ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

LÊ THỊ THU THẢO

XỬ LÝ VĂN BẢN TIẾNG VIỆT VÀ XÂY DỰNG HỆ MẬT KÉP AN TOÀN

LUẬN VĂN THẠC SĨ HỆ THỐNG THÔNG TIN

Hà Nội - 2016

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

LÊ THỊ THU THẢO

XỬ LÝ VĂN BẢN TIẾNG VIỆT VÀ XÂY DỰNG HỆ MẬT KÉP AN TOÀN

Ngành: Hệ thống thông tin
Chuyên ngành: Hệ thống thông tin
Mã số: 60.48.01.04

LUẬN VĂN THẠC SĨ HỆ THỐNG THÔNG TIN

NGƯỜI HƯỚNG DẪN KHOA HỌC: TIẾN SỸ LÊ PHÊ ĐÔ

Hà Nội - 2016

LỜI CẢM ƠN

Trước tiên tôi xin gửi lời cảm ơn sâu sắc nhất đến thầy TS. Lê Phê Đô, người thầy đã tận tâm, tận lực hướng dẫn, định hướng phương pháp nghiên cứu khoa học cho tôi; đồng thời, cũng đã cung cấp nhiều tài liệu và tạo điều kiện thuận lợi trong suốt quá trình học tập và nghiên cứu để tôi có thể hoàn thành luận văn này.

Tôi xin được gửi lời cảm ơn đến các thầy, cô trong Bộ môn Hệ thống thông tin và Khoa Công nghệ thông tin, Trường Đại học Công nghệ - Đại học Quốc gia Hà Nội đã nhiệt tình giảng dạy và truyền đạt những kiến thức, kinh nghiệm quý giá trong suốt thời gian tôi học tập tại trường.

Tôi xin gửi lời cảm ơn đến các bạn học viên lớp K20-HTTT, những người đồng hành trong suốt khóa học và có nhiều góp ý bổ ích cho tôi. Cảm ơn gia đình, bạn bè đã quan tâm và động viên giúp tôi có nghị lực phấn đấu để hoàn thành tốt luận văn này.

Do kiến thức và thời gian có hạn nên luận văn chắc chắn không tránh khỏi những thiếu sót nhất định.

Một lần nữa xin gửi lời cảm ơn chân thành và sâu sắc.

Hà Nội, tháng 10 năm 2016

Học viên thực hiện

Lê Thị Thu Thảo

LỜI CAM ĐOAN

Luận văn thạc sĩ đánh dấu cho những thành quả, kiến thức tôi đã tiếp thu được trong suốt quá trình rèn luyện, học tập tại trường. Tôi xin cam đoan luận văn **"Xử lý văn bản tiếng việt và xây dựng hệ mật kép an toàn"** được hoàn thành bằng quá trình học tập và nghiên cứu của tôi dưới sự hướng dẫn của TS. Lê Phê Đô.

Trong toàn bộ nội dung nghiên cứu của luận văn, các vấn đề được trình bày đều là những tìm hiểu và nghiên cứu của cá nhân tôi hoặc là trích dẫn các nguồn tài liệu và một số trang web đều được đưa ra ở phần Tài liệu tham khảo.

Tôi xin cam đoan những lời trên là sự thật và chịu mọi trách nhiệm trước thầy cô và hội đồng bảo vệ luận văn thạc sĩ.

Hà Nội, tháng 10 năm 2016

Lê Thị Thu Thảo

MỤC LỤC

DANH SÁCH CÁC KÝ HIỆU VÀ CHỮ VIẾT TẮT

STT	Từ viết tắt	Ý nghĩa
1	AES	Advanced Encryption Standard
2	BBS	Blum-Blum-Shub
3	DES	Data Encryption Standard
4	NIST	National Institute of Standards and Technology
5	$\oplus$	Phép toán XOR

DANH MỤC BẢNG BIỂU

DANH MỤC HÌNH VẼ

MỞ ĐẦU

Tính cấp thiết của đề tài luận văn

Trong thời kỳ mà khoa học kỹ thuật phát triển như vũ bão hiện nay, việc liên lạc và trao đổi thông tin trở nên nhanh gọn, dễ dàng hơn rất nhiều, đặc biệt là với sự xuất hiện của Internet và mạng máy tính. Tuy nhiên, bên cạnh những tiện ích mà chúng ta ai cũng có thể dễ dàng nhận thấy, thì việc liên lạc hay trao đổi thông tin qua mạng truyền dẫn cũng có thể gây ra những tác động tiêu cực, nhất là khi những cơ sở dữ liệu, thông tin quan trọng liên quan đến bí mật quốc gia hay bí mật kinh doanh, tài chính của doanh nghiệp bị đánh cắp, làm sai lệch hay giả mạo. Điều này có thể ảnh hưởng nghiêm trọng tới lợi ích, chiến lược kinh doanh của các tổ chức, các doanh nghiệp lớn nhỏ hay lớn hơn là vận mệnh của cả một quốc gia, dân tộc. Do đó, vấn đề bảo mật thông tin trở nên cấp thiết hơn bao giờ hết và cần sự quan tâm, vào cuộc của tất cả các cơ quan chức năng và cộng đồng doanh nghiệp.

Tình hình nghiên cứu

Trước đây, khi công nghệ thông tin còn chưa phát triển, khi nói đến vấn đề bảo mật thông tin, chúng ta thường hay nghĩ đến các biện pháp đơn giản nhằm đảm bảo thông tin được trao đổi hay cất giữ một cách an toàn và bí mật như: Đóng dấu, ký niêm phong, lưu giữ tài liệu trong két sắt có khóa tại nơi được bảo vệ nghiêm ngặt hoặc khi nhận được một văn bản mà nhìn bên góc trái của văn bản có khung chữ "bí mật" tức là văn bản đó cần được giữ bí mật còn nếu là văn bản có dấu mũi tên, bên trong mũi tên có chữ hỏa tốc, tức là văn bản khẩn, nhanh, triển khai gấp hay dùng mật mã mã hóa thông điệp chỉ có người gửi và người nhận mới hiểu được thông điệp…

Ngày nay, với sự phát triển mạnh mẽ của công nghệ thông tin và đi kèm với nó là tốc độ ứng dụng công nghệ thông tin vào cuộc sống, công việc của con người cũng ngày một tăng lên, dường như bất kể một quốc gia, tổ chức, cá nhân nào đều phải dựa vào công nghệ thông tin để phục vụ cho công việc và cuộc sống của mình. Do đó, ngày càng có nhiều thông tin được lưu giữ trên máy vi tính và gửi đi trên mạng Internet và điều này cũng làm nảy sinh hàng loạt vấn đề mới, đặc biệt là sự gia tăng của tội phạm mạng liên quan đến đánh cắp thông tin, lừa đảo…

Lý do chọn đề tài

Theo thống kê của Trung tâm ứng cứu khẩn cấp máy tính Việt Nam - VNCERT, năm 2015 Việt Nam có 4.484 sự cố tấn công lừa đảo, 6.122 sự cố thay đổi giao diện, 14.115 sự cố về mã độc đặc biệt, nhiều trang web, cổng thông tin điện tử của Cơ quan nhà nước bị tấn công thay đổi giao diện; gần đây nhất, vào ngày 29 tháng 7/2016 trang mạng của Vietnam Airlines bị tin tặc nước ngoài tấn công gây ra những tổn thất vô cùng to lớn. Như vậy, ta có thể thấy, việc trao đổi thông tin qua mạng truyền dẫn có ý nghĩa vô cùng quan trọng đối với cuộc sống, công việc của chúng ta nhưng cũng sẽ là

vô cùng nguy hiểm nếu như bị đánh cắp, tấn công mạng, đặc biệt là bí mật quốc gia bị các thế lực thù địch đánh cắp và sử dụng để chống phá ta. Do đó, đi kèm với việc ứng dụng công nghệ thông tin thì đồng nghĩa chúng ta phải gia tăng cảnh giác và có các biện pháp để bảo mật thông tin, phòng chống tấn công mạng.

Với tính chất cấp thiết của cuộc sống và công việc ngày nay của chúng ta khi ứng dụng công nghệ thông tin cần phải được bảo mật, từ việc tận dụng những ưu điểm của mã hóa cổ điển với mã hóa hiện đại tạo nên những ưu điểm để phòng chống tấn công của tội phạm mạng một cách hiệu quả nhất, tôi đã tiến hành nghiên cứu và lựa chọn đề tài: **"Xử lý văn bản tiếng việt và xây dựng hệ mật kép an toàn".**

Nội dung bao gồm:

Chương 1: Xử lý ngôn ngữ tự nhiên và xử lý văn bản tiếng Việt

Chương 2: Một số kiến thức về mật mã

Chương 3: Xây dựng hệ mật kép và ứng dụng

Phần kết luận và hướng phát triển: Rút ra kết luận và hướng phát triển của luận văn.

Mục tiêu nghiên cứu

Thu nhập dữ liệu được xử lý một cách ngẫu nhiên, qua đó có thể tin tưởng mẫu dữ liệu là đảm bảo tính ngẫu nhiên. Xử lý ngôn ngữ tự nhiên nói chung và xử lý văn bản nói riêng có sử dụng mô hình học máy với phương pháp cực đại Entropy để giúp các từ phân tách ra có nghĩa, tạo ra từ điển bao gồm số lượng từ lớn đủ để mã hóa văn bản. Đếm tần suất từ giúp việc gán mã ID sau này thuận tiện, tối ưu. Đánh số định danh cho từng từ giúp quá trình mã hóa và giải mã văn bản dễ dàng. Sử dụng bộ sinh số ngẫu nhiên BBS đảm bảo độ an toàn và tính ngẫu nhiên cho dãy số được sinh ra. Thuận toán đơn giản, hiệu suất cao.

Môi trường thực nghiệm

- Chip: Intel Core i5 CPU 2.4GHz
- Ram: 2.00 GB
- Hệ điều hành: Microsoft Windows 8 32 bits
- Công cụ lập trình: Eclipse

Giới hạn của đề tài:

Do thời gian và sự hiểu biết còn có hạn, nên luận văn chỉ xử lý văn bản text và xây dựng hệ mật kép an toàn với các văn bản text.

CHƯƠNG 1: XỬ LÝ NGÔN NGỮ TỰ NHIÊN VÀ XỬ LÝ VĂN BẢN TIẾNG VIỆT

Con người hiện nay đang phải đối mặt với "cơn đại hồng thủy" dữ liệu: feedback của đối tác, thông tin của đối thủ cạnh tranh, facebook, emails, tweets, thông tin họp báo, các văn bản về sản phẩm và công nghệ. Khai thác những dữ liệu này một cách hiệu quả là một việc cực kỳ quan trọng.

Vấn đề ở đây là gì? Vấn đề ở đây là ở chỗ có quá nhiều thông tin cần xử lý cùng một lúc. Vấn đề này sẽ được giải quyết bởi các chương trình xử lý ngôn ngữ tự nhiên. Các chương trình xử lý ngôn ngữ tự nhiên có khả năng đọc và hiểu văn bản với tốc độ cao để rút trích ra những tri thức đáng giá.

Xử lý ngôn ngữ tự nhiên được ứng dụng cho hàng loạt công việc khác nhau, từ việc đảm bảo an ninh - quốc phòng cho đến các vấn đề kinh doanh và đời sống cá nhân.

1.1 Xử lý ngôn ngữ tự nhiên

Xử lý ngôn ngữ tự nhiên là một nhánh của trí tuệ nhân tạo thực hiện việc phân tích, đoán nhận và sinh ra ngôn ngữ con người một cách tự động hoặc bán tự động.

Xử lý ngôn ngữ tự nhiên nghiên cứu sự tương tác giữa máy tính và ngôn ngữ của con người. Xử lý ngôn ngữ tự nhiên ra đời từ những năm 1940, với rất nhiều công trình nghiên cứu theo hai hướng chính là: 1) ô-tô-mát (automaton) và các mô hình xác suất (probabilistic models) vào những năm 1950; 2) các phương pháp dựa trên ký hiệu (symbolic) và các phương pháp ngẫu nhiên (stochastic) vào những năm 1970. Giai đoạn tiếp theo (1970-1983) chứng kiến sự bùng nổ trong nghiên cứu về xử lý tiếng nói và ngôn ngữ. Ngày nay với sự phát triển nhanh chóng, học máy (machine learning) đã trở thành trung tâm của phần lớn các lĩnh vực thuộc khoa học máy tính, bao gồm xử lý ảnh và thị giác máy tính (computer vision), tin sinh học (bioinformatics), các hệ tư vấn (recommender systems), kỹ nghệ phần mềm, và xử lý ngôn ngữ tự nhiên.

Các bước xử lý ngôn ngữ tự nhiên

- Phân tích hình thái - Trong bước này toàn bộ văn bản sẽ được phân tách thành các từ, cụm từ.
- Phân tích cú pháp - Phân tách các từ trong câu để thấy được sự liên hệ giữa các từ.
- Phân tích ngữ nghĩa - Các văn bản được kiểm tra ngữ nghĩa để rút ra ý nghĩa chính xác của các từ.
- Tích hợp văn bản - Ý Nghĩa của một câu riêng biệt có thể phụ thuộc vào những câu đứng trước, đồng thời nó cũng có thể ảnh hưởng đến các câu đứng sau.
- Phân tích thực nghĩa - Các câu được phân tách để tìm ra ý nghĩa thực sự của nó. Để làm được điều này cần phải có kiến thức thực tế.

Các ứng dụng của xử lý ngôn ngữ tự nhiên:

- Nhận dạng chữ viết: Có hai kiểu nhận dạng, thứ nhất là nhận dạng chữ in, ví dụ nhận dạng chữ trên sách giáo khoa rồi chuyển nó thành dạng văn bản điện tử. Với chương trình nhận dạng chữ viết in có thể chuyển hàng ngàn đầu sách trong thư viện thành văn bản điện tử trong thời gian ngắn. Nhận dạng chữ viết của con người có ứng dụng trong khoa học hình sự và bảo mật thông tin.

- Nhận dạng tiếng nói: Nhận dạng tiếng nói rồi chuyển chúng thành văn bản tương ứng. Giúp thao tác của con người trên các thiết bị nhanh hơn và đơn giản hơn, chẳng hạn thay vì gõ một tài liệu nào đó bạn đọc nó lên và trình soạn thảo sẽ tự ghi nó ra.

- Tổng hợp tiếng nói: Từ một văn bản tự động tổng hợp thành tiếng nói. Thay vì phải tự đọc một cuốn sách hay nội dung một trang web, nó tự động đọc cho chúng ta.

- Dịch tự động (machine translate): Như tên gọi đây là chương trình dịch tự động từ ngôn ngữ này sang ngôn ngữ khác.

- Tìm kiếm thông tin: Đặt câu hỏi và chương trình tự tìm ra nội dung phù hợp nhất. Thông tin ngày càng đầy lên theo cấp số nhân, đặc biệt với sự trợ giúp của internet việc tiếp cận thông tin trở lên dễ dàng hơn bao giờ hết. Việc khó khăn lúc này là tìm đúng nhất thông tin mình cần giữa bề bộn tri thức và đặc biệt thông tin đó phải đáng tin cậy.

- Tóm tắt văn bản: Từ một văn bản dài tóm tắt thành một văn bản ngắn hơn theo mong muốn nhưng vẫn chứa những nội dung thiết yếu nhất.

- Khai phá dữ liệu (data mining) và phát hiện tri thức: Từ rất nhiều tài liệu khác nhau phát hiện ra tri thức mới.

1.1.1 Nội dung xử lý ngôn ngữ tự nhiên

a. Phân tích hình thái

Phân tích hình thái có thể chia thành 3 khâu xử lý dưới đây:

- Phân đoạn từ vựng (word segmentation) phân giải câu văn được nhập vào thành các từ có thứ tự.

- Phân loại từ (part-of-speech tagging) quyết định từ loại của từ vựng

- Phục hồi thể nguyên dạng của từ (lemmatization) làm trở lại nguyên dạng ban đầu các từ vựng bị biến đổi thể (inflection) hoặc được kết hợp (conjugatetion). Trong tiếng Anh, các từ trong câu được sắp xếp với nhau bằng các khoảng trắng nhưng trong tiếng Nhật, tiếng Thái, tiếng Trung Quốc là ngôn ngữ mà giữa các từ vựng không có khoảng trắng. Vì thế xử lý phân đoạn từ câu văn được nhập vào là cần thiết.

b. Phân tích cú pháp

Phân tích cú pháp là bước xử lý quan trọng trong các bài toán hiểu ngôn ngữ tự nhiên. Nó cung cấp một nền tảng vững chắc cho việc xử lý văn bản thông minh như các hệ thống hỏi đáp, khai phá văn bản và dịch máy.

Việc phân tích cú pháp câu có thể chia làm hai mức chính. Mức thứ nhất là tách từ và xác định thông tin từ loại. Mức thứ hai là sinh cấu trúc cú pháp cho câu dựa trên các từ và từ loại do bước trước cung cấp.

1.1.2 Ứng dụng của xử lý ngôn ngữ tự nhiên

a. Nhận dạng chữ viết

Nhận dạng chữ viết tay được chia thành hai lớp bài toán lớn là nhận dạng chữ viết tay trực tuyến (online) và nhận dạng chữ viết tay ngoại tuyến (offline). Trong nhận dạng chữ viết tay ngoại tuyến, dữ liệu đầu vào được cho dưới dạng các ảnh được quét từ các giấy tờ, văn bản. Ngược lại nhận dạng chữ viết tay trực tuyến là nhận dạng các chữ trên màn hình ngay khi nó được viết. Trong hệ nhận dạng này máy tính sẽ lưu lại các thông tin về nét chữ như thứ tự nét viết, hướng và tốc độ của nét…

Các giai đoạn phát triển

- **Giai đoạn 1: (1900 - 1980)**

- Nhận dạng chữ được biết đến từ năm 1900, khi nhà khoa học người Nga Tyuring phát triển một phương tiện trợ giúp cho những người mù.

- Các sản phẩm nhận dạng chữ thương mại có từ những năm1950, khi máy tính lần đầu tiên được giới thiệu tính năng mới về nhập và lưu trữ dữ liệu hai chiều bằng cây bút viết trên một tấm bảng cảm ứng. Công nghệ mới này cho phép các nhà nghiên cứu làm việc trên các bài toán nhận dạng chữ viết tay on-line.

- Mô hình nhận dạng chữ viết được đề xuất từ năm 1951 do phát minh của M. Sheppard được gọi là GISMO, một robot đọc-viết.

- Năm 1954, máy nhận dạng chữ đầu tiên đã được phát triển bởi J. Rainbow dùng để đọc chữ in hoa nhưng rất chậm.

- Năm 1967, Công ty IBM đã thương mại hóa hệ thống nhận dạng chữ.

- **Giai đoạn 2: (1980 - 1990)**

- Với sự phát triển của các thiết bị phần cứng máy tính và các thiết bị thu nhận dữ liệu, các phương pháp luận nhận dạng đã được phát triển trong giai đoạn trước, đã có được môi trường lý tưởng để triển khai các ứng dụng nhận dạng chữ.

- Các hướng tiếp cận theo cấu trúc và đối sánh được áp dụng trong nhiều hệ thống nhận dạng chữ.

- Trong giai đoạn này, các hướng nghiên cứu chỉ tập trung vào các kỹ thuật nhận dạng hình dáng chứ chưa áp dụng cho thông tin ngữ nghĩa. Điều này dẫn đến sự hạn chế về hiệu suất nhận dạng, không hiệu quả trong nhiều ứng dụng thực tế.

- **Giai đoạn 3: (Từ 1990 đến nay)**

- Các hệ thống nhận dạng thời gian thực được chú trọng trong giai đoạn này.

- Các kỹ thuật nhận dạng kết hợp với các phương pháp luận trong lĩnh vực học máy (Machine Learning) được áp dụng rất hiệu quả.

- Một số công cụ học máy hiệu quả như mạng nơ ron, mô hình Markov ẩn, SVM (Support Vector Machines) và xử lý ngôn ngữ tự nhiên...

b. Nhận dạng tiếng nói

Trên thế giới đã và đang có rất nhiều công trình nghiên cứu về vấn đề này với rất nhiều phương pháp nhận dạng tiếng nói khác nhau. Và những nghiên cứu đó cũng có những thành công đáng kể. Có thể kể đến như: hệ thống nhận dạng tiếng nói tiếng Anh Via Voice của IBM, Spoken Toolkit của CSLU (Central of Spoken Laguage Under-standing), Speech Recognition Engine của Microsoft, Hidden Markov Model toolkit của đại học Cambridge, CMU Sphinx của đại học Carnegie Mellon,… ngoài ra, một số hệ thống nhận dạng tiến nói tiếng Pháp, Đức, Trung Quốc,... cũng khá phát triển. Ở Việt Nam thì hầu như chỉ mới có bộ phần mềm Vspeech của nhóm sinh viên trường Đại học Bách Khoa TP. HCM, các phần mềm khác chỉ thử nghiệm trong phòng thí nghiệm, chưa được sử dụng thực tế vì chưa đạt trên 100 từ. Phần mềm Vspeech được phát triển từ mã nguồn mở Microsoft Speech SDK nhận dạng tiếng Anh, thông qua dữ liệu, phương thức trung gian, việc nhận dạng được chuyển trong Vspeech để nhận biết tiếng Việt.

c. Tổng hợp tiếng nói

Tổng hợp tiếng nói (text-to-speech, TTS) có mục tiêu ngược với mục tiêu của nhận dạng tiếng nói. Kiến trúc của một hệ thống TTS giống như kiến trúc đọc chữ của con người, bao gồm một môđun xử lý ngôn ngữ tự nhiên (bộ tiền xử lý nhằm tổ chức các câu thành danh sách, bộ phân tích hình thái, bộ phân tích ngữ cảnh, bộ phân tích câu cú pháp, ngôn điệu, …), có khả năng sinh ra phiên âm phù hợp với cách phát âm của quá trình đọc văn bản cùng với ngữ điệu, ngôn điệu; và một mô đun xử lý tín hiệu số, mô đun này chuyển thông tin tượng trưng nhận được thành tiếng nói (mô đun letter-to-sound và mô đun sinh ra ngôn điệu). Khi hai khối xử lý ngôn ngữ tự nhiên và xử lý tín hiệu số được định nghĩa rõ ràng, việc nghiên cứu về hai quá trình có thể được thực hiện riêng rẽ, độc lập với nhau. Khối xử lý tín hiệu số phải xét đến các hạn chế phát âm, vì sự biến đổi ngữ âm (phần động, chuyển tiếp giữa các âm) là quan trọng đối với việc hiểu lời nói hơn là các phần tĩnh của lời nói. Tổng hợp tiếng nói có thể đạt được

cơ bản theo hai phương pháp thuộc về hai trường phái tổng hợp tiếng nói có nội dung và mục tiêu khác nhau:

- Phương pháp thứ nhất được thực hiện dưới dạng các quy tắc mô tả âm vị, ảnh hưởng lẫn nhau giữa các âm vị khi phát ra một âm (tổng hợp bằng qui luật).
- Phương pháp thứ hai lưu giữ những đơn vị âm cơ bản, biến đổi đơn vị âm cơ bản và đồng thời tạo ra cơ sở dữ liệu tiếng nói, sử dụng chúng như là các đơn vị âm học cơ bản để tạo thành lời nói (phương pháp tổng hợp theo xích chuỗi).

d. Dịch tự động

Dịch máy là một trong những ứng dụng chính của xử lý ngôn ngữ tự nhiên. Mặc dù dịch máy đã được nghiên cứu và phát triển hơn 50 năm qua, song vẫn tồn tại nhiều vấn đề cần nghiên cứu. Ở Việt nam, dịch máy đã được nghiên cứu hơn 20 năm, nhưng các sản phẩm dịch máy hiện tại cho chất lượng dịch còn nhiều hạn chế. Hiện nay, dịch máy được phân chia thành một số phương pháp như: dịch máy trên cơ sở luật, dịch máy thống kê và dịch máy trên cơ sở ví dụ. Do những khác biệt về ngữ hệ, khác biệt về văn hóa và thiếu vắng nguồn tài nguyên, nên các phương pháp dịch máy hiện hữu thường gặp trở ngại khi áp dụng vào cặp ngôn ngữ Anh - Việt.

