

**ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN**



**PBL4: DỰ ÁN HỆ ĐIỀU HÀNH &
MẠNG MÁY TÍNH**

Đề tài 407: Xây dựng chương trình định đường tập trung

GIẢNG VIÊN HƯỚNG DẪN: ThS. Nguyễn Văn Nguyên

SINH VIÊN THỰC HIỆN:

Trần Thị Hồng Nhung	LỚP: 21T_DT2	NHÓM: 21N10A
Hoàng Thị Hồng Thắm	LỚP: 21T_DT2	NHÓM: 21N10A

Đà Nẵng, 12/2023

MỤC LỤC

.....	1
MỤC LỤC	2
DANH SÁCH HÌNH VẼ.....	4
GIỚI THIỆU ĐỀ TÀI	5
Lời mở đầu.....	5
Bố cục của báo cáo	6
CHƯƠNG 1. CƠ SỞ LÝ THUYẾT	7
1.1. Định đường tập trung.....	7
1.1.1. Định đường.....	7
1.1.2. Bộ định tuyến	7
1.1.3. Bảng chọn đường.....	8
1.1.4. Cơ chế định đường	8
1.1.5. Định đường tập trung	8
1.2. Mô hình Client/Server	10
1.3. Giao thức TCP/IP.....	11
1.3.1. TCP.....	11
1.3.2. IP.....	11
1.3.3. Cổng (Port).....	11
1.3.4. Socket	11
1.3.5. Lập trình Socket với TCP trong Java	12
CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ BÀI TOÁN	14
2.1. Phân tích yêu cầu	14
2.2. Phân tích chức năng.....	15
2.2.1. Phía Client	15
2.2.2. Phía Server.....	15
2.3. Phân tích thiết kế chương trình.....	16
CHƯƠNG 3. TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ	17
3.1. Các công cụ sử dụng.....	17
3.2. Lập trình hệ thống.....	17

3.3. Ví dụ	19
3.4. Kết quả thực hiện	21
KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	28
Đánh giá kết quả	28
Hướng phát triển	28
TÀI LIỆU THAM KHẢO	29
[1]. Nguyễn Thúc Hải, <i>Mạng máy tính và các hệ thống mở</i> , NXB Giáo Dục, 1997.	29
[2]. Behrouz A. Forouzan, DeAnza College, <i>TCP/IP Protocol Suite</i> , second edition, McGraw-Hill, 2000.	29
[3]. Douglas E. Comer, <i>Computer Networks and Internets with Internet Applications</i> , Prentice-Hall, 1993.....	29
PHỤ LỤC	30

DANH SÁCH HÌNH VẼ

Hình 1.1-1. Đồ thị biểu diễn mô hình mạng	8
Hình 1.2-1 Mô hình Client/Server	10
Hình 1.3-1 Mô hình truyền tin Socket	12
Hình 2.1-1 Giao diện minh họa đề tài.....	14
Hình 3.3-1 Đồ thị ví dụ	19
Hình 3.4-1 Giao diện máy chủ.....	22
Hình 3.4-2 Giao diện sau khi có kết nối từ client	22
Hình 3.4-3 Giao diện đăng kí.....	23
Hình 3.4-4 Giao diện đăng nhập	23
Hình 3.4-5 Giao diện chính của client sau khi kết nối thành công.....	24
Hình 3.4-6 Giao diện vẽ đồ thị và in ma trận trọng số	24
Hình 3.4-7 Giao diện đọc file và vẽ đồ thị từ file khi nhấn nút “Upload File”	25
Hình 3.4-8 Giao diện hiển thị kết quả đường đi khi nhấn nút “Dijkstra”	25
Hình 3.4-9 Giao diện xử lý khi input không hợp lệ.....	26
Hình 3.4-10 Giao diện khi nhập đỉnh đích hoặc đỉnh nguồn không hợp lệ.....	26
Hình 3.4-11 Giao diện hiển thị kết quả trước khi cập nhật trọng số	27
Hình 3.4-12 Giao diện hiển thị kết quả sau khi cập nhật trọng số.....	27

GIỚI THIỆU ĐỀ TÀI

Lời mở đầu

Ngày nay, nhu cầu về công nghệ thông tin trong đời sống là đa dạng. Việc mở rộng các hệ thống truyền thông và ngày có nhiều máy vi tính kết nối vào mạng Internet. Với việc ứng dụng giao thức TCP/IP làm cho hệ thống mạng ngày càng rộng hơn và phát triển vượt bậc. Vấn đề về an ninh, bảo mật là một thế mạng mà giao thức này đem lại cho công nghệ truyền thông.

Việc định đường hiệu quả là một yếu tố quan trọng để đảm bảo hoạt động ổn định và hiệu suất cao của mạng. Để giải quyết vấn đề này, xây dựng một chương trình định đường hiệu quả là cần thiết. Do đó, trong đề tài này, chúng em tập trung vào việc nghiên cứu và ***“Xây dựng chương trình định đường tập trung sử dụng mô hình Client/Server và giao thức TCP/IP”***.

Mô hình định đường tập trung được thiết kế để quản lý và điều phối quá trình định đường từ một điểm tập trung. Trong mô hình này, máy chủ (Server) đảm nhận vai trò quan trọng trong việc thu thập thông tin định đường từ các thiết bị mạng và cung cấp thông tin này cho các máy khách (Client) khi có yêu cầu. Giao thức TCP/IP là giao thức mạng phổ biến và đáng tin cậy, được sử dụng để truyền dữ liệu giữa các thiết bị trong mạng.

Mục tiêu của chúng em là xây dựng một chương trình định đường tập trung sử dụng mô hình client-server và giao thức TCP/IP để cung cấp một giải pháp hiệu quả cho việc quản lý và điều phối định đường trong mạng máy tính. Chương trình sẽ cho phép các máy khách gửi yêu cầu định đường tới máy chủ, sau đó máy chủ sẽ xử lý yêu cầu và trả về thông tin định đường tương ứng. Việc xây dựng chương trình này không chỉ giúp sinh viên hiểu sâu hơn về cách thức hoạt động của mạng máy tính và giao thức truyền thông, mà còn phát triển kỹ năng lập trình và làm việc nhóm.

Trong quá trình nghiên cứu và phát triển, chúng em sẽ tập trung vào các yếu tố quan trọng như thiết kế giao thức truyền thông giữa Client và Server, thu thập thông tin thiết kế của mô hình mạng, xử lý dữ liệu định đường và đảm bảo tính chính xác của chương trình.

Kết quả dự kiến của đề tài này sẽ là một chương trình định đường tập trung sử dụng mô hình client-server và giao thức TCP/IP có thể áp dụng trong mạng máy tính thực tế. Chương trình sẽ mang lại lợi ích cho việc quản lý và vận hành mạng, giúp tối ưu hóa định tuyến và cải thiện hiệu suất mạng.

Đề tài này không chỉ mang lại giá trị lý thuyết mà còn tạo ra ứng dụng thực tiễn trong lĩnh vực quản lý mạng, điều chỉnh giao thông mạng, và tối ưu hóa định tuyến. Bằng cách thực hiện đề tài này, sinh viên sẽ có cơ hội tiếp cận và ứng dụng kiến thức một cách sâu rộng, từ đó nâng cao trình độ và sẵn sàng cho thị trường lao động ngày càng cạnh tranh trong lĩnh vực công nghệ thông tin và mạng máy tính.

Trong phần tiếp theo của đề tài, chúng tôi sẽ trình bày chi tiết về các bước nghiên cứu, thiết kế và triển khai chương trình định đường tập trung, cũng như phân tích kết quả đạt được và đề xuất các hướng phát triển tiếp theo.

Trong khuôn khổ đồ án, dưới sự hướng dẫn của thầy Nguyễn Văn Nguyên, giảng viên khoa Công nghệ thông tin, Trường Đại học Bách khoa – Đại học Đà Nẵng, chúng em đã nghiên cứu, tìm hiểu và thực hiện đề tài “***Xây dựng chương trình định đường tập trung***”. Do thời gian và kiến thức của chúng em còn hạn chế nên không tránh khỏi những sai sót nhất định trong quá trình thực hiện đồ án. Xin cảm ơn sự hỗ trợ từ thầy Nguyễn Văn Nguyên và rất mong nhận được sự góp ý từ quý thầy cô.

Bố cục của báo cáo

Báo cáo gồm 3 phần chính, thể hiện các nội dung sau:

Chương 1: Trình bày cơ sở lý thuyết

Chương 2: Trình bày quá trình phân tích và thiết kế bài toán

Chương 3: Trình bày kết quả thực thi chương trình, đánh giá kết quả và hướng phát triển

CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

1.1. Định đường tập trung

1.1.1. Định đường

Định tuyến là quá trình lựa chọn đường dẫn trong bất kỳ mạng nào. Một mạng máy tính được tạo thành từ nhiều máy được gọi là *các nút* và các đường dẫn hoặc liên kết để kết nối những nút đó. Quá trình giao tiếp giữa hai nút trong một mạng được kết nối với nhau có thể diễn ra thông qua nhiều đường dẫn khác nhau. Định tuyến là quá trình lựa chọn đường dẫn tốt nhất bằng một số quy tắc định trước.

