TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN/ĐỒ ÁN CUỐI KÌ MÔN PHÂN TÍCH THIẾT KẾ VÀ GIẢI THUẬT**

**Efficient CGM-Based Parallel Algorithms for the Longest Common Subsequence Problem**

**with Multiple Substring-Exclusion Constraints**

*Người hướng dẫn*: **TS NGUYỄN CHÍ THIỆN**

*Người thực hiện*: **DƯƠNG HẢI MINH – 52100263**

**TRẦN GIA ƯU – 52100946**

Lớp **: 21050301**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN/ĐỒ ÁN CUỐI KÌ MÔN PHÂN TÍCH THIẾT KẾ VÀ GIẢI THUẬT**

**Efficient CGM-Based Parallel Algorithms for the Longest Common Subsequence Problem**

**with Multiple Substring-Exclusion Constraints**

Người hướng dẫn: **TS NGUYỄN CHÍ THIỆN**

Người thực hiện: **DƯƠNG HẢI MINH - 52100263**

**TRẦN GIA ƯU - 52100946**

Lớp **: 21050301**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn thầy Nguyễn Chí Thiện đã chỉ dạy cũng như hướng dẫn chúng em thực hiện đề tài này.

Chún gem cũng xin chân thành cảm ơn khoa Trường Đại học Tôn Đức Thắng và Khoa Công Nghệ Thông Tin đã tạo điều kiện cho chúng em thực hiện đề tài

*TP. Hồ Chí Minh, ngày 14 tháng 12 năm 2023.*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

*Dương Hải Minh*

*Trần Gia Ưu*

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi / chúng tôi và được sự hướng dẫn của thầy Nguyễn Chí thiện;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 14 tháng 12 năm 2023*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Dương Hải Minh*

*Trần Gia Ưu*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Giải quyết bài toán M-STR-EC-LCS: Bài toán này yêu cầu tìm chuỗi con chung dài nhất (LCS) của hai chuỗi X và Y mà không chứa bất kỳ chuỗi con nào trong một tập hợp ràng buộc P. Bài toán này có ứng dụng quan trọng trong nhiều lĩnh vực, đặc biệt là sinh học tính toán.

Sử dụng kỹ thuật lập trình động và cây từ khóa: Tác giả đề xuất một giải thuật tuần tự dựa trên kỹ thuật lập trình động và cây từ khóa để giải quyết bài toán M-STR-EC-LCS. Giải thuật này có độ phức tạp thời gian và không gian là O(nmr), trong đó n, m, r là độ dài của X, Y và tổng độ dài của P.

Đề xuất hai giải thuật song song trên mô hình CGM: Tác giả cũng đề xuất hai giải thuật song song trên mô hình Coarse Grained Multicomputer (CGM) để tận dụng sức mạnh của các máy tính đa lõi. Giải thuật thứ nhất dựa trên việc phân chia đồ thị phụ thuộc thành các khối cố định và yêu cầu O(nmr/p) thời gian thực hiện với O(p) vòng truyền thông, trong đó p là số lượng bộ xử lý. Giải thuật thứ hai dựa trên việc phân chia đồ thị phụ thuộc thành các khối không đều và yêu cầu O nmr p thời gian thực hiện với O(kp) vòng truyền thông, trong đó k là một hằng số nguyên cho phép thiết lập phân chia không đều. Cả hai giải thuật đều yêu cầu O(r|Σ|/p) thời gian tiền xử lý, trong đó |Σ| là độ dài của bảng chữ cái.

Thực hiện nghiên cứu thực nghiệm: Tác giả cũng thực hiện nghiên cứu thực nghiệm để so sánh hiệu năng của các giải thuật song song với giải thuật tuần tự và với nhau. Kết quả cho thấy các giải thuật song song có thể đạt được tốc độ tăng lên đáng kể và giảm thời gian thực hiện khi tăng số lượng bộ xử lý. Giải thuật song song thứ hai cũng cho thấy ưu thế hơn giải thuật song song thứ nhất về khả năng giảm thời gian nhàn rỗi của các bộ xử lý.

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc153490131)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN iii](#_Toc153490132)

[TÓM TẮT iv](#_Toc153490133)

[MỤC LỤC 1](#_Toc153490134)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 4](#_Toc153490135)

[CHƯƠNG 1 – GIỚI THIỆU 5](#_Toc153490136)

[CHƯƠNG 2 – Sequential algorithm for the M-STR-EC-LCS problem 5](#_Toc153490137)

[2.1 Định nghĩa cơ bản 5](#_Toc153490138)

[2.2 Phương pháp tự động hữu hạn 6](#_Toc153490139)

[2.3 Thuật toán quy hoạch động của Wang et al 8](#_Toc153490140)

[2.4 Hiện thực hoá: 10](#_Toc153490141)

[2.4.1 Class AhoCorasick 10](#_Toc153490142)

[2.4.2 Class section2 11](#_Toc153490143)

[CHƯƠNG 3 – Mô hình đồ thị mới: đồ thị tuần hoàn có hướng và đa tầng (the multi-level DAG) 12](#_Toc153490144)

[CHƯƠNG 4 – Thuật toán CGM đầu tiên cho bài toán M-STR-EC-LCS 14](#_Toc153490145)

[4.1 Giải pháp CGM cho việc tính toán cho công việc sơ bộ 14](#_Toc153490146)

[4.2 Chiến lược phân chia đồ thị 15](#_Toc153490147)

[4.3 Ánh xạ vào bộ xử lí 17](#_Toc153490148)

