Tạp chí Tin học và Điều khiển học, T. 13, S. 3 (1997) (21–35)

ỨNG DỤNG LUẬT ĐIỀU KHIỂN MỜ TRONG QUÁ TRÌNH QUYẾT ĐỊNH ĐƯỜNG ĐI TRÊN MẠNG

NGUYỄN HẢI CHÂU

Abstract. This paper presents a fuzzy approach to routing decission in networks. This approach allows us to find out more several paths ther low cost (in fuzzy meaning) to a given destination. A fuzzy rule will be presented in order to extend or restrict set of directions from a source node to a given destination by considering queusing status of the node.

I.ĐẶT VẤN ĐỀ

Trong các mạng máy tính, vấn đề xác định đường đi trong mạng đóng vai trò quan trọng và có ảnh hưởng trực tiếp đến hiệu suất hoạt động của mạng. Chính vì vậy, các thuật toán về xác định đường đi tối ưu trong mạng và đồ thị đã được nghiên cứu nhiều (xem [8]). Các thuật toán về đường đi trong mạng có thể được chia thành hai lớp chính:

- Các thuật toán vectơ khoảng cách (Distance vector), ký hiệu là DV.
- Các thuật toán kết nối trạng thái (Link state), ký hiệu là LS.

Cả hai lớp thuật toán này đều được cài đặt sử dụng trên mạng hiện nay, ví dụ RIP (DV), OSPF (LS)... (xem [9]).

Các thuật toán DV đòi hỏi mỗi nút mạng ghi nhớ chi phí cho việc chuyển các gói tin từ nút đó đến tất cả các đích khác trong mạng. Các chi phí này được tính toán qua một quá trình đệ quy căn cứ vào chi phí chuyển gói tin từ một nút bất kỳ trong mạng đến các nút lân cận của nó. Quá trình này được thực hiện trên tất cả các nút cúa mạng. Khi quá trình kết thúc, mỗi nút sẽ có thông tin về chi phí đến từng đích trong mạng và hướng truyền đến đích đó.

Lớp thuật toán thứ hai được sử dụng là các thuật toán kết nối trạng thái (LS). Khác với DV các thuật toán LS hoạt động phức tạp hơn theo 3 giai đoạn sau:

1. Mỗi nút mạng phải biết được tên các nút lân cận và chi phí đến các nút đó. Các thông tin này được xác định bởi topo mạng và có thể thay đổi khi topo

mạng thay đổi, khi chi phí giữa các nút thay đổi hoặc có một nút mạng có sự cố không hoạt động được.

2. Các nút mạng phải thông báo cho tất cả các nút còn lại thông tin về tin nút, tên các nút lân cận và chi phí cho các kết nối với các nút lân cận đó.

3. Khi giai đoạn 2 kết thúc, mỗi nút trong mạng sẽ có thông tin đầy đủ về topo mạng cũng như chi phí cho tất cả cac kết nối trong mạng và sẽ tính toán được đường đi tối ưu từ nút đó đến các nút khác. Các kết quả nghiên cứu về đường đi tối ưu đã có:

1. Các thuật toán tìm đường đi tối ưu của Dijkstra và Ford.

2. Các thuật toán tìm đường đi tối ưu của Floyd và Dantzig.

3. Thuật toán tìm k đường tối ưu của Shier (còn gọi là thuật toán quét đúp - double sweep, đã được chứng minh đúng năm 1974).

Đối với một mạng tính thì xác định một đường đi tối ưu giữa hai nút là chưa đủ. Vì thế thuật toán tìm một đường đi tối ưu của Dijkstra đã được sửa đổi để có thể tìm tất cả các đường tối ưu (xem [9]) và tất cả các đường đi này có chi phí bằng nhau. Tuy nhiên trong thực tế xác suất để có hai đường đi giữa hai nút mạng có chi phí bằng nhau là rất nhỏ. Vì vậy việc tìm nhiều đường đi tối ưu giữa hai nút nhiều khi chỉ cho kết quả là một đường duy nhất. Nếu chỉ xác định một đường đi tối ưu giữa hai nút sẽ có các nhược điểm sau:

1. Khi nhu cầu truyền dữ liệu trên mạng tăng lên rất dễ xảy ra nghẽn tắc vì thông tin giữa hai nút chỉ đi theo đường nhất định.

2. Hiệu suất hoạt động của mạng không cao.

Qua đó chúng ta thấy tiêu chuẩn tìm nhiều đường đi có chi phí bằng nhau tối ưu nhất trên mạng là chặt và có thể nới lỏng tiêu chuẩn này. Ví dụ giữa hai nút mạng N_i, N_j có 3 đường tối ưu nhất với chi phí là 100, 101, 120. Khi đó có thể coi hai đường có chi phí 100, 101 là "bằng nhau". Như vậy chúng ta sẽ chọn được hai đường đi với chi phí được xem là như nhau. Trong trường hợp đó hiệu suất truyền từ N_i đến N_j qua hai đường có chi phí 100, 101 sẽ tăng lên so với truyền qua một đường duy nhất có chi phí 100. Bằng cách nới lỏng tiêu chuẩn "bằng nhau", cách quyết định đường đi dựa trên luật điều khiển mờ được chúng tôi xây dựng như sau.

II. QUYẾT ĐỊNH ĐƯỜNG ĐI DỰA TRÊN LUẬT ĐIỀU KHIỂN MỜ

1. Các giả thiết

– Mô hình mạng được lựa chọn ở đây là mạng có các nút được kết nối qua các đường truyền kiểu điểm-điểm.

– Cấu trúc ban đầu của các nút mạng được thiết lập gồm các thông tin sau:

+ Tên nút mạng.

+ Tên các nút lân cận, chi phí cho các kết nối đó và chúng ta giả sử chi phí cho các kết nối là một số dương.

