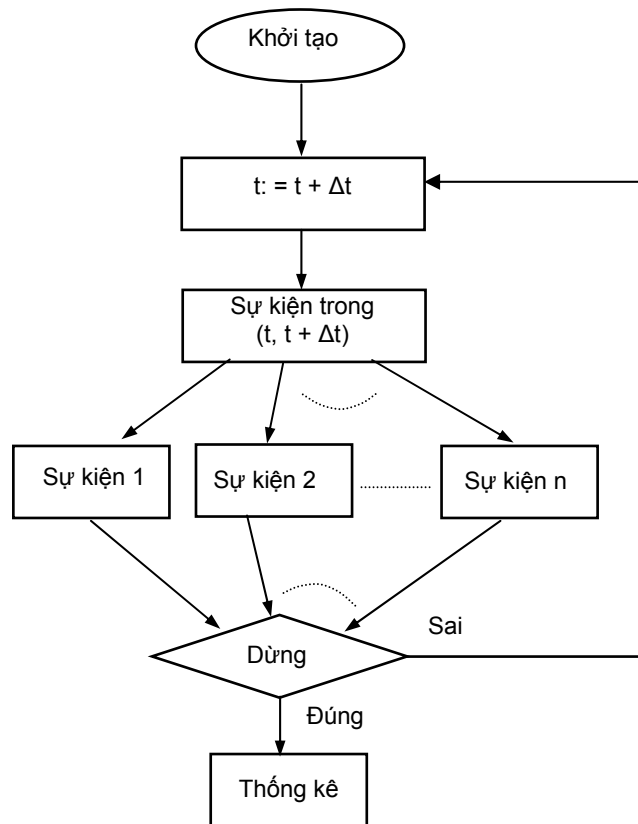
Kỹ thuật mô phỏng theo sự kiện rời rạc, đến lượt nó, được phân thành theo thời gian (*time-based*) và theo sự kiện (*event-based*).



**Hình 6.1. Phân loại kỹ thuật mô phỏng**

Kỹ thuật mô phỏng theo **thời gian** (còn được gọi là mô phỏng đồng bộ – *synchronous simulation*) là kỹ thuật mô phỏng mà vòng lặp điều khiển của quá trình mô phỏng này sẽ điều khiển thời gian biến thiên theo các bước cố định. Khi bắt đầu vòng lặp điều khiển, thời gian  $t$  được tăng thêm một bước  $\Delta t$  thành  $t + \Delta t$ ... Sau đó hệ thống sẽ tiếp tục kiểm tra xem trong khoảng thời gian từ  $[t, t + \Delta t]$  có sự kiện nào xảy ra không. Nếu có, các sự kiện này sẽ được thực hiện và trạng thái của hệ thống sẽ thay đổi tương ứng với các sự kiện này. Giả thiết rằng thứ tự xuất hiện của các sự kiện trong khoảng thời gian này là không quan trọng và các sự kiện này cũng không phụ thuộc vào nhau. Số sự kiện xuất hiện trong khoảng thời gian này cũng có thể thay đổi theo thời gian. Khi  $t$  vượt qua một mức ngưỡng nào đó, quá trình mô phỏng sẽ dừng lại. Hình vẽ 6.2 mô tả quá trình diễn biến của một hành động theo phương pháp mô phỏng theo thời gian.

Phương pháp mô phỏng dựa theo thời gian này có thể được triển khai một cách dễ dàng, tuy nhiên cũng có một số nhược điểm sau: với giả thiết các sự kiện xuất hiện trong khoảng thời gian  $[t, t + \Delta t]$  là không quan trọng và các sự kiện này không phụ thuộc với nhau, thì bước thời gian  $\Delta t$  phải đủ nhỏ để có thể giảm thiểu được xác suất xuất hiện các sự kiện đồng thời trong khoảng thời gian này. Chính vì lý do này, người ta thường chọn các bước thời gian  $\Delta t$  đủ nhỏ và nếu nhỏ quá thì sẽ làm cho không có sự kiện nào xảy ra. Vì vậy mà kỹ thuật mô phỏng theo thời gian thường ít khi được thực hiện.



**Hình 6.2. Các bước thực hiện của mô phỏng theo thời gian hệ thống M/M/1**

Trong ví dụ sau, chúng ta sẽ xem xét chương trình mô phỏng hệ thống  $M/M/1$  theo phương pháp mô phỏng theo thời gian. Giả sử hàng đợi có tốc độ đến là  $\lambda$  và tốc độ phục vụ là  $\mu$ . Trong trường hợp này, chúng ta sử dụng hai biến trạng thái:  $N_S \in \{0,1\}$  là số yêu cầu đang được phục vụ và  $N_q \in \mathbb{Q}$  là số yêu cầu đang nằm trong hàng đợi. Ngoài ra, giữa hai tham số này có mối quan hệ sau:  $N_q > 0 \rightarrow N_S = 1$ . Mục đích của chương trình mô phỏng hệ thống  $M/M/1$  này là phải tạo ra được một danh sách các khoảng thời gian và các biến trạng thái tại các khoảng thời gian đó. Giả sử  $\Delta$  là một biến đủ nhỏ, tiếp theo chúng ta phải sử dụng hàm  $draw(p)$  có giá trị là *đúng* với xác suất  $p$  và *sai* với xác suất  $1 - p$ .

Chương trình mô phỏng được minh họa ở hình vẽ 6.3. Sau khi được khởi tạo (dòng từ 1 – 3), vòng lặp của chương trình bắt đầu. Đầu tiên thời gian được cập nhật (dòng 6). Nếu trong khoảng thời gian  $[t, t + \Delta t]$  có một sự kiện xảy ra với xác suất  $\lambda \Delta t$  của tiến trình Poisson với tốc độ  $\lambda$ , chúng ta phải tăng số yêu cầu chờ trong hàng đợi lên (dòng 7). Sau đó chúng ta sẽ kiểm tra xem có yêu cầu nào đang được phục vụ hay không. Nếu không, yêu cầu sẽ được đưa vào trạm phục vụ (dòng 13 – 14). Nếu như đã có một yêu cầu ở trạm phục vụ, công việc tiếp theo là phải kiểm tra xem quá trình phục vụ yêu cầu đó đã kết thúc ở khoảng thời gian trước đó hay chưa. Nếu đúng như vậy, bộ đếm  $N_s$  được thiết lập bằng 0 (dòng 12), nếu ngược lại tức là đang có một yêu cầu chờ đợi để được phục vụ (dòng 10). Trong trường hợp này một yêu cầu sẽ rời khỏi hàng đợi và trạm phục vụ lại bận.

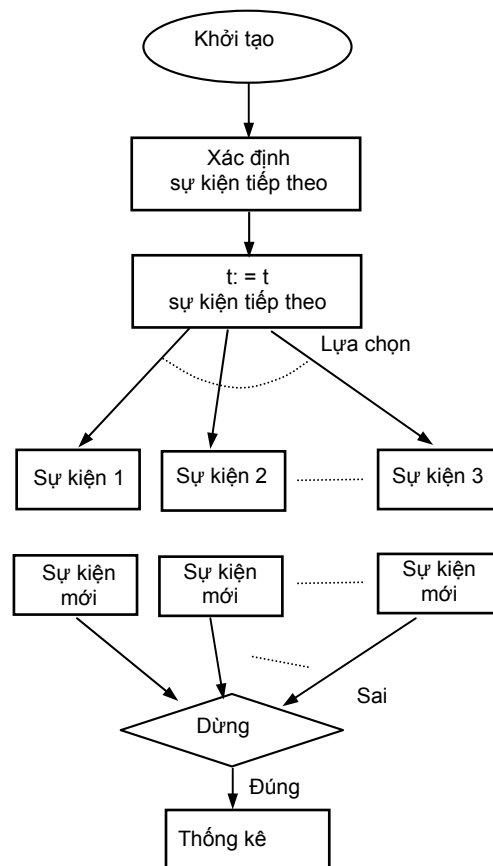
```

1. input ( $\lambda, \mu, t_{\max}$ )
2.  $t := 0$ 
3.  $N_s := 0; N_q := 0$ 
4. while  $t < t_{\max}$ 
5. do
6.      $t := t + \Delta t$ 
7.     if draw ( $\lambda \cdot \Delta t$ ) then  $N_q := N_q + 1$ 
8.     if  $N_s = 1$ 
9.     then if draw ( $\mu \cdot \Delta t$ )
10.         then if  $N_q > 0$ 
11.             then  $N_q := N_q - 1$ 
12.             else  $N_s := 0$ 
13.         if  $N_s = 0$  and  $N_q > 0$ 
14.         then  $N_s := 1; N_q := N_q - 1$ 
15.     writeln ( $t, N_q, N_s$ )
16. od

```

**Hình 6.3. Giả ngôn ngữ của mô phỏng M/M/1 theo thời gian**

**Kỹ thuật mô phỏng theo sự kiện (*event-based simulation*):** Trong phương pháp mô phỏng theo thời gian, bước thời gian  $\Delta t$  là cố định, nhưng số sự kiện xảy ra trong bước thời gian này có thể thay đổi. Còn trong phương pháp mô phỏng theo sự kiện (hay còn được gọi là mô phỏng không đồng bộ *asynchronous simulation*), người ta sẽ thực hiện theo cách ngược lại. Lúc này các bước thời gian sẽ không cố định mà có độ dài thay đổi sao cho luôn luôn chỉ có một sự kiện trong mỗi bước thời gian. Vì thế mô phỏng sẽ được điều khiển bằng sự xuất hiện của các sự kiện. Phương pháp này rất hiệu quả vì các bước thời gian đủ lớn để có thể xử lý mô phỏng và cũng đủ nhỏ để có thể có một hoặc nhiều hơn một sự kiện xảy ra trong nó.



**Hình 6.4. Các bước thực hiện mô phỏng hệ thống M/M/1 theo sự kiện**

Thông thường khi một sự kiện xuất hiện, trạng thái của hệ thống sẽ bị thay đổi và sự kiện đó cũng có thể kéo theo sự xuất hiện của một hoặc nhiều sự kiện khác. Các sự kiện xảy ra trong tương lai thông thường được tập hợp lại trong một danh sách theo thứ tự và sẽ được thực hiện lần lượt theo thứ tự này, đồng thời các sự kiện sẽ xảy ra trong tương lai liên tục được cập nhật trong danh sách này.

Khi đánh giá hiệu năng của các hệ thống truyền thông và máy tính, người ta thường chủ yếu sử dụng phương pháp mô phỏng theo sự kiện. Nhược điểm lớn nhất của phương pháp mô phỏng theo sự kiện là ở chỗ không phải lúc nào chúng ta cũng có thể tính toán được thời gian mà một sự kiện trong tương lai sẽ xảy ra. Trong các trường hợp này, người ta có thể cân nhắc sử dụng phương pháp mô phỏng theo thời gian để thay thế cho nó.

Trong ví dụ sau, chúng ta sẽ xem xét chương trình mô phỏng theo sự kiện của hệ thống  $M/M/1$ . Ở đây chúng ta cũng dùng hai biến:  $N_S \in \{0,1\}$  biểu diễn số yêu cầu đang được phục vụ và  $N_q \in Q$  là số yêu cầu đang nằm trong hàng đợi. Sau đó chúng ta sẽ dùng hai biến của sự kiện tiếp theo:  $narr$  là thời gian xảy ra sự kiện tiếp theo và  $ndep$  biểu diễn thời gian kết thúc sự kiện này. Bởi vì nhiều nhất chỉ có hai sự kiện tiếp theo, nên chúng ta chỉ cần sử dụng hai biến. Mục đích của chương trình này là tạo ra một danh sách các sự kiện và biến trạng thái tại các thời điểm đó. Để thực hiện được điều này cần phải sử dụng hàm  $negexp(\lambda)$  là hàm phát ra một biến ngẫu nhiên với phân bố mũ âm và tốc độ  $\lambda$ .