Dịch máy dựa trên ngữ liệu đang được áp dụng vào nhiều hệ thống dịch tự động trong những năm gần đây, việc lấy đúng được cặp ánh xạ đích và nguồn một cách tự động là một yêu cầu thiết yếu cho các phương pháp dịch dựa trên ngữ liệu. Phương pháp dịch thống kê hiện tại đang cải thiện được chất lượng dịch bằng các mô hình huấn luyện không chỉ dựa trên cơ sở các từ đơn mà còn dựa trên các cụm từ. D.Marcu và W.Wong, Kenji Yamada và Kevin Knight, P.Koehn, F.J.Och, và D.Marcuđã cho kết quả khả quan. Tuy nhiên các cụm từ trong các nghiên cứu này không thực sự là cụm từ của ngôn ngữ học.

1.2 Xử lý văn bản tiếng Việt

Xử lý văn bản tiếng Việt bao gồm nhiều bài toán: Phân tách từ, Phân loại văn bản, **Dịch tự động, Tóm tắt văn bản, ...**

1.2.1 Tách từ

Bài toán phân tách từ (word segmentation) là bài toán quan trọng nhất, nó quyết định thành công của các bài toán khác như dịch tự động (machine translation), tóm tắt văn bản (text summarization), tìm kiếm thông tin (information retrieval), trích chọn thông tin (information extraction), v.v.

Trong văn bản tiếng Việt đặt dấu cách giữa các âm tiết chứ không phải giữa các từ. Một từ có thể có một, hai hoặc nhiều âm tiết nên có nhiều cách phân chia các âm tiết thành các từ, gây ra nhập nhằng. Việc phân giải nhập nhằng này gọi là *bài toán tách từ*.

Tiêu chí quan trọng nhất trong bài toán tách từ đương nhiên là độ chính xác. Hiện tại người ta đã đạt được độ chính xác lên đến 97% tính theo từ. Tuy nhiên nếu tính theo câu (số câu được tách hoàn toàn đúng/tổng số câu) thì độ chính xác chỉ khoảng 50%. Đây là vấn đề nghiêm trọng đối với các bước xử lý sau như phân tích ngữ pháp, ngữ nghĩa vì một từ bị tách sai có ảnh hưởng toàn bộ đến cách phân tích cả câu.

1.2.2 Gán nhãn từ[6]

Quá trình gán nhãn từ loại có thể chia làm 3 bước:

- Phân tách xâu kí tự thành chuỗi các từ. Giai đoạn này có thể đơn giản hay phức tạp tuỳ theo ngôn ngữ và quan niệm về đơn vị từ vựng. Chẳng hạn đối với tiếng Anh hay tiếng Pháp, việc phân tách từ phần lớn là dựa vào các kí hiệu trắng. Tuy nhiên vẫn có những từ ghép hay những cụm từ công cụ gây tranh cãi về cách xử lý. Trong khi đó với tiếng Việt thì dấu trắng càng không phải là dấu hiệu để xác định ranh giới các đơn vị từ vựng do tần số xuất hiện từ ghép rất cao.

- Gán nhãn tiên nghiệm, tức là tìm cho mỗi từ tập tất cả các nhãn từ loại mà nó có thể có. Tập nhãn này có thể thu được từ cơ sở dữ liệu từ điển hoặc kho văn bản đã gán nhãn bằng tay. Đối với một từ mới chưa xuất hiện trong cơ sở ngữ liệu thì có thể dùng một nhãn ngầm định hoặc gắn cho nó tập tất cả các nhãn. Trong các ngôn ngữ biến đổi hình thái người ta cũng dựa vào hình thái từ để đoán nhận lớp từ loại tương ứng của từ đang xét.

- Quyết định kết quả gán nhãn, đó là giai đoạn loại bỏ nhập nhằng, tức là lựa chọn cho mỗi từ một nhãn phù hợp nhất với ngữ cảnh trong tập nhãn tiên nghiệm. Có nhiều phương pháp để thực hiện, trong đó người ta phân biệt chủ yếu *các phương pháp dựa vào quy tắc ngữ pháp* mà đại diện nổi bật là phương pháp Brill và *các phương pháp xác suất* . Ngoài ra còn có các hệ thống sử dụng mạng nơ-ron, các hệ thống lai sử dụng kết hợp tính toán xác suất và ràng buộc ngữ pháp, gán nhãn nhiều tầng.

Về mặt ngữ liệu, các phương pháp phân tích từ loại thông dụng hiện nay dùng một trong các loại tài nguyên ngôn ngữ sau:

- Từ điển và các văn phạm loại bỏ nhập nhằng.

- Kho văn bản đã gán nhãn, có thể kèm theo các quy tắc ngữ pháp xây dựng bằng tay.

- Kho văn bản chưa gán nhãn, có kèm theo các thông tin ngôn ngữ như là tập từ loại và các thông tin mô tả quan hệ giữa từ loại và hậu tố.

- Kho văn bản chưa gán nhãn, với tập từ loại cũng được xây dựng tự động nhờ các tính toán thống kê. Trong trường hợp này khó có thể dự đoán trước về tập từ loại.

Các bộ gán nhãn từ loại dùng từ điển và văn phạm gần giống với một bộ phân tích cú pháp. Các hệ thống học sử dụng kho văn bản để học cách đoán nhận từ loại cho mỗi từ. Từ giữa những năm 1980 các hệ thống này được triển khai rộng rãi vì việc xây dựng kho văn bản mẫu ít tốn kém hơn nhiều so với việc xây dựng một từ điển chất lượng cao và một bộ quy tắc ngữ pháp đầy đủ. Một số hệ thống sử dụng đồng thời từ điển để liệt kê các từ loại có thể cho một từ, và một kho văn bản mẫu để loại bỏ nhập nhằng. Bộ gán nhãn của chúng tôi nằm trong số các hệ thống này.

Các bộ gán nhãn thường được đánh giá bằng độ chính xác của kết quả: [số từ được gán nhãn đúng] / [tổng số từ trong văn bản]. Các bộ gán nhãn tốt nhất hiện nay có độ chính xác đạt tới 98% .

1.2.3 Phân cụm từ tiếng Việt[7]

Việc phân nhóm các cụm từ tiếng Việt đóng một vai trò hết sức quan trọng trong các ứng dụng thực tế như tìm kiếm thông tin, trích chọn thông tin, và dịch máy.

Bài toán phân cụm có thể hiểu là việc gộp một dãy liên tiếp các từ trong câu để gán nhãn cú pháp. Việc nghiên cứu bài toán phân cụm trên thế giới đã được thực hiện khá kỹ lưỡng cho nhiều ngôn ngữ bao gồm: Tiếng Anh, Tiếng Trung, Tiếng Nhật, Tiếng Pháp. Gần đây các phương pháp học máy đã chứng tỏ sức mạnh và tính hiệu quả khi sử dụng cho bài toán xử lý ngôn ngữ tự nhiên.

Bài toán phân cụm tiếng Việt được phát biểu như sau: Gọi X là câu đầu vào tiếng Việt bao gồm một dãy các từ tố kí hiệu $X = (X_1, X_2, \ldots, X_n)$. Chúng ta cần xác định $Y = (Y_1, Y_2, \ldots, Y_n)$ là một dãy các nhãn cụm từ (cụm danh từ, cụm động từ). Bài toán này được quy về vấn đề học đoán nhận dãy (có thể được thực hiện qua việc sử dụng các mô hình học máy. Quy trình học được thực hiện bằng cách sử dụng một tập các câu đã được gán nhãn để huấn luyện mô hình học cho việc gán nhãn câu mới (không thuộc tập huấn luyện).

1.2.4 Tóm tắt văn bản[8]

Tóm tắt văn bản sẽ giúp người dùng tiết kiệm thời gian đọc, cải thiện tìm kiếm cũng như tăng hiệu quả đánh chỉ mục cho máy tìm kiếm.

Bài toán tóm tắt văn bản tự động nhận được sự quan tâm nghiên cứu của nhiều nhà khoa học, nhóm nghiên cứu cũng như các công ty lớn trên thế giới. Các bài báo liên quan đến tóm tắt văn bản xuất hiện nhiều trong các hội nghị nổi tiếng như : DUC 2001-2007, TAC 2008, ACL 2001-2007… bên cạnh đó cũng là sự phát triển của các hệ thống tóm tắt văn bản như: MEAD, LexRank, Microsoft Word (Chức năng AutoSummarize)…

Bài toán tóm tắt đa văn bản được xác định là một bài toán có độ phức tạp cao. Thách thức lớn nhất của vấn đề tóm tắt đa văn là do dữ liệu đầu vào có thể có sự nhập

nhằng ngữ nghĩa giữa nội dung của văn bản này với văn bản khác trong cùng tập văn bản hay trình tự thời gian được trình bày trong mỗi một văn bản là khác nhau, vì vậy để đưa ra một kết quả tóm tắt tốt sẽ vô cùng khó khăn.

Rất nhiều ứng dụng cần đến quá trình tóm tắt đa văn bản như: hệ thống hỏi đáp tự động (Q&A System), tóm tắt các báo cáo liên quan đến một sự kiện, tóm tắt các cụm dữ liệu được trả về từ quá trình phân cụm trên máy tìm kiếm…

1.2.5 Trích xuất thông tin[8][9][10]

Ngày nay, với sự gia tăng nhanh chóng của dữ liệu thì trích xuất thông tin ngày càng có nhiều ứng dụng: lọc thư rác, đối chiếu lý lịch cá nhân, phân tích cảm nghĩ, phân loại tài liệu, ...

Các phương pháp trích xuất văn bản

Về cơ bản, ta có thể chia các phương pháp trích xuất văn bản (text retrieval) thành hai loại: lựa chọn tài liệu (document selection) và sắp xếp tài liệu (document ranking).

1.2.5.1 Phương pháp lựa chọn tài liệu

Đối với phương pháp lựa chọn tài liệu, câu truy vấn được xem như một ràng buộc cụ thể cho việc lựa chọn các tài liệu có liên quan. Một ví dụ điển hình cho phương pháp này đó là mô hình trích xuất boolean (Boolean retrieval model), trong đó mỗi tài liệu được biểu diễn bởi một tập các từ khóa và người sử dụng sẽ cung cấp một biểu thức boolean các từ khóa, chẳng hạn như “car AND repair shops”, “tea OR coffee”, hoặc “database system BUT NOT Oracle”. Hệ thống trích xuất sẽ nhận một truy vấn dạng boolean như vậy và trả về các tài liệu thỏa mãn biểu thức. Khó khăn đối với phương pháp này đó là việc mô tả thông tin mà người sử dụng cần bằng biểu thức boolean, bởi vậy nó chỉ thường hoạt động tốt khi người sử dụng hiểu rõ về tập tài liệu cũng như có khả năng trình bày rõ ràng câu truy vấn.

1.2.5.2 Phương pháp sắp xếp tài liệu

Phương pháp sắp xếp tài liệu sử dụng truy vấn để sắp xếp các tài liệu theo thứ tự liên quan. Thực tế cho thấy phương pháp này thích hợp cho việc trích xuất văn bản hơn so với phương pháp lựa chọn tài liệu. Hầu hết các hệ thống IR hiện đại đều sử dụng cách này để trả về một danh sách có sắp xếp các tài liệu tùy theo câu truy vấn của người sử dụng. Những kỹ thuật được dùng trong những phương pháp dạng này cũng rất đa dạng, bao gồm đại số học, logic học, xác suất, thống kê… Vấn đề chính của hướng tiếp cận này đó là làm cách nào để xấp xỉ độ đo liên quan của một tài liệu dựa vào các từ có sẵn trong tài liệu cũng như trong toàn bộ dataset. Trong phạm vi báo cáo này, chúng ta chỉ xem xét một trong những hướng tiếp cận phổ biến nhất hiện nay, đó là mô hình không gian vector (vector space model - VSM).

Ý tưởng chính của VSM như sau: chúng ta biểu diễn tất cả các tài liệu trong dataset và câu truy vấn thành các vector trong không gian nhiều chiều tương ứng với tất cả các từ khóa, sau đó sử dụng một độ đo tương tự (similarity measure) thích hợp nào đó để tính toán độ tương tự giữa vector truy vấn với các vector tài liệu. Giá trị độ tương tự sẽ được dùng để sắp xếp các tài liệu trả về.

1.2.5.3 Token hóa

Bước đầu tiên của việc trích xuất văn bản đó là định nghĩa các từ khóa đại diện cho các tài liệu, bước tiền xử lý này thường được gọi là token hóa (tokenization). Để tránh việc xử lý các từ vô dụng, chúng ta sẽ áp dụng một danh sách dừng (stop list) cho tập các tài liệu trong dataset. Danh sách dừng là tập các từ được cho rằng không liên quan đến nội dung của tài liệu. Ví dụ "a", "the", "of", "for", "with"… là các từ dừng (stop words), mặc dù chúng có thể xuất hiện rất thường xuyên trong tài liệu. Ngoài ra, ta có thể thấy rằng một nhóm các từ có thể chia sẻ chung một từ gốc (word stem). Do vậy bước tiếp theo chúng ta sẽ định ra các nhóm từ mà trong đó các từ chỉ có sự khác biệt nhỏ về cú pháp. Ví dụ, nhóm các từ "drug", "drugged", và "drugs" sẽ cùng chia sẻ chung một từ gốc là "drug".

1.2.5.4 Mô hình hóa tài liệu

Giả sử một dataset gồm d tài liệu và t từ khóa (term), chúng ta có thể mô hình hóa mỗi tài liệu thành một vecotr v trong không gian t chiều Æt. Tần số từ khóa (term frequency) được định nghĩa là số lần xuất hiện của từ t trong tài liệu d, được ký hiệu là freq(d,t). Tiếp theo, ta xây dựng ma trận trọng số term-frequency TF(d,t) phản ánh độ liên kết của từ t tương ứng với tài liệu d, trong đó 0 nếu tài liệu đó không chứa từ khóa đó và khác không trong trường hợp ngược lại. Có nhiều cách định nghĩa trong trường hợp trọng số khác không. Ví dụ, ta có thể đơn giản gán giá trị TF(d,t) = 1 nếu từ t xuất hiện trong văn bản d, hoặc sử dụng chính giá trị freq(d,t). Có nhiều cách khác nhau để chuẩn hóa giá trị tần số từ khóa.

CHƯƠNG 2. MỘT SỐ KIẾN THỨC VỀ MẬT MÃ

2.1 Giới thiệu các hệ mật

2.1.1 Hệ mật cổ điển

2.1.1.1 Hệ mật dịch chuyển

Hệ mật dịch chuyển là một trong những mật mã đơn giản và được biết đến nhiều nhất, là một dạng của mật mã thay thế. Trong đó mỗi kí tự trong văn bản được thay thế bằng một kí tự cách nó một đoạn trong bảng chữ cái sau khi dịch chuyển K đơn vị để tạo thành bản mã và dịch chuyển K đơn vị ngược lại từ bản mã để được bản rõ.

Ví dụ 2.1:

Bảng 2.1. Bảng chữ cái

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Với K=3

Bản rõ: “ Luan van cua ThaoLTT ”

Bản mã: “ oxdq ydq fxd wkdoww ”

Nhận xét độ an toàn: Độ an toàn của mã dịch chuyển rất thấp. Tập khóa K chỉ có 26 khóa, nên việc phá khóa có thể thực hiện dễ dàng bằng cách thử kiểm tra từng khóa: K = 0, 2, 3,…, 25.

2.1.1.2 Hệ mật thay thế

Trong hệ mật thay thế bộ chữ cái rõ được thay thế bằng một bộ chữ cái mới nhận được bằng cách hoán vị bảng chữ cái ban đầu.

Ví dụ 2.2:

Bảng 2.2. Hoán vị 26 chữ cái

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	
Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A

Bản rõ: “ Luan van cua ThaoLTT ”

Bản mã: “ ofzmaezmaxfzagszlogg ”

Nhận xét độ an toàn: Độ an toàn của hệ mật thay thế tương đối cao vì tập khóa K có 26! Khóa và theo Shannon thì ngưỡng an toàn của mã thay thế trong tiếng anh là 25 kí tự.

2.1.1.3 Hệ mật Vigenere

Hệ mật Vigenere là một dạng của mã khối, mỗi khóa gồm m kí tự:

$k = k_1, k_2 \ldots k_m$, trong đó k_i là các chữ cái trong bảng chữ cái latinh.

Khi mã hóa ta chia bản rõ thành các khối gồm m kí tự. Mỗi chữ cái thứ i trong khối được dịch chuyển k_i bước giống như trong mã dịch chuyển theo quy ước số bảng chữ cái latinh như sau.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Giải mã đơn giản là quá trình làm ngược lại.

Ví dụ 2.3:

Bản rõ chữ: " LETHITHUTHAO "
Chọn khoá: k = " DHCN " = {3, 7, 2, 13} với độ dài d = 4.
Bản rõ số: " 11 4 19 7 8 19 7 20 19 7 0 14 "
Mã hóa:
Chia bản rõ số thành các đoạn, mỗi đoạn gồm d = 4 số.
Với mỗi đoạn, áp dụng công thức mã hóa, ta nhận được bản mã số.

Bảng 2.3. Bản mã số hệ mật Vigenere

11 3	4 7	19 2	7 13	8 3	19 7	7 2	20 13	19 3	7 7	0 2	14 13
14	11	21	21	11	0	9	7	22	14	2	1

Bản mã số: " 14 11 21 21 11 0 9 7 22 14 2 1 "

Bản mã chữ: " OLVV LAJH XOCB "

Nhận xét độ an toàn: Độ an toàn của mã Vigenere tương đối cao. Nếu khoá gồm d ký tự khác nhau, mỗi ký tự có thể được ánh xạ vào 1 trong d ký tự có thể, do đó hệ mật này được gọi là hệ thay thế đa biểu. Như vậy số khoá (độ dài d) có thể có trong mật Vigenere là 26^d. Nếu dùng phương pháp "tấn công vét cạn", thì phải kiểm tra 26^d khóa.

2.1.1.4 Hệ mật Hill[2]

Sơ đồ Hill Lester S. Hill đưa ra năm 1929.

Đặt $P = C = Z_{26}{}^m$, m là số nguyên dương. Bản mã Y và bản rõ $X \in (Z_{26})^m$.

Tập khóa $K = \{K \in Z_{26}{}^{m*m} / (\det(K), 26) = 1\}$. (K phải có K^{-1}).

Mỗi khóa K là một "Chùm chìa khóa" (một Ma trận "Các chìa khóa").

Với mỗi $K \in K$, định nghĩa:

Hàm lập mã: $Y = (y_1, y_2, \ldots, y_m) = e_k (x_1, x_2, \ldots, x_m) = (x_1, x_2, \ldots, x_m) * K$

Hàm giải mã: $X = (x_1, x_2, \ldots, x_m) = d_k (y_1, y_2, \ldots, y_m) = (y_1, y_2, \ldots, y_m) * K^{-1}$

Ví dụ 2.4:

Bản rõ chữ: " HTTT "

Chọn m = 2, khóa $K = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$

bảo đảm UCLN (det (K), 26) = 1, tính $K^{-1} = \begin{matrix} 7 & 18 \\ 23 & 11 \end{matrix}$

Bản rõ số:	7	19	|	19	19
	x_1	x_2	|	x_1	x_2

Với mỗi bộ rõ số (x_1 , x_2), theo hàm lập mã (y_1 , y_2) = (x_1 , x_2) * K, ta tính được:

$y_1 = 11 * x_1 + 3 * x_2$, $y_2 = 8 * x_1 + 7 * x_2$

Bản mã số:	4	7	|	6	25

Bản mã chữ: " EHGZ "

Nhận xét độ an toàn: Nếu dùng phương pháp "tấn công vét cạn", thám mã phải kiểm tra số khóa có thể với m lần lượt là 2, 3, 4,… trong đó m lớn nhất là bằng độ dài bản rõ.

2.1.2 Hệ mật hiện đại

2.1.2.1 Mã khối

Trong mật mã, mã khối được sử dụng rộng rãi và có độ mật cao. Mã khối xuất hiện từ xa xưa với các hệ mật Vigenere, Hill…Trong mật mã hiện đại các hệ mật DES và AES là các mã khối nổi tiếng.

Mã hóa: Để mã hóa bản tin ta chia bản tin thành từng khối có độ dài xác định.

Ưu điểm: Tốc độ mã hoá nhanh và có độ an toàn tốt.

Độ mật: Mã khối được sử dụng hợp lý có độ mật cao.

Tốc độ mã hoá nhanh và có độ an toàn tốt.

2.1.2.2 Hệ mật AES[16][17][18][19][20][23]

- **Nguồn gốc của AES:**

AES (Advanced Encryption Standard - Tiêu chuẩn mã hóa nâng cao) được thiết kế bởi Joan Daemen và Vincent Rijmen, hai nhà khoa học người bỉ. Thuật toán được đặt tên là Rijmen khi tham gia cuộc thi thiết kế AES do Viện chuẩn quốc gia Hoa kỳ US

NIST ra lời kêu gọi tìm kiếm chuẩn mã mới vào năm 1997. Sau đó có 15 đề cử được chấp nhận vào tháng 6 năm 1998 và được rút gọn còn 5 ứng cử viên vào tháng 6 năm 1999. Đến tháng 10 năm 2000, mã Rijndael được chọn làm chuẩn mã nâng cao - AES và được xuất bản là chuẩn FIPS PUB 197 VÀO 11/2001.

- **Yêu cầu của AES:**

Phương pháp mã hóa theo khối có kích thước khối dữ liệu đầu vào và đầu ra là 128 bit, độ dài khóa có thể thay đổi linh hoạt với các giá trị 128, 192 hay 256 bit. Phương pháp mã hóa này thích hợp ứng dụng trên nhiều hệ thống khác nhau, từ các thẻ thông minh cho đến các máy tính cá nhân.

Chuẩn mã mới mạnh và nhanh hơn Triple DES. Mã mới có cơ sở lý thuyết mạnh để thời gian sống của chuẩn khoảng 20 - 30 năm (cộng thêm thời gian lưu trữ).

Khi đưa ra thành phần yêu cầu cung cấp chi tiết thiết kế và đặc tả đầy đủ. Đảm bảo rằng chuẩn mã mới cài đặt hiệu quả trên cả C và Java.

- **Cơ sở toán học của AES:**

Trong AES các phép toán cộng và nhân được thực hiện trên các byte trong trường hữu hạn $GF(2^8)$

- **Phép cộng:**

$A = (a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8)$; $B = (b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7\ b_8)$;

$C = A + B = (c_1\ c_2\ c_3\ c_4\ c_5\ c_6\ c_7\ c_8)$, trong đó: $C_i = a_i + b_i$

Ví dụ 2.5:

$A = 56_H$; $B = 3D_H$

Dạng cơ số Hecxa: $56_H + 3D_H = 93$

Dạng nhị phân: 01010110 + 00111101 = 10010011

Dạng đa thức: $(x^6 + x^4 + x^2 + x) + (x^5 + x^4 + x^3 + x^2 + 1) = (x^7 + x^4 + x + 1)$

- **Phép nhân:**

$A = (a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8)$; $B = (b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7\ b_8)$;

$C = A.B = (c_1\ c_2\ c_3\ c_4\ c_5\ c_6\ c_7\ c_8)$

Ví dụ 2.6:

$A = C3_H$; $B = 85_H$

Dạng cơ số Hecxa: $(C3_H).(85_H) = AE_H$

Dạng nhị phân: (11000011).(10000101) = 10101110

Dạng đa thức: $(x^7 + x^6 + x + 1).(x^7 + x^2 + 1) = (x^7 + x^5 + x^3 + x^2 + 1)$

- **Chu trình tạo khóa con AES[3]:**

AES thực hiện việc mở rộng khóa dựa trên khóa gốc K, tạo thành chu trình tạo khóa để sinh ra 10, 12 hoặc 14 khóa con, tương ứng với 10, 12 hoặc 14 chu kỳ lặp của giải thuật AES.

