|  |  |
| --- | --- |
|  |  |
| ĐINH TUẤN ANH | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
|  |
| ĐỒ ÁN ĐẠI HỌC NGÀNH CÔNG NGHỆ THÔNG TIN |
|  |
| **PHÁT TRIỂN ỨNG DỤNG CYTOSCAPE TÌM K-CORE/R-CORE CỦA MẠNG LƯỚI QUY MÔ LỚN BẰNG THUẬT TOÁN SONG SONG TRÊN JAVA** |
|  |
|  |
| |  |  | | --- | --- | | **CBHD:** | TS. Trần Tiến Dũng | | **Sinh viên:** | Vũ Hải Long | | **Mã số sinh viên:** | 2019600241 | |
| CÔNG NGHỆ THÔNG TIN |  |
|  |
|  |
|  |
|  |
|  |
| Hà Nội – 2023 |

**LỜI CẢM ƠN**

Đầu tiên, em xin gửi lời cảm ơn chân thành đến Trường đại Công Nghiệp Hà Nội và khoa Công nghệ thông tin đã đưa học phần đồ án tốt nghiệp vào chương trình giảng dạy. Đặc biệt, em xin gửi lời cảm ơn sâu sắc đến giảng viên bộ môn – thầy Trần Tiến Dũng đã dạy dỗ, truyền đạt những kiến thức quý báu cho em trong suốt thời gian học tập vừa qua. Trong thời gian học tập em đã được truyền đạt những kiến thức chuyên ngành hết sức quý báu và bổ ích và đó sẽ là hành trang để em có thể vững bước trên con đường tương lai.

Học phần Thực tập doanh nghiệp là học phần thú vị, vô cùng bổ ích và có tính thực tế cao. Đảm bảo cung cấp đủ kiến thức, gắn liền với nhu cầu thực tiễn của sinh viên. Em đã cố gắng nghiên cứu một cách nghiêm túc và kỹ lưỡng để có thể hoàn thành bài nghiên cứu thật chi tiết và hoàn chỉnh. Tuy nhiên do vốn kiến thức còn hạn nên bài nghiên cứu khó có thể tránh khỏi những thiếu sót cũng như chưa chi tiết hoàn chỉnh vậy nên kính mong thầy giáo xem xét và góp ý để bài tiểu luận của em được hoàn thiện hơn.

Em xin chân thành cảm ơn!

MỤC LỤC

[**LỜI CẢM ƠN i**](#_Toc103344686)

[**MỞ ĐẦU 1**](#_Toc103344689)

[**1. Lý do chọn đề tài 1**](#_Toc103344690)

[**2. Mục tiêu đề tài 1**](#_Toc103344691)

[**3. Đối tượng và phạm vi nghiên cứu 1**](#_Toc103344692)

[**4. Ý nghĩa khoa học, thực tiễn và kết quả mong muốn của đề tài 1**](#_Toc103344693)

[**5. Cấu trúc báo cáo 2**](#_Toc103344694)

[**Chương 1 - TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU 3**](#_Toc103344695)

[**1.1. Giới thiệu chung 3**](#_Toc103344696)

[**1.2. Định hướng phát triển 3**](#_Toc103344697)

[**1.3. Tổng kết chương 1 3**](#_Toc103344698)

[**Chương 2 - CƠ SỞ LÝ THUYẾT 4**](#_Toc103344699)

[**2.1. Tìm hiểu về Cytoscape 4**](#_Toc103344700)

[**2.1.1. Cytoscape là gì? 4**](#_Toc103344701)

[**2.1.2. Cytoscape App là gì? 4**](#_Toc103344702)

[**2.2. Tổng quan về K-core và R-core của mạng lưới 5**](#_Toc103344703)

[**2.2.1. K-core là gì? 5**](#_Toc103344704)

[**2.2.2. R-core là gì? 6**](#_Toc103344705)

[**2.3. Tổng quan về lập trình song song 7**](#_Toc103344706)

[**2.3.1. Lập trình song song là gì? 7**](#_Toc103344707)

[**2.3.2. Ưu điểm và nhược điểm của lập trình song song 8**](#_Toc103344708)

[**2.4. Tìm hiểu về ngôn ngữ lập trình 8**](#_Toc103344709)

[**2.4.1. Ngôn ngữ Java 8**](#_Toc103344710)

[**2.4.2. Maven Apache 10**](#_Toc103344711)

[**2.5. Một số thư viện hỗ trợ 11**](#_Toc103344712)

[**2.5.1. Thư viện Aparapi 11**](#_Toc103344713)

[**2.6. Dữ liệu mạng lưới 11**](#_Toc103344714)

[**2.6.1. Dữ liệu dạng file text 11**](#_Toc103344715)

[**2.6.2. Dữ liệu dạng file XML 12**](#_Toc103344716)

[**2.7. Thuật toán 13**](#_Toc103344717)

[**2.7.1. Thuật toán PKC tính K-core 13**](#_Toc103344718)

[**2.7.2. Thuật toán ParR-core 14**](#_Toc103344719)

[**2.8. Tổng kết chương 2 14**](#_Toc103344720)

[**Chương 3 - NỘI DUNG VÀ KẾT QUẢ ĐỒ ÁN 16**](#_Toc103344721)

[**3.1. Phân tích hệ thống 16**](#_Toc103344722)

[**3.1.1. Biểu đồ use case 16**](#_Toc103344723)

[**3.1.2. Mô tả cho tiết các Use case 16**](#_Toc103344724)

[**3.1.3. Biểu đồ trình tự (Sequence diagram) 21**](#_Toc103344725)

[**3.2. Thiết kế hệ thống 23**](#_Toc103344726)

[**3.2.1. Thiết kế giao diện 24**](#_Toc103344727)

[**3.3. Kết quả thực nghiệm 24**](#_Toc103344728)

[**3.3.1. Giao diện Extension 24**](#_Toc103344729)

[**3.3.2. Kết quả chạy phần mềm 25**](#_Toc103344730)

[**3.4. Tổng kết chương 3 31**](#_Toc103344731)

[**KẾT LUẬN 32**](#_Toc103344732)

[**TÀI LIỆU THAM KHẢO 33**](#_Toc103344733)

# MỞ ĐẦU

1. **Lý do chọn đề tài**

Ngày nay, công nghệ thông tin không ngừng phát triển một cách mạnh mẽ và hiện đại. Các ứng dụng của công nghệ thông tin ngày càng đi sâu vào đời sống con người trở thành một bộ phận không thể thiếu của thế giới văn minh. Cùng với đó là sự phát triển của khoa học mạng lưới vào việc phân tích các mạng xã hội, mạng sinh học, mạng lưới internet v.v..

Vì vậy, để hỗ trợ việc phân tích các mạng lưới quy mô, nâng cao khả năng tính toán chính xác và cải thiện hiệu suất tính toán. Em đã tìm hiểu và chọn đề tài: “Phát triển ứng dụng Cytoscape tìm K-core/R-core của mạng lưới quy mô lớn bằng thuật toán song song trên Java” giúp cho việc quản phân tích các mạng lưới quy mô lớn một cách dễ dàng và hiệu quả hơn.

1. **Mục tiêu đề tài**

* Xây dựng các chức năng cơ bản của một Cytoscape App.
* Phần mềm có khả năng tự tương thích, hiển thị được trên tất cả các thiết bị hiện tại và có thể nâng cấp trong tương lai.
* Phần mềm đáp ứng được yêu cầu tính toán K-core và R-core của mạng lưới quy mô lớn.
* Nắm bắt được công nghệ thiết kế Cytoscape App bằng Java.
* Nắm bắt được cách lập trình song song trên CPU và GPU bằng java.
* Ứng dụng Maven Project vào xây dựng phần mềm.

1. **Đối tượng và phạm vi nghiên cứu**

Phần mềm này được thực hiện để phân tích các mạng lưới quy mô lớn. Sử dụng thuật toán song song để tăng hiệu suất tính toán.