Định tuyến giúp hoạt động giao tiếp mạng diễn ra hiệu quả. Lỗi giao tiếp mạng khiến người dùng chờ lâu để tải trang web. Lỗi giao tiếp mạng cũng có thể khiến máy chủ trang web bị sập vì không thể xử lý số lượng người dùng lớn. Định tuyến giúp giảm thiểu lỗi mạng bằng cách quản lý lưu lượng truy cập dữ liệu để mạng có thể phát huy tối đa khả năng của mình mà không gây ra tình trạng tắc nghẽn.

- Các thành phần của định đường:
 - o Bảng chọn đường
 - o Thông tin chọn đường
 - o Giải thuật, giao thức chọn đường

1.1.2. Bộ định tuyến

Bộ định tuyến là một thiết bị mạng kết nối các thiết bị máy tính và mạng với những mạng khác. Các bộ định tuyến chủ yếu đảm nhận ba chức năng chính.

- Xác định đường dẫn

Bộ định tuyến xác định đường dẫn mà dữ liệu sẽ đi khi dữ liệu di chuyển từ nguồn đến điểm đích. Bộ định tuyến cố gắng tìm đường dẫn tốt nhất bằng cách phân tích các chỉ số mạng như độ trì hoãn, dung lượng và tốc độ.

- Chuyển tiếp dữ liệu

Bộ định tuyến chuyển tiếp dữ liệu đến thiết bị tiếp theo trên đường dẫn đã chọn để cuối cùng là đến điểm đích của nó. Thiết bị và bộ định tuyến có thể nằm trên cùng một mạng hoặc trên các mạng khác nhau.

- Cân bằng tải

Đôi khi, bộ định tuyến có thể gửi bản sao của cùng một gói dữ liệu bằng cách sử dụng nhiều đường dẫn khác nhau. Bộ định tuyến làm thế này để giảm lỗi do tổn thất dữ liệu, tạo khả năng dự phòng và quản lý lưu lượng truy cập.

1.1.3. Bảng chọn đường

Chỉ ra danh sách các đường đi có thể, được lưu trong bộ nhớ của router.

Các thành phần chính của bảng chọn đường:

- Địa chỉ đích/mặt nạ mạng
- Router kế tiếp

1.1.4. Cơ chế định đường

Dữ liệu di chuyển dọc theo bất kỳ mạng nào dưới dạng gói dữ liệu. Mỗi gói dữ liệu có một tiêu đề chứa thông tin về điểm đích dự kiến của gói. Khi gói di chuyển đến đích của mình, một số bộ định tuyến có thể định tuyến gói đó nhiều lần. Các bộ định tuyến thực hiện quy trình này hàng triệu lần mỗi giây với hàng triệu gói.

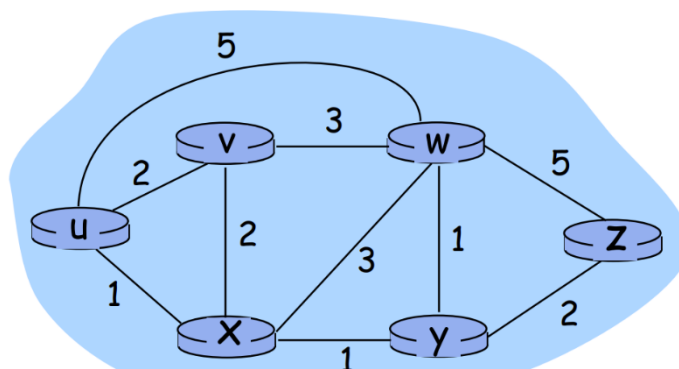
Khi một gói dữ liệu đến, trước tiên, bộ định tuyến sẽ tra cứu địa chỉ của gói trong bảng định tuyến. Quá trình này tương tự như một hành khách tham khảo lịch trình xe buýt để tìm tuyến xe buýt tốt nhất tới điểm đích của mình. Sau đó, bộ định tuyến chuyển tiếp hoặc di chuyển gói đến điểm tiếp theo trong mạng.

Ví dụ: khi bạn truy cập một trang web từ máy tính trong mạng văn phòng của bạn, các gói dữ liệu sẽ đi đến bộ định tuyến mạng văn phòng trước. Bộ định tuyến tra cứu gói có tiêu đề và xác định điểm đích của gói. Sau đó, bộ định tuyến sẽ tự tra cứu bảng nội bộ của nó và chuyển tiếp gói – đến bộ định tuyến tiếp theo hoặc đến một thiết bị khác, chẳng hạn như máy in – trong mạng đó.

1.1.5. Định đường tập trung

1.1.5.1 Biểu diễn mạng bởi đồ thị

- Đồ thị với các nút (bộ định tuyến) và các cạnh (liên kết)
- Chi phí cho việc sử dụng mỗi liên kết $c(x,y)$
 - Băng thông, độ trễ, chi phí, mức độ tắc nghẽn...
- Giả thuật chọn đường: Xác định đường đi ngắn nhất giữa hai nút bất kỳ



Hình 1.1-1. Đồ thị biểu diễn mô hình mạng

1.1.5.2 Định đường tập trung

- Đặc điểm

- Trong định đường tập trung, có một thiết bị duy nhất (thường là máy chủ định đường) giữ toàn bộ thông tin về mạng, bao gồm bảng định đường.
- Mọi yêu cầu định đường đều được gửi đến thiết bị tập trung này, và nó quyết định đường đi tối ưu dựa trên thông tin định đường mà nó lưu trữ.
- Mỗi router có thông tin đầy đủ về trạng thái của mạng
- Sử dụng giải thuật dạng “link state”

- Ưu điểm và nhược điểm

- Ưu điểm: Định đường tập trung đảm bảo rằng mọi quyết định định đường đều được thực hiện theo cùng một tiêu chuẩn và logic, giúp đảm bảo tính nhất quán trong mạng.
- Nhược điểm: Không linh hoạt khi mạng lớn hoặc có nhiều thay đổi định đường, và đòi hỏi nhiều tài nguyên để duy trì bảng định đường trên một thiết bị tập trung

1.1.5.2.1 Giải thuật dạng “link state”

Giải thuật Dijkstra's

- Mỗi nút đều có sơ đồ và chi phí mỗi link
 - Quảng bá “Link-state”
 - Mỗi nút có cùng thông tin
- Tìm đường đi chi phí nhỏ nhất từ một nút (“nguồn”) tới tất cả các nút khác
 - Dùng để xây dựng bảng chọn đường

Ký hiệu

- $G = (V, E)$: Đồ thị với tập đỉnh V và tập cạnh E
- $c(x, y)$: chi phí của liên kết x tới y ; $= \infty$ nếu không phải 2 nút kề nhau
- $d(v)$: chi phí hiện thời của đường đi từ nút nguồn tới nút đích.
- $p(v)$: nút ngay trước nút v trên đường đi từ nguồn tới đích
- T : Tập các nút mà đường đi ngắn nhất đã được xác định

Các thủ tục

- Init():
Với mỗi nút v , $d[v] = \infty$, $p[v] = \text{NIL}$, $d[s] = 0$
- Improve(u, v), trong đó (u, v) là một cạnh nào đó của G
 - if $d[v] > d[u] + c(u, v)$ then
 $d[v] = d[u] + c(u, v)$
 $p[v] = u$

Dijsktra's Algorithm

1. Init();
2. $T = \emptyset$;
3. Repeat
4. $u: u \notin T \mid d(u)$ là bé nhất ;
5. $T = T \cup \{u\}$;
6. for all $v \in \text{neighbor}(u)$ và $v \notin T$
7. update(u, v) ;
8. Until $T = V$

Hình 1.1-2 Thuật toán Dijsktra's

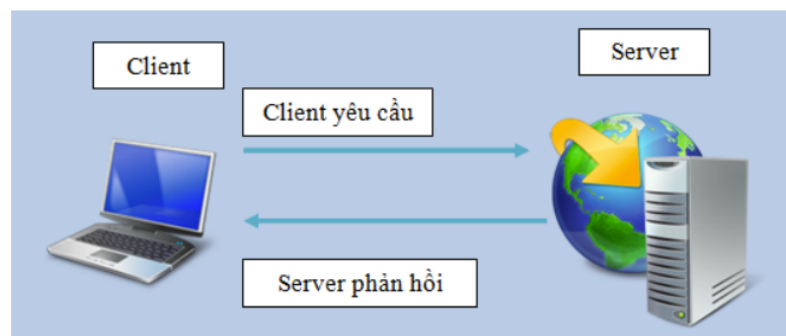
1.2. Mô hình Client/Server

Server được hiểu là máy chủ, thường là một hệ thống máy lớn, có bộ xử lý mạnh, có khả năng hoạt động đáng tin cậy, có khả năng lưu trữ dữ liệu lớn, nó chuyên quản lý tài nguyên (chủ yếu là cơ sở dữ liệu), cung cấp các dịch vụ mạng cho các máy khách (client) sử dụng. Bình thường nó chạy suốt thời gian thực và sẵn sàng chấp nhận các yêu cầu kết nối và các yêu cầu dịch vụ khác từ máy khách.

Client là máy khách, nó thường được sử dụng bởi người dùng cuối. Nó hoạt động dựa trên việc sử dụng dịch vụ mà máy server cung cấp để thực hiện các công việc mà người dùng cuối mong muốn.

Quy trình hoạt động của mô hình này lặp lại 2 quá trình như sau :

- Client gửi yêu cầu lên server.
- Server nhận được yêu cầu thì sẽ xử lý thích hợp và phản hồi lại client.