– Lựa chọn thuật toán tìm đường đi theo kiểu LS, đồng thời giả sử rằng bước 2 của thuật toán LS hoạt động tốt, tức là mỗi mạng có thông tin đầy đủ và thống nhất về topo mạng và chi phí cho từng kết nối trong mạng. Trên thực tế đã có các bước thuật toán về thông báo topo mạng được cài đặt hoạt động ổn định và cũng được chứng minh trên lý thuyết (xem [9]).

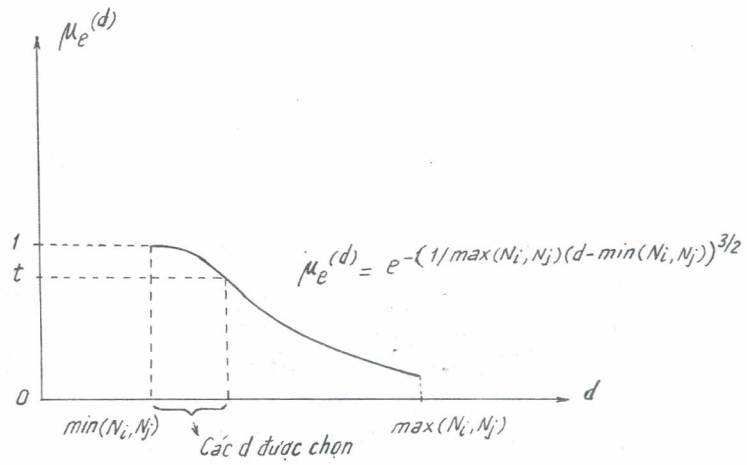
2. Tiêu chuẩn chọn đường đi tối ưu

Giả sử N_i, N_j là 2 nút bất kỳ trong mạng và $\min(N_i, N_j)$, $\max(N_i, N_j)$ tương ứng là chi phí thấp nhất và cao nhất để đi từ N_i đến N_j. Khi đó quá trình chọn đường đi tối ưu (chi phí thấp nhất) có thể được mô tả như sau:

Nếu đường đi P với chi phí $d(P)$ thỏa mãn $d(P) = \min(N_i, N_j)$
Thì bổ sung thêm P vào tập các đường đi có chi phí tối ưu (*).

Tuy nhiên tiêu chuẩn $d(P) = \min(N_i, N_j)$ theo nhận xét ở trên là quá chặt, vì vậy có thể nới lỏng tiêu chuẩn này như sau: Gọi d là chi phí của một đường đi bất kỳ từ N_i đến N_j (d, $\min(N_i, N_j)$, $\max(N_i, N_j)$ là các số dương). Ta có $\max(N_i, N_j) \geq d \geq \min(N_i, N_j) \geq 0$. Chúng ta xây dựng tập mờ $\underset{\sim}{D}$ như sau:

$$\underset{\sim}{D} = \{(d, \mu_{\underset{\sim}{D}}(d)) \mid d \in R^+\}, \text{ trong đó } \mu_{\underset{\sim}{D}}(d) = e^{-(1/\max(N_i, N_j))(d-\min(N_i, N_j))^{3/2}}$$



Trong luật (*), thay vì điều kiện logic thông thường $d(P) = \min(N_i, N_j)$ chúng ta sẽ có điều kiện sau:

$\mu_{\underset{\sim}{D}}(d) = e^{-(1/\max(N_i, N_j))(d-\min(N_i, N_j))^{3/2}}$ với t là tham số có thể thay đổi được.

Luật (*) có thể được chuyển thành luật mờ sau:

Nếu đường đi P với chi phí $d(P)$ thỏa mãn $\underset{\sim}{\mu_D}(d(P)) \geq t$

Thì bổ sung thêm P vào tập các đường đi có chi phí tối ưu (**)

Hàm đo độ thuộc (membership function) $\underset{\sim}{\mu_D}(d)$ được chọn là hàm e với số mũ là 3/2 vì hàm này giảm chậm khi tham số d tăng ít và giảm nhanh khi d tăng nhiều. Chọn số mũ 2 thì giá trị hàm giảm quá nhanh, còn số mũ 1 thì hàm giảm hơi chậm. Ngoài ra tham số của hàm còn chứa hệ số $1/\max(N_i, N_j)$ để có thể so sánh độ giảm nhiều hay ít khi d tăng trong khoảng $[\min(N_i, N_j), \max(N_i, N_j)]$. Ví dụ khi d tăng từ 3 lên 3,1 trong [3, 4] thì hàm $\underset{\sim}{\mu_D}(d)$ sẽ giảm nhiều hơn khi d tăng từ 3 lên 3,1 trong [3, 6]. Trong luật (**) việc chọn ngưỡng t sẽ cho các tập đường đi $\{P\}$ khác nhau. Phần tiếp theo sẽ nêu cách chọn hệ số t (ngưỡng thấp nhất đo mức độ gần nhau của chi phí các đường đi) dựa trên trạng thái hoạt động cúa nút mạng.

3. Chọn ngưỡng t căn cứ vào trạng thái hoạt động cúa nút mạng

Trong phần này chúng tôi sẽ trình bày cách tính ngưỡng t dựa trên mô hình và phương pháp lập luận mờ do Cao và Kandel đề xuất (xem [10]) nhằm mở rộng hay thu hẹp tập các hướng đi ra khỏi nút mạng cho tất cả các gói tin có cùng một địa chỉ đích. Mô hình và phương pháp lập luận mờ sẽ được trình bày dưới đây cùng với phương pháp khử mờ (defuzzification).

a) Mô hình mờ

Mô hình mờ gồm n luật "nếu ... thì...".

Nếu $x = F(1)$ thì $y = G(1)$

Nếu $x = F(2)$ thì $y = G(2)$

...