Hình vẽ 6.5 minh họa chương trình mô phỏng hệ thống  $M/M/1$  theo phương pháp trên. Sau khi khởi tạo (dòng 1 – 3), vòng lặp chương trình bắt đầu. Sử dụng biến  $N_S$  sẽ quyết định được sự kiện tiếp theo là gì (dòng 6). Nếu không có yêu cầu nào được xử lý, thì sự kiện tiếp theo sẽ là một yêu cầu đến: thời gian đến sự kiện tiếp theo được tạo ra, tương ứng theo đó thời gian mô phỏng sẽ được cập nhật và biến trạng thái sẽ được tăng lên 1 (dòng 15 – 17). Nếu như có một yêu cầu được xử lý, thì có thể có hai sự kiện tiếp theo. Thời gian cho hai sự kiện này được tính toán (dòng 7 – 8) và sự kiện nào xảy ra trước sẽ được thực hiện trước (dòng 9). Nếu yêu cầu được xử lý xong trước, thời gian mô phỏng được điều chỉnh lại và nếu đang có yêu cầu chờ được phục vụ thì yêu cầu này sẽ được đưa vào trạm phục vụ. Nếu ngược lại,



hàng đợi và bộ đệm sẽ trống (dòng 10 – 13). Nếu sự kiện xảy ra trước, thời gian sẽ được điều chỉnh tương ứng và hàng đợi sẽ được tăng lên 1 (dòng 13 – 14).

```

1. input ( $\lambda, \mu, t_{\max}$ )
2.  $t := 0$ 
3.  $N_s := 0; N_q := 0$ 
4. while  $t < t_{\max}$ 
5. do
6.     if  $N_s = 1$ 
7.     then  $\text{narr} := \text{negexp}(\lambda)$ 
8.          $\text{narr} := \text{negexp}(\mu)$ 
9.         if  $\text{ndep} < \text{narr}$ 
10.            then if  $t := t + \text{ndep}$ 
11.                if  $N_q > 0$ 
12.                then  $N_q := N_q - 1$ 
13.            else  $t := t + \text{narr}$ 
14.                 $N_q := N_q + 1$ 
15.        else  $\text{narr} := \text{negexp}(\lambda)$ 
16.             $t := t + \text{narr}$ 
17.             $N_s := 1$ 
18.        writeln ( $t, N_q, N_s$ )
19. od

```

**Hình 6.5. Giả ngôn ngữ của mô phỏng hệ thống M/M/1 theo sự kiện**

### 6.1.1. Thực hiện mô phỏng theo hướng sự kiện

Kỹ thuật mô phỏng theo sự kiện, đến lượt nó, lại được chia làm hai loại: mô phỏng theo hướng sự kiện (*event-oriented*) và mô phỏng theo hướng xử lý (*process-oriented*).

Trong phương pháp thứ nhất, người ta phân chia các sự kiện thành nhiều loại khác nhau và với mỗi một loại sự kiện  $i$  người ta định nghĩa một quá trình xử lý  $P_i$  tương ứng. Trong bộ mô phỏng này, một danh sách các sự kiện sẽ được khởi tạo. Sau đó vòng lặp điều khiển chính sẽ được bắt đầu,

bao gồm các bước sau. Sự kiện đầu tiên xảy ra sẽ là sự kiện đứng đầu trong danh sách sự kiện và thời gian của bộ mô phỏng sẽ được tăng thêm một bước tương ứng với thời gian của sự kiện đầu tiên này. Kèm theo sự kiện đầu tiên này là quá trình  $P_1$  xử lý tương ứng với nó cũng sẽ được gọi đến. Quá trình  $P_1$  này xảy ra sẽ làm cho một số sự kiện mới được đưa thêm vào danh sách sự kiện và trạng thái của hệ thống sẽ bị thay đổi. Sau khi quá trình xử lý  $P_1$  kết thúc, người ta sẽ thu thập một số số liệu thống kê để tính toán và vòng lặp điều khiển chính sẽ lại được tiếp tục.

Khi thực hiện phương pháp mô phỏng hướng sự kiện, việc quản lý các sự kiện là công việc bắt buộc. Còn trong các phương pháp mô phỏng theo hướng xử lý, một quá trình xử lý sẽ được gắn với một kiểu sự kiện  $i$  nào đó. Các quá trình này sẽ trao đổi thông tin để thông báo sự thay đổi trạng thái thông qua các biến hoặc các bản tin. Hệ thống được mô phỏng lúc này có thể được coi như quá trình trao đổi của các sự kiện và các quá trình xử lý. Việc quản lý các sự kiện của hệ thống lúc này là không bắt buộc.

Cả hai phương pháp mô phỏng này đều được sử dụng rất rộng rãi. Trong phương pháp mô phỏng hướng sự kiện, người ta có thể sử dụng các ngôn ngữ lập trình Pascal hay C. Trong phương pháp mô phỏng theo hướng xử lý, người ta có thể sử dụng ngôn ngữ C++.

Hiện nay, thay vì phải tự lập trình để mô phỏng, có rất nhiều các phần mềm khác nhau sử dụng một trong hai phương pháp lập trình nói trên. Một số các phần mềm thương mại, sử dụng các giao diện đồ họa, có thể giúp mô phỏng các hệ thống máy tính và truyền thông rất hiệu quả.

### **6.1.2. Bộ phát số ngẫu nhiên (Random Number Generation RNG)**

Để có thể mô phỏng hiệu năng của các hệ thống truyền thông và máy tính, một yếu tố rất quan trọng là phải tạo ra được các số ngẫu nhiên tương ứng với một số luật phân bố xác suất nào đó, còn gọi là các bộ phát số ngẫu nhiên *RNG*.

Các bộ phát số ngẫu nhiên tuân theo luật phân bố xác suất cho trước này thường rất khó có thể tạo ra được và người ta thường phải thay thế bằng các bộ phát số gần ngẫu nhiên (*pseudo-random numbers*). Để có thể tạo ra một số gần ngẫu nhiên tuân theo một luật phân bố cho trước, thường phải

thực hiện theo ba bước. Trước hết phải tạo ra một dãy gồm một tập hữu hạn các số gần ngẫu nhiên gọi là tập  $N$ , thông thường chứa các phần tử  $(0, \dots, m - 1)$  với  $m \in N$ . Từ tập hợp các số gần ngẫu nhiên này, người ta tính toán luật phân bố của các số ngẫu nhiên. Để có thể kiểm tra xem các số gần ngẫu nhiên này có thể được coi như những số ngẫu nhiên thực sự hay không, người ta phải thực hiện một số phép thử thống kê.

#### 6.1.2.1. Tạo các số ngẫu nhiên

Việc tạo ra một dãy các số gần ngẫu nhiên là một công việc không hề đơn giản, mặc dù có rất nhiều phương pháp như vậy. Ở đây chúng ta chỉ quan tâm đến hai phương pháp là phương pháp tuyến tính (*linear*) và phương pháp cộng đồng dư (*additive congruential*), bởi vì đây là hai phương pháp đơn giản và dễ thực hiện nhất. Một bộ phát số ngẫu nhiên được coi là có chất lượng tốt khi:

- Dãy các số gần ngẫu nhiên được tính toán đơn giản và không tốn kém.

- Dãy các số phát ra giống như dãy số ngẫu nhiên, nghĩa là các số phát ra không phụ thuộc vào nhau và tuân theo luật phân bố cho trước.

- Chu kỳ của nó (khoảng thời gian mà dãy số bị lặp lại) là rất dài.

Sau đây chúng ta sẽ tìm hiểu hai phương pháp phát số ngẫu nhiên này và phân tích các đặc điểm của nó dựa vào ba yêu cầu trên.

Ý tưởng cơ bản của bộ phát số ngẫu nhiên theo kiểu tuyến tính đồng dư là rất đơn giản. Bắt đầu với giá trị  $a$  và  $z_0$ , số  $z_{i+1}$  được tính toán từ số  $z_i$  như sau:

$$z_{i+1} = (az_i + c) \text{ modulo } m \quad (6.1)$$

Nếu lựa chọn các thông số  $a$ ,  $c$  và  $m$  một cách chính xác, thuật toán này sẽ tạo ra  $m$  giá trị khác nhau, sau đó nó lại bắt đầu chu kỳ mới. Số  $m$  được gọi là chiều dài chu kỳ (*cycle length*). Bởi vì giá trị tiếp theo của dãy chỉ phụ thuộc vào giá trị hiện tại, nên chu kỳ sẽ bắt đầu khi có một giá trị xuất hiện lại trong dãy. Bộ tạo số ngẫu nhiên theo kiểu tuyến tính này sẽ có chiều dài chu kỳ là  $m$  nếu như ba điều kiện sau được thỏa mãn:

- Ước số chung lớn nhất của  $m$  và  $c$  là 1.
- Tất cả các ước số nguyên tố của  $m$  đều chia hết cho  $a - 1$ .
- Nếu 4 chia hết cho  $m$ , thì 4 cũng chia hết cho  $a - 1$ .

Tuy nhiên cũng cần phải nói thêm rằng, các điều kiện trên không thể được coi là các điều kiện để khẳng định các số phát ra là ngẫu nhiên thực sự.

Ví dụ xét trường hợp  $m = 16$ ,  $c = 7$  và  $a = 5$ , chúng ta có thể dễ dàng kiểm tra điều kiện trên. Bắt đầu với  $z_0 = 0$ , chúng ta nhận được  $z_1 = (5 * 0 + 7) \text{ modulo } 16 = 7$ . Tiếp tục như vậy, chúng ta nhận được 0,7,10,9,4,11,14,...

Nhược điểm quan trọng nhất ở đây là chu kỳ của dãy số ngẫu nhiên này khá ngắn, nghĩa là việc lặp lại khá thường xuyên và như vậy thì tính ngẫu nhiên sẽ không được đảm bảo. Vấn đề này có thể được giải quyết bằng cách sử dụng phương pháp cộng đồng dư (*additive congruential*). Với phương pháp này, giá trị thứ  $i$  là  $z_i$  được tạo ra từ  $k$  giá trị trước  $z_{i-1}, \dots, z_{i-k}$  theo phương pháp sau:

$$z_i = \left( \sum_{j=1}^k a_j z_{i-j} \right) \text{ modulo } m \quad (6.2)$$

Các giá trị ban đầu từ  $z_0$  đến  $z_{k-1}$  được tạo ra bởi phương pháp tuyến tính ở trên. Với việc lựa chọn giá trị  $a_j$ , có thể nhận được chu kỳ là  $m^k - 1$ .

Ví dụ như khi  $k = 7$  và thiết lập các hệ số  $a_1 = a_7 = 1$  và  $a_2 = \dots = a_6 = 0$ , chúng ta có thể mở rộng ra ví dụ đã xét ở trên. Để bắt đầu chúng ta sẽ sử dụng các số ngẫu nhiên đã được tạo ra ở trên là 0,7,10, 9, 4,11,14. Giá trị tiếp theo sẽ là  $(14 + 0) \text{ modulo } 16 = 14$ . Bằng cách này chúng ta nhận được dãy 0,7,10,9, 4,11,14,14,5,15,8,12,7,5... Một điều hiển nhiên rằng, khi một số lặp lại thì điều đó không có nghĩa là chu kỳ mới bắt đầu. Trong trường hợp này, độ dài chu kỳ được giới hạn bởi  $16^7 - 1 = 1268435455$ .

Ngoài ra, nên sử dụng các bộ phát số ngẫu nhiên khác nhau cho mỗi dãy số ngẫu nhiên được sử dụng khi mô phỏng. Việc lựa chọn các tham số ban đầu (hay còn gọi là các *mầm*) là rất quan trọng. Ngay cả những bộ RNG tốt cũng có thể hoạt động không chính xác nếu các mầm ban đầu không thích hợp.

#### **6.1.2.2. Bộ phát các số ngẫu nhiên theo phân bố không đều (Non-uniform Distribution)**

Để nhận được các số ngẫu nhiên tuân theo các luật phân bố khác nhau, có rất nhiều cách sử dụng các số ngẫu nhiên theo phân bố đồng nhất. Sau đây chúng ta sẽ giới thiệu một vài kỹ thuật sau:

### Phương pháp ngược

Xem xét hàm phân bố  $F_Y(y)$  của một biến ngẫu nhiên  $Y$ . Gọi  $Z$  là một hàm của biến ngẫu nhiên  $Y$  được định nghĩa như sau:

$$Z = F_Y(y)$$

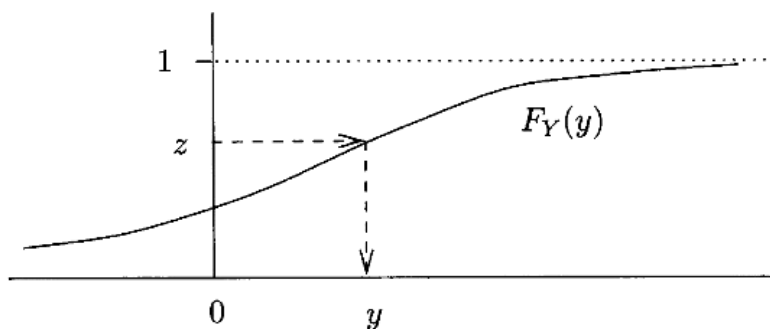
Hàm phân bố của  $Z$ , tức là hàm  $F_Z(z)$  được xác định như sau:

$$F_Z(z) = \Pr\{Z \leq z\} = \Pr\{F_Y(Y) \leq z\} \quad (6.3)$$

Tiếp theo, giải thiết rằng  $F_Y$  có hàm ngược và chúng ta có thể tính được  $\Pr\{Y \leq F_Y^{-1}(z)\}$  với  $0 \leq z \leq 1$ . Bởi vì  $F_Y(y) = \Pr\{Y \leq y\}$ , chúng ta sẽ có:

$$F_Z(z) = F_Y(F_Y^{-1}(z)) = z \quad \text{với } 0 \leq z \leq 1 \quad (6.4)$$

Điều đó có nghĩa là  $Z$  là hàm phân bố đồng nhất trong khoảng  $[0,1]$ . Để phát ra số ngẫu nhiên với hàm phân bố  $F_Y(y)$ , chúng ta có thể thực hiện như sau. Chúng ta sử dụng hàm phân bố ngẫu nhiên đồng nhất phát ra các số ngẫu nhiên  $z$  và áp dụng hàm ngược  $y = F_Y^{-1}(z)$ ,  $y$  lúc này sẽ tuân theo hàm phân bố theo  $F_Y$ .