Việc mở rộng khóa chính tạo thành bảng khóa mở rộng. Bảng khóa mở rộng là mảng 1 chiều chứa các từ, mỗi từ có độ dài 4 byte, được ký hiệu *W[Nb*(Nr+1)]* (với Nb = 4). Việc phát sinh bảng khóa mở rộng phụ thuộc vào độ dài *Nk* của khóa chính.

Chu trình tạo khóa con AES sử dụng hai hàm:

SubWord() thực thiện việc thay thế từng byte thành phần của từ 4 byte được đưa vào và trả về kết quả là một từ 4 byte đã được thay thế. Việc thay thế này sử dụng bảng thay thế S-box.

RotWord() thực hiện việc dịch chuyển xoay vòng 4 byte thành phần (a, b, c,d) của từ được đưa vào. Kết quả trả về của hàm RotWord là một từ 4 byte đã được dịch chuyển (b, c, d, a).

Các hằng số chu kỳ Rcon[i] được xác định:

Rcon[i] = [x^{i-1}, {00}, {00}, {00}]

Trong đó: $x^{i-1} \Leftrightarrow \{02\}^{i-1}$ trong trường GF(2^8)

Như vậy ta có bảng hằng số mở rộng với trường hợp Nr = 10 như sau:

Bảng 2.4. Bảng hằng số mở rộng Rcon của AES - 128

01	02	04	08	10	20	40	80	1B	36
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00

Khóa của chu kỳ thứ *i* bao gồm các từ 4 byte có chỉ số từ *Nb*i* đến *Nb*(i+1) – 1* (với Nb = 4) của bảng mã khóa mở rộng. Như vậy mã khóa của chu kỳ thứ *i* bao gồm các phần tử từ *W[Nb*i], W[Nb*i + 1],…W[Nb*(i+1) – 1]*.

Bảng 2.5. Bảng khóa mở rộng AES - 128

W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	W_9	W_{10}	W_{11}	…

Khóa con chu kỳ 0	Khóa con chu kỳ 1	Khóa con chu kỳ 2	…

- **Quá trình mã hóa AES[3]:**

Giải thuật AES bao gồm nhiều bước biến đổi được thực hiện tuần tự, kết quả đầu ra của bước biến đổi này sẽ là đầu vào của bước biến đổi kia. Kết quả trung gian giữa các bước biến đổi được gọi là trạng thái (State). Độ dài của khối đầu vào, khối đầu ra cũng như độ dài của khối trung gian State là 128 bit. Được biểu diễn bằng một ma trận gồm 4 dòng và 4 cột.

Độ dài của khóa K trong giải thuật AES có thể là 128, 192 hoặc 256 bit. Khóa được biểu diễn bằng một ma trận gồm 4 dòng và Nk cột (Nk = 4, 6 hoặc 8; Nk được tính bằng độ dài của khóa chia 32). Số lượng chu kỳ tính toán trong giải thuật AES được ký hiệu là Nr, độ lớn của Nr phụ thuộc vào độ dài của khóa. Nr được xác định theo công thức:

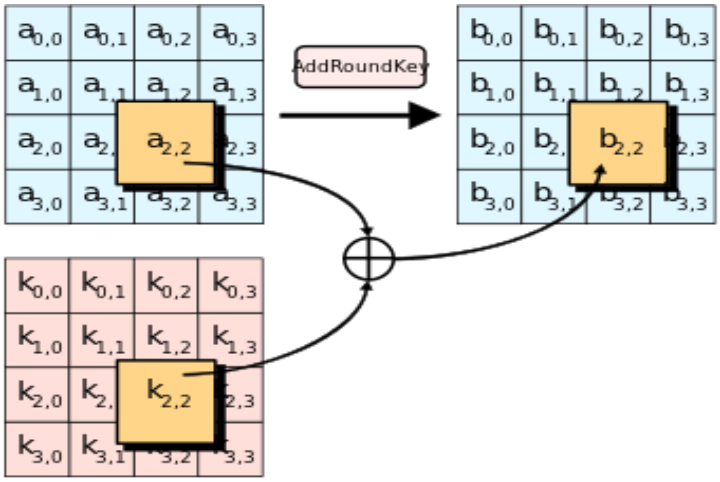
$Nr = \max\{Nb, Nk\} + 6$

Bảng 2.6. Mối liên hệ giữa Nk, Nb và Nr

	Độ dài khóa (Nk words)	**Kích thước khối (Nb words)**	**Số chu kỳ (Nr)**
AES – 128	4	4	10
AES – 192	6	4	12
AES – 256	8	4	14

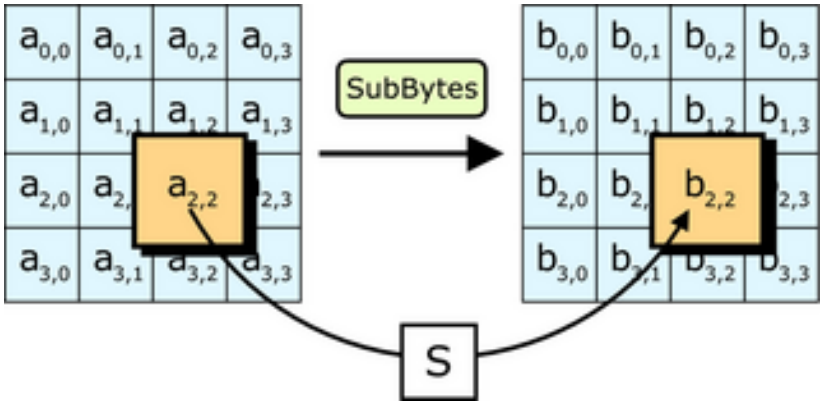
Quá trình mã hóa của giải thuật AES trải qua 10, 12 hoặc 14 chu kỳ, tương ứng với độ dài của khóa là 128, 192 hoặc 256 bit. Mỗi chu kỳ bao gồm 4 bước được thự hiện tuần tự:

Bước 1: AddRoundKey - mỗi byte của khối trạng thái được kết hợp với khóa con. Các khóa con này được tạo ra từ quá trình tạo khóa con (xem Hình 2.1)[23].



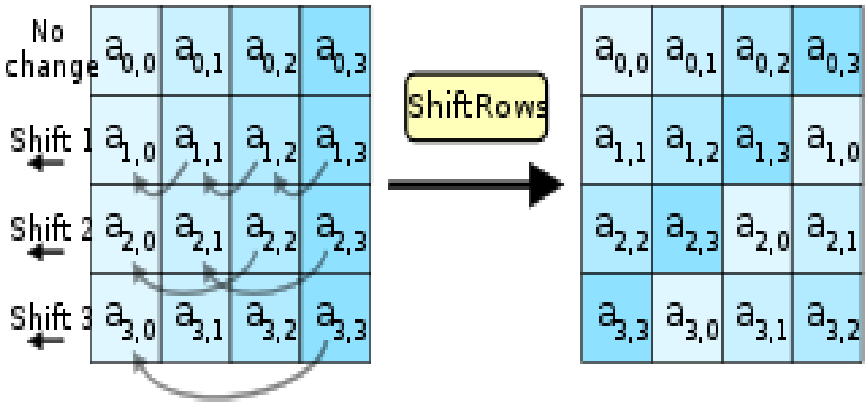
Hình 2.1. AddRoundKey

Bước 2: SubBytes - mỗi byte trong khối trạng thái được thay thế bằng một byte khác trong bảng tra S-box (xem Hình 2.2)[23].



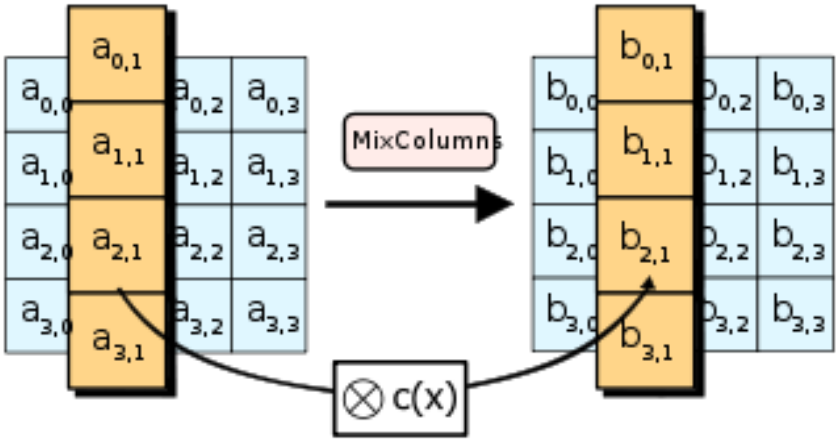
Hình 2.2. SubBytes

Bước 3: ShiftRows - Các hàng trong khối được dịch vòng, số lượng vòng dịch phụ thuộc vào thứ tự của hàng (xem Hình 2.3)[23].



Hình 2.3. ShiftRows

Bước 4: MixColumns - các cột trong khối được trộn theo một phép biến đổi tuyến tính(xem Hình 2.4) [23].



Hình 2.4. MixColumns

Các bước của quá trình mã hóa được thực hiện trên trạng thái hiện hành S. Kết quả S' của mỗi bước sẽ trở thành đầu vào của bước tiếp theo.

Quá trình giải mã là một quá trình ngược của quá trình mã hóa AES. Quá trình giải mã cũng trải qua 10, 12 hoặc 14 chu kỳ tương ứng với số chu kỳ của quá trình mã hóa.

Mỗi chu kỳ gồm 4 bước được thực hiện tuần tự với nhau gồm InvSubBytes, InvShiftRows, InvMixColumns (là các phép biến đổi ngược với SubBytes, ShiftRows, MixColumns) và bước AddRoundKey.

Quá trình giải mã được thể hiện như lưu đồ dưới đây (xem Hình 2.5):[3]

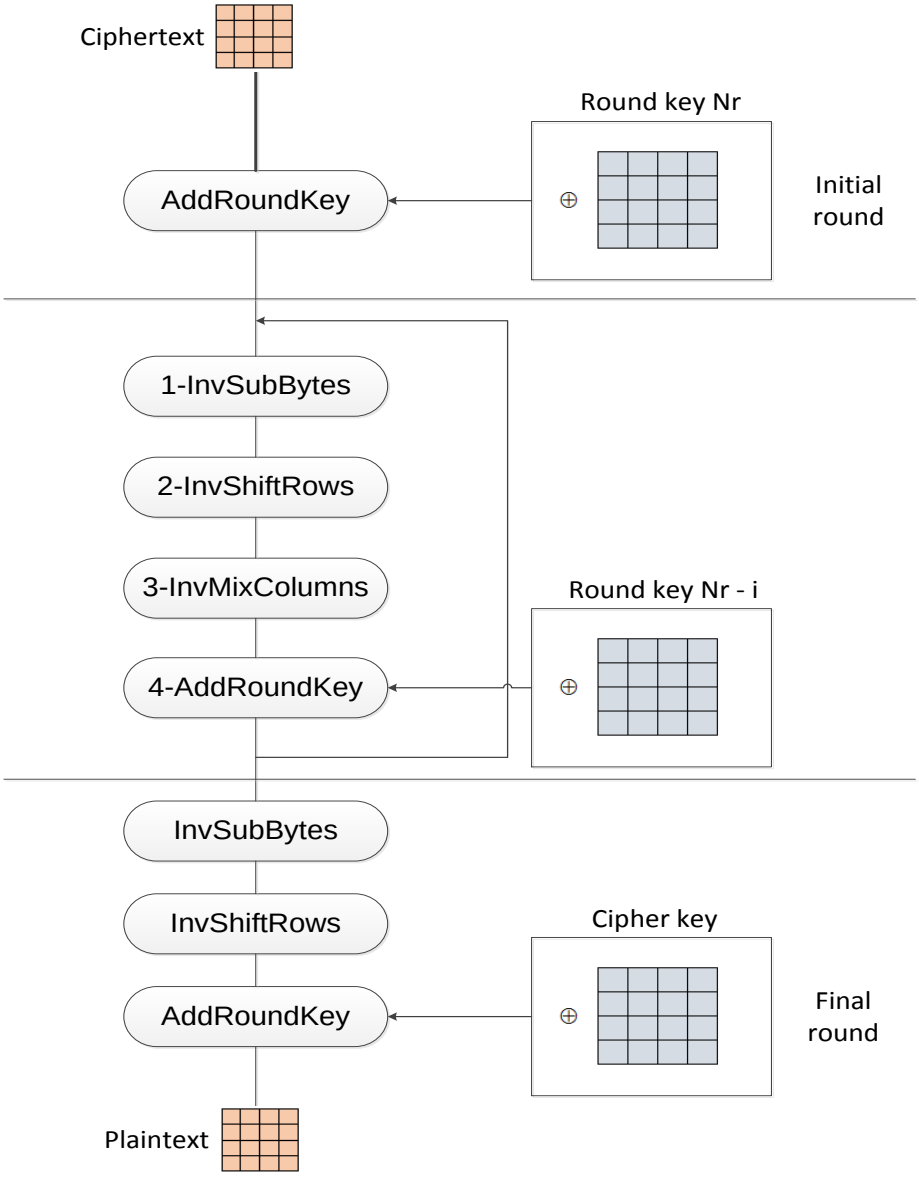
- Khối dữ liệu đầu vào là Ciphertext được sao chép vào mảng trạng thái S.

- Thực hiện thao tác AddRoundKey đầu tiên trước khi thực hiện các chu kỳ mã hóa. Sử dụng khóa ở chu kỳ thứ Nr của chu trình mã hóa.

- Mảng trạng thái sau đó sẽ trải qua Nr = 10, 12 hay 14 chu kỳ biến đổi (tương ứng với 10, 12 hay 14 chu kỳ mã hóa).

+ Nr – 1 chu kỳ đầu tiên: mỗi chu kỳ gồm 4 bước biến đổi liên tiếp nhau.

+ Chu kỳ thứ Nr, thao tác InvMixColumns được thay thế bằng thao tác AddRoundkey.



Hình 2.5. Quy trình giải mã AES

- **Đánh giá giải thuật AES:**

Kể từ khi được công nhận là giải thuật mã hóa tiên tiến, AES ngày càng được xã hội chấp nhận. Ban đầu AES chỉ được sử dụng để mã hóa các dữ liệu nhạy cảm. Về sau này, người ta đã dùng nó để mã hóa các thông tin bí mật. Giải thuật AES-192/256 được sử dụng để bảo vệ các thông tin mật và tối mật. Nó được đưa vào các tiêu chuẩn ISO, IETF, IEEE. Cho đến nay, hàng trăm sản phẩm ứng dụng dựa theo tiêu chuẩn mã hóa AES đã được NIST cấp chứng chỉ. Ở Việt Nam, Thông tư số 01/2011/TT-BTTTT ban hành ngày 04 tháng 01 năm 2011 của Bộ thông tin truyền thông đã khuyến nghị sử dụng AES là giải thuật mã hóa sử dụng cho các thông tin, văn bản trong các cơ quan Nhà nước.

- **Ưu điểm của giải thuật AES:**

AES là giải thuật mã hóa có tốc độ xử lý nhanh, đã được chính phủ Hoa Kỳ tuyên bố là có độ an toàn cao, được sử dụng làm tiêu chuẩn mã hóa mới thay thế cho tiêu chuẩn DES đã lỗi thời. AES được sử dụng để mã hóa các thông tin mật đến tuyệt mật.

AES có cấu trúc đơn giản, rõ ràng và có mô tả toán học rất đơn giản.

- **Độ an toàn của AES:**

Thiết kế và độ dài khóa của thuật toán AES (128, 192 và 256 bit) là đủ an toàn để bảo vệ các thông tin được xếp vào loại tối mật. Các thông tin tuyệt mật sẽ phải dùng khóa 192 hoặc 256 bit. Một vấn đề khác nữa là cấu rúc toán học của AES không giống với các thuật toán mã học khác, AES có mô tả toán học khá đơn giản. Tuy điều này chưa dẫn đến mối nguy hiểm nào nhưng một số nhà nghiên cứu sợ rằng sẽ có người lợi dụng được cấu trúc này trong tương lai. Vào thời điểm năm 2006, dạng tấn công AES duy nhất thành công là tấn công kênh bên tức là không tấn công trực tiếp vào thuật toán mã hóa mà thay vào đó tấn công lên các hệ thống thực hiện thuật toán có sơ hở làm lộ dữ liệu.

- **Nhược điểm của AES:**

Mặc dù AES được đánh giá là an toàn nhưng với phương pháp "tấn công kênh biên" thì nó chưa thực sự an toàn.

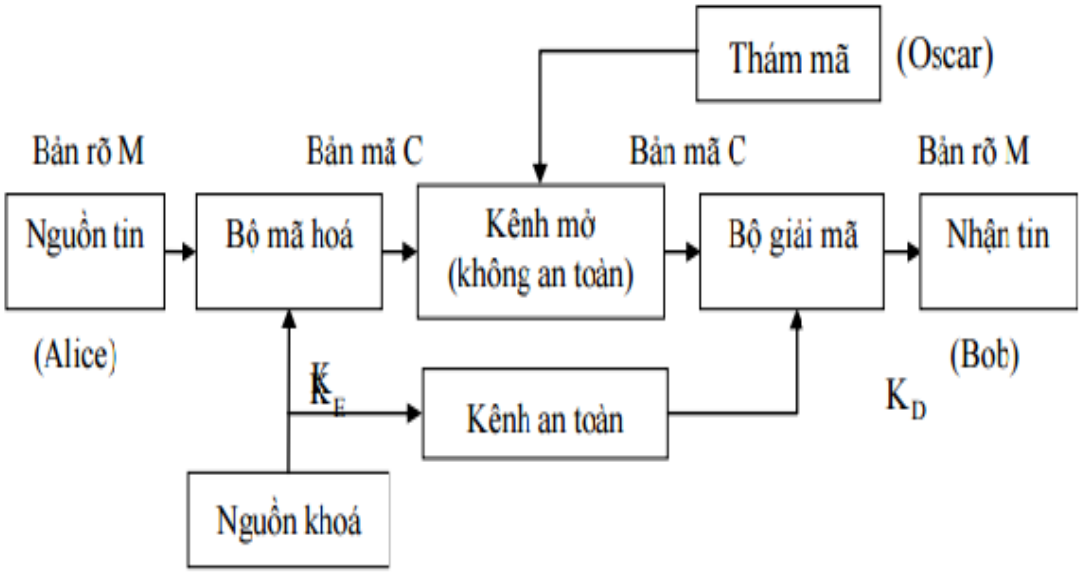
Cấu trúc toán học của AES được mô tả khá đơn giản. Điều này có thể dẫn tới một số mối nguy hiểm trong tương lai.

Giải thuật AES thực hiện hiệu quả cả bằng phần mềm và phần cứng. Thông thường với những ứng dụng không yêu cầu cao về hiệu năng và tốc độ thì AES được thực hiện ở dạng phần mềm. Với việc thực hiện trên phần mềm, thuật toán AES có thể được viết bằng nhiều ngôn ngữ lập trình phổ biến hiện nay như C/C++, VB.NET, Java, C#... và có thể vận hành trên nhiều hệ điều hành như Windows, Linux… Khi thực hiện trên phần cứng, thuật toán AES hỗ trợ thực hiện trên hai dòng thiết bị: dòng thiết bị thứ

nhất dựa vào một hệ vi xử lý phụ kết hợp với hệ vi xử lý của máy tính, dòng thiết bị thứ hai thường được thiết kế ở dạng thẻ thông minh hoặc các thiết bị giao tiếp thông qua cổng USB.

2.1.3 Hệ mật khóa bí mật[24]

Sơ đồ khối chức năng hệ mật khóa bí mật (xem Hình 2.6).



Hình 2.6. Sơ đồ khối chức năng hệ mật khóa bí mật

Một hệ mật là bộ 5 $\{\mathcal{P}, \mathcal{K}, \mathcal{C}, \mathcal{E}, \mathcal{D}\}$ thỏa mãn các điều kiện sau:

$\mathcal{P}$ Là tập hữu hạn các bản rõ có thể

$\mathcal{K}$ Là tập hữu hạn các khóa có thể

$\mathcal{C}$ Là tập hữu hạn các bản mã có thể

Đối với mỗi $K \in \mathcal{K}$, ta có một quy tắc mã hóa $e_K \in \mathcal{E}$ và một quy tắc giải mã tương ứng $d_K \in \mathcal{D}$. Ta định nghĩa $e_K: \mathcal{P} \to \mathcal{C}, d_K: \mathcal{C} \to \mathcal{P}$ sao cho: $d_K(e_K(x)) = x$ với mọi $x \in \mathcal{P}$. Sự xuất hiện của bản rõ thuộc $\mathcal{P}$ là một biến ngẫu nhiên, ký hiệu là X; sự xuất hiện của khóa thuộc $\mathcal{K}$ cũng là một biến ngẫu nhiên, ký hiệu là K. Khi đó bản rõ $x \in \mathcal{P}$ có xác suất tiên nghiệm là $\Pr\{X = x\}$, xác suất để khóa k cụ thể được dùng là $\Pr\{K = k\}$. Ký hiệu sự xuất hiện của bản mã thuộc C là biến ngẫu nhiên Y và ta có thể dễ dàng tính được $\Pr\{Y = y\}$ dựa vào phân phối của không gian các bản rõ $\mathcal{P}$, không gian các khóa $\mathcal{K}$.

$$\Pr\{Y = y\} = \sum_{\{k:y\in C(k)\}} \Pr[K = k] \Pr[x = d_k(y)] \quad (1)$$

Ở đây khóa $k \in \mathcal{K}$, ta xác định $C(k)$ như sau:

$$C(k) = \{e_k(x) : \forall x \in \mathcal{P}\} \quad (2)$$

Ngoài ra, với mỗi $x \in \mathcal{P}$ và $y \in C$ ta có thể tính được xác suất có điều kiện $\{Y = y\}$ khi đã biết $\{X = x\}$:

$\Pr[Y = y/X = x] = \sum_{\{k:x=d_k(y)\}} \Pr[K = k]$ (3)

Chúng ta cũng có thể tính được xác suất có điều kiện $\Pr\{X = x/Y = y\}$:

$$\Pr[X = x/Y = y] = \frac{\Pr[X=x] x \sum_{\{k:x=d_k(y)\}} \Pr[K=k]}{\sum_{\{k:y\in C(k)\}} \Pr[K=k] \Pr[x=d_k(y)]} \text{ (4)}$$

Ví dụ 2.7: Giả sử $\mathcal{P} = \{0,1\}$ với Pr[0] = 1/8, Pr[1] = 7/8

Giả sử $\mathcal{K} = \{k_1, k, k_3\}$ với Pr[k_1] = 1/4, Pr[k_2] = 1/3, Pr[k_3] = 5/12. C = {1, 2, 3, 4} và hàm mã hóa xác định như sau: $e_{k_1}(0) = 1$, $e_{k_1}(1) = 2$, $e_{k_2}(0) = 2$, $e_{k_2}(1) = 3$, $e_{k_3}(0) = 3$, $e_{k_3}(1) = 4$.

Khi đó ta tính được phân phối xác suất như sau:

Pr[Y = 1] = 1/32, Pr[Y = 2] = 25/96, Pr[Y = 3] = 33/96, Pr[y = 4] = 35/96.

Và

Pr[X = 0/Y = 1] = 1, Pr[X = 1/Y = 1] = 0, Pr[X = 0/Y = 2] = 4/25,

Pr[X = 1/Y = 2] = 21/25, Pr[X = 0/Y = 3] = 5/33, Pr[X = 1/Y = 3] = 28/35

Pr[X = 0/Y = 4] = 0, Pr[X = 1/Y = 4] = 1

2.1.4 Hệ mật an toàn[1]

Hệ mật $\{\mathcal{P}, \mathcal{K}, \mathcal{C}, \mathcal{E}, \mathcal{D}\}$ được gọi là hệ mật an toàn nếu với mọi $x \in \mathcal{P}$ và mọi $y \in C$ ta luôn có Pr[X = x/Y = y] = Pr[X = x], nghĩa là xác suất hậu nghiệm của bản rõ x khi đã biết bản mã y cũng bằng xác suất tiên nghiệm của x.

