**Survivable Mapping Algorithm by Ring Trimming (SMART)**

**for large IP-over-WDM networks.**

**Tổng quan**

*Chúng tôi phát triển một giải thuât nhanh và hiệu quả để tìm một ánh xạ chịu lỗi của topo IP trên mạng lưới các sợi quang trong mạng IP-over-WDM. Chúng tôi gọi nó là giải thuật SMART. Một số giải thuât khác giải quyết vấn đề này có thể được tìm thấy trong báo cáo. Do giải pháp ILP (Integer Linear Program – Quy hoạch tuyến tính nguyên) rất phức tạp, nhiều heuristic đã được đề xuất. Chúng thường bắt đầu với một số ánh xạ khởi tạo và sau đó sẽ từng bước cố gắng cải thiện nó. Điều này liên quan đến sự định giá của toàn bộ topo tại mỗi bước lặp, dẫn đến chi phí sẽ là rất cao đối với những mạng có topo lớn. Chúng tôi đề xuất một cách tiếp cận khác. Giải thuât SMART chia nhỏ bài toán ban đầu thành một tập các bài toán con đơn giản và độc lập với nhau. Sự kết hợp các giải pháp cho các bài toán con này tạo thành một ánh xạ chịu lỗi (chia để trị). Đây là lý do tại sao SMART nhanh hơn hàng chục lần so với các đề xuất khác, đặc biệt khi giải quyết những mạng có topo lớn. Chúng tôi cũng mở rộng giải thuật SMART để có được phương pháp ánh xạ mềm dẻo cho các trường hợp lỗi nhip (span failures) , lỗi nút (node failures) và lỗi liên kết đôi (double-link failures). Cuối cùng, chúng tôi cho thấy khả năng mở rộng của những phương pháp tiếp cận chuẩn heuristics là hạn chế (trái với SMART) khi áp dụng cho các lỗi liên kết đôi.*

1. **Giới thiệu**

Một vấn đề quan trọng trong mạng IP-over-WDM là khả năng vượt qua lỗi, hay là tính chịu lỗi. Nhìn chung, có hai cách tiếp cận để cung cấp khả năng chịu lỗi. **Protection** (bảo vệ) sử dụng những đường dự phòng đã được tính toán từ trước. Trong trường hợp xảy ra lỗi, các đường dự phòng được sử dụng để thay thế. **Restoration** (phục hồi), tự tìm những tuyến đường mới khi có lỗi xảy ra. Do đó, Protection ít hiệu quả hơn về mặt tài nguyên (các tài nguyên được dự phòng không cần biết sẽ có lỗi hay không), nhưng nhanh chóng. Trong khi Restoration hiệu quả hơn về tài nguyên thì lại chậm hơn. Các cơ chế Bảo vệ và Phục hồi có thể được cung cấp tại các lớp khác nhau. Những cơ chế tại lớp IP có thể xử lý những lỗi xảy ra tại cả 2 lớp (IP và WDM). Còn những cơ chế tại lớp WDM, chúng trong suốt với topo IP ( không xử lý được những lỗi xảy ra tại lớp IP) . Không rõ sự kết hợp giữa cơ chế và lớp nào là tốt nhất, mỗi cái đều có ưu và khuyết điểm riêng. Trong báo cáo này chúng ta chỉ xem xét cách tiếp cận Phục hồi tại lớp IP, bởi vì nó đã được chứng minh trong thực tế là một giải pháp hiệu quả và ít tốn kém (xem ví dụ: mạng Sprint [3]).

Mỗi liên kết IP được ánh xạ trên topo vật lý như một ***lightpath***. Thông thường, một sợi quang được sử dụng bởi nhiều hơn một lightpath (trong mạng Sprint số lượng tối đa là 25 [4]). Do đó, một lỗi vật lý xảy ra trên 1 sợi quang thường sẽ dẫn đến một số lỗi trên các liên kết IP. Trong cơ chế Phục hồi IP, những liên kết IP lỗi được phát hiện bởi thiết bị định tuyến IP (routers), và các tuyến đường thay thế trong topo IP được tìm thấy. Để kích hoạt, topo IP phải giữ được tính liên thông sau khi lỗi, điều này có thể được đảm bảo bởi một ánh xạ thích hợp của các liên kết IP trên topo vật lý. Chúng tôi gọi là một ánh xạ như vậy là 1 ánh xạ chịu lỗi.

Rõ ràng, không ánh xạ nào có thể chống lại bất cứ số lượng và loại lỗi vật lý. Do đó để 1 ánh xạ được gọi là ‘chịu lỗi’, chúng ta cần phải xác định những lỗi có thể gặp phải trong lớp WDM. Phổ biến nhất là lỗi liên kết vật lý đơn (*single physical link failure*). Nguyên nhân có thể là do một sợi quang ( liên kết vật lý) bị cắt, do lỗi của card giao diện đơn trong bộ chuyển đổi quang, hoặc do lỗi của bộ khuếch đại quang học. Nếu tính tới vị trí vật lý của các sợi quang, chúng ta sẽ mở rộng lỗi liên kết đơn sang lỗi nhịp đơn (*span failure*) [5]. Nhịp (span) là một tập tất cả các sợi quang nằm trong cùng một ống dẫn, giữa hai điểm truy cập liền kề ( giống như 1 trong 1 cống hoặc 1 vùng khuyếch đại). Để giảm chi phí, các liên kết vật lý khác nhau thỉnh thoảng được đặt trong cùng một nhịp (ví dụ, dọc theo đường sắt và đường điện), do đó 1 vết cắt duy nhất có thể hủy hoại toàn bộ chúng. Vậy nên, một lỗi nhịp có thể là lỗi của một liên kết vật lý hoặc là lỗi của đồng thời tất cả các liên kết vật lý đặt trong cùng một nhịp. Chúng ta cũng có thể gặp lỗi tại nút (*node failures*), chúng là kết quả của việc lỗi thiết bị tại các nút, như các thiết bị chuyển mạch. Cuối cùng, chúng tôi sẽ xét tới các lỗi liên kết đôi (*double-link failures*), những lỗi độc lập của hai liên kết vật lý bất kỳ. Thông thường, một tình huống như vậy xảy ra khi có lỗi thứ hai xuất hiện trước khi lỗi đầu tiên được sửa chữa. Đây không phải là trường hợp phổ biến nhưng vẫn có thể xảy ra. Ví dụ, trong mạng Sprint, khoảng thời gian giữa 2 lần lỗi quang học liên tiếp dao động từ 5,5 giây đến 7,5 ngày, trung bình là khoảng 12 giờ [4]. Hầu hết chúng được tự động sửa chữa trong vòng vài phút, nhưng có những lỗi cần đến sự can thiệp của con người (ví dụ, khi sợi quang bị cắt ) thì việc xử lý có thể mất đến hàng giờ hoặc hàng ngày. Một lỗi khác hoàn toàn có thể xảy ra trong khoảng thời gian đó.

Cách tiếp cận chúng tôi giới thiệu trong báo cáo này là thích hợp để đối phó với tất cả các loại lỗi đã được liệt kê ở trên.

