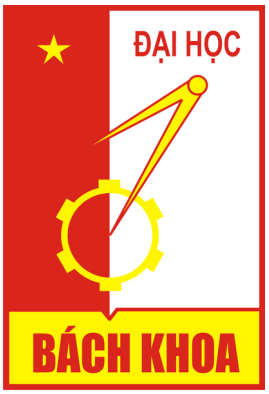
ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC



ĐỒ ÁN II

Tìm hiểu về bài toán Facility Location
Áp dụng thực hiện bài toán chọn vị trí máy bán nước tự động tại Đại học Bách Khoa Hà Nội

Giảng viên hướng dẫn: TS. Lê Hải Hà
Sinh viên thực hiện: Hà Quang Hanh
MSSV: 20195870
Lớp: Toán Tin 01 K64

Hà Nội, 2023

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

1. Mục tiêu và nội dung của đồ án:

2. Kết quả đạt được:

3. Ý thức làm việc của sinh viên:

Hà Nội, tháng . . . năm 2023

Giảng viên hướng dẫn

TS. Lê Hải Hà

Lời cảm ơn

Để có thể hoàn thành đồ án này, em xin được gửi lời cảm ơn chân thành và sâu sắc đến thầy TS. Lê Hải Hà, thầy đã tận tình giảng dạy và hướng dẫn em trong suốt quá trình học tập và làm bài báo cáo.

Hà Nội, tháng . . . năm 2023
Tác giả đồ án

Hà Quang Hanh

Mục lục

Mở đầu **4**

Chương 1. Tổng quan về bài toán vị trí (Facility Location) **5**

1.1 Giới thiệu chung 5

1.2 Một số kiến thức tổng quan 7

1.2.1 Các lớp bài toán 7

1.2.2 Độ phức tạp tính toán của bài toán 10

1.3 Bài toán chọn vị trí một thiết bị cơ bản 11

1.4 Bài toán vị trí cơ sở không hạn chế khả năng 12

Chương 2. Bài toán P-Median **14**

2.1 Giới thiệu bài toán 14

2.2 Mô hình bài toán 15

2.3 Một số thuật toán Heuristic giải bài toán P-Median 18

2.3.1 Mô hình thuật toán 18

2.3.2 Độ phức tạp của thuật toán 30

Chương 3. Bài toán thử nghiệm và đánh giá kết quả **32**

3.1 Giới thiệu bài toán 32

3.2 Mô hình bài toán và hàm mục tiêu 33

3.3 Cài đặt và thử nghiệm 36

3.3.1 Môi trường và các công cụ sử dụng 36

3.3.2 Kết quả chạy chương trình 37

Tổng kết **39**

Mở đầu

Trong cuộc sống, việc đạt lợi nhuận cao hay thấp trong kinh doanh buôn bán, cung cấp dịch vụ phụ thuộc rất nhiều yếu tố. Trong đó, có một yếu tố quan trọng đầu tiên, đóng góp một phần rất lớn đó là xác định được địa điểm đặt dịch vụ thuận lợi (nơi cung cấp dịch vụ). Có rất nhiều tiêu chí đặt ra khi chọn địa điểm: thuận tiện về giao thông, là nơi tập trung đông dân cư...để làm sao thu được lợi nhuận cao nhất. Đặc biệt, đối với các trường hợp khẩn cấp như cứu thương, cứu hỏa thì yêu cầu về khoảng cách nhỏ nhất là vô cùng quan trọng, có thể nói là quan trọng nhất trong các yếu tố. Tất cả các vấn đề này đều là mô hình của bài toán chọn vị trí cơ sở (Facility Loction).

Trong Đồ án 2 này, em sẽ giới thiệu về khái niệm bài toán chọn vị trí (Facility Location), mô hình bài toán P-median và một số thuật toán giải bài toán này. Qua đó, áp dụng mô hình bài toán P-median và các giải thuật vào thực hiện bài toán chọn vị trí đặt cây bán nước tự động tại Đại học Bách Khoa Hà Nội.

Chương 1

Tổng quan về bài toán vị trí (Facility Location)

1.1 Giới thiệu chung

Trong cuộc sống, việc đạt lợi nhuận cao hay thấp trong kinh doanh buôn bán, cung cấp dịch vụ phụ thuộc rất nhiều yếu tố. Trong đó, có một yếu tố quan trọng đầu tiên, đóng góp một phần rất lớn đó là xác định được địa điểm đặt dịch vụ thuận lợi (nơi cung cấp dịch vụ). Có rất nhiều tiêu chí đặt ra khi chọn địa điểm: thuận tiện về giao thông, là nơi tập trung đông dân cư... để làm sao thu được lợi nhuận cao nhất. Đặc biệt, đối với các trường hợp khẩn cấp như cứu thương, cứu hỏa thì yêu cầu về khoảng cách nhỏ nhất là vô cùng quan trọng, có thể nói là quan trọng nhất trong các yếu tố.

Yêu cầu của bài toán vị trí cơ sở là tìm phương án đặt các trạm dịch vụ ở đâu để thời gian di chuyển từ nơi xa nhất (hoặc ngược lại, từ các trạm dịch vụ đến nơi khách hàng) là nhỏ nhất có thể. Còn với các dịch vụ phổ biến như trạm xăng, thùng phiếu, bốt điện thoại,... thì yêu cầu lại là tổng chi phí từ khách hàng (hay người có nhu cầu) đến địa điểm phục vụ gần khách hàng nhất là nhỏ nhất.

Bài toán vị trí cơ sở (ở đây có thể là chọn vị trí cho thiết bị, nhà máy hoặc thiết bị) là một bài toán tối ưu trong lĩnh vực quản lý sản xuất và kinh tế.

Các bài toán về chọn vị trí cơ sở giải quyết câu hỏi đặt cở sở hay thiết bị ở

đâu. Mục tiêu của bài toán chọn vị trí thường là tối thiểu hóa chi phí hoặc tối đa hóa lợi nhuận. Để giải quyết bài toán này, cần xác định các yếu tố quan trọng như chi phí đầu tư, chi phí vận hành, chi phí vận chuyển, tốc độ phục vụ, sự tiếp cận thị trường,...

Bài toán chọn vị trí (Facility Location) là một bài toán quan trọng và có nhiều ứng dụng trong thực tế. Dưới đây là một số ví dụ về ứng dụng của bài toán chọn vị trí:

- Quản lý chuỗi cung ứng: Bài toán chọn vị trí được sử dụng để xác định vị trí tốt nhất để đặt các trung tâm phân phối, nhà kho, hoặc cửa hàng. Những quyết định này có thể giúp tối ưu hóa hoạt động chuỗi cung ứng và giảm chi phí.

- Kế hoạch sản xuất: Bài toán chọn vị trí cũng được sử dụng để quyết định vị trí tốt nhất để đặt các nhà máy sản xuất hoặc trung tâm gia công. Những quyết định này có thể giúp tối ưu hóa quy trình sản xuất và giảm chi phí.

- Dịch vụ chăm sóc sức khỏe: Bài toán chọn vị trí có thể được sử dụng để quyết định vị trí tốt nhất để đặt các bệnh viện, trạm cứu hộ, hoặc các thiết bị y tế khác. Những quyết định này có thể giúp tối ưu hóa việc cung cấp dịch vụ chăm sóc sức khỏe và giảm thời gian phục hồi cho các bệnh nhân.

- Mạng lưới viễn thông: Bài toán chọn vị trí cũng được sử dụng để quyết định vị trí tốt nhất để đặt các trạm phát sóng điện thoại di động, trạm thu phát sóng truyền hình và các thiết bị viễn thông khác. Những quyết định này có thể giúp tối ưu hóa mạng lưới viễn thông và cải thiện chất lượng dịch vụ.

- Vận tải: Bài toán chọn vị trí cũng được sử dụng để quyết định vị trí tốt nhất để đặt các điểm đỗ xe, các trạm dừng xe buýt hoặc các trung tâm vận chuyển. Những quyết định này có thể giúp tối ưu hóa hoạt động vận tải và giảm thời gian di chuyển.

1.2 Một số kiến thức tổng quan

1.2.1 Các lớp bài toán

Bài toán quyết định

Bài toán quyết định là một trong những đối tượng trọng tâm nghiên cứu của lý thuyết độ phức tạp tính toán. Bài toán quyết định là bài toán mà đầu ra chỉ có thể là 'yes' hoặc 'no' (Đúng/sai, 0/1). Đối với một bài toán quyết định, có những bộ dữ liệu vào của nó có câu trả lời (đầu ra) là 'yes' và cũng có những bộ dữ liệu vào có câu trả lời là 'no'. Những bộ dữ liệu vào có câu trả lời 'yes' ('no') sẽ được gọi là bộ dữ liệu vào 'yes' ('no').

Ví dụ 1:

- Bài toán xác định tính chính phương: "Hỏi số nguyên dương n có là số chính phương hay không?" → n=16 là bộ dữ liệu vào 'yes', còn n=15 là bộ dữ liệu vào 'no' của bài toán.
- Bài toán tổng con: "Cho tập I gồm n số nguyên dương $x_1, x_2, \ldots, x_n$ và số nguyên dương T. Hỏi có thể tìm được tập con S của I với tổng các số trong S là bằng T?"
- Bài toán người du lịch dạng quyết định: "Tồn tại hay chăng hành trình của người du lịch với tổng chi phí không vượt quá số K cho trước?

Ví dụ các lớp bài toán

Các bài toán quyết định có một đặc điểm chung, đó là để xác nhận câu trả lời 'yes' đối với bộ dữ liệu vào 'yes' của chúng, ta có thể đưa ra một lời giải có thể kiểm chứng được nhanh chóng xác nhận câu trả lời 'yes' cho bộ dữ liệu vào 'yes' đó.

Ví dụ 2:

1. Đối với bài toán kiểm tra tích hợp số: "Có phải số n là hợp số?", để xác nhận câu trả lời 'yes' cho đầu vào n, ta có thể đưa ra một ước số b ($1 < b < n$) của n. Để kiểm tra xem b đúng là ước số của n ta có thể thực

hiện phép chia n cho b sau thời gian đa thức. Trong ví dụ này b là bằng chứng (vì b < n) và dễ kiểm dàng kiểm chứng: có thuật toán thời gian tính đa thức để kiểm tra b đúng là ước số của n.

2. Đối với bài toán tổng con, bằng chứng xác nhận câu trả lời 'yes' đối với bộ dữ liệu $(x_1, ..., x_n)$ là vecto $c = (c_1, ..., c_n)$, trong đó $c_i = 1$ nếu x_i được chọn vào tập S và $c_i = 0$ nếu trái lại. Việc kiểm chứng xem tập S gồm các số được chọn có thỏa mãn yêu cầu đặt ra hay không, rõ ràng, có thể thực hiện sau thời gian đa thức.