Định lý: Hệ mật $\{\mathcal{P}, \mathcal{K}, \mathcal{C}, \mathcal{E}, \mathcal{D}\}$ được gọi là hệ mật an toàn nếu $|\mathcal{K}| = |C|$, các khóa thuộc $\mathcal{K}$ được dùng đồng xác suất $1/|\mathcal{K}|$ và với mỗi cặp $x \in \mathcal{P}$ và $y \in C$ tồn tại đúng 1 khóa $k \in \mathcal{K}$ thỏa mãn $e_k(x) = y$ và $x = d_k(y)$.

Chứng minh.

Ta dễ dàng tính được $\Pr[Y = y] = 1/|\mathcal{K}|$, do đó mỗi $x \in \mathcal{P}$ đều có duy nhất một $k \in \mathcal{K}$ để $e_k(x) = y$ và $x = d_k(y)$ theo công thức (1).

Tiếp theo, với mọi cặp x, y ta đều có Pr[X =x/Y = y] = Pr[X = x] theo công thức (4).

Ví dụ 2.8: Giả sử ta có hệ mật $\{\mathcal{P}, \mathcal{K}, \mathcal{C}, \mathcal{E}, \mathcal{D}\}$, ở đây $\mathcal{P} = \{1, 2, 3\}$, $\mathcal{K} = \mathcal{C} = \{1, 2, 3, 4\}$. Xác suất của bản rõ lần lượt là $p_1 > 0$, $p_2 > 0$, $p_3 > 0$ với $p_1 + p_2 + p_3 = 1$.

Các khóa trong $\mathcal{K}$ được dùng với xác suất như nhau và bằng 1/4.

Phép mã hóa và giải mã được thực hiện trong nhóm cyclic $\mathbb{Z}_5^*$.

$e_k(x) = x * k \pmod 5, d_k(y) = y * k^{-1} \pmod 5$

Chẳng hạn,

$x = 2, k = 4 \rightarrow y = 2 * 4 \pmod 5 = 3$ và
$d_k(y) = y * k^{-1} \pmod 5 = 3 * 4^{-1} \pmod 5 = 3 * 4 \pmod 5 = 2 = x$

Khi đó, hệ mật đã cho là hệ mật an toàn.

Thực vậy, với mọi $y \in C$ ta luôn có Pr[y] = 1/4, do đó với mỗi $x \in \mathcal{P}$ luôn tồn tại duy nhất một khóa $k_{(y)}^{(x)}$ thỏa mãn $e_{k_{(y)}^{(x)}} = x * k_{(y)}^{(x)} \pmod 5 = y$.

Do $\{y\} = \left\{1, k_{(y)}^{(1)}\right\} + \left\{1, k_{(y)}^{(2)}\right\} + \left\{1, k_{(y)}^{(3)}\right\}$ nên

$$P_c(y) = P_r\left\{X = 1, K = k_{(y)}^{(1)}\right\} + P_r\left\{X = 2, K = k_{(y)}^{(2)}\right\} + P_r\left\{X = 3, K = k_{(y)}^{(3)}\right\}$$

$$= P_r(X = 1) * P_r\left(K = k_{(y)}^{(1)}\right) + P_r(X = 2) * P_r\left(K = k_{(y)}^{(2)}\right) + P_r(X = 3) * P_r\left(K = k_{(y)}^{(3)}\right)$$

$$= p_1 . \frac{1}{4} + p_2 . \frac{1}{4} + p_3 . \frac{1}{4}$$

$$= \frac{1}{4}(p + p_2 + p_3) = \frac{1}{4}$$

Từ đây, với mọi $x \in \mathcal{P}$ và $y \in C$ ta luôn có Pr[X = x/Y = y] = Pr[X = x], nghĩa là hệ mật là an toàn.

2.2 Hệ mật kép an toàn[1]

2.2.1 Mô tả hệ mật kép an toàn

Trước đây người ta chỉ dùng nhóm cyclic để xây dựng hệ mật khóa công khai, ví dụ hệ mật ElGamal, hệ mật trên đường cong Eliptic, nay chúng tôi dùng nhóm cyclic để xây dựng hệ mật mã kép an toàn.

Để sử dụng được nhóm cyclic trong việc xây dựng hệ mật, chúng tôi đã thực hiện một bước trung gian để " định dạng chuẩn" các bản rõ, đưa các bản rõ về thành dãy các phần tử của nhóm cyclic $\mathbf{Z}_p^*$ với tham số p phù hợp. Số nguyên tố p được dùng phải lớn hơn số phần tử của Luật từ điển. Luật từ điển ở đây có 2 phần, một phần là các từ - nhóm từ xuất hiện trong các bản rõ; phần khác là các phần tử của nhóm $\mathbf{Z}_p^*$.

Nhưng Luật từ điển không phải là yếu tố quyết định độ an toàn của hệ mật. Yếu tố quyết định đến sự an toàn của hệ mật là dãy số giả ngẫu nhiên. Dãy số giả ngẫu nhiên cũng không phải được sử dụng trực tiếp mà chỉ được sử dụng sau khi biến đổi thành dãy các phần tử của $\mathbf{Z}_p^*$. Phép mã hóa được tiến hành như là phép nhân theo modulo p giữa 2 phần tử, trong đó 1 phần tử là từ mã theo Luật từ điển ctd, còn phần tử kia là từ khóa (sau khi đã được biến đổi thành phần tử thuộc $\mathbf{Z}_p^*$) k: c=ctd*k(modp). Phép giải mã được tiến hành theo 2 bước, bước 1 khôi phục lại bản mã của Luật từ điển ctd=k^{-1}.c(modp), bước 2 dùng Luật từ điển ngược để giải ra bản rõ ban đầu.[1]

2.2.2 Nhóm cyclic[2]

2.2.2.1 Khái niệm nhóm cyclic

Nhóm (G, *) được gọi là nhóm cyclic nếu nó được sinh ra bởi một trong các phần tử của nó.

Tức là có phần tử g ∈ G mà với mỗi a ∈ G, đều tồn tại số n ∈ N để g^n = g * g * … * g = a. (Chú ý g * g * … * g là g * g với n lần).

Khi đó g được gọi là phần tử sinh hay phần tử nguyên thuỷ của nhóm G.

Nói cách khác: G được gọi là nhóm cyclic nếu tồn tại g ∈ G sao cho mọi phần tử trong G đều là một luỹ thừa nguyên nào đó của g.

Ví dụ 2.9: Nhóm $G = \langle \mathbf{Z}_{19}^*, * \rangle$ là nhóm cyclic với 1 phần tử nguyên thủy (phần tử sinh) g = 2:

$1 \equiv 2^0 \bmod 19$	$7 \equiv 2^6 \bmod 19$	$13 \equiv 2^5 \bmod 19$
$2 \equiv 2^1 \bmod 19$	$8 \equiv 2^3 \bmod 19$	$14 \equiv 2^7 \bmod 19$
$3 \equiv 2^{13} \bmod 19$	$9 \equiv 2^8 \bmod 19$	$15 \equiv 2^{11} \bmod 19$
$4 \equiv 2^2 \bmod 19$	$10 \equiv 2^{17} \bmod 19$	$16 \equiv 2^4 \bmod 19$
$5 \equiv 2^{16} \bmod 19$	$11 \equiv 2^{12} \bmod 19$	$17 \equiv 2^{10} \bmod 19$
$6 \equiv 2^{14} \bmod 19$	$12 \equiv 2^{15} \bmod 19$	$18 \equiv 2^9 \bmod 19$

Khi đó ta có $\log_2 5 (\bmod 19) = 16$. Trong trường hợp tổng quát, biết p, phần tử sinh g và số β thuộc $\mathbf{Z}_p^*$ cần phải tìm $a \equiv \log_g \beta$. Đây là bài toán logarit rời rạc. Người ta đã tính toán được, với p - n bit thời gian tính toán tốt nhất cỡ $2^{O(n^{1/3}\log^{2/3} n)}$.

Phép toán trong $G = \langle \mathbf{Z}_{19}^*, * \rangle$ là phép toán nhân: với x, k thuộc $\mathbf{Z}_{19}^*$ ta có y = x * k ≡ x * k(mod19).

2.2.2.2 Cấp của nhóm cyclic[2]

Cho (G, *) là nhóm cyclic với phần tử sinh g và phần tử trung lập e. Nếu tồn tại số tự nhiên nhỏ nhất n mà g^n = e, thì G sẽ chỉ gồm có n phần tử khác nhau: e, g, g^2, g^3, . . . ,g^{n-1}. Khi đó G được gọi là nhóm cyclic hữu hạn cấp n.

Nếu không tồn tại số tự nhiên n để g^n = e, thì G có cấp ∞.

Ví dụ 2.10: (Z^+, +) gồm các số nguyên dương là cyclic với phần tử sinh g = 1, e = 0.

Đó là nhóm cyclic vô hạn, vì không tồn tại số tự nhiên n để g^n = e

2.2.2.3 Cấp của một phần tử trong nhóm cyclic

Phần tử $\alpha \in G$ được gọi là có cấp d, nếu d là số nguyên dương nhỏ nhất sao cho $\alpha^d = e$, trong đó e là phần tử trung lập của G.

Như vậy phần tử α có cấp 1, nếu $\alpha = e$.

2.2.2.4 Mã hóa xây dựng trên cấp số nhân cyclic[4]

Mỗi cấp số nhân cyclic cấp n có thể coi là một phép biến đổi tuyến tính của vector mã ban đầu.[20]

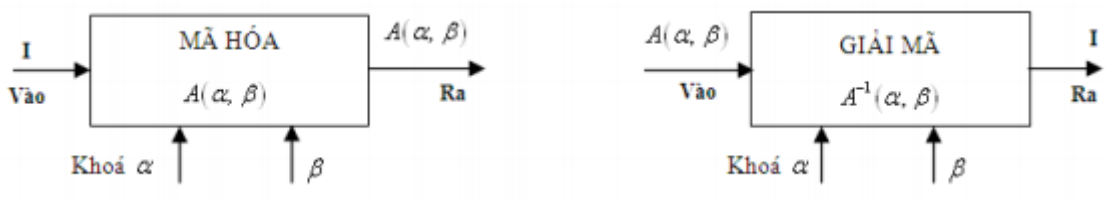
Gọi α là phần tử sinh của một nhóm nhân cyclic cấp n.

Tổng các số hạng của một cấp số nhân cyclic cấp n có công bội α và số hạng đầu β được xác định theo biểu thức sau:

$$S_n = \beta\left[\prod_{i=0}^{k-1}\left(1+\alpha^{2^i}\right)\right]$$

Với $S_n \# 0$

Hệ mật xây dựng trên các cấp số nhân này có thể được mô tả theo sơ đồ khối ….



Mỗi phép biến đổi (mã hoá) A có thể được đặc trưng bởi một ma trận vuông cấp n có dạng sau:

$$A = \begin{bmatrix} \beta.\alpha \\ \beta.\alpha^2 \\ \vdots \\ \beta.\alpha^0 \end{bmatrix}$$

A là một ma trận không suy biến nên luôn tồn tại A^{-1} thoả mãn:

Tập các phép biến đổi này là một tập kín đối với phép tính (nhân ma trận) và tạo nên một nhóm nhân có phần tử đơn vị là phép biến đổi đồng nhất (ma trận đơn vị I).

Nhóm nhân trong vành các ma trận vuông này là nhóm tuyến tính đầy đủ và được ký hiệu là GL(n, GF(2)).

Thuật toán mã hoá khá đơn giản, chỉ dựa trên phép toán nhân và bình phương một đa thức $a(x) \in G$ theo modulo $(x^n + 1)$ (a(x) có cấp n) với một đa thức b(x) bất kỳ $\in G$

2.2.2.5 Giải mã xây dựng trên cấp số nhân cyclic[4][25]

Để giải mã ta phải tìm phép biến đổi ngược A^{-1} là ma trận nghịch đảo của ma trận A. Tuy nhiên ta có thể dễ dàng thực hiện giải mã như sau:

Ma trận A có cấp là n, hoặc là ước của n. Tức là ta luôn có: $A^n = I$.

Ví dụ 2.11:

Xét = 8n chọn đa thức $a(x) = 1 + x + x^2 \leftrightarrow (012)$ làm phần tử sinh, ta có:

A = { (012), (024), 01356, (4), (456), (046), (12457) , (0) }

Ma trận tương ứng: $A = \begin{vmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}$

A^2 = {(014), (2), (236), (4), (045), (6), (267),(0)}

A^3 = {(124), (024), (01235), (4), (046), (046), (14567),(0)} = A^{-1}

A^4 = I = {(1), (2), (3), (4), (5), (6), (7),(0)}

Chú ý: Ở đây ta biểu diễn các đa thức qua các số mũ của các thành phần khác không. Ví dụ: (012435) = $1+ x + x^2 + x^3 + x^5$.

Vào	Mã hoá	Ra	Vào	Giải mã	Ra
I →	A →	A	A →	$(A^2)^2$ =	I

Các ma trận luân hoàn

Khi sử dụng cấp số nhân có công bội x và có số hạng đầu là một đa thức $a(x) \in G$ ta sẽ có một lớp các biến đổi đặc biệt, được đặc trưng bởi một loại ma trận đặc biệt, được gọi là ma trận luân hoàn.

Ma trận vuông A[n x n] trên trường F được gọi là ma trận luân hoàn nếu nó có dạng sau:

$$A = \begin{vmatrix} a(x) \\ xa(x) \\ \dots \\ x^{n-1}a(x) \end{vmatrix} = \begin{vmatrix} a_0 & a_1 & \dots & a_{n-1} \\ a_{n-1} & a_0 & \dots & a_{n-2} \\ \vdots & \vdots & & \vdots \\ a_1 & a_2 & \dots & a_0 \end{vmatrix} \qquad a \in F$$

Đại số các ma trận luân hoàn cấp n trên trường F đẳng cấu với đại số $F[x]/(x^n - 1)$ đối với phép ánh xạ các ma trận luân hoàn thành các đa thức dạng:

$$a(x) = \sum_{i=0}^{n-1} a_i x^i$$

Tổng và tích của hai ma trận luân hoàn là một ma trận luân hoàn.
Ta có: A.B = C

Trong đó: c(x) = a(x).b(x) mod(x^n - 1)

Ma trận luân hoàn A là khả nghịch khi và chỉ khi đa thức a(x) là nguyên tố cùng nhau với (x^n - 1). Ma trận nghịch đảo B nếu tồn tại sẽ tương ứng với đa thức b(x) thoả mãn điều kiện:

$$a(x).b(x) \equiv 1 \bmod(x^{2^k} - 1)$$

Trong trường hợp vành $GF_2[x]/(x^n + 1)$ và $a(x) \in G$, ta luôn có:

$$\left(a(x), (x^{2^k} + 1)\right) = \left(a(x), (x+1)^{2^k}\right) = 1$$

Tập các ma trận luân hoàn A ứng với a(x)∈ G sẽ tạo nên một nhóm con nhân Abel trong nhóm nhân của vành các ma trận vuông. Trong nhóm này tồn tại các nhóm con là các nhóm nhân cyclic có cấp bằng n hoặc ước của n.

Cấp của ma trận luân hoàn A bằng cấp của đa thức a(x) tương ứng của nó. Khi ord (a(x) = 2) thì ma trận luân hoàn A tương ứng là một ma trận tự nghịch đảo.

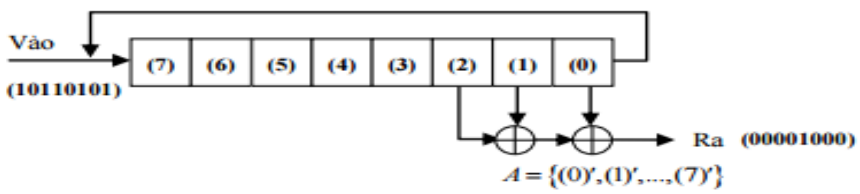
Số các ma trận luân hoàn dùng để lập mã bằng số các phần tử của nhóm nhân trong vành đa thức.

Với trường hợp ma trận luân hoàn, thuật toán mã hoá chỉ là một phép cộng với n bước dịch vòng.

Thuật toán giải mã bao gồm một phép tính nghịch đảo của một đa thức theo modulo (x^n + 1) và n bước dịch vòng tương ứng của phần tử nghịch đảo này.

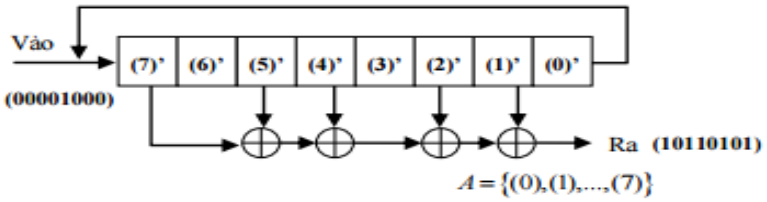
Ví dụ 2.12: Chọn a(x) = 1 + x + x^2

A = {(012), (123), (234), (345), (456), (567), (670), (701)};
A_2 = {(124), (135), (246), (357), (460), (571), (602), (713)};
A_3 = {(01356), (12467), (23570), (34601), (45712), (56023), (67134), (70245)};
A_4 = {(4), (5), (6), (7), (0), (1), (2), (3)};
A_5 = {(456), (567), (670), (701), (012), (123), (234), (345)};
A_6 = {(460), (571), (602), (713), (024), (135), (246), (357)};
A_7 = {(12457), (23560), (34671), (45702), (56031), (67124), (70235), (01346)} = A^{-1};
A_8 = {(1), (2), (3), (4), (5), (6), (7), (0)} = 1;
Sơ đồ thiết bị mã hóa (xem hình 2.7)



Hình 2.7. Sơ đồ thiết bị mã hóa

Sơ đồ thiết bị giải mã (xem hình 2.8)



Hình 2.8. Sơ đồ thiết bị giải mã

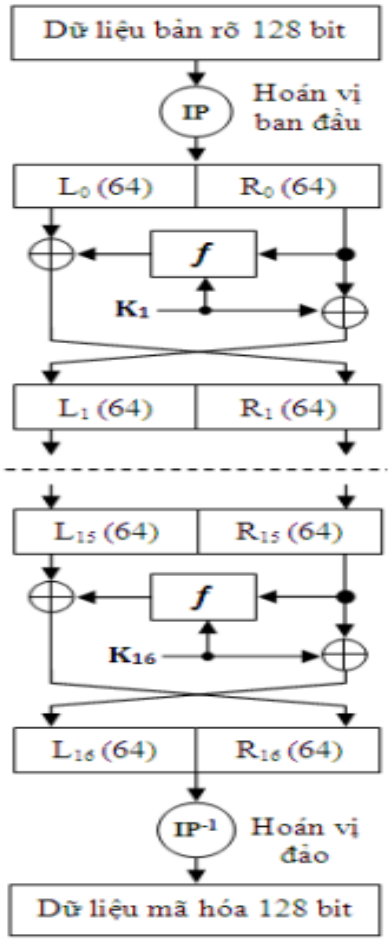
Đa thức nghịch đảo: $a^{-1}(x) = x + x^2 + x^4 + x^5 + x^7$. Ta có:

$$A.A^{-1} = \begin{vmatrix} 1&1&1&0&0&0&0&0\\ 0&1&1&1&0&0&0&0\\ 0&0&1&1&1&0&0&0\\ 0&0&0&1&1&1&0&0\\ 0&0&0&0&1&1&1&0\\ 0&0&0&0&0&1&1&1\\ 0&0&0&0&0&0&1&1 \end{vmatrix} \times \begin{vmatrix} 0&1&1&0&1&1&0&1\\ 1&0&1&1&0&1&1&0\\ 0&1&0&1&1&0&1&1\\ 1&0&1&0&1&1&0&1\\ 1&1&0&1&0&1&1&0\\ 0&1&1&0&1&0&1&1\\ 1&0&1&1&0&1&0&1 \end{vmatrix} = \begin{vmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0 \end{vmatrix} = I$$

2.2.2.6 Xây dựng hệ mật dùng cấp số nhân cyclic[5]

Trong sơ đồ xây dựng hệ mật, sơ đồ Feistel được sử dụng làm nền cho hệ mật dùng các cấp số nhân cyclic trên vành đa thức.

Sơ đồ này (xem hình 2.9) sẽ mã hóa cho chuỗi bit có độ dài 128, các hoán vị IP và hoán vị đảo IP^{-1} được tác giả xây dựng và phát triển từ các bảng IP của hệ mật DES (nêu trong bảng 2.7 và bảng 2.8).



Hình 2.9. Sơ đồ mã hóa khối E

Hàm f được xây dựng trên cơ sở hệ mật sử dụng các cấp số nhân cyclic trên vành đa thức có hai lớp kề. Các khóa K_i là các phần tử trong một cấp số nhân được chọn như sau:

$$K_i = K_a K^i{}_0 \bmod x^{61} + 1$$

Với K_a là một phần tử nguyên thủy của nhóm nhân cyclic có cấp bằng $2^{60} - 1$ và cũng là một đa thức có trọng số lẻ. Cần chú ý rằng với n = 61 vành $Z_2[x]/x^6 + 1$ là một vành có hai lớp kề cyclic.

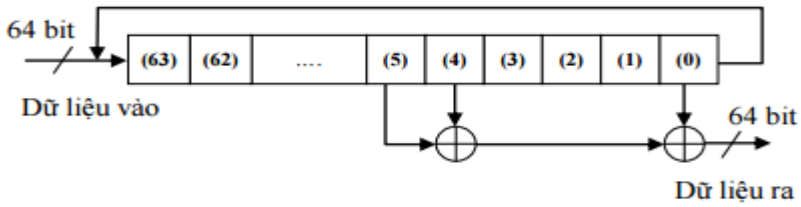
Bảng 2.7. Bảng hoán vị ban đầu (IP)

122	114	106	98	90	82	74	66	58	50	42	34	26	18	10	2
124	116	108	100	92	84	76	68	60	52	44	36	28	20	12	4
126	118	110	102	94	86	78	70	62	54	46	38	30	22	14	6
128	120	112	104	96	88	80	72	64	56	48	40	32	24	16	8
121	113	105	97	89	81	73	65	57	49	41	33	25	17	9	1
123	115	107	99	91	83	75	67	59	51	43	35	27	19	11	3
125	117	109	101	93	85	77	69	61	53	45	37	29	21	13	5
127	119	111	103	95	87	79	71	63	55	47	39	31	23	15	7

Bảng 2.8. Bảng hoán vị đảo (IP^{-1})

80	16	96	32	112	48	128	64
79	15	95	31	111	47	127	63
78	14	94	30	110	46	126	62
77	13	93	29	109	45	125	61
76	12	92	28	108	44	124	60
75	11	91	27	107	43	123	59
74	10	90	26	106	42	122	58
73	9	89	25	105	41	121	57
72	8	88	24	104	40	120	56
71	7	87	23	103	39	119	55
70	6	86	22	102	38	118	54
69	5	85	21	101	37	117	53
68	4	84	20	100	36	116	52
67	3	83	19	99	35	115	51
66	2	82	18	98	34	114	50
65	1	81	17	97	33	113	49

Giả sử ta chọn khóa $K_0 = 1 + x + x^2 \leftrightarrow 1$ (013), phần tử đầu $K_a = 1 + x + x^2 \leftrightarrow 1$ (012), Phần tử đầu của cấp số nhân và cũng là khóa đầu tiên là: $K_1 = K_0.K_a = 1 + x^4 + x^5 \leftrightarrow$(045). Sơ đồ khối bộ mã hóa f với khóa K_1 (xem hình 2.10).