**Hình 6.6. Tạo biến ngẫu nhiên từ các biến ngẫu nhiên phân tán đồng nhất**

### Hàm phát số ngẫu nhiên có phân bố mũ âm

Để phát ra dãy số ngẫu nhiên từ hàm phân bố mũ âm (là phân bố tuân theo công thức  $F_Y(y) = 1 - e^{-\lambda y}$ ,  $y > 0$ ), chúng ta sẽ thực hiện như sau. Chúng ta giải  $z = F_Y(y)$  với  $y$  để tìm được  $y = -\ln(1 - z)/\lambda$ . Như vậy,

chúng ta sẽ phát ra dãy số ngẫu nhiên  $z$  phân bố đồng nhất và áp dụng phương trình trên để nhận được các số ngẫu nhiên  $y$  có phân bố mũ âm.

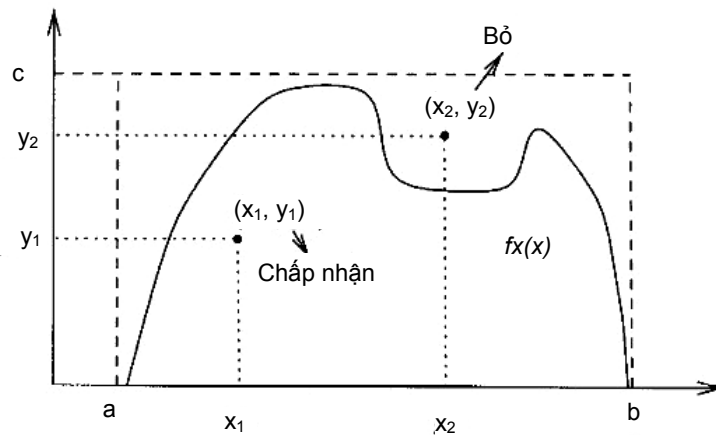
### Hàm phát số ngẫu nhiên theo phân bố Erlang- $k$

Để phát số ngẫu nhiên theo phân bố *Erlang- $k$* , chúng ta phát ra  $k$  số ngẫu nhiên, phân bố theo hàm mũ âm sau đó cộng chúng lại. Để có thể tránh phép tính  $k$  logarithm, chúng ta có thể làm như sau. Gọi  $u_1, \dots, u_k$  là  $k$  số ngẫu nhiên theo phân bố đồng nhất và gọi  $x_i = -\ln(u_i) / \lambda$  là  $k$  số ngẫu nhiên theo phân bố mũ âm tương ứng (với  $\lambda$  là tốc độ đến). Chúng ta tính toán số  $z$  theo phân bố *Erlang- $k$*  như sau:

$$z = \sum_{i=1}^k x_i = -\frac{1}{\lambda} \sum_{i=1}^k \ln(u_i) = -\frac{1}{\lambda} \left( \prod_{i=1}^k u_i \right) \quad (6.5)$$

### Bộ phát số ngẫu nhiên theo phân bố siêu mũ (*hyper-exponential*)

Phân bố siêu mũ có thể được biểu diễn như là sự lựa chọn  $n$  phân bố mũ âm, mỗi phân bố có tốc độ  $\lambda_i$ . Trước hết chúng ta phát ra số nguyên ngẫu nhiên  $i$  từ tập hợp  $\{1, \dots, n\}$ . Sau đó phát ra dãy số ngẫu nhiên có phân bố mũ âm với tốc độ  $\lambda_i$ .



Hình 6.7. Phương pháp phát lại

### Bộ phát số ngẫu nhiên theo phân bố chuẩn

Đối với một vài biến ngẫu nhiên, hàm phân bố thường không xác định được hoặc không có hàm ngược. Ngoài ra hàm mật độ xác suất của chúng

cũng không có một miền giá trị xác định. Trong trường hợp này, chúng ta phải có các phương pháp khác để tạo ra các số ngẫu nhiên.

Giả sử chúng ta quan tâm đến phân bố chuẩn và áp dụng định lý giới hạn trung tâm để tính toán các số ngẫu nhiên theo phân bố chuẩn như sau. Trước hết chúng ta tạo ra  $n$  số ngẫu nhiên độc lập có phân bố giống nhau  $x_1, \dots, x_n$ . Các số ngẫu nhiên này có thể được coi như các biến ngẫu nhiên  $X_1, \dots, X_n$  đều là của biến  $X$  với trung bình  $E(X)$  và phương sai  $V(X)$ . Chúng ta định nghĩa biến ngẫu nhiên  $S_n = X_1 + \dots + X_n$ . Định lý giới hạn trung tâm đã chứng minh rằng:

$$N = \frac{S_n - nE(X)}{\sqrt{nV(X)}} \quad (6.6)$$

Biến  $N$  sẽ gần đến biến ngẫu nhiên có phân bố chuẩn với trung bình là 0 và phương sai 1, ký hiệu là  $N(0,1)$ .

Bây giờ, nếu chúng ta thực hiện biến  $X$  theo phân bố đồng nhất trong khoảng  $[0,1]$ , tức là  $X$  có giá trị trung bình  $\frac{1}{2}$ , phương sai  $\frac{1}{12}$  và lấy  $n = 12$  mẫu,  $S_{12} = X_1 + X_2 + \dots + X_{12}$  sao cho:

$$N = \frac{S_n - nE(X)}{\sqrt{nvar(X)}} = \frac{S_n - 6}{\sqrt{12 \frac{1}{12}}} \quad (6.7)$$

thì  $N$  sẽ tiến dần tới phân bố chuẩn  $N(0,1)$ . Ưu điểm lớn nhất của việc sử dụng  $N$  là nó được tính toán dễ dàng. Tất nhiên khi lựa chọn giá trị  $n$  càng lớn thì độ chính xác của dãy số ngẫu nhiên tạo ra càng lớn.

## 6.2. ĐÁNH GIÁ THỐNG KÊ KẾT QUẢ MÔ PHÒNG

Một chương trình mô phỏng theo phương pháp sự kiện rời rạc sẽ cho phép đánh giá về mặt định lượng các hệ thống thực tế. Khi thực hiện chương trình mô phỏng, các sự kiện liên quan có thể được gắn nhãn. Các kết quả nhận được của chương trình mô phỏng được ghi vào một tệp vết hay tệp log. Mặc dù các tệp này chứa tất cả các thông tin cần thiết, nhưng thông thường

nó ít khi được sử dụng đến. Để có thể nhận được các kết quả rõ ràng hơn, người ta phải tiến hành **xử lý thống kê các dữ liệu** này.

### 6.2.1. Các kết quả thu được

Trong phần này chúng ta sẽ xem xét các vấn đề xảy ra khi xử lý các mẫu dữ liệu riêng biệt và loại bỏ những mẫu không cần thiết.

#### Lấy mẫu các sự kiện riêng biệt

Ở đây chúng ta phân biệt hai phương pháp đo đặc thông qua mô phỏng: phương pháp đo đặc do người dùng thực hiện và phương pháp đo đặc do hệ thống thực hiện. Phương pháp đầu tiên là quá trình người dùng quan sát một tham số nào đó trong một khoảng thời gian nhất định. Một ví dụ của phương pháp này là thời gian lưu lại của yêu cầu trong một bộ phận nào đó của hệ thống. Giả sử khi yêu cầu thứ  $i$  đi vào hệ thống, một nhãn  $t_i^{(a)}$  được tạo ra và khi yêu cầu rời khỏi hệ thống, một nhãn  $t_i^{(d)}$  cũng được tạo ra. Khoảng thời gian yêu cầu ở trong hệ thống  $t_i = t_i^{(d)} - t_i^{(a)}$  sẽ chính là thời gian lưu lại của yêu cầu đó trong hệ thống. Bằng cách tính toán cho tất cả các yêu cầu được mô phỏng, chúng ta sẽ tính được thời gian tồn tại trung bình của  $n$  yêu cầu trong hệ thống đó là:

$$\bar{r} = \frac{1}{n} \sum_{i=1}^n (t_i^{(d)} - t_i^{(a)}) = \frac{1}{n} \left( \sum_{i=1}^n t_i^{(d)} - \sum_{i=1}^n t_i^{(a)} \right) \quad (6.8)$$

Lưu ý rằng trong quá trình mô phỏng, chúng ta không cần thiết phải ghi lại toàn bộ các nhãn thời gian đã tạo ra ở trên, vì khi kết thúc chúng ta chỉ quan tâm đến sự khác nhau của chúng.

Đối với phương pháp đo đặc bằng hệ thống, sẽ không có yêu cầu nào được quan sát mà yếu tố được quan sát ở đây là trạng thái của hệ thống. Một ví dụ điển hình của trường hợp này là khi đo đặc xác suất để một hàng đợi bị đầy. Khi hàng đợi bắt đầu có yêu cầu đến, một nhãn  $t_i^{(f)}$  sẽ được tạo ra và khi hàng đợi đạt mức ngưỡng thì nhãn  $t_i^{(n)}$  được tạo ra. Khoảng thời gian hàng đợi này bằng chính là  $t_i = t_i^{(n)} - t_i^{(f)}$ . Tổng cộng của tất cả các thời gian



này, chia cho thời gian tổng của mô phỏng, chính là xác suất để hàng đợi bị đầy:

$$\bar{b} = \frac{1}{T} \sum_{i=1}^n (t_i^{(n)} - t_i^{(f)}) = \frac{1}{T} \left( \sum_{i=1}^n t_i^{(n)} - \sum_{i=1}^n t_i^{(f)} \right) \quad (6.9)$$

Ở đây chúng ta giả thiết rằng trong khoảng thời gian  $T$  có  $n$  chu kỳ hàng đợi đó bị đầy.

### **Loại bỏ các kết quả chuyển tiếp ban đầu**

Với phần lớn các chương trình mô phỏng, chúng ta cố gắng đo đặc các thông số khi hệ thống ở trạng thái tĩnh. Tuy nhiên, khi bắt đầu mô phỏng, trạng thái của hệ thống biến đổi rất khác nhau. Như vậy việc quan sát và đo đặc các kết quả ban đầu của hệ thống sẽ không mang lại kết luận chính xác về hệ thống, chính vì vậy những số liệu ban đầu này nên được bỏ qua. Tuy nhiên, lại có một vấn đề đặt ra là, khoảng thời gian chuyển tiếp này kéo dài bao lâu là vừa đủ? Để tính toán khoảng thời gian chuyển tiếp này, có rất nhiều cách khác nhau, cụ thể như sau:

Cách đầu tiên là thời gian mô phỏng phải đủ lâu để ảnh hưởng của khoảng thời gian chuyển tiếp này là nhỏ. Tất nhiên đây không phải là một cách hiệu quả vì rất khó định nghĩa được khi nào ảnh hưởng của khoảng thời gian chuyển tiếp là nhỏ.

Một cách nữa là phương pháp phân đoạn (*truncation method*). Phương pháp này sẽ loại bỏ  $l < n$  mẫu từ dãy  $x_l, \dots, x_n$  khi  $l$  là giá trị nhỏ nhất sao cho:

$$\min \{x_{l+1}, \dots, x_n\} \neq x_{l+1} \neq \max \{x_{l+1}, \dots, x_n\} \quad (6.10)$$

Một phương pháp khác nữa là phương pháp dựa trên việc đánh giá phương sai. Giả sử chúng ta có  $n$  mẫu. Gọi  $m$  là trung bình mẫu được tính bởi:

$$m = \frac{\left( \sum_{i=1}^n x_i \right)}{n}$$

Sau đó chúng ta chia  $n$  thành  $k$  bó, sao cho  $k = [n/l]$ . Chúng ta bắt đầu với bó có kích thước  $l = 2$  và tăng dần từng bước một, tính các giá trị  $k$

tương ứng, cho đến khi phương sai của mẫu bắt đầu giảm. Chúng ta tính trung bình của bó như sau:

$$m_i = \frac{1}{l} \sum_{j=1}^l x_{(i-1)l+j} \quad \text{với } i=1, \dots, k \quad (6.11)$$

và phương sai của mẫu được tính bởi:

$$\sigma(m)^2 = \frac{1}{k-1} \sum_{i=1}^k (m_i - m)^2$$

Bằng cách tăng kích thước của bó  $l$ , sẽ có thêm các mẫu của quá trình chuyển tiếp trở thành một phần của bó đầu tiên. Nếu  $l$  là nhỏ, rất nhiều bó sẽ chứa các mẫu của khoảng thời gian chuyển tiếp và làm cho phương sai lớn hơn. Hiển nhiên, khi  $l$  tăng lên đủ lớn để cho bó thứ nhất chứa gần đủ các mẫu của giai đoạn chuyển tiếp ban đầu, chỉ có  $m_1$  sẽ khác khá nhiều so với  $m$ , do đó phương sai của mẫu sẽ lại giảm. Kích thước của bó  $l$  mà với giá trị đó phương sai của mẫu bắt đầu giảm chính là số mẫu cần phải bỏ đi.