[CHƯƠNG 5 – Thuật toán CGM thứ hai cho bài toán M-STR-EC-LCS 19](#_Toc153490149)

[5.1 Chiến lược phân vùng 19](#_Toc153490150)

[5.2 Sự phụ thuộc giữa các khối: 21](#_Toc153490151)

[5.3 Ánh xạ giũa các khối trên các bộ xử lí 22](#_Toc153490152)

[5.3 Cái nhìn tổng quan về thuật toán 22](#_Toc153490153)

[CHƯƠNG 6 – Tổng kết và hướng phát triển 23](#_Toc153490154)

**DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT**

**CÁC KÝ HIỆU**

**CÁC CHỮ VIẾT TẮT**

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

Hình 1: Cây từ khoá của tập P 6

Hình 2: Mô hình đồ thị mới cho bài toán M-STR-EC-LCS 13

Hình 6: Biểu diễn việc phân vùng trên đồ thị DAG 16

Hình 7: Biểu diễn sự phụ thuộc giữa các khối 17

**DANH MỤC BẢNG**

[Bảng 1 Kết quả trả về của hàm pre và output cho cây từ khoá ở hình 1 7](#_Toc387689363)

[Bảng 2 Hàm Next của Aho and corasick 7](#_Toc387689363)

CHƯƠNG 1 – GIỚI THIỆU

Bối cảnh và mục tiêu: Bài toán dãy con chung dài nhất với nhiều ràng buộc loại trừ chuỗi con (M-STR-EC-LCS) là một biến thể của bài toán dãy con chung dài nhất (LCS) có ứng dụng quan trọng trong nhiều lĩnh vực, đặc biệt là sinh học tính toán. Mục tiêu của bài toán là tìm ra dãy con chung dài nhất của hai chuỗi X và Y mà không chứa bất kỳ chuỗi con nào trong một tập ràng buộc P.

Phương pháp nghiên cứu: Các tác giả sử dụng kỹ thuật quy hoạch động để thiết kế các thuật toán song song trên mô hình Coarse Grained Multicomputer (CGM). Họ đề xuất một đồ thị phụ thuộc nhiều cấp độ (multi-level DAG) để xác định thứ tự đánh giá các bài toán con, tránh lặp lại do chồng chéo. Họ cũng đề xuất hai phương pháp phân chia đồ thị thành các khối: phân chia đều và phân chia không đều, để cân bằng tải và giảm thời gian nhàn rỗi của các bộ xử lý.

Kết quả và đóng góp: Các tác giả đã thực hiện các thử nghiệm trên các bộ dữ liệu thực tế và mô phỏng để đánh giá hiệu năng của các thuật toán song song. Kết quả cho thấy các thuật toán song song có thể đạt được tốc độ tăng lên đáng kể so với thuật toán tuần tự, và phương pháp phân chia không đều có thể giảm thiểu thời gian nhàn rỗi hơn phương pháp phân chia đều. Các đóng góp chính của bài báo là: (1) đưa ra một mô hình đồ thị mới để biểu diễn các bài toán con của bài toán M-STR-EC-LCS; (2) đề xuất hai thuật toán song song hiệu quả trên mô hình CGM; (3) thực hiện các thử nghiệm để so sánh kết quả lý thuyết và thực nghiệm.

CHƯƠNG 2 – Sequential algorithm for the M-STR-EC-LCS problem

2.1 Định nghĩa cơ bản

Cây từ khóa: Là một cây gốc có hướng, trong đó mỗi cạnh được gán nhãn bởi một ký tự, và không có hai cạnh ra khỏi cùng một nút có nhãn giống nhau. Mỗi chuỗi Pi ∈ P tương ứng với một nút v của cây sao cho các ký tự trên đường đi từ gốc đến v tạo thành chuỗi Pi. Mỗi nút của cây đại diện cho một tiền tố của một chuỗi trong P.

Hàm goto: Là một hàm ánh xạ một cặp (trạng thái, ký tự) vào một trạng thái khác hoặc thông báo “thất bại”. Hàm này được biểu diễn bởi các cạnh đậm trong hình 1.

Hàm failure: Chỉ ra nút nào cần chuyển đến khi không có ký tự nào khớp tiếp theo. Hàm này được biểu diễn bởi các cạnh nét đứt trong hình 1.

Hàm output: Ký hiệu bởi Oi cho một nút i. Là một tập hợp các chỉ số cho biết khi nút i được đạt đến, chuỗi Pj nào được khớp.

Hàm Next: Cho một cây từ khóa T và hàm failure của nó, cho mỗi nút i của T và mỗi ký tự α ∈ Σ, hàm Next δ(i, α) cho biết đích đến của nút đầu tiên trong danh sách failure của i có một cạnh được gán nhãn α. Nếu không có nút nào như vậy trong danh sách failure, hàm trả về gốc.