* 1. **Công việc liên quan**

Các vấn đề về ánh xạ chịu lỗi không phải là mới, nó đã được được định nghĩa lần đầu trong [6]. Từ đó, đã có nhiều giải thuât giải quyết vấn đề NP-đầy đủ (với các biến thể khác nhau) đã được đề xuất. Nói chung, chúng có thể được chia thành hai nhóm: nhóm tìm kiếm tham lam dựa trên Quy hoạc tuyến tính nguyên (ILP) và các heuristic (chẩn đoán). Các giải pháp ILP có thể được tìm thấy trong [7, 8]. Tuy nhiên, cách tiếp cận này dẫn đến độ phức tạp của thuật toán sẽ cao đến mức không thể chấp nhận được cho các mạng có kích thước không tầm thường (mạng có nhiều hơn vài chục nút). Phương pháp thứ hai sử dụng các heuristic khác nhau, như Tabu Search [6, 10, 11, 8], mô phỏng luyện thép Simulated Annealing [12] và những heuristics khác [1, 13]. Hầu hết chúng bắt đầu với một số ánh xạ khởi tạo (ví dụ, đường đi ngắn nhất) và cố gắng cải thiện chúng tại những bước lặp tiếp theo. Vì vậy, độ phức tạp tại mỗi bước lặp bị chi phối bởi thời gian để đánh giá ánh xạ ứng viên của toàn bộ topo logic. Nó sẽ rất tốn kém cho những topo lớn. Chúng tôi sẽ chứng minh rằng, trong thực tế, các heuristic này chỉ áp dụng giới hạn được cho những topo có khoảng vài trăm nút, hoặc thậm chí ít hơn trong vài trường hợp lỗi cụ thể. Giải thuật chúng tôi đề xuất trong bài báo này mở ra một nhóm thứ ba mới. Nó dựa trên việc chia nhỏ bài toán ban đầu thành một tập bài toán con nhỏ hơn , độc lập và dễ giải quyết hơn. Mỗi bài toán con đó sẽ được giải quyết riêng rẽ. Sau đó các giải pháp này sẽ được kết hợp lại để có được một ánh xạ chịu lỗi cho toàn bộ topo. Điều này khiến SMART nhanh và có khả năng mở rộng. Một giải pháp nhanh tương tự đã được đề xuất trong [13]. Tuy nhiên, sau đó chúng tôi sẽ chỉ ra rằng nó không hiệu quả lắm cho các đồ thị lớn hơn.

Những tình huống lỗi phức tạp hơn tình huống lỗi liên kết đơn cũng đã được giải quyết từ trước, ví dụ như trong [5] (lỗi nhịp), trong[14] (lỗi nút) và [15, 16, 17] (lỗi liên kết đôi). Nhưng chúng chỉ tiếp cận tại lớp WDM sử dụng cơ chế phục hồi . Theo chúng tôi, đây là lần đầu tiên phục hồi IP được áp dụng cho các tình huống lỗi này.

Hầu hết các phương pháp đề cập ở trên có 1 tham số là số bước sóng trong mỗi sợi quang, tức là có tính đến dung lượng sợi quang . Rõ ràng, điều này phản ánh thực tế một cách trung thực hơn. Các phiên bản của SMART được trình bày trong bài báo này, cũng giống như các phương pháp tiếp cận trong [6, 7, 13], đều không xét đến các ràng buộc về dung lượng sợi quang. Chúng tôi tin rằng, SMART có khả năng thích nghi dễ dàng với ràng buộc về dung lượng. Đây là vấn đề chính mà chúng tôi sẽ nghiên cứu trong tương lai.

* 1. **Đóng góp**

Những đóng góp của báo cáo này:

• một cách tiếp cận mới cho việc xây dựng ánh xạ chịu lỗi của topo logic (hay topo IP) trên topo vật lý (WDM), đưa ra một giải thuât có tốc độ nhanh hơn gấp 100 đến 1000 lần so với các giải thuật đã được đề xuất trước đấy.

• một giải thuât có độ phức tạp nhỏ cho phép chạy trên những topo lớn (topo có hàng trăm đến hàng ngàn nút) trong khoảng thời gian hợp lý (trái với những giải thuật trước đây);

• hiệu quả trong nhiều trường hợp lỗi khác nhau, như lỗi nhịp, lỗi nút và lỗi liên kết đôi.

* 1. **Tổ chức của báo cáo**

Trong phần 2, chúng tôi giới thiệu các ký hiệu và phát biểu bài toán. Trong phần 3 chúng tôi mô tả giải thuât SMART cho trường hợp lỗi sợi đơn (*single-fiber failure*) và đưa ra một ví dụ cụ thể. Trong phần 4, chúng tôi trình bày khả năng thích nghi của giải thuật SMART khi lỗi xảy ra tại nhịp/ nút/ liên kết đôi. Trong phần 5 chúng tôi so sánh bằng cách mô phỏng giải thuât SMART với các giải thuât khác và chứng minh hiệu quả của giải thuật SMART đối với các topo lớn . Phần 6 là phần kết luận.

1. **Ký hiệu và phát biểu bài toán**

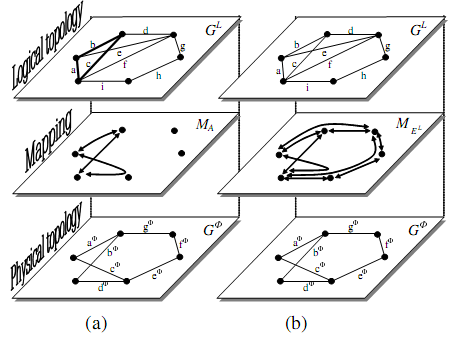
Topo vật lý và topo logic được biểu diễn bằng đồ thị vô hướng = (V, ) và = (V, ), tương ứng. Trong đó V là tập đỉnh, và là tập các cạnh vô hướng. Chú ý, để đơn giản chúng ta giả sử ≡ ≡ V. Hơn nữa, và là các đồ thị 2-cạnh liên thông. (Trong lý thuyết đồ thị, một đồ thị là k-cạnh liên thông nếu nó vẫn duy trì tính liên thông khi có ít hơn k cạnh của nó bị loại bỏ. [20]). Điều này ngụ ý rằng, trên cả hai mức (logic và vật lý), một lỗi liên kết sẽ không làm mất tính liên thông của đồ thị, rõ ràng đây là điều kiện cần thiết cho sự tồn tại của ánh xạ chịu lỗi.

Do chúng ta đã giả định rằng dung lượng các liên kết vật lý là không giới hạn, nên chúng ta không quan tâm các bước sóng được chỉ định trong các light paths và vị trí của các thiết bị chuyển đổi bước sóng. Do đó một light path chỉ là 1 loop-free path trong topo vật lý.

**Định nghĩa 1 (Ánh xạ):** *Cho là một tập tất cả các lightpaths có thể (đường dẫn vật lý) trong topo vật lý, và cho A ⊂ là một tập các liên kết logic. Một ánh xạ là 1 hàm*

*: A → liên kết mỗi logical link từ tâp A đến một lightpath tương ứng trong topo vật lý.*

Với các đối số từ bên ngoài A,  không xác định. Trong trường hợp đặc biệt A = ,  được xác định cho tất cả các liên kết trong topo logic. Hai ví dụ của ánh xạ được biểu diễn như trong hình.1.



**Hình1: Một ánh xạ của topo logic**  = (V, ) **trên topo vật lý**  = (V, ) **. Miền A được ánh xạ** A = {a,b,c} ⊂ **(in đậm) trong hình (a) và** A = **(tập tất cả các cạnh logic) trong hình (b). Trong cả 2 hình các lightpaths chỉ định cho các logical links** a, b **và** c **là** (a) = {,} , (b) = {} **và**  (c) = {} **tương ứng.**

**Định nghĩa 2 (Ánh xạ tách rời – Disjoint mapping**): *Một ánh xạ*  *, A ⊂ , là tách rời (disjoint) nếu trong ánh xạ (A) cho toàn miền A, mỗi liên kết vật lý được sử dụng tối đa một lần.*

Ánh xạ MA trong hình 1a là ánh xạ tách rời.