3. Đối với bài toán người du lịch dạng quyết định, bằng chứng xác nhận câu trả lời 'yes' cho ma trận chi phí $C = c_{ij} : i, j = 1, ..., n$ của bài toán là dãy các thành phố trên hành trình. Việc kiểm chứng xem dãy các thành phố đã cho có phải là hành trình với chi phí không vượt quá K có thể thực hiện xong sau thời gian đa thức.

Ví dụ 3:

Đối với bài toán kiểm tra tính chính phương, để đưa ra một đáp án có thể kiểm chứng được nhanh chóng xác nhận câu trả lời 'yes' cho đầu vào n của nó, ta có thể đưa ra một số $b = \sqrt{n}$.

Lớp bài toán P, NP và co-NP

Lớp các bài toán dễ giải là các bài toán có thể giải được nhờ các thuật toán thời gian tính đa thức.

Định nghĩa: Ta gọi P là lớp các bài toán có thể giải được sau thời gian đa thức.

Bài toán về tính liên thông của đồ thị có thể giải được nhờ thuật toán với thời gian tính là $O(n^2)$, vì vậy, nó là bài toán thuộc lớp P. Bài toán cây khung nhỏ nhất giải được nhờ thuật toán Prim với thời gian $O(n^2)$, cũng thuộc vào lớp P.

Định nghĩa: Ta gọi NP là lớp các bài toán quyết định mà để xác nhận câu trả lời 'yes' của nó ta có thể đưa ra một lời giải có thể kiểm chứng được nhanh chóng.

Các bài toán trình bày trong Ví dụ 2 đều thuộc lớp NP.

Định nghĩa: Ta gọi co-NP là lớp các bài toán quyết định mà để xác nhận câu trả lời 'no' của nó ta có thể đưa ra một lời giải có thể kiểm chứng được nhanh chóng.
Các bài toán ở Ví dụ 3 đều thuộc lớp co-NP.

Lớp bài toán NP-khó và NP-đầy đủ

Ta sẽ đưa ra định nghĩa về những bài toán khó nhất trong lớp NP: bài toán NP-đầy đủ (NP-complete).
Định nghĩa:
Một bài toán quyết định A được gọi là NP-đầy đủ nếu như:

1. A là bài toán trong NP;

2. Mọi bài toán trong NP đều có thể qui dẫn về A.

Như vậy, có thể nói khái niệm về "bài toán khó nhất" trong lớp NP được xây dựng trên cơ sở phép qui dẫn. Nếu tất cả các bài toán trong NP có thể quy dẫn về một bài toán A thì A khó không kém bất cứ bài toán nào trong số chúng. Điều đáng ngạc nhiên là sự tồn tại của những bài toán có tính chất như vậy.

Khó khăn nhất là việc tìm ra được một bài toán như vậy. Bởi vì hễ chúng ta đã có một bài toán NP-đầy đủ thì để ta có thể dễ dàng chứng minh nhiều bài toán khác là NP-đầy đủ nhờ sử dụng kết quả sau đây.
Bổ đề: Giả sử bài toán A là NP-đầy đủ, bài toán B là thuộc NP, và bài toán A qui dẫn về B. Khi đó bài toán B cũng là NP-đầy đủ.
Định nghĩa: Một bài toán A được gọi là NP-khó (NP-hard) nếu như sự tồn tại thuật toán đa thức để giải nó kéo theo sự tồn tại thuật toán đa thức để giải bài toán trong NP.

Một cách không hình thức, có thể nói rằng nếu ta có thể giải được một cách hiệu quả một bài toán NP-khó cụ thể, thì ta cũng có thể giải hiệu quả bất kỳ bài toán nào trong NP bằng cách sử dụng thuật toán giải bài toán NP-khó như là một chương trình con.

Từ định nghĩa NP-khó suy ra rằng mỗi bài toán NP-đầy đủ đều là NP-khó. Tuy nhiên, như đã nêu ở trên, một bài toán là NP-khó không nhất thiết phải

là NP-đầy đủ.

Từ Bổ đề suy ra rằng để chứng minh một bài toán A nào đó là NP-khó, ta chỉ cần chỉ ra phép qui dẫn một bài toán đã biết là NP-khó về nó.

1.2.2 Độ phức tạp tính toán của bài toán

Gọi $T_A(X)$ là thời gian tính của thuật toán A đối với đầu vào X. Khi đó thời gian tính trong tình huống tồi nhất của thuật toán A đối với dữ liệu đầu vào kích thước n được định nghĩa như là:

$$T_A(n) = \max_{|X|=n} T_A(X)$$

Độ phức tạp trong tình huống tồi nhất của thuật toán P là thời gian tính trong tình huống tồi nhất của thuật toán nhanh nhất để giải nó:

$$T_p(n) = \min_{A \in A} T_A(n) = \min_{A \in A} \left(\max_{|X|=n} T_A(X) \right)$$

Trong đó Δ là tập tất cả các thuật toán giải bài toán P.

Việc đánh giá đúng độ phức tạp của bài toán là một vấn đề hết sức phức tạp. Vì vậy chúng ta quan tâm đến việc đưa ra các cận trên và cận dưới cho nó.

Nếu ta có thuật toán A với thời gian tính trong tình huống tồi nhất là $T_A(n) = O(f(n))$ thì

$$T_P(n) \leq T_A(n) \leq O(f(n))$$

Tức là ta có cận trên cho độ phức tạp của bài toán P. Thuật toán nhanh hơn sẽ cho cận trên tốt hơn.

Chúng ta còn quan tâm đến việc đánh giá cận dưới độ phức tạp của bài toán, nghĩa là quan tâm đến việc nó khó đến mức độ nào. Để chỉ ra rằng:

$$T_P(n) = \Omega(f(n))$$

Ta cần phải chỉ ra rằng:

1. Có thuật toán với thời gian tính $\Omega(f(n))$ để giải bài toán P.

2. Mọi thuật toán giải bài toán P đều đòi hỏi thời gian tính trong tình huống tệ nhất là $\Omega(f(n))$ hoặc cận dưới cho độ phức tạp tính toán của bài toán P là $\Omega(f(n))$.

1.3 Bài toán chọn vị trí một thiết bị cơ bản

Giả sử rằng một máy sao chép mới sẽ được lắp đặt trong một khu phức hợp cao ốc văn phòng. Máy có thể được thiết lập ở bất kỳ một trong năm vị trí có thể và sáu văn phòng sẽ được chỉ định sử dụng nó. Nhu cầu từ mỗi văn phòng (nghĩa là số chuyến đi dự kiến được thực hiện hàng ngày) và chi phí (được đo bằng thời gian mà một người dành để đi bộ đến và đi từ mỗi địa điểm) được đưa ra trong Bảng dưới.

Khách hàng	Vị trí					Nhu cầu
	1	2	3	4	5	
A	3	5	4	1	6	15
B	5	2	4	4	2	20
C	3	6	3	4	5	10
D	5	3	8	5	6	12
E	3	9	5	2	8	10
F	9	10	7	9	2	7

Bảng 1.1: Bảng chi phí vận chuyển và nhu cầu

Để tìm duy nhất một nơi đặt máy, ta chỉ cần thực hiện 3 bước như sau:

1. Tính toán ma trận chi phí (bằng chi phí vận chuyển nhân với nhu cầu)
2. Tính tổng mỗi cột của ma trận chi phí.
3. Chọn vị trí với chi phí ít nhất. Đó là vị trí tốt nhất nhất để đặt máy.

Khách hàng	Vị trí				
	1	2	3	4	5
A	45	75	60	15	90
B	100	40	80	80	40
C	30	60	30	40	50
D	60	36	96	60	72
E	30	90	50	20	80
F	63	70	49	63	14
Total	328	371	565	278	346

Bảng 1.2: Bảng chi phí vận chuyển nhân nhu cầu

Như ta thấy trong bảng trên, vị trí thứ 4 là vị trí tốt nhất để đặt máy.

1.4 Bài toán vị trí cơ sở không hạn chế khả năng

Bài toán vị trí cơ sở không hạn chế khả năng (Uncapacitated Facility Location Problem - UFLP) có thể được gọi với nhiều tên khác nhau, chẳng hạn như: Simple Plant Location Problem, the location of bank accounts problem, warehouse location problem, the standardization and unification problem, the problem of a nonrecoverable tools optimal system,...

Bài toán UFLP là một trong những bài toán được nghiên cứu rộng rãi nhất trong lớp các bài toán tối ưu hóa tổ hợp. Bài toán được đề xuất lần đầu tiên bởi Erlenkotter năm 1978 dưới dạng một bài toán quy hoạch tuyến tính. Neebe và Khumawala năm 1981, Karkazis và Boffey năm 1981 đề xuất bài toán với giả định mỗi cơ sở chỉ giao dịch được một sản phẩm. Năm 1987 Klincewicz và Luss là người đầu tiên nghiên cứu một mô hình vị trí cơ sở mà không bị hạn chế về số lượng sản phẩm tại mỗi cơ sở.

Bài toán UFLP được phát biểu như sau:

Xét một tập $I = 1, 2, 3, \ldots, N$ là tập các khách hàng sử dụng dịch vụ. Và một tập $J = 1, 2, 3, \ldots, M$ các cơ sở tiềm năng cung cấp sản phẩm hoặc dịch

vụ. Một cơ sở $j \in J$ có chi phí xây dựng là $C_j (C_j > 0)$. Mỗi cơ sở mở có thể cung cấp một số lượng không giới hạn hàng hóa cho mỗi khách hàng. Giá trị G_{ij} (với $i \in I$ và $j \in J$) là chi phí vận chuyển từ cơ sở j đến khách hàng i. Mục tiêu là xác định một tập hợp con S của tập hợp các địa điểm cơ sở tiềm năng J ($S \subseteq J, S \neq \varnothing$) để cung cấp cho tất cả các khách hàng sao cho tổng chi phí xây dựng và chi phí vận chuyển là nhỏ nhất.

$$F(S) = \sum_{j \in S} C_j + \sum_{i \in I} \min_{j \in S} G_{\mathrm{ij}} \rightarrow \min_{S \subseteq J}$$

Bài toán UFLP còn có thể được mô tả dưới dạng một bài toán quy hoạch tuyến tính nguyên như sau:

$$\textbf{MINIMIZE} \qquad \sum_{j \in J} C_j X_j + \sum_{j \in J} \sum_{i \in I} G_{ij} Y_{ij}$$

SUBJECT TO:

$$Y_{ij} \leqslant X_j \quad (i \in I, j \in J)$$

$$\sum_{j \in J} Y_{ij} = 1 \quad (i \in I)$$

$$Y_{ij} \in \{0, 1\} \quad (i \in I, j \in J)$$

$$X_j \in \{0, 1\} \quad (j \in J)$$

Tất cả các phương pháp tiếp cận quan trọng có liên quan đến bài toán UFLP có thể được chia thành 2 loại chính là: Thuật toán chính xác và phương pháp dựa trên metaheuristics. Các thuật toán chính xác để giải quyết bài toán UFLP chẳng hạn như nhánh cận, quy hoạch tuyến tính (linear programing), thuật toán nới lỏng Lagrăng (Lagrangean relaxation). Cách tiếp cận đối ngẫu (dual approach (DUALLOC)) và phương pháp đối ngẫu nguyên thủy (primaldual approaches).