Hình 2.10. Sơ đồ khối mã hóa f, với khóa $K_1 = 1 + x^4 + x^5$

Một khâu mã hóa được thực hiện theo quy tắc:

$$\begin{cases} L_i = R_{i-1} \oplus K_i \\ R_i = L_{i-1} \oplus f\left(R_{i-1}, K_i\right) \end{cases}$$

Trong sơ đồ mã hóa hình k tác giả có thay đổi so với sơ đồ Feistel, đó là nửa trái ở bước thứ i bằng nửa phải ở bước thứ i - 1 cộng với khóa K_i (chỉ cộng 61 bit đầu tiên với khóa). Sở dĩ phải làm như vậy để tránh trường hợp khi các bit dữ liệu đầu vào toàn là bit "0" thì dữ liệu mã hóa đầu ra cũng sẽ toàn là bit "0", vì hàm f trong sơ đồ của chúng tôi thực hiện cộng các bit dữ liệu chứ không cộng với các bit của khóa

Bảng dưới là kết quả tính toán phân bố của bộ mã khi thay đổi 32 bản tin rõ, mỗi bản tin chỉ khác 1 bit, với cùng một bộ khóa K. Với phần tử sinh của khóa $K = 1 + x + x^3 + x^7 + x^9$, phần tử đầu $K = 1 + x + x^2$; phần tử đầu của cấp số nhân (khóa đầu tiên) là:

$$K_1 = K.K_a = 1 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11}$$

Bản tin rõ đầu tiên gồm 32 ký tự dạng hexa (tương ứng 128 bit) là:

M1 = 0123456789ABCDEF0123456789ABCDEF

Bảng 2.9 Khoảng cách Hamming $d_H(C_1,C_i)$ giữa các cặp bản mã

TT	Bản rõ M_i	Bản mã C_i	$d_H(C_1,C_i)$
1	0123456789ABCDEF0123456789ABCDEF	E9D8211132A6A374BC286082EFA45DA8	0
2	2123456789ABCDEF0123456789ABCDEF	3FEDC41619D4B600C059740D7A7D3DB6	63
3	0023456789ABCDEF0123456789ABCDEF	BA63820AED3E453E46236D0BBCE621CB	66
4	0133456789ABCDEF0123456789ABCDEF	64ED9A2B835B2A1A1887D0527791796F	66
5	0122456789ABCDEF0123456789ABCDEF	318B9AB22979JB92F6D26B8F66F71FD4	66
6	0123656789ABCDEF0123456789ABCDEF	F15FF724D7A18806A95C1CF3FB2BC871	63
7	0123446789ABCDEF0123456789ABCDEF	F60072AA91BD7CEC5A629A89E22D0EEA	66
8	0123457789ABCDEF0123456789ABCDEF	E029AC24899C12893546C42D5F74C59D	66
9	0123456389ABCDEF0123456789ABCDEF	ABA4425CDC28CF0BDEB34969C3907BE5	66
10	0123456789ABCDEF0123456789ABCDEF	DCFCE627E6484BB24B0ED9105:661ECA	66
11	0123456789ABCDEF0123456789ABCDEF	BA9A5D727F482D18C34AFBAB0488698E	66
12	0123456789BBCDEF0123456789ABCDEF	CF9528E0BF93184E0DD5E9EC4B0BED78	66
13	0123456789A9CDEF0123456789ABCDEF	D78E46904ACBF02ACC9A47D3A8634AE9	63
14	0123456789ABEDEF0123456789ABCDEF	EC3B44672A217541592F4BF0FAD021D9	63
15	0123456789ABCCEF0123456789ABCDEF	AABAF5812D7EF0CF1F33BF1A09EEA7A3	66
16	0123456789ABCDFF0123456789ABCDEF	C5EC075C3B572E410712D17F66CAF907	66
17	0123456789ABCDEB0123456789ABCDEF	E2D5A84270DAC03952A60CFD8D3F7443	66
18	0123456789ABCDEF41234567 89ABCDEF	4668F1890782644268C688441882E43A	66
19	0123456789ABCDEF052345678 9ABCDEF	13D32C9861E4DF17F1C6EEEE90C6C681	66
20	0123456789ABCDEF0103456789ABCDEF	F4C77D14D1C3D56C3BFE5567E88F2FBD	63
21	0123456789ABCDEF0121456789ABCDEF	3829E4410CF0C4F5C44533DC9F167AF9	63
22	0123456789ABCDEF0123556789ABCDEF	72F1CA3D0680EE7D4DA55539D515A021	66
23	0123456789ABCDEF0123476789ABCDEF	BD09D0D46298F5133D500DD1B:D4EF8F	63
24	0123456789ABCDEF0123452789ABCDEF	60B685BE82763B4198EF56560:4C9B5F	66
25	0123456789ABCDEF0123456F89ABCDEF	CE8966D625E75B2D212E8137A2DD9C62	63
26	0123456789ABCDEF0123456799ABCDEF	96BABA38D98A9752F121910FDA1F6719	66
27	0123456789ABCDEF0123456788ABCDEF	1EFE98838C64E01668B87F5ABC1FFEB3	66
28	0123456789ABCDEF0123456789EBCDEF	5825A87F960913A4241D4445D970B340	66
29	0123456789ABCDEF0123456789AACDEF	2F2F07A8A0186137DEFCF09D37F7E60B	66
30	0123456789ABCDEF0123456789ABBDEF	D369DC985C020CC46CB055A628928946	66
31	0123456789ABCDEF0123456789ABC5EF	B7A8933663E16463FDD0391FE945E8E5	63
32	0123456789ABCDEF0123456789ABCDFF	07564D6E503D8A9F901C46CFE655D09D	66
33	0123456789ABCDEF0123456789ABCDEE	0730E7E6141F31CA7E6B02567FBB85FB	66

Khoảng cách Hamming $d_H(C_1,C_i)$ giữa các cặp bản mã khi các bản rõ khác nhau 1 bit, $d_H(M_1,M_i) = 1$, với cùng một khóa K (Bảng 2.9) và trong bảng những ký tự hexa in đậm chứa bit thay đổi

Khoảng cách Hamming trung bình giữa các bản mã là:

$$d_{H(tb)} = \frac{1}{32}\sum_{i=2}^{33} d_H(C_1,C_i) = 65,16$$

Bảng dưới là kết quả tính toán phân bố của bộ mã khi thay đổi khóa K, mỗi khóa khác với khóa đầu tiên 2 bit, với cùng một bản rõ[5]

M = 0123456789ABCDEF0123456789ABCDEF

Sở dĩ ta phải thay đổi 2 bit (tương ứng thay đổi 2 vị trí) là để đảm bảo đa thức sinh của khóa có trọng số lẻ.

Chú ý 1: Chiều dài của khóa là 61 bit, do đó khi mô tả khóa bằng 16 ký tự hexa nhưng thực tế chỉ có 15 ký tự đầu là dạng hexa, còn ký tự cuối cùng chỉ có 1 bit nên nó nhận giá trị "1" hoặc "0".

Chọn phần tử đầu của cấp số nhân tạo khóa vẫn là $K_a = 1 + x + x^2$ Phần tử sinh khóa đầu tiên K_1 có trọng số $W(K_1) = 33$ như sau:

$K_{1(BIN)}$ = 1000.0100.1100.0010.1010.0110.1110.0001.1001.0101.1101.0011.1011.0111.1111. **1**

$K_{1(HEX)}$ = 1 2 3 4 5 6 7 8 9 A B C D E F Bit 61

Các khóa K_i chỉ thay đổi 2 bit trong một số hexa của khóa đầu tiên K_1.
Chú ý 2: vị trí các bit "1" trong các khóa K_i tương ứng là số mũ của x trong đa thức sinh tạo khóa.

Ví dụ 2.13:

$K_1 = (12\ldots F.1)_{Hex} <-> (1000.0100\ldots1111.1)_{Bin}$

$<-> 1 + x^5 + \ldots + x^{56} + x^{57} + x^{58} + x^{59} + x^{60}$

Bảng 2.10. Khoảng cách Hamming $d_H(C_1,C_i)$ giữa các cặp bản mã

TT	Khóa K_i	Bản mã C_i	$d_H(C_1,C_i)$	$W(K_i)$
1	123456789ABCDEF1	B5733992DC61C07BD38B6196D2434300	0	33
2	**2**23456789ABCDEF1	61E6E5A5C93122EFB5D6D4A81DED9184	66	33
3	1**1**3456789ABCDEF1	CCDE27DCCFF1C9853E4996E02A1619A9	70	33
4	12**5**456789ABCDEF1	ADDC3348559C4D7101D62EB489A5F66F	68	33
5	123**7**56789ABCDEF1	DA18C6B04FFA4F97C0412EAB528B5C78	70	35
6	123**4**36789ABCDEF1	77F9E155A67092ABF8F7A5F66D0827EB	62	33
7	12345**5**789ABCDEF1	C4014F02F5A63F6E8C2C98855D2A34EA	74	33
8	123456**1**89ABCDEF1	15629261C809D76B6CD525B168622F21	56	31
9	12345674**9**ABCDEF1	68E00F4AB5A9061FD7822787633024AC	58	33
10	123456780ABCDEF1	176D974F14841A053D9B5B79E607A4AA	70	31
11	1234567896BCDEF1	398840062A8D0BD055EB5B961AEB0691	60	33
12	123456789A7CDEF1	93CFF32DD509BDE42409312E8818E3FF	70	33
13	123456789AB**A**DEF1	56A632C099DF4A1AF8C9AD2A928845B1	58	33
14	123456789ABC**1**EF1	2180C04805CF8DEAA539B3BF86BAD257	70	31
15	123456789ABCD**2**F1	83390D707279E0F64EE8C762334DD474	60	31
16	123456789ABCDE**C**1	AF304D735EDF45A4397E7CF3832B8346	62	31

Khoảng cách Hamming $d_H(C_1,C_i)$ giữa các cặp bản mã khi các khóa khác khóa K_1 2 bit $d_H(K_1,K_i) = 2$ với cùng một bản rõ M(Bảng 3.10) và các ký tự hexa in đậm chứa các bit của khóa đã thay đổi, trọng số các đa thức sinh tạo khóa luôn đảm bảo là lẻ.

Khoảng cách Hamming trung bình giữa các bản mã:

$$d_{H(tb)} = \frac{1}{15}\sum_{i=2}^{16} d_H(C_1,C_i) = 64,93$$

Phép giải mã được thực hiện theo chu trình ngược lại so với sơ đồ mã hóa, các khóa phải được sử dụng theo thứ tự ngược lại. Tức là, nếu các khóa mã hóa cho mỗi vòng là K_1, K_2 ,..., K_{16} thì các khóa giải mã cho vòng tương ứng sẽ là K_{16}, K_{15} ,..., K_1.

2.2.3 Luật từ điển

Luật từ điển ở đây giữ vai trò gán "mặt nạ" cho bản rõ, một việc mà ta thường làm khi tiến hành mã hoá trong hệ mật trên đường cong Eliptic. Với Luật từ điển chúng ta biến đổi bản rõ thành dãy các phần tử của $\mathbf{Z}_p^*$.

Ví dụ 2.14: Chúng ta cần mã hóa bản rõ sau: "Luật từ điển ở đây giữ vai trò gán mặt nạ cho bản rõ".

Giả sử chúng ta có luật từ điển:

Bảng 2.11. Luật từ điển

Từ rõ	Từ mã của luật TĐ
bản rõ	1
cho	2
gán	3
giữ vai trò	4
Luật từ điển	5
mặt nạ	6
ở đây	7

Khi đó bản mã nhận được sau khi sử dụng Luật từ điển là: " 5 7 4 3 6 2 1"

2.2.4 Khóa giả ngẫu nhiên

2.2.4.1 Tạo số giả ngẫu nhiên[11][22]

Để tạo ra khóa sử dụng trong mã hóa, chúng tôi sử dụng thuật toán tạo số giả ngẫu nhiên. Các thuật toán tạo số giả ngẫu nhiên cần phải thỏa mãn 15 bài kiểm tra đã được NIST đề cập đến trên trang web http://www.random.org/

1. Kiểm tra tần số (Frequency Test): Monobit

2. Kiểm tra tần số: Block

3. Kiểm tra chạy thử (Runs Test)

4. Kiểm tra lượt chạy dài nhất của số 1 trong một khối (Test for the Longest Runs ò Ones in a Block)

5. Kiểm tra thứ hạng ma trận nhị phân (Binary Matrix Rank Test)

6. Kiểm tra quang phổ - Chuyển đổi Fourier rời rạc (Discrete Fourier Transform)

7. Kiểm tra đối chiếu mẫu chồng chéo (Overlapping Template Matching Test)

8. Kiểm tra đối chiếu mẫu không chồng chéo (Non-Overlapping TMT)

9. Kiểm tra thống kê toàn bộ (Maurer's Universal Statistical Test)

10. Kiểm tra phức tạp tuyến tính (Linear Complexity Test)

11. Kiểm tra nối tiếp (Serial Test)

12. Kiểm tra xấp xỉ Entropy (Approximate Entrory Test)

13. Kiểm tra tổng tích lũy (Cumulative Sums Test)

14. Kiểm tra độ lệch ngẫu nhiên (Random Excursion Test)

15. Biến thể kiểm tra độ lệch ngẫu nhiên (Random Excursion Varian Test)

2.2.4.2 Tạo các dãy giả ngẫu nhiên

Trong các hệ mật nghiên cứu ở trên, các phần tử liên tiếp của bản rõ đều được mã hoá bằng cùng một khoá k. Tức xâu bản mã y nhận được có dạng:

$$y = y_1y_2 = \ldots = e_k(x_1)e_k(x_2)\ldots$$

Các hệ mật thuộc dạng này thường được gọi là các mã khối. Một quan điểm sử dụng khác là mật mã dòng. Ý tưởng cơ bản ở đây là tạo ra một dòng khoá $z = z_1z_2$ ... và dùng nó để mã hoá một xâu bản rõ $x = x_1x_2$ ... theo quy tắc:

$$y = y_1y_2 = \ldots = e_z^1(x_1)e_z^2(x_2)\ldots$$

Mã dòng hoạt động như sau. Giả sử $k \in K$ là khoá, và $x = x_1x_2$ ... là xâu bản rõ. Hàm f_i được dùng để tạo z_i (z_i là phần tử thứ i của dòng khoá), trong đó f_i là một hàm của khoá k và i –1 là ký tự đầu tiên của bản rõ:

$$Z_i = f_i(k, VDx_1,\ldots,x_{i-1})$$

Phần tử z_i của dòng khoá được dùng để mã x_i tạo ra $y_i = e_i^z(x_i)$.

Bởi vậy;

Để mã hoá xâu bản rõ x_1x_2... ta phải tính liên tiếp z_1, y_1, z_2, y_2...

Việc giải mã xâu bản mã y_1y_2... có thể được thực hiện bằng cách tính liên tiếp z_1, x_1, z_2, x_2, ...

Định nghĩa dưới dạng toán học:

Mật mã dòng là một bộ (P, C, K, L, F, E, D) thoả mãn các điều kiện sau:

- P là một tập hữu hạn các bản rõ có thể.

- C là tập hữu hạn các bản mã có thể.

- K là tập hữu hạn các khoá có thể (không gian khoá)

- F = (f_1f_2 ...) là bộ tạo dòng khoá. Với i ≥1, f_1: K x $P^{i-1} \rightarrow$ L

- Với mỗi z ∈L có một quy tắc mã e_z ∈ Evà một quy tắc giải mã tương ứng d_z ∈ D; e_z: P→ C và d_z: C → P là các hàm thoả mãn $d_z(e_z(x)) = x$ với mọi bản rõ x ∈ P.

Ta có thể coi mã khối là một trường hợp đặc biệt của mã dòng, trong đó dùng khoá không đổi: $Z_i = k$ với mọi i ≥1.

Mã dòng được gọi là đồng bộ nếu dòng khoá không phụ thuộc vào xâu bản rõ, tức là nếu dòng khoá được tạo ra chỉ là hàm của khoá k. Khi đó, ta coi k là một "mầm" để mở rộng thành dòng khoá z_1z_2 ...

Một hệ mã dòng được gọi là tuần hoàn với chu kỳ d nếu $z_{i+d} = z_i$ với mọi số nguyên i ≥1. Mã Vigenere với độ dài từ khoá m có thể coi là mã dòng tuần hoàn với chu kỳ m. Trong trường hợp này, khoá là k = (k_1,…,k_m). Bản thân k sẽ tạo m phần tử đầu tiên của dòng khoá: $z_i = k_i$, 1≤ i≤ m. Sau đó, dòng khoá sẽ tự lặp lại. Nhận thấy rằng, trong mã dòng tương ứng với mật mã Vigenere, các hàm mã và giải mã được dùng giống như các hàm mã và giải mã được dùng trong MDV:

$$E_z(x) = x + z \text{ và } d_z(y) = y – z$$

Các mã dòng thường được mô tả trong các bộ chữ nhị phân tức là P = C = L = Z_2. Trong trường hợp này, các phép toán mã và giải mã là phép cộng theo modulo 2.

$$E_z(x) = x + z \bmod 2 \text{ và } d_z(y) = y – z \bmod 2$$

Ta xem xét một phương pháp tạo một dòng khoá (đồng bộ) khác. Giả sử bắt đầu với (k_1,...,k_m) và $z_i = k_i$, 1≤ i≤m (cũng giống như trước đây), tuy nhiên bây giờ ta tạo dòng khoá theo một quan hệ đệ quy tuyến tính cấp m:

$$z_{i+m} = \sum_{j=0}^{m-1} c_j z_{i+j} \bmod 2$$

trong đó c_0,…,c_{m-1}∈z_2 là các hằng số cho trước.

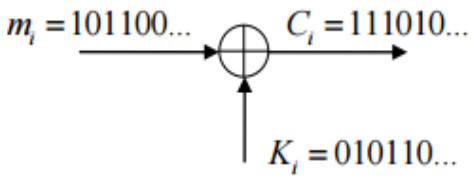
Nhận xét:

Phép đệ quy được nói là có bậc m vì mỗi số hạng phụ thuộc vào m số hạng đứng trước. Phép đệ quy này là tuyến tính bởi vì Z_{i+m} là một hàm tuyến tính của các số hạng đứng trước. Chú ý ta có thể lấy $c_0 = 1$ mà không làm mất tính tổng quát. Trong trường hợp ngược lại, phép đệ quy sẽ là có bậc m - 1.

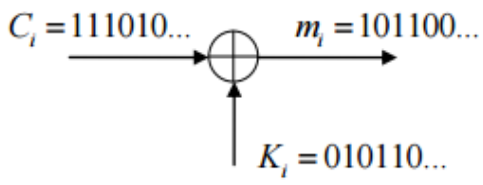
Ở đây khoá k gồm 2m giá trị $k_1,\ldots, k_m, c_0,\ldots, c_{m-1}$. Nếu $(k_1,\ldots, k_m) = (0,...,0)$ thì dòng khoá sẽ chứa toàn các số 0. Dĩ nhiên phải tránh điều này vì khi đó bản mã sẽ đồng nhất với bản rõ. Tuy nhiên, nếu chọn thích hợp các hằng số $c_0,...,c_{m-1}$ thì một vector khởi đầu bất kì khác $(k_1,..., k_m)$ sẽ tạo nên một dòng khoá có chu kỳ $2^m - 1$.

Việc mã hóa và giải mã hệ mật mã dòng mô tả như dưới đây.

+ Mã hóa: $C_i = m_i + k_i \bmod 2$



+ Giải mã: $m_i = C_i + k_i \bmod 2$



Để hệ thống an toàn, dãy bit khóa ngẫu nhiên phải dài hơn bản tin: $|k_i| \geq |m_i|$ (Dãy ngẫu nhiên có p(0) = p(1) = 0,5).

Việc tạo dãy ngẫu nhiên tốn kém và việc lưu trữ không hiệu quả, do đó phải tạo dãy giả ngẫu nhiên.

2.2.4.3 Đánh giá tính ngẫu nhiên của dãy ngẫu nhiên tạo ra

a. Kiểm tra chu kỳ

Bản chất của một chuỗi giả ngẫu nhiên là sự lặp lại một chuỗi giả ngẫu nhiên con có độ dài xác định. Độ dài của chuỗi con này chính là chu kỳ của dãy giả ngẫu nhiên. Độ ngẫu nhiên của một dãy càng được đánh giá cao khi chu kỳ của dãy đó càng lớn, tức là dãy càng khó lặp lại.

Theo những kết quả nghiên cứu mới đây, thuật toán BBS cho dãy nhị phân có chu kỳ tối đa là $\lambda(\lambda(n))$. Trong đó $n = p * q$, với p, q là hai số nguyên tố đủ lớn thỏa mãn $p \equiv q \equiv 3(\mathrm{mod}\ 4)$, và hàm $\lambda(n)$ là hàm Camichael được xác định như sau:

Định lý 1. Với số nguyên $m \geq 1$, hàm Camichael $\lambda(m)$ là bậc lớn nhất có thể của các phần tử trong nhóm nhân theo modulo m: $(\mathbb{Z}_m^*; *)$, được tính theo công thức sau:

$$\lambda(p^k) = \begin{cases} p^{k-1}(p-1), & khi\ p \geq 3\ hoặc\ k \leq 2 \\ 2^{k-2}, & khi\ p = 2\ và\ k \geq 3 \end{cases}$$

và tổng quát $\lambda(m) = lcm\left(\lambda(p_1^{k_1}), \ldots, \lambda(p_v^{k_v})\right)$, trong đó $m = p_1^{k_1} \ldots p_v^{k_v}$.

Ta có thể giới hạn chu kỳ của dãy giả ngẫu nhiên sinh ra bởi bộ sinh BBS thông qua định lý sau.

Định lý 2. Cho bất kỳ số nguyên dương m và với mọi số $K_1, K_2 \geq 1$, chúng ta gọi W biểu diễn số lượng cặp các số nguyên s, e với $1 \leq s \leq m,\ 1 \leq e \leq \lambda(m)$ và $gcd(s,m) = gcd(e, \lambda(m)) = 1$, nhờ đó mà chu kỳ sinh dãy ngẫu nhiên cho bởi (1) là $\lambda(\lambda(m))/K_1K_2$. Khi đó

$$W \leq \varphi(\lambda(m))\left(\frac{\tau(\lambda(m))}{K_1} + \frac{\tau(\lambda(\lambda(m)))}{K_2}\right)$$

Về phần chứng minh cụ thể định lý trên có thể xem chi tiết trang 250 của tài liệu[13].

Trong đó hàm tau $\tau(n)$ được xác định bởi đồng nhất thức sau:

$$\sum_{n \geq 1} \tau(n) q^n = q \prod_{n \geq 1} (1-q^n)^{24} = \eta(z)^{24} = \Delta(z)$$

với $q = \exp(2\pi i z)$, $\Im z > 0$, η là hàm Dedekind eta $\eta(\tau) = e^{\frac{\pi i \tau}{12}} \prod_{n=1}(1-q^n)$, hàm $\Delta(z)$ là dạng thức điểm lùi giải thích trọng số 12 và mức 1 $\Delta(z) = (2\pi)^{12}\eta^{24}(\tau)$.

Bảng 2.12. Một vài giá trị của hàm tau

N	1	2	3	4	5	6
$\tau(n)$	1	-24	252	-1472	4830	-6048
N	7	8	9	10	11	12
$\tau(n)$	-16744	84480	-113643	-115920	534612	-370944

b. Kiểm tra tính ngẫu nhiên

Cho K = $k_0\, k_1 \ldots k_{n-1}$ là dãy nhị phân độ dài n (dãy khóa được sinh ra từ thuật toán trên). Xét các bài kiểm tra tần số lệch dưới đây để xác định xem dãy k có vi phạm một số đặc trưng đặc biệt của dãy ngẫu nhiên thật sự hay không. Tuy nhiên vượt qua các tiêu chuẩn này không khẳng định một cách chắc chắn rằng k là thật sự ngẫu nhiên, mà vẫn có xác suất sai là α.

Việc kiểm tra sự phù hợp của dữ liệu với phân phối lý thuyết dựa theo tiêu chuẩn "Khi bình phương" - ký hiệu χ^2. Đặt $\chi^2 = \sum_{j=1}^{n} \frac{(o_j - e_j)^2}{e_j}$ ở đây o_j là tần số quan sát được; e_j là tần số mong đợi của nhóm thứ j, j = 1, 2, …, n. Với mức ý nghĩa

α = 0.01, và số bậc tự do là n-1, với n = 2^k, ở đây k là độ dài các bộ, nếu $\chi^2 \geq \chi^2_{n-1}(\alpha)$ ta bác bỏ. Ngược lại, nếu $\chi^2 < \chi^2_{n-1}(\alpha)$ thì dãy được chấp nhận là phù hợp.

