**TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHỆ CẦN THƠ**

KHOA CÔNG NGHỆ THÔNG TIN

**CHUYÊN NGÀNH KHOA HỌC MÁY TÍNH**

**★★★🕮★★★**

**A white circle with blue text and a book and a symbol

Description automatically generated**

ĐỀ TÀI ĐỒ ÁN:

**TÌM HIỂU VỀ GA VÀ ỨNG DỤNG**

Giảng viên hướng dẫn: **LÊ ANH NHÃ UYÊN**

Sinh viên thực hiện: **LÊ PHƯỚC HỮU 2100450**

**NGUYỄN MỸ HẰNG 2101166**

***Cần Thơ, tháng 12, năm 2023***

**TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHỆ CẦN THƠ**

KHOA CÔNG NGHỆ THÔNG TIN

**CHUYÊN NGÀNH KHOA HỌC MÁY TÍNH**

**★★★🕮★★★**

**A white circle with blue text and a book and a symbol

Description automatically generated**

ĐỀ TÀI ĐỒ ÁN:

**TÌM HIỂU VỀ GA VÀ ỨNG DỤNG**

Giảng viên hướng dẫn: **LÊ ANH NHÃ UYÊN**

Sinh viên thực hiện: **LÊ PHƯỚC HỮU 2100450**

**NGUYỄN MỸ HẰNG 2101166**

***Cần Thơ, tháng 12, năm 2023***

**PHIẾU CHẤM ĐIỂM**

**LỜI CAM ĐOAN**

Chúng em xin cam đoan đề tài đồ án “Tìm Hiểu Về GA Và Ứng Dụng” là quá trình tìm hiểu, nghiên cứu của riêng chúng em, dưới sự hướng dẫn của giảng viên Lê Anh Nhã Uyên. Các nội dung, khái niệm, dữ liệu trong đồ án đều được chúng em tự nghiên cứu, trích dẫn từ các giáo trình, bài viết có đủ cơ sở khoa học.

**LỜI CẢM ƠN**

Trong lần đầu tiên chúng em thực hiện đồ án 1 của học kỳ một năm thứ ba tại Trường Đại Học Kỹ Thuật Công Nghệ Cần Thơ, thật may mắn khi được giảng viên là cô Lê Anh Nhã Uyên đồng thời cũng là cố vấn học tập của chúng em đã đồng hành, hướng dẫn chúng em trong quá trình thực hiện đồ án 1 này. Với sự chân thành, nhiệt quyết của Cô chúng em muốn gửi lời cảm ơn chân thành đến cô Lê Anh Nhã Uyên vì đã cùng chúng em hoàn thành đồ án 1 này.

Tuy nhiên, kiến thức về đề tài GA (Genetic Algorithm) của chúng em vẫn còn những hạn chế nhất định. Do đó, không tránh khỏi những thiếu sót trong quá trình hoàn thành bài đồ án này. Mong cô xem và góp ý để đồ án của chúng em được hoàn thiện hơn.

Kính chúc cô hạnh phúc và thành công hơn nữa trong sự nghiệp trồng người. Kính chúc cô luôn dồi dào sức khỏe để tiếp tục dìu dắt nhiều thế hệ học trò đến những bến bờ tri thức.

Em xin chân thành cảm ơn!

**MỞ ĐẦU**

1. **Lý do chọn đề tài**

Lĩnh vực Công Nghệ Thông Tin (Information Technology) hiện nay đóng một vai trò không thể thay thế đối với sự phát triển của mọi thành phần ở một quốc gia cụ thể là về các lĩnh vực như: Tài chính, giáo dục, giao thông vận tải, cơ sở hạ tầng, giải trí, thể thao,... có thể thấy rằng lĩnh vực này dường như đã và đang không thể thiếu đối với sự phát của mọi lĩnh vực khác có mặt trong xã hội. Những vấn đề nan giải, những công việc tưởng chừng khó có thể hoàn thành một cách nhanh chóng, những bài toán thực tế chưa có lời giải tối ưu thì hiện nay đã có giải pháp, đã có phương pháp để giải quyết vấn đề một cách gọn gàng, những công việc phải sử dụng tay chân quá nhiều không cần thiết, nghiệp vụ rườm rà thiếu hiệu quả, mất nhiều thời gian thì nay đã được khắc phục một cách triệt để hơn rất nhiều so với trước kia, nhờ vào việc đã ứng ứng dụng thực tiễn các phần mềm, công cụ, công nghệ vào trong quá trình xử lý công việc, học tập, giảng dạy, giúp cho những công việc này được tiến hành hiệu quả hơn, tiết kiệm được một số chi phí dư thừa, từ đó cải thiện được các khía cạnh khác của một cá nhân của một tổ chức và khi cá nhân, tổ chức phát triển thì toàn xã hội đều được phát triển. Nhắc đến lĩnh vực Công Nghệ Thông Tin thì chúng ta không thể không nhắc đến một lĩnh vực chuyên sâu hơn đó là Trí Tuệ Nhân Tạo.

Lĩnh vực Trí Tuệ Nhân Tạo (Artificial Intelligence) được biết đến là một nhánh của chuyên ngành Khoa Học Máy Tính (Computer Science) và chuyên ngành này là một phần của lĩnh vực Công Nghệ Thông Tin. Ta có thể thấy Trí Tuệ Nhân Tạo như là bộ não của một phần mềm, công cụ, thiết bị,... Trí Tuệ Nhân Tạo sẽ quyết định tính hiệu quả của phần mềm, công cụ, thiết bị đó ở mức nào. Nhắc đến Trí Tuệ Nhân Tạo thì không thể không nhắc đến Máy Học (Machine Learning) và Học Sâu (Deep Learning), hai thành phần này đều nằm bên trong Trí Tuệ Nhân Tạo và là nền tảng giúp cho Trí Tuệ Nhân Tạo phát triển. Chung quy lại những lĩnh vực chuyên sâu được nêu trên đều cần đến Giải Thuật và chỉ có Giải Thuật mới giúp cải thiện được khả năng giao tiếp giữa Máy và Người và cải thiện được tính hiệu quả khi giải quyết các bài toán thực tế thông qua việc ứng dụng giải thuật vào trong Artificial Intelligence, Machine Learning, Deep Learning.

Trong quá trình học tập và nghiên cứu về chuyên ngành khoa học máy tính, nhận thấy rằng những phương pháp, những kỹ thuật được ứng dụng để tìm kiếm giải pháp cho các bài toán khó, làm sao có được một kết quả tối ưu nhất. Có rất nhiều phương pháp, kỹ thuật được ứng dụng thì một trong số những phương pháp, kỹ thuật được biết đến là Giải Thuật Di Truyền (Genetic Algorithm) kỹ thuật này được dựa trên nguyên lý sinh học tiến hóa để tìm ra một kết quả tối ưu nhất và khi tìm hiểu sơ bộ về Giải Thuật Di Truyền thì chúng em thấy được sự hứng thú nhất định về giải thuật này, thấy được những ứng dụng hữu ích mà kỹ thuật này mang lại trong việc đóng góp sự phát triển các giải thuật để tìm ra lời giải cho các bài toán nan giải.

Và trong lần đầu tiên thực hiện đề án 1 này, chúng em sẽ cùng nhau nghiên cứu, tìm hiểu, thực hành ứng dụng một giải thuật tối ưu kết quả cho bài toán đó là Giải Thuật Di Truyền (Genetic Algorithm).

1. **Đối tượng, phạm vi nghiên cứu**

* **Đối tượng**
* Giải Thuật Di Truyền.
* Phương pháp biểu diễn Giải Thuật Di Truyền.
* Ứng dụng Giải Thuật Di Truyền vào Demo.
* **Phạm vi nghiên cứu**
* Nghiêu cứu các giáo trình, Website, video về Giải Thuật Di Truyền.
* Demo chương trình Giải Thuật Di Truyền vào “Tìm Kiếm Chuỗi Mục Tiêu Được Nhập Từ Bàn Phím”.
* IDE: Visual Studio Code.
* Programming Language: Python 3.11.6

1. **Mục đích của đề tài**

Mục đích sau khi hoàn thành đề tài tìm hiểu, nghiên cứu về Giải Thuật Di Truyền là:

* Có khả năng nghiên cứu độc lập, nghiên cứu cùng các thành viên trong nhóm.
* Kỹ năng lập kế hoạch, nắm bắt được tiến độ trong quá trình thực hiện đề tài.
* Pháp triển kiến thức chuyên ngành, cụ thể hơn là về Giải Thuật Di Truyền.
* Phát triển tư duy, có cái nhìn khách quan hơn về một Giải Thuật.

1. **Mục tiêu của đề tài** (chưa viết)

**GIỚI THIỆU**

Trong đề tài Giải Thuật Di Truyền của đồ án 1 này chúng em trình bày các nội dung như sau:

* **Chương 1. GIỚI THIỆU TỔNG QUAN**
* **Chương 2. GIẢI THUẬT DI TRUYỀN**
* **Chương 3. ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN VÀO CÁC BÀI TOÁN**
* **Chương 4. THỰC HÀNH ỨNG DỤNG GIẢI THUẬT DI TRUYỀN THỰC TẾ**

**Chương 1**: Nêu các khái niệm liên quan đến Giải Thuật Di Truyền như: Khoa Học Máy Tính là gì?, lịch sử Khoa Học Máy Tính, vai trò của Khoa Học Máy Tính đối với sự phát triển đa ngành nghề. Trình bày thêm về Giải Thuật, lý do phải sử dụng đến giải thuật và một số giải thuật tối ưu kết quả bài toán.

**Chương 2**: Ở chương này chúng em sẽ trình bày thuyết di truyền của Charles Darwin, các thành phần cấu thành quá trình di truyền và vẽ sơ đồ về quá trình di truyền xảy ra trong tự nhiên. Tiếp theo nghiên cứu sâu về Giải Thuật Di Truyền, lịch sử của Giải Thuật Di Truyền, vẽ sơ đồ tiến trình, tiếp theo trình bày về cơ chế thực hiện và nguyên lý hoạt động của Giải Thuật Di Truyền. Nêu ra các phương pháp biểu diễn bài toán của Giải Thuật Di Truyền và những toán tử cơ bản trong Giải Thuật Di Truyền.