Bài toán UFLP được chứng minh là NP-khó nên các thuật toán chính xác trên có thể không thực sự hiệu quả khi giải quyết các trường hợp số lượng cơ sở lớn. Vì vậy, đã có rất nhiều các nghiên cứu giải bài toán UFLP dựa trên phương pháp Heuristics hay Metaheuristics.

Chương 2

Bài toán P-Median

2.1 Giới thiệu bài toán

Bài toán P-Median hay còn là bài toán vị trí cơ sở có hạn chế khả năng (Capacitated Facility Location Problem – CFLP) là khái quát hóa bài toán UFLP khi mà những cơ sở bị giới hạn về số lượng sản phẩm. Mặc dù mô hình toán học của hai bài toán này không khác nhau nhiều nhưng các phương pháp giải bài toán CFLP thì thường khó hơn.

Trong bài toán CFLP, mỗi khách hàng có nhu cầu nhất định để đáp ứng và các cơ sở có hạn chế về công suất phục vụ hay hạn chế về sản phẩm cung cấp, tức là tổng nhu cầu của khách hàng được phân công một cơ sở không thể vượt quá khả năng của cơ sở đó. Cả hai bài toán UFLP và CFLP được coi là NP-khó (Garey & Johnson, 1990; Kariv & Hakimi, 1979). Các bài toán vị trí cơ sở có thể được nghiên cứu trên không gian rời rạc hoặc liên tục. Khi cơ sở có thể được đặt ở bất cứ nơi nào trong khu vực, bài toán được coi là là liên tục. Khi cơ sở có thể được đặt chỉ tại các địa điểm cụ thể, bài toán được coi là rời rạc.

Mục tiêu của bài toán CFLP là tìm ra p vị trí đặt cơ sở sao cho tổng chi phí xây dựng và chi phí vận chuyển giữa các khách hàng và cơ sở là nhỏ nhất.

2.2 Mô hình bài toán

Bài toán chọn p vị trí trên bản đồ (hay mạng lưới các vị trí đặt thiết bị) sao cho tổng chi phí được giảm thiểu. Chi phí phục vụ nhu cầu tại nút $i \in I$ được tính bởi nhân nhu cầu h_i của khách hàng $i \in I$ và khoảng cách d_{ij} giữa khách hàng $i \in I$ với vị trí được chọn $j \in J$ gần nhất với khách hàng đó.
Bài toán P-Median được mô tả chi tiết như sau:
Đầu vào

- $I = \{1, 2 \dots n\}$ là các khách hàng
- $J = \{1, 2 \dots m\}$ là các vị trí tiềm năng
- h_i là nhu cầu của khách hàng $i \in I$
- d_{ij} là khoảng cách giữa khách hàng $i \in I$ và vị trí được ứng cử để chọn $j \in J$
- P là số thiết bị cần đặt

Các biến quyết định

$$X_j = \begin{cases} 1 & \text{nếu chọn vị trí } j \in J \\ 0 & \text{nếu ngược lại} \end{cases}$$

$$Y_{ij} = \begin{cases} 1 & \text{nếu yêu cầu của khách hàng } i \in I \text{ được đáp ứng tại vị trí } j \in J \\ 0 & \text{nếu ngược lại} \end{cases}$$

Hàm mục tiêu có thể được biểu diễn như sau:

$$\textbf{MINIMIZE} \quad \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij} \tag{2.1}$$

$$\textbf{SUBJECT TO:} \quad \sum_{j \in J} Y_{ij} = 1 \quad \forall i \in I \tag{2.2}$$

$$\sum_{j \in J} X_j = P \tag{2.3}$$

$$Y_{ij} - X_j \leq 0 \quad \forall i \in I; j \in J \tag{2.4}$$

$$X_j \in \{0, 1\} \quad \forall j \in J \tag{2.5}$$

$$Y_{ij} \in \{0, 1\} \quad \forall i \in I; j \in J \tag{2.6}$$

- Hàm mục tiêu (2.1) giảm thiểu tổng khoảng cách theo trọng số nhu cầu giữa mỗi khách hàng và thiết bị gần nhất.

- Ràng buộc (2.2) yêu cầu mỗi khách hàng $i \in I$ được chỉ định cho chính xác một thiết bị $j \in J$.

- Ràng buộc (2.3) thể hiện chính xác P thiết bị sẽ được đặt.

- Ràng buộc (2.4) liên kết các biến vị trí (X_j) và các biến phân bổ (Y_{ij}). Các nhu cầu tại khách hàng $i \in I$ chỉ có thể được chỉ định cho một thiết bị tại vị trí $j \in J$ ($Y_{ij} = 1$) nếu một thiết bị được đặt ở nút $j \in J$ ($X_j = 1$).

  Ràng buộc này đảm bảo rằng nếu một cơ sở ($j \in J$) không được chọn (tức là $X_j = 0$), thì chắc chắn không có nút $i \in I$ nào có thể được gán tới cơ sở đó (tức là $Y_{ij} = 0$ với $\forall i \in I$). Còn khi ta chọn một cơ sở $j \in J$ (tức là $X_j = 1$), nếu nút $i \in I$ được đáp ứng tại cơ sở j đó thì $Y_{ij} = 1$ còn nếu ngược lại thì $Y_{ij} = 0$.

- Các ràng buộc (2.5) và (2.6) là các điều kiện tích phân tiêu chuẩn.

Ràng buộc (2.4) là một ràng buộc mạnh của liên kết các biến vị trí và phân bổ. Một phiên bản yếu hơn là:

$$\sum_{i \in I} Y_{ij} - |I| X_j \leq 0 \quad \forall j \in J \tag{2.7}$$

trong đó $|I|$ là số khách hàng yêu cầu.
Sử dụng dạng ràng buộc mạnh hơn (2.4) đã được chứng minh là có lợi về mặt tính toán.

Hàm mục tiêu bài toán P-Median được đưa ra ở trên giả định rằng các thiết bị được đặt trên các nút của sơ đồ mạng cơ sở. Hakimi (1965) đã chỉ ra rằng đối với bài toán P-Median, ít nhất một giải pháp tối ưu bao gồm việc chọn vị trí các thiết bị P trên các nút của mạng cơ sở.

Để chứng minh rằng điều này là đúng, ta xem một giải pháp trong đó, ít nhất một thiết bị nằm trên liên kết (i, j) cách nút $i \in I$ một khoảng a với $(0 < a < d_{ij})$. Đặt H_i biểu thị tổng nhu cầu được phục vụ bởi thiết bị này và đi vào liên kết (i, j) với $i \in I$. Định nghĩa H_j tương tự (với $j \in J$). Giả sử $H_i \geq H_j$, bằng cách di chuyển thiết bị từ vị trí của nó a đơn vị, từ nút $i \in I$ đến nút $i \in I$ của chính nó (không làm thay đổi bất kỳ phân bổ nhu cầu nào), ta thay đổi hàm mục tiêu bằng $(H_j - H_i)a$. Vì $H_i \geq H_j$ nên đại lượng này không dương. Vì vậy, thực hiện thay đổi này tại vị trí của thiết bị sẽ không làm suy giảm giải pháp. Điều này đủ để chứng minh rằng ít nhất một giải pháp tối ưu chỉ bao gồm việc chọn vị trí trên các nút của mạng.

Tính chất mà có ít nhất một giải pháp tối ưu chỉ bao gồm việc xác định vị trí trên các nút đã được mở rộng cho một số biến thể của bài toán P-Median. Điều kiện then chốt cần thiết cho các phần mở rộng này là hàm mục tiêu phải lõm. Handler và Mirchandani (1979), Mirchandani (1990), và Mirchandani và Odoni (1979), trong số những người khác, thảo luận về các phần mở rộng của tính chất này. Tính chất này giới hạn số lượng các giải pháp thay thế phải được kiểm tra để tìm ra giải pháp cho

$$\begin{pmatrix} N \\ P \end{pmatrix} = \frac{N!}{P!(N-P)!}$$

trong đó N là số nút và P là số thiết bị được đặt. Mặc dù số này là $O\left(N^P\right)$ (với $P \ll N$) và do đó là đa thức trong N đối với một giá trị nhất định là P, nhưng số lượng kết hợp có thể rất lớn.

Ví dụ: Nếu $N = 20$ và $P = 5$, $\begin{pmatrix} 20 \\ 5 \end{pmatrix} = 15,504$. Đối với $N = 50$ và $P = 10$, một vấn đề hoàn toàn không lớn theo hầu hết các tiêu chuẩn, $\begin{pmatrix} 50 \\ 10 \end{pmatrix} > 10^{10}$. Nếu bạn có thể đánh giá một triệu kết hợp mỗi giây, thì sẽ mất gần 3h để liệt kê mọi giải pháp khả thi. Đây là một việc hết sức khó khăn vì việc đánh giá từng kết hợp liên quan đến việc tìm ra cở sở (thiết bị) gần nhất với mỗi trong số 40 nút nhu cầu (khách hàng) mà tại đó không đặt cở sở (thiết bị). Như vậy, để đánh giá mỗi giải pháp sẽ cần ít nhất 400 phép so sánh, 40 phép nhân và 40 phép cộng. Cuối cùng, nếu $N = 100$ và $P = 15$, thì sẽ mất hơn

tám thiên niên kỷ để liệt kê mọi giải pháp khả thi ngay cả khi chúng ta có thể làm như vậy với tốc độ một triệu mỗi giây. Hầu hết chúng ta không muốn chờ đợi giải pháp lâu như vậy!

Như đã chỉ ra ở trên, đối với P cố định, bài toán P-Median về mặt kỹ thuật có thể được giải quyết trong một khoảng thời gian đa thức với số lượng nút. Tuy nhiên, đối với các giá trị vừa phải của N và P, số lượng các giải pháp khả thi phải được liệt kê trở nên rất lớn. Hơn nữa, đối với biến P, bài toán vẫn là NP-đầy đủ (Garey và Johnson, 1979). Để biết tại sao lại như vậy, hãy lưu ý rằng thời gian cần thiết để giải tất cả các bài toán P-Median bằng phép liệt kê cho $P = 1$ đến $P = N$ cho bất kỳ giá trị nào của N là

$$\sum_{j=1}^{N} \left(\begin{array}{c} N \\ j \end{array} \right) = 2^N - 1 \approx O\left(2^N\right)$$

là cấp số nhân trong N.