Nếu $x = F(n)$ thì $y = G(n)$

trong đó x, y là các biến thông thường (ví dụ: cường độ dòng điện, tốc độ quay của một động cơ, tuổi của người,...), $F(i)$, $G(i)$ $i = \overline{1, n}$ là các mô tả bằng lời của các biến x, y (chẳng hạn cường độ dòng điện "tương đối nhỏ", tốc độ quay của động cơ "khá nhanh", người kia "rất trẻ"...). Trong mô hình này, tất cả các mô tả bằng lời $F(i)$, $G(i)$ $i = \overline{1, n}$ được biểu diễn bởi các tập mờ. Mục đích của mô hình mờ nhằm suy ra thông tin mờ của biến y từ các thông tin mờ của biến x và quá trình này được gọi là quá trình lập luận mờ (Fuzzy implication).

b) Phương pháp lập luận mờ

Quá trình lập luận mờ nhằm xác định được tập mờ Y mô tả biến y qua n luật "nếu... thì..." căn cứ trên tập mờ đầu vào X bất kỳ mô tả biến x. Quá trình

lập luận mờ gồm có hai bước: trước hết xây dựng một quan hệ mờ R giữa hai biến x và y, sau đó từ bất kỳ thông tin mờ nào về biến x suy ra thông tin mờ của biến y dựa vào R. Để xây dựng quan hệ mờ R, trước hết chúng ta chuyển mỗi luật "nếu ... thì..." thành các quan hệ mờ $R(i)$ $(i = \overline{1, n})$ tương ứng, sau đó kết hợp n quan hệ mờ này để có được quan hệ mờ R. Nói cách khác chúng ta phải xây dựng các toán tử $\oplus$ và $\otimes$ để:

$F(i) \otimes G(i) = R(i),\ i = \overline{1, n}$

$R(1) \oplus R(2) \oplus \cdots \oplus R(n) = R$

với $F(i)$, $G(i)$ $i = \overline{1, n}$ là các tập mờ, $R(i)$ $i = \overline{1, n}$ và R là các quan hệ mờ.

Sau khi xây dựng được quan hệ mờ R nêu trên chúng ta chuyển sang bước thứ hai của quá trình lập luận mờ: với bất kỳ tập mờ X nào mô tả biến x, chúng ta luôn xác định được tập mờ Y tương ứng mô tả biến y bằng cách "kết hợp" (composition) X với quan hệ R : $Y = X \cdot R$ (trong bài này chúng tôi sử dụng luật kết hợp max-min).

Kết thúc quá trình suy luận mờ chúng ta có tập mờ Y là mô tả của biến thông thường y.

c) Khử mờ (defuzzification)

Khử mờ là phương pháp tính giá trị của biến thông thường từ tập mờ mô tả biến đó. Giả sử Y là tập mờ mô tả thông thường y với hàm đo độ thuộc $\mu_Y(y)$. Có nhiều phương pháp khử mờ để tính y từ Y. Ở đây chúng tôi sử dụng phương pháp sau:

$$y = \Big(\sum_{i=1}^{m} y_i\, \mu_Y(y_i)\Big) \,/\, \Big(\sum_{i=1}^{m} \mu_Y(y_i)\Big),\ \forall y_i \in Y\ i = \overline{1, m}$$

Kết quả của quá trình lập luận và khử mờ phụ thuộc vào các yếu tố sau: Cách chọn các hàm đo độ thuộc, cách chọn các luật "nếu ... thì...", cách chọn các toán tử suy diễn $\oplus$, $\otimes$ và cách chọn phương pháp khử mờ, trong đó cách chọn toán tứ suy diễn $\oplus$ và $\otimes$ có ảnh hưởng quan trọng nhất đến kết quả.

Mô hình mờ và phương pháp lập luận mờ nêu trên được áp dụng để xác định ngưỡng t như sau: Giả sử rằng với mỗi hướng đi ra khỏi một nút cho trước đều có một hàng chờ tương ứng và tất cả các hàng chờ này đều có độ lớn như nhau và bằng $q_{\max}$ $(q_{\max} > 0)$. Gọi Q_i là số lượng các gói tin trong hàng chờ của một hướng (nếu đi theo hướng này có thể đến một đích là nút N nào đó), $i = \overline{1, m}$, như vậy $q_1, q_2, ..., q_m$ là số lượng các gói tin trong các hàng chờ của m hướng đi có thể đến được đích N. Ở đây chúng ta có $m \geq n$ với n là số các nút lân cận của nút đang xét. Gọi $q = (q_1 + q_2 + \cdots + q_m)/q_{\max}.m$ = mức độ đầy của các hàng chờ trên các hướng có thể đến đích N (Các hàng chờ này có thể chứa các gói tin đến các đích khác của mạng).

Qui tắc chọn t được xây dựng như sau:

1. Nếu rỗng thì t rất rất gần 1
2. Nếu q tương đối đầy thì t rất gần 1
3. Nếu q khá đầy thì t khá gần 1
4. Nếu q rất đầy thì t tương đối gần 1 (R)

trong đó "rỗng", "tương đối đầy", "khá đầy", "rất đầy", "rất rất gần 1", "rất gần 1", "khá gần 1" và "tương đối gần 1" là các tập mờ có các hàm đo độ thuộc được xác định như sau:

q	rỗng	tương đối đầy	khá đầy	rất đầy
0,0	1,0	0,0	0,0	0,0
0,1	0,8	0,5	0,0	0,0
0,2	0,6	1,0	0,2	0,0
0,3	0,5	0,8	0,6	0,0
0,4	0,4	0,7	1,0	0,1
0,5	0,3	0,7	0,9	0,3
0,6	0,1	0,4	0,6	0,7
0,7	0,0	0,2	0,3	0,7
0,8	0,0	0,1	0,1	0,8
0,9	0,0	0,0	0,0	0,9
1,0	0,0	0,0	0,0	1,0

t	rất rất gần 1	rất gần 1	khá gần 1	tương đối gần 1
0,0	0,0	0,0	0,0	0,0
0,1	0,0	0,0	0,0	0,1
0,2	0,0	0,0	0,0	0,3
0,3	0,0	0,0	0,0	0,5
0,4	0,0	0,0	0,1	0,8
0,5	0,0	0,0	0,5	0,9
0,6	0,0	0,0	1,0	0,9
0,6	0,0	0,0	1,0	0,9
0,7	0,0	0,7	0,6	0,9
0,8	0,0	1,0	0,0	1,0
0,9	0,8	0,4	0,0	1,0
1,0	1,0	0,0	0,0	1,0