**Chương 3**: (chưa viết)

**Chương 4**: (chưa viết)

**MỤC LỤC**

[**Chương 1** 1](#_Toc149718737)

[**1.** **KHOA HỌC MÁY TÍNH** 1](#_Toc149718738)

[**1.1** **Khái niệm về Khoa Học Máy Tính** 1](#_Toc149718739)

[**1.2** **Lịch sử của Khoa Học Máy Tính** 1](#_Toc149718740)

[**1.3** **Ứng dụng giải thuật trong Khoa Học Máy Tính** 2](#_Toc149718741)

[**2.** **GIẢI THUẬT** 4](#_Toc149718742)

[**2.1** **Khái niệm về Giải Thuật** 4](#_Toc149718743)

[**2.2** **Các đặc trưng của Giải Thuật** 5](#_Toc149718744)

[**2.3** **Các giải thuật phổ biến** 6](#_Toc149718745)

[**Chương 2** 11](#_Toc149718746)

[**1.** **DI TRUYỀN VÀ LỊCH SỬ CỦA DI TRUYỀN** 11](#_Toc149718747)

[**1.1** **Khái niệm về Di Truyền** 11](#_Toc149718748)

[**1.2** **Lịch sử của Di Truyền** 12](#_Toc149718749)

[**1.3** **Các thành phần trong quy trình Di Truyền** 13](#_Toc149718750)

[**1.4** **Sơ đồ quy trình của Di Truyền** 14](#_Toc149718751)

[**2.** **GIẢI THUẬT DI TRUYỀN** 14](#_Toc149718752)

[**2.1** **Khái niệm về Giải Thuật Di Truyền** 14](#_Toc149718753)

[**2.2** **Lịch sử về Giải Thuật Di Truyền** 15](#_Toc149718754)

[**2.3** **Các thành phần trong Giải Thuật Di Truyền** 16](#_Toc149718755)

[**3.** **PHƯƠNG PHÁP BIỂU DIỄN BÀI TOÁN TRONG GIẢI THUẬT DI TRUYỀN** 17](#_Toc149718756)

[**4.** **MÃ HÓA** 18](#_Toc149718757)

[**4.1** **Binary encoding** 19](#_Toc149718758)

[**4.2** **Real – valued encoding** 19](#_Toc149718759)

[**4.3** **Tree encoding** 19](#_Toc149718760)

[**4.4** **Permutation encoding** 19](#_Toc149718761)

[**5.** **CÁC TOÁN TỬ TRONG GIẢI THUẬT DI TRUYỀN** 19](#_Toc149718762)

[**6.** **CƠ CHẾ THỰC HIỆN** 19](#_Toc149718763)

[**7.** **NGUYÊN LÝ HOẠT ĐỘNG** 20](#_Toc149718764)

**DANH MỤC THUẬT NGỮ TIẾNG ANH**

**DANH MỤC HÌNH**

# **Chương 1**

**GIỚI THIỆU TỔNG QUAN**

1. **KHOA HỌC MÁY TÍNH**
2. **Khái niệm về Khoa Học Máy Tính**

*Khoa Học Máy Tính* là chuyên ngành nghiên cứu về các cơ sở lý thuyết về các dữ liệu thông tin và tính toán, cùng với sự thực hiện và ứng dụng của chuyên ngành này trong các hệ thống máy tính. *Khoa Học Máy Tính* là cách tiếp cận khoa học và thực tiễn nhất để có thể tính toán nghiên cứu tìm ra các giải pháp thích hợp, tối ưu nhất cho các vấn đề nan giải, việc thua thập dữ liệu, phân tích, xử lý, lưu trữ, truyền thông và truy cập dữ liệu tất cả đều là một phần trong *Khoa Học Máy Tính*.

Một định nghĩa khác về chuyên ngành này, *Khoa Học Máy Tính* nghiên cứu sâu về các quy trình, cách thức để thực hiện của các giải thuật từ đó trên nền tảng của các giải thuật đã nghiên cứu, kế tiếp là áp dụng vào các bài toán thực tế chưa có lời giải tối ưu nhất.

Có rất nhiều bài toán tối ưu chưa thật sự có một giải thuật tối ưu thõa đáng, như thế dẫn đến việc một chương trình, hệ thống giảm đi tính hiệu quả, tốn nhiều tài nguyên và thời gian. Công việc của một kỹ sư *Khoa Học Máy Tính* là nghiên cứu, thấu hiểu vấn đề, chọn lọc các giải pháp, tích hợp các dữ liệu đã được thu thập, từ đó đưa ra các giải thuật tối ưu thõa đáng cho một bài toán.

1. **Lịch sử của Khoa Học Máy Tính**

Lịch sử của Khoa Học Máy Tính đã bắt đầu từ nhiều năm trước khi các kỹ sư phát minh ra máy tính hiện đại. Những máy tính dành cho việc tính toán các bài toán số học đã tồn tại từ thời cổ đại, chẳng hạn như bàn tính. Hơn nữa, các giải thuật để thực hiện tính toán đã có kể từ thời cổ đại, ngay cả trước khi phát triển các thiết bị tính toán phức tạp.

*Wilhelm Schickard* đã thiết kế và hoàn thành chiếc máy tính cơ học đầu tiên năm 1623. Đến năm 1673, *Gottfried Leibniz* trình diễn một máy tính cơ học số, được gọi là *Stepped Reckoner*. Ông có thể được coi là nhà *Khoa Học Máy Tính* đầu tiên và nhà lý thuyết thông tin mà đã ghi lại hệ thống số nhị phân.

*Blaise Pascal* thiết kế và xây dựng máy tính cơ học hoạt động được mang tên *Pascaline.* Năm 1642 *Charles Babbage* đã thiết kế một máy tính theo hiệu (*Difference Engine*) vào thời *Victoria* và *Ada Lovelace* đã viết bản hướng dẫn sử dụng máy. Nhờ công trình này, ngày nay bà được biết đến là lập trình viên đầu tiên trên thế giới. Vào khoảng năm 1900, tập đoàn *IBM* đã bán những chiếc máy tính dùng thẻ đục lỗ. Tuy nhiên, tất cả những chiếc máy này đều chỉ thực hiện một nhiệm vụ đơn hoặc cùng lắm là một tập nhỏ các nhiệm vụ.

Trước năm 1920, công việc tính toán được thực hiện chủ yếu bởi những nhân viên chuyên nghiệp. Những nhà nghiên cứu đầu tiên về ngành mà sau này được gọi là *Khoa Học Máy Tính*, chẳng hạn như *Kurt Gödel*, *Alonzo Church* và *Alan* *Turing*, đã quan tâm đến câu hỏi về khả năng tính toán: “*những gì có thể được tính toán bởi một người thủ quỹ người chỉ đơn giản dùng giấy và bút chì để làm một danh sách các bước tính toán, cho đến khi nào xong việc mà không cần đến trí thông minh hay hiểu biết?*”. Một phần của động cơ này là ước muốn phát triển các máy tính có khả năng tự động hóa các công việc tính toán thường là buồn tẻ và dễ sai của một người tính toán. Vấn đề then chốt là xây dựng các hệ thống tính toán phổ dụng có khả năng thực hiện mọi nhiệm vụ tính toán có thể cần đến và nhờ đó tổng quát hóa tất cả các máy tính chuyên biệt trước kia thành một khái niệm đơn nhất về chiếc máy tính phổ dụng.

Trong những năm 1940, khi các máy tính mới hơn và mạnh hơn được phát triển, người ta thấy rõ ràng hơn rằng máy tính có thể được sử dụng cho nhiều mục đích khác ngoài các tính toán toán học, lĩnh vực *Khoa Học Máy Tính* được mở rộng thành ngành nghiên cứu tính toán nói chung. Từ thập kỷ 60, *Khoa Học Máy Tính* bắt đầu được thiết lập như là một ngành học riêng biệt, với sự ra đời của các khoa *Khoa Học Máy Tính* đầu tiên và các chương trình đào tạo đại học chuyên ngành *Khoa Học Máy Tính*. Từ khi các máy tính được sử dụng trong thực tiễn, nhiều ứng dụng của tính toán đã trở thành các lĩnh vực nghiên cứu riêng biệt.

1. **Ứng dụng giải thuật trong Khoa Học Máy Tính**

Lĩnh vực Công Nghệ Thông Tin nói chung và Khoa Học Máy Tính nói riêng đối với sự phát triển đa ngành nghề hiện nay thì không thể thiếu ứng dụng thực tiễn của hai lĩnh vực này. Lấy một ví dụ “*Một doanh nghiệp cần đến một hệ thống phân tích hành vi tiêu dùng của các khách hàng của họ thông qua việc những dữ liệu đã được thu thập được, dựa trên sự phân tích của hệ thống mà doanh nghiệp sẽ đưa ra các chương trình khuyên mãi, các chiến lược dài hạn trong tương lai.*”, tùy theo tính hiệu quả của các giải thuật đã áp dụng trong hệ thống phân tích đó, mà có thể ảnh hưởng đến hiệu năng phân tích. Một giải thuật đủ tối ưu thì sẽ cho ra kết quả gần đúng nhất với thực tế từ đó tạo một gốc nhìn khách quan hơn cho các tổ chức doanh nghiệp, để đưa ra các lựa chọn đúng đắn. Giải thuật được sử dụng trong mọi lĩnh vực khoa học máy tính, từ hệ điều hành, mạng cho đến trí tuệ nhân tạo. Dưới đây là một vài ví dụ:

* **Operating systems**: Giải thuật được sử dụng để quản lý CPU, bộ nhớ và các tài nguyên khác của hệ thống máy tính. Ví dụ: giải thuật lập lịch xác định tiến trình nào sẽ chạy tiếp theo trên CPU.
* **Networking**: Giải thuật được sử dụng để định tuyến các gói dữ liệu giữa các máy tính trên internet. Ví dụ: giải thuật Dijkstra có thể được sử dụng để tìm đường đi ngắn nhất giữa hai nút trong mạng.
* **Databases**: Giải thuật được sử dụng để lưu trữ và truy xuất dữ liệu hiệu quả từ cơ sở dữ liệu. Ví dụ: giải thuật cây B là cấu trúc dữ liệu có thể được sử dụng để tìm kiếm nhanh dữ liệu trong cơ sở dữ liệu đã được sắp xếp.
* **Compilers**: Giải thuật được sử dụng để biên dịch các chương trình máy tính từ ngôn ngữ cấp cao (chẳng hạn như C hoặc Java) sang ngôn ngữ cấp thấp (chẳng hạn như mã máy) mà máy tính có thể hiểu được.
* **Graphics**: Giải thuật được sử dụng để tạo và hiển thị hình ảnh và hoạt ảnh. Ví dụ: giải thuật dò tia có thể được sử dụng để tạo ra hình ảnh chân thực bằng cách mô phỏng cách ánh sáng truyền qua một cảnh.
* **Machine Learning**: Các giải thuật được sử dụng để huấn luyện các mô hình học máy học từ dữ liệu và đưa ra dự đoán. Ví dụ: giải thuật máy vectơ hỗ trợ có thể được sử dụng để phân loại các điểm dữ liệu thành các danh mục khác nhau.

Các giải thuật rất cần thiết để giải quyết các vấn đề và nhiệm vụ có độ phức tạp , nâng cao hiệu quả và hiệu suất cũng như hỗ trợ các công nghệ và ứng dụng mới. Một số ví dụ cụ thể về các giải thuật được sử dụng ngày nay: Công cụ tìm kiếm Google sử dụng giải thuật phức tạp để xếp hạng các trang web và trả về kết quả phù hợp nhất cho người dùng. Hệ thống đề xuất Netflix sử dụng giải thuật để đề xuất phim và chương trình truyền hình cho người dùng dựa trên lịch sử xem của họ. Giỏ hàng Amazon sử dụng giải thuật để giới thiệu sản phẩm cho người dùng dựa trên những lần mua hàng trước đây của họ. Xe tự lái sử dụng nhiều giải thuật khác nhau để điều hướng đường đi và tránh chướng ngại vật. Bộ lọc thư rác trong ứng dụng email của bạn sử dụng giải thuật để xác định và chặn email spam.