2.2 Phương pháp tự động hữu hạn

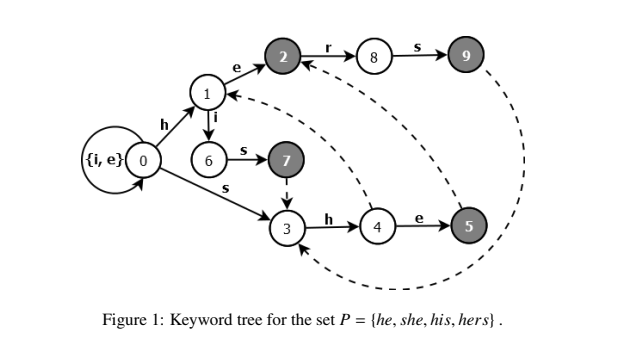
Được giới thiệu bởi Aho and corasick, Phương pháp này cho phép xác định tất cả các chuỗi con của một chuỗi X nào đó là từ khóa trong một tập hợp P = {P1, P2, . . . , Pd}

Cách xây dựng một cây từ khóa (keyword tree) T cho tập hợp P, là một cây có gốc và có hướng, trong đó:

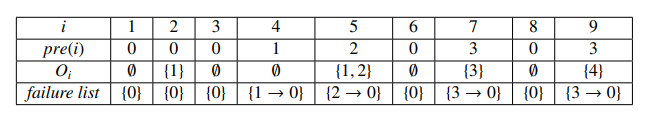
* Mỗi cạnh được gắn nhãn bởi đúng một ký tự;
* Bất kỳ hai cạnh nào ra khỏi cùng một nút đều có nhãn khác nhau;
* Mỗi chuỗi Pi ∈ P ánh xạ đến một nút v của T sao cho các ký tự trên đường đi từ gốc của T đến v chính xác tạo thành Pi, và mỗi lá của T được ánh xạ đến một chuỗi trong P.

Để áp dụng phương pháp này, bài viết trình bày các hàm chính sau:

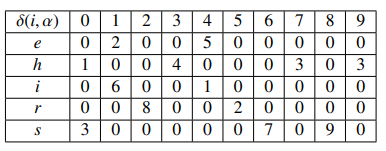
* Hàm goto: Là một hàm ánh xạ một cặp (trạng thái, ký tự) vào một trạng thái khác hoặc thông báo “thất bại”. Hàm này được biểu diễn bởi các cạnh đậm trong hình 1.
* Hàm failure: Chỉ ra nút nào cần chuyển đến khi không có ký tự nào khớp tiếp theo. Hàm này được biểu diễn bởi các cạnh nét đứt trong hình 1.
* Hàm output: Ký hiệu bởi cho một nút i. Là một tập hợp các chỉ số cho biết khi nút i được đạt đến, chuỗi Pj nào được khớp.
* Hàm Next: Cho một cây từ khóa T và hàm failure của nó, cho mỗi nút i của T và mỗi ký tự α ∈ Σ, hàm Next δ(i, α) cho biết đích đến của nút đầu tiên trong danh sách failure của i có một cạnh được gán nhãn α. Nếu không có nút nào như vậy trong danh sách failure, hàm trả về gốc.



Hình 1 - Cây từ khoá của tập P



Bảng 1 – Kết quả trả về của hàm *pre* và *output cho cây từ khoá ở hình 1*



Bảng 2 – Hàm Next của Aho and corasick

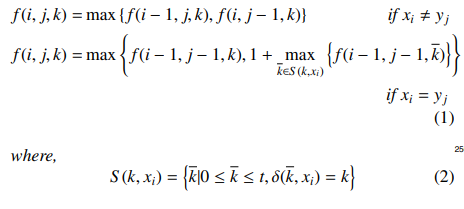
Thời gian và không gian cần thiết để xây dựng hàm Next là O(r|Σ|), trong đó r là tổng độ dài của các chuỗi trong P và |Σ| là độ dài của bảng chữ cái.

2.3 Thuật toán quy hoạch động của Wang et al

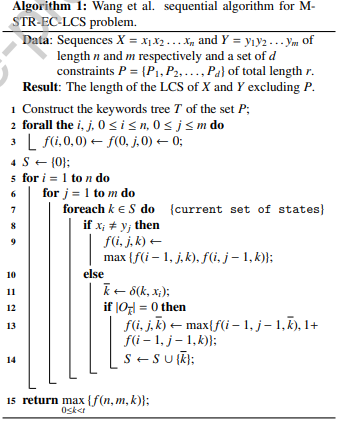
Đây là thuật toán tuần tự dựa trên kỹ thuật quy hoạch động để giải quyết bài toán M-STR-EC-LCS, tức tìm chuỗi con chung dài nhất (LCS) của hai chuỗi X và Y mà không chứa bất kỳ chuỗi con nào trong tập ràng buộc P.

Thuật toán này sử dụng một công cụ là cây từ khóa (keyword tree) để biểu diễn tập ràng buộc P và các hàm goto, pre và failure để xây dựng một hàm Next, cho biết trạng thái tiếp theo của một nút trong cây khi gặp một ký tự nhất định.

Thuật toán này định nghĩa một công thức đệ quy (1) để tính giá trị của f(i, j, k), tức độ dài của một LCS của X[1 : i] và Y[1 : j] với trạng thái k, dựa trên các giá trị của f(i − 1, j, k), f(i, j − 1, k), f(i − 1, j − 1, k) và f(i − 1, j − 1, k) với k , k. Công thức này cũng phụ thuộc vào hàm Next và tập S(k, xi) được định nghĩa bởi công thức (2).



Dựa trên công thức (1) và (2) và thuật toán được cung cấp bởi Wang et al chúng ta có được một thuật toán quy hoạch động tiêu chuẩn và điều này được thể hiện trong thuật toán 1 sau:



*Thuật toán 1*

Thuật toán này có độ phức tạp thời gian và không gian là O(nmr) với n, m là độ dài của X, Y và r là tổng độ dài của các chuỗi trong P. Thuật toán này cũng cần O(r|Σ|) thời gian tiền xử lý, với |Σ| là kích thước của bảng chữ cái.

2.4 Hiện thực hoá:

2.4.1 Class AhoCorasick

Class này dùng để tạo và mô phỏng cây từ khoá như ở hình 1.