Kiểm tra tần số lệch đơn: Tiêu chuẩn này dùng để kiểm tra xem các số 0 và 1 có xuất hiện đều nhau hay không.

Kiểm tra bộ đôi móc xích: Tiêu chuẩn này dùng để kiểm tra xem các bộ đôi 00, 01, 10, 11 có xác suất xuất hiện như nhau hay không.

Kiểm tra bộ k: Tiêu chuẩn này dùng để kiểm tra xem các bộ k bits (chỉnh hợp lặp k của các bit 0, 1) có xác suất xuất hiện như nhau hay không.

c. Kết quả thực nghiệm:

Với dãy giả ngẫu nhiên có độ dài N = 194304 (bộ khóa được sinh ra từ hệ thống)

+ Test tần số lệch đơn: P(0) = 0.50180, P(1) = 0.49820; $\chi^2 = 2,518 < \chi^2_1(0,01) = 6,635$ => chấp nhận

+ Bộ đôi móc xích: P(00) = 0.25187, P(01) = 0.24992, P(10) = 0.24992, P(11) = 0.24828; $\chi^2 = 5,027 < \chi^2_3(0,01) = 11,345$ => chấp nhận

+ Bộ ba:

Bảng 2.13. Bộ 3 móc xích

P(000)	0.12670	P(100)	0.12517
P(001)	0.12517	P(101)	0.12457
P(010)	0.12539	P(110)	0.12453
P(011)	0.12453	P(111)	0.12375

$\chi^2 = 8,221 < \chi^2_7(0,01) = 18,475$ => chấp nhận

+ Bộ bốn:

Bảng 2.14. Bộ 4 móc xích

P(0000)	0.06349	P(0100)	0.06245
P(0001)	0.06321	P(0101)	0.06294
P(0010)	0.06279	P(0110)	0.06268
P(0011)	0.06273	P(0111)	0.06186
P(1000)	0.06321	P(1100)	0.06272
P(1001)	0.06196	P(1101)	0.06181
P(1010)	0.06259	P(1110)	0.06186
P(1011)	0.06216	P(1111)	0.06189

$\chi^2 = 13,943 < \chi^2_{15}(0,01) = 30,578$ => chấp nhận

+ Bộ năm:

Bảng 2.15. Bộ 5 móc xích

P(00000)	0.03192	P(01000)	0.03156
P(00001)	0.03156	P(01001)	0.03089
P(00010)	0.03178	P(01010)	0.03155
P(00011)	0.03143	P(01011)	0.03139
P(00100)	0.03156	P(01100)	0.03159
P(00101)	0.03123	P(01101)	0.03109
P(00110)	0.03134	P(01110)	0.03077
P(00111)	0.03103	P(01111)	0.03109
P(10000)	0.03157	P(11000)	0.03165
P(10001)	0.03164	P(11001)	0.03107
P(10010)	0.03101	P(11010)	0.03104
P(10011)	0.03095	P(11011)	0.03077
P(10100)	0.03088	P(11100)	0.03133
P(10101)	0.03171	P(11101)	0.03073
P(10110)	0.03133	P(11110)	0.03109
P(10111)	0.03083	P(11111)	0.03080

$\chi^2 = 22,688 < \chi^2_{31}(0,01) = 52,191$ => chấp nhận

2.2.4.4 Tốc độ thực hiện

Module sinh khóa có thể sinh ra khóa có độ dài lớn (56, 128, 256,… bits) với số lượng lớn một cách nhanh chóng. Dưới đây là kết quả thực nghiệm:

Bảng 2.16. Kết quả thực nghiệm

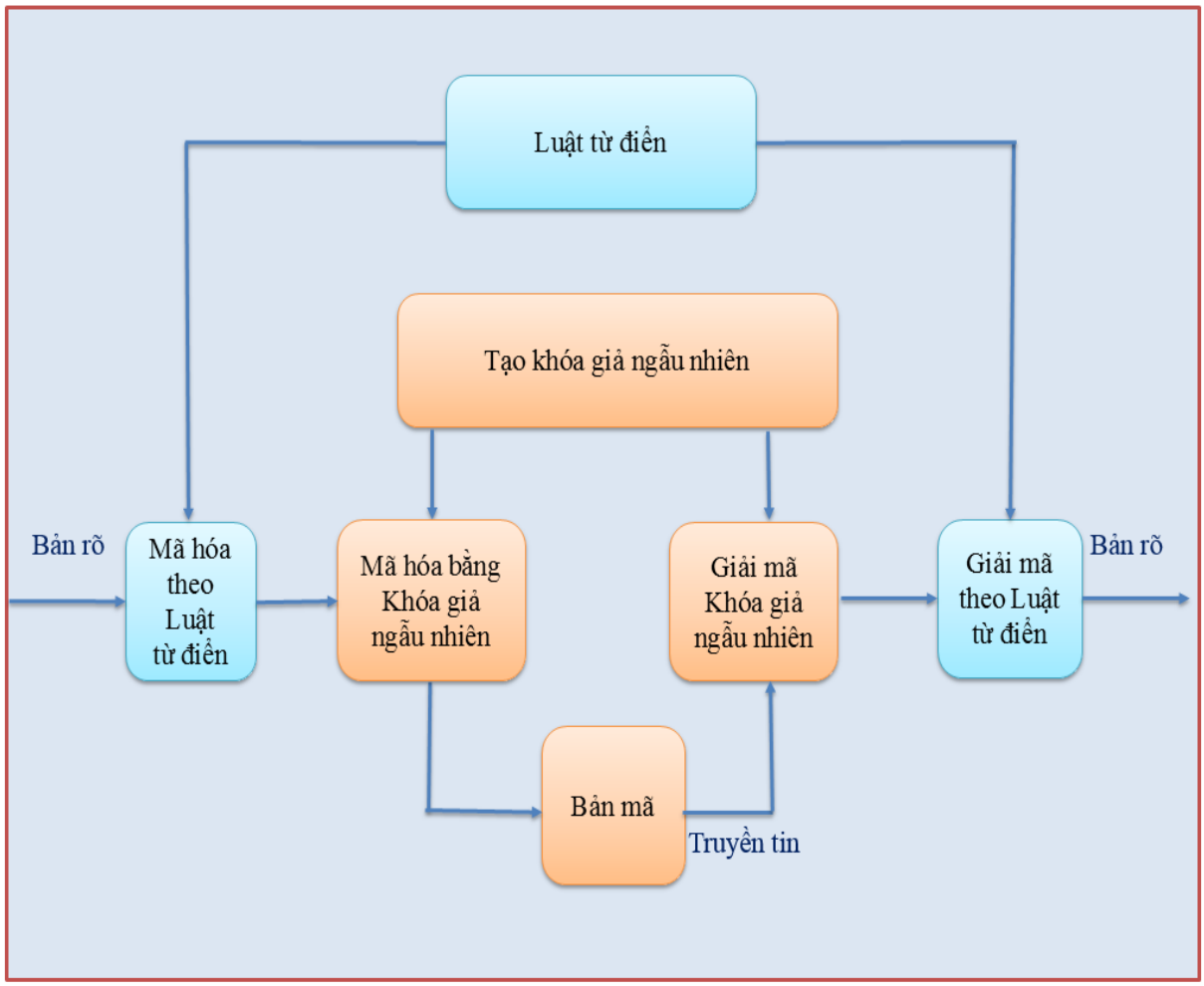
STT	Độ dài mỗi khoá (bits)	Số lượng khóa	Thời gian thực hiện (s)
1	56	1000	2.103
2	56	10000	13.143
3	56	100000	102.958
4	128	1000	2.137
5	128	10000	13.11
6	128	100000	107.636
7	256	10000	15.772
8	256	100000	186.02

CHƯƠNG 3. XÂY DỰNG HỆ MẬT KÉP VÀ ỨNG DỤNG

3.1 Xây dựng hệ mật kép[1][11][12][13][14][15]

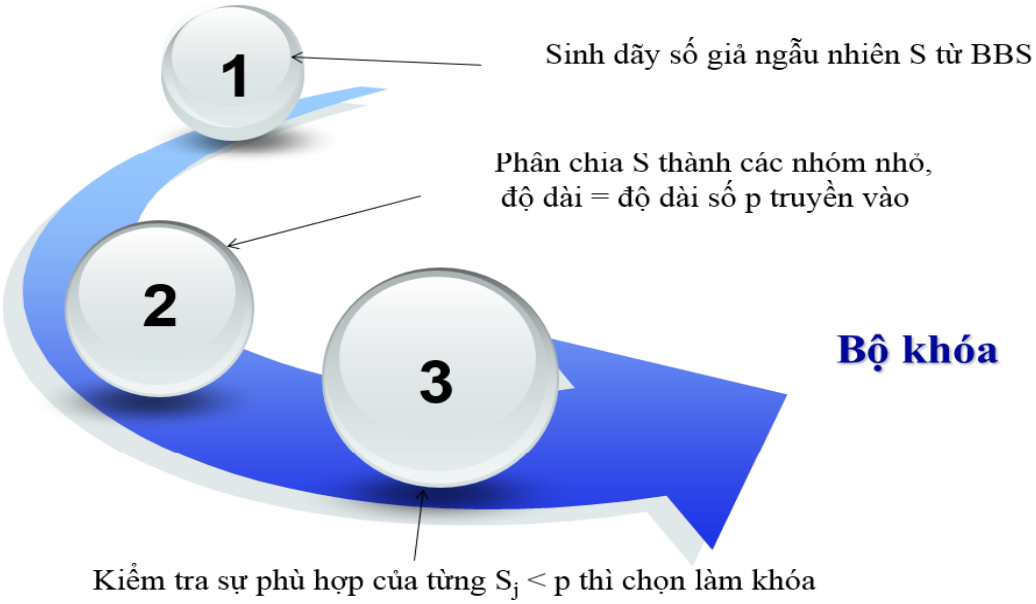
3.1.1 Sơ đồ hệ thống

Với một văn bản đầu vào chương trình sẽ tiến hành mã hóa lần 1 qua luật từ điển thành một chuỗi số dựa theo mã và độ ưu tiên của chúng trong từ điển. Sau khi có được một chuỗi số chương trình sẽ tiến hành mã hóa lần 2. Vậy để có được một bản mã kép chúng ta cần có thêm một quá trình. Chúng ta sẽ "Tạo ra được một bộ khóa đảm bảo tính chính xác và ngẫu nhiên tương ứng với bộ số vừa nhận được". Để tạo được bộ khóa như vậy chúng ta sẽ kết hợp hai phương pháp là phương pháp sinh dãy số giả ngẫu nhiên và phương pháp nhân trong nhóm cyclic. Với các phương pháp sinh dãy số giả ngẫu nhiên chương trình đã tạo ra được những bộ khóa ngẫu nhiên đồng thời kết hợp với phương pháp nhân trong nhóm cyclic với từng khóa trong bộ khóa và chương trình thực hiện mã hóa lần 2 với khóa giả ngẫu nhiên cho từng số trong chuỗi số, cuối cùng thu được một bản mã. Chúng ta có thể gửi bản mã này đi nhưng người nhận phải thực hiện quá trình giải mã lần 1(giải mã khóa giả ngẫu nhiên) và Giải mã lần 2 phải dựa vào luật từ điển để có thể tìm ra được bản rõ ban đầu (xem Hình 3.11).



Hình 3.11. Sơ đồ hệ thống xây dựng hệ mật kép

3.1.3 Sinh khóa ngẫu nhiên[1][15]



Hình 3.12. Sinh khóa ngẫu nhiên

Chương trình sử dụng phương pháp sinh số giả ngẫu nhiên Blum-Blum-Shub (BBS) được đề xuất vào năm 1986 bởi Lenore Blum, Manuel Blum và Michael Shub.

Nội dung của thuật toán được (mô tả trong Hình 3.13) phát biểu như sau:

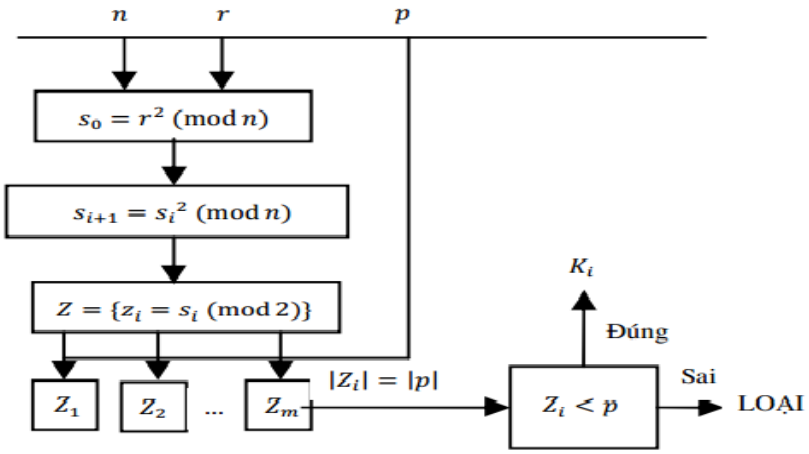
Bước 1: Chọn hai số nguyên tố, p và q thỏa mãn $p \equiv q \equiv 3 \pmod 4$ (điều này đảm bảo rằng mỗi thặng dư bình phương có một căn bậc hai cũng là một thặng dư bình phương) và UCLN (φ (p-1), φ (q-1)) nên nhỏ (điều này làm cho độ dài chu kỳ lớn).

Bước 2: Định nghĩa n = p.q và QR(n) là tập hợp các thặng dư bình phương của n.

Bước 3: Chọn hạt giống $s_0 \in QR(n)$.

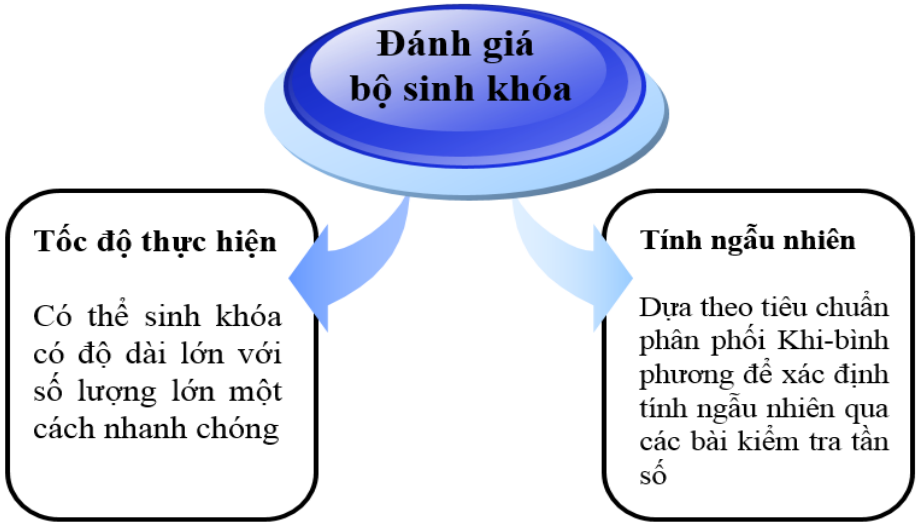
Bước 4: Với $0 \leq i \leq l-1$, xác định $s_{i+1} = s_i^2 \bmod n$ và tập $f(s_0) = (z_1, z_2, \ldots, z_l)$ sao cho $z_i = s_i \bmod 2$

Khi đó tập $f(s_0)$ sẽ là dãy số giả ngẫu nhiên sinh ra từ thuật toán BBS.



Hình 3.13. Thuật toán BBS

Đánh giá bộ sinh khóa



Hình 3.14. Đánh giá bộ sinh khóa

Ví dụ 3.15: Với $n = 192649 = 383 \times 503$ và $s_0 = 1013552 \,(\mathrm{mod}\, n) = 20749$. Dãy 20 bits đầu tiên được sinh ra từ thuật toán BBS: 11001110000100111010

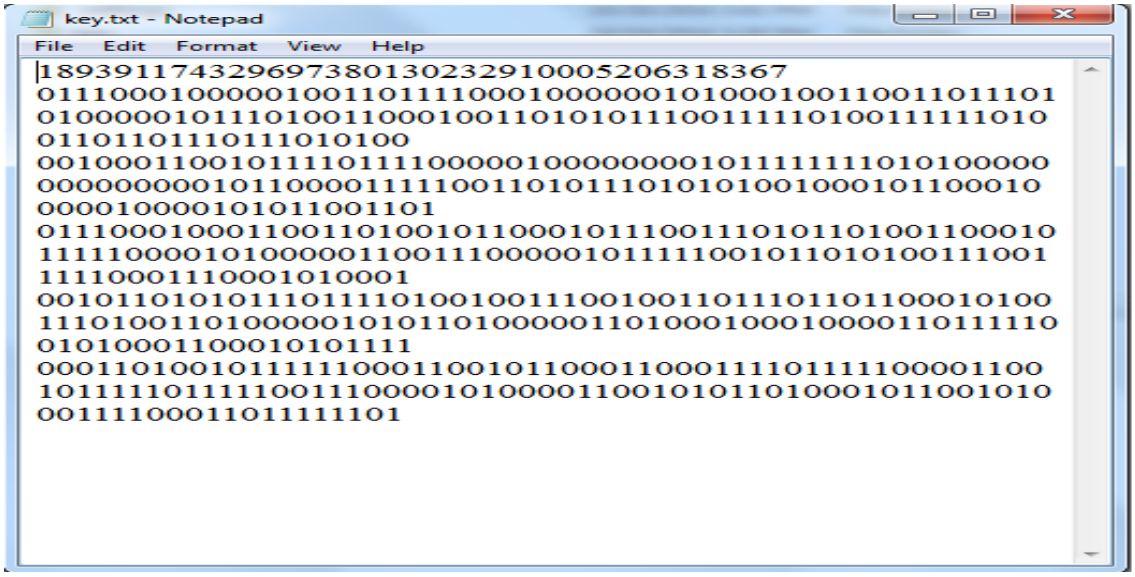
Hệ thống sử dụng thuật toán BBS để tạo ra bộ khóa ngẫu nhiên theo phương pháp sau:

Bước 1: Sinh dãy bit ngẫu nhiên Z từ thuật toán BBS

Bước 2: Phân chia dãy Z thành các nhóm nhỏ. Mỗi nhóm có độ dài bằng độ dài của số nguyên tố p truyền vào $Z = \{Z_1, Z_2, \ldots, Z_m\}$. Loại bỏ những bit thừa.

Bước 3: Kiểm tra sự phù hợp của các Z_i thỏa mãn $Z_i < p$. Mỗi Z_i thỏa mãn sẽ là một khóa ngẫu nhiên được sinh ra thuộc $\mathbb{Z}_p^*$.

Hình 3.15 là kết quả trong quá trình chạy chương trình sinh khóa ngẫu nhiên



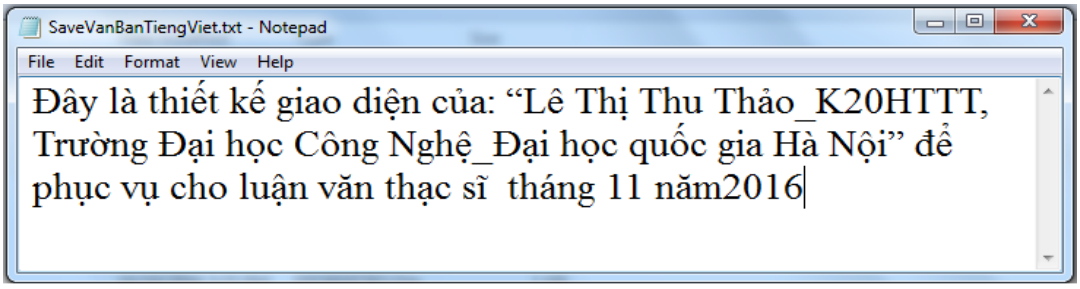
Hình 3.15. Kết quả sinh khóa ngẫu nhiên

3.1.2 Từ điển

Để xây dựng được từ điển như mong muốn cần phải dựa vào luật từ điển và thực hiện các quá trình như thu nhập dữ liệu sau đó lọc tần xuất, gán mã định danh.

3.1.2.1 Thu nhập dữ liệu

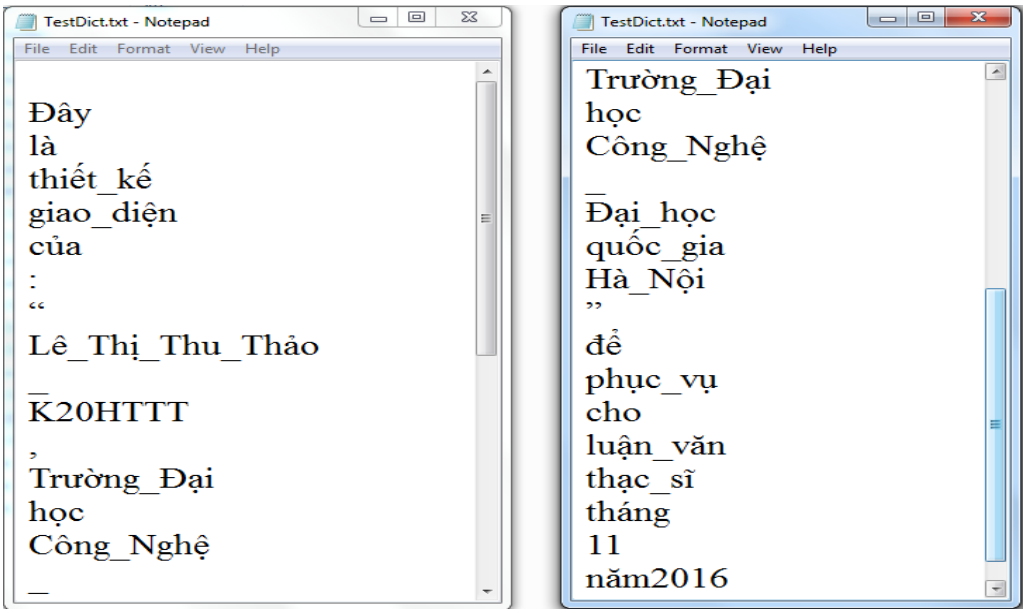
Có nhiều cách thu nhập dữ liệu về để tạo từ điển nhưng tôi muốn lấy dữ liệu là các nội dung trên web để các vốn từ được phong phú, đa dạng và đồng thời có được nhiều từ thông dụng hơn. Trang web tôi sử dụng là trang http://dantri.com.vn với 1000 bài báo về nhiều lĩnh vực như sự kiện, kinh tế, văn hóa, xã hội, giáo dục…Để lấy được nội dung của trang web tôi dùng ngôn ngữ java sử dụng thư viện jsoup, viết tool crawl dữ liệu trang web tự động. Tôi lọc link và chỉ lấy link có bài viết, bỏ qua các link có video, hình ảnh…và tiến hành lấy nội dung sau đó lưu vào file .txt (xem Hình 3.16).