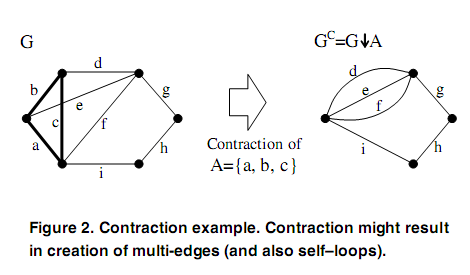
**Định nghĩa 3** (k-**survivability**): *Một ánh xạ của topo logic (tất cả các cạnh thuộc ) trên topo vật lý được gọi là là k-survivable nếu k liên kết vật lý trong Gφ bị lỗi đồng thời không làm mất tính liên thông của .*

Rất dễ để kiểm tra thấy trong hình.1b ánh xạ  là 1-survivable. Để làm mất đặc tính này chỉ cần thay lightpath tương ứng với logical link a từ (a) = {,} thành (a) = {, }}, khi đó nếu lỗi xảy ra tại aφ sẽ làm mất tính liên thông của topo logic .

Cho topo logic = (V, ), và topo vật lý = (V, ) mục tiêu của chúng ta là tìm ánh xạ 1-survivable  của trên . (Sau đó, chúng ta sẽ mở rộng sang các tình huống lỗi phức tạp hơn).

Một trong những thao tác chính trong giải thuật SMART là thao tác rút gọn (**contraction**), được định nghĩa như sau:

**Định nghĩa 4** (**Rút gọn** - **Contration** [20]): *Rút gọn một cạnh e ∈ E trong đồ thị G = (V, E) là xóa cạnh đó và sáp nhập các nút nối với nó (end-nodes) thành một. Đồ thị thu được là đồ thị rút gọn và được biểu diễn bởi = G ↓ e. Tương tự, rút gọn 1 tập cạnh A ⊂ E tạo thành đồ thị rút gọn = G↓A (xem hình 2). Chú ý rằng thứ tự các cạnh trong A không ảnh hưởng đến kết quả.*



1. **Giải thuật SMART**

Ý tưởng của giải thuật SMART như sau. Trước tiên lựa chọn từ topo logic một chu trình (ring) C ⊂ và ánh xạ của nó là tách rời. Việc có ánh xạ tách rời của chu trình logic đảm bảo nó sẽ duy trì tính liên thông khi có 1 sợi quang đơn nào đó bị lỗi. Nói cách khác, chu trình C sẵn sàng được ánh xạ bằng 1 cách 1-survivable. Bây giờ, chúng ta phải chú ý tới phần còn lại của đồ thị logic \ C và một kết nối chịu lỗi của \ C với C. Do đó chúng ta rút gọn chu trình C trong topo logic (ta ‘trim’ nó) và lặp lại các quá trình trên cho đồ thị kết quả ↓ C. Chúng ta lặp lại quá trình này cho đến khi topo logic rút gọn hội tụ về một nút duy nhất, điều này đảm bảo tính chịu lỗi. Một ví dụ của giải thuật SMART được minh họa trong hình. 3.

* 1. **Giải thuật**

Giả code của giải thuât SMART như sau:

**Khởi tạo:** Topo logic rút gọn : = . Ánh xạ và miền A của nó là rỗng:

= ∅, A = ∅

**Bước 1:** Chọn một chu trình C trong . Ưu tiên những chu trình ngắn hơn, nhưng chúng phải là những cái không được xét tới từ bước lặp thành công cuối cùng (bỏ qua những chu trình đã xét) . IF không có chu trình nào được tìm thấy, THEN RETURN ánh xạ là không chịu lỗi. END

**Bước 2:** Ánh xạ tách rời chu trình C trên topo vật lý bằng cách sử dụng hàm **DisjointMap** (xem mục 3.2).

IF không tìm thấy ánh xạ tách rời, THEN bước lặp không thành công; GOTO Step1

IF tìm thấy ánh xạ tách rời, THEN bước lặp là thành công.Biểu diễn ánh xạ này bởi .

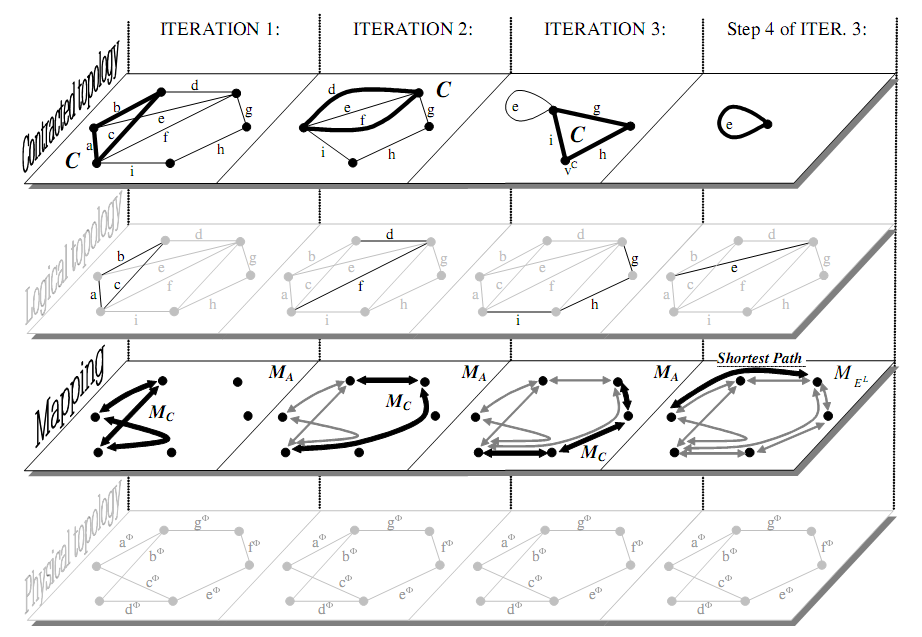
**Bước 3**: Mở rộng ánh xạ bởi , tức là, kết hợp những ánh xạ đó thu được 1 ánh xạ mới. Thao tác này xác định được do tập các logical links của A và C là tách rời.

**Bước 4:**  Rút gon C trong ( . := . ↓ C). IF chỉ gồm có 1 nút, THEN ánh xạ các logical link còn lại (self-loops trong GC) theo 1 cách nào đấy và RETURN ánh xạ chịu lỗi . END

**Bước 5:** GOTO Bước 1

**Chú ý:**

Lưu ý, ngay cả khi mỗi bước lặp của giải thuât này là không thành công, thuật toán vẫn không dừng lại cho đến kiểm tra hết tất cả các chu trình tồn tại trong đồ thị . Do tổng số lượng các chu trình tăng lên nhanh chóng cùng với kích thước của đồ thị, nên sẽ hợp lý khi hạn chế số lần lặp liên tiếp không thành công. Trong khai triển của chúng tôi, chúng tôi cho phép tối đa là 10 lần.