Nói tóm lại, ta sẽ cần tìm các thuật toán Heuristic hiệu quả nếu chúng ta muốn giải các bài toán có kích thước thực tế trong một khoảng thời gian hợp lý.

2.3 Một số thuật toán Heuristic giải bài toán P-Median

2.3.1 Mô hình thuật toán

Ở phần này em trình bày 3 thuật toán Heuristic sau: thuật toán cận thị, thuật toán trao đổi và thuật toán tìm kiếm lân cận.

Các thuật toán Heuristics này được chia thành hai loại thuật toán Heuristic chính (Golden et al., 1980): thuật toán xây dựng và thuật toán cải tiến. Thuật toán cận thị là một thuật toán xây dựng, trong đó chúng ta cố gắng xây dựng một giải pháp tốt từ đầu, còn thuật toán trao đổi và tìm kiếm vùng lân cận đều là các thuật toán cải tiến.

Với bài toán chọn vị trí cho một thiết bị duy nhất trên mạng, chúng ta có thể dễ dàng tìm thấy vị trí tối ưu bằng cách liệt kê tất cả các vị trí có thể và

chọn vị trí tốt nhất. Cụ thể, vì chúng ta biết rằng có ít nhất một giải pháp tối ưu cho bất kỳ bài toán P-median nào chỉ bao gồm việc xác định vị trí trên các nút yêu cầu bằng cách ta có thể đánh giá hàm mục tiêu cho bài toán 1-median, ta tính giá trị hàm mục tiêu $Z_j = \sum_{i \in I} h_i d_{ij}$ cho mỗi nút yêu cầu nếu chúng ta chọn tại vị trí nút yêu cầu $j \in J$ đó. Sau đó, chúng ta sẽ chọn vị trí có giá trị Z_j nhỏ nhất. Khi ta chỉ muốn xác định vị trí của một thiết bị duy nhất, thì phương pháp này sẽ đưa ra giải pháp tối ưu.

Giả sử bây giờ chúng ta có vị trí của $P-1$ cơ sở hay thiết bị. Đặt $\mathbf{X}_{P-1}$ là tập hợp vị trí của $P-1$ thiết bị này. Ngoài ra, đặt $d\left(i, \mathbf{X}_{P-1}\right)$ là khoảng cách ngắn nhất giữa nút cầu $i \in I$ và nút gần nhất trong tập hợp $\mathbf{X}_{P-1}$. Tương tự, chúng ta đặt $d\left(i, \{j \cup \mathbf{X}_{P-1}\}\right)$ là khoảng cách ngắn nhất giữa nút cầu $i \in I$ và nút gần nhất trong tập hợp $\mathbf{X}_{P-1}$ được bổ sung bởi vị trí cơ sở $j \in J$. Vị trí tốt nhất để xác định vị trí để đặt một thiết bị hay cơ sở mới, với điều kiện là $P-1$ cơ sở đầu tiên được đặt tại các vị trí đã cho trong tập hợp $\mathbf{X}_{P-1}$ nằm tại vị trí $j \in J$ mà Minimizes $Z_j = \sum_{i \in I} h_i d\left(i, \{j \cup \mathbf{X}_{P-1}\}\right)$. Cách tiếp cận này dẫn đến thuật toán cận thị để xây dựng giải pháp cho bài toán P-Median. Về mặt hình thức, chúng ta có thể phát biểu thuật toán như sau:

Thuật toán cận thị cho bài toán P-Median:

Bước 1: Khởi tạo $k = 0$ (k sẽ đếm số thiết bị mà chúng ta đã xác định được cho đến nay) và $\mathbf{X}_k = \varnothing$, tập hợp rỗng ($\mathbf{X}_k$ sẽ cung cấp vị trí của các thiết bị k mà chúng ta đã đặt ở mỗi giai đoạn của thuật toán).

Bước 2: Tăng k, bộ đếm số lượng thiết bị được đặt.

Bước 3: Tính $Z_j^k = \sum_{i \in I} h_i d\left(i, \{j \cup \mathbf{X}_{k-1}\}\right)$ cho mỗi nút $j \in J$, không có trong tập hợp $\mathbf{X}_{k-1}$.

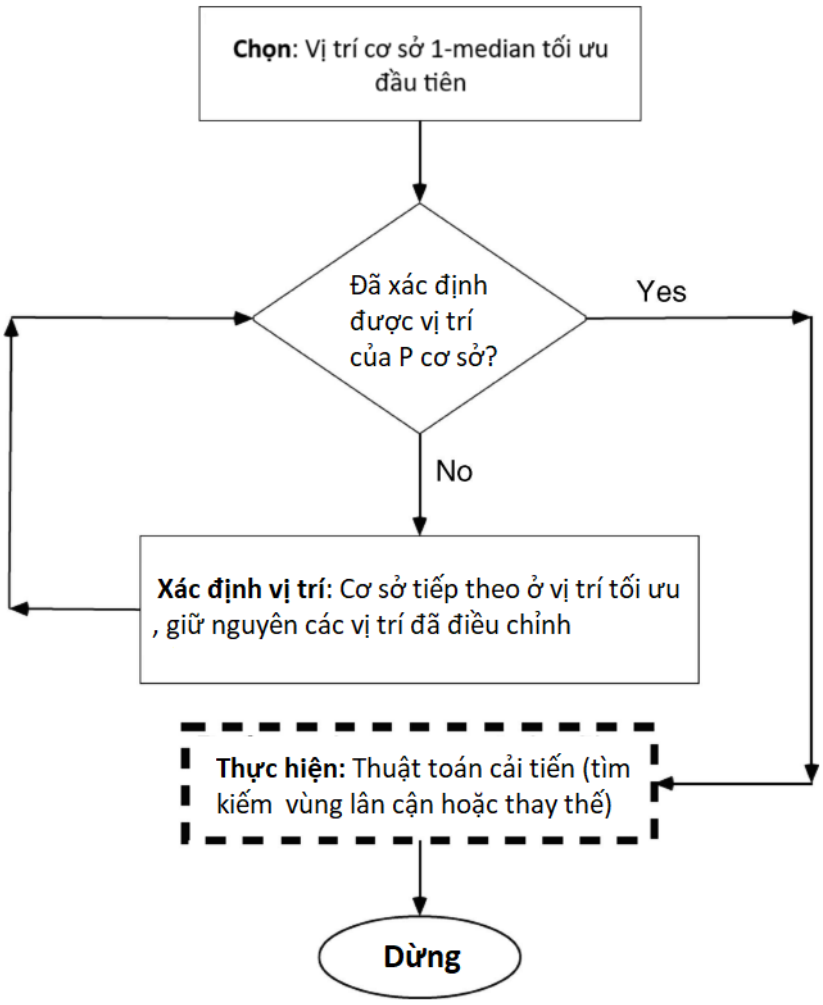
Lưu ý rằng Z_j^k cung cấp giá trị của hàm mục tiêu P-median nếu chúng ta xác định thiết bị thứ k tại nút $j \in J$, biết rằng $k-1$ thiết bị đầu tiên nằm ở các vị trí đã cho trong tập hợp $\mathbf{X}_{k-1}$ (và nút $j \in J$ không thuộc tập hợp đó).

Bước 4: Tìm nút $j^*(k)$ có Z_j^k nhỏ nhất, nghĩa là $j^*(k) = argmin_{j \in J} Z_j^k$. Lưu ý rằng $j^*(k)$ là vị trí tốt nhất cho thiết bị thứ k, với vị trí của $k-1$ thiết bị đầu tiên. Thêm nút $j^*(k)$ vào tập $\mathbf{X}_{k-1}$ để được tập $\mathbf{X}_k$; nghĩa là

đặt $\mathbf{X}_k = j^*(k) \cup \mathbf{X}_{k-1}$.

Bước 5: Nếu $k = P$ (tức là ta đã định vị được thiết bị P) thì dừng; tập $\mathbf{X}_p$ là giải pháp cho thuật toán cận thị. Nếu $k < P$, chuyển về bước 2.

Dưới đây là một lưu đồ đơn giản của thuật toán Heuristic này.



Hình 2.1: Thuật toán cận thị

Thuật toán cận thị sẽ cho ta một phương án tốt từ đầu. Một trong các thuật toán cải tiến như tìm kiếm vùng lân cận hay trao đổi sẽ được trình bày bên dưới có thể được áp dụng với giải pháp ban đầu thu được bằng cách sử dụng thuật toán cận thị.

Giải pháp thu được bằng cách sử dụng thuật toán cận thị sẽ không nhất thiết là giải pháp tối ưu, vì tại mỗi lần sử dụng thuật toán, chúng ta đang giữ cố định vị trí của $k-1$ cơ sở cố định nên chúng ta sẽ sử dụng các thuật

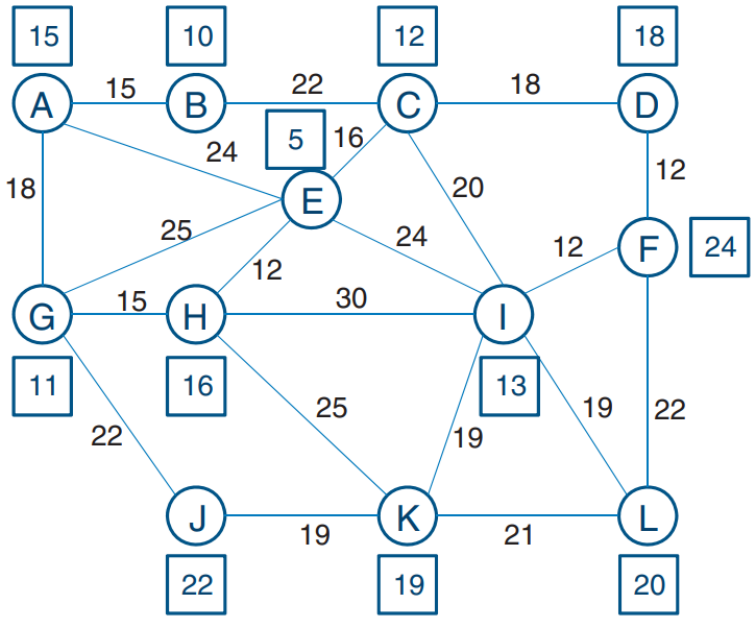
toán cải thiện để được giải pháp tốt hơn.