Các giải thuật hiện nay đã có ở khắp mọi nơi trong thế giới hiện đại và rất cần thiết cho nhiều công nghệ mới, nâng cao chất lượng công việc, đời sống, sức khỏe, giáo dục,...cho con người.

1. **GIẢI THUẬT**
2. **Khái niệm về Giải Thuật**

[Giải thuật](https://quantrimang.com/cong-nghe/cau-truc-du-lieu-va-giai-thuat) (*Algorithms*)  là một [tập hợp hữu hạn](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_h%E1%BB%A3p#L%E1%BB%B1c_l%C6%B0%E1%BB%A3ng_c%E1%BB%A7a_t%E1%BA%ADp_h%E1%BB%A3p_-_H%E1%BB%AFu_h%E1%BA%A1n_v%C3%A0_v%C3%B4_h%E1%BA%A1n) các hướng dẫn [được xác định rõ ràng](https://vi.wikipedia.org/w/index.php?title=%C4%90%C6%B0%E1%BB%A3c_x%C3%A1c_%C4%91%E1%BB%8Bnh_r%C3%B5&action=edit&redlink=1), có thể thực hiện được bằng máy tính, thường để giải quyết một lớp vấn đề hoặc để thực hiện một phép tính. Các giải thuật luôn [rõ ràng](https://vi.wikipedia.org/w/index.php?title=T%E1%BB%91i_ngh%C4%A9a&action=edit&redlink=1) và được sử dụng chỉ rõ việc thực hiện các [phép tính](https://vi.wikipedia.org/w/index.php?title=Ph%C3%A9p_t%C3%ADnh&action=edit&redlink=1), [xử lý dữ liệu](https://vi.wikipedia.org/wiki/X%E1%BB%AD_l%C3%AD_d%E1%BB%AF_li%E1%BB%87u), [suy luận tự động](https://vi.wikipedia.org/w/index.php?title=Suy_lu%E1%BA%ADn_t%E1%BB%B1_%C4%91%E1%BB%99ng&action=edit&redlink=1) và các tác vụ khác. Nói cách khác, giải thuật cung cấp một bộ các quy tắc hay quy trình giải quyết một vấn đề nào đó sau một số bước hữu hạn xuất phát từ một tập hợp các dữ kiện cho trước. Trạng thái xuất phát (dữ liệu đầu vào) của giải thuật thường được kí hiệu là INPUT, trạng thái đạt được (kết quả đầu ra) là OUTPUT.

Từ tiếng Anh của giải thuật là *Algorithm* (là tên gọi Latin hóa của *Al-Khwarizmi* nhà toán học vùng *Vịnh Ba Tư* nửa đầu thế kỉ thứ IX) ban đầu được sử dụng để chỉ các bộ quy tắc và kỹ thuật được *Al-Khwarizmi* sử dụng để giải các phương trình đại số, trước khi được khái quát hóa để chỉ bất kỳ bộ quy tắc hoặc kỹ thuật nào.

Sự phát triển của lý thuyết giải thuật bắt đầu bằng chứng minh của *Kurt Gödel* các định lý về tính không đầy đủ của các hệ thống tiên đề (định lý đầu tiên công bố vào năm 1931) đòi hỏi cần phải chuẩn hóa khái niệm giải thuật (Định lý bất toàn *Gödel*). Các phiên bản chuẩn hóa đầu tiên của khái niệm này được phát triển vào những năm 1930 bởi *Alan Turing*, *Emil Post* và *Alonzo Church*. Sau đó, lý thuyết giải thuật đã nhận được sự phát triển hơn nữa trong các công trình của *Stephen Cole Kleene*, *Andrey Andreyevich Markov* và những người khác. Vào năm 1954 *Andrey Andreyevich Markov* đề xuất tinh chỉnh khái niệm giải thuật bằng phương tiện mà ông gọi là giải thuật bình thường, theo đánh giá được coi là “*một trong những phiên bản tiêu chuẩn hóa thành công nhất của giải thuật*”. Cách tiếp cận chung nhất khái niệm giải thuật được đề xuất bởi *Andrey Nikolaevich Kolmogorov* vào năm 1965. Trong những năm tiếp theo, *Donald Knuth*, *Alfred Aho* và *Jeffrey Ullman* đã đóng góp đáng kể cho lý thuyết giải thuật.

Giải thuật có thể được mô tả như một thủ tục hoặc công thức để giải quyết vấn đề. Chính vì vậy giải thuật được sử dụng rộng rãi trong các lĩnh vực khác nhau, đặc biệt trong toán học và khoa học máy tính, cũng như trong cuộc sống hàng ngày.

Riêng trong lĩnh vực toán học, ngoài các ứng dụng cụ thể trong đại số, giải tích, lý thuyết số, khái niệm giải thuật là cơ sở cho một trong những khái niệm trung tâm của logic toán học, cụ thể là khái niệm tính toán. Đó là lý do tại sao, định lý *Gödel* về tính không đầy đủ của các hệ thống tiên đề có thể được coi là hệ quả của các định lý của lý thuyết giải thuật. Lý thuyết giải thuật cũng có mối liên kết chặt chẽ với chính nền tảng của toán học, trong đó mối quan hệ giữa tính kiến thiết (Constructive) và không kiến thiết (Non-Constructive) chiếm một vị trí trung tâm. Có thể nói, lý thuyết giải thuật cung cấp các công cụ cần thiết để phát triển xu hướng mang tính kiến thiết trong toán học.

Trong Khoa Học Máy Tính, năm 1965, *Andrey Nikolaevich Kolmogorov* đã đề xuất sử dụng lý thuyết giải thuật làm cơ sở cho lý thuyết thông tin. Lý thuyết giải thuật cũng là nền tảng lý thuyết cho một số vấn đề của toán học tính toán và có liên quan chặt chẽ với điều khiển học, trong đó việc nghiên cứu các giải thuật điều khiển đóng vai trò quan trọng. Cuối cùng, cần phải nói, giải thuật là bước không thể thiếu trong quá trình giải quyết bất kì bài toán nào trên máy tính, do vậy khái niệm giải thuật là một trong các khái niệm cơ bản cần biết đối với bất kì ai muốn lập trình cho máy tính.

Khi một giải thuật đã hình thành, ta không xét đến việc chứng minh tính đầy đủ và tính đúng đắn của giải thuật đó mà chỉ chú trọng đến việc áp dụng các bước theo đúng trình tự quy định để thu được kết quả cần thiết. Việc chứng minh ấy phải được tiến hành trong quá trình thiết kế giải thuật (Thiết kế giải thuật). Nói cách khác, giải thuật chỉ là việc áp dụng các công thức hay quy tắc, quy trình đã được công nhận là đúng hay đã được chứng minh về mặt toán học.

1. **Các đặc trưng của Giải Thuật**

Một đối tượng trong tự nhiên đều có các đặc trưng riêng của đối tượng đó, dự trên các đặc trưng đó chúng ta có thể xác định được đối tượng đó là gì. Giải thuật cũng không ngoại lệ, một giải thuật sẽ có các đặc trưng và dự trên các đặc trưng đó chúng ta có thể nhận biết được đối tượng đó có thật sự là một giải thuật, không phải thủ tục nào cũng được xem là giải thuật, mà phải có các đặc trưng sau:

* **Tính xác định**:Mộtgiải thuật nên có một cấu trúc rõ ràng và không mơ hồ. Mỗi một giai đoạn (hay mỗi bước) nên rõ ràng và chỉ mang một mục đích nhất định.
* **Dữ liệu đầu vào xác định**:Một giải thuật nên có không hoặc nhiều hơn dữ liệu đầu vào đã xác định.
* **Kết quả đầu ra**:Một giải thuật nên có một hoặc nhiều dữ liệu đầu ra đã xác định và nên kết nối với kiểu kết quả bạn mong muốn.
* **Tính dừng**:Các giải thuật phải kết thúc sau một số hữu hạn các bước.
* **Tính hiệu quả**: Một giải thuật nên là có thể thi hành được với các nguồn có sẵn, có nghĩa là có khả năng giải quyết hiệu quả vấn đề trong điều kiện thời gian và tài nguyên cho phép.
* **Tính phổ biến**:Một giải thuật có tính phổ biến nếu giải thuật này có thể giải quyết được một lớp các vấn đề tương tự.
* **Độc lập**:Một giải thuật nên có các chỉ thị độc lập với bất kỳ phần code lập trình nào.

1. **Các giải thuật phổ biến**
2. **Giải thuật tìm kiếm**

Giải thuật tìm kiếm là quy trình từng bước để tìm một phần tử cụ thể trong cấu trúc dữ liệu. Các giải thuật tìm kiếm được sử dụng trong nhiều ứng dụng, bao gồm tìm kiếm trên web, hệ thống cơ sở dữ liệu và trí tuệ nhân tạo. Có nhiều loại giải thuật tìm kiếm khác nhau, mỗi loại đều có điểm mạnh và điểm yếu riêng. Các giải thuật tìm kiếm phổ biến nhất bao gồm:

* ***Linear search*:** Giải thuật này chỉ đơn giản kiểm tra từng phần tử trong cấu trúc dữ liệu theo thứ tự cho đến khi tìm thấy phần tử đích. Tìm kiếm tuyến tính dễ thực hiện nhưng cũng là giải thuật tìm kiếm kém hiệu quả nhất, đặc biệt đối với các cấu trúc dữ liệu lớn.
* ***Binary search*:** Giải thuật này chỉ hoạt động trên các cấu trúc dữ liệu được sắp xếp. Giải thuật hoạt động bằng cách chia đôi cấu trúc dữ liệu và so sánh phần tử đích với phần tử ở giữa. Nếu phần tử đích nhỏ hơn phần tử ở giữa thì việc tìm kiếm sẽ tiếp tục ở nửa dưới của cấu trúc dữ liệu. Nếu không, việc tìm kiếm sẽ tiếp tục ở nửa trên của cấu trúc dữ liệu. Tìm kiếm nhị phân hiệu quả hơn nhiều so với tìm kiếm tuyến tính, đặc biệt đối với các cấu trúc dữ liệu lớn.
* ***Hashing*:** Giải thuật này hoạt động bằng cách tạo bảng băm, đây là cấu trúc dữ liệu ánh xạ các khóa tới các giá trị. Để tìm kiếm một phần tử, giải thuật băm chuyển phần tử đó thành một khóa và sau đó tra cứu khóa đó trong bảng băm. Nếu tìm thấy khóa thì giá trị tương ứng sẽ được trả về. Ngược lại, phần tử không được tìm thấy. Băm là giải thuật tìm kiếm hiệu quả nhất cho cấu trúc dữ liệu lớn, nhưng đòi hỏi nhiều bộ nhớ hơn các giải thuật tìm kiếm khác.

