MỘT THUẬT TOÁN ĐỊNH TUYẾN CẢI THIỆN ĐỘ TRỄ TRONG MẠNG MANET SỬ DỤNG HỌC TĂNG CƯỜNG

Nguyễn Quốc Cường[1,2*], Lê Hữu Bình[1], Võ Thanh Tú[1]

[1] Khoa Công nghệ thông tin, Trường Đại học Khoa học, Đại học Huế

[2] Khoa Khoa học Tự nhiên và Công nghệ, Trường Đại học Tây Nguyên

*Email: nqcuong.dhkh23@hueuni.edu.vn

Ngày nhận bài: 19/01/2024; ngày hoàn thành phản biện: 20/01/2024; ngày duyệt đăng: 5/3/2024

TÓM TẮT

Ứng dụng của học tăng cường vào các giao thức định tuyến trong mạng MANET nhận được sự quan tâm của nhiều nhóm nghiên cứu trong thời gian gần đây. Đặc điểm chính của MANET là tính di động cao, dẫn đến tô-pô mạng thay đổi thường xuyên. Vì vậy, việc ứng dụng học tăng cường để tính toán bảng định tuyến tại mỗi nút sao cho hiệu quả nhất là một thách thức lớn. Bài báo này, chúng tôi đề xuất một thuật toán định tuyến sử dụng học tăng cường nhằm nâng cao hiệu năng mạng. Trong đó chúng tôi xây dựng hàm thưởng cho thuật toán Q-Learning có xem xét đến tải lưu lượng và chi phí khoảng cách đến nút đích để chọn tuyến đường ngắn nhất, tránh tắc nghẽn và giảm thiểu độ trễ. Kết quả mô phỏng cho thấy thuật toán đề xuất đã cải thiện được tỉ lệ gửi gói dữ liệu thành công, thông lượng mạng và độ trễ đầu-cuối so với thuật toán AODV.

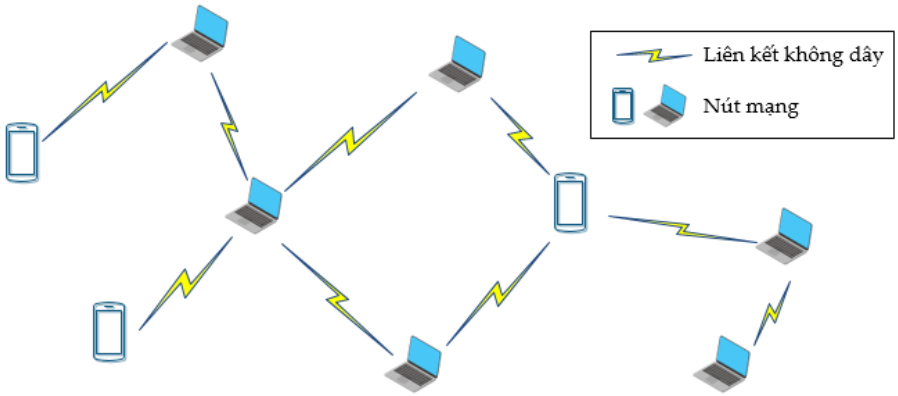
Từ khóa: AODV, giao thức định tuyến, hàng đợi, MANET, Q-Learning.

1. MỞ ĐẦU

MANET (Mobile Ad Hoc Network) [1] là mạng tùy biến di động, trong đó các nút mạng có thể di chuyển thay đổi vị trí theo thời gian, không phụ thuộc vào cơ sở hạ tầng cố định, trong vùng phát sóng các nút mạng vừa có vai trò gửi/nhận thông tin vừa có vai trò như là bộ định tuyến để chuyển tiếp thông tin đến nút đích (hình 1). Do những đặc tính này, nên MANET dễ dàng triển khai và ứng dụng trong nhiều lĩnh vực như: thành phố thông minh [2, 3], hệ thống giao thông thông minh [4], nông nghiệp thông minh [5], hệ sinh thái IoT [6, 7].

MANET với đặc điểm là tính di động cao, tô-pô thay đổi liên tục do đó bảng định tuyến tại mỗi nút phải được cập nhật thường xuyên và luôn lựa chọn được tuyến đường

ổn định trong suốt quá trình truyền dữ liệu. Vì vậy, một giao thức định tuyến hiệu quả là rất cần thiết cho MANET.



Hình 1. Mô hình mạng MANET

Gần đây, nhiều nhóm nghiên cứu đã áp dụng phương pháp học tăng cường (Reinforcement Learning - RL) để cải thiện các thuật toán định tuyến MANET [8–12]. Trong nghiên cứu [8], các tác giả đã đưa ra thuật toán định tuyến cho mạng MANET nhằm tăng tuổi thọ của mạng dựa trên AODV (Ad hoc On-Demand Distance Vector) có tên là EQ-AODV (Energy Q-learning AODV). Thuật toán này sử dụng Q-Learning để xây dựng phần thưởng dựa trên hai giá trị là tốc độ tiêu hao năng lượng và năng lượng còn lại. Giá trị phần thưởng sẽ được dùng để lựa chọn nút kế tiếp tốt nhất trong quá trình khám phá tuyến. Kết quả mô phỏng cho thấy rằng giao thức EQ-AODV vượt trội so với các giao thức khác về tỷ lệ phân phối gói và mức tiêu thụ năng lượng.