### 6.2.2. Giá trị trung bình và khoảng tin cậy (Confidence Intervals)

Giả sử chúng ta phải thực hiện một chương trình mô phỏng để đánh giá giá trị trung bình của biến ngẫu nhiên  $X$  là  $E(X) = a$ . Để thực hiện điều này, chương trình mô phỏng sẽ phải tạo ra  $n$  mẫu  $x_i$  với  $i = 1, \dots, n$ , trong đó mỗi mẫu có thể được thực hiện từ một biến ngẫu nhiên  $X_i$ . Chương trình mô phỏng này được tạo ra sao cho các biến  $X_i$  cũng phân bố giống biến ngẫu nhiên  $X$ . Tiếp theo, để tính toán khoảng tin cậy, các số  $X_i$  phải không phụ thuộc vào nhau.

Sau đây chúng ta sẽ xem xét cách đánh giá giá trị trung bình của  $X$  và giới thiệu cách tính khoảng tin cậy cho kết quả vừa nhận được.

#### Giá trị trung bình

Để đánh giá  $E(X)$ , chúng ta gọi biến  $\tilde{X}$  là biến đánh giá của  $X$ . Nếu có  $E(\tilde{X}) = a$ , thì lúc này  $\tilde{X}$  được gọi là đánh giá không trượt (*unbiased*).

Khi  $Pr\{|\tilde{X} - a| < \varepsilon\} \rightarrow 0$  khi  $n \rightarrow 0$ , lúc đó biến đánh giá  $\tilde{X}$  được gọi là ổn định (*consistent*) vì điều kiện ổn định này tương đương với điều kiện  $V(\tilde{X}) \rightarrow 0$  khi  $n$  tiến đến vô cùng. Một cách hiển nhiên thì mục đích quan trọng nhất của một biến đánh giá là phải không trượt và ổn định. Khi  $X_1, \dots, X_n$  độc lập thì:

$$\tilde{X} = \frac{1}{n} \sum_{i=1}^n X_i \quad (6.12)$$

là biến đánh giá không trượt và ổn định cho  $E(X)$  bởi vì  $E(\tilde{X}) = E(X)$  và  $V(\tilde{X}) = V(X)/n$ , tức là  $V(\tilde{X}) \rightarrow 0$  khi  $n \rightarrow \infty$ .

Mặc dù biến đánh giá  $\tilde{X}$  được coi là có chất lượng tốt nhưng nó lại có yêu cầu là các biến ngẫu nhiên  $X_i$  phải độc lập với nhau. Trong thực tế, các mẫu liên tiếp nhau được tạo ra bởi mô phỏng thường phụ thuộc vào nhau. Vì vậy để tạo ra các mẫu độc lập với nhau, có thể có nhiều phương pháp như sau.

#### **Tạo biến ngẫu nhiên độc lập**

Để tạo ra các biến độc lập, có ba phương pháp phổ biến như sau:

Phương pháp tạo biến ngẫu nhiên giả độc lập (*independent replicas*) là phương pháp mô phỏng được lặp lại  $n$  lần, mỗi lần có một giá trị mầm khác nhau cho RNG. Với mô phỏng lần thứ  $i$ , các mẫu  $x_{i,1}, \dots, x_{i,m}$  được tạo ra. Mặc dù các mẫu riêng biệt này không độc lập với nhau, giá trị trung bình  $\bar{x}_i = \left( \sum_{j=1}^m x_{i,j} \right) / m$  với  $i = 1, \dots, n$  vẫn được coi là độc lập với nhau.  $N$  giá trị trung bình này được coi như các mẫu có giá trị trung bình và khoảng tin cậy đã biết. Nhược điểm của phương pháp này là mô phỏng phải chạy từ thời điểm bắt đầu rất nhiều lần, dẫn đến khoảng thời gian chuyển tiếp cũng phải lặp lại nhiều lần.

Để khắc phục nhược điểm này, người ta sử dụng phương pháp trung bình bó (*batch means*). Phương pháp này đòi hỏi chỉ chạy một mô phỏng

duy nhất và các mẫu  $x_1, \dots, x_{n-m}$  được chia là  $n$  bó với kích thước  $m$ . Với mỗi bó, các mẫu được lấy giá trị trung bình như sau:

$$y_i = \frac{1}{m} \sum_{j=1}^m x_{(i-1)m+j}$$

Các mẫu  $y_1, \dots, y_n$  được giả thiết coi như độc lập và được sử dụng để tính trung bình và khoảng tin cậy. Ưu điểm của phương pháp này là chỉ chạy mô phỏng một lần và do đó chỉ phải loại bỏ các giá trị chuyển tiếp ban đầu một lần. Nhược điểm của phương pháp này là các mẫu không phải hoàn toàn độc lập với nhau, hay phương pháp này cũng chỉ là phương pháp gần đúng.

Một phương pháp khác cho phép khắc phục nhược điểm không độc lập của các bó của phương pháp trên là phương pháp phát lại (*regeneration*). Với phương pháp này việc thực hiện một mô phỏng được chia làm nhiều bó, tuy nhiên việc chia này được thực hiện tại các điểm phát lại của mô phỏng. Các điểm phát lại này là các điểm mà bắt đầu từ điểm này, hệ thống sẽ hoạt động một cách hoàn toàn độc lập so với hoạt động trước đó của nó. Ví dụ như với hàng đợi  $M/G/1$ , điểm bắt đầu phát lại của hàng đợi này chính là điểm mà hàng đợi không có một yêu cầu nào trong nó. Sau khi chia mô phỏng thành những bó rồi, thì lúc này trung bình của các bó có thể được coi như các mẫu để tính trung bình chung và khoảng tin cậy.

Ưu điểm của phương pháp phát lại này là các mẫu được sử dụng để tính toán khoảng tin cậy độc lập với nhau. Tuy nhiên việc tìm thấy các điểm phát lại thực sự của hệ thống không phải dễ dàng.

### **Khoảng tin cậy**

Khi các biến ngẫu nhiên  $X_i$  được coi là độc lập và có phân bố giống nhau, biến đánh giá  $\tilde{X}$  được định nghĩa như ở trên, theo định lý giới hạn trung tâm, sẽ gần xấp xỉ với phân bố chuẩn với trung bình là  $a$  và phương sai là  $\sigma^2/n$  theo định lý giới hạn trung tâm. Điều này dẫn đến biến ngẫu nhiên:

$$Z' = \frac{\tilde{X} - a}{\sigma / \sqrt{n}}$$

sẽ có phân bố chuẩn  $N(0,1)$ . Tuy nhiên, vì chúng ta không biết được phương sai của biến  $X$ , nên phải tìm cách đánh giá nó. Một biến đánh giá không trượt cho phương sai, hay chính là phương sai, được xác định bởi:

$$S^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \tilde{X})^2$$

Lúc này biến:

$$Z' = \frac{\tilde{X} - a}{\tilde{S} / \sqrt{n}}$$

sẽ có phân bố *Student t* với bậc tự do  $n - 1$ . Lưu ý rằng  $a$  và  $\tilde{\sigma}^2$  có thể dễ dàng được tính toán khi  $\sum_i x_i$  và  $\sum_i x_i^2$  đã biết trước:

$$a = \frac{\sum_{i=1}^n x_i}{n}$$

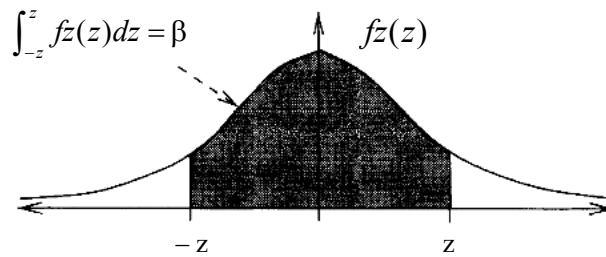
và:

$$\tilde{\sigma}^2 = \frac{\sum_{i=1}^n x_i^2}{n-1} - \frac{\left(\sum_{i=1}^n x_i\right)^2}{n(n-1)}$$

Phân bố Student với ba hay nhiều hơn bậc tự do là phân bố hình chuông đồng trục, gần giống hình dạng của phân bố chuẩn. Khi  $n \rightarrow \infty$ , phân bố này gần đến phân bố  $N(0,1)$ . Bằng cách sử dụng bảng dưới, chúng ta có thể tìm được giá trị  $z > 0$  sao cho  $Pr\{|Z| \leq z\} = \beta$  với  $z$  được gọi là giá trị ngưỡng hai bên của phân bố  $t_n$  với giá trị  $\beta$  cho trước. Dòng cuối cùng ở bảng dưới tương ứng với trường hợp giá trị ngưỡng theo luật  $N(0,1)$ . Sử dụng giá trị ngưỡng hai bên này, chúng ta có thể viết:

$$Pr\{|Z| \leq z\} = Pr\left\{\left|\frac{\tilde{X} - a}{\sigma / \sqrt{n}}\right| \leq z\right\} = Pr\{|\tilde{X} - a| \leq z\sigma / \sqrt{n}\} = \beta \quad (6.13)$$

Có nghĩa là xác suất để biến đánh giá  $\tilde{X}$  sai khác nhỏ hơn  $z\sigma / \sqrt{n}$  từ giá trị trung bình  $a$  là  $\beta$ . Nói cách khác, xác suất để  $X$  nằm trong khoảng tin cậy  $|a - z\sigma / \sqrt{n}, a + z\sigma / \sqrt{n}|$  là  $\beta$ . Xác suất  $\beta$  này được gọi là mức tin cậy. Chúng ta có thể thấy rằng, để giảm khoảng tin cậy đi  $l$  lần, thì mô phỏng phải kéo dài thêm  $l^2$  lần.



**Hình 6.8. Phân bố Student với số bậc tự do lớn hơn 3**

Ví dụ, khi chạy 5 lần mô phỏng, chúng ta nhận được 5 mẫu  $(x_1, \dots, x_5) = (0,108; 0,112; 0,111; 0,115; 0,098)$ . Trung bình của các mẫu này là:

$$a = \frac{\sum_{i=1}^5 x_i}{5} = 0,1088$$

Và tính được:  $\sigma^2 = \sum_{i=1}^5 (x_i - a)^2 / (5 - 1) = 0,0000427$ . Giả sử mức tin cậy ở đây là  $\beta = 0,9$ . Từ bảng 6.1, chúng ta tìm được  $z = 2,132$  cho phân bố  $t_4$  với mức tin cậy 90%. Từ đó ta có:

$$Pr\{|Z| \leq 2,123\} = Pr\{|\tilde{X} - m| \leq 2,123\sigma / \sqrt{5}\}$$

và nhận được:  $Pr\{|\tilde{X} - m| \leq 0,00623\} = 0,90$ . Như vậy chúng ta biết rằng:

$$\tilde{X} \in [0,1026; 0,1150] \text{ với mức tin cậy } 90\%.$$

**Bảng 6.1. Giá trị ngưỡng hai bên của phân bố Student**

$n$	$\alpha = 90\%$	$\alpha = 95\%$	$\alpha = 99\%$	$n$	$\alpha = 90\%$	$\alpha = 95\%$	$\alpha = 99\%$
3	2.353	3.182	5.841	18	1.734	2.101	2.878
4	2.132	2.776	4.604	20	1.725	2.086	2.845
5	2.015	2.571	4.032	22	1.717	2.074	2.819
6	1.943	2.447	3.707	24	1.711	2.064	2.797
7	1.895	2.365	3.499	26	1.706	2.056	2.779
8	1.860	2.306	3.355	28	1.701	2.048	2.763
9	1.833	2.262	3.250	30	1.697	2.042	2.750
10	1.812	2.228	3.169	60	1.671	2.000	2.660
12	1.782	2.179	3.055	90	1.662	1.987	2.632
14	1.761	2.145	2.977	120	1.658	1.980	2.617
16	1.746	2.120	2.921	$\infty$	1.645	1.960	2.576

### 6.3. GIỚI THIỆU MỘT SỐ CÔNG CỤ MÔ PHỎNG

Hiện tại có nhiều phần mềm mô phỏng rời rạc như *NS2*, *NS3*, *OMNeT++*, *OPNET*, *REAL*, *SSFNet*, *J-Sim* và *QualNet*.... Để có thể mô phỏng các hệ thống truyền thông và mạng máy tính khác nhau, các phần mềm này đều có thể mạnh và điểm yếu riêng. Trong số các công cụ trên, *OPNET* là công cụ mô phỏng mang tính chất thương mại, trong khi đó *NS2* được sử dụng chủ yếu trong lĩnh vực nghiên cứu và là một phần mềm mã nguồn mở được tạo nên bởi rất nhiều các module khác nhau. Ngoài *NS2*, còn có một công cụ mô phỏng khác là *NS3* được phát triển dựa trên *NS2* và bổ sung thêm rất nhiều tài liệu khác nhau. Cuối cùng còn phải kể đến *OMNeT++* là một công cụ mô phỏng với giao diện đồ họa được cải tiến hơn so với các công cụ mô phỏng khác. *OMNeT++* là phần mềm mã nguồn mở được sử dụng rộng rãi trong lĩnh vực nghiên cứu.