Theo phương pháp luận mờ trình bày ở trên, trước hết ta xác định quan hệ mờ $R(q, t)$ bằng toán tử suy diễn (implication operator). Trong [10] Cao và Kandel đưa ra 72 toán tử suy diễn (xem [10]) có thể áp dụng để xác định quan hệ mờ $R(q, t)$. Kết luận nêu trong [10] cho thấy năm toán tử 5*, 22*, 8, 25 và 31 cho kết quả tốt trong tất cả các ví dụ đưa ra trong [10]. Trong quá trình lập luận mờ và khử mờ đối với quy tắc (R) chúng tôi đã cài đặt thử nghiệm hầu hết 72 toán tử suy diễn nói trên khi không thay đổi các hàm đo thuộc "rất đầy", "rất rất gần 1",... (Các toán tử chưa được thử nghiệm là 4, 4*, 15, 15*, 28, 28*, 29, 29*). Căn cứ vào kết quả thử nghiệm có thể thấy rằng đối với qui tắc (R) thì các toán tử suy diễn 8, 25, 31 cho kết quả tốt hơn các toán tử khác. Dưới đây là các quan hệ $R(q, t)$ được xác định tương ứng bằng ba toán tứ 8, 25, 31:

$R_8(q, t)$	0,00	0,10	0,20	0,30	0,40	0,50	0,60	0,70	0,80	0,90	1,00
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,80	1,00
0,10	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,50	0,50	0,80	0,80
0,20	0,00	0,00	0,00	0,00	0,10	0,20	0,20	0,70	1,00	0,60	0,60
0,30	0,00	0,00	0,00	0,00	0,10	0,50	060	0,70	0,80	0,50	0,50
0,40	0,00	0,10	0,10	0,10	0,10	0,50	1,00	0,70	0,70	0,40	0,40
0,50	0,00	0,10	0,30	0,30	0,30	0,50	0,90	0,70	0,70	0,40	0,30
0,60	0,00	0,10	0,30	0,50	0,70	0,70	0,70	0,70	0,70	0,70	0,70
0,70	0,00	0,10	0,30	0,50	0,70	0,70	0,07	0,70	0,70	0,70	0,70
0,80	0,00	0,10	0,30	0,50	0,80	0,80	0,80	0,80	0,80	0,80	0,80
0,90	0,00	0,10	0,30	0,50	0,80	0,90	0,90	0,90	0,90	0,09	0,09
1,00	0,00	0,10	0,30	0,50	0,80	0,90	0,90	0,90	1,00	1,00	1,00

$R_{25}(q, t)$	0,00	0,10	0,20	0,30	0,40	0,50	0,60	0,70	0,80	0,90	1,00
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,80	1,00
0,10	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,35	0,50	0,64	0,80
0,20	0,00	0,00	0,00	0,00	0,20	0,10	0,20	0,70	1,00	0,48	0,60
0,30	0,00	0,00	0,00	0,00	0,06	0,30	0,60	0,56	0,80	0,40	0,50
0,40	0,00	0,01	0,03	0,05	0,10	0,50	1,00	0,60	0,70	0,32	0,40
0,50	0,00	0,03	0,09	0,15	0,24	0,45	0,90	0,54	0,70	0,30	0,30
0,60	0,00	0,07	0,21	0,35	0,56	0,63	0,63	0,63	0,70	0,70	0,70
0,70	0,00	0,07	0,21	0,35	0,56	0,63	0,63	0,63	0,70	0,70	0,70
0,80	0,00	0,08	0,24	0,40	0,64	0,72	0,72	0,72	0,80	0,80	0,80
0,90	0,00	0,09	0,27	0,45	0,72	0,81	0,81	0,90	0,90	0,90	0,90
1,00	0,00	0,10	0,30	0,50	0,80	0,90	0,90	0,90	1,00	1,00	1,00

$R_{31}(q, t)$	0,00	0,10	0,20	0,30	0,40	0,50	0,60	0,70	0,80	0,90	1,00
0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,80	1,00
0,10	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,20	0,50	0,60	0,80
0,20	0,00	0,00	0,00	0,00	0,00	0,00	0,30	0,70	1,00	0,40	0,60
0,30	0,00	0,00	0,00	0,00	0,00	0,10	0,60	0,50	0,80	0,30	0,50
0,40	0,00	0,00	0,00	0,00	0,10	0,50	1,00	0,60	0,70	0,20	0,50
0,50	0,00	0,00	0,00	0,00	0,10	0,40	0,90	0,50	0,70	0,30	0,30
0,60	0,00	0,00	0,00	0,20	0,50	0,60	0,60	0,60	0,70	0,70	0,70
0,70	0,00	0,00	0,00	0,20	0,50	0,60	0,60	0,60	0,70	0,70	0,70
0,80	0,00	0,00	0,10	0,30	0,60	0,70	0,70	0,70	0,80	0,80	0,80
0,90	0,00	0,00	0,20	0,40	0,70	0,80	0,80	0,80	0,90	0,90	0,90
1,00	0,00	0,10	0,30	0,50	0,80	0,90	0,90	0,90	1,00	1,00	1,00

trong đó các toán tử 8, 25, 31 được xác định như sau:

$$\mu_{R_8}(q, t) = \bigvee_{s=1}^{4} (\mu_Q(t) \wedge \mu_T(t))$$

$$\mu_{R_{25}}(q, t) = \bigvee_{s=1}^{4} (\mu_Q(t) \cdot \mu_T(t))$$

$$\mu_{R_{31}}(q, t) = \bigvee_{s=1}^{4} (0 \vee \mu_Q(t) + \mu_T(t) - 1)$$

Khi đó với bất kỳ tập mờ Q nào có hàm đo độ thuộc là μ_Q, chúng ta có thể xác định được tập mờ T tương ứng bằng luật kết hợp max-min: $T = Q \circ R$:

$$\mu_T(t) = \bigvee_{q \in Q} (\mu_Q(q) \wedge \mu_R(q, t)) \quad \forall T$$

sau đó sử dụng phương pháp khử mờ đã nêu để tính giá trị cụ thể của t từ T.

Chúng ta định nghĩa các tập mờ Q_i mô tả mức độ đầy của hàng chờ là $i/10$ $(i = \overline{0, 10})$ có các hàm đo độ thuộc μ_i thỏa mãn:

$$\mu_i(q) = \begin{cases} 1 & \text{nếu } q = i/10 \\ 0 & \text{nếu } q \neq i/10 \end{cases} \qquad i = \overline{0, 10}$$

Thực hiện quá trình lập mờ và khử mờ với các tập mờ Q_i và các quan hệ mờ $R_8(q, t)$, $R_{25}(q, t)$, $R_{31}(q, t)$ chúng ta có kết quả tính ngưỡng t như sau:

Với $R_8(q, t)$: Bảng (T1)

q	t
0,00	0,956
0,10	0,873
0,20	0,791
0,30	0,738
0,40	0,573
0,50	0,618
0,60	0,629
0,70	0,629
0,80	0,637
0,90	0,646
1,00	0,657

Với $R_{25}(q, t)$: Bảng (T2)

q	t
0,00	0,956
0,10	0,883
0,20	0,806
0,30	0,753
0,40	0,695
0,50	0,658
0,60	0,657
0,70	0,657
0,80	0,657
0,90	0,657
1,00	0,657

Với $R_{31}(q, t)$: Bảng (T3)

q	t
0,00	0,956
0,10	0,895
0,20	0,817
0,30	0,775
0,40	0,700
0,50	0,706
0,60	0,702
0,70	0,702
0,80	0,685
0,90	0,673
1,00	0,657

Trong 3 bảng trên chúng ta sẽ dựa vào bảng (T2) cho bởi toán tử suy diễn 25 để tìm các đường đi "gần tối ưu" vì kết quả cho thấy nếu mức độ đầy của hàng chờ tăng lên thì ngưỡng t giảm đi, đồng thời khi hàng chờ đầy đến một mức độ nhất định thì t không thay đổi nữa: t bị chặn dưới giúp chúng ta không chọn các đường đi có chi phí quá lớn so với chi phí tối ưu nhất. Các kết quả trong (T1) và (T3) không cho thấy điều này.

III. TÌM CÁC ĐƯỜNG ĐI VỚI CHI PHÍ "GẦN TỐI ƯU" CĂN CỨ THEO TIÊU CHUẨN MỜ

Trong phần này thuật toán chọn các đường đi với chi phí "gần tối ưu" sẽ được trình bày và so sánh với các thuật toán đã có. Trước hết chúng ta xem xét cách hoạt động và kết quả của các thuật toán đã có.

a) Thuật toán cúa Dijktra cho phép tìm đường đi với chi phí thấp nhất từ một nút đến tất cả các nút còn lại trong mạng. Thuật toán hoạt động đúng khi chi phí cho kết nối giữa hai nút bất kỳ trong mạng là một số dương (Nếu hai nút không có kết nối trực tiếp thì giả sử rằng chi phí kết nối giữa hai nút đó là $+\infty$). Gọi k là số lượng các nút lân cận của một nút mạng bất kỳ và n là tổng số nút mạng. Khi đó độ phức tạp tính toán của thuật toán thực hiện tại nút mạng này sẽ là $O(n.k)$. (xem [9]).

b) Thuật toán của Ford cho kết quả tương tự thuật toán của Dijkstra nhưng có thể tìm đường đi tối ưu trên mạng có chi phí kết nối giữa một số cặp nút là số âm.

c) Các thuật toán của Floyd, Dantzig cho phép tìm đường tối ưu giữa hai cặp nút bất kỳ trong mạng.

d) Thuật toán của Shier tìm k đường tối ưu nhất từ một nút bất kỳ trong mạng đến tất cả các nút mạng.

e) Các thuật toán tổng quát hóa dựa trên cơ sở các thuật toán cúa Floyd, Dantzing cho phép tìm k đường tối ưu nhất giữa hai nút bất kỳ trong mạng.

Để tìm các đường đi "gần tối ưu" căn cứ theo tiêu chuẩn mờ, chúng tôi sẽ lựa chọn một trong các thuật toán đã có để cải tiến. Vì quá trình tìm các đường đi tối ưu được thực hiện độc lập tại các nút mạng nên chỉ cần chọn một trong các thuật toán tìm đường đi tối ưu từ một nút đến các nút khác trong mạng để cải tiến. Ở đây chúng ta đã giả thiết rằng chi phí kết nối giữa hai nút bất kỳ trong mạng là một số dương nên có thể lựa chọn thuật toán của Dijkstra hoặc Shier. Tuy nhiên thuật toán của Shier có một số điểm cần phải xem xét khi lựa chọn:

- Đường đi tối ưu tìm được bởi thuật toán này có thể chứa các nút bị lặp tức là có một số nút xuất hiện l lần trong đường đi với $l > 1$.

- Với các đường đi chi phí bằng nhau thì chỉ có một đường được chỉ ra.