1. **Ý nghĩa khoa học, thực tiễn và kết quả mong muốn của đề tài**

* Ý nghĩa khoa học thực tiễn:
* Phần mềm được xây dựng đã áp dụng thành tựu của công nghệ thông tin vào lĩnh vực phân tích mạng lưới.
* Sự ra đời của phần mềm này không chỉ giải quyết những khó khăn của con người trong việc phân tích mạng một cách thủ công, mà còn cải thiện hiệu suất tính toán các mạng quy mô lớn.
* Kết quả mong muốn của đề tài:
* Xây dựng thành công Cytoscape App tính toán K-core và R-core.
* Nắm bắt được các kiến thức về Java, Maven Project, lập trình song song, v.v.
* Ứng dụng, phát triển được công nghệ thiết kế Cytoscape App bằng Java.

1. **Cấu trúc báo cáo**

Báo cáo gồm 3 phần:

* Phần mở đầu
* Phần Nội dung
* Phần Kết luận và kiến nghị.

Phần nội dung sẽ gồm 3 chương:

**Chương 1: TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU**

Đưa ra cái nhìn tổng quan về nội dung nghiên cứu.

**Chương 2: CƠ SỞ LÝ THUYẾT**

Để làm được một Cytoscape App thì chúng ta phải hiểu được bản chất ngôn ngữ mình dùng để lập trình. Ở chương này báo cáo sẽ tập trung giới thiệu các ngôn ngữ lập trình, framework và các thư viện.

**Chương 3: NỘI DUNG VÀ KẾT QUẢ ĐỒ ÁN**

Tại đây chúng ta sẽ khảo sát, phân tích thiết hế hệ thống và đưa ra các kết quả đạt được.

# Chương 1 - TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU

* 1. **Giới thiệu chung**

Với sự phát triển nhanh chóng của khoa học và công nghệ, thế giới ngày càng trở nên kết nối. Nhu cầu cấp thiết sau đây về việc hiểu mối quan hệ giữa các cá nhân và các đặc điểm cấu trúc toàn cầu sẽ thúc đẩy việc nghiên cứu khoa học mạng và nhiều môn học liên ngành trong những năm gần đây. Do đó, điều quan trọng là phải có các phương pháp và thuật toán giúp chúng ta khám phá các đặc tính cấu trúc của mạng. Trong vài thập kỷ qua, nhiều thuật toán thiết yếu đã được phát triển bởi các nhà khoa học thuộc nhiều lĩnh vực khác nhau. Đề tài này sẽ tập trung vào một trong những phương pháp được sử dụng rộng rãi nhất được gọi là phân rã core. Phân tách core là tìm đồ thị con lớn nhất của mạng, trong đó mỗi nút có ít nhất các nút lân cận trong đồ thị con.

Nghiên cứu cấu trúc liên kết của một mạng là rất quan trọng để suy ra các động lực cơ bản như khả năng chịu đựng thất bại, hành vi nhóm và các mô hình lan truyền. phân rã K-core, R-core là một số liệu được thiết lập tốt để phân chia một mạng lưới thành các lớp từ các đỉnh bên ngoài đến các đỉnh trung tâm hơn.

* 1. **Định hướng phát triển**

Với quy mô phần mềm vừa và nhỏ, cộng thêm thời gian không cho phép nên đề tài “Phát triển ứng dụng Cytoscape tìm K-core/R-core của mạng lưới quy mô lớn bằng thuật toán song song trên Java” có thể chưa có đầy đủ và còn thiếu sót. Trong tương lai em sẽ cố gắng khắc phục những thiếu sót này bằng một số định hướng như:

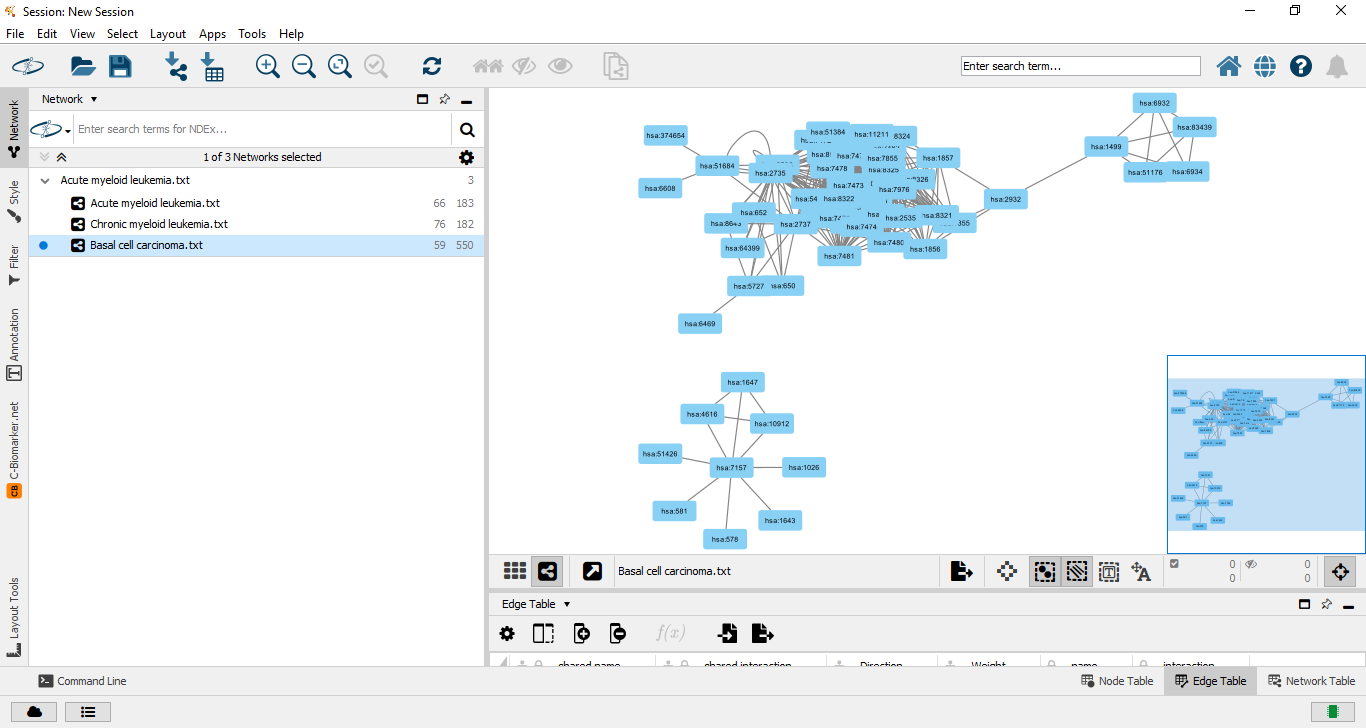
* Chỉnh sửa giao diện, tối ưu hoá tốc độ thực hiện tính toán.
* Thêm một số chức năng khác vào phần mềm.
* Cải tiến đưa phần mềm vào sử dụng.
  1. **Tổng kết chương 1**

Trong chương này em đã giới thiệu về khao học mạng lưới một cách bao quát nhất để mọi người có thể thấy được tổng quan về khoa học mạng lưới những lợi ích mà nó đem lại cũng như là những vấn đề còn tồn tại. Từ đó để rút ra dược những hướng đi đúng đắn nhất để hoàn thiện về nội dung nghiên cứu của đề tài này.