**Hình 3.** Minh họa của giải thuật SMART. Chúng ta có bốn lớp, từ dưới lên trên lần lượt là: topo vật lý , ánh xạ , topo logic , và topo logic rút gọn . Trong khi chạy giải thuât SMART, chỉ có topo logic rút gọn và ánh xạ thay đổi từ bước lặp này sang bước lặp khác. Các topo logic và topo vật lý nằm trong bối cảnh, do đó chúng có màu xám. Tại mỗi bước lặp, một chu trình C được chọn từ lớp topo logic rút gọn và được bôi đậm. Ví dụ tại bước lặp 2, chúng ta có C = {d, f}. Sau đó, một ánh xạ tách rời được tìm cho chu trình này, nó được bôi đậm. (Tại mỗi bước lặp, các lightpaths có màu xám là những lightpaths được tìm thấy ở tất cả các lần lặp trước.). Tiếp theo, chu trình C được rút gọn, tạo thành 1 topo logic rút gọn mới và được sử dụng trong bước lặp tiếp theo. Khi đã hội tụ về một nút duy nhất, ánh xạ về cơ bản đã là 1-survivable. Tuy nhiên, có thể có một số logical links còn lại chưa được ánh xạ, hình thành self-loops trong . Trong ví dụ này, e là một liên kết như vậy, chúng ta ánh xạ nó bằng 1 cách nào đấy (ở đây là là đường ngắn nhất). Bây giờ, chúng ta kết hợp các ánh xạ được tìm thấy trong mỗi lần lặp để có được ánh xạ (cột cuối cùng) của toàn bộ topo logic. Ánh xạ là 1-survivable.

* 1. **Hàm DisjointMap**

Trong bước 2 của giải thuật SMART, chúng tôi áp dụng một hàm tìm kiếm ánh xạ tách rời cho một tập các logical link, gọi là hàm DisjointMap. Vấn đề này tương đương với vấn đề tìm kiếm các đường dẫn cạnh tách rời (edge-disjoint paths) [21], đã được chứng minh là bài toán NP-đầy đủ. Do đó chúng tôi áp dụng một heuristic DisjointMap đơn giản như sau. Cho mỗi cạnh vật lý một trọng số (các trọng số này chỉ được sử dụng trong DisjointMap) và khởi tạo trọng số này bằng 1. Tại mỗi bước lặp, heuristic DisjointMap ánh xạ các logical link từ C với đường đi ngắn nhất. Nếu không có liên kết vật lý nào được sử dụng quá một lần, giải pháp tách rời đã được tìm thấy. Nếu không, trọng số của mỗi liên kết vật lý được sử dụng nhiều hơn một lần sẽ được tăng lên, và một vòng lặp mới bắt đầu. Sau một số bước lặp thất bại, hàm DisjointMap thất bại.

* 1. **Độ phức tạp của SMART là đa thức**

Độ phức tạp của một bước lặp trong giải thuật SMART phụ thuộc vào hàm DisjointMap, nó sử dụng O(1) lần giải thuât Dijkstra để tìm đường đi ngắn nhất. Do đó độ phức tạp của một bước lặp trong giải thuật SMART bằng độ phức tạp của giải thuât Dijkstra, tức là O(Dijkstra).

Bây giờ chúng ta thử ước tính số bước lặp cần thiết để SMART hội tụ đến một giải pháp chịu lỗi. Chúng ta cần ánh xạ O(N) cạnh. Một bước lặp thành công ánh xạ một chu trình ngắn, độ dài là hàm O(1). Vì vậy, chúng ta cần O(N) bước lặp thành công. Giữa hai lần lặp thành công bất kỳ, chúng ta có thể có một số bước lặp thất bai. Như đã giải thích ở Chú ý trong phần 3.1, khi triển khai thực tế, chúng ta giữ con số này là không đổi, tức là, hàm của O (1). Do đó, tổng số lần lặp là hàm O(N).

Độ phức tạp của SMART bằng độ phức tạp của một bước lặp nhân với số lần lặp = O (Dijkstra).O (N). Do trong trường hợp xấu nhất O (Dijkstra) = O(), nên trong trường hợp xấu nhất độ phức tạp của giải thuât SMART là O(), Tuy nhiên, với trường hợp trung bình thì đơn giản hơn. Trong mô phỏng cho topo lớn (xem Phần 5), chúng tôi quan sát thấy độ phức tạp của giải thuật bằng O(),

1. **Phù hợp với nhiều tình huống lỗi khác nhau**

Trong phần này chúng tôi đưa ra các phiên bản mở rộng đơn giản của phiên bản tiêu chuẩn SMART, để giải quyết những tình huống lỗi phức tạp hơn so với trường hợp lỗi xảy ra tại 1 liên kết vật lý đơn. Ý tưởng chính vẫn vậy, tại mỗi lần lặp, chúng ta tìm một ánh xạ chịu lỗi của đồ thị con trong topo logic rút gọn rồi sau đó rút gọn đồ thị con này thành một nút.

* 1. **Lỗi nhịp (Span failures)**

Một ánh xạ được gọi là *span-survivable* nếu nó vẫn đảm bảo tính liên thông của đồ thị logic sau khi có lỗi ở bất kỳ sợi đơn nào hoặc sau khi 1 trong những *multi-link spans* (nhịp đa liên kết) bị cắt. Giải thuật SMART chỉ đòi hỏi một bản nâng cấp nhỏ của hàm DisjointMap để giải quyết trường hợp lỗi nhịp. Giả sử có 1 topo vật lý như trong hình 3. Sợi và được đặt trong cùng một nhịp. Để ánh xạ của chu trình đảm bảo tính chịu lỗi, hàm DisjointMap có thể xem các sợi này chỉ như là một lightpath (tức là gom , lại thành 1 lightpath ). Ví dụ, tại bước lặp 3, ánh xạ hiện thời của chu trình C = {g, h, i} không được chấp nhận, vì việc cắt nhịp - sẽ tách đỉnh khỏi đồ thị. Nhưng nếu ta thay đổi cách chọn lightpath của cạnh logic i từ (i) = {} thành (i) = {, ​​, } , khi đó chu trình C vẫn còn liên thông sau khi cắt đồng thời sơi và sợi . Phiên bản sửa đổi của SMART để xử lý lỗi nhịp được gọi là *SMART-Span*.

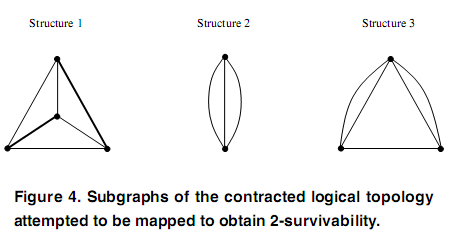
* 1. **Lỗi nút (Node failures)**

Rõ ràng, khi lỗi xảy ra tại một đỉnh v ∈ V, topo logic sẽ mất tính liên thông- do đỉnh v bị tách khỏi đồ thị (tức là không có đường đi từ đỉnh khác trong đồ thị đến nó nữa). Điều tốt nhất chúng ta có thể làm là giữ cho phần còn lại của topo logic liên thông*. Do đó, một ánh xạ được gọi là node-survivable nếu sau khi có lỗi xảy ra tại 1 đỉnh bất kỳ v ∈ V, phần topo logic còn lại \ {v} vẫn còn liên thông*. *Node-survivable* là bài toán khó giải quyết hơn so với *1-survivability*. Tuy nhiên, SMART giải quyết vấn đề này khá hiệu quả. Chúng ta chỉ cần sửa hàm DisjointMap để nó tìm ánh xạ nút tách rời (node-disjoint) thay vì tìm ánh xạ liên kết tách rời (link-disjoint) (Tức là nếu như trước đó để đảm báo tính tách rời thì không có link nào được dùng quá 1 lần, thì nay là không có nút nào được dùng quá 1 lần ). Khi đó, sẽ không nút đơn nào bị lỗi có thể làm mất tính liên thông của ánh xạ chu trình này. Phiên bản của giải thuật SMART để giải quyết lỗi nút này được gọi là *SMART-Node*.