Thuộc tính:

* max\_states: Số lượng trạng thái tối đa trong máy Aho-Corasick.
* max\_characters: Số lượng ký tự tối đa (26 chữ cái tiếng Anh).
* out: Mảng lưu trữ các trạng thái đầu ra.
* fail: Mảng lưu trữ hàm thất bại (failure function).
* transitions: Bảng chuyển trạng thái.
* words: Mảng chứa các từ cần tìm kiếm.
* states\_count: Số lượng trạng thái thực tế trong máy Aho-Corasick.

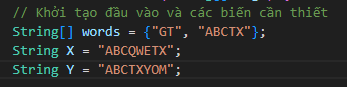
Hàm trong class:

* buildMatchingMachine(): Xây dựng máy tìm kiếm Aho-Corasick dựa trên các từ khóa đầu vào.
* getStatesCount(): Trả về số lượng trạng thái trong máy Aho-Corasick.
* getTransition(int state, int character): Trả về trạng thái tiếp theo dựa trên trạng thái hiện tại và chỉ số ký tự.
* getOut(int state): Trả về các bit đầu ra cho một trạng thái cụ thể.

2.4.2 Class section2

Ở class này chúng ta sẽ cùng lớp AhoCorasick để giải quyết vấn đề tìm chiều dài chuỗi con chung lớn nhất (LCS) giữa hai chuỗi X và Y.

Đầu vào: bao gồm các chuỗi X, Y, và tập ràng buộc P



Đầu ra: chiều dài của chuỗi con chung dài nhất.



CHƯƠNG 3 – Mô hình đồ thị mới: đồ thị tuần hoàn có hướng và đa tầng (the multi-level DAG)

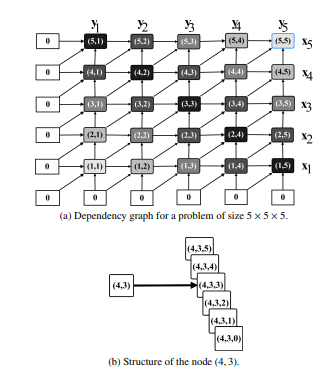
Ở đây, tác giả đã giới thiệu một mô hình đồ thị mới để phân tích các tương tác trong bài toán M-STR-EC-LCS. Nếu chúng ta áp dụng đồ thị này thì chúng ta sẽ nhận được gì:

* Mô hình này chia bài toán thành các bài toán con, nghiên cứu các mối quan hệ phụ thuộc giữa các bài toán con và sắp xếp chúng thành một đồ thị có hướng không chu trình với nhiều cấp độ.
* Mỗi bài toán con được đại diện bởi một nút trong đồ thị, và mỗi nút có một nhãn là một bộ ba (i, j, k) chỉ ra bài toán con tìm chuỗi con chung dài nhất của X[1:i] và Y[1:j] với trạng thái k.
* Mỗi cạnh trong đồ thị biểu thị một mối quan hệ phụ thuộc giữa hai bài toán con, tức là để giải bài toán con đích thì cần biết kết quả của bài toán con nguồn.
* Mỗi cấp độ trong đồ thị là một tập hợp các nút độc lập với nhau, tức là có thể giải các bài toán con tương ứng đồng thời mà không cần chờ đợi kết quả của nhau.
* Mô hình này cho phép minh họa các khái niệm như tương tác bị nhiễu, tương tác toàn phần, tương tác trực tiếp và tương tác gián tiếp một cách trực quan và chặt chẽ.
* Mô hình này cũng giúp xác định thứ tự đánh giá hợp lý của các bài toán con để tránh lặp lại và tối ưu hóa hiệu suất tính toán.

Bây giờ chúng ta sẽ xem xét định nghĩa 7 trong bài báo cáo và hiểu rõ hơn về đồ thị này:

* Cho hai chuỗi đầu vào X=x1x2…x\_n và Y=y1y2…y\_m với n và m là kích thước của hai chuỗi đầu vào và tập chuỗi ràng buộc P={P1,P2,…,P\_d } có kích thước là r. Đồ thị DAG cho bài toán M-STR-EC-LCS là một ma trận có kích thước n×m với (n+m)-1 đường chéo.
* Một nút được ký hiệu là (i, j) biểu thị tập hợp các bài toán con (i, j, k) với 0 ≤ k ≤ r;
* Nút không có cung đi ra đại diện cho vấn đề ban đầu cần giải quyết và những nút không có cung đi vào tương ứng với đầu vào ban đầu của bài toán;
* Nếu có một cung đi từ nút N1 đến nút N2 thì việc tính toán nghiệm tối ưu của các bài toán con tương ứng với nút N2 phụ thuộc vào kết quả của các bài toán con tương ứng với nút N1;
* Hai nút không được kết nối bằng một vòng cung độc lập lẫn nhau;
* Một cấp độ (hoặc bước) là một tập hợp các nút độc lập lẫn nhau.
* Cho Một level được xác định bởi tập hợp các nút có cùng đường chéo.
* Hai bài toán con là (i,j,k), 0≤k≤r và (t,u,v), 0≤v≤r được biểu diễn bằng hai nút tương ứng đó là (i,j) và (t,u) sẽ thuộc cùng cấp nếu |t-i|=|u-j|;
* Ở nút (i,j) thuộc về đường chéo (n+m)-(i+j)+1 .
* Trong đồ thị này, tất cả bài toán con thuộc về cùng một level thì sẽ độc lập với nhau.

