Tạp chí Khoa học Đại học Cần Thơ
Phần A: Khoa học tự nhiên, Công nghệ và Môi trường
ISSN 1859-2333 | e-ISSN 2815-5599

DOI:10.22144/ctujos.2024.396

BÀI TOÁN LẬP LỊCH-VỊ TRÍ NGƯỢC VỚI ĐỘ TRỄ TỐI ĐA TRÊN CÂY

Lê Minh Huy[1*] và Trần Thanh Hiệp[2,3,4]
[1]*Khoa Khoa học Cơ bản, Trường Đại học Văn Lang, Thành phố Hồ Chí Minh Việt Nam*
[2]*Khoa Khoa học ứng dụng, Trường Đại học Bách khoa Thành phố Hồ Chí Minh*
[3]*Đại học Quốc gia Thành phố Hồ Chí Minh*
[4]*Trường Đại học FPT, Phân hiệu Hồ Chí Minh, Việt Nam*
**Tác giả liên hệ (Corresponding author): huy.lm@vlu.edu.vn*

Thông tin chung (Article Information)

Nhận bài (Received): 02/12/2023
Sửa bài (Revised): 23/01/2024
Duyệt đăng (Accepted): 20/02/2024

Title: *The reverse scheduling-location problem with maximum lateness on trees*

Author(s): *Huy Minh Le[1*] and Tran Thanh Hiep[2,3,4]*

Affiliation(s): *[1]Van Lang University; [2]Ho Chi Minh City University of Technology; [3]Vietnam National University, Ho Chi Minh City; [4]FPT University, Ho Chi Minh Campus*

TÓM TẮT

Bài báo giải quyết vấn đề tìm vị trí của máy trên cây và lập kế hoạch cho các công việc nhất định (nằm ở các đỉnh) sao cho độ trễ tối đa được giảm thiểu. Khách hàng (công việc) di chuyển đến máy bất cứ khi nào họ được gọi do thiếu không gian dành cho công việc tại vị trí đặt máy. Một thuật toán giải bài toán tương ứng trong thời gian $O(n^2)$ đã được phát triển, trong đó n là số đỉnh trên cây. Mặt khác, vấn đề giảm độ dài cạnh trong một ngân sách nhất định được xem xét để cải thiện độ trễ tối đa càng nhiều càng tốt tại một điểm được xác định trước. Đây được gọi là bài toán lập lịch-vị trí ngược với độ trễ tối đa trên cây. Một thuật toán tham lam được đề xuất để giải quyết vấn đề trong thời gian bậc hai.

Từ khoá: *Bài toán lập lịch, bài toán vị trí, cây, độ trễ, tối ưu ngược*

ABSTRACT

We address the problem of finding the location of a machine on trees and scheduling jobs (located at vertices) so that the maximum lateness is minimized. The customers (jobs) move to the machine whenever they are called due to the slackness of space for jobs waiting at the machine location. We develop an algorithm that solves the corresponding problem in $O(n^2)$ time, where n is the number of vertices in the tree. On the counterpart, we investigate the problem of reducing edge lengths within a given budget to improve the maximum lateness at a prespecified point as much as possible. We call it the reverse scheduling location problem with maximum lateness on trees. A greedy-type algorithm that solved the problem in quadratic time is discussed.

Keywords: *Lateness, Location problem, reverse optimization, Scheduling problem, trees*

1. GIỚI THIỆU

Cuộc cách mạng khoa học kỹ thuật lần thứ tư đề cập đến sự mở rộng và phát triển của những điều chỉnh đối với mạng máy tính, hệ thống viễn thông hoặc các hệ thống kết nối khác theo thời gian. Mô hình của bài toán điều chỉnh mạng lưới có hai hướng tiếp cận khác nhau: mô hình bài toán nghịch đảo và mô hình bài toán ngược. Trong mô hình bài toán nghịch đảo, Burkard et al. (2004), Bonab et al.

(2010) và Sepasian and Rahbarnia (2015) cho rằng điều chỉnh mạng lưới liên quan đến việc thay đổi một số thuộc tính trong mạng để tối ưu hóa các vị trí được xác định trước trong khi giảm thiểu chi phí sửa đổi. Trong khi đó, Berman et al. (1992, 1994), Nazari and Fathali (2023) phát biểu bài toán ngược là điều chỉnh các tham số để tối đa hóa mục tiêu hoạt động tại một địa điểm cố định trong sự giới hạn của một ngân sách nhất định. Vì mỗi loại mô hình đều có những điểm độc đáo riêng biệt nên ta cần có những phương pháp tiếp cận khác nhau.