Hình 1.2-1 Mô hình Client/Server

1.3. Giao thức TCP/IP

1.3.1. TCP

Giao thức TCP (Transmission Control Protocol) là giao thức hướng kết nối (connection-oriented), nó đòi hỏi thiết lập kết nối trước khi bắt đầu gửi dữ liệu và kết thúc kết nối khi việc gửi dữ liệu hoàn tất theo đúng thứ tự: thiết lập kết nối, truyền dữ liệu và kết thúc kết nối.

1.3.2. IP

IP là địa chỉ của một máy tính trên mạng, dựa vào địa chỉ IP giao thức TCP có thể truyền dữ liệu chính xác từ một máy này qua máy kia thông qua hệ thống mạng. Ở trên mạng, một máy tính sẽ có một địa chỉ IP khác nhau, từ địa chỉ IP có thể biết được máy nào trên mạng và ngược lại.

1.3.3. Cổng (Port)

Với IP, giao thức TCP chỉ mới có thể truyền dữ liệu chính xác từ máy này qua máy kia mà chưa thể truyền chính xác đến từng ứng dụng trên máy được. Hiện nay, các hệ thống máy thông thường hoạt động theo chế độ đa nhiệm, nghĩa là có nhiều ứng dụng chạy cùng một lúc và trong đó có thể có nhiều ứng dụng sử dụng dịch vụ mạng. Yêu cầu, khi máy chủ A truyền dữ liệu cho một ứng dụng u trên máy B thì trên máy B phải đảm bảo dữ liệu đó phải đến được ứng dụng u, chứ không phải ứng dụng v.

Để thực hiện điều đó thì máy chủ A khi truyền dữ liệu đi thì trên dữ liệu đó có một thành phần thông tin giúp máy B xác định được đúng ứng dụng u. Phần thông tin đó chính là địa chỉ port trên máy B, nó có thể hiểu là lỗ cắm ảo trên máy B mà ứng dụng u đã đăng ký để độc quyền sử dụng nhận dữ liệu từ máy chủ A.

Trên thực tế, địa chỉ port là một số nguyên 2 byte có giá trị từ 0 đến 65535.

Nó có đặc điểm:

- Giá trị từ 0 đến 1023 là các cổng phổ biến dành cho các ứng dụng thông dụng như http: 80, mail: 25, ftp: 21, telnet: 23.... Các giá trị còn lại có thể được sử dụng linh hoạt.
- Mỗi cổng trong mỗi thời điểm được sử dụng cho tối đa là 1 ứng dụng. Ví dụ như một ứng dụng nào đó đã sử dụng cổng 55 thì ứng dụng khác không thể sử dụng cổng 55 đó nữa chừng nào ứng dụng trước đó chưa có đóng cổng 55 lại.

1.3.4. Socket

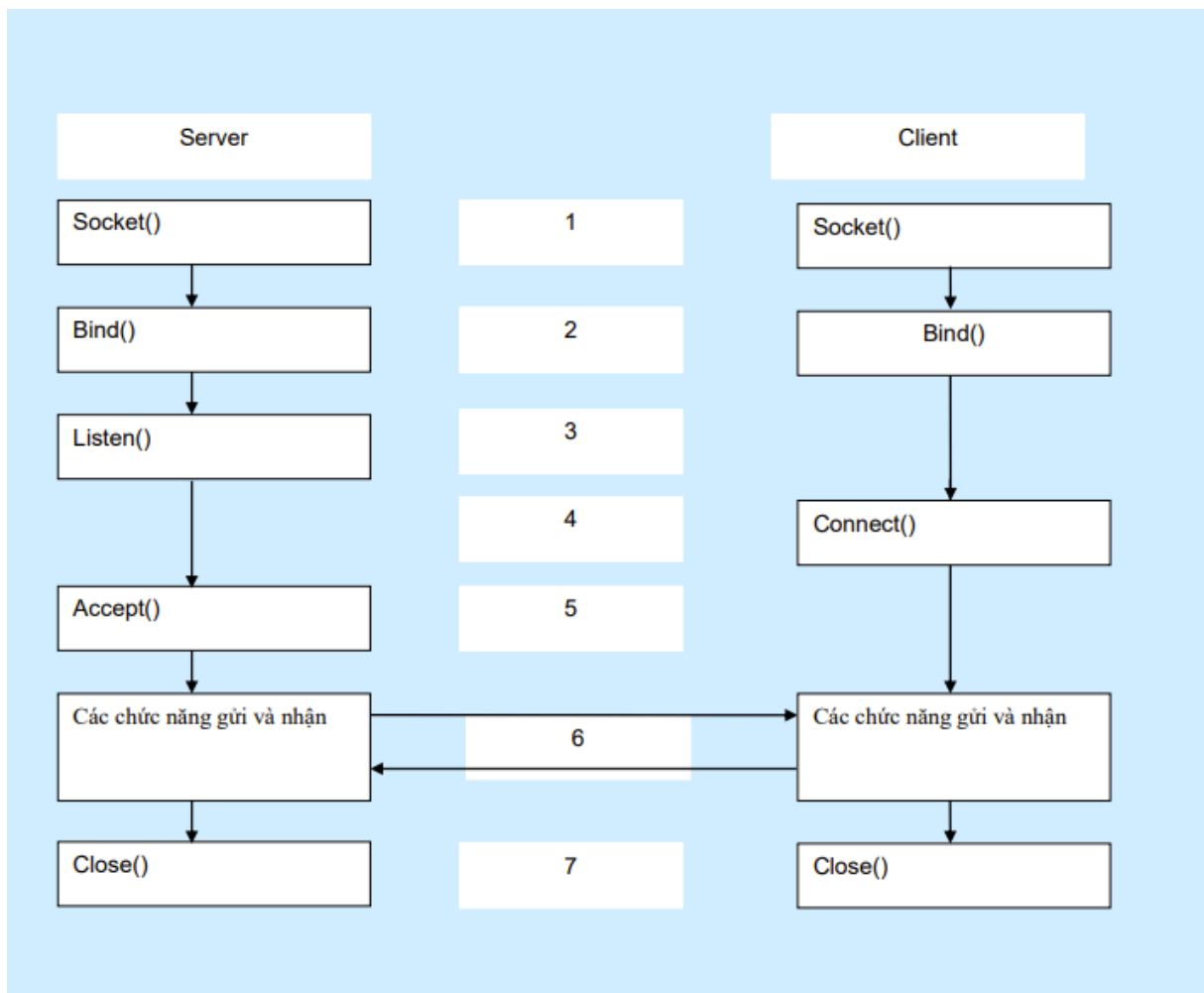
Socket là một khái niệm để định vị một dịch vụ của một máy tính trên mạng khi kết hợp hai khái niệm IP và Port lại.

- Socket có những nhiệm vụ như sau:

- Gắn một cổng trên máy
- Lắng nghe các kết nối.
- Kết nối/Đóng kết nối đến máy tính ở xa qua cổng đã gắn
- Gửi/nhận dữ liệu
- Lắng nghe dữ liệu đến

Thực chất, Socket chẳng qua là sự kết hợp giữa địa chỉ IP của máy tính và cổng Port mà ứng dụng sử dụng. Chính vì sự kết hợp đó mà nó trở thành một khái niệm mà từ đó, các ngôn ngữ lập trình có chứa các gói, các giao diện lập trình (API) để hỗ trợ các nhà lập trình dễ dàng trong việc lập trình liên quan đến mạng qua giao thức TCP.

1.3.5. Lập trình Socket với TCP trong Java



Hình 1.3-1 Mô hình truyền tin Socket

Trong lập trình mạng, việc sử dụng Socket và giao thức TCP/IP là rất phổ biến. Socket là một cách để thiết lập kết nối giữa các thiết bị trong mạng và truyền dữ liệu

qua mạng. Giao thức TCP/IP là một giao thức mạng tiêu chuẩn được sử dụng rộng rãi để truyền dữ liệu tin cậy giữa các máy tính.

Trong Java, lập trình Socket với TCP được thực hiện thông qua các lớp và giao thức có sẵn trong gói `java.net`. Gói này cung cấp các lớp và giao thức để tạo và quản lý các kết nối mạng.

Để lập trình Socket với TCP trong Java, bạn cần thực hiện các bước sau:

- Tạo Socket: Bạn cần tạo một đối tượng Socket để thiết lập kết nối với máy chủ (server) hoặc máy khách (client). Đối tượng Socket được khởi tạo bằng cách chỉ định địa chỉ IP và cổng của máy chủ.
- Thiết lập luồng dữ liệu: Sau khi tạo Socket, cần tạo luồng dữ liệu để truyền thông tin giữa client và server. Trong Java, có thể sử dụng lớp `InputStream` và `OutputStream` để đọc và ghi dữ liệu thông qua Socket.
- Gửi và nhận dữ liệu: Có thể sử dụng các phương thức của lớp `InputStream` và `OutputStream` để gửi và nhận dữ liệu qua Socket. Ví dụ, phương thức `write()` được sử dụng để gửi dữ liệu từ client đến server, trong khi phương thức `read()` được sử dụng để nhận dữ liệu từ server đến client.
- Đóng kết nối: Sau khi hoàn thành truyền dữ liệu, nên đóng kết nối Socket bằng cách gọi phương thức `close()`. Điều này giải phóng tài nguyên và đảm bảo kết nối được đóng đúng cách.

Việc lập trình Socket với TCP trong Java đòi hỏi kiến thức về cú pháp Java cơ bản và các khái niệm về mạng và giao thức TCP/IP. Ngoài ra, còn phải xử lý các ngoại lệ (exceptions) có thể xảy ra trong quá trình truyền dữ liệu.