Trong nghiên cứu [9], các tác giả đã xây dựng thuật toán mới có tên là AQ-Routing (Adaptive Routing) dựa trên học tăng cường cho MANET. Sử dụng độ đo về mức độ di động của nút (Mobility Factor) và số lần truyền dự kiến (Expected Transmission count) làm phần thưởng cho thuật toán Q-Learning. Kết quả mô phỏng cho thấy AQ-Routing đã cải thiện được thông lượng và chọn được tuyến đường ổn định hơn.

Trong [10], các tác giả đã đưa ra một thuật toán định tuyến đảm bảo QoS sử dụng RL có tên là QGIR (QoS-guaranteed Intelligent Routing). Trong đó xác định độ đo xác suất phân phối gói (Packet Delivery Probability - PDP) từ nguồn đến đích là phần thưởng cho thuật toán Q-Learning để chọn đường đi sao cho tỷ lệ gửi gói tin là cao nhất. Đồng thời, hệ số tỷ lệ học tập được thay đổi linh hoạt theo ràng buộc của độ trễ đầu-cuối. Tại mỗi nút có cơ chế tự học thông tin định tuyến theo từng chu kỳ cập nhật tải lưu lượng. Kết quả mô phỏng cho thấy thuật toán QGIR đã cải thiện đáng kể hiệu suất mạng với thuật toán định tuyến khác.

Trong công trình [11], nhóm tác giả đã đưa ra thuật toán định tuyến sử dụng RL cho mạng MANET dựa trên AODV. Để cung cấp thông tin trạng thái mạng cho quá trình

khám phá tuyến đường, mỗi nút duy trì cơ sở dữ liệu thông tin trạng thái bao gồm hai giá trị là tải lưu lượng tối đa và năng lượng tiêu thụ của các nút. Mỗi nút sử dụng RL để cập nhật cơ sở dữ liệu về hai giá trị trên. Trong giai đoạn khám phá tuyến đường, nút nhận được gói RREQ sẽ chuyển tiếp nếu nó đủ năng lượng và ít tải lưu lượng. Các kết quả mô phỏng chứng minh rằng thuật toán được đề xuất đạt hiệu quả cao về mặt tiêu thụ năng lượng và chi phí mạng.

Trong công trình [12], các tác giả đã xây dựng thuật toán định tuyến có tên là RLI-AODV (Reinforcement Learning-based Improved AODV) sử dụng RL cho mạng MANET-5G. Trong đó thiết kế phần thưởng cho thuật toán Q-Learning là hai độ đo: tải lưu lượng và tỉ lệ tín hiệu trên nhiễu (Signal to Noise Ratio - SNR) tại các nút trung gian dọc tuyến đường tới nút đích. Mỗi nút sử dụng RL để cập nhật cơ sở dữ liệu về hai độ đo trên, khi khám phá tuyến đường mới, thuật toán định tuyến tham khảo cơ sở dữ liệu này để tìm lộ trình đảm bảo QoS. Các kết quả mô phỏng cho thấy RLI-AODV đạt hiệu quả về mặt thông lượng, độ trễ đầu-cuối và SNR.

Việc sử dụng RL cho định tuyến trong mạng MANET đã mang lại hiệu quả cao, thể hiện khả năng thích ứng với các mô hình mạng có tính di động cao và không thể đoán trước kịch bản. Tuy nhiên khi hình thành được các tuyến đường tối ưu thì nhiều nút sẽ ưu tiên lựa chọn tuyến đường này để gửi dữ liệu. Điều này dẫn đến sự khai thác tuyến đường tối ưu quá nhiều có thể gây ra tắc nghẽn, tiêu thụ năng lượng nhiều hơn, làm giảm hiệu suất mạng.Do đó, trong nghiên cứu này, chúng tôi sử dụng RL cho định tuyến MANET với những đóng góp mới như sau:

(i) Đề xuất hàm thưởng cho thuật toán Q-Learning dựa trên hai giá trị là tải lưu lượng và chi phí khoảng cách tuyến đường từ nguồn tới đích trong MANET.

(ii) Đề xuất thuật toán định tuyến thông minh sử dụng RL với tiêu chí chọn tuyến là ít chặng và ít tải lưu lượng.

Các phần tiếp theo của bài báo được bố cục như sau: Phần 2 trình bày cơ bản về phương pháp học tăng cường, phần 3 trình bày thuật toán định tuyến đề xuất của chúng tôi. Phần 4 là một số kết quả đánh giá bằng mô phỏng trên NS2. Cuối cùng, phần 5 là kết luận và đề xuất các hướng nghiên cứu tiếp theo.