* 1. **Lỗi liên kết đôi (Double-link failures)**

Để có một ánh xạ 2-survivable chúng ta sử dụng ý tưởng cơ bản giống như trước – tại mỗi bước lặp, chúng ta tìm 1 ánh xạ 2-survivable của các đồ thị con của topo logic rút gọn và sau đó rút gọn đồ thị con này thành một nút. Tuy nhiên, đồ thị con chúng ta cần tìm sẽ khác. Trước đây, trong trường hợp lỗi đơn, đồ thị con đơn giản nhất chúng ta có thể tìm thấy là chu trình, ánh xạ tách rời của chu trình này đảm bảo tính 1-survivability cho nó . Trong trường hợp lỗi liên kết đôi, các đồ thị con phải tìm sẽ phức tạp hơn vì it nhất chúng phải đảm bảo tính 3-cạnh liên thông ( 3-edge-connected) . Hình 4 biểu diễn ba cấu trúc đã được chúng tôi sử dụng. Chú ý rằng chỉ có hình đầu tiên là dạng đồ thị đơn giản không có đa cạnh (giữa 2 nút có nhiều hơn 1 cạnh). Trong thực tế, nó là đồ thị 3-cạnh liên thông đơn giản, nhỏ nhất có thể. Do topo logic cũng là một đồ thị đơn giản, chỉ có cấu trúc đầu tiên là có thể được tìm thấy tại bước lặp đầu của giải thuật .Tuy nhiên, tại các bước lặp tiếp theo chúng ta phải làm việc trên các topo logic rút gọn, chúng có thể là đa cạnh (multi-edges) và tự lặp (self-loop : cạnh có đỉnh đầu và đỉnh cuối trùng nhau). Khi đó các cấu trúc 2 và 3 sẽ trở nên hữu ích.

Phần mở rộng của giải thuật tìm kiếm ánh xạ 2-survivable được gọi là SMART-DF (DF - Double-link failures). Các mã giả của SMART-DF tương tự như của SMART đã được trình bày tại mục 3.1 với một ngoại lệ: trong Bước 1, thay vì tìm kiếm các chu trình, chúng ta tìm kiếm các cấu trúc được mô tả trong hình 4. Phần còn lại của mã này vẫn giữ nguyên. Đặc biệt, SMART-DF tìm kiếm một ánh xạ tách rời của các cấu trúc được tìm thấy trong Bước 1.Hàm DisjointMap được sử dụng như trong Mục 3.2. Lưu ý đây là một cách tiếp cận hơi hạn chế. Ví dụ các ánh xạ của cấu trúc 1 không nhất thiết phải là hoàn toàn tách rời - hai cạnh in đậm trong hình 4.1 có thể cùng sử dụng các liên kết vật lý, mà vẫn đảm bảo tính 2-survicability của toàn bộ cấu trúc 1 (giả định rằng các lightpaths còn lại là link-disjoint). Cặp cạnh tương tự có thể được tìm thấy trong cấu trúc 3. Với lý do như vậy, chúng ta không cần thiết phải mở rộng tập các cấu trúc cần tìm ở Bước 1, mà chỉ cần dùng các cấu trúc trong hình 4. Đó là những topo 3-cạnh liên thông thường gặp nhất trong các bước lặp của giải thuật.



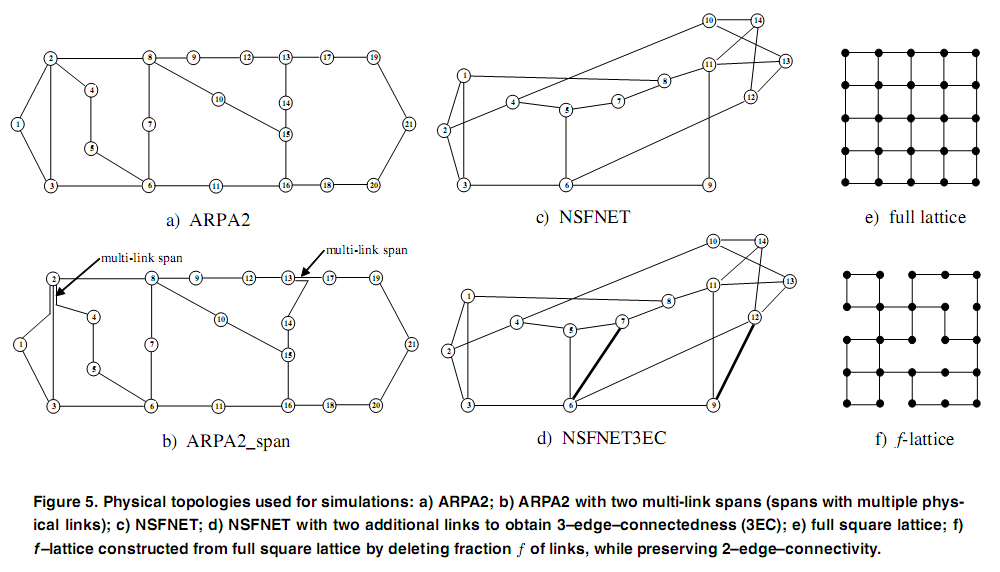
1. **Kết quả mô phỏng**

Chúng tôi chạy giải thuật SMART trên nhiều topo logic và topo vật lý khác nhau. Topo vật lý được biểu diễn trong hình 5. Một số hiệu chỉnh của đồ thị tiêu chuẩn cho vấn đề lỗi nhịp (b) và lỗi liên kết đôi (d). Để mô phỏng các mạng quy mô lớn, chúng tôi tạo ra các lattices vuông ngẫu nhiên (e, f) (*In*[*mathematics*](http://en.wikipedia.org/wiki/Mathematics)*, the****square lattice****is a type of*[*lattice*](http://en.wikipedia.org/wiki/Lattice_(group))*in a two-dimensional*[*Euclidean space*](http://en.wikipedia.org/wiki/Euclidean_space)*. It is the two-dimensional version of the*[*integer lattice*](http://en.wikipedia.org/wiki/Integer_lattice)*. It is one of the five types of two-dimensional lattices as classified by their*[*symmetry groups*](http://en.wikipedia.org/wiki/Symmetry_group)*; its symmetry group is known symbolically as*[*p4m*](http://en.wikipedia.org/wiki/Wallpaper_group#Group_p4m)*.*). Các topo logic là các đồ thị ngẫu nhiên và xác đồ thị ngẫu nhiên có bậc là hằng số, cả 2 đều có số bậc nút trung bình khác nhau. Chúng tôi cũng sử dụng các ngăn xếp của 300 topo đồng nhất với những cái được sử dụng trong [7].

Phần này được chia thành nhiều tiểu mục theo những vấn đề chúng tôi giải quyết. Đầu tiên chúng ta xét khả năng chịu lỗi khi 1 sợi quang đơn bị lỗi (1-survivability). Chúng tôi so sánh kết quả của chúng tôi với kết quả của những giải thuật khác được tìm thấy trong các tài liệu khác. Đặc biệt, chúng tôi chứng minh hiệu quả và khả năng mở rộng của SMART khi ánh xạ cho những topo lớn. Tiếp theo, chúng tôi theo dõi hiệu quả sự thích ứng của giải thuật SMART với những tình huống lỗi phức tạp hơn, như lỗi khoảng, lỗi nút và lỗi liên kết đôi. Đây là lần đầu tiên những vấn đề trên được giải quyết bằng cách tiếp cận phục hồi IP. Do đó ko có điểm benchmark để so sánh giải thuật của chúng tôi. Việc so sánh chỉ được thực hiện với phiên bản tiêu chuẩn của SMART.