**Bảng 6.2. Các công cụ mô phỏng mạng**

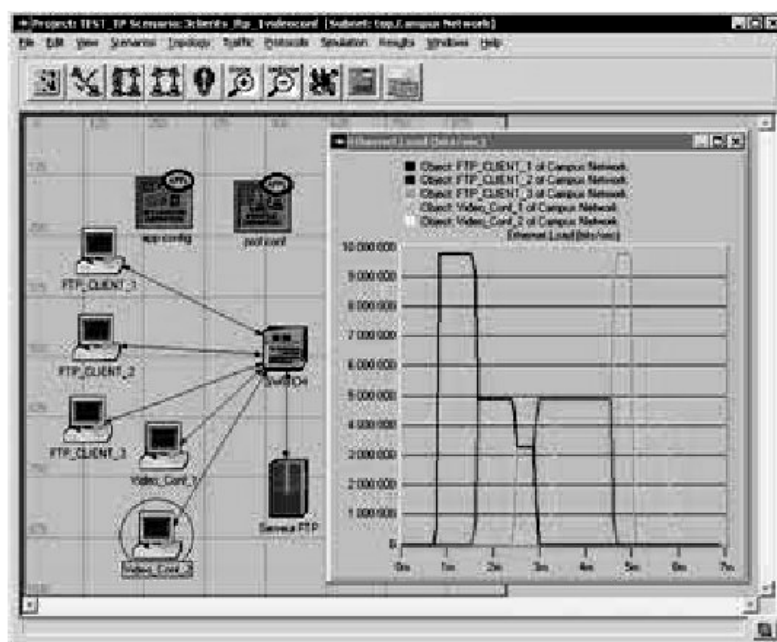
	Công cụ mô phỏng mạng
Thương mại	OPNET, QualNet
Mã nguồn mở	NS2, NS3, OMNeT++, SSFNet, J-Sim

Sau đây chúng ta sẽ đi vào tìm hiểu một số công cụ mô phỏng này.

### 6.3.1. OPNET

*OPNET* là thương hiệu được đăng ký bản quyền bởi công ty *OPNET* Technologies. Công cụ mô phỏng này được cho là rất dễ sử dụng để đánh giá hiệu năng của các mạng truyền thông, các thiết bị, các giao thức và các ứng dụng. Vì là một công cụ thương mại, nên *OPNET* có thể cung cấp cho chúng ta một giao diện đồ họa mạnh và thuận tiện cho người sử dụng. Giao diện này có thể giúp người dùng tạo nên các cấu hình mạng và các thực thể từ lớp vật lý cho đến lớp ứng dụng.

*OPNET* sử dụng kỹ thuật mô phỏng theo sự kiện rời rạc và mạng được tạo ra một cách có cấu trúc. Giống như các công cụ mô phỏng mạng khác, *OPNET* cũng cung cấp giao diện để người sử dụng có thể tự lập trình cho giao thức riêng của mình.



**Hình 6.9. Giao diện OPNET**

*OPNET* thực hiện ba chức năng chính: mô hình hóa, mô phỏng và phân tích. Để mô hình hóa, giao diện đồ họa của nó cho phép tạo nên tất cả các loại giao thức khác nhau. Để mô phỏng, nó sử dụng ba loại kỹ thuật mô phỏng tiên tiến. *OPNET* cho phép các kết quả mô phỏng có thể được hiển thị



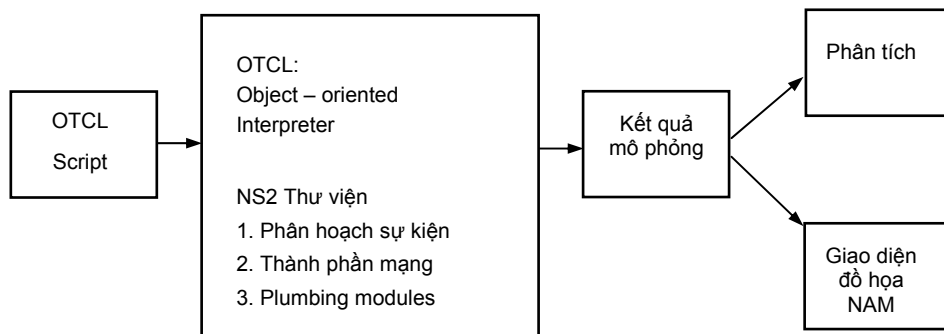
và phân tích một cách rất dễ dàng. Sau đây là các đặc trưng quan trọng nhất của *OPNET*:

- Kỹ thuật mô phỏng theo sự kiện rời rạc.
- Thư viện đầy đủ.
- Kỹ thuật lập trình hướng đối tượng.
- Môi trường mô hình hóa có cấu trúc.
- Hỗ trợ mô phỏng cho các hệ thống không dây.
- Giao diện đồ họa 32 bit và 64 bit.
- Hỗ trợ tính toán lưới.

### **6.3.2. NS2**

*NS2* là công cụ mô phỏng mã nguồn mở được sử dụng rộng rãi nhất, đặc biệt là trong lĩnh vực nghiên cứu. Đầu tiên *NS2* được xây dựng dựa trên phương pháp mô phỏng sự kiện rời rạc và sử dụng chủ yếu để đánh giá hiệu năng cho các loại mạng khác nhau. *NS2* được phát triển dựa trên *NS* (*Network Simulator*) và được đưa ra lần đầu tiên vào năm 1989. *NS2* được tạo nên bởi rất nhiều các module do các tổ chức khác nhau đưa ra.

*NS2* được phát triển lần đầu tiên bởi trường Đại học Berkeley. Ngôn ngữ sử dụng ở đây là *C++* và *OTcl*. Hai ngôn ngữ này được sử dụng với các mục đích khác nhau. *C++* là ngôn ngữ lập trình hướng đối tượng cho phép thiết kế các hệ thống một cách hiệu quả, tuy nhiên phần giao diện hiển thị lại không được thuận tiện. Nhược điểm này của *C++* làm cho việc thay đổi các thông số hoặc cấu hình của hệ thống trở nên khó khăn. Chính vì các nhược điểm này của *C++*, người ta đã sử dụng thêm ngôn ngữ *OTcl*, vốn là một ngôn ngữ lập trình script. Sự kết hợp của hai ngôn ngữ này giúp cho *NS2* có rất nhiều ưu điểm vượt trội: *C++* được sử dụng để triển khai các giao thức phức tạp, trong khi đó *OTcl* được sử dụng để điều khiển các kịch bản khác nhau.



**Hình 6.10. Các module của NS2**

### 6.3.3. NS3

Tương tự như *NS2*, *NS3* cũng là một công cụ mô phỏng mã nguồn mở theo kiểu sự kiện rời rạc. Ý tưởng phát triển *NS3* được đưa ra từ các công cụ khác nhau đã có sẵn như *NS2*, *YANS* và *GTNetS*. Sự khác nhau cơ bản giữa *NS2* và *NS3* là:

- *NS3* sử dụng ngôn ngữ lập trình *C++* và *Python*, trong khi *NS2* sử dụng *C++* và *OTcl*.
- Các phần tử của giao thức của *NS3* được thiết kế gần với hệ thống thật hơn so với *NS2*.
- Cho phép kết hợp với nhiều phần mềm mã nguồn mở mạng khác nhau.
- Cấu trúc đánh dấu: *NS3* cho phép thu thập dữ liệu một cách dễ dàng hơn *NS2*.

Như vậy, có thể thấy so với *NS2*, *NS3* có những ưu điểm sau đây:

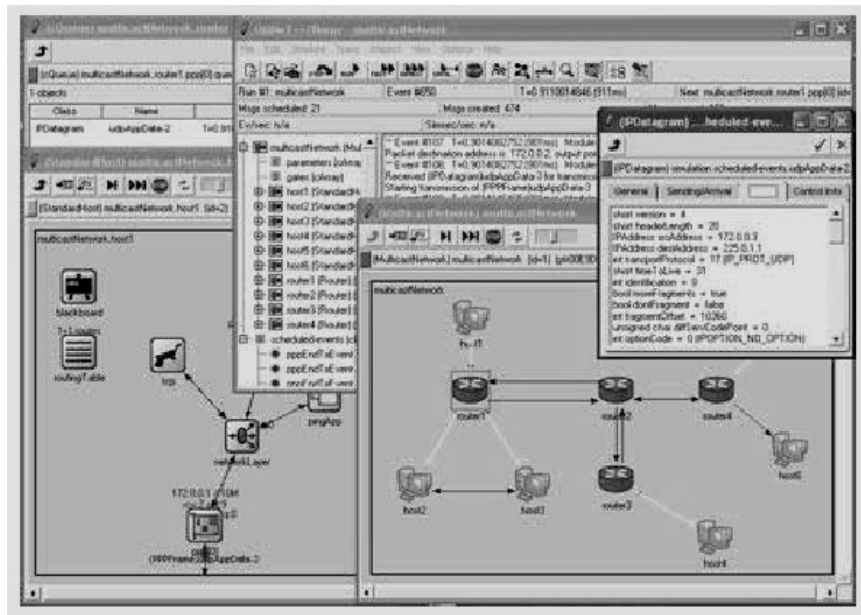
- Ngôn ngữ lập trình *C++* và *Python*.
- Được thiết kế gần với hệ thống thực.
- Có thể kết hợp với các phần mềm mã nguồn mở khác.
- Giao diện đồ họa dễ sử dụng.

Mặc dù vậy, *NS3* vẫn còn một số nhược điểm chưa khắc phục được. Một trong những nhược điểm lớn nhất của nó là độ tin cậy của quá trình mô phỏng vẫn cần phải được cải thiện. Nhược điểm thứ hai của *NS3* là được tạo

nên bởi quá nhiều module do các tổ chức khác nhau đưa ra. Cuối cùng, *NS3* cần phải liên tục được duy trì bởi một số người để có thể phát triển các tài liệu hướng dẫn sử dụng nó.

#### 6.3.4. OMNeT++

*OMNeT++* (*Objective Modular Network Testbed in C++*) là phần mềm mã nguồn mở thuận tiện cho việc nghiên cứu và sử dụng. *OMNeT++* có nhiều phiên bản khác nhau. *OMNeT++* được viết dựa trên hai ngôn ngữ chính là *GNED* và *Visual C++*, kết nối với giao diện đồ họa.



**Hình 6.11. Giao diện OMNeT++**

*OMNeT++* được thiết kế dựa trên các module khác nhau và các module này có thể được sử dụng lại theo nhiều cách để tạo nên các hệ thống khác nhau. *OMNeT++* sử dụng ngôn ngữ lập trình *C++* và có thể chạy trên các hệ điều hành Linux, Unix hoặc Window.

Trong phần sau, chúng ta sẽ đi sâu vào tìm hiểu hai công cụ mô phỏng được sử dụng rộng rãi trong các trường đại học, đó là *NS2* và *OMNeT++*.

## 6.4. NS2 VÀ OMNET++

### 6.4.1. OMNeT++

*OMNeT++* là một ứng dụng cung cấp cho người sử dụng môi trường để tiến hành mô phỏng hoạt động của mạng. Mục đích chính của ứng dụng là mô phỏng hoạt động mạng thông tin, tuy nhiên do tính phổ cập và linh hoạt của nó, *OMNeT++* còn được sử dụng trong nhiều lĩnh vực khác như mô phỏng các hệ thống thông tin phức tạp, các mạng kiểu hàng đợi (*queueing networks*) hay các kiến trúc phần cứng...

*OMNeT++* cung cấp sẵn các thành phần tương ứng với các mô hình thực tế. Các thành phần này (còn được gọi là các module) được lập trình theo ngôn ngữ *C++*, sau đó được tập hợp lại thành những thành phần hay những mô hình lớn hơn bằng một ngôn ngữ bậc cao (*NED*). *OMNeT++* hỗ trợ giao diện đồ họa, tương ứng với các mô hình cấu trúc của nó, đồng thời phần nhân mô phỏng (*simulation kernel*) và các module của *OMNeT++* cũng rất dễ dàng nhúng vào trong các ứng dụng khác.