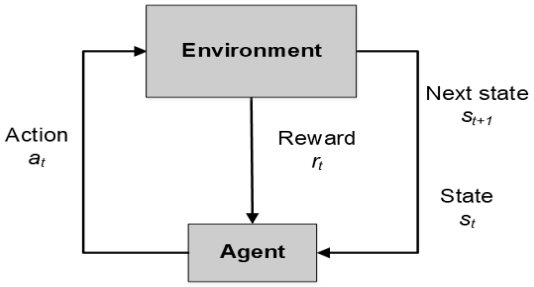
2. CƠ BẢN VỀ HỌC TĂNG CƯỜNG

Học tăng cường [13] là một lĩnh vực con của học máy, với nguyên lý cơ bản là hệ thống học từ các hành động trước đó của nó để chọn hành động tốt hơn trong tương lai. Bản chất của RL là thử và sai, nghĩa là lặp lại các thử nghiệm và học hỏi từ mỗi thử nghiệm. Hình 2 mô tả nguyên lý hoạt động của RL, trong đó một tác tử (agent) đóng vai trò là người học. Tại mỗi thời điểm *t*, tác tử này tương tác với môi trường bằng hành

động a_t. Với hành động này, môi trường chuyển từ trạng thái s_t thành trạng thái s_{t+1}, đồng thời tác tử nhận được một giá trị phản hồi (reward) r_t. Ở các lần học tiếp theo, dựa trên các giá trị phản hồi thu được ở các lần học trước, tác tử lựa chọn hành động sao cho giá trị phản hồi thu được là tốt nhất. Tổng giá trị giá trị phản hồi nhận được khi thực hiện hành động a_t tại trạng thái s_t là $Q(s_t, a_t)$, thường được xác định theo thuật toán Q-Learning như sau:

$$Q(s_t, a_t) = (1-\alpha) \times Q(s_t, a_t) + \alpha \times [R(s_t, a_t) + \gamma \times \max_{\forall a_{t+1}} Q(s_{t+1}, a_{t+1})] \quad (1)$$

Với α và γ là tỉ lệ học và hệ số chiết khấu, α và $\gamma \in [0,1]$, $R(s_t, a_t)$ là phần thưởng nhận được tức thì khi thực hiện hành động a_t tại trạng thái s_t.



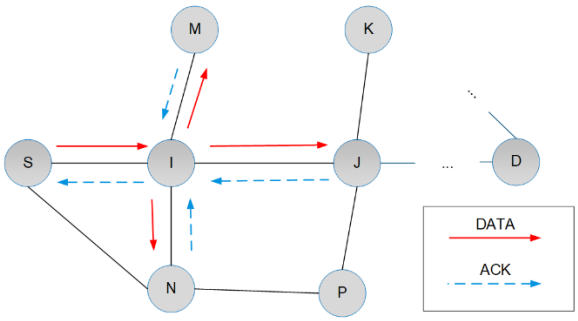
Hình 2. Minh họa nguyên lý của học tăng cường

3. THUẬT TOÁN ĐỊNH TUYẾN ĐỀ XUẤT

Trong phần này, trình bày thuật toán định tuyến được chúng tôi đề xuất cho MANET sử dụng RL. Trong đó xây dựng hàm thưởng cho thuật toán Q-Learning để chọn tuyến đường ngắn, tránh được các nút có hàng đợi đầy, hạn chế tỉ lệ rớt gói dữ liệu và giảm thiểu độ trễ đầu-cuối.

3.1. Mô hình hệ thống

Mô hình hệ thống của thuật toán đề xuất được minh họa trong hình 3, trong đó chúng tôi xem xét quá trình lựa chọn nút kế tiếp để truyền gói dữ liệu và cập nhật bảng định tuyến tại nút *I*. Trong trạng thái hiện tại, nút *I* có bốn láng giềng là *S, J, N* và *M*. Trong quá trình truyền dữ liệu, các nút đọc bảng định tuyến để lựa chọn nút kế tiếp, khi nút kế tiếp nhận được gói dữ liệu sẽ tạo gói phản hồi ACK, đính kèm thông tin phần thưởng vào rồi gửi cho nút gửi. Tại nút gửi khi nhận được gói phản hồi ACK sẽ tiến hành trích xuất các thông tin trên và dùng thuật toán Q-Learning để cập nhật bảng định tuyến.



Hình 3. Mô hình hệ thống của thuật toán đề xuất

3.2. Cập nhật Q-value trong thuật toán đề xuất

Sử dụng RL để định tuyến trong MANET, trong đó mỗi nút mạng đóng vai trò là tác tử sẽ tương tác với các nút láng giềng của nó để tìm hiểu đường đi tới nút đích. Tại mỗi thời điểm, nút tương tác với láng giềng và sẽ nhận được một giá trị phần thưởng. Giá trị phần thưởng của các hành động trước là cơ sở để nút lựa chọn hành động tiếp theo với mục tiêu dần dần thu được giá trị phần thưởng tốt nhất. Để thực hiện điều này, quá trình định tuyến được mô hình hóa như một bộ ba gồm {*trạng thái, hành động, phần thưởng*} [14]. Trong thuật toán đề xuất của chúng tôi, RL được dùng để thường xuyên cập nhật các độ đo định tuyến là tải lưu lượng và chi phí khoảng cách đến nút đích. Bộ ba {*trạng thái, hành động, phần thưởng*} được mô tả như sau:

3.2.1. Trạng thái

Mỗi trạng thái s được định nghĩa là một cặp $\{P, K\}$, trong đó $P = \{ \rho_1, \rho_2, \rho_3, \ldots, \rho_n \}$ đại diện cho tập hợp tải lưu lượng các nút mạng. Mỗi phần tử $\rho_i \in P$ đại diện cho tải lưu lượng tại nút I được xác định bởi số lượng gói dữ liệu trong hàng đợi tại nút đó. $K = \{ \kappa_1, \kappa_2, \kappa_3, \ldots, \kappa_n \}$, đại diện cho tập hợp giá trị chi phí khoảng cách đến nút đích.