Tuy nhiên, Java cung cấp các lớp và giao thức sẵn có để làm việc với Socket và TCP/IP, giúp đơn giản hóa quá trình lập trình mạng trong Java và cho phép tạo các ứng dụng mạng linh hoạt và tin cậy qua Socket.

CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ BÀI TOÁN

2.1. Phân tích yêu cầu

Bài toán:

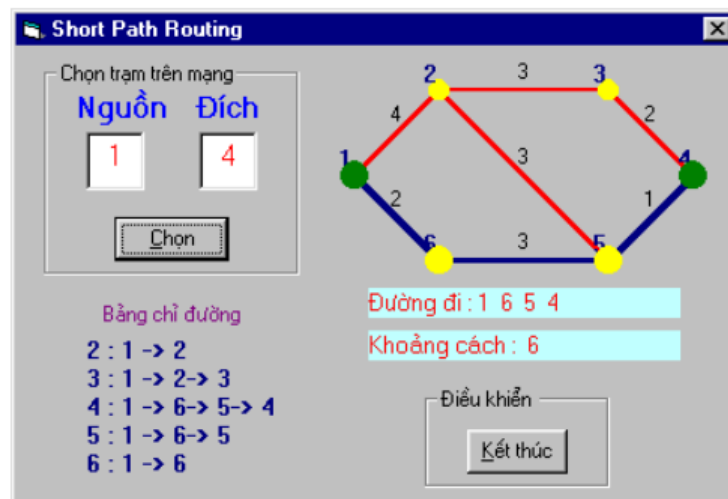
Xây dựng chương trình định đường tập trung.

Input:

- Đỉnh nguồn và đỉnh đích
- Ma trận của đồ thị của mạng

Output:

- Bảng lưu đường đi
- Số nút đường đi ngắn nhất tìm được đi qua
- Khoảng cách ngắn nhất
- Có giao diện thực hiện thao tác và hiển thị kết quả
- Giao diện gồm các thành phần tương tự như sau



Hình 2.1-1 Giao diện minh họa đề tài

Thuật toán: Sử dụng thuật toán Dijkstra

- Đồ thị trong thuật toán này gồm mỗi điểm đại diện cho mỗi điểm đại diện cho mỗi route của mạng, cung giữa 2 điểm của đồ thị là đường đi giữa 2 router trong mạng. Việc chọn đường đi giữa 2 nút trong mạng là tìm đường đi ngắn nhất giữa chúng. Thuật toán này được mô tả như sau:
 - Gọi C là tập hợp các đỉnh chưa được chọn, S là tập hợp các đỉnh được chọn. Tại mỗi thời điểm, tập S chứa các đỉnh mà khoảng cách nhỏ nhất

từ nguồn đến đỉnh đã được xác định. Khi đó, tập C chứa các đỉnh còn lại. Giải thuật bắt đầu từ tập S chứa đỉnh nguồn, khi giải thuật kết thúc thì tập S chứa tất cả các đỉnh của đồ thị. Tại mỗi bước ta chọn một đỉnh của tập C mà khoảng cách từ nguồn đến đích này là nhỏ nhất và đưa vào tập S. Ta nói rằng đường đi từ nguồn đến đích khác là riêng biệt nếu tất cả các đỉnh trung gian trên đường này đều nằm trong tập S. Tại mỗi bước của giải thuật, một mảng 1 chiều D dùng để chứa chiều dài đường đi riêng biệt.

- Giả sử các đỉnh của đồ thị được đánh số từ 1 đến n, không mất tính tổng quát ta chọn đỉnh nguồn là 1 và L là ma trận chứa chiều dài các cung.
- Ma trận được mô tả như sau:
- $L[i,i] = 0$ với $\forall i = 1..n$
- $L[i,j] \geq 0$ nếu tồn tại cung từ đỉnh i đến j
- $L[i,j] = \infty$ nếu không tồn tại cung từ đỉnh i đến j

2.2. Phân tích chức năng

2.2.1. Phía Client

- Kết nối với máy chủ.
- Đăng kí tài khoản, đăng nhập tài khoản
- Chọn file có chứa ma trận để vẽ đồ thị và hiển thị kết quả
- Thực hiện các thao tác trên giao diện để vẽ đồ thị.
- Nhập vào đỉnh nguồn và đỉnh đích để hiển thị ra kết quả đường đi ngắn nhất.
- Đăng xuất tài khoản

2.2.2. Phía Server

- Tạo cổng kết nối và sẵn sàng chờ nhận kết nối từ các client.
- Xử lý yêu cầu từ client.
- Trả về kết quả cho client.
- Đóng kết nối.

2.3. Phân tích thiết kế chương trình

Chương trình lập trình bằng ngôn ngữ Java, gồm các class để triển khai vẽ đồ thị phía client và xử lý theo mô hình Client/Server như sau:

- GUI
- DrawPanel
- Graph
- Edge
- Vertex
- Login
- Register
- Client
- Client_Handle
- Server

CHƯƠNG 3. TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

3.1. Các công cụ sử dụng

- IDE: Eclipse
 - Eclipse là một môi trường phát triển tích hợp (IDE - Integrated Development Environment) nổi tiếng và mạnh mẽ, được sử dụng chủ yếu cho việc phát triển ứng dụng Java.
 - Có hệ thống plugin mạnh mẽ, cho phép người phát triển tích hợp các công cụ và chức năng mở rộng.
 - Được sử dụng rộng rãi trong cộng đồng phát triển Java và nhiều dự án mã nguồn mở.
 - Hỗ trợ cho nền tảng đa dạng: Eclipse có thể chạy trên nhiều hệ điều hành như Windows, Linux, và macOS.
- Các công cụ hỗ trợ làm việc nhóm: Microsoft Teams, Messenger, Github.

3.2. Lập trình hệ thống

- Phía Server:

```
ServerSocket server = new ServerSocket(5056);  
System.out.println("Server is started");
```

Một ServerSocket được tạo trên cổng mặc định “5056” để lắng nghe các kết nối từ client.

```
while (true) {  
    Socket socket = server.accept();  
    System.out.println("New client connected");  
  
    // Tạo một luồng mới để xử lý client  
    Thread clientThread = new ClientHandler(socket);  
    clientThread.start();  
}
```

Server chờ đợi các kết nối mới bằng vòng lặp vô hạn (while (true)). Mỗi khi một client kết nối (server.accept()), một Socket mới được tạo để xử lý giao tiếp với client đó.

Một đối tượng ClientHandler mới được tạo với Socket tương ứng và một luồng mới được bắt đầu để xử lý client.

- Lớp ClientHandler

```
class ClientHandler extends Thread {  
    private Socket socket;  
  
    public ClientHandler(Socket socket) {  
        this.socket = socket;  
    }  
}
```

ClientHandler là một lớp con của Thread, được thiết kế để xử lý giao tiếp với một client cụ thể.

Trong hàm run(), nơi mà luồng mới bắt đầu, một cặp DataOutputStream và DataInputStream được tạo từ Socket để gửi và nhận dữ liệu từ client.

```
@Override
public void run() {
    try {
        DataOutputStream dos = new DataOutputStream(socket.getOutputStream());
        DataInputStream din = new DataInputStream(socket.getInputStream());
```

Trong vòng lặp vô hạn, server đọc một chuỗi từ client (din.readUTF()), sau đó gọi phương thức processMessage để xử lý tin nhắn và cuối cùng gửi kết quả về client (dos.writeUTF(send)).

```
        while (true) {
            String receive = din.readUTF();
            String send = processMessage(receive);
            dos.writeUTF(send);
        }
```

Nếu có lỗi trong quá trình đọc hoặc gửi dữ liệu, luồng này sẽ bắt ngoại lệ và thông báo rằng client đã ngắt kết nối.

```
    } catch (Exception e) {
        // Xử lý ngoại lệ hoặc đóng kết nối khi client ngắt kết nối
        System.out.println("Client disconnected");
    }
```

- Phía Client: một đối tượng client sử dụng sockets để kết nối và giao tiếp với một server.