Các thành phần chính của *OMNeT++*:

- Thư viện phần nhân mô phỏng.
- Trình biên dịch cho ngôn ngữ mô tả hình trạng (*topology description language*) – *NED* (*nedc*).
- Trình biên tập đồ họa (*graphical network editor*) cho các tệp *NED* (*GNED*).
- Giao diện đồ họa thực hiện mô phỏng, các liên kết bên trong các file thực hiện mô phỏng (*Tkenv*).
- Giao diện dòng lệnh thực hiện mô phỏng (*Cmdenv*).
- Công cụ (giao diện đồ họa) vẽ đồ thị kết quả vector ở đầu ra (*Plove*).
- Công cụ (giao diện đồ họa) mô tả kết quả vô hướng ở đầu ra (*Scalars*).
- Công cụ tài liệu hóa các mô hình.
- Các tiện ích khác.
- Các tài liệu hướng dẫn, các ví dụ mô phỏng...

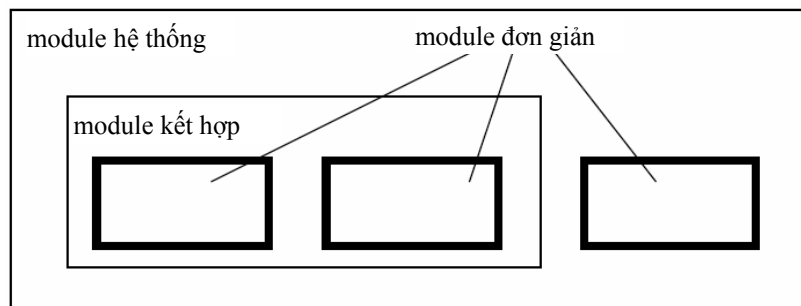
#### 6.4.1.1. Mô hình mô phỏng trong OMNeT++

Một mô hình trong OMNeT++ bao gồm các module lồng nhau có cấu trúc phân cấp. Độ sâu của của các module lồng nhau là không giới hạn, điều này cho phép người sử dụng có thể biểu diễn các cấu trúc logic của các hệ thống trong thực tế bằng các cấu trúc mô hình. Các module trao đổi thông tin với nhau thông qua việc gửi các bản tin. Các bản tin này có thể có cấu trúc phức tạp tùy ý. Các module có thể gửi các bản tin theo hai cách, một là gửi trực tiếp tới địa chỉ nhận, hai là gửi đi theo một đường dẫn được định sẵn, thông qua các cổng và các kết nối. Các module có thể có các tham số của riêng nó. Các tham số này được sử dụng để chỉnh sửa các thuộc tính của module và để biểu diễn cho cấu hình của mô hình.

Các module ở mức thấp nhất trong cấu trúc phân cấp đóng gói các thuộc tính. Các module này được coi là các module đơn giản và chúng được lập trình trong ngôn ngữ C++ bằng cách sử dụng các thư viện mô phỏng.

Mỗi mô hình này thường biểu diễn cho một hệ thống mạng. Module mức cao nhất trong cấu trúc phân cấp được gọi là module hệ thống. Module này có thể chứa các module con, mỗi module con cũng có thể chứa các module con của riêng nó.

Cấu trúc của mô hình có thể được mô tả bằng ngôn ngữ NED của OMNeT++.



Hình 6.12. Cấu trúc module NED

#### Message, cổng, liên kết

Các module trao đổi thông tin bằng việc gửi các bản tin. Trong thực tế, bản tin có dạng khung (*frame*) hoặc các gói tin (*packet*) được truyền đi trong mạng. Các bản tin có thể có cấu trúc phức tạp tùy ý. Các module đơn

giản gửi các bản tin đi một cách trực tiếp đến vị trí nhận hoặc gửi đi theo một đường dẫn định sẵn thông qua các cổng và các liên kết.

Thời gian mô phỏng địa phương (*local simulation time*) của một module tăng lên khi module nhận được một bản tin. Bản tin có thể đến từ một module khác hoặc đến từ cùng một module.

Cổng (*gate*) là các giao tiếp vào ra của module. Bản tin được gửi đi qua các cổng ra và được nhận vào thông qua các cổng vào.

Mỗi kết nối (*connection*) hay còn gọi là liên kết (*link*) được tạo ra bên trong một mức đơn trong cấu trúc phân cấp của các module: bên trong một module kết hợp, một kết nối có thể được tạo ra giữa các cổng tương ứng của hai module con, hoặc giữa cổng của module con với cổng của module kết hợp.

### **Tham số**

Các module có thể có các tham số. Các tham số này được đặt giá trị trong các file *NED* hoặc các tệp cấu hình *ompnetpp.ini*. Người ta có thể dùng các tham số này để thay đổi các thuộc tính của các module đơn giản hoặc dùng để biểu diễn cho cấu hình của mô hình.

Các tham số có thể có kiểu là chuỗi, số học, giá trị logic hoặc cũng có thể chứa cây dữ liệu *XML* (*XML data tree*). Các biến kiểu số trong các biểu thức có thể nhận giá trị từ các tham số khác, gọi hàm, sử dụng các biến ngẫu nhiên từ các nguồn phân tán hoặc nhận giá trị trực tiếp được nhập vào bởi người sử dụng.

Các tham số có kiểu số có thể được dùng để tạo ra các cấu hình rất dễ dàng. Nằm trong các module kết hợp, các tham số này có thể được dùng để chỉ ra số module con, số cổng giao tiếp và cách các kết nối nội bộ được tạo ra.

#### **6.4.1.2. Xây dựng và chạy thử mô hình mô phỏng**

Một mô hình *OMNeT++* bao gồm những phần sau:

- Ngôn ngữ mô tả cấu hình – *NED* (*file có phần mở rộng .ned*): mô tả cấu trúc của module với các tham số, các cổng... Các tệp *.ned* có thể được viết bằng bất kỳ bộ soạn thảo nào hoặc sử dụng chương trình *GNED* có trong *OMNeT++*.

- Định nghĩa cấu trúc của các bản tin (*các tệp có phần mở rộng .msg*): Người sử dụng có thể định nghĩa rất nhiều kiểu bản tin và thêm các trường

dữ liệu cho chúng. *OMNeT++* sẽ dịch những định nghĩa này sang các lớp *C++* đầy đủ.

- Mã nguồn của các module đơn giản: Đây là các tệp *C++* với phần mở rộng là *.h* hoặc *.cc*.

Hệ thống mô phỏng cung cấp cho ta các thành phần sau:

- Phần nhân mô phỏng: Phần này chứa chương trình để quản lý quá trình mô phỏng và các thư viện lớp mô phỏng. Nó được viết bằng *C++*, được biên dịch và được đặt cùng dạng với các tệp thư viện (*các tệp có phần mở rộng là .a hoặc .lib*).

- Giao diện người sử dụng: Giao diện này được sử dụng khi thực hiện quá trình mô phỏng, tạo sự dễ dàng cho quá trình sửa lỗi, biểu diễn hoặc khi thực hiện mô phỏng theo từng khối. Có một vài kiểu giao diện trong *OMNeT++*, tất cả đều được viết bằng *C++* được biên dịch.

Ví dụ, một chương trình mô phỏng được xây dựng cho các giao thức định tuyến bao gồm các thành phần cơ bản sau:

- Tệp *wsn.ned* mô tả các module trong mô hình mạng.

- Các tệp mã nguồn *Sensor.h*, *Sensor.cpp*, *BaseStation.h*, *BaseStation.cpp* xử lý hoạt động các module đơn giản được khai báo trong *wsn.ned*.

- Tệp *RoutingDef.h* khai báo các hằng số được sử dụng trong chương trình bao gồm: giá trị kiểu số phân loại các gói tin, độ dài các loại bản tin...

- Tệp *Message.msg* khai báo các loại gói tin và các trường dữ liệu nằm trong mỗi bản tin.

- Tệp *omnetpp.ini* khai báo các tham số đầu vào cho mô phỏng như kích thước mạng, số nút cảm biến, mức năng lượng ban đầu, giới hạn truyền tải, giới hạn cảm biến...

- Tệp *WSN.sca* lưu các kết quả đầu ra của chương trình như tổng năng lượng toàn mạng sau mỗi vòng, mức năng lượng của mỗi nút, thời gian sống của mạng, tổng số gói tin gửi đi... Dữ liệu đầu ra có dạng bảng và có thể được lọc theo tên hay theo module ghi ra kết quả đó. Dữ liệu có thể được chuyển sang các chương trình như Excel hay Matlab để thực hiện thao tác tính toán tổng hợp hoặc vẽ đồ thị mô tả.

Sau đây là mô tả chi tiết về cấu trúc của hai thành phần quan trọng nhất trong cấu trúc một chương trình mô phỏng *OMNeT++*.

### **Tập mô tả cấu hình mạng *wsn.ned***

Ngôn ngữ *NED* được sử dụng để mô tả cấu hình của một mô hình trong *OMNeT++*. *NED* sử dụng phương pháp mô tả module hóa. Điều này có nghĩa là một mạng có thể được mô tả như một tập hợp các module thành phần (các kênh, các kiểu module đơn giản hay kết hợp). Các kênh, các kiểu module đơn giản và kết hợp được sử dụng để mô tả một mạng nào đó có thể được sử dụng lại khi mô tả một mạng khác.

Nội dung tập *wsn.ned* khai báo lần lượt các module như sau:

- Simple module Sensor: module đơn đại diện cho các nút cảm biến.
- Simple module BaseStation: module đơn đại diện cho trạm gốc.
- Module mạng WSN: bao gồm một module đơn BaseStaion và một tập hợp các module đơn Sensor với số lượng lấy từ file tham số đầu vào *omnetpp.ini*.

Mỗi module đơn sẽ bao gồm các khai báo cơ bản như sau:

- Khai báo các tham số: vị trí *xpos*, *ypos* – của module BS là cố định còn của module Sensor là ngẫu nhiên trong khoảng nhỏ hơn kích thước mạng, tham số giới hạn truyền tải, giới hạn cảm biến, mức năng lượng, chuỗi hiển thị...

Khai báo các cổng: mỗi module đơn sẽ có một tập hợp cổng đầu vào *in[]* và một tập hợp cổng đầu ra *out[]*.

Module mạng cảm biến không dây bao gồm các module con đã nói ở trên, trong khai báo module mạng cũng tiến hành gán giá trị cho các tham số của từng module con thường lấy từ khai báo giá trị tham số trong file *omnetpp.ini*.

### **Các tệp mã nguồn**

Các tệp mã nguồn bao gồm *Sensor.h*, *Sensor.cpp*, *BaseStation.h*, *BaseStation.cpp*. Mã nguồn được viết trên ngôn ngữ *C++* để xây dựng các lớp tương ứng với các module con khai báo trong file *wsn.ned*. Nội dung các tệp này lập trình cách thức thực hiện các sự kiện của mỗi module, hay nói cách khác là thực hiện các hoạt động của mô hình. Các module *Sensor*



và *BaseStation* là các lớp kế thừa từ lớp *SimpleModule* trong thư viện của *OMNeT++*.

Cấu trúc mã nguồn của mỗi module gồm có các hàm chính như sau:

- *Void initialize()*: đây là hàm khởi tạo của lớp. Trong quá trình khởi tạo, *OMNeT++* sẽ xây dựng mạng, nó tạo ra các module đơn và các module kết hợp (*compound module*). Sau đó kết nối chúng theo các khai báo và định nghĩa trong file *NED*, đồng thời với đó là khai báo và gán giá trị cho các biến của module.

- *Void handleMessage(cMessage \*msg)*: hàm này được gọi trong quá trình xử lý sự kiện. Như vậy hầu hết hoạt động của hệ thống được mô phỏng sẽ được lập trình trong các hàm này. Hàm *handleMessage()* sẽ được nhân mô phỏng gọi khi module nhận được một gói tin.

- *Void finish()*: Hàm *finish()* được gọi khi quá trình mô phỏng kết thúc thành công. Ngoài ra, một ứng dụng chủ yếu của hàm này còn là thu thập các thống kê về quá trình mô phỏng. Trong hàm *finish()* của mỗi module thường thực hiện việc ghi thông tin dữ liệu là giá trị hiện thời của các tham số vào tệp kết quả *wsn.sca*.