- Không chỉ ra các hướng xuất phát ra khỏi nút xuất phát cho mỗi đường đi có độ dài d nào đó.

- Độ phức tạp lớn: $O(n^3)$ (xem [8]).

Bởi vậy thuật toán của Dijkstra được lựa chọn để cải tiến tìm đường đi tới tối ưu (có tính chi phí thấp). Có nhiều cách phát biểu thuật toán của Dijstra. Ở đây thuật toán được phát biểu lại dựa theo [9] (trang 222). Trước hết chúng ta liệt kê các cấu trúc dữ liệu chính được sử dụng trong thuật toán.

- Cơ sở dữ liệu về topo mạng và chi phí kết nối giữa các nút được ký hiệu là LSDB. Ví dụ có 3 nút A, B có topo mạng và chi phí kết nối được mô tả như sau:

A	B	C
B/2	A/2	A/1
C/1	C/2	B/2

có nghĩa là nút A có 2 nút lân cận B, C với chi phí kết nối tương ứng là 2, 1; B có 2 nút lân cận A, C với chi phí kết nối tương ứng là 2, 2...

- Các tập P và T có các phần tử là bộ ba (tên nút, chi phí, hướng truyền).

Thuật toán của Dijkstra (1)

Vào: Tập các nút mạng NODES, cơ sở dữ liệu LSDB chứa thông tin về topo mạng và các chi phí kết nối, nút xuất phát N_0 thuộc NODES.

Các bước:

1. Gán $T = \emptyset$, $P = \{N_0, 0, N_0)\}$

2. Với mọi nút N trong bộ (nút, chi phí, hướng) vừa được bổ sung vào tập P, Với mọi nút M là nút lân cận của N,

   Tính chi phí c từ N_0 đến M bằng tổng chi phí từ N_0 đến N và chi phí từ N đến M.

   Nếu M không xuất hiện trong một bộ (nút, chi phí, hướng) của P hoặc T với chi phí thấp hơn chi phí c vừa tính toán thì bổ sung (hoặc thay thế nếu đã tồn tại) bộ (M, c hướng) vào T

3. Nếu T rỗng thì kết thúc, ngược lại bộ (N, chi phí, hướng) với chi phí thấp nhất và chuyển bộ này từ T sang P và quay lại bước 2.

Ra: Tập các nút bộ (nút, hướng, chi phí) chỉ ra hướng truyền ra khỏi N_0 và chi phí (tối ưu) đến các nút khác trong mạng.

Thuật toán cho phép tìm một đường tối ưu từ N_0 đến tất cả các nút còn lại trong mạng đồng thời cũng có thể được cải tiến để tìm được tất cả các đường có chi phí bằng nhau đến mỗi nút mạng (xem [9]). Tuy nhiên thuật toán cúa Dijkstra cũng như các thuật toán đã nêu đều chú trọng đến chi phí của đường đi (tối ưu) mà chưa xét đến hướng ra khỏi nút xuất phát. Như đã nêu trên ở phần trên nếu chúng ta tìm được nhiều đường đi với chi phí gần như nhau thì sẽ cải tiến được hiệu suất mạng và phần nào tránh được nghẽn tắc. Tuy vậy nếu tất cả các đường đi đó đều ra khỏi nút xuất phát theo một hướng thì việc tìm các đường đi này không có ý nghĩa. Do đó phương pháp xác định đường đi tối ưu ở đây là: Xác định đường đi tối ưu từ một nút N đến các nút còn lại trong mạng trên từng hướng ra khỏi nút xuất phát N. Như vậy thuật toán của Dijkstra được cải tiến như sau:

Thuật toán Dijkstra cải tiến (2)

Vào: Tập các nút mạng NODES, cơ sở dữ liệu LSDB chứa thông tin về topo mạng và các chi phí kết nối, nút xuất phát N_0 thuộc NODES.

Các bước:

1. Với mọi nút M là nút lân cận của N_0,

   Thực hiện thuật toán (1) với dữ liệu vào là NODES $\{N_0\}$, cơ sở dữ liệu LSDB và nút xuất phát là M.

2. Kết thúc.

Ra: Tập các bộ (nút, chi phí, hướng) chỉ ra hướng truyền, chi phí tối ưu (trên các tất cả các hướng khác nhau ra khỏi N_0) đến tất cả các nút khác N_0 trong mạng.

Giả sử N_0 có k nút lân cận N_{01}, N_{02}, ..., N_{0k}, trong đó có 1 hướng có thể đến được một đích N nào đó ($1 \leq k$). Khi đó có 1 đường đi theo 1 hướng phân biệt ra khỏi N_0 đến N với độ dài $d_1 \leq d_2 \leq ... \leq d_1$ (***). Đến đây chúng ta có thể áp dụng qui tắc (R) để thực hiện việc mở rộng hay thu hẹp các hướng đi ra khỏi N_0. Thuật toán (2) lặp lại k lần thuật toán (1) do đó sẽ có độ phức tạp tính toán là $O(N.m.m)$, trong đó:

$m = \max\{$số nút lân cận của các nút N_0, N_{01}, N_{02}, ..., $N_{0k}\}$.

Sử dụng thuật toán (2) sẽ có các ưu điểm sau:

- Đường đi tìm được không chứa nút nào quá 1 lần.

- Tìm được các đường đi "tối ưu" theo luật (**). Nếu sử dụng thuật toán cúa Shier để tìm k đường tối ưu nhất thì số bước lặp của thuật toán có thể khá lớn (tức là k lớn) vì một đường đi tối ưu thứ 3 theo (***) có thể là đường đi tối ưu thứ 5 theo thuật toán của Shier.