- o Kết nối đến Server:

```
public Client(String serverAddress) {
    try {
        socket = new Socket(serverAddress, 5056);
        dataInputStream = new DataInputStream(socket.getInputStream());
        dataOutputStream = new DataOutputStream(socket.getOutputStream());
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Trong hàm khởi tạo (Client(String serverAddress)), một đối tượng Socket được tạo để kết nối đến server qua địa chỉ serverAddress và cổng 5056.

Một DataInputStream và một DataOutputStream cũng được tạo để đọc và ghi dữ liệu qua socket.

- Gửi dữ liệu từ Client đến Server:

```
public void sendMessage(String message) {  
    try {  
        dataOutputStream.writeUTF(message);  
        dataOutputStream.flush();  
        System.out.println("send");  
    } catch (IOException e) {  
        e.printStackTrace();  
    }  
}
```

Phương thức sendMessage(String message) được sử dụng để gửi dữ liệu từ client đến server. Dữ liệu ở đây gồm có đỉnh nguồn, đỉnh đích, số đỉnh, và ma trận trọng số được xử lý thành một chuỗi ở class GUI khi nhấn button tìm đường đi

Dữ liệu được gửi dưới dạng chuỗi UTF-8 qua dataOutputStream.

- Nhận dữ liệu được trả về từ Server:

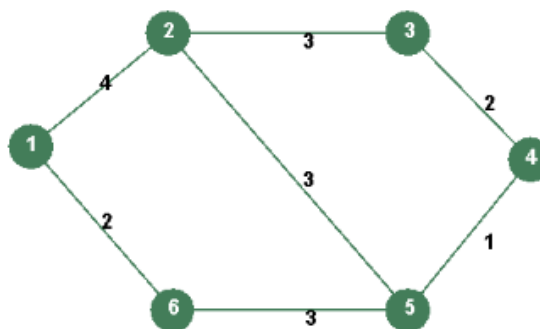
```
public String receiveMessage() {  
    try {  
        String response = dataInputStream.readUTF();  
  