Mặc dù giải pháp thu được bằng cách sử dụng thuật toán cận thị có thể không tối ưu, nhưng thuật toán này vẫn rất hay dùng vì một số lý do:

1. Đầu tiên, nó dễ hiểu và dễ sử dụng.

2. Thứ hai, khi ta được cung cấp vị trí của một số cơ sở không thể di chuyển. Sau đó, chúng ta được yêu cầu tìm vị trí của một vài (thường chỉ một hoặc hai) cơ sở mới. Nếu chúng ta chỉ được yêu cầu xác định vị trí của một cơ sở bổ sung và các cơ sở hiện tại không thể bị thay thế, thì phương pháp này rõ ràng sẽ là tối ưu.

Để minh họa cho thuật toán cận thị, ta sẽ thực hiện một ví dụ có sơ đồ mạng cơ sở (thiết bị) được hiển thị trong hình dưới đây:



Hình 2.2: Ví dụ sơ đồ bài toán P-Median mẫu

Trong Hình 2.2, các ô tròn là các nút khách hàng $i \in I$, giá trị trong ô vuông bên cạnh nút khách hàng là nhu cầu h_i của khách hàng i.

	A	B	C	D	E	F	G	H	I	J	K	L
A	0	15	37	55	24	60	18	33	48	40	58	67
B	15	0	22	40	38	52	33	48	42	55	61	61
C	37	22	0	18	16	30	41	28	20	58	39	39
D	55	40	18	0	34	12	59	46	24	62	43	34
E	24	38	16	34	0	36	25	12	24	47	37	43
F	60	52	30	12	36	0	57	42	12	50	31	22
G	18	33	41	59	25	57	0	15	45	22	40	61
H	33	48	28	46	12	42	15	0	30	37	25	46
I	48	42	20	24	24	12	45	30	0	38	19	19
J	40	55	58	62	47	50	22	37	38	0	19	40
K	58	61	39	43	37	31	40	25	19	19	0	21
L	67	61	39	34	43	22	61	46	19	40	21	0

Bảng 2.1: Ma trận khoảng cách của sơ đồ Hình 2.2

Bảng trên đưa ra ma trận khoảng cách của sơ đồ Hình 2.2.

	A	B	C	D	E	F	G	H	I	J	K	L
A	0	225	555	825	360	900	270	495	720	600	870	1005
B	150	0	220	400	380	520	330	480	420	550	610	610
C	444	264	0	216	192	360	492	336	240	696	468	468
D	990	720	324	0	612	216	1062	828	432	1116	774	612
E	120	190	80	170	0	180	125	60	120	235	185	215
F	1440	1248	720	288	864	0	1368	1008	288	1200	744	528
G	198	363	451	649	275	627	0	165	495	242	440	671
H	528	768	448	736	192	672	240	0	480	592	400	736
I	624	546	260	312	312	156	585	390	0	494	247	247
J	880	1210	1276	1364	1034	1100	484	814	836	0	418	880
K	1102	1159	741	817	703	589	760	475	361	361	0	399
L	1340	1220	780	680	860	440	1220	920	380	800	420	0
Total	**7816**	**7913**	**5855**	**6457**	**5784**	**5760**	**6936**	**5971**	**4772**	**6886**	**5576**	**6371**

Bảng 2.2: Ma trận chi phí $h_i d_{ij}$

Bảng trên đưa ra các giá trị $h_i d_{ij}$.

Bằng cách tính tổng các giá trị $h_i d_{ij}$ trong mỗi cột của bảng trên, ta thu

được các giá trị của Z_j^1. Giá trị Z_j^1 nhỏ nhất tương ứng với $j = I$, với giá trị là 4772. Do đó, tổng khoảng cách nhân trọng số nhu cầu $h_i d_{ij}$ tối ưu nếu chúng ta chỉ xác định 1-trung vị duy nhất cho mạng của Hình 2.2 là 4772 suy ra khoảng cách trung bình là 4772/185 hay 25,795 (có 185 yêu cầu trong mạng sơ đồ).

Để xác định vị trí trung vị thứ hai, chúng ta cần tính $h_i \cdot \min\{d(i, I); d(i, j\}$ cho mỗi cặp khách hàng/ứng viên (i, j).
Bảng dưới cho thấy kết quả của tính toán này:

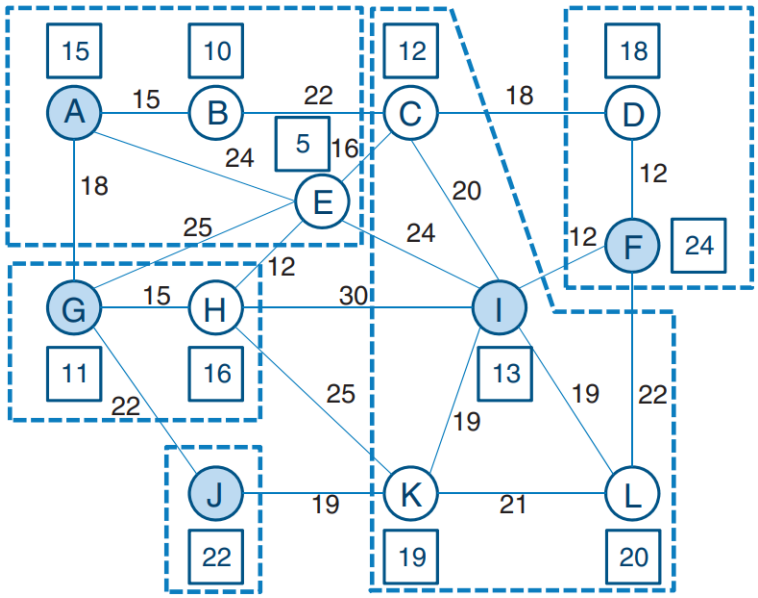
	A	B	C	D	E	F	G	H	I	J	K	L
A	0	225	555	720	360	720	270	495	720	600	720	720
B	150	0	220	400	380	420	330	420	420	420	420	420
C	240	240	0	216	192	240	240	240	240	240	240	240
D	432	432	324	0	432	216	432	432	432	432	432	432
E	120	120	80	120	0	120	120	60	120	120	120	120
F	288	288	288	288	288	0	288	288	288	288	288	288
G	198	363	451	495	275	495	0	165	495	242	440	495
H	480	480	448	480	192	480	240	0	480	480	400	480
I	0	0	0	0	0	0	0	0	0	0	0	0
J	836	836	836	836	836	836	484	814	836	0	418	836
K	361	361	361	361	361	361	361	361	361	361	0	361
L	380	380	380	380	380	380	380	380	380	380	380	0
Total	**3485**	**3725**	**3943**	**4296**	**3696**	**4268**	**3145**	**3655**	**4772**	**3563**	**3858**	**4392**

Bảng 2.3: Bảng tính $h_i \cdot \min\{d(i, I); d(i, j\}$ cho mỗi cặp (i, j)

Tổng số cột tương ứng với Z_j^2. Bây giờ, nó là tốt nhất khi thêm một cơ sở tại nút G. Tổng khoảng cách nhân trọng số yêu cầu $h_i d_{ij}$ tại đó là 3145, khoảng cách trung bình là 17.000. Để thêm cơ sở thứ ba, ta sẽ tính $h_i \cdot \min\{d(i, G); d(i, I); d(i, j)\}$ cho mỗi cặp khách hàng/ứng viên (i, j), tìm tổng cột tương ứng với Z_j^3, tìm cột có tổng nhỏ nhất và định vị tại nút tương ứng. Tiến hành theo cách này, chúng ta thu được kết quả như trong bảng dưới cho 5-trung vị (5-median) cận thị đầu tiên.

Số trung vị	Vị trí	Tổng nhu cầu-khoảng cách trọng	Khoảng cách trung bình
1	I	4772	25795
2	G	3145	17000
3	F	2641	14276
4	J	2157	11659
5	A	1707	9.227

Bảng 2.4: Kết quả 5 trung vị cận thị đầu tiên



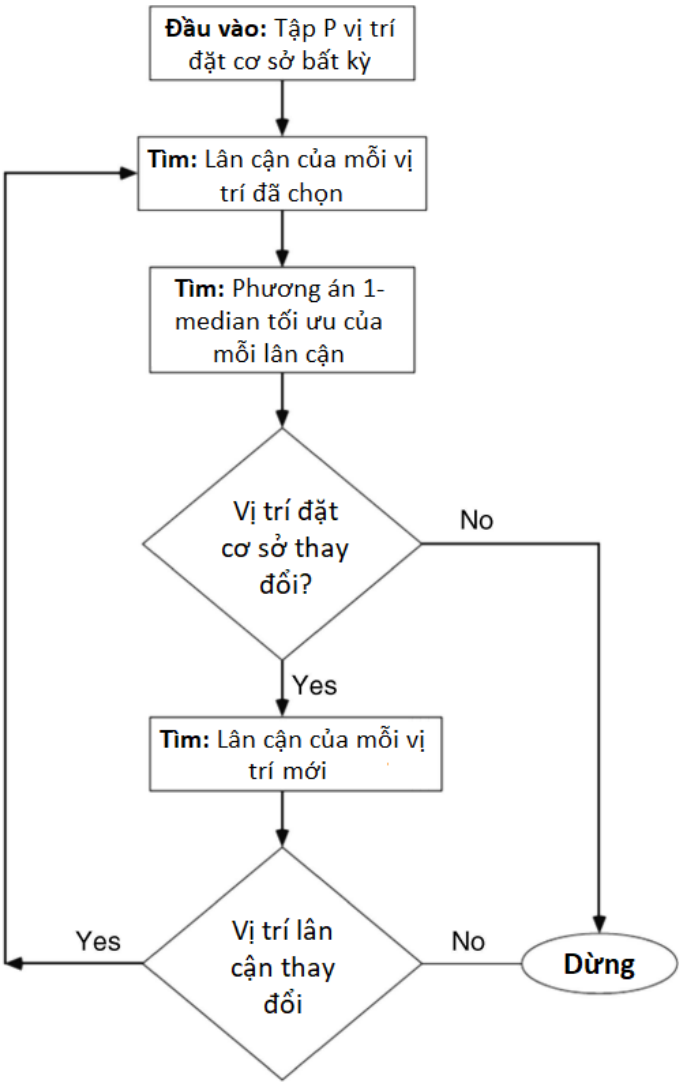
Hình 2.3: Các vùng lân cận được liên kết với thuật toán 5-median cận thị

Hình 2.3 hiển thị các lân cận được liên kết với 5 trung vị cận thị của mạng được hiển thị trong Hình 2.2. Lưu ý rằng khi gán các nút nhu cầu cho các cơ sở, chúng ta có thể phá vỡ các ràng buộc tùy ý. Do đó, nút yêu cầu E được gán cho vùng lân cận của cơ sở tại nút A mặc dù nó cũng có thể được gán cho vùng lân cận của cơ sở nằm tại nút I.