Giải thuật tìm kiếm tốt nhất để sử dụng tùy thuộc vào ứng dụng cụ thể. Ví dụ: nếu cấu trúc dữ liệu nhỏ và chưa được sắp xếp thì tìm kiếm tuyến tính có thể là giải pháp đơn giản và hiệu quả nhất. Nếu cấu trúc dữ liệu lớn và được sắp xếp thì tìm kiếm nhị phân là giải pháp hiệu quả nhất. Nếu cấu trúc dữ liệu lớn và chưa được sắp xếp thì băm là giải pháp hiệu quả nhất.

Ngoài 3 giải thuật tìm kiếm nêu trên còn có rất nhiều loại giải thuật tìm kiếm khác như:

* *Jump search*
* *Interpolation search*
* *Exponential search*
* *Fibonacci search*
* *Depth – first search*
* *Breadth – first search*

1. **Giải thuật sắp xếp**

Giải thuật sắp xếp là giải thuật sắp xếp một tập hợp các mục theo một thứ tự cụ thể. Thứ tự phổ biến nhất là thứ tự số tăng dần hoặc giảm dần, nhưng giải thuật sắp xếp cũng có thể được sử dụng để sắp xếp các loại dữ liệu khác, chẳng hạn như chuỗi hoặc đối tượng. Các giải thuật sắp xếp được sử dụng trong rất nhiều ứng dụng, bao gồm:

* ***Database systems*:** Các giải thuật sắp xếp được sử dụng để sắp xếp các bản ghi cơ sở dữ liệu nhằm giúp hệ thống dễ dàng tìm kiếm và truy xuất hơn.
* ***Website search*:** Giải thuật sắp xếp được sử dụng để sắp xếp kết quả tìm kiếm nhằm cung cấp kết quả phù hợp nhất cho người dùng.
* ***Machine learning*:** Các giải thuật sắp xếp được sử dụng để sắp xếp dữ liệu huấn luyện nhằm huấn luyện các mô hình học máy.
* ***Data visualization*:** Các giải thuật sắp xếp được sử dụng để sắp xếp dữ liệu trước khi được hiển thị trực quan để dễ hiểu hơn.

Có nhiều loại giải thuật sắp xếp khác nhau, mỗi loại đều có điểm mạnh và điểm yếu riêng. Các giải thuật sắp xếp phổ biến nhất bao gồm:

* ***Selection sort*:** Giải thuật này hoạt động bằng cách tìm phần tử nhỏ nhất trong danh sách và hoán đổi với phần tử đầu tiên. Sau đó, tìm phần tử nhỏ thứ hai trong danh sách và hoán đổi với phần tử thứ hai, v.v. Sắp xếp lựa chọn rất đơn giản để thực hiện, nhưng cũng là giải thuật sắp xếp kém hiệu quả nhất.
* ***Insertion sort*:** Hoạt động bằng cách chèn từng phần tử vào danh sách đã sắp xếp vào đúng vị trí. Sắp xếp chèn hiệu quả hơn sắp xếp lựa chọn, nhưng vẫn không hiệu quả lắm đối với các danh sách lớn.
* ***Merge sort*:** Giải thuật này hoạt động bằng cách chia danh sách làm đôi, sắp xếp đệ quy từng nửa, sau đó hợp nhất hai nửa đã sắp xếp lại với nhau. Sắp xếp hợp nhất là một giải thuật sắp xếp rất hiệu quả nhưng yêu cầu bộ nhớ bổ sung để lưu trữ hai nửa được sắp xếp của danh sách.
* ***Quick sort*:** Hoạt động bằng cách phân vùng danh sách xung quanh một phần tử trụ. Các phần tử nhỏ hơn phần tử trục được đặt ở một bên của trục và các phần tử lớn hơn phần tử trục được đặt ở phía bên kia của trục. Giải thuật quicksort sau đó được áp dụng đệ quy cho mỗi bên của trục xoay. Quicksort là giải thuật sắp xếp hiệu quả nhất cho hầu hết các đầu vào, nhưng có thể có độ phức tạp về thời gian trong trường hợp xấu nhất là O(n^2).

Việc lựa chọn giải thuật sắp xếp phụ thuộc vào một số yếu tố, bao gồm kích thước của danh sách, loại dữ liệu được sắp xếp và các đặc tính hiệu suất mong muốn. Giải thuật sắp xếp tốt nhất phụ thuộc vào ứng dụng cụ thể. Ví dụ: nếu danh sách nhỏ và chưa được sắp xếp thì sắp xếp lựa chọn hoặc sắp xếp chèn có thể là giải pháp đơn giản và hiệu quả nhất. Nếu danh sách lớn và được sắp xếp thì sắp xếp hợp nhất hoặc sắp xếp nhanh là giải pháp hiệu quả nhất. Nếu danh sách lớn và chưa được sắp xếp thì sắp xếp nhanh là giải pháp hiệu quả nhất, nhưng điều quan trọng là phải nhận thức được độ phức tạp về thời gian trong trường hợp xấu nhất. Ngoài ra còn có một số giải thuật sắp xếp khác chuyên biệt hơn cho một số loại dữ liệu hoặc ứng dụng nhất định. Ví dụ: sắp xếp cơ số là giải thuật sắp xếp rất hiệu quả cho số nguyên và sắp xếp đếm là giải thuật sắp xếp rất hiệu quả cho các danh sách nhỏ có phạm vi giá trị giới hạn. Nói chung, tốt nhất nên chọn giải thuật sắp xếp đã được nghiên cứu và triển khai kỹ lưỡng trong một thư viện tốt. Điều này sẽ giúp đảm bảo rằng giải thuật hiệu quả và chính xác.

1. **Giải thuật cập nhật**

Giải thuật cập nhật là một thủ tục sửa đổi trạng thái của hệ thống. Các giải thuật cập nhật được sử dụng trong rất nhiều ứng dụng, bao gồm:

* ***Software systems*:** Giải thuật cập nhật được sử dụng để cập nhật trạng thái của hệ thống phần mềm, chẳng hạn như cơ sở dữ liệu, hệ điều hành và ứng dụng web.
* ***Machine learning systems*:** Giải thuật cập nhật được sử dụng để cập nhật các tham số của mô hình học máy khi được huấn luyện trên dữ liệu mới.
* ***Control systems*:** Giải thuật cập nhật được sử dụng để cập nhật tín hiệu điều khiển của hệ thống điều khiển, chẳng hạn như robot và ô tô tự lái.

Có nhiều loại giải thuật cập nhật khác nhau, mỗi loại đều có điểm mạnh và điểm yếu riêng. Các giải thuật cập nhật phổ biến nhất bao gồm:

* ***Gradient descent*:** Giải thuật này được sử dụng để cập nhật các tham số của mô hình học máy. Hoạt động bằng cách tính toán liên tục độ dốc của hàm mất đối với các tham số và sau đó cập nhật các tham số theo hướng ngược lại của độ dốc.
* ***Backpropagation*:** Giải thuật này là một kiểu giảm độ dốc được sử dụng để huấn luyện mạng lưới thần kinh. Hoạt động bằng cách tính toán độ dốc của hàm mất mát tương ứng với các trọng số của mạng nơ-ron và sau đó cập nhật các trọng số theo hướng ngược lại của độ dốc.
* ***Kalman filter*:** Giải thuật này được sử dụng để cập nhật trạng thái của hệ thống khi có nhiễu. Hoạt động bằng cách kết hợp kiến thức tiên nghiệm về trạng thái của hệ thống với các phép đo của hệ thống để tạo ra ước tính cập nhật về trạng thái.
* ***Particle filter*:** Giải thuật này là một loại bộ lọc Kalman có thể được sử dụng để theo dõi nhiều mục tiêu cùng một lúc. Hoạt động bằng cách duy trì một tập hợp các hạt có trọng số, mỗi hạt đại diện cho một trạng thái có thể có của hệ thống. Giải thuật sau đó cập nhật trọng lượng của các hạt dựa trên các phép đo của hệ thống.

Việc lựa chọn giải thuật cập nhật phụ thuộc vào một số yếu tố, bao gồm loại hệ thống đang được cập nhật, sự hiện diện của nhiễu và các đặc tính hiệu suất mong muốn. Dưới đây là một số ví dụ về giải thuật cập nhật đang hoạt động:

* Hệ thống cơ sở dữ liệu sử dụng giải thuật cập nhật để cập nhật số dư tài khoản ngân hàng khi khách hàng gửi tiền hoặc rút tiền.
* Mô hình học máy sử dụng giải thuật cập nhật để cập nhật các tham số khi được đào tạo dựa trên dữ liệu mới về sở thích của khách hàng.
* Xe tự lái sử dụng giải thuật cập nhật để cập nhật các tín hiệu điều khiển khi lái xe trên đường và nhận thông tin mới từ các cảm biến.

Các giải thuật cập nhật là một phần thiết yếu của nhiều hệ thống hiện đại. Cho phép các hệ thống được sửa đổi và thích ứng với thông tin mới và các điều kiện thay đổi.

1. **Giải thuật xóa**

Giải thuật xóa là quy trình từng bước để xóa một phần tử khỏi cấu trúc dữ liệu. Các giải thuật xóa được sử dụng trong rất nhiều ứng dụng, bao gồm cơ sở dữ liệu, hệ điều hành và trình biên dịch. Giải thuật xóa cụ thể được sử dụng tùy thuộc vào loại cấu trúc dữ liệu đang được sửa đổi. Ví dụ: để xóa một phần tử khỏi một mảng, chúng ta chỉ cần dịch chuyển tất cả các phần tử tiếp theo sang trái để điền vào chỗ trống. Để xóa một phần tử khỏi danh sách liên kết, chúng ta cần tìm nút chứa phần tử cần xóa sau đó cập nhật con trỏ của các nút xung quanh để bỏ qua nút đã xóa.

Giải thuật này hoạt động bằng cách trước tiên tìm nút trước đó trong danh sách liên kết. Nếu nút cần xóa là nút đầu của danh sách liên kết thì con trỏ đầu chỉ được cập nhật tới nút tiếp theo trong danh sách. Ngược lại, con trỏ của nút trước đó sẽ được cập nhật để bỏ qua nút đã xóa. Giải thuật xóa có thể phức tạp hơn đối với các cấu trúc dữ liệu phức tạp hơn, chẳng hạn như cây tìm kiếm nhị phân và bảng băm. Tuy nhiên, nguyên tắc chung là như nhau: tìm phần tử cần xóa và sau đó cập nhật cấu trúc dữ liệu để phản ánh việc xóa.