3.2.2. Hành động

Mỗi hành động được xác định là một nút gửi gói dữ liệu tới nút kế tiếp để truyền tới đích và nhận được gói phản hồi ACK từ nút kế tiếp đó. Các nút trao đổi thông tin với nhau thông qua gói ACK này. Thông tin phản hồi bao gồm tải lưu lượng và giá trị chi phí khoảng cách đến nút đích.

3.2.3. Phần thưởng

Khi một nút I thực hiện hành động chọn nút kế tiếp J để gửi gói dữ liệu, nếu nút J nhận thành công gói dữ liệu sẽ gửi gói phản hồi ACK cho I. Trong gói phản hồi ACK này bao gồm phần thưởng là tải lưu lượng lớn nhất ký hiệu là $R_\rho(i, j)$ và chi phí khoảng cách từ nút hiện tại tới nút đích ký hiệu là $R_\kappa(i, j)$.

Gọi $\rho_{max}(s,d)$ là tải lưu lượng lớn nhất qua các nút dọc tuyến đường từ nút nguồn *S* đến đích *D* ($r_{s,d}$) và được xác định bởi:

$$\rho_{max}(s,d) = \max_{\forall i \in r_{s,d}} (\rho_i) \quad (2)$$

Trong đó ρ_i là số lượng gói dữ liệu trong hàng đợi của nút *I*.

Theo (2), giá trị phần thưởng tải lưu lượng $R_\rho(i,j)$ được xác định bởi:

$$R_\rho(i,j) = \max_{\forall i \in r_{s,d}} (\rho_j, \rho_{max}(j,d)) \quad (3)$$

Và giá trị phần thưởng chi phí khoảng cách tới nút đích $R_\kappa(i,j)$ được tính:

$$R_\kappa(i,j) = \begin{cases} +r, & nếu\ J\ là\ nút\ đích \\ -r, & nếu\ J\ chỉ\ có\ 1\ láng\ giềng\ là\ I \\ 0, & trường\ hợp\ còn\ lại \end{cases} \quad (4)$$

Trong đó $r \in [1,100]$, với giá trị nhỏ này để giảm thiểu việc tăng kích thước gói ACK. Theo công thức tính thưởng trên, giá trị thưởng $R_\kappa(i,j)$ là lớn nhất (+*r*) nếu *J* là nút đích. Trường hợp *J* không phải là nút đích và chỉ có duy nhất một láng giềng là *I* (đây là trường hợp gặp hố định tuyến) thì giá trị thưởng (-*r*) ở đây được chọn mang tính "phạt" nút *I* cho hành động đã chọn nhầm nút *J* làm nút kế tiếp. Trường hợp còn lại, *J* là nút trung gian thì phần thưởng là 0.

Trong thuật toán đề xuất của chúng tôi, tại mỗi nút sử dụng thuật toán Q-Learning để thường xuyên cập nhật hai giá trị trên. Dựa trên công thức cơ bản của thuật toán Q-Learning, các *Q-value* của tuyến đường từ nút *I* đến nút đích *D* được cập nhật theo các công thức sau:

$$Q_\kappa(i,j,d) = (1-\alpha) \times Q_\kappa(i,j,d) + \alpha \times [R_\kappa(i,j) + \gamma \times \max_{\forall k \in L_j} Q_\kappa(j,k,d)] \quad (5)$$

$$Q_\rho(i,j,d) = (1-\alpha) \times Q_\rho(i,j,d) + \alpha \times [R_\rho(i,j) + \gamma \times \max_{\forall k \in L_j} Q_\rho(j,k,d)] \quad (6)$$

Với $Q_\kappa(i,j,d)$ và $Q_\rho(i,j,d)$ là *Q-value* cho chi phí khoảng cách đường đi và tải lưu lượng lớn nhất từ nút *I* đến nút đích *D* với *J* là nút kế tiếp (*I* → *J* → …→ *D*). $R_\rho(i,j)$ và $R_\kappa(i,j)$ xác định theo công thức (3) và (4). L_j là tập các nút láng giềng của nút *J*. α và γ là tỉ lệ học và hệ số chiết khấu, α và $\gamma \in [0,1]$.

3.3. Thuật toán đề xuất

Tại mỗi nút, để lựa chọn nút kế tiếp gửi gói dữ liệu dựa vào hai độ đo định tuyến là tải lưu lượng và chi phí khoảng cách đến nút đích. Một láng giềng sẽ được chọn là nút kế tiếp nếu thõa mãn 2 điều kiện: chi phí khoảng cách đến nút đích là lớn nhất và tải lưu lượng tương ứng nhỏ hơn giá trị ngưỡng hàng đợi (P_{th} = 35). Để xác định giá trị ngưỡng hàng đợi này, chúng tôi thực hiện một số kịch bản mô phỏng với các giá trị ngưỡng khác nhau. Kết quả thu được cho thấy P_{th} = 35 cho hiệu suất tốt nhất phù hợp với các kịch bản mô phỏng.