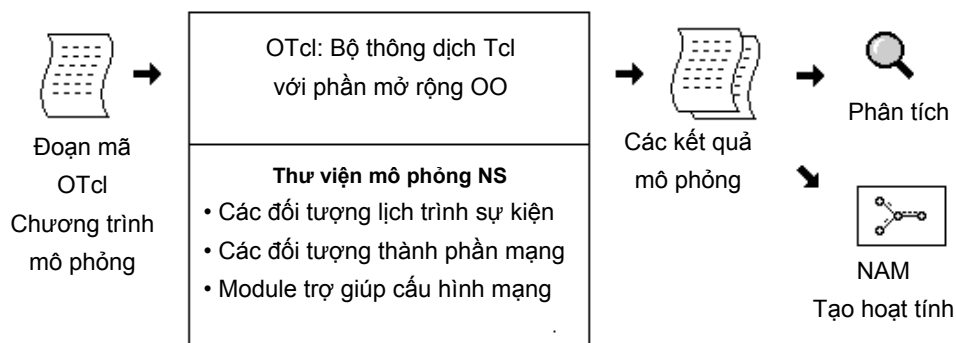
#### 6.4.2. NS2

Công cụ mô phỏng mạng *NS2* là một chương trình mô phỏng sự kiện rời rạc hướng đối tượng nhằm mục đích nghiên cứu hoạt động của mạng. Phiên bản đầu tiên *NS1* ra đời năm 1989 là biến thể của bộ mô phỏng mạng *REAL* ([www.cs.cornell.edu/skeshav/real/overview.html](http://www.cs.cornell.edu/skeshav/real/overview.html)) và được phát triển bởi nhóm nghiên cứu ở phòng thí nghiệm quốc gia Lawrence Berkeley (*Lawrence Berkeley National Laboratory – LBNL*), *USA*. Sau đó nó là một phần của dự án *VINT* được hỗ trợ bởi *DARPA*, *Xerox PARC* và *UCB*. Tiếp đó phiên bản 2 ra đời là sự phát triển của phiên bản 1. Mục đích của *VINT* không phải là thiết kế bộ mô phỏng mạng mới mà là tập hợp các nghiên cứu của nhiều người trong lĩnh vực mô phỏng mạng. Do đó *NS2* được sử dụng rộng rãi trong cộng đồng nghiên cứu mạng và được xem như công cụ để thí nghiệm, kiểm tra những ý tưởng, các giao thức và các thuật toán mới. Hiện tại *NS2* vẫn được hỗ trợ bởi *DARPA*. Trong *NS2* cũng bao gồm sự đóng góp của các tổ chức nghiên cứu và cá nhân khác như các đoạn mã mô phỏng

mạng không dây gồm mạng *ad hoc* và mạng *WLAN* được phát triển bởi *UCB Daedalus*, dự án *CMU Monarch* và *Sun Microsystems*.

Cấu trúc của *NS2* gần giống như mô hình *OSI*, là một mô hình mạng đại diện cho các liên kết của các thành phần mạng. Nó bao gồm các nút và các đường nối. Một hay nhiều bộ khởi tạo lưu lượng như các bộ khởi tạo thống kê và các bộ khởi tạo quen thuộc khác như *FTP*, *Telnet*, có thể được gắn vào bất cứ nút nào. Thêm vào đó, các hoạt động của các giao thức vận chuyển hay giao thức mạng được mô phỏng bằng cách gắn các tác nhân (*agent*) thích hợp vào các nút mong muốn.

*NS2* sử dụng bộ thông dịch kịch bản *Tcl* hướng đối tượng (*OTcl*) bao gồm: một bộ định thời sự kiện mô phỏng, các thư viện đối tượng thành phần mạng, các thư viện module để cấu thành mạng. Nói cách khác để sử dụng *NS2*, bạn phải lập trình bằng ngôn ngữ kịch bản *OTcl*. Để thiết lập và chạy một chương trình mô phỏng mạng, người sử dụng phải viết một đoạn mã *OTcl* để khởi tạo lịch trình sự kiện, thiết lập cấu hình mạng sử dụng các đối tượng mạng và các chức năng cơ bản trong thư viện, cho nguồn lưu lượng biết khi nào bắt đầu và kết thúc truyền gói thông qua bộ lịch trình sự kiện.

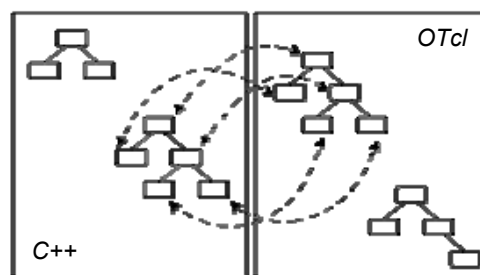


**Hình 6.13. Quy trình hoạt động của NS2**

Một thành phần quan trọng của *NS2* bên cạnh các đối tượng mạng là bộ lịch trình sự kiện. Mỗi sự kiện trong *NS2* là một ID của gói, duy nhất cho gói đó với thời gian được đặt trước và một con trỏ trỏ tới một đối tượng điều khiển sự kiện này. Trong *NS2*, một bộ lịch trình sự kiện sẽ theo thời gian mô phỏng và kích hoạt tất cả các sự kiện trong hàng đợi các sự kiện được định thời trước cho thời điểm hiện tại bằng cách gọi các thành phần mạng

thích hợp, thường là những thành phần sinh ra sự kiện và cho chúng thực hiện những hoạt động đã được liên kết với các gói. Một ứng dụng khác của bộ lịch trình sự kiện là bộ định thời. Bộ định thời đo thời gian liên quan tới gói và thực hiện hành động đã được định trước cho gói sau khi khoảng thời gian trôi qua và không phải mô phỏng khoảng trễ.

Mã nguồn của *NS2* là sự kết hợp của *C++* cho các cơ cấu hoạt động ở nhân và *OTcl*, một phần mở rộng cho *Tcl* do *MIT* thực hiện, để viết các đoạn mã mô phỏng và cấu hình. Sự kết hợp của hai ngôn ngữ tạo cho chương trình sự linh động và dễ sử dụng. Để làm giảm thời gian xử lý sự kiện và gói (không phải thời gian mô phỏng), các bộ lịch trình sự kiện và các đối tượng thành phần mạng cơ bản được viết và được biên dịch bằng *C++*. Các đối tượng đã được biên dịch này luôn sẵn sàng cho bộ thông dịch *OTcl* sử dụng thông qua một liên kết *OTcl*. Liên kết này có nhiệm vụ tạo ra các đối tượng *OTcl* phù hợp cho mỗi một đối tượng *C++* và khiến cho các chức năng điều khiển và các biến có thể cấu hình được chỉ ra bởi các đối tượng *C++* hoạt động như là các hàm thành viên và biến thành viên của của đối tượng *OTcl* tương ứng. Bằng cách này, các điều khiển của đối tượng *C++* được đưa cho *OTcl*. Cũng có thể thêm các hàm và các biến thành viên vào *C++* liên kết với đối tượng *OTcl*. Người sử dụng tạo ra các đối tượng mới thông qua bộ thông dịch *OTcl*. Các thủ tục viết bằng *Tcl* được sử dụng để cung cấp quyền điều khiển linh hoạt trong quá trình mô phỏng (bắt đầu, kết thúc các sự kiện, mạng bị lỗi, thu thập các thông số thống kê và cấu hình mạng). Bộ thông dịch *OTcl* cung cấp các lệnh để tạo ra các nút, các đường liên kết của mạng và các giao thức hoạt động tại các nút.



**Hình 6.14. Sự tồn tại song song của *OTcl* và *C++***

Để đưa một ứng dụng vào *NS2* để mô phỏng cần có **bốn bước**:

- **Đưa giao thức vào** bằng cách thêm các đoạn mã kết hợp giữa *C++* và *OTcl* vào mã nguồn của *NS2*.
- **Mô tả các hoạt động** cần mô phỏng trong các đoạn mã *OTcl*.
- **Chạy mô phỏng**.
- **Phân tích các tệp vết** được tạo ra.

Ứng dụng một giao thức mới cần thêm các mã *C++* cho các chức năng của giao thức cũng như cập nhật các file cấu hình *OTcl NS2* để *NS2* có thể nhận ra giao thức mới và các thông số mặc định của nó. Mã *C++* đồng thời cũng phải mô tả các thông số và phương thức trước khi dùng các đoạn mã *OTcl* để định nghĩa các hoạt động của chúng.

Quá trình mô phỏng được cấu hình, điều khiển và vận hành thông qua việc sử dụng các giao diện cung cấp bởi lớp Simulator của *OTcl*. Lớp này cung cấp các thủ tục để tạo và quản lý mạng, khởi tạo định dạng gói và chọn bộ lịch trình sự kiện. Nó sẽ lưu các tham số vào mỗi thành phần của mạng. Người dùng sử dụng *OTcl* tạo ra mô hình mạng thông qua việc dùng các lớp nút và liên kết để cung cấp các chức năng cơ bản là tạo nút là đường liên kết. Chức năng của một nút là nhận một gói, kiểm tra và đưa nó ra giao diện tương ứng. Một nút bao gồm các đối tượng phân loại (*classifier*) đơn giản là phân loại địa chỉ và phân loại cổng. Chức năng của các bộ phân loại này là để phân phối các gói đến tới đúng tác nhân hoặc tới đường ra khác.

Tác nhân là một thành phần quan trọng của nút: chúng được đặt tại các điểm đầu cuối của mạng nơi gói được tạo ra và xử lý. Người dùng tạo ra các nguồn hoặc các đích mới từ lớp tác nhân. *NS2* hiện tại hỗ trợ rất nhiều kiểu *TCP* khác nhau, *UDP* và một vài giao thức khác như *RTP*, *RTCP*, *SRM*, *SCTP*.

Các đường liên kết có dạng hoặc là một hướng hoặc là cả hai hướng với băng thông, độ trễ và kiểu hàng đợi được định nghĩa ngay khi tạo ra liên kết. Thêm vào đó, các liên kết có thể bị ngắt hoặc khôi phục lại ở bất cứ thời điểm nào của quá trình mô phỏng. Các liên kết được xây dựng từ một chuỗi các đối tượng kết nối (*connectors*). Cấu trúc dữ liệu đại diện cho liên kết bao gồm các đối tượng kết nối của hàng đợi, tiêu đề của nó, loại liên kết, thời gian sống và một đối tượng để xử lý tình trạng mất gói của liên kết. Các

đối tượng kết nối nhận một gói, xử lý gói đó, rồi chuyển cho đối tượng kết nối tiếp theo hay cho đối tượng quản lý tình trạng mất gói. Nhiều loại liên kết được hỗ trợ như điểm tới điểm, quảng bá hay không dây. Hàng đợi được xem như một phần của liên kết. NS2 cho phép mô phỏng nhiều loại hàng đợi và các lịch trình cho gói trong các lớp C++ như: Drop-tail (FIFO), phát hiện sớm ngẫu nhiên (RED), CBQ, hàng đợi theo kiểu công bằng về trọng lượng (WFQ), hàng đợi theo kiểu công bằng về thống kê (SFQ) hay DRR (Deficit Round Robin).

Người dùng phải xác định kiểu định tuyến (*tĩnh, động*) và giao thức được sử dụng. Các đặc điểm của các kiểu định tuyến được hỗ trợ bao gồm định tuyến đối xứng, định tuyến nhiều đường, trạng thái kết nối và thuật toán vector khoảng cách, định tuyến quảng bá, cùng một vài phương pháp định tuyến *ad hoc* khác. Các giao thức mới có thể được đưa vào bằng cách chỉ rõ các tác nhân trong C++ và định nghĩa các thông số liên quan cho bộ thông dịch Tcl.

Rất nhiều loại ứng dụng có thể được mô phỏng. Chẳng hạn như FTP, Telnet, Http đều dùng TCP như là một giao thức ở lớp vận chuyển và các ứng dụng yêu cầu lưu lượng có tốc độ bit cố định (*Constant Bit Rate-CBR*) sử dụng trên nền giao thức UDP.

Tồn thất gói cũng được mô phỏng bằng việc cho tràn bộ đệm ở các router, đây cũng là trường hợp phổ biến của tình trạng mất gói trên Internet. Ngoài ra còn có thể áp dụng các mô hình gây lỗi để làm mất gói với các thông số được đặt trước là số thứ tự gói bị mất, số thứ tự bit bị mất hay thời gian mất gói.

Một cấu hình mạng ngẫu nhiên bao gồm các router, đường liên kết, các phương tiện chia sẻ, có thể được định nghĩa hoặc bằng cách liệt kê các nút mạng và các giới hạn của mạng trong file cấu hình hoặc sử dụng các bộ tạo cấu hình mạng được phát triển cho NS2 (*tiers* hay *GT-ITM*).

Để thu thập các thông tin ở đầu ra hay tệp vết trong mô phỏng, NS2 tạo ra các dấu vết, đó là các bản ghi ghi thông tin về việc tới, đi và mất trên đường đi hay trên hàng đợi của các gói riêng lẻ và các thông tin giám sát, hay là các bản ghi lưu thông tin tổng hợp như sự đến, đi của các gói, byte... Trong đó các vết và thông tin giám sát được liên kết với cả các gói và các

luồng. Bộ tạo ảnh chuyển động mạng (*Network Animator–NAM*) là công cụ hoạt động dựa trên *Tcl/Tk* được dùng để xem các tệp vết của *NS2* phục vụ cho việc tiền xử lý, phân tích và xem lại tình huống mô phỏng (thực ra *NAM* có thể được dùng với bất cứ bộ mô phỏng nào cũng như bất cứ định dạng dữ liệu nào).