Trong mỗi vùng lân cận, chúng ta muốn vị trí trung vị sẽ được chọn một cách tối ưu. Nói cách khác, chúng ta hy vọng rằng cơ sở phục vụ mỗi vùng lân cận sẽ được đặt tại vị trí 1-median tối ưu cho các nút trong vùng lân cận. Vì việc tìm 1-median tối ưu có thể được thực hiện đơn giản bằng phương pháp liệt kê (total emuration), nên chúng ta có thể dễ dàng đảm bảo rằng điều kiện này được thỏa mãn. Điều này dẫn đến thuật toán cải tiến tìm kiếm

vùng lân cận.

Hình dưới là lưu đồ của thuật toán tìm kiếm lân cận. [Maranzana (1964) là người đầu tiên đề xuất thuật toán này.]



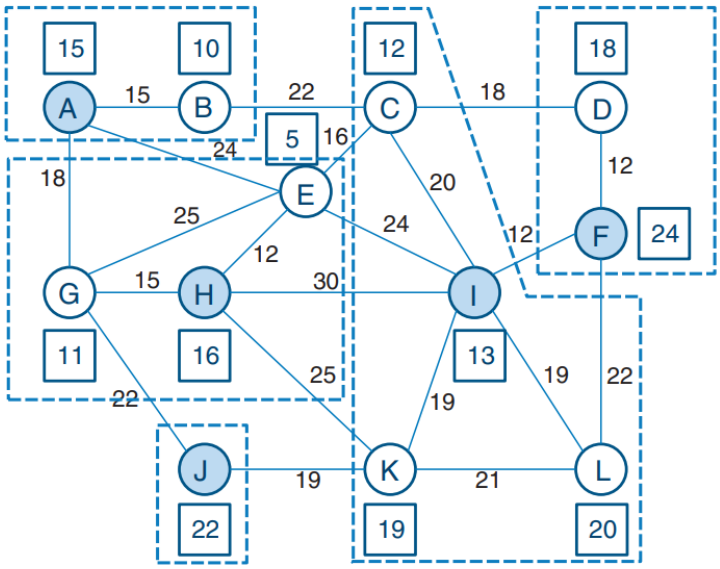
Hình 2.4: Lưu đồ thuật toán tìm kiếm lân cận

Thuật toán tìm kiếm vùng lân cận có thể bắt đầu với bất kỳ tập hợp các vị trí P cơ sở nào.

Ví dụ: Chúng ta có thể bắt đầu với P vị trí được xác định bởi thuật toán tìm kiếm cận thị. Đối với mỗi địa điểm của cơ sở, thuật toán xác định tập hợp các nút nhu cầu tạo thành vùng lân cận xung quanh địa điểm của cơ sở. Trong mỗi vùng lân cận, 1-median tối ưu được tìm thấy. Nếu bất kỳ vị trí nào đã thay đổi, thuật toán sẽ phân bổ lại nhu cầu cho cơ sở gần nhất và hình thành các vùng lân cận mới. Nếu bất kỳ vùng lân cận nào thay đổi,

thuật toán sẽ tìm lại 1-median trong mỗi vùng lân cận,. . . .

Quay trở lại các vùng lân cận được xác định cho giải pháp 5-trung vị cận thị được hiển thị trong Hình 2.3, vùng lân cận duy nhất mà vị trí của 1-median là dưới mức tối ưu là vùng được liên kết với nút G. Cụ thể, chúng ta di chuyển cơ sở từ nút G sang nút H. Điều này làm giảm tổng khoảng cách theo trọng số nhu cầu từ 1707 xuống 1632 (giảm khoảng cách trung bình từ 9,227 xuống 8,822 đơn vị). Bây giờ, chúng ta cố gắng chỉ định lại các nhu cầu cho nhóm cơ sở mới và nhận thấy rằng nút E bây giờ sẽ được chỉ định cho cơ sở tại nút H. Điều này càng làm giảm tổng khoảng cách trọng số theo nhu cầu xuống còn 1572 và khoảng cách trung bình xuống còn 8,497.
Sự thay đổi các vùng lân cận kết quả được hiển thị trong Hình 2.5 ở dưới:



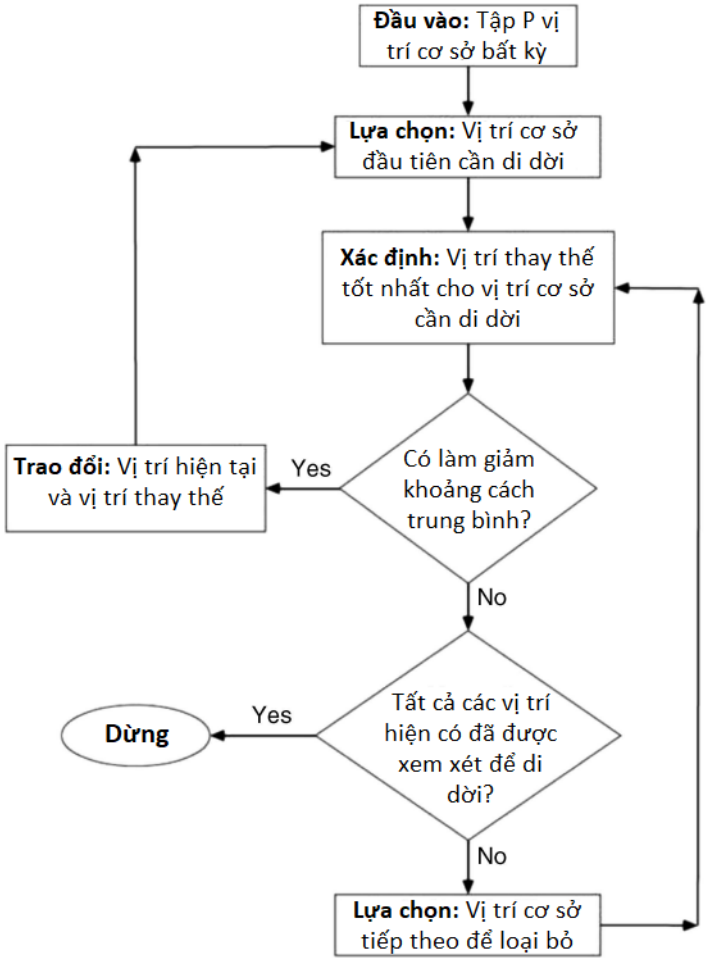
Hình 2.5: Các vùng lân cận mới sau khi thay thế

Trong mỗi vùng lân cận, chúng ta lại tìm thấy 1-trung vị tối ưu. Bây giờ các vị trí 1-trung vị không thay đổi và do đó thuật toán dừng lại.
Một trong những hạn chế liên quan đến thuật toán tìm kiếm lân cận là khi đánh giá tác động của bất kỳ quyết định thay thế nào, chỉ các tác động đối với các nút đó trong lân cận được xem xét.

Tiềm năng lợi ích đối với các nút nằm ngoài khu vực lân cận không được xem xét khi quyết định liệu việc di chuyển có nên được thực hiện hay không. Ví dụ, việc giảm khoảng cách liên quan đến việc phục vụ nút E (do cơ sở

được chỉ định lại từ cơ sở tại nút A đến cơ sở được thay thế tại nút H) đã không được xem xét khi quyết định có di chuyển cơ sở từ G sang H hay không. Do hạn chế này, một số thay thế được có lợi theo nghĩa toàn cầu (chứ không phải địa phương hoặc vùng lân cận) không được xem xét.

Điều này khiến ta sử dụng thuật toán trao đổi như một quy trình cải tiến thay thế.



Hình 2.6: Lưu đồ thuật toán trao đổi hay thay thế

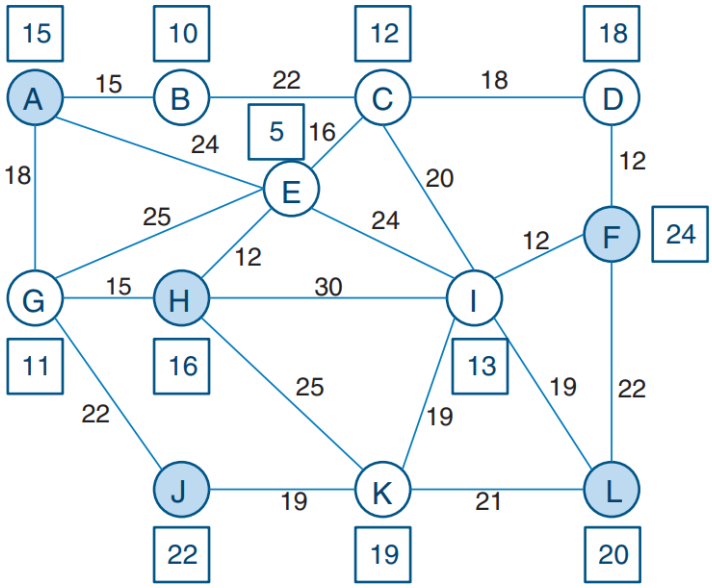
Hình 2.6 là lưu đồ của thuật toán trao đổi. Thuật toán thể hiện trong lưu đồ trên chúng ta tìm nút tốt nhất để đưa vào giải pháp thay cho nút đang được xem xét để loại bỏ. [Teitz và Bart (1968) là một trong số những người đầu tiên đề xuất phương pháp trao đổi heuristic cho bài toán P-median. Densham và Rushton (1992) đã đề xuất một số cải tiến cho thuật toán cũng như các cấu trúc dữ liệu cụ thể được thiết kế để tăng tốc thuật toán khi áp dụng cho các sơ đồ mạng lớn.]

Để minh họa thuật toán trao đổi, hãy xem xét giải pháp cho bài toán 5-trung vị cho Hình 2.5 thu được bằng cách sử dụng thuật toán heuristic tìm kiếm vùng lân cận.