Thuật toán 1 hiển thị mã giả của việc khám phá tuyến đường mới từ nút nguồn *S* đến nút đích *D* và cập nhật các giá trị *Q-value*. Dòng (2) đến dòng (9) thực hiện việc chọn nút kế tiếp tốt nhất *J* để gửi gói dữ liệu. Tiếp đến khi nút *J* nhận được gói dữ liệu thì tìm và lưu các *Q-value* vào gói phản hồi ACK, gửi cho nút *I* (dòng 10-14). Khi nút *I* nhận được gói phản hồi ACK của *J*, trích xuất các *Q-value* để cập nhật bảng định tuyến (dòng 15-18).

Trong thuật toán 1, xác suất ε của hành động chọn nút kế tiếp theo phương pháp ε-greedy. Tỉ lệ học α, hệ số chiết khấu γ và ε được đặt cố định tương ứng là 0.8, 0.8 và 0.9. $Q_\kappa(i,j,d)$ và $Q_\rho(i,j,d)$ được tính theo công thức (5) và (6), sau đó cập nhật vào bảng định tuyến để phục vụ cho việc chọn nút kế tiếp tốt nhất các lần tiếp theo.

Thuật toán 1. Thuật toán định tuyến đề xuất

Input: Gói dữ liệu *P*; Nút hiện tại *I*; Tập các nút láng giềng của *I*

Output: Nút kế tiếp *J;* Bảng định tuyến của *I* được cập nhật các *Q-value*

1: **Begin**
2: **for** (*mỗi lần nút I truyền gói dữ liệu P đến đích D*) **do**
// Tại nút I
3: **if** (*random(1)* < ε) **then** // *đọc bảng định tuyến của I để tìm nút kế tiếp*
4: **if** (*argmax* $Q_\kappa(i,j,d)$ and $Q_\rho(i,j,d) < P_{th}$) **then**
5: Chọn *J* là nút kế tiếp để gửi gói dữ liệu;
6: **end if**
7: **else**
8: Chọn ngẫu nhiên nút kế tiếp *J* để gửi gói dữ liệu;
9: **end if**
10: Gửi gói dữ liệu cho *J*;
11: **if** (*J nhận được gói dữ liệu từ I*) **then**
// Tại nút J
12: Tạo gói ACK để phản hồi cho *I*;
13: Tìm max $Q_\kappa(j,k,d)$ và $Q_\rho(j,k,d)$ tương ứng của nút *J*, lưu vào trong gói ACK;
14: Gửi gói ACK cho *I*;
15: **if** (*I nhận được gói ACK từ J*) **then**

// Tại nút I

16: Đọc $Q_\kappa(j,k,d)$ và $Q_\rho(j,k,d)$ trong gói ACK;

17: Cập nhật giá trị $Q_\kappa(i,j,d)$ và $Q_\rho(i,j,d)$ trong bảng định tuyến theo công thức (5) và (6);

18: **end if**

19: **end if**

20: **end for**

21: **End**

4. ĐÁNH GIÁ KẾT QUẢ BẰNG MÔ PHỎNG

Để đánh giá hiệu quả của thuật toán đề xuất, sử dụng phương pháp mô phỏng, với hệ mô phỏng NS2 phiên bản 2.35 [15]. Thuật toán đề xuất được so sánh với thuật toán định tuyến cơ bản của MANET là AODV về tỉ lệ gửi gói dữ liệu thành công (PDR: Packet Delivery Ratio), thông lượng mạng (Throughput) và thời gian trễ đầu-cuối (EED: End-to-End Delay). Với các thông số mô phỏng được trình bày như trong bảng 1.

Bài báo sử dụng công cụ *./setdest* của NS2 để tạo ra các nút mạng có vị trí ban đầu và di chuyển ngẫu nhiên theo mô hình Random Waypoint. Sử dụng công cụ *cbrgen* của NS2 để tạo 20 cặp kết nối trao đổi dữ liệu với giao thức truyền tin là UDP. Các mô phỏng được thực hiện trong thời gian 600 s đủ lớn để các nút gửi/nhận số gói tin đáng kể và các nút di chuyển được nhiều vị trí. Các kịch bản mô phỏng được chạy với số lần là 10 và các kết quả số liệu trong bài báo này là giá trị trung bình của 10 lần chạy.