Đối với mô hình bài toán ngược, lĩnh vực này đã được các nhà khoa học trên thế giới dành nhiều sự quan tâm. Một số nghiên cứu nổi bật trong thời gian gần đây có thể được kể đến như Etemad and Alizadeh (2018), Sepasian and Tayyebi (2020), Yim et al. (2020), Tayyebi and Sepasian (2023). Tuy nhiên, các nhà khoa học chỉ dừng lại ở việc nghiên cứu mô hình ngược cho các bài toán riêng lẻ mà chưa xem xét đến mô hình ngược đột phá mang tính tổ hợp trong nhiều lĩnh vực. Tiêu biểu như các bài toán thuộc hai lĩnh vực của vận trù học là lập lịch và vị trí.

Người ta đã từng quan niệm rằng cách cài đặt của hai vấn đề này là độc lập và không tương quan nhau. Tuy nhiên, có nhiều ứng dụng trong thực tế đòi hỏi việc định vị vị trí và việc lập kế hoạch phải thực hiện đồng thời. Để minh họa cho tuyên bố này, chúng ta hãy xem xét ví dụ về kế hoạch sơ tán. Mục đích là để giảm thiểu thời gian cần thiết để sơ tán khỏi khu vực nguy hiểm càng nhanh càng tốt theo một trật tự. Mối đe dọa hoặc sự xuất hiện của một sự kiện thảm khốc (ví dụ: bom, lũ lụt, tai nạn công nghiệp, bão,…) có thể là lý do phải sơ tán khỏi các mạng lưới của tòa nhà hoặc mạng lưới khu vực. Rõ ràng, xem xét đồng thời việc xác định vị trí địa điểm an toàn trong mạng lưới và xác định quy trình sơ tán dự kiến dẫn đến một kế hoạch sơ tán được tối ưu hóa. Hennes (2005), Kalsch and Drezner (2010) và Hessler (2016) đã chỉ ra rằng sự kết hợp này giải quyết được rất nhiều bài toán thực tế trong cuộc sống mà các mô hình riêng lẻ chưa thể làm được. Krumke and Le (2022) đã giải quyết bài toán tối ưu tổ hợp này với hàm mục tiêu là độ trễ tối đa dưới dữ liệu không chắc chắn trên cây. Vì thế, việc đề xuất mô hình ngược cho những bài toán mang tính tổ hợp nhiều lĩnh vực khác nhau là rất cần thiết.

Với những lý do trên, bài toán tối ưu lập lịch-vị trí ngược được tìm hiểu sâu với hàm mục tiêu là độ trễ tối đa trên cây.

2. BÀI TOÁN LẬP LỊCH-VỊ TRÍ VỚI ĐỘ TRỄ TỐI ĐA TRÊN CÂY

Cho cây $T = (V, E)$ là một cây với n đỉnh. Mỗi công việc j với $j = 1, 2, \ldots, n$ đặt tại các đỉnh tương ứng v_j. Mỗi cạnh e của cây được gắn với một độ dài cạnh $l(e) > 0$, và được xem là một khoảng liên tục của những điểm nằm trên nó. Mỗi điểm ρ trên cạnh $e = (u,v)$ được định nghĩa bởi một tham số λ trong (0,1) với $d(\rho,u) = \lambda l(e)$ và $d(\rho,v) = (1 - \lambda)l(e)$. Khoảng cách $d(x, y)$ giữa hai điểm x và y là độ dài của đường đi kết nối chúng. Lấy $A(T)$ là tập hợp tất cả các điểm trên cây. Mỗi đỉnh v_j với $j = 1, 2, \ldots, n$ được gắn với những dữ liệu sau:

Thời gian chờ σ_j và tốc độ di chuyển ν_j. Đặt $\tau_j = \frac{1}{\nu_j}$, chúng ta có thể định nghĩa thời gian xử lý của công việc j tại vị trí v_j và vị trí máy đặt tại điểm ρ như sau:

$$t_j(\rho) = \tau_j d(v_j, \rho) + \sigma_j,$$

là bằng tổng thời gian di chuyển và thời gian chờ.

– Thời gian đến hẹn d_j.

Trong cài đặt của *bài toán lập lịch-vị trí với độ trễ tối đa trên cây (gọi tắt là bài toán (P))*, chúng ta giả sử rằng không có đủ không gian để các công việc đến và chờ tại vị trí của máy. Mỗi công việc chỉ được phép di chuyển nếu được gọi bởi máy và máy đã hoàn thành xử lý công việc trước đó. Kí hiệu Π là tập hợp tất cả các hoán vị của n công việc và $\pi \in \Pi$ là một hoán vị hoặc một lịch của n công việc. Thêm vào đó, $\pi(i) = j$ ngụ ý rằng công việc j được xử lý tại vị trí i trên lịch π. Khi đó, thời gian hoàn thành của công việc $\pi(i)$ được tính như sau:

$$C_{\pi(i)}(\rho) := C_{\pi(i-1)}(\rho) + t_{\pi(i)}(\rho),$$

với $i = 2, \ldots, n$ và $C_{\pi(1)}(\rho) := t_{\pi(1)}(\rho)$. Mặt khác, chúng ta có thể viết:

$$C_{\pi(i)}(\rho) = \sum_{s=1}^{i} t_{\pi(s)}(\rho).$$

Độ trễ của mỗi công việc $\pi(i)$ được tính bởi:

$$\max\{0, C_{\pi(i)}(\rho) - d_{\pi(i)}\}.$$

Mục tiêu của *bài toán (P)* là tìm một vị trí trên cây T để đặt máy và đồng thời tìm một lịch sao cho độ trễ đạt giá trị tối thiểu. Bài toán được công thức hoá như sau:

$$\min_{\rho \in A(T), \pi \in \Pi} \max_{i=1,2,\ldots,n} \{0, C_{\pi(i)}(\rho) - d_{\pi(i)}\}. \quad (1)$$

Chúng ta thảo luận về ứng dụng của mô hình này. Trình tự các công việc được xác định theo cách này khác với khái niệm về bài toán lập lịch-vị trí của Hennes and Hamacher (2001), trong đó các tác giả cho rằng tất cả các công việc có thể di chuyển đồng thời đến máy. Tuy nhiên, mô hình này tập trung vào các công việc được gọi sau khi máy hoàn thành quá trình xử lý của công việc trước đó. Mô hình này được áp dụng khi không có nơi chờ cho các công việc tại vị trí đặt máy và chúng chỉ có thể di chuyển đến máy nếu công việc trước đó đã hoàn thành. Lấy ví dụ về việc sản xuất một loại sản phẩm, chúng ta cần một loại tài nguyên tại một thời điểm và gọi cho nhà cung cấp đến khi cần, trong khi sức chứa của nhà kho không đủ lớn. Hơn nữa, chúng ta không gọi một nhà cung cấp nếu họ không đến đúng như thời gian hẹn. Vì thế, vấn đề này có thể được mô hình dưới dạng *bài toán (P)*.

Tại một điểm cố định ρ trên cây, độ trễ đạt được giá trị nhỏ nhất nếu chúng ta lập lịch các công việc dựa trên thứ tự không giảm của các thời gian đến hẹn (Brucker, 2007). Hơn nữa, thứ tự này không phụ thuộc vào vị trí của máy, nó là như nhau cho tất cả các điểm trên cây. Vì thế, một lịch tối ưu, kí hiệu $\pi = (1,2,\dots,n)$, cho *bài toán (P)* được cho bởi một dãy các công việc sao cho $d_1 \le d_2 \le \cdots \le d_n$. Không mất tính tổng quát, chúng ta có thể giả sử rằng $\pi(i) = i$ cho tất cả công việc i = 1, 2, …, n. Thời gian hoàn thành của công việc i được viết lại như sau: $C_i(\rho) = \sum_{s=1}^{i} t_s(\rho)$. Bài toán (1) có thể được viết một cách đơn giản như sau:

$$\min_{\rho\in A(T)} \max_{i=1,2,\dots,n} \{0, C_i(\rho) - d_i\}. \quad (2)$$

Đặt $z(\rho) = \max_{i=1,2,\dots,n} \{0, C_i(\rho) - d_i\}$, bài toán (2) được mô hình thành:

$$\min_{\rho\in A(T)} z(\rho)$$
$$\text{s.t.} \quad z(\rho) \ge \sum_{s=1}^{i} (\tau_s d(v_s,\rho) + \sigma_s) - d_i, \quad (3)$$
$$z(\rho) \ge 0,$$
$$\rho \in A(T).$$

Đặt $\omega_i = \sum_{s=1}^{i} \sigma_s$ là tổng thời gian chờ của các công việc từ 1 đến i, và $\Delta_i = \omega_i - d_i$. Ngoài ra, chúng ta xét những cây phụ trợ T^i với i = 1, 2, …, n, trong đó độ dài cạnh của cây này vẫn giống như trong cây T và trọng số các đỉnh v_s trong cây T^i được định nghĩa như sau: $w_s^i = \tau_s$ với s = 1, 2, …, i và $w_s^i = 0$ với $s > i$. Khi đó, tại mỗi điểm $\rho \in A(T)$

$$f^i(\rho) = \sum_{s=1}^{i} w_s^i d(v_s,\rho)$$

được định nghĩa là hàm trung vị trong cây T^i.