Đánh giá mỗi nút của DAG đòi hỏi O(r) thời gian và không gian tính toán, trong đó r là tổng độ dài của tập ràng buộc.



*Hình 2 – Mô hình đồ thị mới cho bài toán M-STR-EC-LCS*

Giải thích thêm cho hình 2: trong đó các nút có cùng cấp độ được lấp đầy bằng cùng mức độ xám và cấu trúc của một nút trong biểu đồ tương ứng được hiển thị trong hình 2b. Với DAG này, việc giải bài toán bao gồm đánh giá từng cấp độ của biểu đồ này bắt đầu từ cấp 1 đến cấp n + m − 1 chứa lời giải của bài toán (node (n, m)).

CHƯƠNG 4 – Thuật toán CGM đầu tiên cho bài toán M-STR-EC-LCS

4.1 Giải pháp CGM cho việc tính toán cho công việc sơ bộ

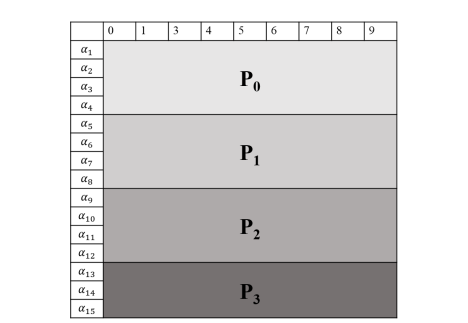
Các công việc chuẩn bị bao gồm xây dựng cây từ khóa (keyword tree) cho tập ràng buộc P, tìm liên kết thất bại (failure link) cho mỗi nút trong cây từ khóa, và tìm trạng thái tiếp theo (next state) cho mỗi nút trên mỗi ký tự của bảng chữ cái Σ.

Để song song hóa việc tính toán các trạng thái tiếp theo, tác giả chia bảng chữ cái Σ thành p tập con, mỗi tập gồm () ký tự, và gán cho mỗi bộ xử lý i tập con thứ i. Mỗi bộ xử lý sẽ tính toán giá trị của hàm δ(i, α) cho mọi 0 ≤ i ≤ t và α thuộc tập con của nó, với t là số nút của cây từ khóa.

Sau khi tính toán xong, các bộ xử lý sẽ thực hiện một vòng giao tiếp toàn cục (all-to-all communication) để chia sẻ kết quả của mình với nhau.

Thời gian và không gian tính toán cho các công việc chuẩn bị là O(), với r là tổng độ dài của các ràng buộc trong P.

Gán cho bộ xử lý i, the i th set (Σi) of the ⌈|Σ|/p⌉ giá trị đầu tiên của bảng chữ cái Σ:



Hình 3 - Ví dụ về ánh xạ tác vụ trên bộ xử lý cho các công việc sơ bộ

4.2 Chiến lược phân chia đồ thị

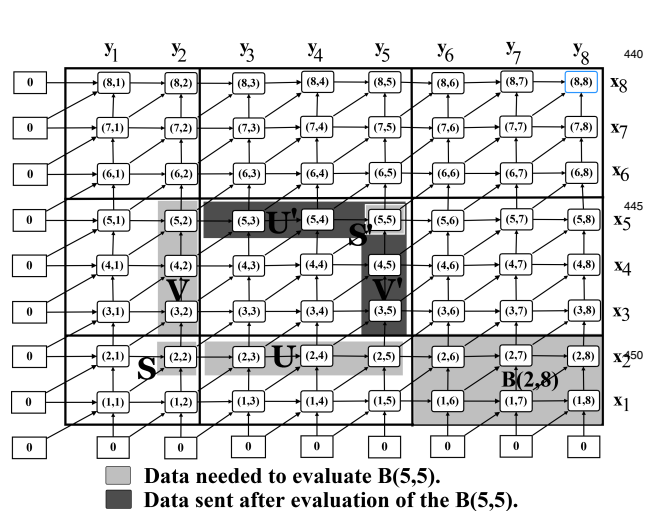
Quy tắc:

* Với α = ⌈ n / p ⌉ và β = ⌈ m / p ⌉
* 1. nếu α × p = n và β × p = m thì tất cả các khối của DAG có cùng kích thước α × β;
* 2. nếu α × p > n thì các khối của hàng khối thứ p sẽ có kích thước là [n−α×(p−1)]×β. Trường hợp này được minh họa bằng các khối B(2, 2), B(2, 5) và B(2, 8);
* 3. nếu β × p > m thì các khối ở cột 1 sẽ có kích thước là α × [m − β × (p − 1)] như các khối B(2, 2), B(5, 2) và B(8, 2).

Bổ đề 3: Bằng cách phân chia DAG thành các khối có kích thước ⌈ n/p⌉ x ⌈ m/p ⌉ , DAG chứa p lũy thừa 2 khối

Bổ đề 4: Bổ đề 4 (Sự phụ thuộc giữa các khối). với α = ⌈ n / p ⌉ và β = ⌈ m / p ⌉ , hãy xem xét khối B(i, j) của DAG:

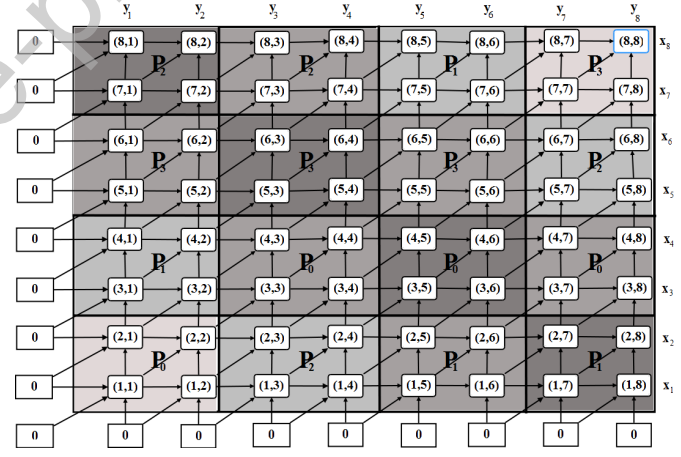
* việc đánh giá khối này chỉ phụ thuộc vào tập hợp các các nút S , U và V của các khối B(i−α, j−β), B(i−α, j) và B(i, j − β) tương ứng trong đó S , U và V được xác định bởi:
* S = {(i − α, j − β)};
* U = {(i−α, j−β+1), (i−α, j−β+2), . . . ,(i−α, j)};
* V = {(i−α+1, j−β), (i−α+2, j−β), . . . ,(i, j−β)};
* sau khi đánh giá khối này, tập mới S', U' , V‘ sẽ được truyền đạt tương ứng tới các bộ xử lý sẽ đánh giá các khối B(i + α, j + β), B(i + α, j) và B(i, j + β). Các bộ này được xác định bởi:
* S’ = {(i,j)};
* U’ = {(i, j − 1), (i, j − 2), . . . ,(i, j − β + 1)};
* V’= {(i − 1, j), (i − 2, j), . . . ,(i − α + 1, j)}.



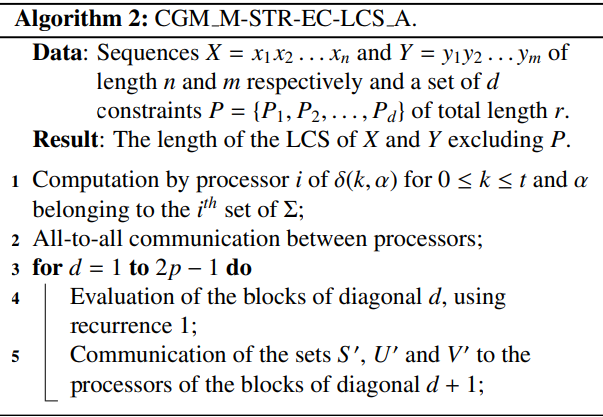
Hình 4 – Phân vùng đồ thị DAG bởi các khối 3 × 3 với n = m = 8 và p = 3.

4.3 Ánh xạ vào bộ xử lí

Sử dụng snake\_mapping gán tất cả các khối có đường chéo từ góc trên cùng bên trái đến góc dưới cùng bên phải. quá trình được làm mới cho đến khi tất cả các bộ xử lý đã được sử dụng, bắt đầu với bộ xử lý 0 và di chuyển qua các khối bằng một con đường “như con rắn”.



Hình 5 - Ánh xạ các khối vào bộ xử lí với n = m = 8 và p = 4.



Mã giả cho thuật toán CGM\_M\_STR\_EC\_LCS\_A

CHƯƠNG 5 – Thuật toán CGM thứ hai cho bài toán M-STR-EC-LCS

Thuật toán này dựa trên sự phân chia không đồng đều của DAG. Việc song song hoá các công việc sơ bộ giống hệt với thuật toán đầu tiên ở phần 4.1.

5.1 Chiến lược phân vùng

Mục đích của chiến lược này là giảm thời gian nhàn rỗi của các bộ xử lý bằng cách phân mảnh các khối có kích thước lớn thành kích thước nhỏ khi có nhiều bộ xử lý nhàn rỗi. Nguyên lý của kỹ thuật này như sau:

* DAG trước tien sẽ chia thành *p* hàng và *p* cột của khối(blocks) như được trình bày trong phần 4.2. Điều này tạo ra (2p-1) đường chéo cho tổng số các khối, nơi mà một khối đường chéo sẽ có kích thước trong đó và
* Một phân khu không đều được xây dựng trên các khối bắt đầu từ đường chéo có số khối tối đa, tức là đường chéo p. Lưu ý rằng số khối trên mỗi đường chéo sẽ giảm đi một đơn vị khi di chuyển lên(hướng Đông Bắc) và xuống (hướng Tây Nam) từ chính đường chéo này. Điều này cho phép số lượng bộ xử lí tối đa duy trì hoạt động, việc phân mảnh sẽ được thực hiện trên các đường chéo có số khối đạt sao cho các đường chéo này đạt được (hoặc vượt quá nhiều nhất một đơn vị) số khối của đường chéo p. Mặc dù kích thước các khối bị giảm do sự phân mảnh được thực hiện trên các đường chéo có số khối đạt mức , sơ đồ phân vùng không đều này sẽ giảm thiểu thời gian nhàn rỗi của bộ xử lý. Để tránh sự gia tăng mạnh mẽ của số vòng giao tiếp do việc phân mảnh quá mức, quá trình phân mảnh chỉ được lặp lại k lần, sau đó việc phân vùng sẽ trở nên truyền thống.

Về mặt hình thức, việc phân mảnh đường chéo t của các khối bao gồm việc chia 4 tất cả các khối thuộc đường chéo lớn hơn khi hoặc thuộc đường chéo nhỏ hơn khi . Việc phân mảnh lần thứ k tạo ra các khối có kích thước . Minh hoạ cho chiến lược này được mô tả trong hình 6. Số khối biểu thị đường chéo mà nó thuộc về.