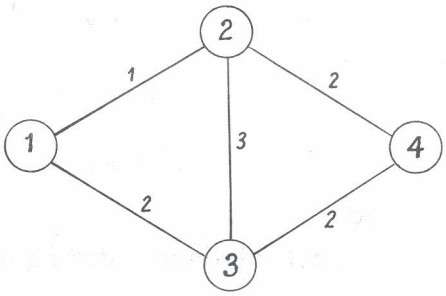
- Độ phức tạp tính toán của thuật toán (2) nhỏ hơn độ phức tạp tính toán của thuật toán Shier (vì $m \ll n$). Mặc dù độ phức tạp tính toán của thuật toán (2) lớn hơn độ phức tạp tính toán của thuật toán (1) song thuật toán (2) lại tìm thêm được các đường đi mà (1) không chỉ ra.

- Các đường đi "tối ưu" không phải là tuyệt đối mà phụ thuộc vào thời điểm nên cơ chế chọn đường đi sẽ mềm dẻo tùy thuộc vào tình trạng hoạt động của nút mạng và chi phí các kết nối. Chẳng hạn khi có một kết nối giữa hai nút bị gián đoạn, hoặc có một số nút mạng bị hỏng thì một số kết nối sẽ có chi phí là $+\infty$. Khi đó đường đi trên mạng sẽ được tự động thay đổi nhờ vào các qui tắc chọn đường tối ưu nêu trên để các sự cố không làm ảnh hưởng đến hoạt động của mạng.

Kết quả cài đặt thử nghiệm

Các thuật toán (1), (2) đã được cài đặt thử nghiệm bằng ngôn ngữ lập trình

C. Sau đây là kết quả thử nghiệm các thuật toán (1), (2) và qui tắc chọn t nêu trong (R) trên hai mô hình mạng giả định.



Mô hình thử 1. Giả sử mạng có topo như hình vẽ và nút xuất phát là nút 1.

a) Thuật toán của Dijkstra cho kết quả là bộ nút (nút, chi phí, hướng) như sau:

(2, 1, 2), (3, 2, 3), (4, 3, 1)

có nghĩa là đi từ nút 1 đến nút 2 theo hướng qua nút 2 có chi phí thấp nhất là 1, đi từ nút 1 đến nút 4 theo hướng qua nút 1 chi phí thấp nhất là 3,... Khi đó chúng ta có thể dễ dàng thấy các đường đi tối ưu sau (với độ dài tương ứng): 1 - 2(1), 1 - 3(2), 1 - 2 - 4(2).

b) Thuật toán của Shier. Giả sử chúng ta tìm 3 đường đi tối ưu nhất từ nút 1 đến các nút trong mạng. Kết quả như sau:

Từ 1 đến 1: 1 - 1(0), 1 - 2 - 1(2), 1 - 3 - 1(4)

Từ 1 đến 2: 1 - 2(1), 1 - 2 - 1 - 2(3), 1 - 3 - 2(5)

Từ 1 đến 3: 1 - 3(2), 1 - 2 - 3(4), 1 - 2 - 4 - 3(5)

Từ 1 đến 4: 1 - 2 - 4(3), 1 - 3 - 4(4), 1 - 2 - 1 - 2 - 4(5)

cho thấy thuật toán này chỉ ra các đường đi có các nút mạng xuất hiện nhiều lần.

c) Thuật toán (2): Tìm 2 đường đi tối ưu từ nút 1 đến các nút 2, 3, 4 (vì nút 1 chỉ có 2 nút lân cận là 2 và 3). Kết quả như sau:

Theo hướng qua nút 2: 1 - 2(1), 1 - 3(4), 1 - 4(3)

Theo hướng qua nút 3: 1 - 2(5), 1 - 3(2), 1 - 4(4)

Như vậy các đường đi tối ưu từ nút 1 đến các nút 2, 3, 4 theo hai hướng khác nhau qua hai nút 2 và 3 có chi phí là:

nút 2: 1, 5

nút 3: 2, 4

nút 4: 3, 4

So sánh với các kết quả nêu ở a), b) ta thấy:

- Thuật toán (2) tìm thêm được các đường mà (1) không chỉ ra.

- Để thuật toán của Shier cho ra kết quả như c), số lần lặp khi thực hiện thuật toán phải tăng lên mức là chúng ta có thể phải tìm 5, 6, 7... đường tối ưu nhất.

Sau khi đã tìm được tập các đường đi tối ưu bởi thuật toán (2), ta có thể áp dụng qui tắc (R) để mở rộng hoặc thu hẹp các hướng ra khỏi nút. Căn cứ theo bảng (T2), ta có:

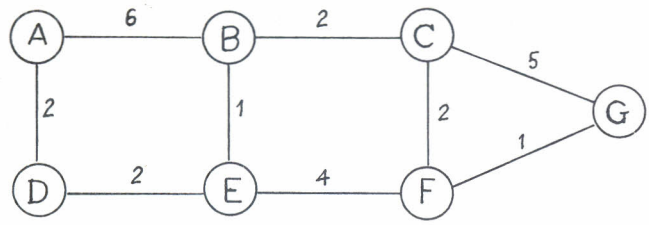
- Từ nút 1 đến nút 2 có 2 đường đi với chi phí $d_1 = 1$, $d_2 = 5$:

$\underset{\sim}{\mu}_D(d_1) = 1$, $\underset{\sim}{\mu}_D(d_2) = 0,201897$. Vậy với bất kỳ q nào thì cũng chỉ có d_1 được chọn.

- Từ 1 đến 3 có 2 đường đi với chi phí $d_1 = 2$, $d_2 = 4$: $\underset{\sim}{\mu}_D(d_1) = 1$, $\underset{\sim}{\mu}_D(d_2) = 0,493069$. Tương tự như trên chỉ có d_1 được chọn trong mọi trường hợp.