# **Chương 2**

**GIẢI THUẬT DI TRUYỀN**

1. **DI TRUYỀN VÀ LỊCH SỬ CỦA DI TRUYỀN**
2. **Khái niệm về Di Truyền**

Di truyền là hiện tượng truyền đạt các đặc tính, tính cách, ngoại hình,... của bố mẹ, ông bà tổ tiên cho các thế hệ con, cháu. Lấy ví dụ người bố và đứa con có đôi tai rất giống nhau, có đôi mắt xanh dương giống nhau, có chiếc mũi cũng rất giống nhau,... thì được nhận định nôm na rằng "Bố đã di truyền những đặc điểm này cho con mình", hoặc "Đứa con đã được di truyền những đặc điểm của bố".

Trong sinh học và di truyền học, di truyền là quá trình di chuyển những đặc trưng sinh học từ một cá thể cha mẹ đến cá thể con cái và đồng nghĩa với việc di chuyển các gen, gen thừa nhận mang thông tin sinh học (hay thông tin di truyền). Ngoài ra, các đặc điểm về tính cách, nhận thức và tư duy của con cái có thể được tiếp nhận từ cha mẹ thông qua môi trường sinh hoạt gia đình (các thói quen, quy định của gia đình gọi là gia phong, nề nếp). Ở con người, xác định đặc trưng nào phụ thuộc vào di truyền và đặc trưng nào phụ thuộc vào môi trường thường gây tranh cãi; đặc biệt là đối với những đặc tính phức tạp như trí thông minh và màu da; giữa tự nhiên và nuôi dưỡng.

Di truyền, là tổng hợp của tất cả các quá trình sinh học mà qua đó các đặc điểm cụ thể được truyền từ cá thể cha mẹ sang cá thể con cái của họ. Khái niệm di truyền bao gồm hai quan sát dường như nghịch lý về các sinh vật: đầu tiên là về sự bất biến của một loài từ thế hệ này sang thế hệ khác và kế tiếp về sự khác biệt giữa các cá thể trong một loài. Như đã trở nên rõ ràng trong nghiên cứu về di truyền học. Cả hai khía cạnh của di truyền có thể được giải thích bằng gen, các đơn vị chức năng của vật liệu di truyền được tìm thấy trong tất cả các tế bào sống. Mỗi thành viên của một loài có một bộ gen đặc trưng cho loài đó. Chính bộ gen này đã cung cấp cấu trúc của loài. Tuy nhiên, trong một số các cá thể trong một loài, các biến thể có thể xảy ra ở dạng mỗi gen, tạo cơ sở di truyền cho thực tế là không có hai cá thể nào có những đặc điểm giống hệt nhau.

Việc truyền các đặc tính di truyền từ bố mẹ sang con cái phụ thuộc vào sự phân ly và tái tổ hợp của các gen trong quá trình phân bào và thụ tinh và việc này dẫn đến sự hình thành một số cá thể mới tương tự các loài khác, nhưng biểu hiện một số biến thể nhất định do sự kết hợp của các gen cụ thể và sự tương tác của chúng với môi trường. Và các sinh vật, động vật cứ thế tiếp tục quá trình di truyền qua nhiều thế hệ sau.

1. **Lịch sử của Di Truyền**

Từ thời xa xưa, người ta đã nhận thức được rằng mọi sinh vật trên thế giới từ thực vật đến động vật đều thể hiện ít nhiều những đặc tính từ cha và mẹ; và khác biệt giữa con và cha mẹ cũng thường được truyền lại cho thế hệ cháu. Trên cơ sở đó, con người dò đoán và tìm được phương pháp tuyển chọn tự nhiên và phát triển giống tốt cho gia súc cũng như rau cải, lúa gạo, v.v.... Hiện tượng di truyền được con người chấp nhận như chuyện hiển nhiên cho đến cuối thế kỷ XIX mới có giải thích khoa học hơn. Nhóm nghiên cứu di truyền theo chủ thuyết của *Jean-Baptiste Lamarck* cho rằng ngoài di truyền của những biểu hiện bẩm sinh còn có di truyền của những biểu hiện mắc phải bởi môi trường.

*Charles Darwin* đưa lên giả thuyết về tiến hoá vào năm 1859 nhưng gặp một số khúc mắc, khó khăn nhất là giải thích phương thức của di truyền. *Charles Darwin* cho rằng có pha trộn giữa di truyền bẩm sinh và di truyền của những biểu hiện gây nên bởi môi trường. Nhưng nếu thật sự có sự pha trộn này, chỉ sau một vài thế hệ sẽ nảy sinh ra hiện tượng đồng dạng của toàn chủng và sẽ không có đủ biến dị để sự tuyển chọn tự nhiên có thể xảy ra. Do đó, Darwin phải tiếp thu phần nào giả thuyết của *Jean-Baptiste Lamarck* vào công trình nghiên cứu của mình. Cách trình bày của *Charles Darwin* về di truyền là cho thấy xảy ra như thế nào và người ta có thể dự đoán hướng di truyền trong thế hệ tới (*ví dụ những đặc tính được di truyền nhưng không biểu hiện ở cha hay mẹ vào lúc thụ thai, nhiều đặc tính được di truyền phân biệt theo giống đực hay cái*). Với phương thức của di truyền này thì *Charles Darwin* không giải thích được.

Khái niệm di truyền của *Charles Darwin* được người anh em bà con của ông *Francis Galton* cải tiến sau đó, từ đó tạo cơ sở cho nền tảng khoa học di truyền. Tuy nhiên *Francis Galton* không chấp nhận thuyết tiến hóa toàn diện của *Charles Darwin* đó là về thuyết Pangenesis: là một cơ chế giả thuyết về di truyền được đề xuất bởi *Charles Darwin* trong cuốn sách The Variation of Animals and Plants under Domestication (Sự biến đổi của động vật và thực vật được thuần hóa) năm 1868. *Charles Darwin* cho rằng mỗi bộ phận của cơ thể phát ra các hạt nhỏ gọi là *Gemmules*, chúng lưu thông khắp cơ thể và cuối cùng tập trung lại ở các cơ quan sinh sản. Những hạt *Gemmules* này sau đó được truyền lại cho con cái và định hướng sự phát triển của các bộ phận tương tự trên cơ thể ở thế hệ tiếp theo.

Năm 1880, *August Friedrich Leopold Weismann* cắt đuôi của nhiều thế hệ chuột trong phòng thử nghiệm và cho thấy các con chuột trong thế hệ sau vẫn có đuôi. Từ đó, ông chứng minh rằng không có sự di truyền của các tính trạng mắc phải trong cuộc sống.

1. **Các thành phần trong quy trình Di Truyền**

Để một cá thể có thể nhận những đặc điểm di truyền từ ông bà, ba mẹ tổ tiên của mình thì cần đến quá trình biến đổi qua nhiều giai đoạn, mỗi giai đoạn mang một ý nghĩa riêng để có thể giúp cho các cá thể đó thừa hưởng những đặc điểm của ông bà, ba mẹ tổ tiên hoặc có thể các cá thể đó nhận được thêm sự đột biến riêng biệt mà thiên nhiên ban tặng. Để có thể hiểu một cách rõ ràng hơn chúng ta sẽ lấy ví dụ của nhà sinh học người Anh *Charles Darwin* về sự xuất hiện của Hươu cao cổ, *Charles Darwin* đưa ra giải thuyết rằng: *“Trong quần thể Hươu vốn đã tồn tại những con Hươu có cổ cao hơn bình thường nhờ gen di truyền và sự đột biến. Trải qua quá trình sinh sống và phát triển, môi trường thay đổi khiến cho thức ăn càng ngày càng khó kiếm hơn, khiến những con Hươu có chiếc cổ cao sẽ chiếm ưu thế sinh tồn hơn. Lâu dần thì thế hệ Hươu mới sẽ được thay bằng những con Hươu cao cổ có khả năng sinh sản và thích nghi với môi trường lớn hơn”*. Dựa trên giả thuyết Hươu cao cổ của *Charles Darwin*, chúng ta có thể nhận thấy các thành phần cấu thành quá trình di truyền, đột biến gồm các thành phần sau: Quần thể, chọn lọc tự nhiên, đột biến và tiến hóa.

1. **Population**

Quần thể trong tự nhiên vô cùng đa dạng, khác về số lượng, về loài. Nhưng thường giữa các quần thể với nhau luôn luôn có điểm chung đó là tính di truyền, để một quần thể có thể tồn tại trong tự nhiên thì di truyền là yếu tố không thể thiếu, một quần thể tồn tại theo năm tháng dĩ nhiên sẽ có sự khác biệt với các quần thể trước đó để có được sự khác biệt đó quá trình di truyền luôn được vận hành liên tục trong môi trường tự nhiên để thúc đẩy sự phát triển của các loài, các quần thể.

A group of lions walking on a path in tall grass

Description automatically generated

*Hình 1. Hình về quần thể sư tử hoang dã trong tự nhiên.*

Nguyên nhân dẫn đến sự biến mất vĩnh viễn của một số quần thể trong tự nhiên là do khí hậu quá khắc nghiệt, môi trường sống không đủ đáp ứng hoặc quần thể đó không thể thích nghi, các đối thủ săn mồi trong tự nhiên và cả việc thức ăn của quần thể đó khan hiếm, nhưng những điều này cũng là một phần của thế giới tự nhiên.

1. **Natural Selection**

Quá trình chọn lọc tự nhiên là một bước rất quan trọng đối với sự phát triển của các quần thể trong tự nhiên, quá trình này có thể là môi trường xung quanh, khí hậu, thời tiết, các biến cố trong tự nhiên. Chọn lọc tự nhiên sẽ chọn ra các cá thể tốt nhất, có tính thích nghi cao đối với môi trường xung quanh, như vậy đồi hỏi các cá thể trong một quần thể phải có các thế mạnh riêng để có thể tiếp tục tồn tại trong môi trường tự nhiên, quá trình chọn lọc tự nhiên sẽ luôn luôn được tiếp diễn trong toàn thế giới tự nhiên, việc này dẫn đến một số quần thể không thể thích nghi tốt dần dần theo thời gian biến mất khỏi tự nhiên có thể nói là tuyệt chủng, nhưng ngược lại nhờ quá trình này mà có các quần thể được cơ hội tiến hóa, mạnh mẽ hơn, có tính thích nghi cao hơn từ đó sinh ra con cái có các bộ gen tốt và tiếp tục được phát triển bởi các thế hệ sau.

A silhouette of a person in a line

Description automatically generated

*Hình 2. Mô phỏng loài người trải qua các giai đoạn tiến hóa*

Hình bên trên cho chúng ta thấy được quá trình tiến hóa của loài người được mô phỏng tóm tắt, nhờ quá trình chọn lọc của tự nhiên mà loài người được cơ hội để phát triển các đặc điểm như thể chất, cảm xúc, tư duy, tin thần..., có thể khẳng định vai trò đóng góp, thúc đẩy sự phát triển của quá trình chọn lọc tự nhiên cho mọi quần thể trong tự nhiên là không thể thiếu cho muôn loài trong tự nhiên.