# Chương 2 - CƠ SỞ LÝ THUYẾT

* 1. **Tìm hiểu về Cytoscape**
     1. **Cytoscape là gì?**

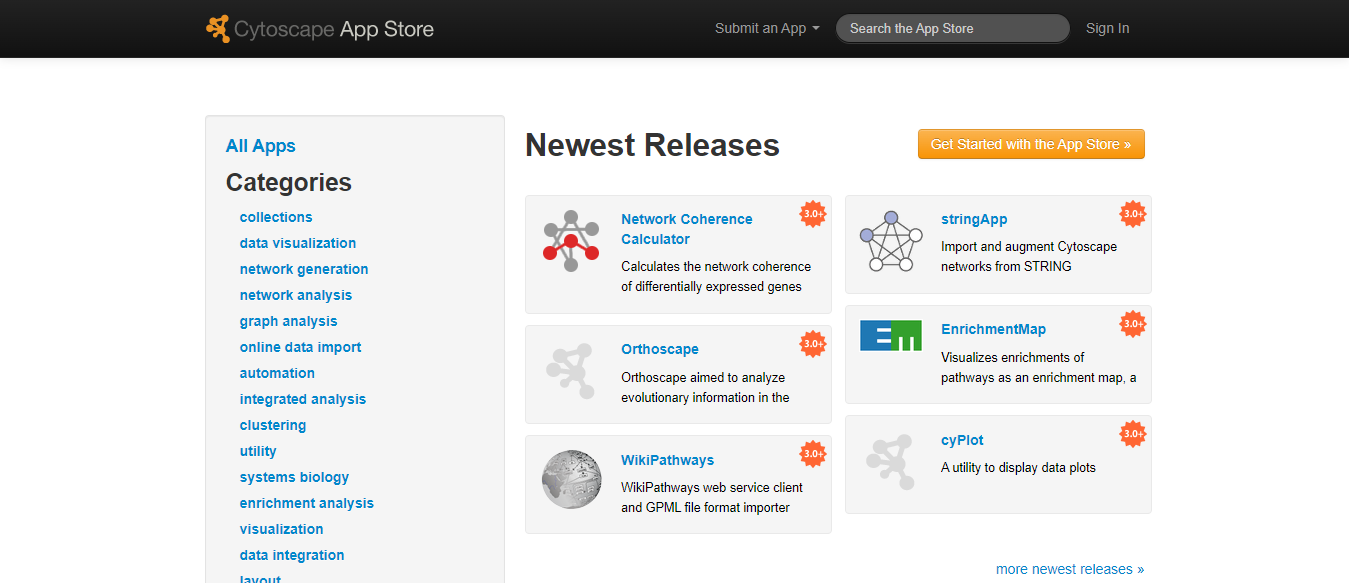
Cytoscape là phần mềm hiển thị và phân tích dữ liệu các đồ thị(mạng lưới). Cytoscape là một nền tảng phần mềm mã nguồn mở để trực quan hóa các mạng tương tác phân tử và các con đường sinh học, đồng thời tích hợp các mạng này với các chú thích, hồ sơ biểu hiện gen và dữ liệu trạng thái khác. Mặc dù Cytoscape ban đầu được thiết kế để nghiên cứu sinh học, nhưng giờ đây nó là một nền tảng chung để phân tích và hình dung mạng phức tạp. Phân phối lõi Cytoscape cung cấp một tập hợp các tính năng cơ bản để tích hợp, phân tích và trực quan hóa dữ liệu.



Hình 2. 1. Cytoscape

* + 1. **Cytoscape App là gì?**

Cytoscape App là các module được phát triển độc lập và có thể tích hợp vào Cytoscape. Các tính năng bổ sung có sẵn dưới dạng ứng dụng. Các ứng dụng có sẵn để phân tích mạng và cấu hình phân tử, bố cục mới, hỗ trợ định dạng tệp bổ sung, tập lệnh và kết nối với cơ sở dữ liệu. Chúng có thể được phát triển bởi bất kỳ ai sử dụng Cytoscape open API dựa trên công nghệ Java™.



Hình 2. 2. Cytoscape App

* 1. **Tổng quan về K-core và R-core của mạng lưới**
     1. **K-core là gì?**

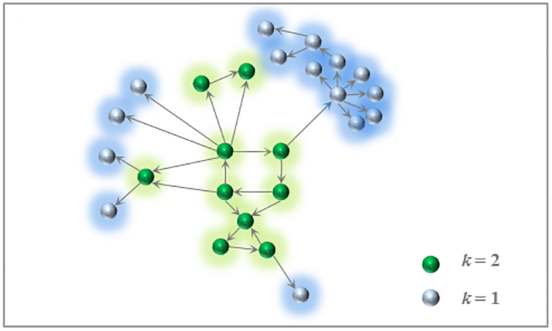
K-core của một mạng lưới là đồ thị con tối đa trong đó tất cả các đỉnh có mức độ ít nhất là k. Vấn đề phân rã K-core đề cập đến việc tìm ra độ lõi của mọi đỉnh trong mạng lưới. Một đỉnh có lõi l nếu nó thuộc về lõi l nhưng không thuộc về lõi (l + 1). Các thuật toán thời gian tuyến tính được biết đến với việc thực hiện phân rã K-core. Điều này làm cho sự phân rã K-core rất hữu ích như một công cụ phân tích mạng chung.

Một trong những ứng dụng ban đầu của sự phân rã K-core là trực quan hóa các mạng lớn. Trong công việc này, các đỉnh được gán tọa độ trên các chu kỳ đồng tâm dựa trên các giá trị K-core của chúng. Các đỉnh với giá trị lõi cao hơn được đặt gần trung tâm hơn. Ngoài ra, nếu các đỉnh từ các thành phần được kết nối khác nhau nằm trong cùng một K-core,chúng ở cùng một chu kỳ đồng tâm, nhưng được đặt trong các cụm khác nhau.

sự phân rã K-core cũng đã được áp dụng để phân tích các mạng trong sinh học. Các cụm được tìm thấy trong dữ liệu mạng gen của con người bằng cách sử dụng sự phân rã K-core. Tuy nhiên, các K-core có thể không đại diện cho các cụm tốt

Phân rã K-core đã được sử dụng để phân tích mạng lưới internet quy mô lớn. Người ta quan sát thấy rằng các đồ thị con K-core được kết nối trong các mạng này. Các đồ thị con  K-core thể hiện một cấu trúc phân cấp, tức là K-core là một phần nhỏ của (K-1)-core, và kích thước của các đồ thị con K-core về các đỉnh được hiển thị theo luật quyền lực. phân hủy K-core cũng được sử dụng như một chiến lược tiền xử lý để giải quyết các vấn đề NP-hard, ví dụ, tìm nhóm tối đa trong một biểu đồ.

K-core của một mạng lưới được tính bằng quy tắc cắt tỉa sau đây. Cho một mạng lưới, tất cả các nút có bậc nhỏ hơn k đều bị loại bỏ, cùng với các tương tác sự cố của chúng khỏi mạng. Quá trình loại bỏ này được lặp lại cho đến khi mức độ của mọi nút trong mạng còn lại là k. K-core biểu thị tập hợp các nút còn lại, và do đó nó cho rằng K1-core là một tập con của K2-core nếu k1 ≥ k2.

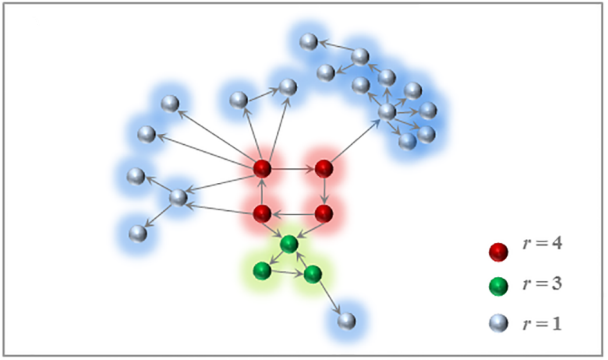


Hình 2. 3. Ví dụ về phân rã K-core

* + 1. **R-core là gì?**

R-core cũng tương tự như K-core, sự phân rã R-core dựa trên giá trị của khả năng tiếp cận R.

Cách tính R-core của một mạng lưới sử dụng quy tắc cắt tỉa tương tự như phương pháp phân rã K-core ngoại trừ R được sử dụng thay cho mức độ nút. Nói cách khác, tất cả các nút có R < r và các tương tác sự cố của chúng được loại bỏ ở mỗi bước cắt tỉa. Kết quả là, phân rã R-core một mạng lưới thành các mạng con gọi là r-core.



Hình 2. 4. ví dụ về phân rã R-core

* 1. **Tổng quan về lập trình song song**
     1. **Lập trình song song là gì?**

Hiện nay, để giải quyết các bài toán lớn người ta thường nghĩ đến việc sử dụng các siêu máy tính hoặc việc kết hợp nhiều máy tính với nhau để tính toán. Tuy nhiên, với phương pháp lập trình cổ điển thì không thể nào phát triển được chương trình có thể tận dụng được sức mạnh của các hệ thống đó, nhất là ngày nay công nghệ phần cứng cho phép nhiều hơn một bộ vi xử lý(CPU) vào trong một chíp, mà trong khi một chương trình ứng dụng đa phần chỉ chạy trên một lõi(core) duy nhất. Điều đó dẫn đến việc lãng phí nguồn tài nguyên có sẵn trong một máy tính. Đó chính là lý do lập trình song song ra đời.