        return dataInputStream.readUTF();  
    } catch (IOException e) {  
        e.printStackTrace();  
        return null;  
    }  
}
```

Phương thức receiveMessage() được sử dụng để nhận dữ liệu từ Server.

Dữ liệu được đọc thông qua dataInputStream dưới dạng chuỗi UTF-8. Dữ liệu này sẽ được xử lý và hiển thị ở GUI.

3.3. Ví dụ

Ta có đồ thị ví dụ như sau:



Hình 3.3-1 Đồ thị ví dụ

- Ta có được ma trận trọng số như sau:

Đỉnh	1	2	3	4	5	6
1	0	4	∞	∞	∞	2
2	4	0	3	∞	3	∞
3	∞	3	0	2	∞	∞
4	∞	∞	2	0	1	∞
5	∞	3	∞	1	0	3
6	2	∞	∞	∞	3	0

Bảng 3.3-1 Ma trận trọng số

- Giả sử chúng ta sẽ tìm đường đi ngắn nhất từ đỉnh 1 đến đỉnh 4.

○ **Bước 1:** Khởi tạo

Với đỉnh $v \in V$, gọi nhãn $d[v]$ là độ dài đường đi ngắn nhất từ s tới v , $t[v]$ là đỉnh được đi đến trước đỉnh v trên đường đi ngắn nhất. Ban đầu $d[v]$ được khởi gán như sau: ($d[s] = 0$ và $d[v] = \infty$ với $\forall v \neq s$). Nhãn của mỗi đỉnh có hai trạng thái tự do hay cố định, nhãn tự do có nghĩa là có thể còn tối ưu hơn được nữa và nhãn cố định tức là $d[v]$ đã bằng độ dài đường đi ngắn nhất từ s tới v nên không thể tối ưu thêm. Để làm điều này ta có thể sử dụng kỹ thuật đánh dấu: $Free[v] = TRUE$ hay $FALSE$ tùy theo $d[v]$ tự do hay cố định. Ban đầu các nhãn đều tự do.

○ **Bước 2:** Vòng lặp: gồm có hai thao tác:

- **Cố định nhãn:** Chọn trong các đỉnh có nhãn tự do, lấy ra đỉnh u là đỉnh có $d[u]$ nhỏ nhất, và cố định nhãn đỉnh u .
- **Sửa nhãn:** Dùng đỉnh u , xét tất cả những đỉnh v và sửa lại các $d[v]$, $t[v]$ theo công thức: $d[v] = \min(d[v], d[u] + c[u, v])$; $t[v] = u$;

Bước lặp sẽ kết thúc khi mà đỉnh đích f được cố định nhãn (tìm được đường đi ngắn nhất từ s tới f); hoặc tại thao tác cố định nhãn, tất cả các đỉnh tự do đều có nhãn là $+\infty$ (không tồn tại đường đi).

	Free	Not free	uMin	1	2	3	4	5	6
0	{1,2,3,4,5,6}			(0,1)	(∞ , -)	(∞ , -)	(∞ , -)	(∞ , -)	(∞ , -)
1	{2,3,4,5,6}	{1}	u=1	(0,1)	(4,1)	(∞ , -)	(∞ , -)	(∞ , -)	(2,1)
2	{2,3,4,5}	{1,6}	u=6	-	(4,1)	(∞ , -)	(∞ , -)	(5,6)	(2,1)
3	{3,4,5}	{1,2,6}	u=2	-	(4,1)	(7,2)	(∞ , -)	(5,6)	-

PBL4: DỰ ÁN HỆ ĐIỀU HÀNH & MẠNG MÁY TÍNH

4	{3,4}	{1,2,5,6}	u=5	-	-	(7,2)	(6,5)	(5,6)	-
5	{3}	{1,2,4,5,6}	u=4	-	-	(7,2)	(6,5)	-	-
6		{1,2,3,4,5,6}	u=3	-	-	(7,2)	-	-	-

- **Bước 3:** Thông báo và in ra đường đi ngắn nhất tìm được hoặc cho biết không tồn tại đường đi ($d[f] = +\infty$).

Bắt đầu từ đỉnh f , chúng ta sẽ sử dụng $t[f]$ để truy vết ngược lại đỉnh trước đó. Chúng ta đặt $u = f$; Lặp lại quá trình cho đến khi $t[u] = u$ (nghĩa là $u = s$): in ra giá trị u và gán $u = t[u]$.

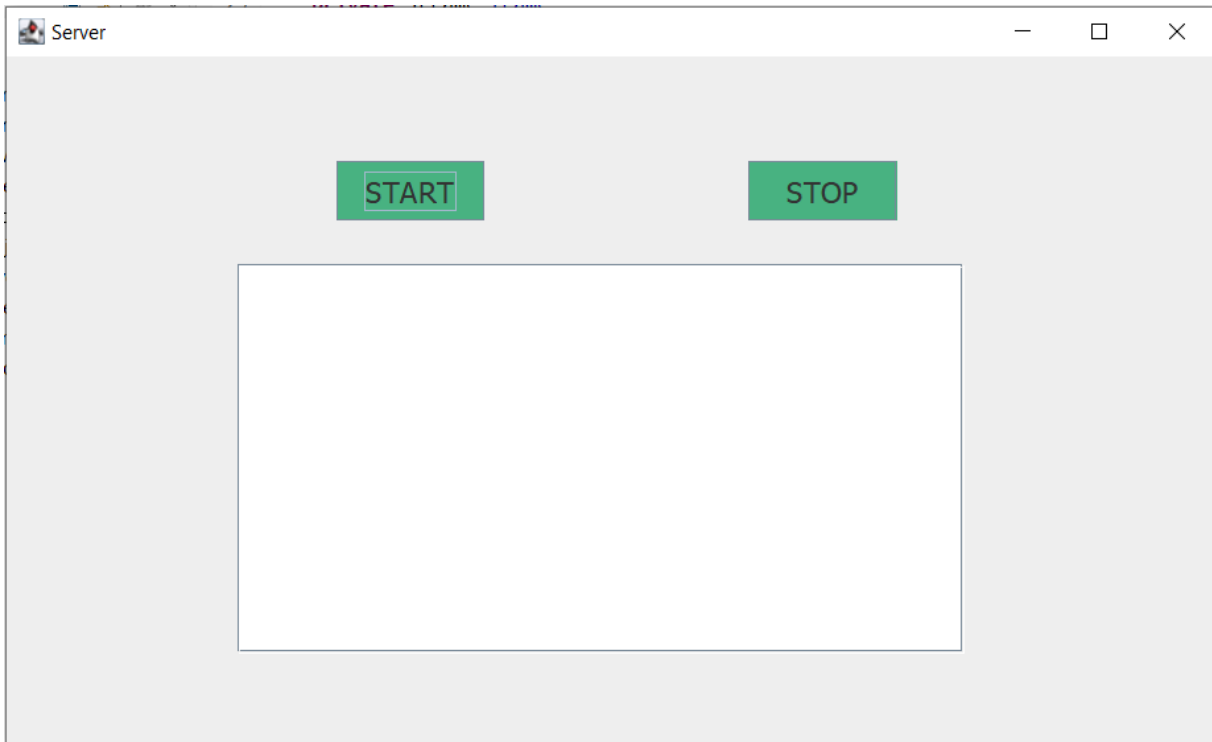
(d[1], t[1])	(d[2], t[2])	(d[3], t[3])	(d[4], t[4])	(d[5], t[5])	(d[6], t[6])