* 1. **1-survivability**

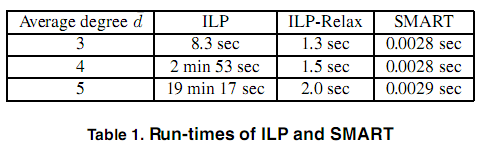
Nhiều giải pháp cho vấn đề ánh xạ 1-survivable có thể được tìm thấy trong các tài liệu [7, 8, 6, 11, 13]. So với phương pháp cơ bản của chúng tôi, chúng có một số ràng buộc cần xét đến, chẳng hạn như dung lượng, độ trễ, số lượng và vị trí của các thiết bị chuyển đổi bước sóng. Để tiện so sánh, chúng tôi đã chọn ba giải thuật, tập trung chủ yếu vào vấn đề 1-survivability. Đầu tiên là giải thuật Quy hoạch tính toán nguyên (ILP). Hai giải thuật còn lại là giải thuật Tabu Search và Simple Layout Algorithm, đều là các heuristics.



* + 1. **Giải thuật ILP**

Trong [7] các tác giả xác định các điều kiện cần và đủ cho một ánh xạ là 1-survivable. Các điều kiện này được đưa vào xây dựng ILP, chúng được sử dụng để tìm một ánh xạ 1-survivable. Sau đó, một phiên bản đơn giản nới lỏng (ILP-Relax) cho ILP được giới thiệu, làm giảm đáng kể thời gian xử lý.

Chúng tôi chạy giải thuật SMART cho cùng một topo như trong [7], cụ thể NSFNET là topo vật lý và topo logic là 300 đồ thị ngẫu nhiên có bậc = 3, 4 và 5 như trong [7] . Một ánh xạ 1-survivable được tìm thấy trong tất cả các lần chạy khi sử dụng ILP, ILP-Relax và SMART. Vì vậy khá là thú vị để so sánh thời gian chạy của các giải thuât. Các máy tính chạy giải thuât không giống nhau, nhưng cũng khá tương đương (Sun Sparc Ultra-10 và Pentium 500). Tuy nhiên, chúng tôi phải nhấn mạnh rằng SMART đã được cài đặt bằng C++ thuần trong khi, ILP đòi hỏi một chương trình chuyên dụng (CPLEX) , điều này ảnh hưởng đáng kể đến kết quả. Thời gian chạy trong [7] được thể hiện trong Bảng 1. Cột cuối cùng cho thấy kết quả của giải thuật SMART. Giải thuật SMART nhanh hơn gấp 10 mũ vài lần giải thuật ILP thuần, và khoảng lần so với phiên bản nới lỏng của ILP. Lưu ý rằng bậc của topo logic thực tế không ảnh hưởng đến thời gian chạy của SMART. Điều này là do hầu hết thời gian xử lý được dùng cho việc tìm kiếm ánh xạ 1-survivable (hội tụ của topo logic rút gọn về 1 nút); các liên kết logic còn lại (ví dụ như cạnh tự lặp e trong hình 3) được ánh xạ bởi các giải thuât tìm đường đi ngắn nhất Dijkstra, giải thuật này tốn thời gian không đáng kể.



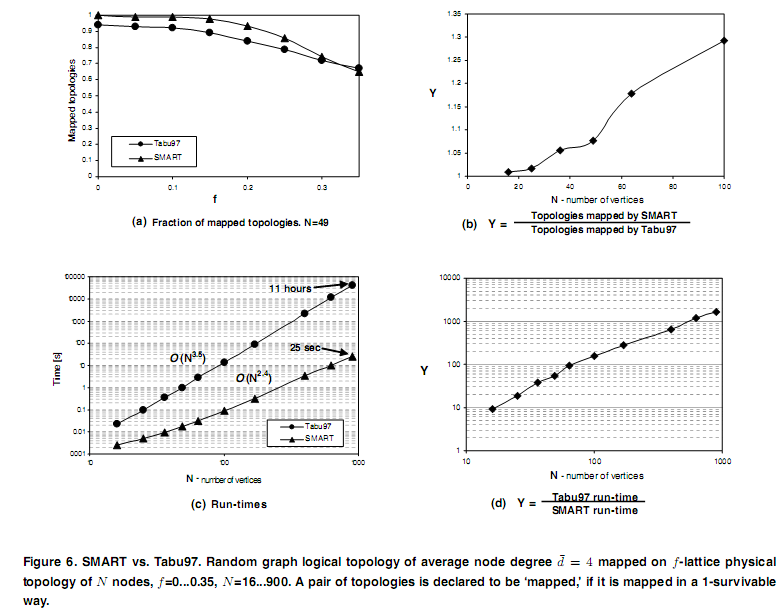
* + 1. **Tìm kiếm Tabu với Topo lớn**

Một trong những kỹ thuật được sử dụng rộng rãi và hiệu quả nhất để giải quyết vấn đề ánh xạ 1-survivable là Tabu Search. Chúng tôi thực hiện Tabu Search theo một trong những cách có trong [6], chúng tôi gọi nó là Tabu97. Do Tabu Search nhanh hơn đáng kể so với phương pháp ILP (được mô tả trong phần trước), chúng tôi thực hiện các mô phỏng cho các đồ thị tương đối lớn để nghiên cứu Tabu Search và khả năng mở rộng của SMART.

Topo vật lý là một f-lattice (mạng tinh thể ) (Hình 5f) với tỷ lệ cạnh bị xóa f từ 0 đến 0,35. Giá trị tối đa là 0,35 được chọn theo cách sao cho topo nhỏ nhất có thể là 2-cạnh liên thông . (Một lattice đầy đủ 16 nút sẽ có 24 cạnh. Vậy với f = 0,35, thì f-lattice tương ứng sẽ có 24(1−0.35) ≈ 16 cạnh, đây là giá trị nhỏ nhất để đảm bảo đồ thị là 2-cạnh liên thông – 1 topo dạng chu trình). Topo logic là đồ thị ngẫu nhiên 2-cạnh liên thông với bậc đỉnh trung bình = 4. Trong hình 6, chúng tôi trình bày các kết quả thu được trong việc mô phỏng trên một máy Pentium 4. Chúng tôi theo dõi các topo khác nhau có số đỉnh từ 16-900. Hình 6a, b liên quan đến hiệu năng của giải thuât, tức là, khả năng tìm thấy ánh xạ 1-survivable. Trong hình 6a, tỷ lệ của các topo được ánh xạ thành công được vẽ cùng với tỷ lệ cạnh xóa f cho topo có số cạnh là hằng số N = 49. Chúng tôi quan sát thấy tỷ suất của topo được ánh xạ trong giải thuật SMART cao hơn đáng kể so với giải thuật Tabu97. Nhìn chung, Hình 6a cũng khẳng định rằng ánh xạ cùng 1 topo logic trên các topo vật lý thưa hơn sẽ khó khăn hơn.