Lập trình song song(đa luồng) là cách lập trình để một tiến trình thực hiện nhiều luồng đồng thời. Một ứng dụng ngoài luồng chính có thể có các luồng khác thực thi đồng thời làm ứng dụng chạy nhanh và hiệu quả hơn.

Lập trình song song là một công việc phức tạp hơn so với lập trình tuần tự thông thường, người phát triển phải thực hiện một quá trình “song song hóa”, biến đổi các chương trình tuần tự thành chương trình song song có khả năng tận dụng tối đa sức mạnh của hệ thống.

Lập trình song song bao gồm các bước :

* Phân hủy một thuật toán hay các dữ liệu đầu vào.
* Phân chia nhiệm vụ cho các bộ phận để làm việc trong bộ vi xử lý cùng một lúc.
* Phối hợp và trao đổi giữa các bộ vi xử lý.
  + 1. **Ưu điểm và nhược điểm của lập trình song song**

Ưu điểm của lập trình song song:

* Nó không chặn người sử dụng vì các luồng là độc lập và bạn có thể thực hiện nhiều công việc cùng một lúc.
* Mỗi luồng có thể dùng chung và chia sẻ nguồn tài nguyên trong quá trình chạy, nhưng có thể thực hiện một cách độc lập.
* Luồng là độc lập vì vậy nó không ảnh hưởng đến luồng khác nếu ngoại lệ xảy ra trong một luồng duy nhất.
* Có thể thực hiện nhiều hoạt động với nhau để tiết kiệm thời gian. Ví dụ một ứng dụng có thể được tách thành : luồng chính chạy giao diện người dùng và các luồng phụ nhiệm gửi kết quả xử lý đến luồng chính.

Nhược điểm của lập trình song song:

* Càng nhiều luồng thì xử lý càng phức tạp.
* Xử lý vấn đề về tranh chấp bộ nhớ, đồng bộ dữ liệu khá phức tạp.
* Cần phát hiện tránh các luồng chết (dead lock), luồng chạy mà không làm gì trong ứng dụng cả.
  1. **Tìm hiểu về ngôn ngữ lập trình**
     1. **Ngôn ngữ Java**

Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động.

Java được tạo ra với tiêu chí “Viết một lần, thực thi khắp nơi” (Write Once, Run Anywhere  – WORA). Chương trình phần mềm viết bằng Java có thể chạy trên mọi nền tảng khác nhau thông qua một môi trường thực thi với điều kiện có môi trường thực thi thích hợp hỗ trợ nền tảng đó.

Đặc điểm của ngôn ngữ lập trình Java:

* Hướng đối tượng hoàn toàn.
* Độc lập phần cứng và hệ điều hành.
* Ngôn ngữ thông dịch.
* Cơ chế thu gom rác tự động.
* Đa luồng.
* Tính an toàn và bảo mật.

Ứng dụng của ngôn ngữ Java:

* Các ứng dụng cho hệ điều hành Android: Nếu bạn có một chiếc điện thoại Android thì đồng nghĩa với việc mọi ứng dụng mà bạn đang sử dụng đều được hình thành và phát triển trên nền tảng Java bởi vì hệ điều hành Android được lập trình Java hỗ trợ tối đa. Vậy nên đây quả là một cơ hội lớn cho các lập trình viên Java khi sự phát triển của Android ngày càng lớn mạnh. Android tuy sử dụng máy ảo JVM và các package khác, nhưng phần code thì vẫn được viết bằng Java.
* Các ứng dụng bảo mật cao: Java là ngôn ngữ lập trình có tính bảo mật cao nên nó thường được sử dụng vào trong các dịch vụ tài chính, ngân hàng. Trên thế giới hiện nay có rất nhiều ngân hàng sử dụng Java để viết các hệ thống giao dịch điện tử, các hệ thống xác nhận và kiểm toán, các dự án xử lý dữ liệu và một số công việc quan trọng khác như Goldman Sachs, Citigroup, Barclays, Standard Charted và các ngân hàng khác.
* Các ứng dụng web Java: Lập trình Java không chỉ được sử dụng trên thiết bị di động mà còn được biết đến là nền tảng của của các ứng dụng website. Đây cũng là ngôn ngữ được sử dụng nhiều trong các ứng dụng web và thương mại điện tử. Nhiều tổ chức chính phủ, y tế, bảo hiểm, giáo dục, quốc phòng và một số bộ phận khác có ứng dụng web được xây dựng bằng Java.
* Các công cụ phần mềm: Hiện nay có rất nhiều các công cụ phát triển và phần mềm hữu ích được viết bằng Java, ví dụ: Eclipse, InetelliJ Idea và Netbans IDE.
* Không gian nhúng: Ban đầu Java được phát triển để phục vụ cho các thiết bị nhúng nhưng sau đó ngày càng được phát triển mở rộng ra các lĩnh vực khác. Bởi vậy với châm ngôn "viết một lần, chạy mọi nơi" cho thấy rằng Java đang mang lại nhiều lợi ích to lớn.
* Các công nghệ Big Data: Hadoop và các công nghệ big data khác cũng đang sử dụng Java theo cách này hay cách khác, ví dụ: HBase, Accumulo (mã nguồn mở) và ElasticSearch.
* Trong lĩnh vực giao dịch chứng khoán: Java có một lợi thế nữa đó là sử dụng phù hợp với các hệ thống có hiệu suất cao, bởi vì mặc dù hiệu suất có thua chút ít so với ngôn ngữ native, nhưng bạn lại có được sự an toàn, linh động và bảo trì với tốc độ nhanh hơn.
* Các ứng dụng khoa học: Hiện nay Java thường là một lựa chọn mặc định cho các ứng dụng khoa học, bao gồm cả xử lý ngôn ngữ tự nhiên.
  + 1. **Maven Apache**

Maven là công cụ quản lý và thiết lập tự động 1 dự án phần mềm. Chủ yếu dùng cho các lập trình viên java, nhưng nó cũng có thể được dùng để xây dựng và quản lý các dự án dùng C#, Ruby, Scala hay ngôn ngữ khác.

Maven phục vụ mục đích tương tự như Apache Ant, nhưng nó dựa trên khái niệm khác và cách hoạt động khác.

Maven hỗ trợ việc tự động hóa các quá trình tạo dự án ban đầu, thực hiện biên dịch, kiểm thử, đóng gói và triển khai sản phẩm.

Được phát triển bằng ngôn ngữ Java cho phép Maven chạy trên nhiều nền tảng khác nhau: Windows, Linux và Mac OS…

Maven dùng khái niệm Project Object Model (POM) để mô tả việc build project, các thành phần phụ thuộc và các module. Nó định nghĩa trước các target cho việc khai báo task, trình biên dịch, đóng gói và thứ tự hoạt động để mọi việc diến ra tốt nhất.

Trong mỗi project Maven tạo ra một file .pom, trong file này định nghĩa ra những task như task khi chạy test, task khi build và khi chạy Maven sẽ dựa vào những định nghĩa này để thao tác với project.

* 1. **Một số thư viện hỗ trợ**
     1. **Thư viện Aparapi**

Aparapi là một framework để thực thi mã Java gốc trên GPU, được hình thành và phát triển bởi tập đoàn AMD.

Aparapi cho phép các nhà phát triển Java tận dụng sức mạnh tính toán của các thiết bị GPU và APU bằng cách thực thi các đoạn mã song song dữ liệu trên GPU thay vì bị giới hạn trong CPU cục bộ. Nó thực hiện điều này bằng cách chuyển đổi Java bytecode thành OpenCL trong thời gian chạy và thực thi trên GPU. Bởi vì nó được hỗ trợ bởi OpenCL Aparapi tương thích với tất cả các Card đồ họa tương thích OpenCL. nếu vì bất kỳ lý do gì Aparapi không thể thực thi trên GPU, nó sẽ thực thi trong một nhóm luồng Java.