Hình 3.16. Kết quả thu nhập dữ liệu

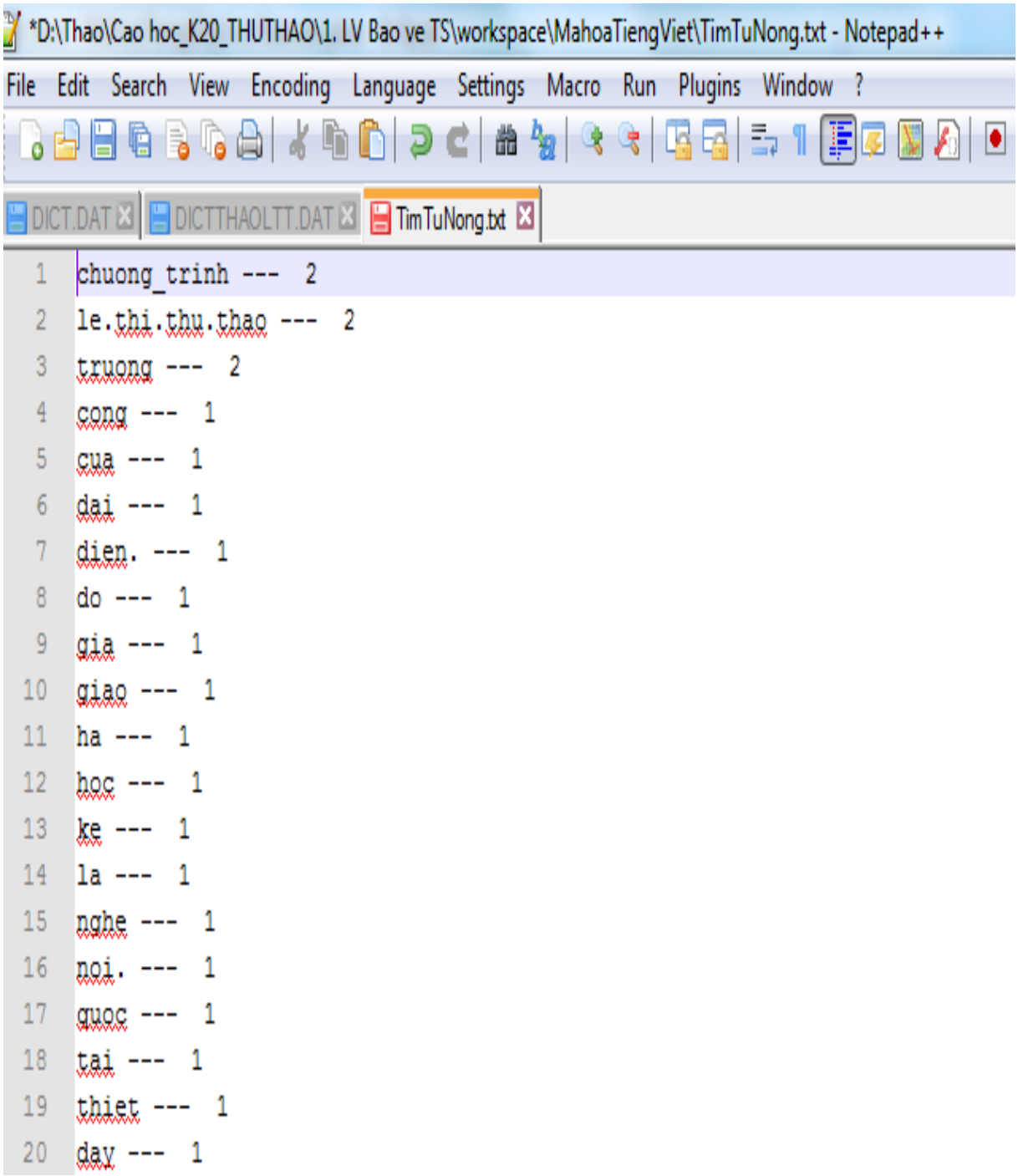
3.1.2.2 Lọc tần suất

Sau khi thu nhập dữ liệu tiếp tục dùng ngôn ngữ java và kết hợp sử dụng bộ thư viện của tác giả Lê Hồng phong, Khoa Toán-Cơ-Tin, Trường đại học khoa học tự nhiên, ĐHQGTPHCM và sử dụng mô hình học máy với phương pháp cực đại Entropy để nhận diện các dấu hiệu hết câu và các luật được viết dưới dạng biểu thức chính quy để tách các câu thành các từ riêng biệt (xem Hình 3.17).



Hình 3.17. Kết quả phân tách

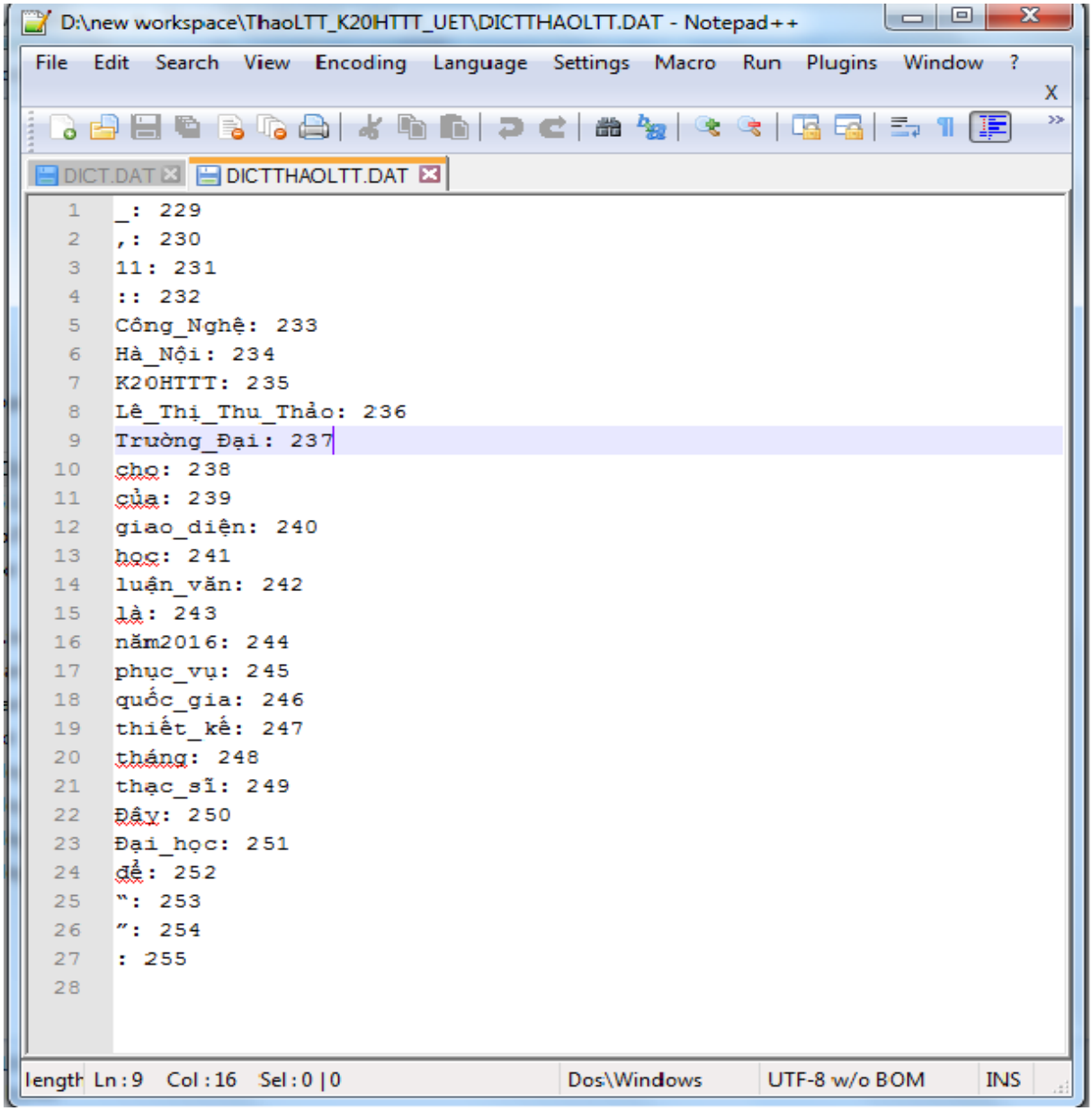
Sau khi phân tách các từ thì việc đếm tần suất các từ sẽ giúp thống kê từ khóa nào đang được sử dụng nhiều nhất và có thể biết được vấn đề nào đang nóng hiện nay, đồng thời dễ dàng cho việc đánh số định danh cho từng từ sau này (xem Hình 3.18).



Hình 3.18. Kết quả từ khóa được sử dụng nhiều nhất

3.1.2.3 Gán mã định danh

Với 229 từ đầu tiên là các ký tự cơ bản trong tiếng Việt: tổ hợp của 29 chữ cái trong bảng chữ cái Tiếng Việt cùng các thanh điệu, bộ số 0-9, các ký tự đặc biệt (!, @, #,…). Từ ID 230 trở đi là các từ có tần suất xuất hiện cao được trích xuất từ dữ liệu mẫu lấy từ trang web (xem Hình 3.19).



Hình 3.19. Gán mã định danh

3.1.2.4 Kết quả

Từ điển xây dựng được lưu trong file có tên là DICT.DAT. Định dạng trong từ điển với mỗi mục từ nằm trên 1 dòng với cú pháp:

<word>:<space><ID>

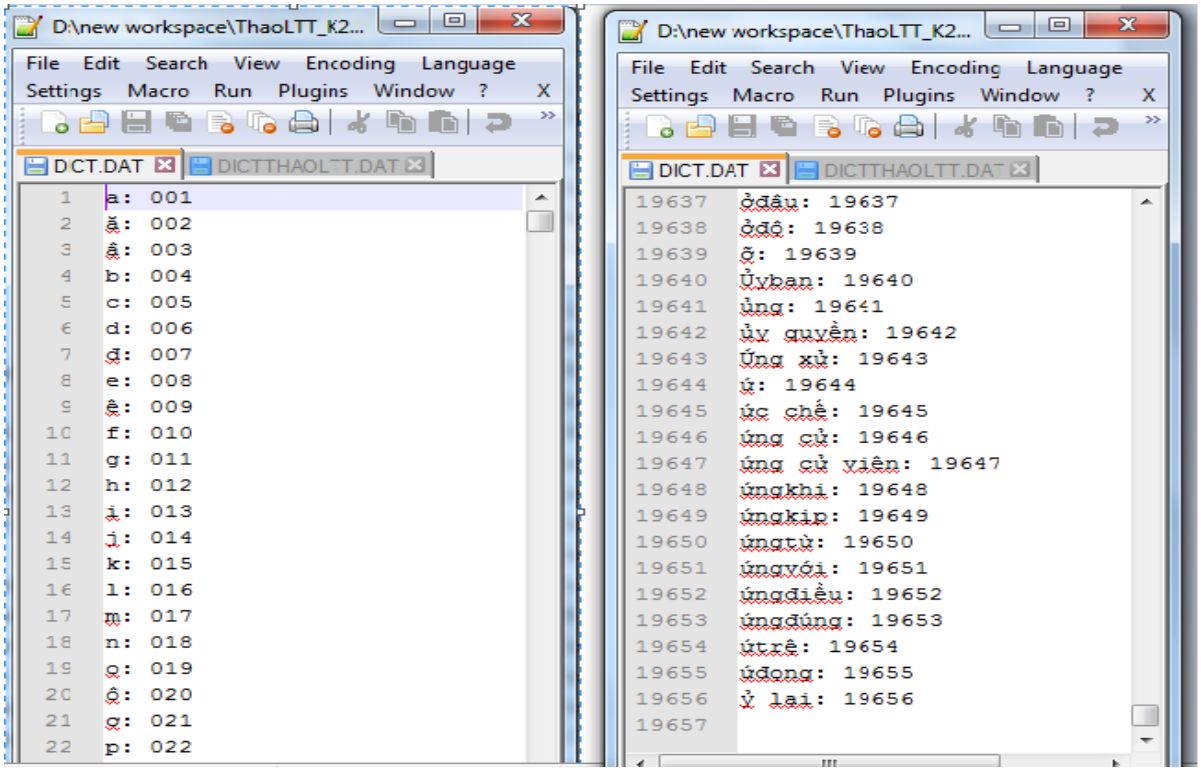
Trong đó:

<word> là một chuỗi kí tự không chứa dấu hai chấm ":"

<space> chỉ bao gồm đúng 01 kí tự space;

<ID> là mã ID cho từ đó

Với nguồn dữ liệu các từ có tần suất xuất hiện cao và các ký tự cơ bản trong tiếng Việt, bộ từ điển đã có gần 20000 từ để sử dụng trong nhiều lĩnh vực (xem Hình 3.20).



Hình 3.20. Kết quả DICT.DAT

3.2 Ứng dụng

3.2.1 Mã hóa kép

3.2.1.1 Mã hóa lần 1 qua từ điển

Đầu vào: Văn bản Tiếng Việt cần mã hóa.

Đầu ra: Bản mã từ điển.

Thực hiện:

Bước 1: Phân tách văn bản đầu vào thành các từ/cụm từ có nghĩa.

Bước 2: Đối chiếu các từ/cụm từ vừa phân tách với bản mã (ID) của nó trong từ điển đưa ra dãy các ID được phân tách nhau bởi dấu cách. Đối với những từ chưa xuất hiện trong từ điển, mã hóa từng ký tự của từ đó theo ID của các ký tự đó trong từ điển. Dãy các ID tạo ra là bản mã từ điển.

3.2.1.2 Mã hóa lần 2 bằng khóa giả ngẫu nhiên

Đầu vào: Bản mã từ điển.

Đầu ra: Bản mã của bản mã từ điển.

Khóa: Dãy khóa được sinh ra từ module sinh khóa. Số lượng khóa tương ứng với số lượng số trong dãy bản mã từ điển.

Thực hiện:

Bước 1

- Tách lấy số đầu tiên trong chuỗi đầu vào: ký hiệu n_i.
- Lấy khóa đầu tiên trong bộ khóa: ký hiệu k_i.
- Thực hiện mã hóa: $c_i = n_i * k_i \pmod p$.
- Thêm c_i vào chuỗi output

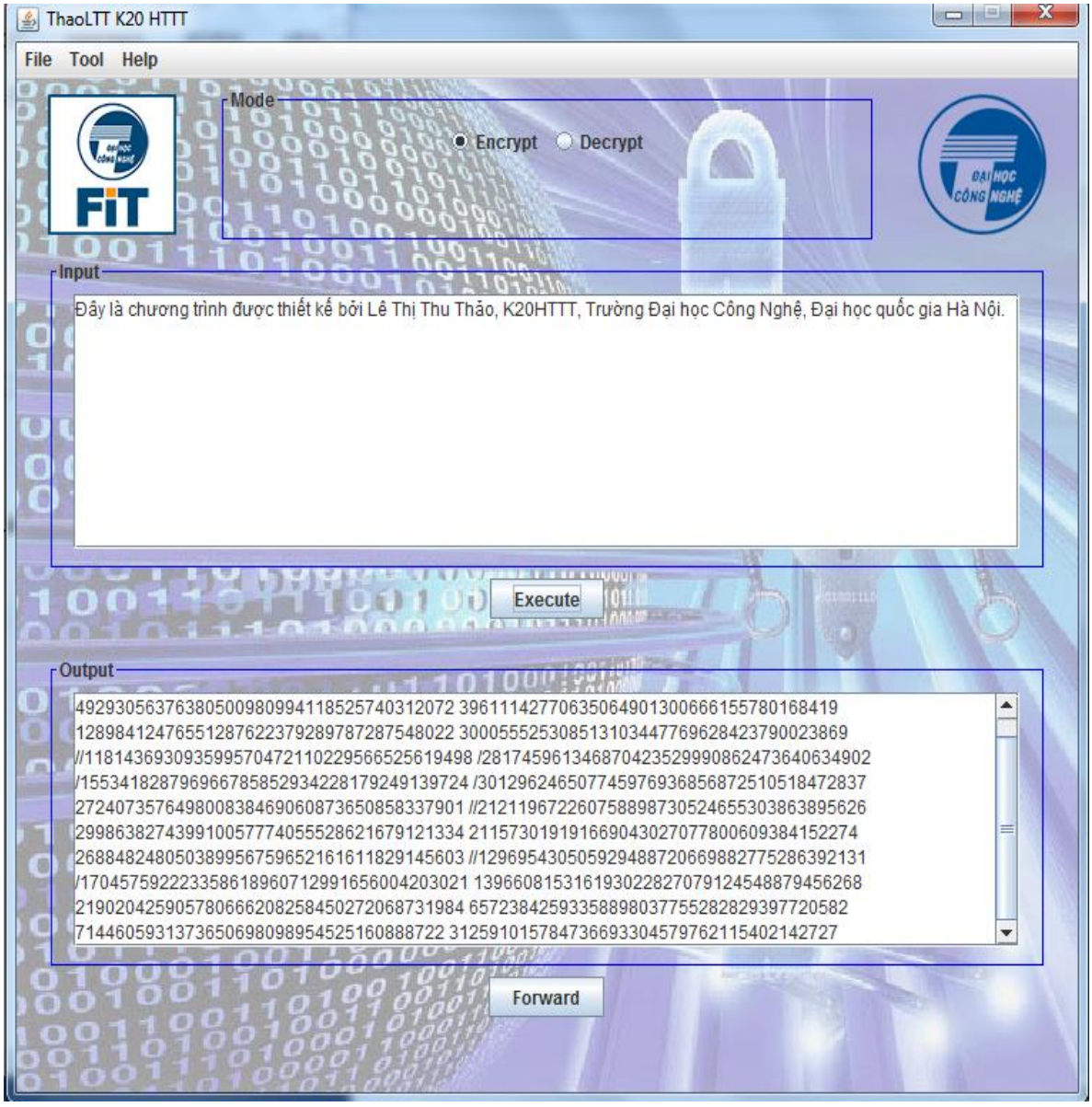
Bước 2

Lặp bước 1 cho số và khóa tương ứng, cho đến cuối chuỗi đầu vào

3.2.1.3 Kết quả mã hóa kép

Giả sử ta muốn mã hóa thông tin sau: "Đây là chương trình được thiết kế bởi Lê Thị Thu Thảo, K20HTTT, Trường Đại học Công Nghệ, Đại học quốc gia Hà Nội".

Chọn Encrypt Nhập hoặc hiển thị Bản rõ vào phần Input sau đó ấn Execute sẽ cho ra bản mã ở phần Output.



Hình 3.21. Mã hóa kép

3.2.2 Giải mã kép

3.2.2.1 Giải mã lần 1 bằng khóa giả ngẫu nhiên

Đầu vào: Bản mã của bản mã từ điển.

Đầu ra: Bản mã từ điển.

Khóa: Dãy khóa được sinh ra từ module sinh khóa.

Thực hiện:

Bước 1:

- Tách lấy số đầu tiên trong chuỗi đã mã đầu vào: ký hiệu c_i.
- Lấy khóa đầu tiên trong bộ khóa: ký hiệu k_i.
- Thực hiện tìm phần tử nghịch đảo của k_ibằng thuật toán Euclid mở rộng: ký hiệu k_j.
- Thực hiện giải mã: $n_i = c_i * k_j \pmod p$
- Thêm $n_i$vào chuỗi output.

Bước 2:

Lặp bước 1 cho số và khóa tương ứng, cho đến cuối chuỗi mã

3.2.2.2 Giải mã lần 2 qua từ điển

Đầu vào: Bản mã từ điển.

Đầu ra: Bản rõ ban đầu.

Thực hiện:

Bước 1: Đối chiếu từ số trong bản mã từ điển với bản rõ của nó trong từ điển (từ/cụm từ tương ứng với từng ID).

Bước 2: Chuẩn hóa bản rõ thu được và lưu lại.

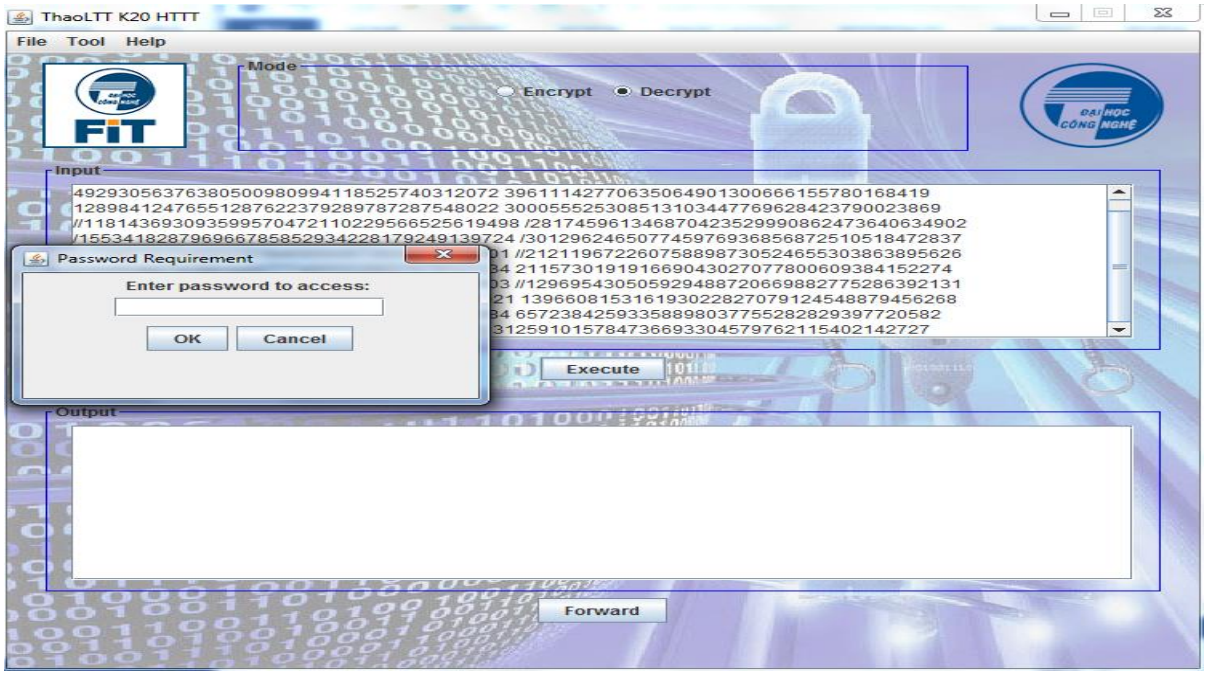
3.2.2.3 Kết quả giải mã

- **Chọn Decrypt:**

  Nhập và hiển thị bản mã vào phần Input sau đó ấn Execute và nhập quyền được giải mã sẽ cho ra bản rõ ở phần Output.

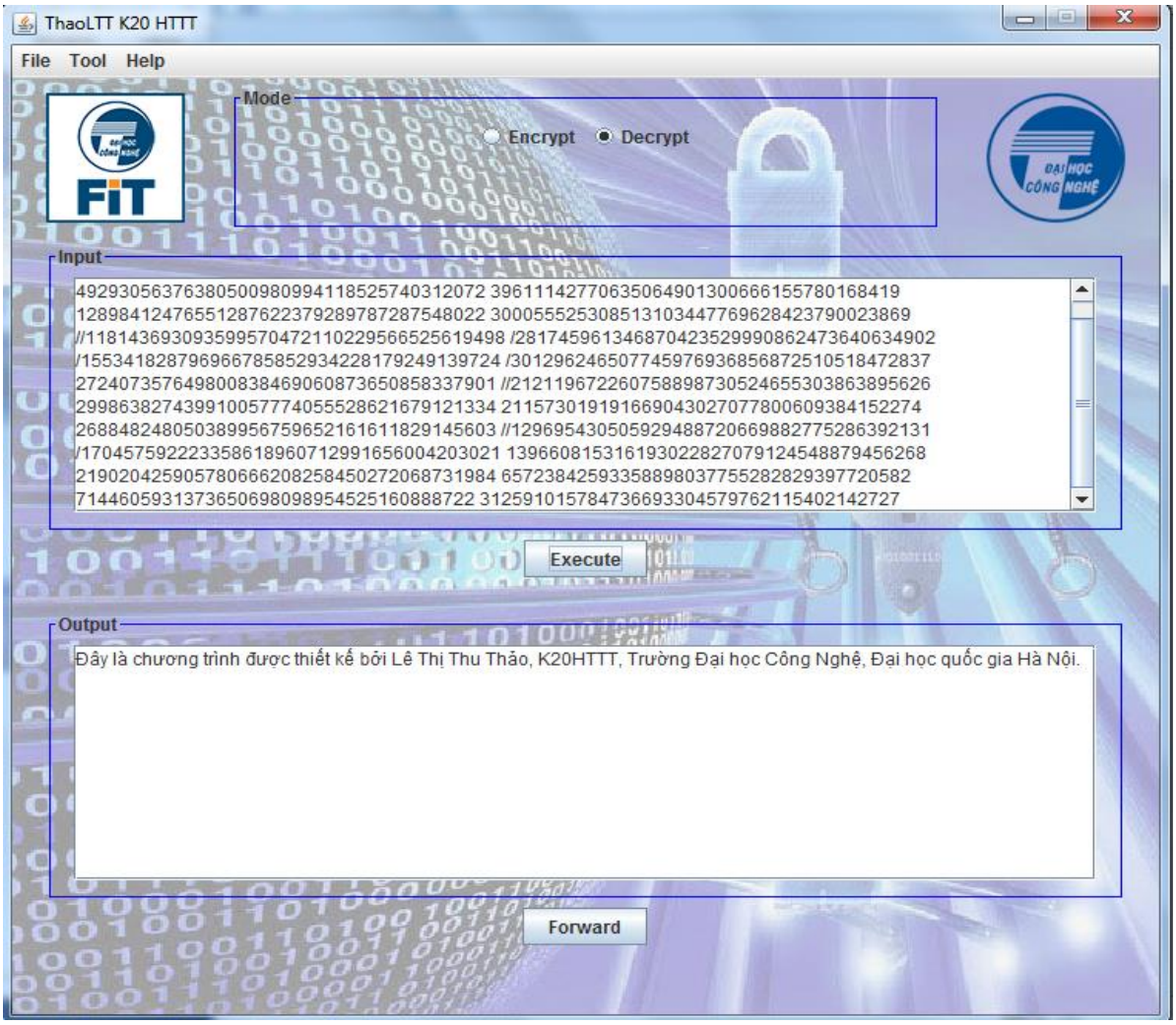
- **Chọn Decrypt:**

  Ấn Forward sau đó ấn Execute và nhập quyền được phép giải mã.