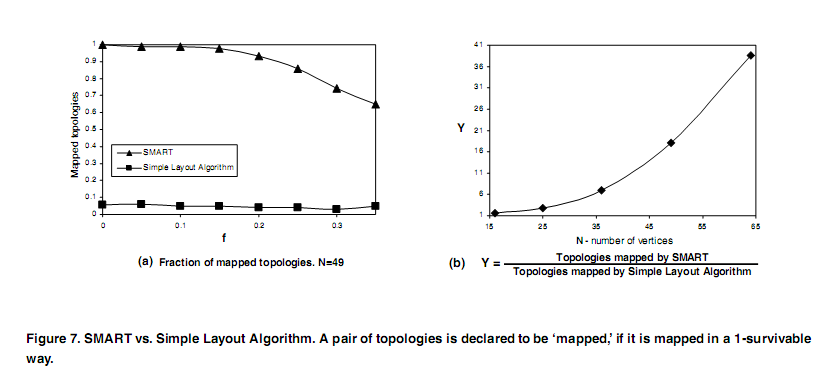
Hình 6b mô tả sự phụ thuộc của hiệu năng giải thuât với kích thước của topo. Ưu điểm của SMART tăng cùng với N, với N = 16 tỷ lệ này là không đáng kể nhưng với N = 100 tỷ lệ đã đạt tới 30%. Một lần nữa, điều này là do SMART đã chia để trị. Với N nhỏ, không có nhiều cái để chia và SMART không thể tận dụng đặc tính này của nó.

Khá là thú vị xem xét thời gian chạy của các giải thuât. Do cài đặt của Tabu97 đồng nhất với cài đặt của SMART (cả 2 đều viết bằng C + +, sử dụng cùng các cấu trúc và hàm), nên sẽ hợp lý để so sánh thời gian xử lý thực sự của chúng. Chúng được đưa ra trong hình. 6c dưới dạng đồ thị log-log sccale. Các đường cong gần như tuyến tính, do đó độ phức tạp của giải thuât là đa thức, với O () cho Tabu97 và O () cho SMART. Cả hai giá trị này đều phù hợp với giới hạn tối đa trong lý thuyết, đó là O () [6] và O (), tương ứng. Chú ý rằng Tabu97 mất khoảng 11 giờ để giải quyết trường hợp topo 900 nút, trong khi SMART chỉ mất 25 giây. Hình 6d là một so sánh trực tiếp về thời gian chạy của hai giải thuật. SMART nhanh hơn Tabu97 theo hàm 10 mũ, một lần nữa sự khác biệt phụ thuộc lớn vào N.



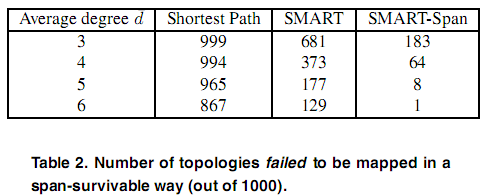
**5.1.3. Simple Layout Algorithm**

Giải thuât Simple Layout Algorithm do Sasaki đề xuất trong [13], tương tự như SMART, cũng sử dụng nguyên lý chia để trị. Vì vậy, chúng tôi hy vọng nó cũng sẽ nhanh như SMART. Chúng tôi thực hiện các phiên bản của Simple Layout Algorithm với tính toán đa liên kết, DEGR node ordering và EL-M link code (mô tả chi tiết có thể được tìm thấy trong [13]). Các tác giả đã thông báo tập các tham số này có hiệu quả nhất. Topo vật lý và topo logic tương tự như mô tả trong tiểu mục trước (Tabu Search). Thời gian chạy của Simple Layout Algorithm ít hơn khoảng 3 lần so với SMART. Tuy nhiên, như minh họa trong hình 7, tỷ lệ topo được ánh xạ (1-survivability) khi dùng Simple Layout Algorithm nhỏ hơn rất nhiều so với SMART. Mặc dù với N = 16 sự khác biệt là không đáng kể, nhưng với N = 64 nó đã gấp 40 lần. Nguyên nhân dẫn đến hiệu năng nghèo nàn của Simple Layout Algorithm và sự phụ thuộc mạnh vào kích thước của topo là như sau. Khi xây dựng, Simple Layout Algorithm chỉ ngăn chặn các nút đơn khỏi bị tách ra trong trường hợp một sợi bị lỗi. Do đồ thị nhỏ (tác giả sử dụng topo 6 nút cho các mô phỏng), sự tách ra của một nút đơn là trường hợp phổ biến nhất của mất kết nối, Simple Layout Algorithm khi đó sẽ hiệu quả. Nhưng với các đồ thị lớn, sẽ càng có thêm khả năng chia tách thành các đồ thị con lớn hơn (hơn một nút đơn). Simple Layout Algorithm không tính đến chúng, khiến cho hiệu năng giảm đáng kể. Đây là một bằng chứng cho thấy tính chịu lỗi là một vấn đề khó và đòi hỏi các giải thuât tương đối phức tạp.



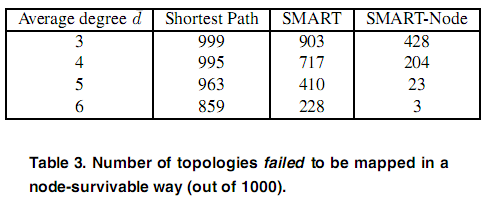
* 1. **Span failure survivability**

Để minh họa cho phiên bản giải thuật SMART-Span (xem phần 4.1), chúng tôi lấy mạng ARPA2 và giả sử có 2 nhịp đa liên kết như mô tả trong hình 5b. Các topo logic là các đồ thị 2-cạnh liên thông ngẫu nhiên có bậc nút trung bình = 3 ... 6. Chúng tôi tạo ra 1000 topo logic cho mỗi . Các kết quả được trình bày trong Bảng 2. Giải thuật đường đi ngắn nhất và giải thuật SMART (không có bảo vệ nhịp) được đưa vào để tham khảo. Kết quả cho thấy chỉ vài nhịp đa liên kết có thể làm mất tính chịu lỗi khi ánh xạ không tính đến nó.Tuy nhiên, SMART-Span cho phép bảo vệ hiệu quả. Thời gian chạy của SMART và SMART-Span (không được thể hiện ở đây) là có thể so sánh được.



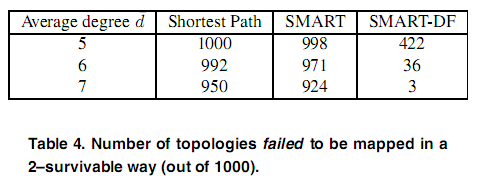
* 1. **Node failure survivability**

SMART-Node (xem 4.2) được chạy thử nghiệm với các thiết lập tương tự như SMART-Span trong phần trước, ngoại việc các topo vật lý là một mạng ARPA2 không bị sửa đổi, như trong hình. 5a. Kết quả của mô phỏng này được trình bày trong Bảng 3. SMART-Node có hiệu năng khá tốt và thời gian chạy của nó (không được hiển thị ở đây) chỉ hơi cao hơn SMART thuần 1 chút.



* 1. **2–survivability**

Để đạt được 2-survivability, chúng tôi áp dụng SMART-DF (xem Phần 4.3). Do các điều kiện cần thiết để 2-survivability là tính 3-cạnh liên thông của các topo vật lý và topo logic, chúng tôi sử dụng NSFNET3EC (xem hình 5d) làm topo vật lý và các topo logic là các đồ thị 3-cạnh liên thông ngẫu có bậc đỉnh trung bình là = 5 ... 7. Chú ý rằng bậc này lớn hơn so với các ví dụ trước đó, bởi vì 2-survivability đòi hỏi các topo phải liên thông mạnh hơn. Bảng 4 trình bày các kết quả. Rõ ràng, SMART thuần là hoàn toàn không hiệu quả khi đối mặt với những lỗi liên kết đôi, trong khi đó, SMART-DF thực hiện tốt hơn rất nhiều. Thời gian chạy của SMART-DF (không được hiển thị ở đây) lâu hơn 3 lần so với của SMART.