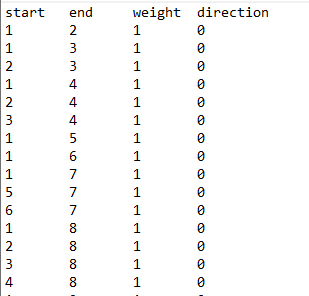
GPU có một kiến ​​trúc độc đáo khiến chúng hoạt động khác với CPU. Một trong những điểm khác biệt đáng chú ý nhất là trong khi một CPU điển hình có ít hơn một chục lõi thì GPU cao cấp có thể có hàng trăm lõi. Điều này làm cho chúng phù hợp duy nhất cho tính toán song song dữ liệu có thể dẫn đến tốc độ tăng gấp hàng trăm lần so với những gì có thể với CPU trung bình của bạn. Điều này có nghĩa là sự khác biệt giữa việc cần cả một trung tâm dữ liệu để chứa ứng dụng của bạn so với chỉ một hoặc hai máy tính, có khả năng tiết kiệm hàng triệu chi phí máy chủ.

* 1. **Dữ liệu mạng lưới**
     1. **Dữ liệu dạng file text**

Dữ liệu dạng file text là dữ liệu mạng lưới được lưu trữ dưới dạng danh sách kề. Bao gồm: đỉnh bắt đầu, đỉnh kết thúc, trọng số và kiểu liên kết

Trong đó:

* Start: là tên của node bắt đầu một liên kết.
* End: là tên của node kết thúc một liên kết.
* Weight: là trọng số của một liên kết.
* Direction: là biểu thị kiểu liên kết(1-liên kết có hướng, 0-liên kết vô hướng).



Hình 2. 5. ví dụ về file dữ liệu dạng text

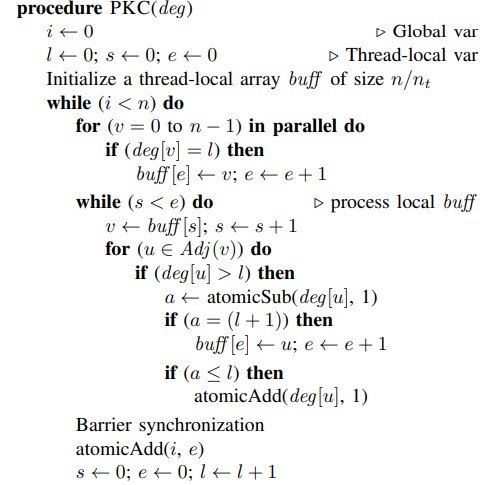
* + 1. **Dữ liệu dạng file XML**

Dữ liệu dạng file XML là dữ liệu được lấy từ cơ sở dữ liệu Kegg pathway.



Hình 2. 6. Dữ liệu file XML

* 1. **Thuật toán**
     1. **Thuật toán PKC tính K-core**



Hình 2. 7. Thuật toán PKC

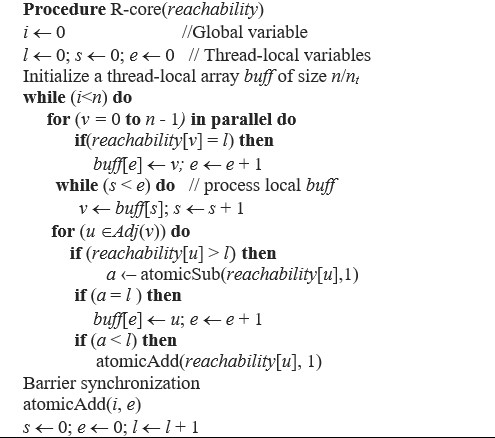
Thuật toán sử một mảng *buff* có kích thước *n / n t*cho mỗi luồng, trong đó *n* là số đỉnh của mạng lưới *n t* là số luồng được sử dụng. ngoài ra mảng *deg*để giữ giá trị bậc của các đỉnh.

Đầu tiên mảng *deg*được quét song song bởi các luồng và luồng xử lý một đỉnh sẽ thêm nó vào mảng *buff* của nó nếu mức độ hiện tại của đỉnh bằng mức độ đang xét.

Giai đoạn tiếp theo trong thuật toán là xử lý các đỉnh trong mảng buffcủa mỗi luồng. Mỗi luồng xử lý các đỉnh trong mảng *buff*cục bộ của nó. Một đỉnh *v*của *buff*được xử lý bằng cách xử lý các đỉnh liền kề của nó. Tuy nhiên, hai đỉnh xử lý luồng trong bộ đệm cục bộ của chúng có thể cần xử lý một đỉnh liền kề cùng một lúc. Do đó, nếu giá trị mức độ hiện tại lớn hơn so với mức độ đang xét, nó được giảm về mặt nguyên tử. Ngoài ra, mức độ của một liền kề có thể bị giảm xuống dưới mức độ đang xét nếu nhiều luồng đang xử lý cùng một đỉnh liền kề. Điều này được khắc phục bằng cách tăng mức độ của đỉnh liền kề nếu bất kỳ mức độ của đỉnh liền kề nào thấp hơn mức độ đang xét. Các luồng thêm đỉnh liền kề vào mảng *buff* cục bộ của nó nếu bất kỳ đỉnh liền kề nào có mức độ bằng mức độ đang xét. Quá trình xử lý kết thúc khi mảng *buff*của mọi luồng trở nên trống.

Mã giả chỉ ra các vòng lặp được thực hiện song song. Thuật toán cũng cần một cuộc gọi đồng bộ hóa sau mỗi cấp độ. Thuật toán kết thúc khi tất cả các đỉnh trong đồ thị được xử lý.

* + 1. **Thuật toán ParR-core**



Hình 2. 8. Thuật toán ParR-core

Thuật toán ParR-core tính R-core cũng giống như Thuật toán PKC tính K-core, chỉ khác ở thuật toán ParR-core sẽ sử dụng một mảng *reachability* để lưu khả năng truy cập của các đỉnh thay cho mảng *deg* lưu mức độ của các đỉnh.

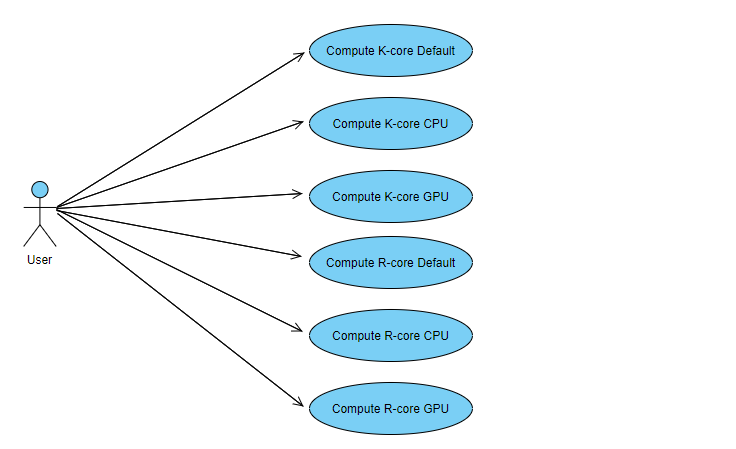
* 1. **Tổng kết chương 2**

Trong chương này em đã làm rõ hơn về Cytoscape và Cytoscape App, khái niệm về K-core và R-core của một mạng lưới, cũng như cách tính K-core và R-core. Tổng quan, ưu điểm và nhược điểm của lập trình song song. Tiếp đến là ngôn ngữ lập trình sử dụng trong phần mềm, những thư viện cần thiết và các dạng dữ liệu mạng lưới để làm đầu vào cho phần mềm.

# Chương 3 - NỘI DUNG VÀ KẾT QUẢ ĐỒ ÁN

## Phân tích hệ thống

### Biểu đồ use case



Hình 3. 1. Biểu đồ Use case

### Mô tả cho tiết các Use case

* + - 1. **Mô tả Use Case ComputeK-coreAuto**
* **Tên use case:** Compute K-core Auto.
* **Mô tả vắn tắt:** Use case này cho phép người sử dụng có thể tính được K-core của mạng lưới tự động.
* **Luồng sự kiện**
  + **Luồng cơ bản**

Use case bắt đầu khi người sử dụng tích vào chọn Choose function là K-core và tích chọn Choose device là Auto rồi bấm nút Run. Phần mềm sẽ hiển thị cửa sổ để người dùng chọn địa chỉ lưu file và tên file. Phần mềm sẽ lấy dữ liệu mạng lưới được import vào Cytopscape và tính toán K-core của mạng lưới, cuối cùng xuất kết quả ra file. Use case kết thúc.