1. **Mutation**

Đột biến là quá trình thay đổi trình tự của DNA của một cá thể tronng tự nhiên, xảy ra việc đột biến có thể do lỗi trong quá trình di truyền từ cá thể truyền đi sang cá thể thừa hưởng, đột biến xảy ra ở các bộ gen, kể cả một gen đơn lẻ. Đột biến đôi khi không mang đến sự thay đổi cụ thể nào, khó có thể nhìn thấy được các đột biến như dạng này ngược lại có một số loại đột biến lại ảnh hưởng rất lớn đến một cá thể, sự ảnh hưởng này có thể là tích cực hoặc tiêu cực. Một số dạng đột biến thường được biết đến:

* ***Point mutations***

Đột biến điểm là một loại đột biến gen xảy ra khi một bazơ nucleotide đơn lẻ bị thay đổi, chèn hoặc xóa khỏi chuỗi DNA. Đột biến điểm có thể xảy ra ở bất kỳ gen nào và có thể gây ra nhiều tác động khác nhau, tùy thuộc vào gen bị ảnh hưởng và mức độ nghiêm trọng của đột biến. Một số đột biến điểm là vô hại, trong khi những đột biến khác có thể gây ra các vấn đề sức khỏe nghiêm trọng. Ví dụ, đột biến điểm ở gen liên quan đến sự phát triển và phân chia tế bào có thể làm tăng nguy cơ ung thư. Đột biến điểm cũng có thể gây ra các rối loạn di truyền như bệnh hồng cầu hình liềm và bệnh xơ nang.

***A diagram of a sequence of dna

Description automatically generated***

*Hình 3. Ảnh minh họa về Point Mutations*

* ***Frame – shift mutations***

Đột biến dịch khung là một loại đột biến gen xảy ra khi một hoặc nhiều nucleotide được chèn vào hoặc xóa khỏi chuỗi DNA. Điều này có thể làm thay đổi khung đọc của DNA, đó là cách trình tự được đọc theo nhóm gồm ba nucleotide được gọi là codon. Mỗi codon mã hóa cho một axit amin cụ thể, do đó đột biến dịch khung có thể phá vỡ trình tự axit amin trong protein, dẫn đến protein không có chức năng hoặc bị biến đổi. Đột biến dịch khung có thể xảy ra một cách tự nhiên hoặc do các yếu tố môi trường gây ra như tiếp xúc với tia cực tím hoặc hóa chất. Chúng cũng là đặc điểm chung của một số loại ung thư, chẳng hạn như ung thư đại trực tràng và ung thư phổi.

Đột biến Frameshift có thể gây ra nhiều tác động khác nhau, tùy thuộc vào gen bị ảnh hưởng và mức độ nghiêm trọng của đột biến. Một số đột biến dịch chuyển khung hình có thể không gây ra triệu chứng nào đáng chú ý, trong khi những đột biến khác có thể dẫn đến các vấn đề sức khỏe nghiêm trọng. Ví dụ, đột biến dịch khung ở các gen liên quan đến sự phát triển và phân chia tế bào có thể làm tăng nguy cơ ung thư. Không có cách chữa trị cho đột biến dịch khung, nhưng có các phương pháp điều trị để kiểm soát các triệu chứng rối loạn di truyền do đột biến dịch khung gây ra. Ví dụ, một số rối loạn di truyền có thể được điều trị bằng liệu pháp thay thế enzyme, liệu pháp gen hoặc các liệu pháp nhắm mục tiêu nhắm vào protein cụ thể bị ảnh hưởng bởi đột biến.

* ***Chromosomal mutations***

Đột biến nhiễm sắc thể là sự thay đổi về số lượng hoặc cấu trúc của nhiễm sắc thể. Nhiễm sắc thể là những cấu trúc dạng sợi có chứa DNA, vật liệu di truyền của mọi sinh vật sống. Đột biến nhiễm sắc thể có thể xảy ra một cách tự nhiên hoặc do các yếu tố môi trường gây ra như tiếp xúc với bức xạ hoặc hóa chất.

Đột biến nhiễm sắc thể có thể gây ra nhiều tác động khác nhau, tùy thuộc vào loại đột biến và gen bị ảnh hưởng. Một số đột biến nhiễm sắc thể có thể không gây ra triệu chứng đáng chú ý, trong khi những đột biến khác có thể dẫn đến các vấn đề sức khỏe nghiêm trọng, chẳng hạn như chậm phát triển, dị tật bẩm sinh và ung thư.

1. **Evolution**

Tiến hóa là quá trình di truyền các đặc tính của tổ tiên sang các thế hệ con cháu trong quần thể tự nhiên, các đặc tính được di truyền sang các thế hệ sau sẽ có sự khác biệt, thích ứng với môi trường xung quanh, khí hậu,... Những cá thể nhận được các đặc tính di truyền sẽ vẫn giữ được các đặc tính giống với tổ tiên nhưng theo thời gian thông qua quá trình di truyền các đặc tính này sẽ được cá thể thể hiện một cách khác. Nguyên nhân dẫn đến quá trình tiến hóa một phần là do sự đột biến ở một cá thể hoặc cả một quần thể, nhưng đột biến thường khó đoán trước bởi vì không đảm bảo được cá thể đó sẽ nhận được bộ gen tốt hay bộ gen xấu, còn tiến hóa thì hướng đến một cá thể hoặc một quần thể tốt hơn, có thể hiểu quá trình tiến hóa là quá trình đột biến theo hướng tốt.

A bird sitting on a branch

Description automatically generated

*Hình 4. Kingfishers loài chim tiến hóa mỏ dài để bắt cá*

1. **GIẢI THUẬT DI TRUYỀN**
2. **Khái niệm về Giải Thuật Di Truyền**

Giải thuật di truyền là một kỹ thuật trong chuyên ngành Khoa Học Máy Tính nhằm tìm kiếm giải pháp thích hợp nhất trong tập hợp các giải pháp cho bài toán tối ưu tổ hợp (Combinatorial Optimization). Giải Thuật Di Truyền cũng như các giải thuật tiến hóa nói chung, hình thành dựa trên quan niệm cho rằng, quá trình tiến hóa tự nhiên là quá trình hoàn hảo nhất, hợp lý nhất và chính bản thân đã mang tính tối ưu nhất. Quan niệm này có thể được xem như một tiên đề đúng, không thể chứng minh được, nhưng rất phù hợp với thực tế khách quan. Quá trình tiến hóa thể hiện tính tối ưu ở chỗ, thế hệ sau luôn luôn bao giờ cũng tốt hơn, phát triển hơn, hoàn thiện hơn thế hệ trước. Tiến hóa tự nhiên được duy trì nhờ hai quá trình cơ bản sau: sinh sản và chọn lọc tự nhiên. Xuyên suốt quá trình tiến hóa tự nhiên, các thế hệ mới luôn được sinh ra để bổ sung thay thế các thế hệ cũ. Cá thể nào phát tiển hơn, thích ứng hơn với môi trường sẽ được tồn tại còn cá thể nào không thích ứng được với môi trường sẽ bị đào thải. Sự thay đổi môi trường là động lực thúc đẩy quá trình tiến hóa. Ngược lại, tiến hóa cũng tác động trở lại góp phần làm thay đổi môi trường.

Các cá thể mới sinh ra trong quá trình tiến hóa nhờ sự lai ghép ở các thế hệ trước. Một cá thể mới có thể mang những đặc tính của ông bà, cha mẹ (gọi là di truyền), cũng có một số cá thể mang những tính trạng hoàn toàn mới (gọi là đột biến). Di truyền và đột biến là hai cơ chế có vai trò quan trọng như nhau trong quá trình tiến hóa, dù rằng khả năng đột biến ở một cá thể xảy ra với xác suất nhỏ hơn rất nhiều so với khả năng di truyền. Các giải thuật tiến hóa tuy có những đặc điểm khác nhau, nhưng đều mô phỏng bốn quá trình cơ bản bao gồm: Fitness – Selection – Mating – Mutation.

1. **Lịch sử về Giải Thuật Di Truyền**

Lịch sử của giải thuật di truyền có thể bắt nguồn từ công trình của *John Holland* vào những năm 1960. *John Holland* quan tâm đến việc phát triển một mô hình về sự thích ứng xảy ra trong các hệ thống tự nhiên. Ông đề xuất rằng sự thích ứng có thể được giải thích bằng sự tương tác của ba yếu tố chính: biến thể, chọn lọc và kế thừa. Sự biến đổi đề cập đến những thay đổi ngẫu nhiên xảy ra trong gen. Chọn lọc đề cập đến quá trình các cá thể có gen thích nghi tốt hơn có nhiều khả năng sống sót và sinh sản hơn. Di truyền đề cập đến việc truyền gen từ cha mẹ sang con cái. *John Holland* đã chỉ ra rằng ba yếu tố này có thể dẫn đến sự xuất hiện của các quần thể ngày càng thích nghi tốt theo thời gian. Ông đã phát triển một khuôn khổ toán học để mô tả quá trình này mà ông gọi là định lý lược đồ.

Năm 1975, *John Holland* xuất bản cuốn sách mang tên *Adaptation in Natural and Artificial Systems* (Thích ứng trong các hệ thống tự nhiên và nhân tạo), trong đó mô tả công trình của ông về giải thuật di truyền. Cuốn sách này được coi là tác phẩm tiêu biểu về lĩnh vực này. Các nhà nghiên cứu khác bắt đầu khám phá việc sử dụng giải thuật di truyền để giải quyết các vấn đề trong thế giới thực vào cuối những năm 1970 và đầu những năm 1980. Một trong những ứng dụng ban đầu đáng chú ý nhất là thiết kế ăng-ten cho tàu vũ trụ Voyager 1. Giải thuật di truyền đã trở nên phổ biến rộng rãi hơn vào cuối những năm 1980 và đầu những năm 1990. Điều này một phần là do sự phát triển của máy tính nhanh hơn và sự sẵn có ngày càng tăng của phần mềm để thực hiện các giải thuật di truyền.

Ngày nay, giải thuật di truyền được sử dụng để giải quyết rất nhiều vấn đề trong nhiều lĩnh vực khác nhau, bao gồm kỹ thuật, tài chính và học máy. Dưới đây là dòng thời gian của một số sự kiện quan trọng trong lịch sử giải thuật di truyền:

* ***1962*:** John Holland xuất bản một bài báo về việc sử dụng giải thuật di truyền để giải các bài toán tối ưu hóa.
* ***1975*:** John Holland xuất bản cuốn sách *Adaptation in Natural and Artificial Systems*, trong đó mô tả chi tiết công trình của ông về giải thuật di truyền.
* ***1975*:** Giải thuật di truyền được sử dụng để thiết kế ăng-ten cho tàu vũ trụ Voyager 1.
* ***1985*:** David Goldberg xuất bản một cuốn sách có tên Giải thuật di truyền trong tìm kiếm, tối ưu hóa và học máy, giúp phổ biến lĩnh vực này.
* ***1992*:** John Koza xuất bản cuốn sách có tên Lập trình di truyền, trong đó mô tả một loại giải thuật di truyền mới có thể được sử dụng để phát triển các chương trình máy tính.
* ***Thập niên 1990*:** Giải thuật di truyền được sử dụng để giải quyết nhiều vấn đề trong nhiều lĩnh vực khác nhau, bao gồm kỹ thuật, tài chính và học máy.
* ***Thập niên 2000 đến nay*:** Các giải thuật di truyền tiếp tục được phát triển và áp dụng cho các bài toán mới.