Hình 3.22. Yêu cầu nhập mã giải mã

Kết quả bản rõ:



Hình 3.23. Bản rõ

KẾT LUẬN

Các kết quả đã đạt được

Luận văn đã đạt được các kết quả sau:

- Trình bày các bài toán xử lý ngôn ngữ tự nhiên và ứng dụng.
- Giới thiệu xử lý văn bản tiếng Việt.
- Đưa ra giới thiệu tổng quan các hệ mật từ cổ điển đến hiện tại.
- Xây dựng hệ mật kép an toàn, kết hợp giữa luật từ điển với hệ mật sử dụng khóa một lần bằng dãy khóa giả ngẫu nhiên.
- Viết chương trình demo hệ mật kép an toàn.

Khả năng ứng dụng thực tiễn của luận văn

Kết quả của luận văn có thể ứng dụng trong thực tiễn để bảo vệ giữ liệu trong lưu trữ và bảo mật thông tin trên đường truyền.

Hướng phát triển của luận văn

- Phần mềm demo trong luận văn có thể phát triển thành 1 phần mềm thương mại để bảo mật thông tin trong các thiết bị di động.
- Các thuật toán trong hệ mã kép có thể hoàn thiện để tạo ra sản phẩm phục vụ an ninh quốc gia.

TÀI LIỆU THAM KHẢO

Tiếng Việt

[1] Lê Phê Đô, Mai Mạnh Trừng, Lê Thị Len, Nguyễn Văn Thắng, Lê Trung Thực, Lê Thị Thu Thảo, Đỗ Năng Thuận, Đỗ Công Thành, *Hệ mật mã kép an toàn*, từ trang 88 đến trang 95. Hội thảo quốc gia lần thứ XVII: Một số vấn đề chọn của công nghệ thông tin và Truyền thông – Đắk Lắk, 30 – 31/10/2014.

[2] Trịnh Nhật Tiến, *Giáo trình an toàn dữ liệu*, NXB Đại học Quốc Gia, 2008.

[3] Trần Xuân Phương, *Xác thực điện tử và ứng dụng trong giao dịch hành chính*, luận văn thạc sỹ, Trường ĐHCN – ĐHQG HN, 2015.

[4] Nguyễn Bình (2004), *Giáo trình Mật mã học*, Học viện Công nghệ Bưu chính Viễn thông, Nxb Bưu điện, 2004.

[5] Đặng Hoài Bắc, (2010) "*Các mã cyclic và cyclic cục bộ trên vành đa thức có hai lớp kề cyclic*", Luận án TS kỹ thuật.

[6] Nguyễn Thị Minh Huyền, Vũ Xuân Lương, Lê Hồng Phương, Sử dụng bộ gán nhãn từ loại xác suất Qtag cho văn bản tiếng Việt. Hội thảo khoa học quốc gia lần thứ nhất về Nghiên cứu phát triển và ứng dụng công nghệ thông tin và truyền thông, ICT.rda, 2003.

[7] Nguyễn Lê Minh, Cao Hoàng Trụ, Phân cụm từ tiếng Việt bằng phương pháp học máy cấu trúc, ICT08 – VLSP – VP84 – 2.

[8] Trần Mai Vũ, Tóm tắt đa văn bản dựa vào trích xuất câu, Luận văn thạc sĩ , Trường ĐHCN – ĐHQG HN, 2009.

[9] Vũ Tiến Thành, Bài toán trích xuất thông tin cho dữ liệu bán cấu trúc và áp dụng xây dựng hệ thống tìm kiếm giá cả sản phẩm, Khóa luận tốt nghiệp Đại học hệ chính quy, 2009.

[10] Lê Hoàng Thanh, Text Mining – Kỹ thuật trích xuất thông tin từ văn bản, http://www.ntu.edu.vn/Portals/7/KTPM/thanhlh/intro%20text%20mining.pdf.

Tiếng Anh

[11] Elaine Barker, John Kelsey, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, NIST Special Publication 800-90A, 2012.

[12] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangle, David Banks, Alan Heckert, James Dray, San Vo, *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, April 2010, NIST Technology Administration U.S Department of Commerce.

[13] John C.Cherniavsky, Robert Constable, Jean Gallier, Richard Platek, Richard Statman, *Igor-criptographic applications of analytic number theory – Complexity Lower Bounds and Pseudorandomness*, 2003, Macquarie University.

[14] Douglas R.Stinson, *Cryptography theory and practice 3th*, 2003.

[15] *Concrete Security of the Blum – Blum – Shub Pseudorandom Generator*, Cryptography and coding 10th IMA International conference.

[16] *FIPS – 197 Advanced Encryption Standard (AES)*, NIST, 2001.

[17] Vincent Rijmen, *10 years of Rijndael*, 2008.

[18] Joan Deamen, Vincent Rijmen, *AES Proposal: Rijndael*, 2003.

[19] Adam Berent, *Advanced Encryption Standard by example.*

[20] Joan Deamen, Vincent Rijmen, *A Specification for Rijndael, the AES Algorithm*, 2003.

Internet

[21] http://www.random.org/

[22] https://vi.wikipedia.org/wiki/AES_(m%C3%A3_h%C3%B3a)

[23] http://image.slidesharecdn.com/chuong2-131012164541-phpapp01/95/matma-chuong2-8-638.jpg?cb=1381596494

[24] http://ptit.edu.vn/wps/wcm/connect/70d8a20047ec5b559ca5dda81258549d/Luan+van+new+12-2013.pdf?MOD=AJPERES&CONVERT_TO=url&CACHEID=70d8a20047ec5b559ca5dda81258549d

PHỤ LỤC I

HỆ MẬT MÃ KÉP AN TOÀN

PHỤ LỤC II

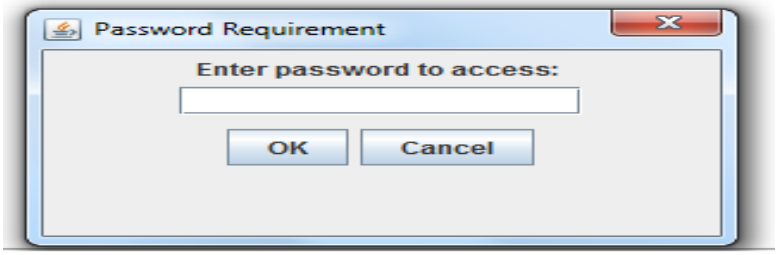
Hướng dẫn sử dụng chương trình

Phần mềm hiển thị được viết bằng ngôn ngữ Java. Công cụ lập trình Eclipse.

Giao diện hiển thị gồm có giao diện đăng nhập vào chương trình và giao diện chương trình chính.

1. Giao diện đăng nhập vào chương trình

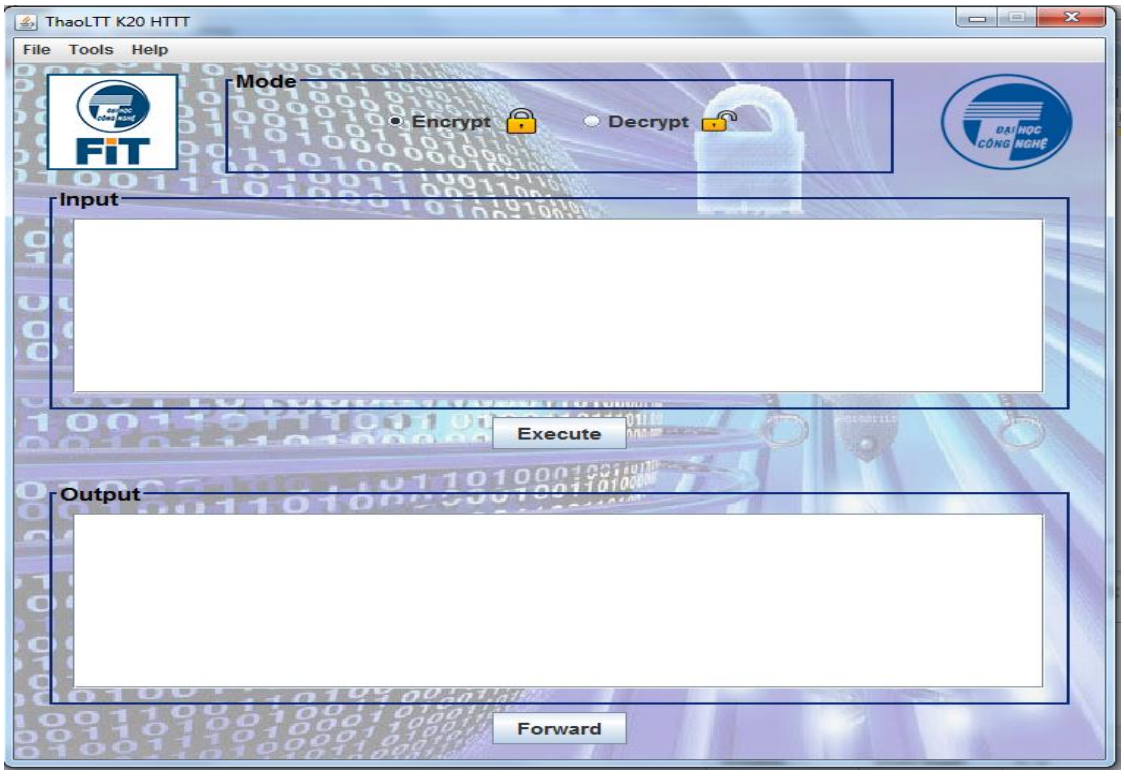
Trước khi vào giao diện chương trình chính để sử dụng chúng ta phải đăng nhập. Ở đây có 2 quyền đăng nhập là: Root và Guest. Guest được cấp mật khẩu dành cho Guest và cũng có thể thay đổi mật khẩu sau khi đăng nhập thành công. Root có thể thể vô hiệu hóa Guest đăng nhập bằng tài khoản Root.



Hình 1. Đăng nhập chương trình

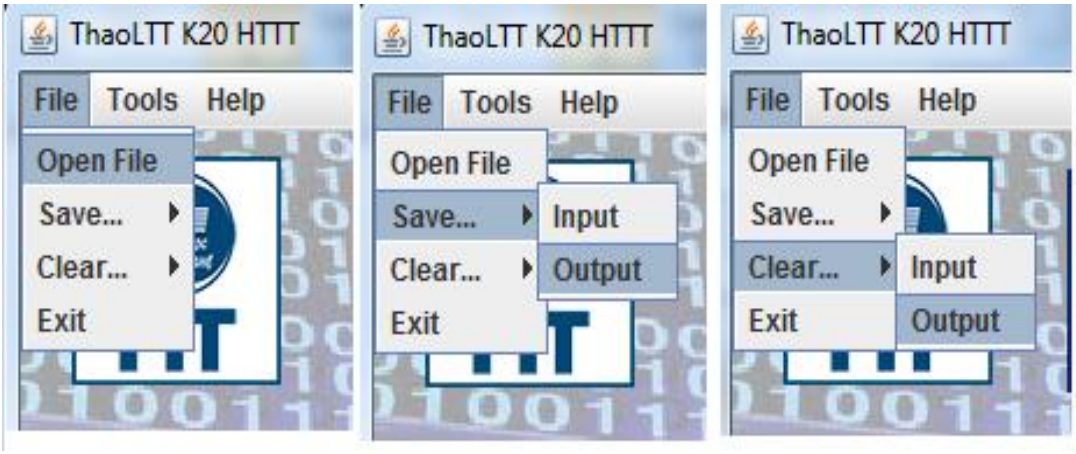
2. Giao diện chương trình chính

Giao diện chương trình Có các chức năng: Input, output, Execute, Forword, File, Tool, Help, Encrypt và Decrypt.



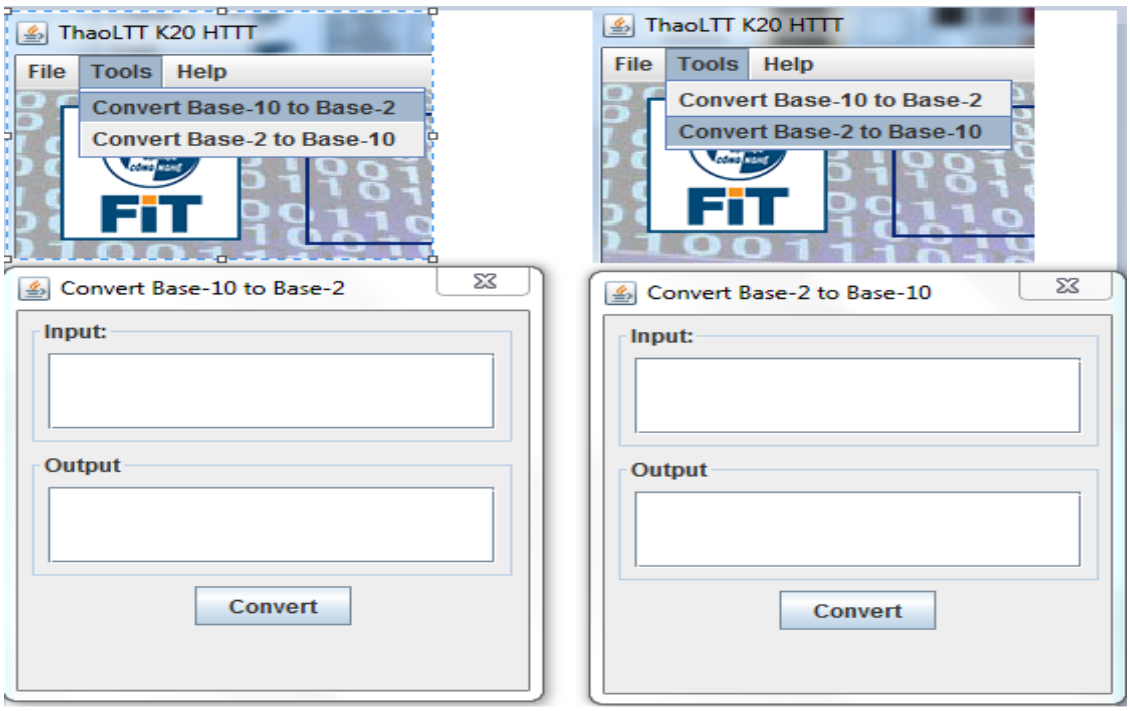
Hình 2. Giao diện chương trình

- Encrypt là phần mã hóa.
- Dearypt là phần giải mã
- Execute phần thực thi mã hóa hoặc giải mã.
- Input là phần dành cho nhập, hiển thị nội dung bản rõ hoặc bản mã.
- Output là phần hiển thị nội dung bản mã hoặc bản rõ.
- Forward với mục đích khi mã hóa ra bản mã và muốn kiểm tra lại bản rõ.
- File: Có các chức năng như Open File, Save Input, Save Output, Clear Input, Clear Output, Exit



Hình 3. Các chức năng File

- Tool: Có 2 chức năng là Convert Base-10 to Base-2 và ngược lại.

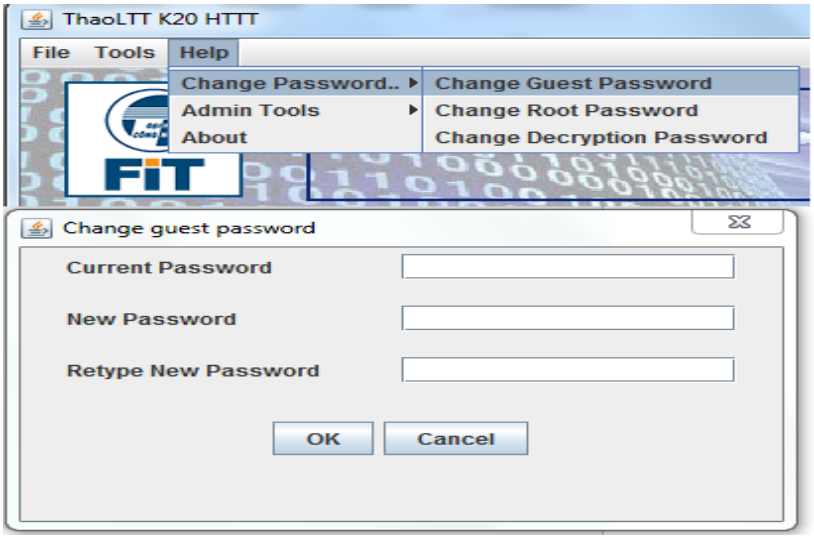


Hình 4. Chức năng Tools

- Help: Có 3 chức năng Change Pasword, Admin Tools và About

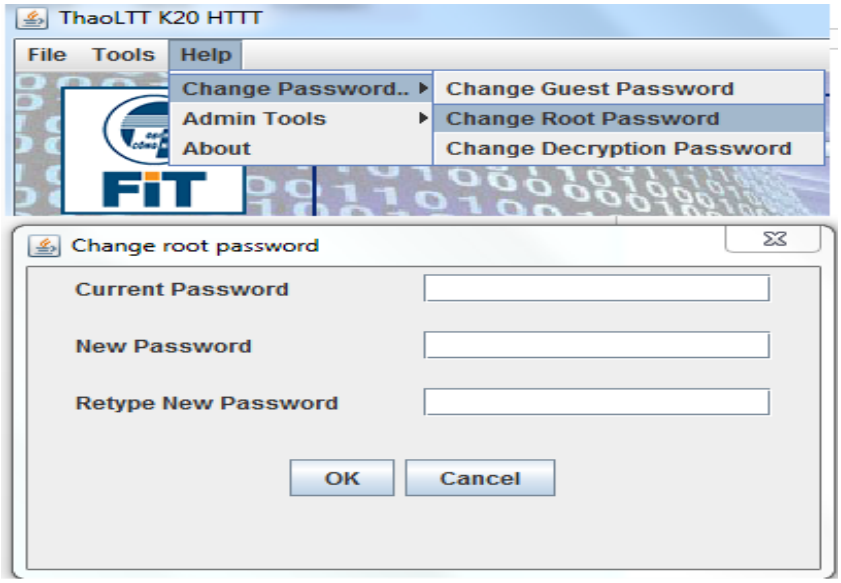
Hình 5. Chức năng Hellp

- **Chức năng change Password**
  - **Dành cho Guest:** Thay đổi mật khẩu đăng nhập



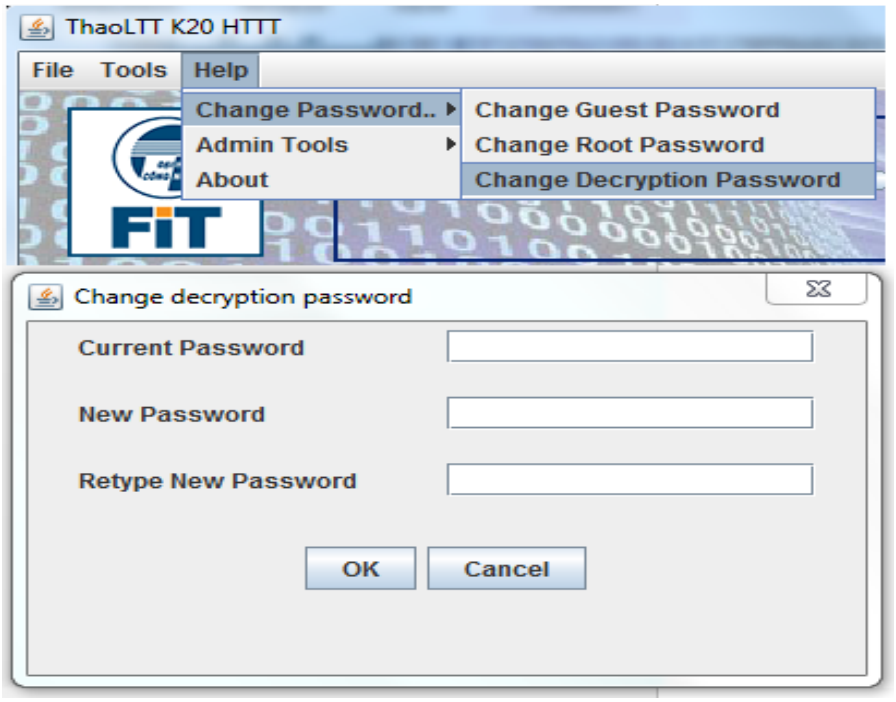
Hình 6. Chức năng dành cho Guest

  - **Dành cho Root:** Thay đổi mật khẩu đăng nhập



Hình 7. Chức năng dành cho Root

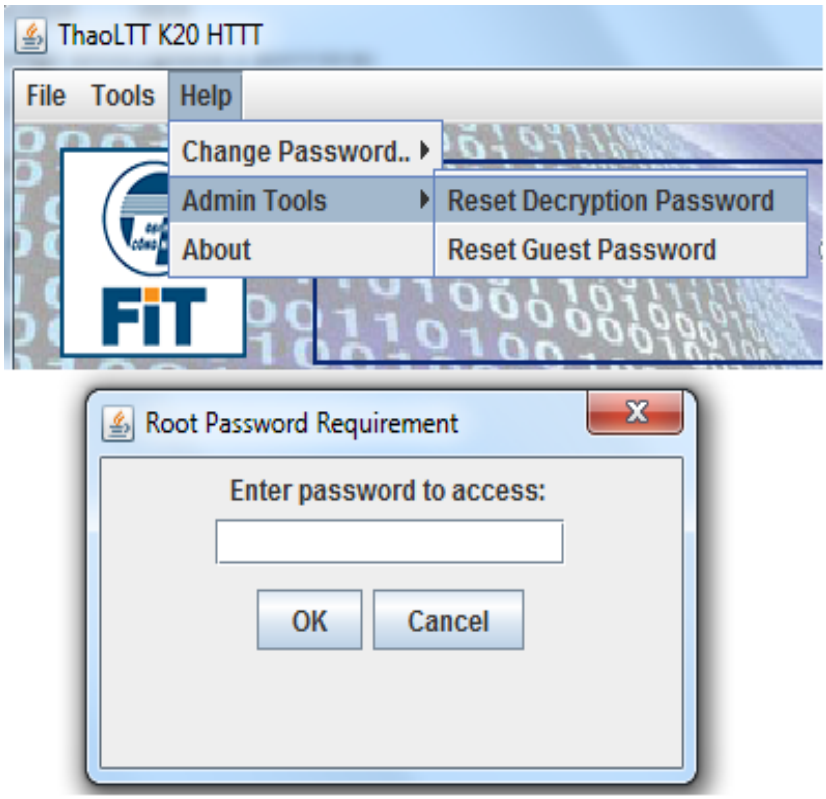
➢ **Change Decryption Password:** Thay đổi mật khẩu giải mã.



Hình 8. Chức năng thay đổi mật khẩu giải mã

- **Chức năng Admin Tools:**

Reset lại mật khẩu giải mã và mật khẩu dành cho guest. Chức năng này chỉ dành riêng cho Admin.



Hình 9. Reset lại mật khẩu giải mã và mật khẩu dành cho Guest

- **Phần About: Thông tin của người thiết kế chương trình**

Hình 10. Thông tin của người thiết kế chương trình