* + **Luồng rẽ nhánh**

Không có.

* **Các yêu cầu đặc biệt**

Không có.

* **Tiền điều kiện**

Không có.

* **Hậu điều kiện**

Không có.

* **Điểm mở rộng**

Không có.

* + - 1. **Mô tả Use Case ComputeK-coreCPU**
* **Tên use case:** Compute K-core CPU.
* **Mô tả vắn tắt:** Use case này cho phép người sử dụng có thể tính được K-core của mạng lưới song song trên CPU.
* **Luồng sự kiện**
  + **Luồng cơ bản**

Use case bắt đầu khi người sử dụng tích vào chọn Choose function là K-core và tích chọn Choose device là CPU rồi bấm nút Run. Phần mềm sẽ hiển thị cửa sổ để người dùng chọn địa chỉ lưu file và tên file. Phần mềm sẽ lấy dữ liệu mạng lưới được import vào Cytopscape và tính toán K-core của mạng lưới, cuối cùng xuất kết quả ra file. Use case kết thúc.

* + **Luồng rẽ nhánh**

Không có.

* **Các yêu cầu đặc biệt**

Không có.

* **Tiền điều kiện**

Không có.

* **Hậu điều kiện**

Không có.

* **Điểm mở rộng**

Không có.

* + - 1. **Mô tả Use Case ComputeK-coreGPU**
* **Tên use case:** Compute K-core GPU.
* **Mô tả vắn tắt:** Use case này cho phép người sử dụng có thể tính được K-core của mạng lưới song song trên GPU.
* **Luồng sự kiện**
  + **Luồng cơ bản**

Use case bắt đầu khi người sử dụng tích vào chọn Choose function là K-core và tích chọn Choose device là GPU rồi bấm nút Run. Phần mềm sẽ hiển thị cửa sổ để người dùng chọn địa chỉ lưu file và tên file. Phần mềm sẽ lấy dữ liệu mạng lưới được import vào Cytopscape và tính toán K-core của mạng lưới, cuối cùng xuất kết quả ra file. Use case kết thúc.

* + **Luồng rẽ nhánh**

Không có.

* **Các yêu cầu đặc biệt**

Không có.

* **Tiền điều kiện**

Không có.

* **Hậu điều kiện**

Không có.

* **Điểm mở rộng**

Không có..

* + - 1. **Mô tả Use Case ComputeR-coreAuto**
* **Tên use case:** Compute R-core Auto.
* **Mô tả vắn tắt:** Use case này cho phép người sử dụng có thể tính được R-core của mạng lưới tự động.
* **Luồng sự kiện**
  + **Luồng cơ bản**

Use case bắt đầu khi người sử dụng tích vào chọn Choose function là R-core và tích chọn Choose device là Default rồi bấm nút Run. Phần mềm sẽ hiển thị cửa sổ để người dùng chọn địa chỉ lưu file và tên file. Phần mềm sẽ lấy dữ liệu mạng lưới được import vào Cytopscape và tính toán R-core của mạng lưới, cuối cùng xuất kết quả ra file. Use case kết thúc.

* + **Luồng rẽ nhánh**

Không có.

* **Các yêu cầu đặc biệt**

Không có.

* **Tiền điều kiện**

Không có.

* **Hậu điều kiện**

Không có.

* **Điểm mở rộng**

Không có.

* + - 1. **Mô tả Use Case ComputeR-coreCPU**
* **Tên use case:** Compute R-core CPU.
* **Mô tả vắn tắt:** Use case này cho phép người sử dụng có thể tính được K-core của mạng lưới song song trên CPU.
* **Luồng sự kiện**
  + **Luồng cơ bản**

Use case bắt đầu khi người sử dụng tích vào chọn Choose function là R-core và tích chọn Choose device là CPU rồi bấm nút Run. Phần mềm sẽ hiển thị cửa sổ để người dùng chọn địa chỉ lưu file và tên file. Phần mềm sẽ lấy dữ liệu mạng lưới được import vào Cytopscape và tính toán R-core của mạng lưới, cuối cùng xuất kết quả ra file. Use case kết thúc.

* + **Luồng rẽ nhánh**

Không có.

* **Các yêu cầu đặc biệt**

Không có.

* **Tiền điều kiện**

Không có.

* **Hậu điều kiện**

Không có.

* **Điểm mở rộng**

Không có.

* + - 1. **Mô tả Use Case ComputeR-coreGPU**
* **Tên use case:** Compute R-core GPU.
* **Mô tả vắn tắt:** Use case này cho phép người sử dụng có thể tính được R-core của mạng lưới song song trên GPU.
* **Luồng sự kiện**
  + **Luồng cơ bản**

Use case bắt đầu khi người sử dụng tích vào chọn Choose function là R-core và tích chọn Choose device là GPU rồi bấm nút Run. Phần mềm sẽ hiển thị cửa sổ để người dùng chọn địa chỉ lưu file và tên file. Phần mềm sẽ lấy dữ liệu mạng lưới được import vào Cytopscape và tính toán K-core của mạng lưới, cuối cùng xuất kết quả ra file. Use case kết thúc.

* + **Luồng rẽ nhánh**

Không có.

* **Các yêu cầu đặc biệt**

Không có.

* **Tiền điều kiện**

Không có.

* **Hậu điều kiện**

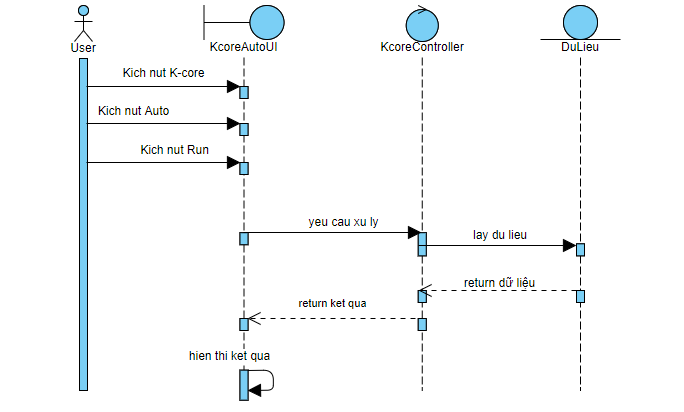
Không có.

* **Điểm mở rộng**

Không có.

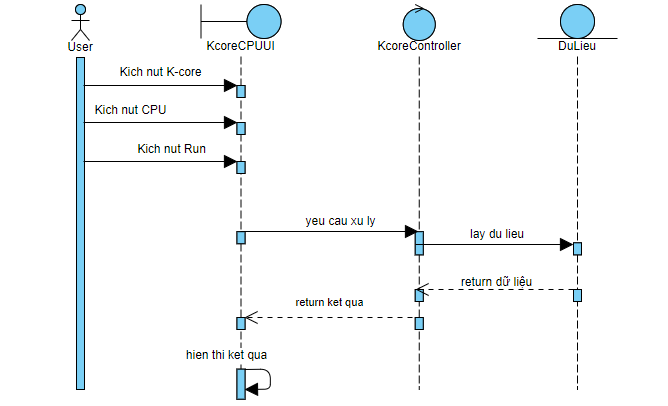
### Biểu đồ trình tự (Sequence diagram)

* + - 1. **Biểu đồ trình tự ComputeK-coreAuto**



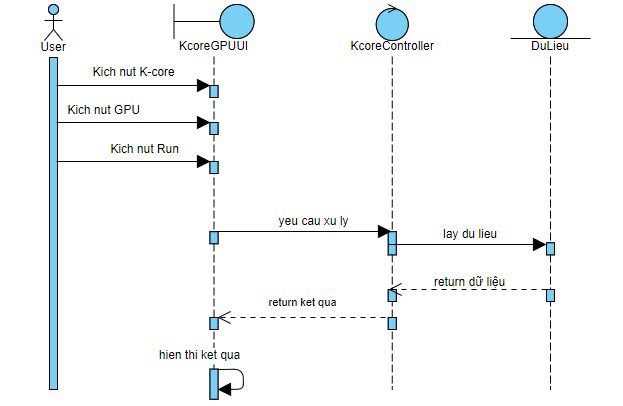
Hình 3. 2. Biểu đồ trình tự ComputeK-coreAuto