Vị trí bị thay thế	Vị trí thay thế	Tổng nhu cầu-khoảng cách trọng	Số thay đổi	Khoảng cách trung bình
A	B	1647	75	8903
F	D	1620	48	8757
H	E	1689	117	9130
I	L	1444	-128	7805
J	K	1629	57	8805

Bảng 2.5: Các trao đổi có thể từ thuật toán tìm kiếm vùng lân cận

Bảng trên đưa ra nút thay thế tốt nhất cho mỗi nút trong 5 nút giải pháp. Giải pháp được minh họa trong Hình 2.5 liên quan đến việc xác định vị trí tại các nút A, F, H, I và J để có tổng khoảng cách theo trọng số nhu cầu là 1572. Nếu chúng ta xem xét việc loại bỏ nút A, thì nút tốt nhất để chèn vào thay thế nút A là nút B. Điều này làm tăng tổng khoảng cách trọng số theo yêu cầu thêm 75 đơn vị lên 1647. Tương tự, loại bỏ nút F và thay thế nó bằng nút thay thế tốt nhất của nó, nút D, làm tăng hàm mục tiêu thêm 48 đơn vị lên 1620. Việc loại bỏ nút H hoặc nút J cũng làm giảm hàm mục tiêu. Tuy nhiên, nếu chúng ta loại bỏ nút I và thay thế nó bằng nút thay thế tốt nhất của nó, nút L, thì hàm mục tiêu sẽ giảm 128 đơn vị xuống còn 1444 (hoặc khoảng cách trung bình là 7,805). Chúng ta thực hiện thay đổi này để có được một giải pháp mới với các cơ sở đặt tại các nút A, F, H, J và L. Giải pháp này được thể hiện trong Hình 2.7 dưới đây.



Hình 2.7: Giải pháp sau khi dùng thuật toán trao đổi

Lưu ý rằng theo lưu đồ thể hiện trong Hình 2.6, chúng ta sẽ tính toán cải thiện tốt nhất có thể có liên quan đến việc loại bỏ nút A, sau đó loại bỏ nút F, sau đó loại bỏ nút H, và sau đó loại bỏ nút I. Lúc này, ta sẽ tìm thấy một trao đổi giúp cải thiện giải pháp, ta sẽ thực hiện trao đổi đó mà không cần đánh giá tác động có thể có của việc loại bỏ nút J và thay thế nó bằng nút thay thế tốt nhất của nó.

Ngoài ra, lưu ý rằng ta không thể có được giải pháp cải tiến này bằng cách sử dụng kỹ thuật tìm kiếm vùng lân cận vì để có được giải pháp cải tiến đòi hỏi phải đồng thời di chuyển cơ sở và phân bổ lại nhu cầu đến các địa điểm mới. Cải thiện không thể đạt được bằng cách di chuyển cơ sở hoặc phân bổ lại nhu cầu đến các địa điểm cơ sở khác nhau.

Sau khi thực hiện trao đổi, chúng ta sẽ quay lại hộp thứ hai trong lưu đồ của Hình 2.6.

Vị trí bị thay thế	Vị trí thay thế	Tổng nhu cầu-khoảng cách trọng	Số thay đổi	Khoảng cách trung bình
A	B	1447	3	7822
F	D	1487	43	8038
H	E	1465	21	7919
I	K	1501	57	8114
J	K	1503	59	8124

Bảng 2.6: Các trao đổi từ các giải pháp đã cải thiện từ Bảng 2.5

Bảng trên cho thấy kết quả của việc loại bỏ từng vị trí cơ sở trong giải pháp mới và thay thế từng vị trí bằng nút thay thế tốt nhất của nó. Mỗi trao đổi ứng cử viên như vậy làm suy giảm giải pháp. Do đó, chúng ta giữ lại giải pháp được hiển thị trong Hình 2.7 và thuật toán sẽ dừng lại.

Các giải pháp mà ta đã thử trong Bảng 2.6 không thành công trong việc tìm ra một giải pháp có tổng khoảng cách nhân nhu cầu $h_i d_{ij}$ nhỏ hơn so với giải pháp được tìm thấy trong Bảng 2.5. Tuy nhiên, những thay thế đã thử trong Bảng 2.6 đã xác định được 5 giải pháp, tỷ lệ lệch so với giải pháp tối ưu là dưới 4,1% của giải pháp tối ưu. Ngoài ra, ta có thể tìm thấy một giải pháp chỉ kém hơn 0,21% so với giải pháp tối ưu. Để thu được giải pháp này, chúng ta cần di chuyển cơ sở từ nút A đến nút B và sau đó chuyển phân bổ nhu cầu tại nút C từ cơ sở tại nút H sang cơ sở mới tại nút B và gán nhu cầu tại nút A cho cơ sở tại nút B. Một giải pháp khác (di chuyển cơ sở từ H sang E) giảm chất lượng giải pháp chỉ dưới 1,5%.

2.3.2 Độ phức tạp của thuật toán

Về độ phức tạp tính toán của từng thuật toán Heuristic mà chúng ta đã nêu ở trên.

Đối với thuật toán cận thị, khi chúng ta thêm cơ sở thứ k, đối với mỗi cặp nút yêu cầu/vị trí ứng viên, chúng ta cần thực hiện một so sánh (để xác định xem vị trí ứng viên có gần hơn cơ sở gần nhất trong số $k-1$ cơ sở đầu tiên đã được định vị hay không) và một phép nhân. Như vậy, đối với mỗi cơ sở mới, chúng ta cần thực hiện các phép toán cơ bản $O(n^2)$. (Ngoài ra, chúng ta cần thêm tất cả các sản phẩm cho từng vị trí ứng cử viên và để tìm số

tiền tối thiểu trên tất cả địa điểm ứng cử viên mà tại đó một cơ sở chưa được đặt. Điều này không thay đổi thứ tự độ lớn của số lượng hoạt động cơ bản cho mỗi cơ sở mới). Như vậy, để xác định vị trí cơ sở P, chúng ta cần độ phức tạp $O(Pn^2)$ cho thuật toán cận thị.

Đối với hai thuật toán cải tiến, gần như không thể dự đoán số lần thuật toán sẽ cần được thực hiện bắt đầu với bất kỳ giải pháp ban đầu nào. Tuy nhiên, chúng ta có thể xác định số lượng tính toán cần thiết cho mỗi lần lặp chính của thuật toán.

Đối với thuật toán tìm kiếm lân cận, về bản chất, chúng ta phải giải bài toán 1-trung vị trên mỗi P mạng hoặc lân cận nhỏ hơn. Tuy nhiên, vì mỗi nút nằm trong chính xác một vùng lân cận, nên chúng ta cần các phép toán $O(n^2)$ cho mỗi bước lặp chính của thuật toán tìm kiếm vùng lân cận.

Đối với thuật toán trao đổi, chúng ta phải xem xét mọi cặp cơ sở hiện có/vị trí thay thế có thể. Có P cơ sở hiện có và $n - P$ vị trí thay thế, hoặc $P(n-P)$ cặp như vậy được xem xét. Với mỗi cặp như vậy, chúng ta cần kiểm tra từng nút nhu cầu để xác định xem cơ sở mới có gần nút nhu cầu hơn cơ sở ban đầu được chỉ định để phục vụ nút nhu cầu hay không. [Nếu nút đang được kiểm tra để loại bỏ được chỉ định phục vụ nút yêu cầu, chúng ta cần xác định xem cơ sở mới có gần nút yêu cầu hơn cơ sở gần thứ hai trong số các cơ sở ban đầu hay không. Trong trường hợp không có cấu trúc dữ liệu phức tạp theo dõi (và cập nhật giữa các lần lặp) tốt thứ hai, tốt thứ ba,. . . , cơ sở tốt nhất thứ P cho mỗi nút yêu cầu, việc tìm nút gần thứ hai trong số các nút ban đầu sẽ yêu cầu so sánh $O(nP)$ ở đầu mỗi lần lặp lại chính của thuật toán trao đổi. Vì điều này có thể được thực hiện một lần tại bắt đầu mỗi lần lặp và vì thời gian cần thiết ít hơn thời gian cần thiết cho mỗi lần lặp chính, nên chúng ta có thể bỏ qua thời gian này khi ước tính độ phức tạp của thuật toán trao đổi.] Như vậy, mỗi lần lặp chính của thuật toán trao đổi yêu cầu $O(nP(n - P))$ các phép toán. Nếu P xấp xỉ $n/2$, điều này trở thành $O(n^3)$.

Chương 3

Bài toán thử nghiệm và đánh giá kết quả

3.1 Giới thiệu bài toán

Áp dụng mô hình bài toán P-median vào thực hiện bài toán chọn vị trí cây bán nước tự động tại Đại học Bách Khoa Hà Nội.
Các bước thực hiện của bài toán:

1. Xác định mục tiêu của bài toán:

   Trong bài toán này, mục tiêu của chúng ta là chọn vị trí tối ưu để đặt cây bán nước tự động sao cho nó có thể phục vụ được nhiều sinh viên nhất trong đại học Bách Khoa Hà Nội.

2. Thu thập dữ liệu:

   Thu thập các thông tin về lượng sinh viên và nhân viên, địa hình, vị trí của các tòa nhà, các khu vực sử dụng nhiều, các khu vực đông đúc, các điểm hẹn hò và nghỉ ngơi, và các thông tin liên quan khác.

3. Xác định các ràng buộc:

   Xác định các tiêu chí và ràng buộc cho việc chọn vị trí máy bán nước tự động tại đại học Bách Khoa Hà Nội. Các tiêu chí có thể bao gồm: số lượng sinh viên và nhân viên, khoảng cách từ vị trí đến các khu vực sử dụng nhiều, tiện lợi cho việc truy cập và di chuyển, không ảnh hưởng

đến hoạt động của các khu vực khác trong trường. Ràng buộc có thể bao gồm: khu vực phải được phép đặt máy bán nước, không đặt máy bán nước trên đường đi, và các ràng buộc khác.

4. Áp dụng thuật toán của bài toán chọn vị trí cơ sở (Facility Location):

   Áp dụng các thuật toán tối ưu để tìm ra vị trí tối ưu cho máy bán nước tự động tại Đại học Bách Khoa Hà Nội. Các thuật toán có thể bao gồm các thuật toán Heuristic, thuật toán Lagrange và các thuật toán khác tùy thuộc vào yêu cầu của đại học và cách thức triển khai.

5. Đánh giá và tinh chỉnh kết quả:

   Sau khi áp dụng các thuật toán để tìm được vị trí tối ưu cho cây bán nước, chúng ta cần đánh giá kết quả để đảm bảo rằng nó thực sự phù hợp với nhu cầu của đại học Bách Khoa Hà Nội và các sinh viên. Chúng ta có thể đánh giá hiệu quả của vị trí bằng cách đo lường số lượng sinh viên sử dụng cây bán nước, đánh giá việc truy cập và di chuyển đến vị trí đó, đảm bảo rằng cây bán nước không gây cản trở hoạt động của các khu vực khác trong trường.