1. **Các thành phần trong Giải Thuật Di Truyền**

Giải thuật di truyền là một kỹ thuật được lấy ý tưởng từ di truyền trong tự nhiên, đảm bảo được độ tin cậy cho giải thuật này, bởi vì trong di truyền tự nhiên sẽ không có sự tuyệt đối cho các cá thể được sinh ra, mọi cá thể ở thế hệ sau đều tốt hơn cá thể trước sẽ không có cá thể nào hoàn hảo và các cá thể sẽ được phát triển qua nhiều thế hệ. Nhưng vẫn có một số cá thể may mắn được đột biến sẽ có cơ hội tách biệt ra khỏi thành một loài mới, khả năng đột biến trong tự nhiên rất thấp và đột biến trên một cá thể cũng không đảm bảo rằng cá thể đó sẽ tốt hơn các cá thể trong quần thể. Trong giải thuật di truyền cũng như vậy, các giải pháp cho bài toán sẽ được xem như một cá thể, một tổ hợp các giải pháp sẽ được gọi là không gian giải pháp hoặc được xem như một quần thể các giải pháp.

Đánh giá độ thích nghi được xem như môi trường để kiểm tra giải pháp đó có tương thích với mục tiêu ban đầu đã đặt ra, các giải pháp sẽ phải trải qua bước đánh giá độ thích nghi này. Bước tiếp sẽ là chọn lọc ra các giải pháp tốt nhất cho bài toán. Kế tiếp sẽ thực hiện ghép đôi hai giải pháp ngẫu nhiên trong quần thể với nhau cho ra một giải pháp mới. Bước cuối sẽ là đột biến, đột biến các giải pháp một cách ngẫu nhiên như đã nêu ở trên bước đột biến này có thể sinh ra giải pháp tốt hoặc xấu, không thể kiểm soát được bước này. Sau đây sẽ đi sâu hơn từng thành phần của quá trình thực hiện của giải thuật di truyền để có thể hiểu rõ và thực tiễn hơn chúng ta lấy một demo nhỏ ứng dụng vào quy trình này là “*Tìm chuỗi mục tiêu được nhập ngẫu nhiên từ bàn phím*”.

Với demo này chúng ta sẽ dụng:

* Programming Language: Python version 3.11.6
* IDE: Visual Studio Code
* Module: random

1. **Initialization**

Initialization sẽ là bước đầu tiên trong giải thuật di truyền, bước này khởi tạo một quần thể giải pháp có kích thước xác định. Trong demo này việc khởi tạo một quần thể chuỗi khá đơn giản, bước đầu tiên khởi tạo một biến *Population\_Size* và kế tiếp gán giá trị cho biến này, trong demo này sẽ gán giá trị 100 với kiểu dữ liệu là *int*. Nghĩa là quần thể chuỗi đầu tiên được khởi tạo có 100 chuỗi.

Population\_Size : int = 100

1. **Population**

* **Gen**

Về mặt sinh học gen được biết đến là một đoạn xác định của phân tử acid nucleic có chức năng di truyền. Trong hầu hết các trường hợp, phân tử acid nucleic này là DNA. Đối với giải thuật di truyền gen được xem như một phân tử nhỏ của một giải pháp, phân tử đó được xác định là gì thì tùy thuộc vào mỗi bài toán, điều quan trọng chúng ta sẽ làm là thực hiện công việc xác định được đâu sẽ là phân tử tạo thành một giải pháp từ đó việc tìm được giải pháp cho bài toán sẽ dễ dàng hơn. Với demo “*Tìm chuỗi mục tiêu được nhập ngẫu nhiên từ bàn phím*” chúng ta sẽ xác định mục tiêu cần đạt được thông qua giải thuật di truyền sẽ là một chuỗi ngẫu nhiên được nhập, chuỗi này sẽ là tập hợp các chữ cái có dấu hoặc không có dấu, con số, ký tự đặc biệt.

Genes = '''aáàảãạâấầẩẫậăắằẳẵặ bcd đ eéèẻẽẹ êếềểễệ fgh iíìỉĩị jklmn oóòỏõọơớờởỡợôốồổỗộ pqrst uúùủũụưứừửữự vwx yýỳỷỹỵ z AÁÀẢÃẠÂẤẦẨẪẬĂẮẰẲẴẶ BCD Đ EÉÈẺẼẸ ÊẾỀỂỄỆ FGH IÍÌỈĨỊ JKLMN OÓÒỎÕỌƠỚỜỞỠỢÔỐỒỔỖỘ PQRST UÚÙỦŨỤƯỨỪỬỮỰ VWX YÝỲỶỸỴ Z 0123456789,.-;:\_!"#%&/()=?@${[]}'''

Với demo tìm chuỗi mục tiêu này việc khai báo một bộ gen gồm các chữ cái có dấu hoặc không có dấu, con số, ký tự đặc biệt là rất quan trọng, một bộ gen đầy đủ sẽ hỗ trợ rất nhiều trong việc tìm kiếm chuỗi mục tiêu, phải luôn đảm bảo rằng bộ gen đầy đủ, vì một số trường hợp nếu thiếu đi một gen mà gen đó được nhập từ bàn phím thì việc tìm kiếm sẽ diễn ra vô tận, vì gen khai báo đã thiếu không tìm thấy được.

* **Chromosome**

Trong sinh học nhiễm sắc thể được hiểu đơn giản là tập hợp các chuỗi gen liên kết lại với nhau, nhờ những bộ gen khác nhau mà một cá thể khi được sinh ra cũng mang cho mình những đặc điểm khác hoàn toàn với các cá thể khác, trong demo này chúng ta sẽ xem một nhiễm sắc thể như một cá thể.

A close-up of a number of rectangular objects

Description automatically generated

*Hình 5. Mô phỏng về Gen và Chromosome*

def Create\_Chromosome(self):  
        global Target  
        Chromosome\_Len = len(Target)  
        return [self.Genetic\_Mutation() for \_ in range(Chromosome\_Len)]

Demo tìm chuỗi mục tiêu, các chữ cái, con số, ký tự đặc biệt đóng vai trò là các gen riêng lẻ, khi quá trình tạo ra các nhiễm sắc thể cho quần thể đầu tiên thì các gen sẽ được gọi ngẫu nhiên để liên kết tạo thành một chuỗi (*nhiễm sắc thể*). Khởi tạo hàm Create\_Chromosome() để tạo ra các chuỗi ngẫu nhiên, biến Target sẽ được khởi tạo là toàn cục, biến Chromosome\_Len sẽ nhận giá trị là độ dài của chuỗi mục tiêu được nhập vào, sử dụng vòng lập For với tham số là Chromosome\_Len (*độ dài của chuỗi mục tiêu*).

Đối với hàm Genetic\_Mutation() gọi các ký tự ngẫu nhiên, từ đó tạo ra các chuỗi với độ dài thì như nhau nhưng sẽ khác về các ký tự.

* **Population**

Trong tự nhiên một quần thể được hiểu là tập hợp các cá thể cùng loài, không nhất thiết các cá thể đó phải sống thành bầy đàn mà có thể sống ở các khu vực khác nhau trên thế giới. Trong demo này chúng ta xem quần thể là tập hợp các chuỗi đã được tạo ra ngẫu nhiên với số lượng là 100 chuỗi, điểm chung của các cá thể này đều được tạo ra từ cùng một bộ gen là các chữ cái, con số, ký tự đặc biệt. Trong quá trình giải thuật đi tìm chuỗi mục tiêu quần thể ban đầu sẽ thay đổi qua từng giai đoạn của giải thuật di truyền, vì tìm ra chuỗi mục tiêu quần thể sẽ được đánh giá độ thích nghi của từng cá thể, lai tạo sinh ra cá thể mới, đột biến gen ký tự.

A row of rectangular objects

Description automatically generated

*Hình 6. Mô phỏng về Population*

Để khởi tạo một quần thể chuỗi trong demo này, chúng ta sử dụng lại hàm Create\_Chromosome() để tạo ra quần thể chuỗi với các chuỗi được tạo ra mang tính ngẫu nhiên, để các chuỗi được tạo ra mang tính ngẫu nhiên chúng ta cần đến hàm Genetic\_Mutation(), hàm này sẽ được trình bày trong phần sau.

def Create\_Chromosome(self):  
        global Target  
        Chromosome\_Len = len(Target)  
        return [self.Genetic\_Mutation() for \_ in range(Chromosome\_Len)

Kế tiếp dùng vòng lập For với tham số được đưa vào là Population\_Size ban đầu được gán giá trị 100, đồng nghĩa với việc quần thể chuỗi sẽ có 100 chuỗi với các gen ngẫu nhiên. Các chuỗi được tạo ra bởi hàm Create\_Chromosome() sẽ được thêm vào mảng Population bằng cách sử dụng hàm append().

for \_ in range(Population\_Size):   
        Chromosome = Individual.Create\_Chromosome()  
        Population.append(Individual(Chromosome))

1. **Fitness**

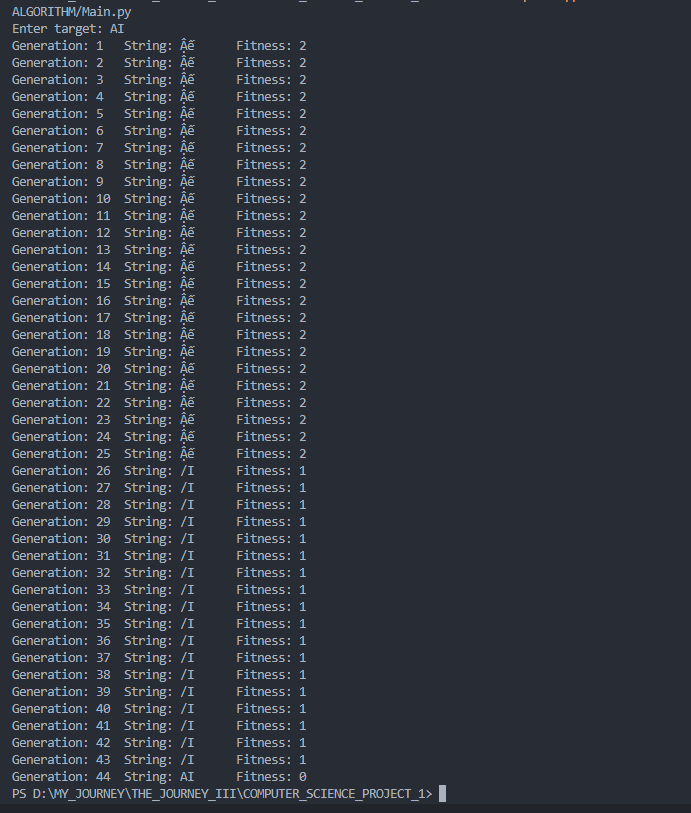
Tìm kiếm chuỗi mục tiêu để hoàn thành nhiệm vụ này thì giai đoạn Fitness không kém phần quan trọng so với các giai đoạn khác trong giải thuật di truyền. Fitness được hiểu là đánh giá độ thích thích nghi hay là độ tương thích của các cá thể trong quần thể. Có rất nhiều phương pháp khác nhau để thực hiện giai đoạn Fitness này, như có thể kể đến: Absolute fitness; Relative fitness; Pareto optimality. Với demo này sẽ được ứng dụng phương pháp Absolute fitness.