(0,1)	(∞ , -)	(∞ , -)	(∞ , -)	(∞ , -)	(∞ , -)
(0,1)	(4,1)	(∞ , -)	(∞ , -)	(∞ , -)	(2,1)
-	(4,1)	(∞ , -)	(∞ , -)	(5,6)	(2,1)
-	(4,1)	(7,2)	(∞ , -)	(5,6)	-
-	-	(7,2)	(6,5)	(5,6)	-
-	-	(7,2)	(6,5)	-	-
-	-	(7,2)	-	-	-

Chúng ta có thể kết luận rằng đường đi ngắn nhất từ 1 đến 4 là: $1 \rightarrow 6 \rightarrow 5 \rightarrow 4$ với độ dài đường đi ngắn nhất là 6.

3.4. Kết quả thực hiện

- Về cơ bản, chương trình thực hiện các chức năng là tạo kết nối, tìm ra đường đi tốt nhất với đầu vào là một đồ thị vô hướng
- Một số hình ảnh hóa các chức năng của chương trình

- Giao diện máy chủ



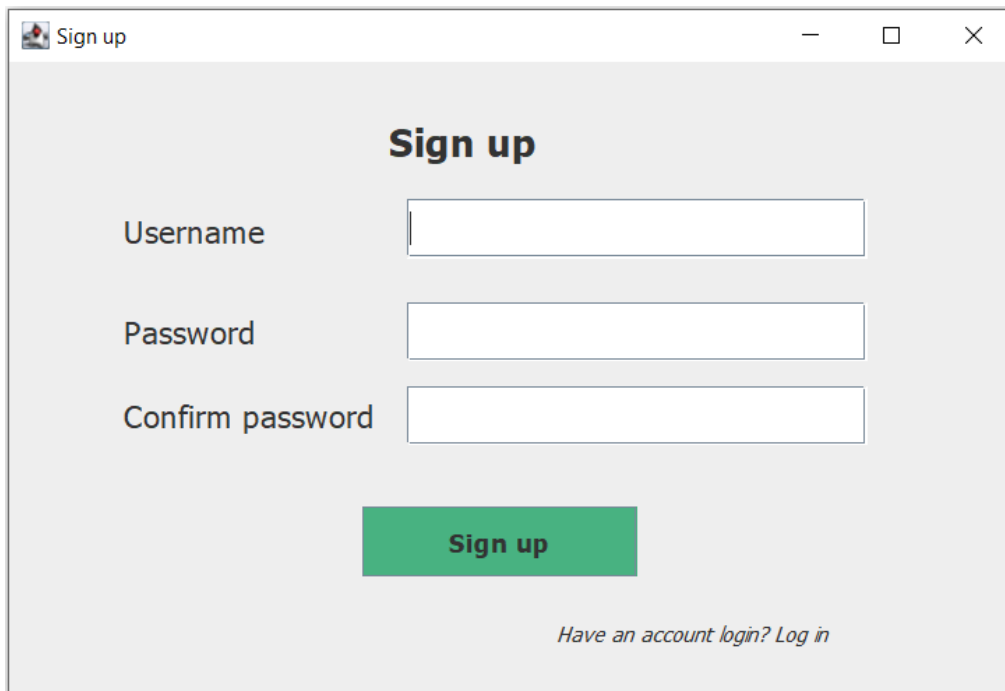
Hình 3.4-1 Giao diện máy chủ

- Giao diện sau khi có kết nối từ client



Hình 3.4-2 Giao diện sau khi có kết nối từ client

○ Giao diện đăng kí

A screenshot of a web browser window titled "Sign up". The window has a light gray background. At the top, the title "Sign up" is displayed in bold. Below the title, there are three input fields: "Username", "Password", and "Confirm password". Each field is a white rectangle with a thin gray border. Below the input fields, there is a green button with the text "Sign up" in white. At the bottom right, there is a link that says "Have an account login? Log in".

Sign up

Username

Password

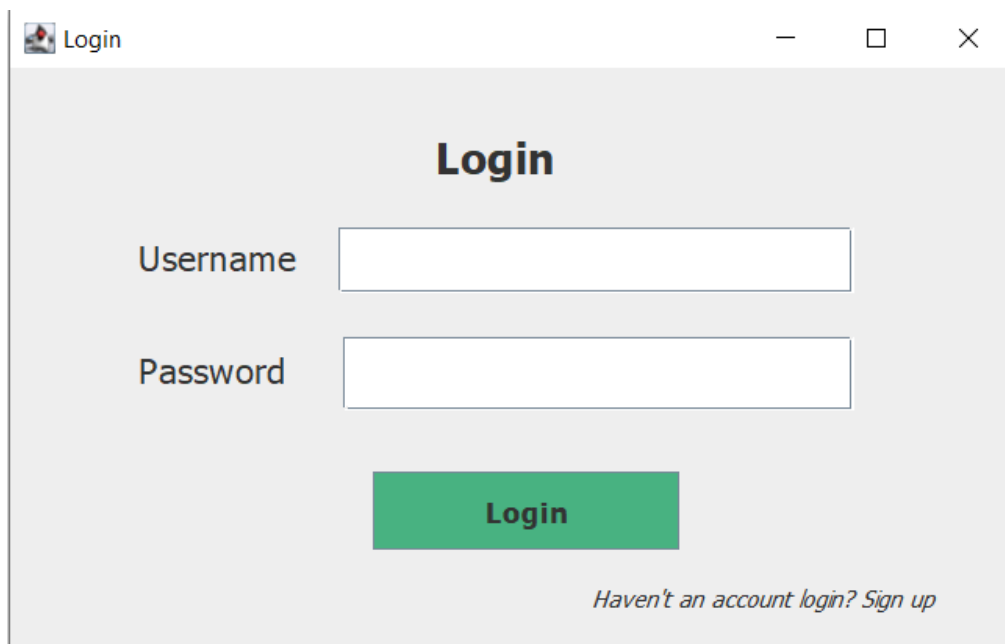
Confirm password

Sign up

Have an account login? Log in

Hình 3.4-3 Giao diện đăng kí

○ Giao diện đăng nhập

A screenshot of a web browser window titled "Login". The window has a light gray background. At the top, the title "Login" is displayed in bold. Below the title, there are two input fields: "Username" and "Password". Each field is a white rectangle with a thin gray border. Below the input fields, there is a green button with the text "Login" in white. At the bottom right, there is a link that says "Haven't an account login? Sign up".

Login

Username

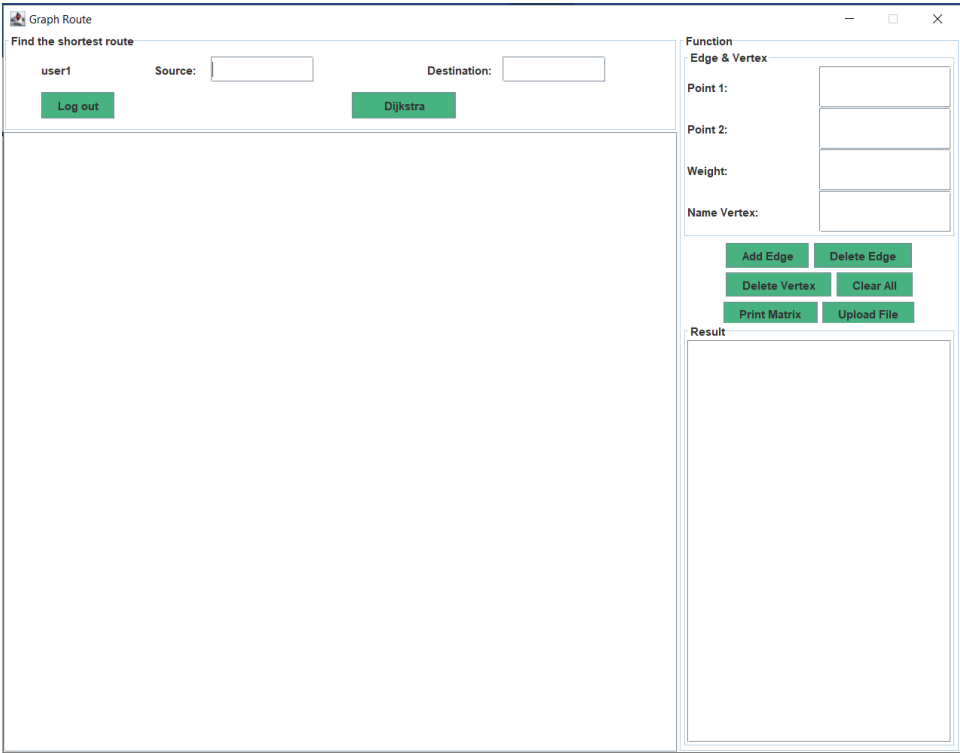
Password

Login

Haven't an account login? Sign up

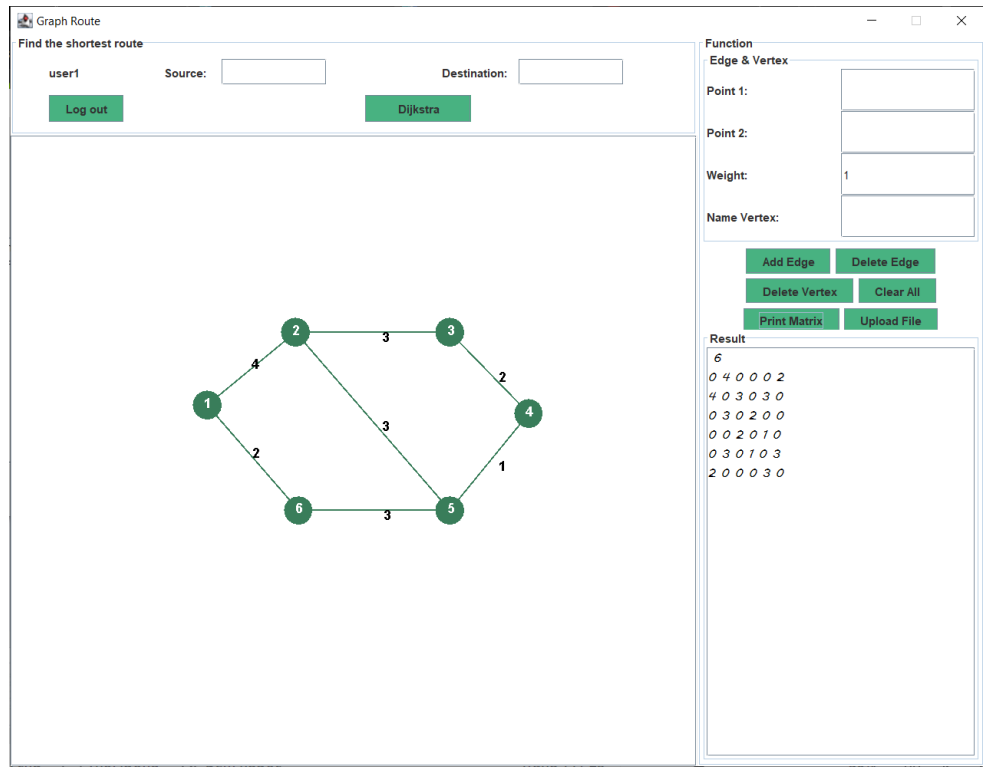
Hình 3.4-4 Giao diện đăng nhập

- Giao diện chính của client sau khi kết nối thành công



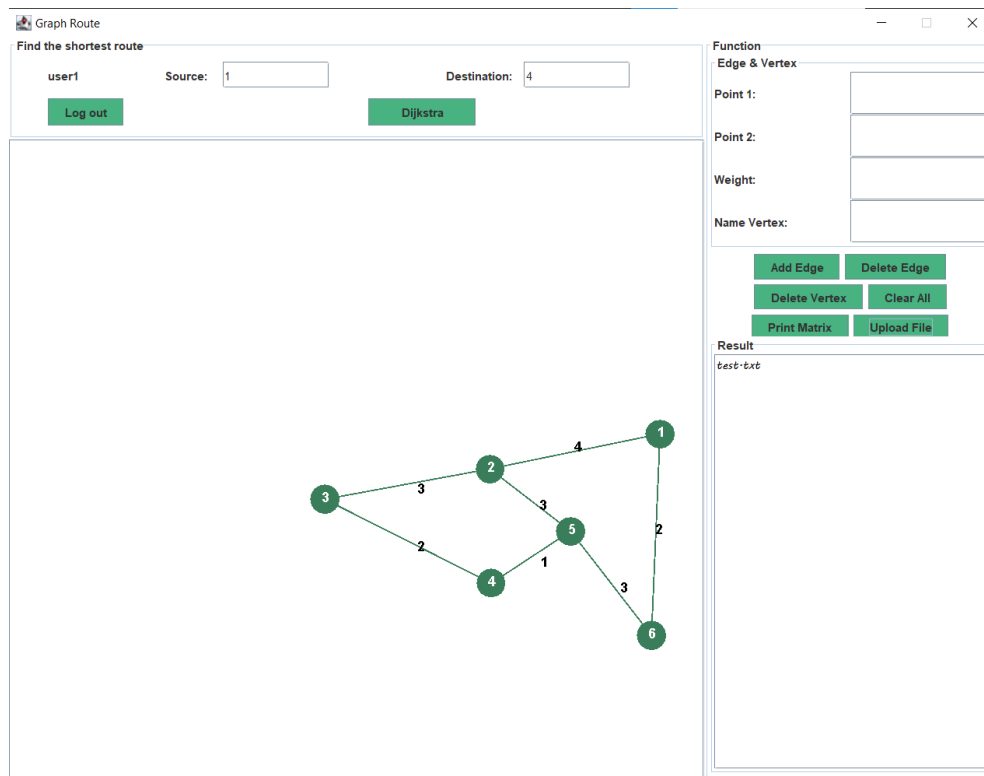
Hình 3.4-5 Giao diện chính của client sau khi kết nối thành công

- Giao diện vẽ đồ thị và in ma trận trọng số



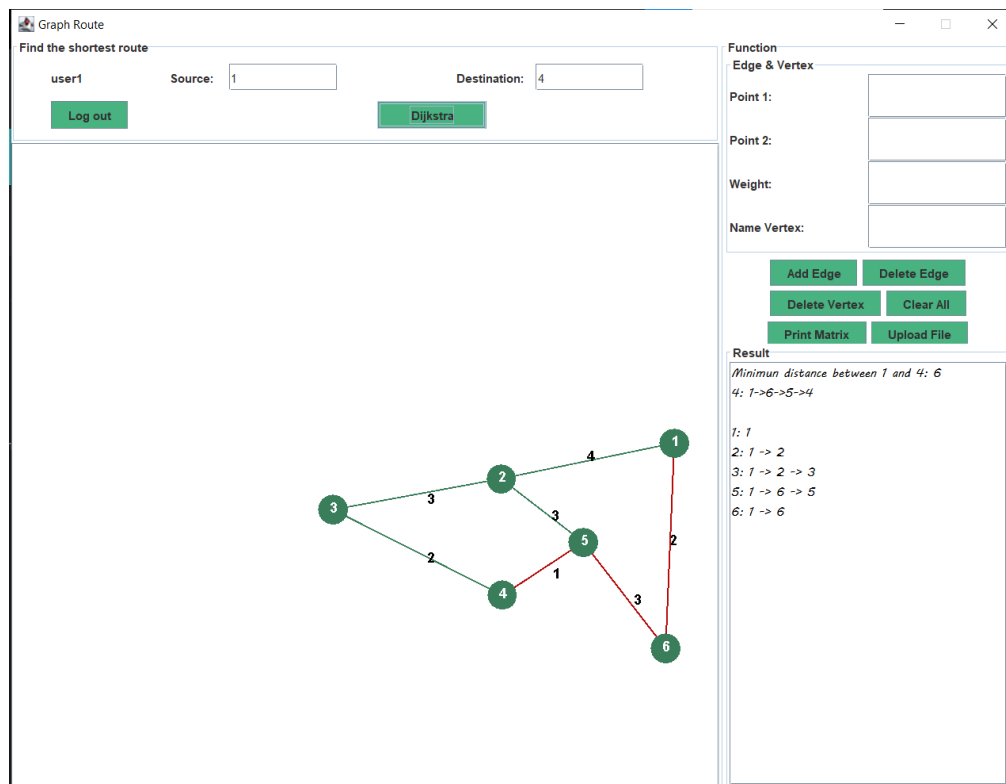
Hình 3.4-6 Giao diện vẽ đồ thị và in ma trận trọng số

○ Giao diện đọc file và vẽ đồ thị từ file



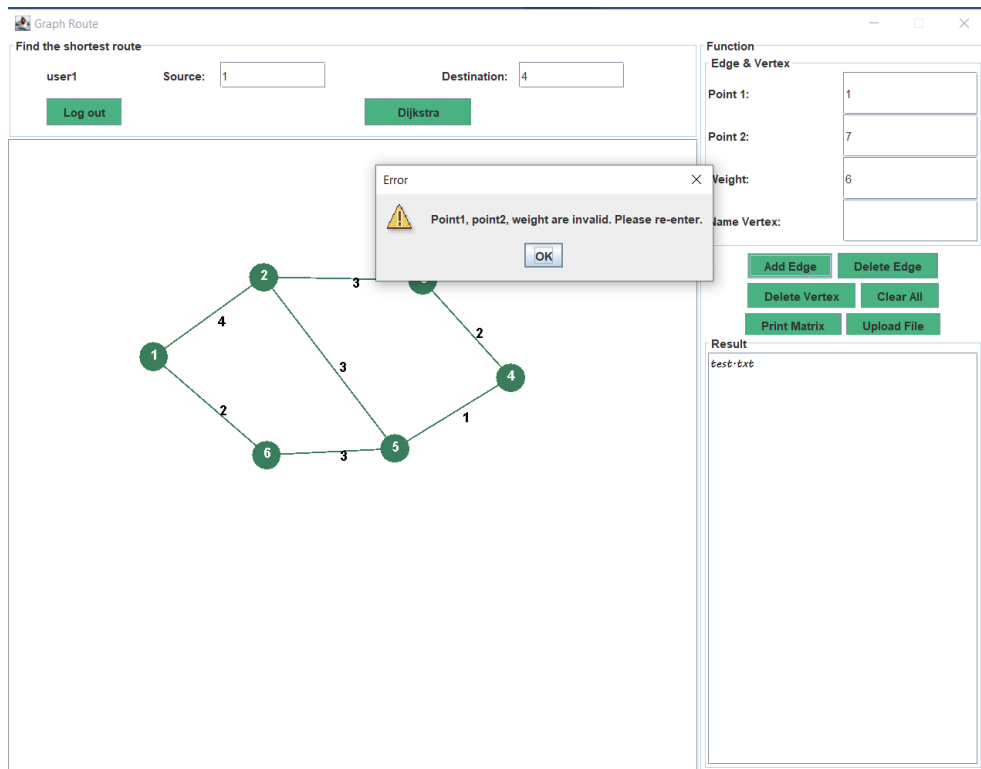
Hình 3.4-7 Giao diện đọc file và vẽ đồ thị từ file khi nhấn nút “Upload File”

○ Giao diện hiển thị kết quả tìm đường đi ngắn nhất từ đỉnh đích đến đỉnh nguồn