3.2 Mô hình bài toán và hàm mục tiêu

Bài toán thử nghiệm được mô tả chi tiết như sau:
Đầu vào

- $I = \{1, 2 \dots n\}$ là các toà nhà học của sinh viên tại Đại học Bách Khoa Hà Nội
- $J = \{1, 2 \dots m\}$ là các vị trí tiềm năng đặt máy bán nước
- h_i là nhu cầu của sinh viên (số lượng sinh viên hay số lượng phòng học trong toà nhà i) $i \in I$
- d_{ij} là khoảng cách giữa toà nhà $i \in I$ và vị trí được ứng cử để chọn đặt máy bán nước $j \in J$
- P là số thiết bị cần đặt

Các biến quyết định

$$X_j = \begin{cases} 1 & \text{nếu chọn vị trí } j \in J \\ 0 & \text{nếu ngược lại} \end{cases}$$

$$Y_{ij} = \begin{cases} 1 & \text{nếu yêu cầu của khách hàng } i \in I \text{ được đáp ứng tại vị trí } j \in J \\ 0 & \text{nếu ngược lại} \end{cases}$$

Hàm mục tiêu có thể được biểu diễn như sau:

$$\textbf{MINIMIZE} \quad \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij} \tag{3.1}$$

$$\textbf{SUBJECT TO:} \quad \sum_{j \in J} Y_{ij} = 1 \quad \forall i \in I \tag{3.2}$$

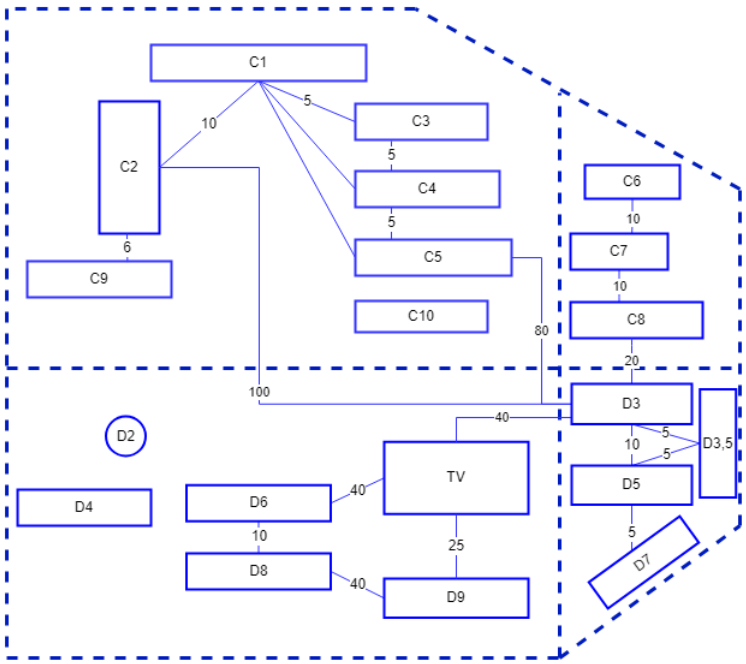
$$\sum_{j \in J} X_j = P \tag{3.3}$$

$$Y_{ij} - X_j \leq 0 \quad \forall i \in I; j \in J \tag{3.4}$$

$$X_j \in \{0, 1\} \quad \forall j \in J \tag{3.5}$$

$$Y_{ij} \in \{0, 1\} \quad \forall i \in I; j \in J \tag{3.6}$$

Sơ đồ mạng toà nhà Đại học Bách Khoa Hà Nội:



Hình 3.1: Sơ đồ mạng toà nhà Đại học Bách Khoa Hà Nội

Em chọn các địa điểm D3, D3,5, D5, D6, D7, D8, D9, TV, C1, C3, C4 và C5 là các vị trí khách hàng và là các vị trí tiềm năng đặt máy vì các toà này thường xuyên được sinh viên học.

Ma trận khoảng cách:

	D3	D3,5	D5	D6	D7	D8	D9	TV	C1	C3	C4	C5
D3	0	5	10	80	15	80	60	30	95	90	85	80
D3,5	5	0	10	85	20	85	65	35	100	95	90	85
D5	10	5	0	85	5	85	20	35	115	100	95	90
D6	80	85	85	0	5	10	45	40	90	85	80	75
D7	15	5	5	85	0	80	5	35	110	105	100	95
D8	80	85	85	10	80	0	40	50	80	75	70	65
D9	60	65	20	45	5	40	0	25	90	85	80	75
TV	30	35	35	40	35	50	25	0	80	75	70	65
C1	95	100	115	90	110	80	90	80	0	5	10	15
C3	90	95	100	85	105	75	85	75	5	0	5	10
C4	85	90	95	80	100	70	80	70	10	5	0	5
C5	80	85	90	75	95	65	75	65	15	10	5	0

Bảng 3.1: Ma trận khoảng cách các toà trong Đại học Bách Khoa Hà Nội

Ở đây, các cột là các vị trí tiềm năng đặt máy bán nước tự động (J), các hàng là các sinh viên có nhu cầu sử dụng máy trong từng toà học (I).

Nhu cầu h_i của các toà học (bằng số phòng học của toà):

Vị trí	D3	D3,5	D5	D6	D7	D8	D9	TV	C1	C3	C4	C5
h_i	10	7	18	25	25	40	30	8	10	10	10	10

Bảng 3.2: Nhu cầu h_i của các toà học

3.3 Cài đặt và thử nghiệm

3.3.1 Môi trường và các công cụ sử dụng

Cấu hình phần cứng:

Phần mềm sử dụng	Chỉ số
Bộ xử lý	Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz
Bộ nhớ	8.00 GB (7.88 GB usable)
Ổ cứng	SSD 256GB
Hệ điều hành	Windows 11 Home Insider Preview

Phần mềm sử dụng:

Phần mềm sử dụng
Visual Studio Code

- Ngôn ngữ lập trình sử dụng: Python
- Giới thiệu thư viện Pulp

  Thư viện Pulp là một thư viện Python được sử dụng để mô hình hóa và giải các bài toán quy hoạch tuyến tính và quy hoạch nguyên (linear and integer programming). Pulp cung cấp các công cụ và giao diện dễ sử dụng để xây dựng các mô hình tối ưu và giải chúng bằng cách sử dụng các solver tối ưu hóa mạnh mẽ như CBC, CPLEX, Gurobi, và GLPK.

Tạo danh sách các vị trí, ma trận khoảng cách d_{ij} và vector nhu cầu h_i:

```
location = ["D3", "D3.5", "D5", "D6", "D7", "D8", "D9", "TV", "C1",
    "C3", "C4", "C5"]
D = dict(zip(location,
[dict(zip(location, [0, 5, 10, 80, 15, 80, 60, 30, 95, 90, 85, 80])),
dict(zip(location, [5, 0, 10, 85, 20, 85, 65, 35, 100, 95, 90, 85])),
dict(zip(location, [10, 5, 0, 85, 5, 85, 20, 35, 115, 100, 95, 90])),
dict(zip(location, [80, 85, 85, 0, 85, 10, 45, 40, 90, 85, 80, 75])),
dict(zip(location, [15, 5, 5, 85, 0, 80, 5, 35, 110, 105, 100, 95])),
dict(zip(location, [80, 85, 85, 10, 80, 0, 40, 50, 80, 75, 70, 65])),
```

```
dict(zip(location, [60, 65, 20, 45, 5, 40, 0, 25, 90, 85, 80, 75])),
dict(zip(location, [30, 35, 35, 40, 35, 50, 25, 0, 80, 75, 70, 65])),
dict(zip(location, [95, 100, 115, 90, 110, 80, 90, 80, 0, 5, 10,
   15])),
dict(zip(location, [90, 95, 100, 85, 105, 75, 85, 75, 5, 0, 5, 10])),
dict(zip(location, [85, 90, 95, 80, 100, 70, 80, 70, 10, 5, 0, 5])),
dict(zip(location, [80, 85, 90, 75, 95, 65, 75, 65, 15, 10, 5,
   0]))]))

h = {"D3": 10, "D3.5": 7, "D5": 18, "D6": 25, "D7": 25, "D8": 40,
   "D9": 30, "TV": 8, "C1": 10, "C3": 10, "C4": 10, "C5": 10}
```

Định nghĩa biến quyết định, hàm mục tiêu và các ràng buộc:

```
Y = LpVariable.dicts("Y", (location, location), cat="Binary",
   lowBound=0, upBound=1)

prob = LpProblem("P Median", LpMinimize)

prob += sum(sum(h[i] * D[i][j] * Y[i][j] for j in location) for i in
   location)
prob += sum(Y[i][i] for i in location) == p

for i in location:
    prob += sum(Y[i][j] for j in location) == 1

for i in location:
    for j in location:
        prob += Y[i][j] <= Y[j][j]
```

3.3.2 Kết quả chạy chương trình

Với p = 1:

- Giá trị tối ưu hàm mục tiêu: 7765

- Vị trí được đặt là: D9
- Thời gian chạy chương trình: 0.03s

Với p = 2:

- Giá trị tối ưu hàm mục tiêu: 3960
- Các vị trí được đặt là: D8, D7
- Thời gian chạy chương trình: 0.02s

Với p = 3:

- Giá trị tối ưu hàm mục tiêu: 1260
- Các vị trí được đặt là: C4, D7, D8
- Thời gian chạy chương trình: 0.03s

Với p = 4:

- Giá trị tối ưu hàm mục tiêu: 915
- Các vị trí được đặt là: C3, D3,5, D8, D9
- Thời gian chạy chương trình: 0.02s

Tổng kết

Các kết quả đồ án

Đã đạt được

1. Tìm hiểu về khái niệm bài toán chọn vị trí.
2. Trình bày mô hình bày toán P-median và một số các thuật toán Heuristic giải bài toán P-median.
3. Áp dụng mô hình bài toán P-median thực hiện bài toán chọn vị trí cây bán nước tự động tại Đại học Bách Khoa Hà Nội.

Chưa đạt được

1. Chưa tìm ra được một số ràng buộc khác cho bài toán thử nghiệm.

Hướng phát triển đồ án trong tương lai

1. Tìm hiểu thêm một số bài toán chọn vị trí, một số các giải thuật giải bài toán chọn vị trí khác.
2. Thực hiện chọn vị trí cho các thiết bị ở quy mô mạng, sơ đồ lớn hơn có thể là chọn vị trí đặt nhà máy, cửa hàng, cây ATM,... cho cả một khu vực thành phố lớn.

Tài liệu tham khảo

Tiếng Anh

[1] Mark S. Daskin (2013), *Network and Discrete Location: Models, Algorithms, and Applications, Second Edition.*

[2] Dileep R. Sule (2008), *Manufacturing Facilities Location, Planning, and Design, Third Edition.*

[3] Tingying Wu (2015), *Models and algorithms for two echelon capacitated facility location problem with facility size selection.*

[4] Kathrin Klamroth (2002), *Single-Facility Location Problems with Barriers.*

Tiếng Việt

[5] Vũ Đức Quang (2016), *Luận văn thạc sĩ, Áp Dụng Thuật Toán Tối Ưu Hoá Đàn Kiến Để Giải Quyết Bài Toán Vị Trí Cơ Sở.*