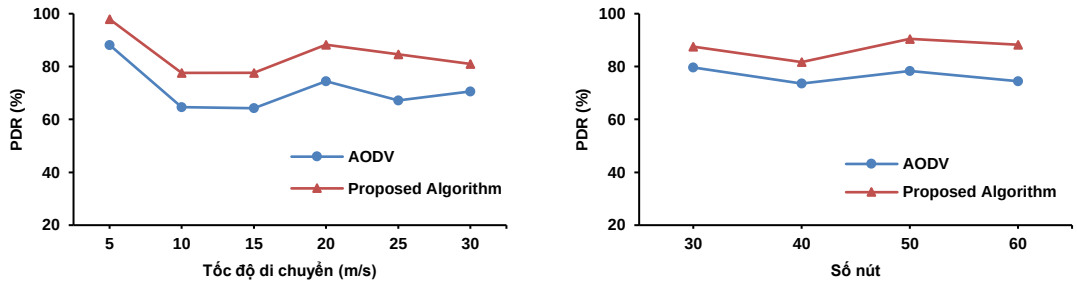
Bảng 1. Tham số thiết lập mô phỏng

Thông số	**Giá trị**
Giao thức định tuyến	AODV, Proposed Algorithm
Phạm vi	1000 m x 1000 m
Giao thức MAC	802.11
Số nút mạng	30, 40, 50, 60
Mô hình di động	Random Waypoint
Vùng thu phát sóng	250 m
Thời gian mô phỏng	600 s
Tốc độ di chuyển	5-30 m/s

Loại lưu lượng	CBR
Giao thức truyền tin	UDP
Kích thước gói tin	512 bytes
Độ dài hàng đợi tối đa	50 gói
Ngưỡng hàng đợi (P_{th})	35 gói

4.1. Tỉ lệ gửi gói dữ liệu thành công (PDR)

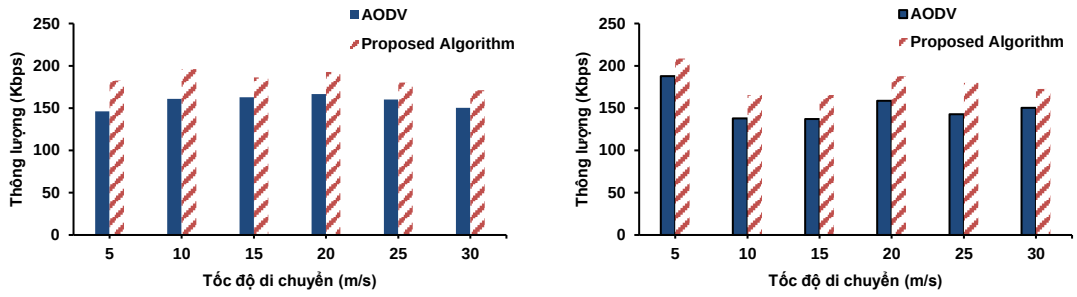
Hình 4 (a) thể hiện tỉ lệ gửi gói dữ liệu thành công với tốc độ di chuyển trung bình của các nút từ 5-30 m/s trong trường hợp mạng có 60 nút, PDR trung bình của thuật toán đề xuất là 84,496% của AODV là 71,538%. Trong trường hợp, tăng số nút mạng từ 30 lên 40, 50, 60 (hình 4 (b)) các nút di chuyển với tốc độ trung bình 20 m/s, thuật toán đề xuất cũng cho kết quả PDR cao hơn so với AODV trung bình là 10,468%. Như vậy tỉ lệ gửi gói dữ liệu thành công thì thuật toán đề xuất luôn cho kết quả tốt hơn so với thuật toán AODV. Kết quả tốt này là do thuật toán đề xuất có cơ chế chọn nút kế tiếp có số gói dữ liệu trong hàng đợi nhỏ hơn ngưỡng hàng đợi P_{th} dẫn đến tuyến đường tránh được các nút có hàng đợi đầy, giảm thiểu số gói dữ liệu bị rơi và giảm được tình trạng tắc nghẽn.



(a) (b)

Hình 4. So sánh tỉ lệ gửi gói dữ liệu thành công trong các trường hợp (a) tốc độ di chuyển khác nhau và (b) số nút mạng khác nhau

4.2. Thông lượng

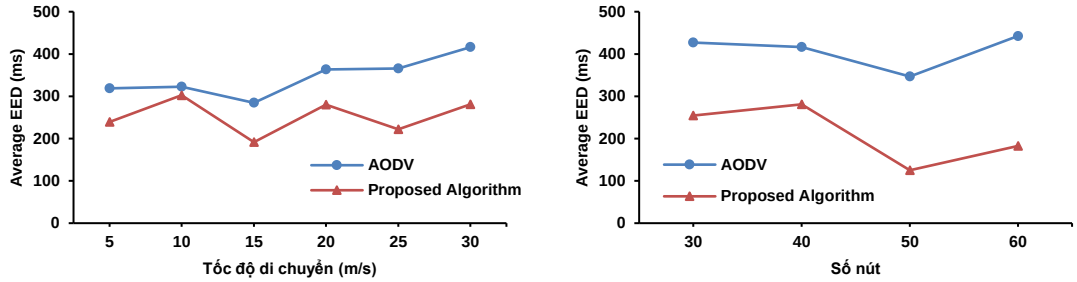


(a) (b)

Hình 5. So sánh thông lượng với các tốc độ di chuyển khác nhau (a) 50 nút và (b) 60 nút

Hình 5 thể hiện thông lượng trung bình của 2 thuật toán trong kịch bản mô phỏng với tốc độ di chuyển trung bình của các nút từ 5-30 m/s. Thuật toán đề xuất luôn cho thông lượng trung bình cao hơn so với AODV. Với số nút mạng là 50 (hình 5 (a)), thuật toán đề xuất có thông lượng trung bình là 184,851 Kbps cao hơn so với AODV là 157,875 Kbps. Khi tăng số nút mạng lên 60 (hình 5 (b)), thuật toán đề xuất có thông lượng trung bình cao hơn AODV là 27,688 Kbps.