* + - 1. **Biểu đồ trình tự ComputeK-coreCPU**



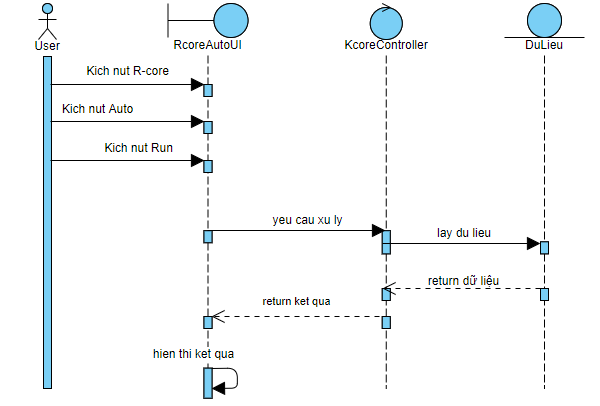
Hình 3. 3. Biểu đồ trình tự ComputeK-coreCPU

* + - 1. **Biểu đồ trình tự ComputeK-coreGPU**



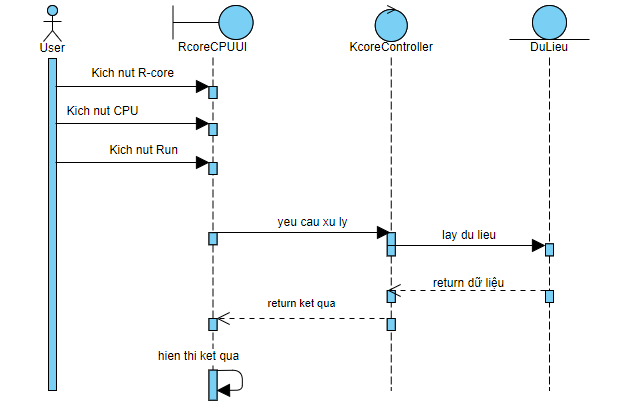
Hình 3. 4. Biểu đồ trình tự ComputeK-coreGPU

* + - 1. **Biểu đồ trình tự ComputeR-coreAuto**



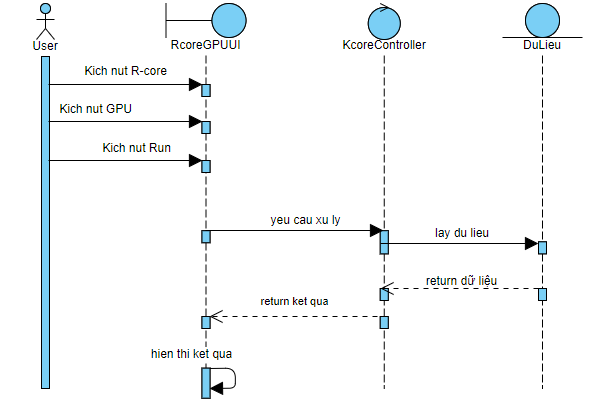
Hình 3. 5. Biểu đồ trình tự ComputeR-coreAuto

* + - 1. **Biểu đồ trình tự ComputeR-coreCPU**



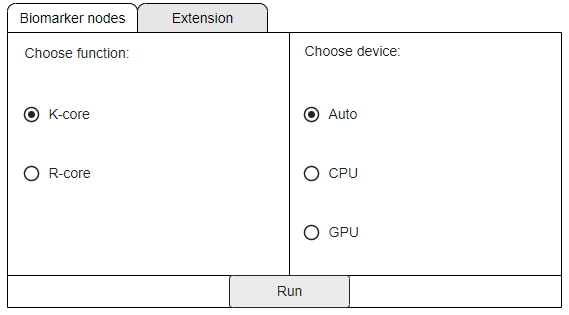
Hình 3. 6. Biểu đồ trình tự ComputeR-coreCPU

* + - 1. **Biểu đồ trình tự ComputeR-coreGPU**



Hình 3. 7. Biểu đồ trình tự ComputeR-coreGPU

* 1. **Thiết kế hệ thống**
     1. **Thiết kế giao diện**
        1. **Giao diện Extension**

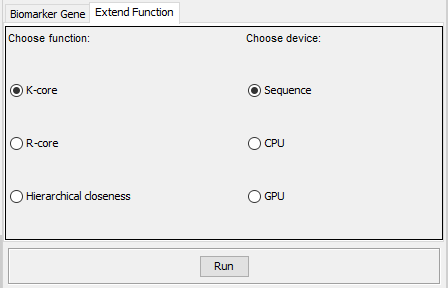


Hình 3. 8. Thiết kế giao diện extension

## Kết quả thực nghiệm

### Giao diện Extension

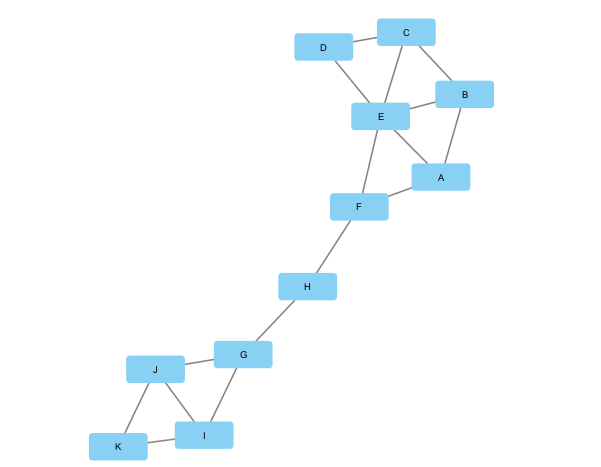
Khi người dùng mở phần mềm và kích chọn tab Extension phần mềm sẽ hiển thị giao diện extension với bố cục 3 phần là đầu, nội dung và chân phần mềm. Ở phần đầu bao gồm các tab như Biomarker nodes và Extension.



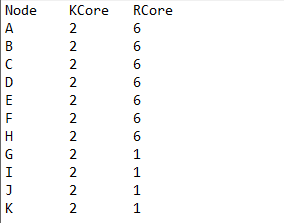
Hình 3. 9. Giao diện đầu trang website

Tiếp theo là phần nội dung của trang web bao gồm hai phần: phần đầu tiên hiển thị chọn các chức năng, phần thứ hai hiển thị giao diện chọn các thiết bị. Cuối cùng là phần chân hiển thị nút Run.

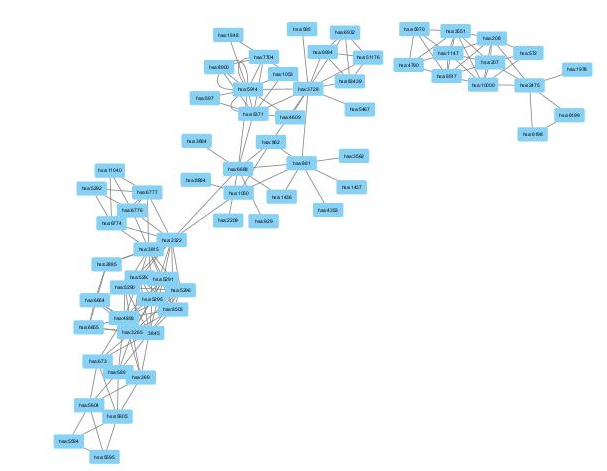
* + 1. **Kết quả chạy phần mềm**



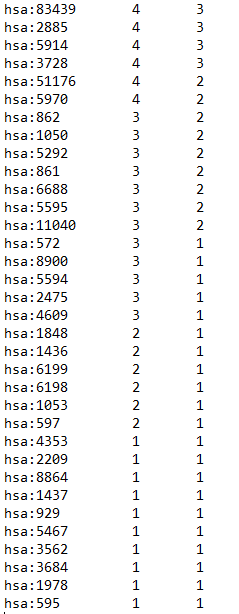
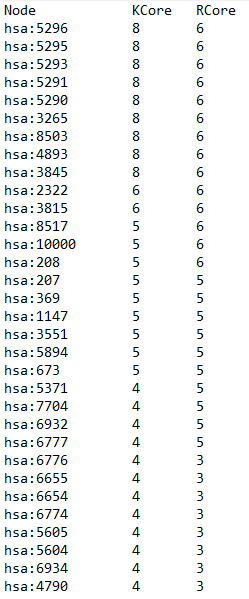
Hình 3. 10. Dữ liệu mạng lưới Fig1HC