Chúng tôi cũng đã thử nghiệm SMART-DF cho những topo lớn, có N = 49. Các topo vật lý là 3-cạnh liên thông f-lattice, trong đó f = 0,1. Bốn đỉnh góc của f-lattice này được bổ sung kết nối với các đỉnh gần nhất (trên đường chéo) để có thể là 3-cạnh liên thông. Topo logic là đồ thị ngẫu nhiên 3-cạnh liên thông có bậc nút trung bình bằng 7. SMART-DF đã ánh xạ được 732 trong số 1000 topo. Thời gian chạy trung bình là 0,08 giây, chỉ lâu hơn 4 lần thời gian chạy của SMART thuần trong cùng điều kiện. Những thời gian này được ghi nhận trong lúc chạy mà không cần xác minh tính 2-survivability. Điều này là bởi quá trình xác minh đã đóng một vai trò rất quan trọng trong trường hợp lỗi liên kết đôi. Thời gian xác minh tính 2-survivability trung bình gấp đôi tổng thời gian để SMART-DF tự ánh xạ, nó kéo dài khoảng 0,15 giây. Điều này là do việc kiểm tra tính liên thông của topo logic phài tiến hành cho | | ( | |- 1 ) / 2 cặp sợi (Trong đó | | là số cạnh trong topo vật lý). Chú ý rằng, việc xác minh tính 1-survivability chỉ đòi hỏi kiểm tra | | lần, vậy nên việc xác minh tính 2-survivability có độ phức tạp lớn hơn ( ||- 1 ) / 2 lần so với xác minh 1-survivability. Trong trường hợp của chúng tôi, | | = 80, tỷ lệ sẽ là gấp khoảng 40 lần. Quả thật, thời gian xác minh tính 1-survivability trong tình huống này nằm trong khoảng từ 0.004 ≈ 0,15/ 40 giây. Hãy nhớ lại rằng trong các phương pháp tiếp cận heuristic tiêu chuẩn (ví dụ, Tabu Search hoặc mô phỏng luyện kim), giải pháp tạm thời được đánh giá (kiểm tra) nhiều lần trước khi hội tụ tới kết quả cuối cùng. Do đó, tổng thời gian chạy của các heuristics này bị chi phối bởi thời gian đánh giá giải pháp. Do việc xác minh tính 2-survivability có độ phức tạp lớn hơn(|| -1) / 2 lần so với xác minh tính 1-survivability, khả năng mở rộng như của Tabu Search tìm ánh xạ 2 -survivable sẽ bị hạn chế đáng kể, trái với cách tiếp cận SMART-DF.

1. **Kết luận và công việc sắp tới**

Giải thuât SMART có vẻ như là 1 kỹ thuật rất hứa hẹn để xây dựng ánh xạ chịu lỗi trong mạng WDM. Các mô phỏng của chúng tôi đã chỉ ra rằng SMART nhanh hơn gấp 100-1000 lần so với các giải thuật trước đó và có nhiều khả năng mở rộng hơn. Hơn nữa SMART còn tỏ ra thích hợp để giải quyết các tình huống lỗi tại nhịp, nút và lỗi kép. Đây có lẽ là lần đầu tiên, những tình huống lỗi trên được giải quyết bởi IP Restoration.

Các phân tích chính thức và chứng minh tính đúng đắn của giải thuật SMART sẽ được chúng tôi tiến hành trong tương lai. Chúng tôi cũng sẽ áp dụng giải thuật cho các trường hợp thực tế có tính đến ràng buộc về dung lượng, tính liên tục của bước sóng và các lưu lượng mạng.

**Tài liệu tham khảo**

[1] L. Sahasrabuddhe, S. Ramamurthy, and B. Mukherjee. Fault management in IP-Over-WDM Networks: WDM Protection vs. IP Restoration. IEEE Journal on Selected Areas in Communications, 20(1), January 2002.

[2] D. Colle et al. Data-centric optical networks and their survivability. IEEE Journal on Selected Areas in Communications, 20(1):6–20, January 2002.

[3] G. Iannaccone, C.-N. Chuah, S. Bhattacharyya, and C. Diot. Feasibility of IP restoration in a tier-1 backbone. Sprint ATL Research Report Nr. RR03-ATL-030666.

[4] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C-N.Chuah, and C. Diot. Characterization of Failures in an IP Backbone. Proc. of IEEE INFOCOM’04, 2004.

[5] G. Li, B. Doverspike, and Ch. Kalmanek. Fiber Span Failure Protection in Mesh Optical Networks. Optical Networks Magazine, 3(3):21–31, May/June 2002.

[6] J. Armitage, O. Crochat, and J. Y. Le Boudec. Design of a Survivable WDM Photonic Network. Proceedings of IEEE INFOCOM 97, April 1997.

[7] E. Modiano and A. Narula-Tam. Survivable lightpath routing: a new approach to the design of WDM-based networks. IEEE Journal on Selected Areas in Communications, 20(4):800–809, May 2002.

[8] F. Giroire, A. Nucci, T. Taft, and C. Diot. Increasing the Robustness of IP Backbones in the Absence of Optical Level Protection. Proc. of IEEE INFOCOM 2003, 2003.

[9] E. Leonardi, M.Mellia, andM. Ajmone Marsan. Algorithms for the Logical Topology Design in WDM All-Optical Networks. Optical Networks Magazine, January 2000.

[10] O. Crochat and J. Y. Le Boudec. Design Protection forWDM Optical Networks. IEEE Journal of Selected Areas in Communication, 16(7):1158–1165, September 1998.

[11] A. Nucci et al. Design of Fault-Tolerant Logical Topologies in Wavelength-Routed Optical IP Networks. Proc. of IEEE Globecom 2001, 2001.

[12] A. Fumagalli and L. Valcarenghi. IP Restoration vs. WDM Protection: Is There an Optimal Choice? IEEE Network, Nov/Dec 2000.

[13] G. H. Sasaki, C.-F. Su, and D. Blight. Simple layout algorithms to maintain network connectivity under faults. In Proceedings of the 2000 Annual Allerton Conference, 2000.

[14] S. il Kim and S. Lumetta. Addressing node failures in all optical networks. Journal of Optical Networking, 1(4):154–163, April 2002.

[15] H. Choi, S. Subramaniam, and H.-A. Choi. On Double-Link Failure Recovery inWDM Optical Networks. Proc. of IEEE INFOCOM’02, 2002.

[16] W. He,M. Sridharan, and A. K. Somani. Capacity Optimization for Surviving Double-Link Failures in Mesh-Restorable Optical Networks. Proc. of OptiComm’02, 2002.

[17] M. Clouqueur and W. D. Grover. Mesh-restorable Networks with Complete Dual-failure Restorability and with Selectively Enhanced Dual-failure Restorability Properties,. Proc. of OptiComm’02, 2002.

[18] I. Chlamtac, A. Farag, and T. Zhang. Lightpath (Wavelength) RoutinginLargeWDMNetworks. IEEE Journal on Selected Areas in Communications, 14(5):909–913, 1996.

[19] D. Banerjee and B. Mukherjee. A practical approach for routing and wavelength assignment in large wavelength routed optical networks. IEEE Journal on Selected Areas in Communications Special Issue on Optical Networks, 14(5):903–908, June 1996.

[20] J. Gross and J. Yellen. Graph Theory and its Applications. CRC Press, 1999.

[21] A. Frank. Packing paths, circuits and cuts - a survey (in Paths, Flows and VLSI-Layout). Springer, Berlin, 1990.