4.3. Độ trễ trung bình đầu-cuối (EED)



(a) (b)

Hình 6. So sánh độ trễ trung bình đầu-cuối trong các trường hợp (a) tốc độ di chuyển khác nhau và (b) số nút mạng khác nhau

Cuối cùng, chúng tôi phân tích thời gian trễ trung bình đầu-cuối của 2 giao thức. Xét trường hợp mạng có 40 nút, di chuyển với tốc độ trung bình từ 5-30 m/s (hình 6 (a)) cho thấy EED trung bình của thuật toán đề xuất là thấp hơn với mức 252,669 ms so với AODV là 345,16 ms. Ở hình 6 (b), khi tăng số nút mạng từ 30 lên 40, 50, 60 với tốc độ di chuyển của các nút là 30 m/s thì EED của thuật toán đề xuất thấp hơn trung bình 197,305 ms so với AODV. Như vậy EED của thuật toán đề xuất tốt hơn so với AODV. Điều này do thuật toán đề xuất thích ứng nhanh với sự kiện mất liên kết của tuyến đường, khi phát hiện sự kiện ngay lập tức chọn được nút kế tiếp thay thế, không cần phải chờ khám

phá lại tuyến đường như AODV. Mặt khác thuật toán đề xuất xem xét tải lưu lượng ở các nút trung gian trong quá trình chọn tuyến đường, do đó làm giảm thời gian chờ tại hàng đợi của các nút này, dẫn đến giảm EED.

5. KẾT LUẬN

Ứng dụng học tăng cường cho định tuyến mạng tùy biến di động đã thu hút được nhiều nhóm nghiên cứu trong thời gian gần đây. Các phương pháp RL khác nhau có thể được áp dụng một cách hiệu quả để cải thiện các giao thức định tuyến cho MANET. Trong bài báo này, chúng tôi đã đề xuất thuật toán định tuyến dựa trên RL cho MANET. Với ý tưởng là sử dụng *Q-value* trong công thức cơ bản của RL để lưu trữ tải lưu lượng và chi phí khoảng cách đến nút đích, được dùng cho mục đích lựa chọn tuyến đường. Kết quả mô phỏng cho thấy thuật toán đề xuất đạt hiệu suất cao hơn về mặt tỉ lệ gửi gói dữ liệu thành công, thông lượng mạng và độ trễ đầu-cuối so với thuật toán AODV.

Trong thời gian tới, chúng tôi tiếp tục nghiên cứu ứng dụng học tăng cường cho định tuyến MANET, kết hợp các tham số định tuyến phù hợp để làm phần thưởng cho việc học.

TÀI LIỆU THAM KHẢO

[1]. Hoebeke J., Moerman I., Dhoedt B., Demeester P. (2004). An Overview of Mobile Ad Hoc Networks: Applications and Challenges, *Journal of the Communications Network*, vol. 3(3), pp. 60–66.

[2]. K.Q. Vu, V.K. Solanki, A.N. Le (2022). A saving energy MANET routing protocol in 5G, in: S. Velliangiri, M. Gunasekaran, P. Karthikeyan (Eds.), Secure Communication for 5G and IoT Networks, Springer International Publishing, Cham, pp. 213–220.

[3]. A. Kirimtat, O. Krejcar, A. Kertesz and M. F. Tasgetiren (2020). Future Trends and Current State of Smart City Concepts: A Survey, *IEEE Access*, 8, pp. 86448-86467. https://doi.org/10.1109/access.2020.2992441.

[4]. F. Zhu, Y. Lv, Y. Chen, X. Wang, G. Xiong and F. -Y. Wang (2020). Parallel Transportation Systems: Toward IoT-Enabled Smart Urban Traffic Control and Management, *IEEE Transactions on Intelligent Transportation Systems*, 21(10), pp. 4063-4071. https://doi.org/10.1109/tits.2019.2934991.

[5]. M. S. Farooq, S. Riaz, A. Abid, K. Abid and M. A. Naeem (2019). A Survey on the Role of IoT in Agriculture for the Implementation of Smart Farming. *IEEE Access*, 7, pp. 156237-156271. https://doi.org/10.1109/access.2019.2949703.

[6]. S. O. Olatinwo et al. (2019). Enabling Communication Networks for Water Quality Monitoring Applications: A Survey, *IEEE Access*, 7, pp. 100332-100362. https://doi.org/10.1109/ access.2019.2904945.

[7]. J. Xu et al. (2020). Design of Smart Unstaffed Retail Shop Based on IoT and Artificial Intelligence, *IEEE Access*, 8, pp. 147728-147737. https://doi.org/10.1109/access.2019.2904945.

[8]. R. Mili, S. Chikhi (2019). Reinforcement learning based routing protocols analysis for mobile ad-hoc networks, in: E. Renault, P. Mühlethaler, S. Boumerdassi (Eds.), Machine Learning for Networking, Springer, pp. 247–256.