* **Absolute fitness**

Phương pháp này sẽ đánh giá độ tương thích của từng chuỗi có trong quần thể chuỗi, các chuỗi này sẽ được đánh giá dựa trên chuỗi mục tiêu bằng cách so sánh các gen (*chữ cái, con số, ký tự đặc biệt*) của hai chuỗi với nhau. Cần đến một hàm Fitness dựa trên phương pháp Absolute fitness để thực hiện nhiệm vụ này.

def Fitness(self):  
        global Target  
        fitness = 0  
        for fitness\_Chromosome, fitness\_Target in zip(self.chromosome, Target):  
            if fitness\_Chromosome != fitness\_Target:  
                fitness += 1  
        return fitness

Trong hàm Fitness sẽ khởi tạo biến fitness được gán giá trị mặc định là 0, vòng lập For sử dụng hai biến lần lượt là fitness\_Chromosome và fitness\_Target đại diện cho gen của chuỗi được so sánh và gen của chuỗi mục tiêu là self.chromosome và Target. Sử dụng hàm zip() để so sánh từng từ ký tự với nhau của hai chuỗi, với mỗi ký tự khác nhau giữa hai chuỗi thì biến fitness sẽ được cộng 1, nghĩa là chuỗi có điểm fitness càng cao thì độ tương thích với chuỗi mục tiêu càng thấp và ngược lại chuỗi có điểm fitness thấp thì lại tương thích với chuỗi mục tiêu. Quá trình so sánh sẽ kết thúc khi vòng lặp For so sánh hết các ký tự của hai chuỗi và trả về chuỗi đã được so sánh cùng với điểm fitness, dựa trên điểm fitness này chúng ta đánh giá được độ tương thích của từng chuỗi qua những lần so sánh với chuỗi mục tiêu.



*Hình 7. Ví dụ về hàm Fitness()*

1. **Selection**

Selection nghĩa là chọn lọc, giai đoạn này sẽ thực hiện công việc tìm ra các cá thể tốt nhất dựa trên điểm fitness, quá trình này được thực hiện nhiều lần nhầm tìm ra các cá thể tốt nhất để mang đi sinh sản các cá thể mới. Giải thuật di truyền có khá nhiều phương pháp chọn lọc khác nhau, trong đó có thể kể đến như: Roulette Wheel Selection, Tournament Selection, Rank – Based Selection, Stochatic Universal Sampling, Boltzmann Selection, Elitist Selection, Sigma Scaling, Truncation Selection. Để chọn một phương pháp phù hợp với bài toán cũng không phải dễ dàng, tùy độ phức tạp, tập dữ liệu của bài toán mà sẽ lựa chọn phương pháp phù hợp, demo tìm kiếm chuỗi mục tiêu sẽ kết hợp hai phương pháp Rank – Based Selection và Elitist Selection.

* **Rank – Based Selection**

Rank – Based Selection là một trong những phương pháp khá phổ biến trong giải thuật di truyền, phương pháp này sẽ chọn lọc các cá thể có điểm fitness tương đối cao, có thể hiểu chọn ra các cá thể nào ở đầu danh sách các cá thể có điểm fitness tốt nhất, điểm hay của phương pháp này là chọn nhiều cá thể tốt tránh việc bỏ soát. Trong demo chúng ta sẽ ứng dụng phương pháp Rank – Based Selection để chọn các chuỗi có độ tương thích cao so với chuỗi mục tiêu.

Population = sorted(Population, key = lambda x:x.fitness)

Population được khởi tạo trước đó và đã có các cá thể được tạo ra, nhiệm vụ là sắp xếp các chuỗi có điểm fitness thấp nhất cho đến cao nhất, vì trong demo này chuỗi có điểm fitness thấp nhất sẽ có độ tương thích với chuỗi mục tiêu nhất và ngược lại chuỗi có điểm fitness càng cao thì độ tương thích với chuỗi mục tiêu càng thấp. Mặc định của hàm sorted() là sắp xếp từ giá trị nhỏ nhất cho đến giá trị lớn nhất, tham số của hàm sẽ là Population, hàm lambda sẽ giúp cho hàm sorted() sắp xếp các chuỗi dựa trên điểm fitness.

* **Elitist Selection**

Mục tiêu của phương pháp này là luôn giữ được các cá thể tốt nhất cho các thế hệ sau, để thực hiện được việc đó Elitist Selection cần đến một số lượng hoặc phần trăm nhất định được xác định từ trước, ví dụ: “*Một quần thể có 100 cá thể, thì Elitist sẽ lấy 20% cá thể tốt nhất từ quần thể đó để thực hiện các quá trình khác trong giải thuật di truyền*”.

New\_Generation = []  
Size = int((10 \* Population\_Size) / 100)  
New\_Generation.extend(Population[:Size])

Đầu tiên khởi tạo mảng New\_Generation (*thế hệ mới*), biến Size sẽ mang giá trị là 10% chuỗi đầu tiên trong mảng Population đã được sắp xếp trong thứ tự của chuỗi bằng phương pháp Rank – Based Selection trước đó, sử dụng hàm extend() thêm 10% chuỗi đó vào mảng New\_Generation.

1. **Crossover**

Crossover (*lai tạo*) là một quá trình quan trọng trong giải thuật di truyền, mục tiêu của quá trình này là tạo ra cá thể dựa trên những thông tin hoặc gen của hai hoặc nhiều cá thể đã có sẵn (*được gọi là bố mẹ*). Trong Crossover lại chia ra nhiều phương pháp khác nhau như: Single – Point Crossover, Tow – Point Crossover, Uniform Crossover, Multi – Point Crossover, Arithmetic Crossover, Order Crossover, Cycle Crossover, Edge Recombination Crossover. Demo “Tìm kiếm chuỗi mục tiêu được nhập từ bàn phím” sẽ được áp dụng phương Uniform Crossover.

Uniform Crossover là phương pháp lai tạo dựa trên xác suất ngẫu nhiên, đầu tiên chọn ra hai cá thể tốt từ quần thể, dựa trên xác suất ngẫu nhiên mà giải thuật sẽ quyết định gen của bố hoặc mẹ sẽ được chèn vào gen của cá thể con, quá trình này sẽ được lập lại cho đến khi tạo ra cá thể con hoàn chỉnh có gen của bố hoặc mẹ.

def Crossover(self, Parents):  
        Child\_Chromosome = []   
        for Parent\_1, Parent\_2 in zip(self.chromosome, Parents.chromosome):  
            Probability = random.random()  
            if Probability < 0.45:  
                Child\_Chromosome.append(Parent\_1)  
            elif Probability < 0.90:  
                Child\_Chromosome.append(Parent\_2)  
            else:   
                Child\_Chromosome.append(self.Genetic\_Mutation())  
        return Individual(Child\_Chromosome)

Trong demo tìm kiếm chuỗi mục tiêu cũng tương tự, đầu tiên khởi tạo hàm Crossover() có một tham số là Parents, mảng Child\_Chromosome sẽ chứa các cá thể con đã được lai tạo, dùng vòng lập For để duyệt qua hai chuỗi bố mẹ, hai biến Parent\_1 và Parent\_2 đại diện cho các ký tự của self.chromosome , Parents.chromosome, xác suất được tạo ra ngẫu nhiên bởi biến Probability với giá trị gọi từ hàm random(). Nếu như Probability nhỏ hơn 0.45 thì giải thuật sẽ dùng hàm append() để chèn Parent\_1 vào biến Child\_Chromosome hoặc Probability nhỏ hơn 0.90 thì giải thuật sẽ chèn Parent\_2 vào biến Child\_Chromosome và ngược lại hai trường hợp trên thì giải thuật sẽ gọi hàm Genetic\_Mutation() để đột biến một ký tự ngẫu nhiên và chèn ký tự đó vào biến Child\_Chromosome. Quá trình lai tạo sẽ kết thức khi vòng lập For duyệt hết hai chuỗi.

1. **Mutation**

Cũng như Selection hay Crossover thì Mutation là quá trình quan trọng không kém, đột biến trong giải thuật di truyền là một quá trình thiết yếu nhầm mang lại sự đa dạng các cá thể trong quần thể, công việc của quá trình này là thay đổi gen hoặc một số đặc tính của cá thể đó, mục tiêu của quá trình này là duy trì sự đa dạng cho quần thể. Có khá nhiều phương pháp để thực hiện quá trình Mutation như: Bit Flip Mutation, Swap Mutation, Scramble Mutation, Inversion Mutation, Insertion Mutation, Shuffle Mutation, Creep Mutation, Boundary Mutation

Trong demo này sẽ ứng dụng phương pháp Shuffle Mutation, Shuffle Mutation là một dạng đột biến ngẫu nhiên, bước đầu tiên của phương pháp này là chọn một đoạn hoặc một gen đơn lẻ ngẫu nhiên, sau đó sắp xếp các đoạn đó vào cá thể. Với demo tìm chuỗi mục tiêu phương pháp Shuffle Mutation sẽ thực hiện chọn ngẫu nhiên một chữ cái, con số, ký tự đặc biệt trong bộ gen đã có sẵn, sau đó các ký tự này sẽ được sắp xếp thành các chuỗi bằng cách đưa vào quá trình Create\_Chromosome().

def Genetic\_Mutation(self):  
        global Genes  
        Gen = random.choice(Genes)  
        return Gen

Biến Genes sẽ được khởi tạo toàn cục trong suốt quá trình thực hiện giải thuật di truyền, khởi tạo biến Gen biến này được gán giá trị là các ký tự được chọn ngẫu nhiên bằng hàm choice() với tham số sẽ là biến Genes toàn cụ, kết quả được trả về là một ký tự ngẫu nhiên đã được chọn. Các ký tự đã được chọn sẽ được tạo thành chuỗi bằng hàm Create\_Chromosome().

def Create\_Chromosome(self):  
        global Target  
        Chromosome\_Len = len(Target)  
        return [self.Genetic\_Mutation() for \_ in range(Chromosome\_Len)]

1. **MÃ HÓA**
2. **CƠ CHẾ THỰC HIỆN**
3. **NGUYÊN LÝ HOẠT ĐỘNG**