Trong *NS2* có hỗ trợ mô phỏng mạng không dây nhưng mới chỉ ở mức mô phỏng việc trao đổi thông tin qua lại hay sự di chuyển giữa hai hay nhiều nút trong đó sử dụng các giao thức định tuyến của mạng *ad hoc* như *AODV*, *DSDV*, *DSR*, *TORA* trên nền *MAC* là 802.11 hay *TDMA*.

Lớp *MobileNode* là sự mở rộng các tính năng cơ bản của lớp cơ sở *Node* bằng cách thêm các chức năng của mạng không dây và di động như khả năng di chuyển trong cấu hình mạng cho trước, khả năng nhận và truyền tín hiệu tới hay từ một giao diện mạng bằng anten,... Thêm vào đó, một nút di động không được kết nối bằng dây với các node khác hay các node di động khác. *MobileNode* là một đối tượng được chia ra thành nhiều phần. Về phần di động bao gồm sự di chuyển của node, sự cập nhật vị trí định kỳ, duy trì giới hạn của cấu hình mạng,... được đưa vào trong *C++* trong khi các thành phần mạng trong bản thân *MobileNode* (giống như *Classifiers*, *MAC*, *Channel*,...) được định nghĩa chức năng trong *OTcl*.

Các thành phần mạng trong nút di động bao gồm:

- Lớp liên kết.
- Module chứa giao thức phân giải địa chỉ (*Address resolution Protocol–ARP*) nối vào lớp liên kết.
- Giao diện hàng đợi ưu tiên để đặt quyền ưu tiên cho các gói giao thức định tuyến và hỗ trợ việc thực hiện bộ lọc trên tất cả các gói trong hàng đợi.
- Lớp *MAC* theo chuẩn *IEEE 802.11* (cũng như giao thức *MAC TDMA*).
- Một bộ phận tiếp nhận (*tap agent*) để nhận, nếu cho phép, tất cả các gói từ lớp *MAC* trước khi tiến hành lọc địa chỉ.
- Giao diện mạng phục vụ như một giao diện phần cứng được nút di động sử dụng để truy nhập vào kênh không dây. Giao diện mạng là đối

tượng để áp dụng mô hình va chạm và mô hình truyền dẫn vô tuyến. Nó sẽ nhận các gói được truyền bởi các giao diện nút tới kênh không dây. Giao diện đánh dấu mỗi gói được truyền với một siêu dữ liệu (*meta-data*) liên quan tới giao diện truyền dẫn như năng lượng truyền dẫn, bước sóng,.... Siêu dữ liệu trong tiêu đề gói này được mô hình truyền dẫn ở giao diện mạng bên nhận xác định liệu gói này có đủ năng lượng để được nhận và/hoặc giữ lấy và/hoặc phát hiện (cảm nhận sóng mang) bởi nút nhận. Mô hình này xấp xỉ giao diện vô tuyến *DSSS* (trải phổ trực tiếp Lucent WaveLan).

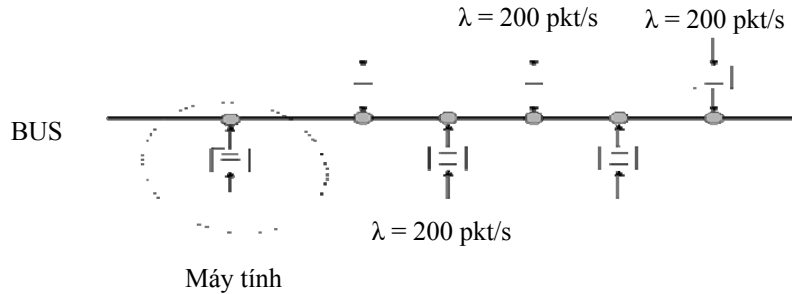
*NS2* là một công cụ mô phỏng phổ biến nhất hiện nay được sử dụng trong lĩnh vực nghiên cứu. *NS2* được trang bị đầy đủ các giao thức, mô hình, thuật toán, các công cụ kèm theo khác và tất cả là miễn phí, có mã nguồn mở. Đối với sinh viên đang cần tìm hiểu các mô hình, các giao thức mạng thì đây là chương trình phù hợp nhất.

## BÀI TẬP CHƯƠNG 6

1. Sử dụng phương pháp tuyến tính với  $m = 18$ ,  $c = 17$ ,  $a = 7$  và nhân  $z_0 = 3$  để phát ra các số ngẫu nhiên trong khoảng từ  $(0,17)$ .  
Sau đó sử dụng phương pháp cộng đồng dư để phát ra các số ngẫu nhiên, sử dụng  $m = 18$ ,  $a_0 = \dots = a_{17} = 1$  và sử dụng các nhân ở trên.
2. Sử dụng phương pháp cộng đồng dư ở bài 1 để nhận được 1000 số ngẫu nhiên có phân bố đồng nhất trong khoảng từ  $(0,1)$ .
3. Sử dụng phương pháp mô phỏng phân bố trung bình, chúng ta nhận được các giá trị trung bình sau:  
 $2,45; 2,55; 2,39; 2,41; 2,49; 2,67; 2,38; 2,44; 2,47$ .  
Hãy tính khoảng tin cậy, với  $\alpha = 0,9; 0,95; 0,99$ . Độ rộng của khoảng tin cậy này thay đổi thế nào khi khoảng tin cậy tiến dần đến 1? Cần phải có bao nhiêu bó để giảm độ rộng của khoảng tin cậy đi  $l$  lần?
4. Sử dụng phương pháp mô phỏng theo sự kiện của hàng đợi  $M/M/1$  như đã trình bày ở ví dụ trên, làm thế nào để thay đổi chương trình của  $M/M/1$  để mô phỏng hàng đợi  $M/G/1$  và  $G/G/1$ ?
5. Chúng ta tính xác suất  $p$  của sự kiện ném một đồng xu và nhận được mặt sấp. Giả sử quăng đồng xu  $n$  lần. Giá trị  $n$  phải như thế nào để với khoảng tin cậy  $\alpha = 0,9$  thì độ rộng của khoảng tin cậy chỉ bằng  $0,1$  lần trung bình  $np$ ? Giá trị  $n$  sẽ như thế nào khi  $p$  dần đến 0, nghĩa là khi tung đồng xu rất hiếm khi nhận được mặt sấp?



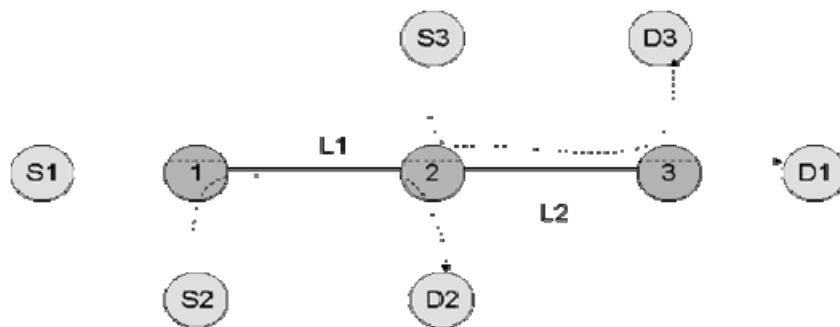
6. Đo hiệu năng của mạng Ethernet.



Mạng Ethernet với tốc độ  $10\text{ Mbit/s}$ , sử dụng cấu hình kênh truyền bus (coaxial cable). Trễ truyền dẫn  $100\text{ ms}$ . Biết rằng bộ đệm card mạng của các máy tính đều liên tục có các gói được gửi đến với tốc độ như nhau  $\lambda = 200\text{ gói/s}$ , tuân theo phân bố Poisson. Độ dài của gói Ethernet là  $1500\text{ byte}$ . Chạy mô phỏng dùng *NS2* trong khoảng thời gian  $30\text{ s}$ . Đánh giá và vẽ đồ thị tổng dung lượng băng thông bị chiếm trên bus khi truyền gói  $b(t)$  với tốc độ mất gói  $e(t)$  (tính bằng gói/s) khi số máy tính nối mạng là: 3 máy, 5 máy và 10 máy.

Giả thiết mỗi máy  $x$  sẽ phát gói đến một đích là máy  $y$  bất kỳ do sinh viên tự chọn.

7. Bảng thông công bằng giữa các luồng.



Cho một mạng gồm ba nút như hình vẽ. Nút 1, 2, 3 là các hàng đợi đơn hoạt động theo nguyên tắc FIFO với độ lớn hàng đợi  $K = 5\text{ gói}$ .

Có ba luồng dữ liệu được gửi qua mạng tương ứng là  $(S1, D1)$ ,  $(S2, D2)$  và  $(S3, D3)$ . Trong đó  $S_i$  là nguồn phát dữ liệu còn  $D_i$  là đích. Đường nối  $L1$  có dung lượng là  $1\text{ MB/s}$  trễ lan truyền  $100\text{ ms}$ ; đường  $L2$  có dung lượng  $0,6\text{ Mb/s}$ , trễ lan truyền  $50\text{ ms}$ . Các nguồn  $S_i$  đều phát gói với độ dài cố định là  $125\text{ byte}$ , khoảng thời gian giữa các gói tuân theo phân bố Poisson.

- a. Giả thiết băng thông tối đa tổng cộng mà các luồng được chiếm trên một kênh truyền vật lý bằng 95% dung lượng kênh truyền. Tính tốc độ  $\lambda_{s1}, \lambda_{s2}, \lambda_{s3}$  ( $kbit/s$ ) để ba luồng trên chia sẻ băng thông kênh truyền theo nguyên lý công bằng cực đại – cực tiểu (max – min fairness).
- b. Dựng kịch bản mô phỏng mạng trên với tốc độ các luồng  $\lambda_{s1}, \lambda_{s2}, \lambda_{s3}$  đã được tính toán trong phần trên. Chạy mô phỏng trong  $100s$ .
- c. Vẽ đồ thị băng thông  $r_i(t)$  mà các luồng  $(S1, D1)$ ,  $(S2, D2)$  và  $(S3, D3)$  sử dụng. Vẽ đồ thị tốc độ mất gói  $e_i(t)$  của ba luồng  $(S1, D1)$ ,  $(S2, D2)$  và  $(S3, D3)$  tại nút 3.

## TÀI LIỆU THAM KHẢO

1. [WANsim] WAN simulators and emulators, <http://www.wan-sim.net/>
2. [OPNET] OPNET Modeler, <http://www.opnet.com/>
3. [NS2] NS2 official website, <http://www.isi.edu/nsnam/ns/>
4. [NS2-wiki] NS2 resource webpage,  
[http://nsnam.isi.edu/nsnam/index.php/Main\\_Page](http://nsnam.isi.edu/nsnam/index.php/Main_Page)
5. [NS3] NS3 official website, <http://www.nsnam.org/documents.html>
6. [OMNeT] OMNeT++ official website, <http://www.omnetpp.org/>
7. [REAL] REAL 5.0 simulator overview,  
<http://www.cs.cornell.edu/skeshav/real/overview.html>
8. [SSFNet] Scalable Simulation Framework (SSF), SSFNet homepage,  
<http://www.ssfnet.org/homePage.html>
9. [J-Sim] J-Sim homepage, <http://www.j-sim.org/>
10. [QualNet] QualNet official site, <http://www.scalable-networks.com/products/>
11. [YANS], Yet Another Network Simulator,  
<http://yans.inria.fr/code/yans/?summary>
12. [GTNetS] The Georgia Tech Network Simulator (GTNetS),  
<http://www.ece.gatech.edu/research/labs/MANIACS/GTNetS/>

# ĐÁNH GIÁ HIỆU NĂNG MẠNG

---

NHÀ XUẤT BẢN BÁCH KHOA – HÀ NỘI  
Ngõ 17 Tạ Quang Bửu – Hai Bà Trưng – Hà Nội  
ĐT: 04. 38684569; Fax: 04. 38684570  
[www.nxbbk.hust.edu.vn](http://www.nxbbk.hust.edu.vn)

*Chịu trách nhiệm xuất bản:*

*Giám đốc – Tổng Biên tập:* GVC. TS. PHÙNG LAN HƯƠNG

*Chịu trách nhiệm nội dung:*

*Tác giả:* TS. NGÔ QUỲNH THU  
*Phản biện:* PGS. TS. ĐẶNG VĂN CHUYẾT  
TS. NGUYỄN KIM KHÁNH

*Biên tập:* ĐỖ THANH THÙY  
*Chế bản:* VŨ THỊ HẰNG  
*Trình bày bìa:* TRIỆU VĂN NAM

---

In 300 cuốn khổ 16 × 24cm tại Xưởng thực hành kỹ thuật in ĐHBK – Hà Nội.

Số đăng ký KHXB: 58 – 2013/CXB/92 – 01/BKHN; ISBN: 9786049113420,  
do Cục Xuất bản cấp ngày 10/1/2013.

Số QĐXB: 161/QĐ – ĐHBK – BKHN ngày 7/5/2013.

In xong và nộp lưu chiểu quý III năm 2013.