[9]. Serhani, Abdellatif, Najib Naja, and Abdellah Jamali (2020). AQ-Routing: mobility-, stability-aware adaptive routing protocol for data routing in MANET–IoT systems, *Cluster Computing*, 23.1, pp. 13-27.

[10]. Duong, T. V. T., Binh, L. H. and Ngo, V. M. (2022). Reinforcement learning for QoS-guaranteed intelligent routing in Wireless Mesh Networks with heavy traffic load, *ICT Express*, 8(1), pp. 18-24.

[11]. Vishwanathrao , B. A. ., & Vikhar , P. A. . (2023). Reinforcement Machine Learning-based Improved Protocol for Energy Efficiency on Mobile Ad-Hoc Networks, *IJISAE*, *12*(8s), pp. 654–670. Website: https://ijisae.org/index.php/IJISAE/article/view/4259

[12]. L.H. Binh and T.-V.T. Duong (2023). An improved method of AODV routing protocol using reinforcement learning for ensuring QoS in 5G-based mobile ad-hoc networks, *ICT Express*, https://doi.org/10.1016/j.icte.2023.07.002.

[13]. Richard S. Sutton and Andrew G. Barto (2018), Reinforcement Learning: An Introduction, The MIT Press Cambridge, Massachusetts London, England.

[14]. H.A.A. Al-Rawi, M.A. Ng., K.-L.A. Yau (2015). Application of reinforcement learning to routing in distributed wireless networks: A review, *Artif Intell Rev*, 43, pp. 381–416.

[15]. DARPA. The Network simulator ns-allinone 2.35. [Online]. Website: http://www.isi.edu/nsnam/ns

A ROUTING ALGORITHM TO IMPROVE DELAY IN MANET USING REINFORCEMENT LEARNING

Nguyen Quoc Cuong[1,2*], Le Huu Binh[1], Vo Thanh Tu[1]

[1]Faculty of Information Technology, University of Sciences, Hue University

[2]Faculty of Natural Science and Technology, Tay Nguyen University

*Email: nqcuong.dhkh23@hueuni.edu.vn

ABSTRACT

Several research groups have recently interested in applying reinforcement learning to MANET routing systems. High mobility is an attribute of MANETs, which frequently results in changes to the network topology. It is therefore highly challenging to apply reinforcement learning to figure out the routing table at each node in the most efficient way. In this paper, we proposed a reinforcement learning-based routing algorithm to enhance network performance. Specifically, we built a reward function for the Q-Learning algorithm, which takes traffic load and the distance to the destination node into account to select the shortest route, prevent congestion, and minimize delay. Comparing the proposed approach to the AODV algorithm, simulation results demonstrate improvements in data packet delivery rates, network throughput, and end-to-end delay.

Keywords: AODV, routing protocol, queue, MANET, Q-Learning.

Nguyễn Quốc Cường sinh năm 1985 tại Nghệ An. Ông tốt nghiệp cử nhân ngành Công nghệ thông tin năm 2008 và thạc sĩ chuyên ngành Hệ thống thông tin năm 2012 tại Trường Đại học Sư phạm Hà Nội. Từ tháng 04 năm 2023 đến nay, ông là nghiên cứu sinh tại trường Đại học Khoa học, Đại học Huế. Ông công tác tại Khoa Khoa học Tự nhiên và Công nghệ, trường Đại học Tây Nguyên từ năm 2009.

Lĩnh vực nghiên cứu: Công nghệ mạng máy tính, ứng dụng của học máy và trí tuệ nhân tạo trong công nghệ mạng, mô phỏng và thiết kế mạng.

Lê Hữu Bình sinh năm 1978 tại Quảng Trị. Ông tốt nghiệp Kỹ sư ngành Điện tử Viễn thông năm 2001 tại Trường Đại học Bách khoa, Đại học Đà Nẵng và Thạc sĩ ngành Khoa học máy tính năm 2007 tại Trường Đại học Khoa học, Đại học Huế. Ông nhận học vị Tiến sĩ ngành Hệ thống thông tin năm 2020 tại Học viện Khoa học và Công nghệ, Viện Hàn lâm Khoa học và Công nghệ Việt Nam. Hiện nay, ông là giảng viên Khoa Công nghệ thông tin, Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: Công nghệ mạng thế hệ mới, ứng dụng của học máy và trí tuệ nhân tạo trong công nghệ mạng, mô phỏng và thiết kế mạng.

Võ Thanh Tú sinh năm 1965 tại Thừa Thiên Huế. Ông tốt nghiệp trường ĐH Tổng hợp Huế chuyên ngành Vật lý Điện tử năm 1987. Ông nhận bằng Thạc sỹ Công nghệ thông tin năm 1998 tại trường ĐH Bách khoa Hà Nội, nhận bằng Tiến sỹ tại Viện CNTT năm 2005, được phong chức danh Phó Giáo sư năm 2012. Hiện ông công tác tại Khoa Công nghệ thông tin, Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: Mạng truyền dẫn quang và không dây, đánh giá hiệu năng mạng, định tuyến và an toàn thông tin trên mạng.