Nhận xét 3: Điều quan trọng cần lưu ý là khi p lẻ, thì sẽ có thêm 2 đường chéo sau mỗi lần phân mảnh. Điều này được minh hoạ trong hình 6a trong đó 3 khối của đường chéo 9 và 4 khối của đường chéo 7 và 11 khi chúng ta thực hiện việc phân mảnh đầu tiên. Điều này cho phép chúng ta phát biểu thêm các bổ đề 8, 9 và 10 như sau:

Bổ đề 8: Một lần phân mảnh sẽ cộng thêm đường chéo. Chứng minh: việc phân mảnh được thực hiện khi số khối của một đường chéo bằng . Sự phân mảnh này sẽ giảm kích thước của khối bằng ¼ kích thước ban đầu, tạo ra đường chéo của khối; các đường chéo theo sau sẽ lần lượt chứa khối, khối, …, khối; điều này tương đương với đường chéo(ví dụ: các đường chéo 10, 11 và 12 trong hình 6a).

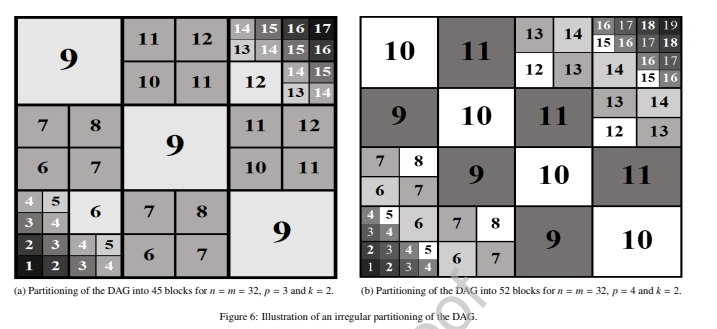
Bổ đề 9: Bằng cách thực hiện phân vùng không đều DAG, tổng số đường chéo sẽ là nếu các khối trải qua k lần phân mảnh liên tiếp. Chứng minh: Sự phân mảnh thường xuyên của DAG tạo ra khối. Nếu k lần phân mảnh hoàn thành thì theo Bổ đề 8 sẽ tăng số đường chéo lên thêm .

Bổ đề 10: Sau khi thực hiện k lần phân mảnh, số lượng các khối của DAG thu được là:

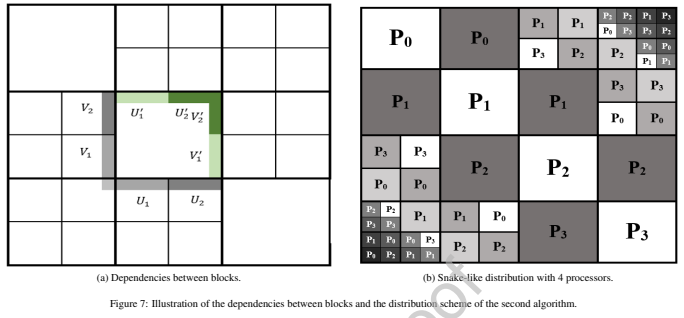
Với .

Chứng minh: Sẽ có chính xác khối không bị phân mảnh trong DAG sau quá trình phân vùng. Tùy thuộc vào tính chẵn lẻ của p, chúng ta có hai kịch bản sau:

* Khi p chẵn, có khối bên trong đường chéo từ lần phân mảnh đầu tiên đến lần thứ lần phân mảnh(ví dụ: các đường chéo 6, 7, 8, 12, 13 và 14 trong hình 6b ở lần phân mảnh thứ 2). Số lượng khối sẽ tăng sau k lần phân mảnh.
* Khi p lẻ, nguyên tắc cũng giống như khi nó chẵn, ngoại trừ việc phân mảnh ở đây làm tăng các khối bổ sung bằng trên số khối ban đầu(nhìn vào luận điểm thứ 2 của nhận xét 3)
* Kí hiệu bởi , biến xác định tính chẵn lẻ của p. Vì vậy chúng ta có:



*Hình 6 – Biểu diễn việc phân vùng trên đồ thị DAG*



*Hình 7 – Biểu diễn sự phụ thuộc giữa các khối*

5.2 Sự phụ thuộc giữa các khối:

Sự phụ thuộc giữa các khối tương tự như bổ đề 4 đã trình bày. Sự khác biệt ở đây là các tập và có thể được tạo ra bởi 2 bộ xử lí. Trong tường hợp tương tự và có thể được chia thành 2 và gửi vào hai bộ xử lí khác nhau. Những sự phụ thuộc này được mô tả ở trong hình 7a

5.3 Ánh xạ giũa các khối trên các bộ xử lí

Ở đây chúng ta sẽ xử dụng snake-like mapping để cho phép bộ xử lý hoạt động sớm nhất có thể. Sự phân phối này có thể đảm bảo cân bằng tải rất tốt trong trường hợp DAG được phân vùng không đều vì nó sẽ phân bổ đều các khối kích thước nhỏ và lớn giữa các bộ xử lý. Tuy nhiên, ánh xạ này không tối ưu hóa thông tin liên lạc. Đối với một bộ 4 bộ xử lý, sự phân bố này được mô tả trong hình 7b.

Bổ đề 11: Số khối tối đa trên mỗi bộ xử lý có phân vùng không đều DAG và sơ đồ snake-like là:

5.3 Cái nhìn tổng quan về thuật toán

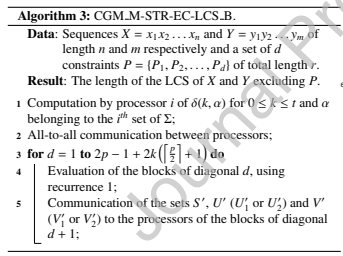
Thuật toán thứ 2 có nguyên tắc như thuật toán ở phần 4.4. Sau khi phân vùng, đường chéo của các khối sẽ đi vào bộ xử lí. the DAG is evaluated diagonal(Không biết dịch nghĩa) sau khi bắt đầu từ đường chéo đầu tiến đến kết thúc là đường chéo .