- Từ 1 đến 4 có 2 đường đi với độ dài $d_1 = 3$, $d_2 = 4$: $\underset{\sim}{\mu}_D(d_1) = 1$, $\underset{\sim}{\mu}_D(d_2) = 0,778801$. Khi mức độ đầy của hàng chờ từ 0 đến 0,2 thì chỉ có d_1 được chọn. Khi mức độ đầy của hàng chờ từ 0,3 đến 1,0 thì d_2 được chọn thêm và các gói tin từ 1 đến 4 sẽ ra khỏi 1 theo 2 hướng qua nút 2 và 3.

Mô hình thứ 2: Giả sử mạng có topo như hình vẽ và nút xuất phát là C.



Áp dụng thuật toán (2) chúng ta có kết quả:

Hướng B: C-A (7), C-B(2), C-D(5), C-E(3), C-F(7), C-C(8)
Hướng F: C-A (10), C-B(7), C-D(8), C-E(6), C-F(2), C-C(3)
Hướng G: C-A (14), C-B(11), C-D(12), C-E(10), C-F(6), C-C(5)

như vậy có 3 đường đi "tối ưu" đến mỗi đích trong mạng theo 3 hướng khác nhau xuất phát từ C (Vì C có 3 nút lân cận B, F, G). Theo công thức tính hàm $\underset{\sim}{\mu}_D(d)$ ta có:

A: $\underset{\sim}{\mu}_D(7) = 1$,	$\underset{\sim}{\mu}_D(10) = 0,689938$,	$\underset{\sim}{\mu}_D(14) = 0,266368$
B: $\underset{\sim}{\mu}_D(2) = 1$,	$\underset{\sim}{\mu}_D(7) = 0,361897$,	$\underset{\sim}{\mu}_D(11) = 0,08590$
D: $\underset{\sim}{\mu}_D(5) = 1$,	$\underset{\sim}{\mu}_D(8) = 0,648552$,	$\underset{\sim}{\mu}_D(12) = 0,213663$
E: $\underset{\sim}{\mu}_D(3) = 1$,	$\underset{\sim}{\mu}_D(6) = 0,594749$,	$\underset{\sim}{\mu}_D(10) = 0,156919$
F: $\underset{\sim}{\mu}_D(7) = 0,202464$,	$\underset{\sim}{\mu}_D(2) = 1$,	$\underset{\sim}{\mu}_D(6) = 0,318907$
G: $\underset{\sim}{\mu}_D(8) = 0,247202$,	$\underset{\sim}{\mu}_D(3) = 1$,	$\underset{\sim}{\mu}_D(5) = 0,702189$

Căn cứ theo bảng (T2) ta có: Với đích A: Khi mức độ đầy của hàng chờ từ 0,0 đến 0,4 ta chọn được 1 hướng ra khỏi C là hướng B. Khi mức độ đầy từ 0,5

đến 1,0 thì ta chọn thêm được hướng F. Hướng G không bao giờ được chọn. Dựa vào bảng (T2) và các kết quả tính toán trên chúng ta có thể làm công việc tương tự cho các đích B, D, E, F, G.

IV. KẾT LUẬN

Phương pháp lựa chọn đường đi "gần tối ưu" dựa trên mô hình mờ và phương pháp lập luận mờ cho phép chúng ta chọn thêm được các đường đi với chi phí gần bằng chi phí tối ưu, do đó tăng hiệu suất hoạt động của mạng và giảm nghẽn tắc, giải tỏa các hàng chờ ở nút mạng nhanh hơn. Các đường đi gần tối ưu được chọn bằng phương pháp mờ không phải tuyệt đối mà có nghĩa thời điểm nên đường đi trên mạng sẽ mềm dẻo tùy thuộc vào trạng thái hoạt động của nút. Do đó nếu các sự cố trên mạng như kết nối giữa hai nút bị hỏng, một nút nào đó có sự cố,... đường đi trên mạng sẽ được tự động thay đổi nhờ vào phương pháp chọn đường đi này. Một ưu điểm nữa phương pháp này là đưa ra cách quyết định đường đi thống nhất và đơn giản. Vì kết quả của phương pháp mờ phụ thuộc vào các yếu tố như cách lựa chọn các hàm đo độ thuộc, lựa chọn các toán tử suy diễn, lựa chọn luật kết hợp mờ... nên khi muốn có kết quả tốt hơn có thể thay đổi các yếu tố trên mà không phải thay đổi mô hình đã chọn.

TÀI LIỆU THAM KHẢO

1. A. Kaufmann. *Introduction to the theory of fuzzy subsets.* Academic Press Inc., 1975.
2. S. G. Tzafestas & A. N. Venetsanopoulos (ed.). *Fuzzy reasining in information, decsion and control system.* Kluwer Academic Publisher, 1994.
3. N. Honda, F. Sugimoto, M. Tanaka, S. Aida, *Decision support system using fuzzy reasoning and evaluation.* Artificial Intellgence in econmics and management (et. by L. F. Pau), Elsevier Science Pulisher B. V., 1986.
4. A. R. Bonde, S. Ghosh. *Comparative study of fuzzy versus "fixed" threshold for robust queue mangement in cell-swing networks.* IEEE/ACM Transactions on networking, Vol. 2, No. 4 (1994) 337-344.
5. W. Stalling, *Data computer communication, 3dr edition.* Macmillan Publ. Comp., 1991.
6. A. Tanenbaum, *Computer networks.* Prentice-Hall, 1989.
7. D. Bertsekas, R. Gallager, *Data betworks.* Prentice-Hall, 1989.
8. E. Minieka, *Optimization algprithms for network and graphs.* Marcel Dekker, Inc., 1978.
9. R. Perlman, *Iterconnections: Bridges & routers.* Addison-Wealey Publishing Company, Inc., 1992.
10. Z. Cao, A. Kandel, *Applicability of some fuzzy implication operators.* Fuzzy sets and systems, Vol. 31 (1989) 151-186.

Viện Công nghệ thông tin
Trung tâm KHTN và CNQG

Nhận bài ngày 18-9-1996