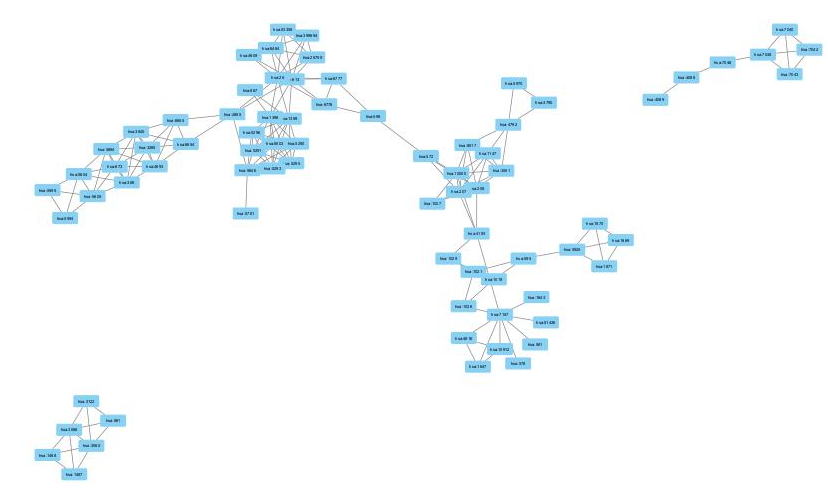
Hình 3. 11. Kết quả K-core và R-core của mạng Fig1HC



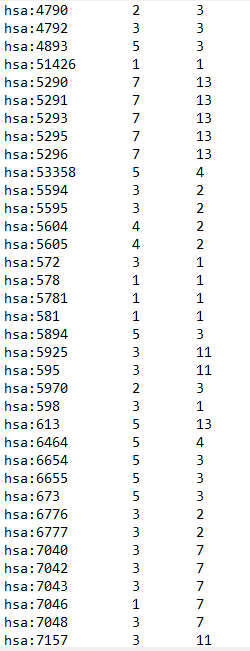
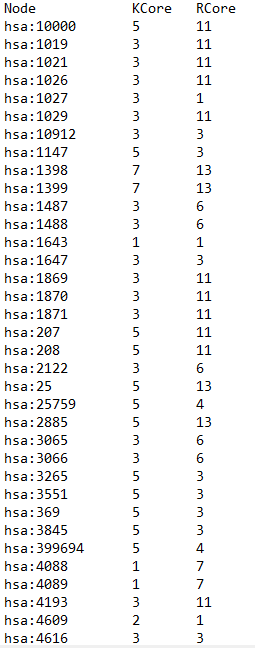
Hình 3. 12. Dữ liệu mạng lưới Acute myeloid leukemia



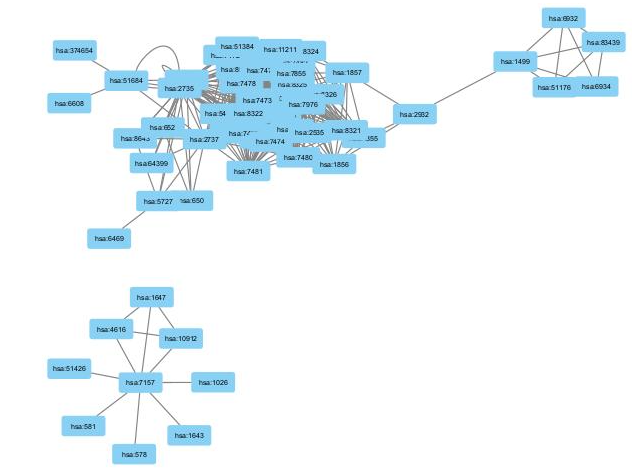
Hình 3. 13. Kết quả K-core và R-core của mạng Acute myeloid leukemia



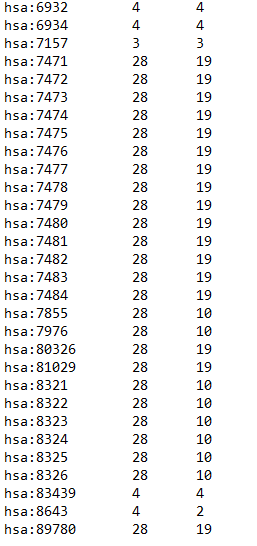
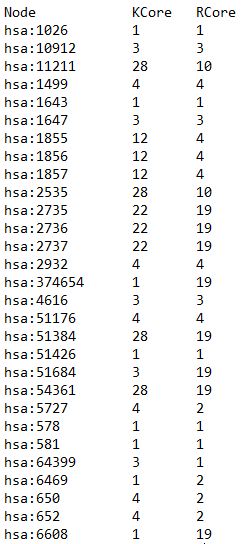
Hình 3. 14. Dữ liệu mạng lưới Chronic myeloid leukemia



Hình 3. 15. Kết quả K-core và R-core của mạng Chronic myeloid leukemia



Hình 3. 16. Dữ liệu mạng lưới Basal cell carcinoma



Hình 3. 17. Kết quả K-core và R-core của mạng Basal cell carcinoma

* 1. **Tổng kết chương 3**

Kết thúc chương này em đã vận dụng những kiến thức tích luỹ được trong khoảng thời gian học tập tại trường để vận dụng, thực hành, khảo sát và đưa ra được những bản phân tích thiết kế hệ thống từ đó xây dựng lên được các chức năng của một Cytoscape app.

# KẾT LUẬN

**Kết quả đạt được của đề tài**

Sau một thời gian tập trung triển khai đề tài, em đã hoàn thành được **Phát triển ứng dụng Cytoscape tìm K-core/R-core của mạng lưới quy mô lớn bằng thuật toán song song trên Java** với giao diện đẹp, các thao tác sử dụng dễ dàng, thân thiện với người dùng.

Sau khi hoàn thành xong được được Website bán hàng điện tử Online, em có thêm những kỹ năng nền tảng để xây dựng được một phần mềm bằng ngôn ngữ Java và Maven. Bên cạnh đó, em cũng được mở rộng thêm các kiến thức chuyên ngành mới nhờ việc tìm hiểu một số kiến thức lập trình để áp dụng vào việc phát triển ứng dụng này. Nhờ vậy, kỹ năng tự học và vận dụng các kỹ năng mới của em được nâng cao hơn.

Khi thực hiện và hoàn thành dự án, em cũng đã biết thêm được nhiều kinh nghiệm quý giá khi xây dựng Cytoscape app nói riêng và phát triển phần mềm. Những kỹ năng này sẽ nền tảng để giúp em nâng cao trình độ bản thân, kinh nghiệm trong thực tế để có thể làm việc ngoài các doanh nghiệp sau này.

**Hướng phát triển của đề tài**

Sau khi hoàn thành đề tài xây dựng được **Phát triển ứng dụng Cytoscape tìm K-core/R-core của mạng lưới quy mô lớn bằng thuật toán song song trên Java**. Em sẽ tiếp tục nghiên cứu và phát triển phần mềm này này nhằm tăng các tính năng và tối ưu hóa tốc độ xử lý để đem lại hiệu quả cao hơn, phát triển thêm các chức nâng cao để đưa phần mềm vào ứng dụng thực tế.

# TÀI LIỆU THAM KHẢO

1. https://sci-hub.se/https:/pubmed.ncbi.nlm.nih.gov/25462327/
2. https://ieeexplore.ieee.org/abstract/document/7965211
3. https://ieeexplore.ieee.org/abstract/document/6189336
4. https://github.com/cytoscape/cytoscape/wiki//Cytoscape-App-Ladder
5. https://apps.cytoscape.org/
6. https://www.javadoc.io/doc/com.aparapi/aparapi/1.8.0/com/aparapi/Kernel.html