Bổ đề 12: điều bắt buộc của thuật toán này là cần có về độ phức tạp về thời gian thực thi trên một bộ xử lí.

Định lý 4: Thuật toán CGM M-STR-EC-LCS-B gỉai quyết vấn đề độ phức tạp của thuật toán M-STR-EC-LCS với vòng giao tiếp và dành thời gian thực thi cho công việc sơ bộ

Nhận xét 4: khi k=0, thuật toán CGM M-STR-EC-LCS-B đã tối ưu được phần nào CGM M-STR-EC-LCS-A chúng ta được thấy ở phần 4.

Sau đây là thuật toán hai:



*Thuật toán hai*

CHƯƠNG 6 – Tổng kết và hướng phát triển

Trong bài báo này, tác giả đã đề xuất các thuật toán song song hiệu quả trên mô hình CGM để giải quyết bài toán M-STR-EC-LCS, một biến thể của bài toán LCS có nhiều ứng dụng trong lĩnh vực sinh học tính toán. Các thuật toán được xây dựng dựa trên một mô hình đồ thị đa cấp DAG, một kỹ thuật phân tách bài toán thành các bài toán con và sắp xếp chúng theo thứ tự đúng để tránh trùng lặp. Các thuật toán được phân tích về mặt lý thuyết và thực nghiệm để đánh giá hiệu năng và khả năng mở rộng.

Các kết quả chính của bài báo là:

* Đề xuất một mô hình đồ thị đa cấp DAG để biểu diễn các mối quan hệ phụ thuộc giữa các bài toán con trong bài toán M-STR-EC-LCS.
* Đề xuất hai thuật toán song song trên mô hình CGM dựa trên DAG, một thuật toán sử dụng phân tách đều và một thuật toán sử dụng phân tách không đều để giảm thiểu thời gian nhàn rỗi của các bộ xử lý.
* Chứng minh tính đúng đắn và phân tích độ phức tạp của các thuật toán đề xuất.
* Thực hiện các thí nghiệm trên các bộ dữ liệu thực tế và mô phỏng để so sánh hiệu năng và khả năng mở rộng của các thuật toán đề xuất với thuật toán tuần tự của Wang et al. [12].
* Các kết quả thí nghiệm cho thấy các thuật toán đề xuất có thể đạt được tốc độ tăng lên gần tuyến tính và giảm thiểu thời gian nhàn rỗi của các bộ xử lý. Các thuật toán đề xuất cũng có thể xử lý các bộ dữ liệu lớn hơn so với thuật toán tuần tự do giảm thiểu không gian bộ nhớ cần thiết.

Bài báo này mở ra một số hướng nghiên cứu tiềm năng trong tương lai, chẳng hạn như:

* Cải tiến các thuật toán đề xuất bằng cách sử dụng các kỹ thuật tối ưu hóa khác nhau, ví dụ như sử dụng các cấu trúc dữ liệu hiệu quả hơn, áp dụng các kỹ thuật phân cụm hoặc phân tầng để giảm thiểu số lượng vòng truyền tin, hoặc sử dụng các kỹ thuật cân bằng tải động để phân phối công việc một cách linh hoạt hơn.
* Mở rộng các thuật toán đề xuất cho các biến thể khác của bài toán LCS, ví dụ như bài toán LCS với các ràng buộc khác nhau, bài toán LCS với các thao tác chỉnh sửa khác nhau, hoặc bài toán LCS với các độ đo tương đồng khác nhau.
* Áp dụng các thuật toán đề xuất cho các bài toán thực tế khác có liên quan đến bài toán LCS, ví dụ như bài toán so sánh chuỗi, bài toán tìm kiếm mẫu, bài toán phân tích chuỗi, hoặc bài toán phân loại chuỗi.

**TÀI LIỆU THAM KHẢO**

**Tiếng Việt**

**Tiếng Anh**

[1] V. K. Tchendji, A. N. Ngomade, J. L. Zeutouo, and J. F. Myoupo, ‘Efficient CGM-based parallel algorithms for the longest common subsequence problem with multiple substring-exclusion constraints’, *Parallel Computing*, vol. 91, p. 102598, Mar. 2020, doi: 10.1016/j.parco.2019.102598.

[2] ‘Aho-Corasick Algorithm for Pattern Searching’, GeeksforGeeks. Accessed: Dec. 14, 2023. [Online]. Available: https://www.geeksforgeeks.org/aho-corasick-algorithm-pattern-searching/

**PHỤ LỤC**

Phần này bao gồm những nội dung cần thiết nhằm minh họa hoặc hỗ trợ cho nội dung luận văn như số liệu, biểu mẫu, tranh ảnh. . . . nếu sử dụng những câu trả lời cho một *bảng câu hỏi thì bảng câu hỏi mẫu này phải được đưa vào phần Phụ lục ở dạng nguyên bản* đã dùng để điều tra, thăm dò ý kiến; **không được tóm tắt hoặc sửa đổi**. Các tính toán mẫu trình bày tóm tắt trong các biểu mẫu cũng cần nêu trong Phụ lục của luận văn. Phụ lục không được dày hơn phần chính của luận văn