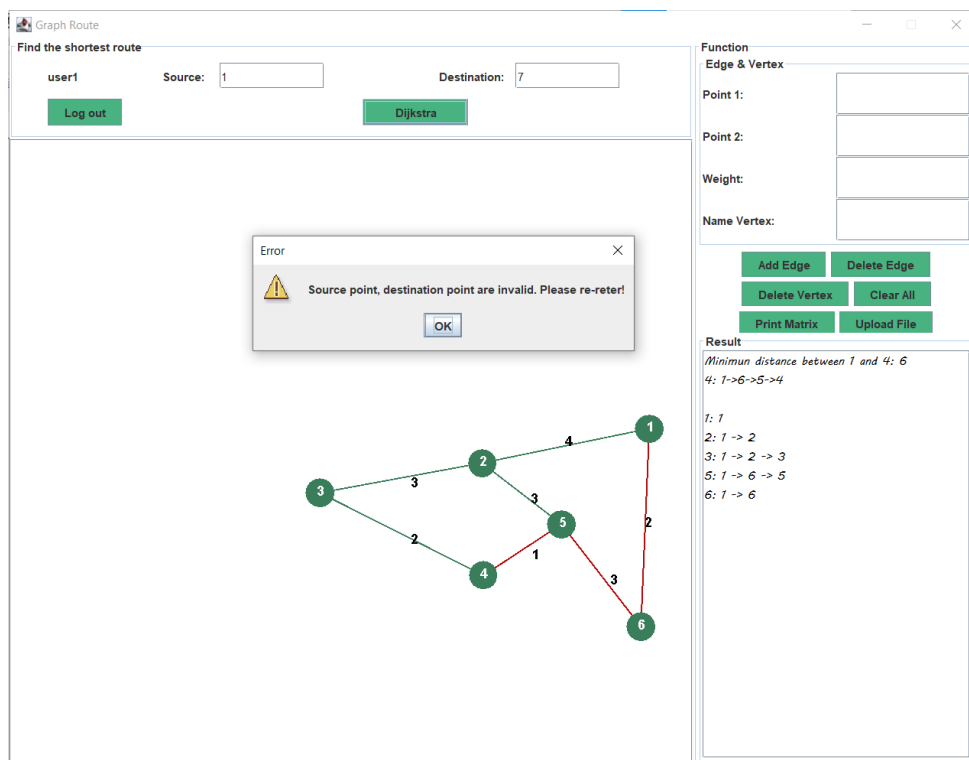
Hình 3.4-8 Giao diện hiển thị kết quả đường đi khi nhấn nút “Dijkstra”

- Giao diện xử lý khi input đưa vào không hợp lệ



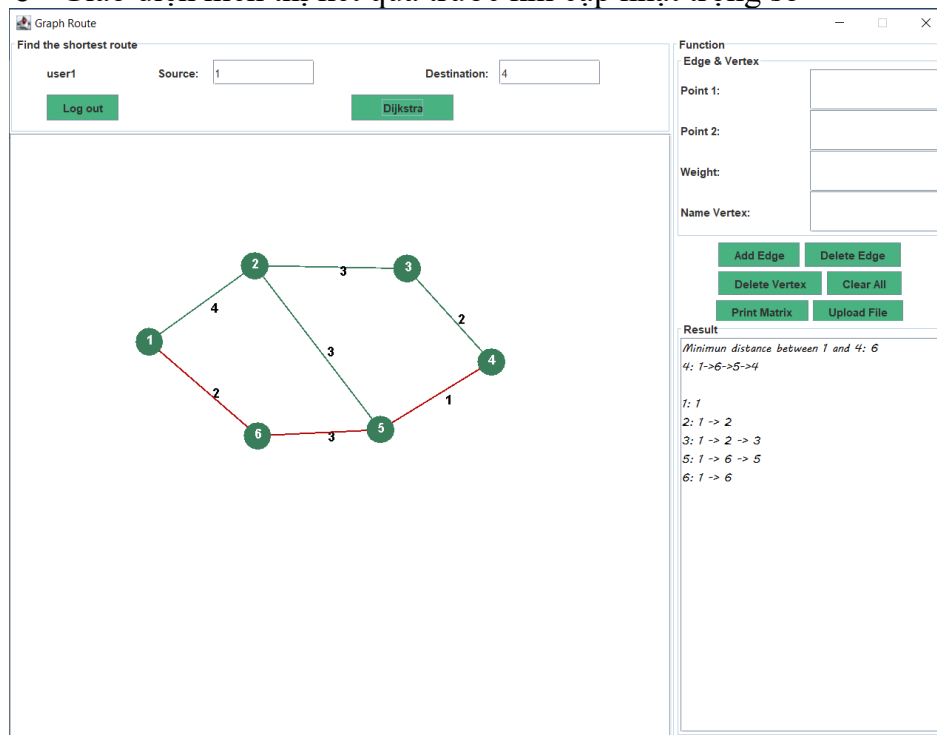
Hình 3.4-9 Giao diện xử lý khi input không hợp lệ

- Giao diện khi nhập đỉnh đích hoặc đỉnh nguồn không hợp lệ



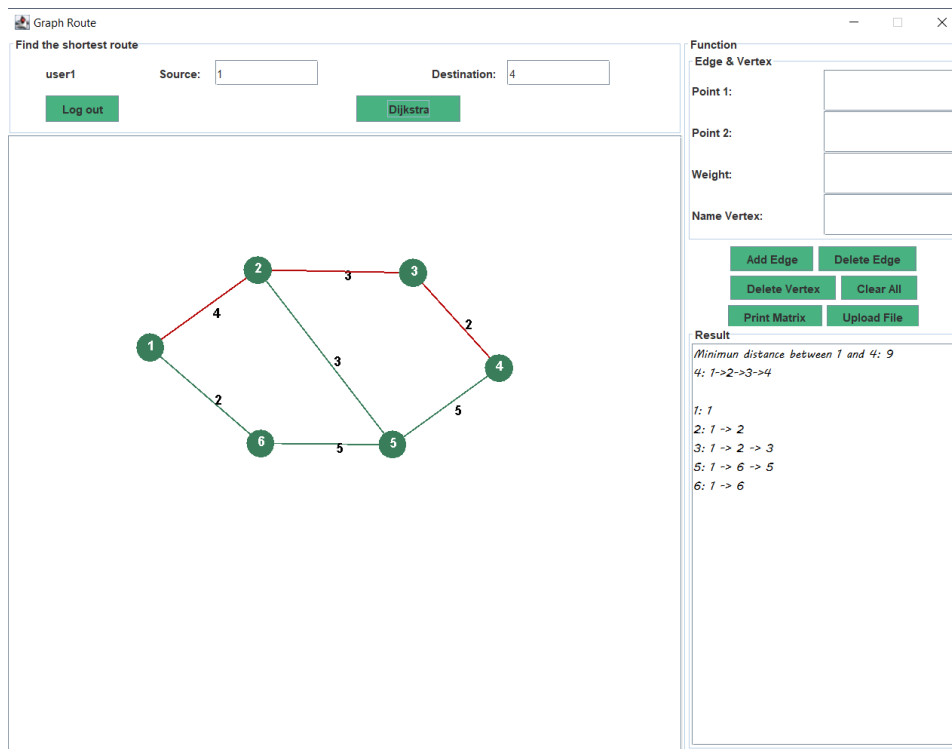
Hình 3.4-10 Giao diện khi nhập đỉnh đích hoặc đỉnh nguồn không hợp lệ

○ Giao diện hiển thị kết quả trước khi cập nhật trọng số



Hình 3.4-11 Giao diện hiển thị kết quả trước khi cập nhật trọng số

○ Giao diện hiển thị kết quả sau khi cập nhật trọng số



Hình 3.4-12 Giao diện hiển thị kết quả sau khi cập nhật trọng số

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Đánh giá kết quả

- Chương trình về cơ bản đã đáp ứng được yêu cầu của đề tài là định đường tập trung trong mô hình mạng. Đã xây dựng được giao diện cơ bản để người dùng có thể thao tác một cách dễ dàng.
- Tuy nhiên, chương trình vẫn còn một số hạn chế:
 - o Giao diện còn khá đơn giản, chưa đẹp mắt. Cần cải tiến để nâng cao trải nghiệm của người dùng.
 - o Chỉ đáp ứng yêu cầu cho đồ thị vô hướng. Do đó, cần phát triển thuật toán, nâng cấp phiên bản của chương trình để có thể thực hiện yêu cầu đối với cả đồ thị có hướng.
 - o Chưa đánh giá được độ tối ưu của thuật toán đang dùng so với các thuật toán khác.

Hướng phát triển

Trong thời gian tìm hiểu, nghiên cứu cơ sở lý thuyết và triển khai thực hiện đề tài, nhóm em đã có cơ hội được tìm hiểu giao thức TCP/IP và mô hình Client/server cũng như các thuật toán tìm đường đi, cách hoạt động của mô hình mạng để vận dụng, thực hành xây dựng được chương trình định đường tập trung.

Trong quy mô của đề tài, sản phẩm được triển khai còn khá đơn giản về giao diện, chưa được đầu tư kỹ lưỡng để mở rộng các chức năng, tối ưu hóa thuật toán. Vì vậy, nhóm em đã đề ra một số hướng phát triển:

- o Cải thiện giao diện sao cho thân thiện với người dùng, bắt mắt và trực quan hơn.
- o Có thể đáp ứng một lượng lớn các client truy cập một lúc đến server.
- o Nghiên cứu thêm thuật toán để có thể áp dụng với cả đồ thị có hướng cũng như là tối ưu hóa thuật toán.
- o Nghiên cứu về khía cạnh bảo mật thông tin cho client khi sử dụng.

TÀI LIỆU THAM KHẢO

- [1]. Nguyễn Thúc Hải, *Mạng máy tính và các hệ thống mở*, NXB Giáo Dục, 1997.
- [2]. Behrouz A. Forouzan, DeAnza College, *TCP/IP Protocol Suite*, second edition, McGraw-Hill, 2000.
- [3]. Douglas E. Comer, *Computer Networks and Internets with Internet Applications*, Prentice-Hall, 1993.

PHỤ LỤC

Link source code: https://github.com/ch1ckenD4nce/PB4_HDH_MMT.git