Ta có thể mô hình lại bài toán (3) như sau:

$$\min_{\rho\in A(T)} z(\rho)$$
$$\text{s.t.} \quad z(\rho) \ge \Delta_i + f^i(\rho),\ i = 1, 2, \dots, n, \quad (4)$$
$$z(\rho) \ge 0,$$
$$\rho \in A(T).$$

Đặt v^i là vị trí tối ưu 1-trung vị của cây phụ trợ T^i với i = 1, 2, …, n. Lấy $\bar{T}$ là cây con nhỏ nhất của T mà chứa tất cả các 1-trung vị v^i với i = 1, 2, …, n. Cây này được tạo ra như sau: nếu có một đường đi nối giữa hai trung vị v^i và v^k bất kỳ thì đường đi này phải nằm trong cây con nhỏ nhất. Hơn nữa, số lượng đỉnh có trong $\bar{T}$ phải là nhỏ nhất.

Mệnh đề 1. *Bài toán (4) luôn có nghiệm là một vị trí tối ưu nằm trong* $\bar{T}$.

Chứng minh. Cho một điểm bất kỳ ρ nằm trong $T\backslash\bar{T}$, ta xét đỉnh $v^* = \arg\min_{1\le i\le n} d(\rho, v^i)$. Chú ý rằng tất cả các hàm $\Delta_i + f^i(\rho)$ giảm dần trên đường đi từ điểm ρ đến v^i. Do đó, $z(v^*) < z(\rho)$. ■

Mệnh đề 2. $z(\rho)$ *là một hàm lồi tại mỗi điểm* ρ *nằm trên mỗi đường đi của cây.*

Chứng minh. Hàm trung vị $f^i(\rho)$ là lồi trên mỗi đường đi của cây, dựa theo Goldman (1971). Vì thế, $z(\rho) = \max_{1\le i\le n}\{0, \Delta_i + f^i(\rho)\}$ cũng là một hàm lồi, bởi vì nó là hàm tối đa của những hàm lồi. ■

Xét một cạnh bất kỳ $e = (u, v)$. Tại đỉnh u, ta tính toán tất cả các giá trị $f^i(u)$ với $i = 1, 2, \dots, n$. Khi đó,

$$f^i(v) = f^i(u) + [W^i - 2W^i(v)]l(e),$$

trong đó W^i là tổng trọng số của tất cả các đỉnh trong cây T^i và $W^i(v)$ là tổng trọng số đỉnh của các cây con của T^i có gốc tại v. Từ đó, giá trị của $f^i(\rho)$ tại một điểm $\rho \in e$ với $d(\rho, u) = \lambda l(e)$ được tính như sau:

$$f^i(\rho) = f^i(u) + [W^i - 2W^i(v)]\lambda l(e).$$

Đồ thị của mỗi hàm $f^i(\rho)$ là một đoạn tuyến tính với $\lambda \in [0,1]$ và được tìm trong thời gian tuyến tính, nên chúng ta cần $O(n^2)$ thời gian để tìm các giá trị $f^i(\rho)$ với $i = 1, 2, \dots, n$. Vì thế giá trị $z(\rho)$ có thể được tính toán trong thời gian $O(n^2)$.

Từ đây, chúng ta tập trung tìm một vị trí tối ưu của bài toán (4) trên cây $\bar{T}$ và đề xuất thuật toán để tìm một vị trí tối ưu trên cây này thông qua việc tìm trọng tâm của cây. Trọng tâm của một cây được định nghĩa là một đỉnh sao cho khi cây có gốc tại đó thì không có đỉnh nào khác có cây con có tổng số đỉnh lớn hơn *n*/2. Trọng tâm của một cây có thể được tìm thấy trong thời gian tuyến tính bằng cách áp dụng thuật toán sự phân huỷ của trọng tâm (Handler & Mirchandani, 1979). Chúng ta đề xuất thuật toán để giải *bài toán (P)* như sau:

Xét v^c là trọng tâm của cây $\bar{T}$ và một đỉnh kề với nó là v. Giả sử rằng $e = (v^c, v)$. Khi đó, cây $\bar{T}$ bị chia thành ba phần như sau $\bar{T} = \bar{T}^{v^c} \cup \{e\} \cup \bar{T}^{v}$, trong đó $\bar{T}^{v^c}$ và $\bar{T}^{v}$ lần lượt là các cây con có gốc tại v^c và v đạt được bằng cách xoá đi cạnh e khỏi cây $\bar{T}$. Nếu như $z(v^c) = z(v)$, thì $\bar{T} = e$. Nếu $z(v^c) > z(v)$, thì ta xét những đỉnh nằm trong phần $\bar{T}^{v}$. Ngược lại, chúng ta xét cây $\bar{T}^{v^c}$. Do đó, thuật toán tỉa bớt một nửa số đỉnh trong cây $\bar{T}$ sau mỗi vòng lặp. Thuật toán kết thúc khi cây $\bar{T}$ chỉ còn lại một cạnh e^*. Nó chính là cạnh chứa vị trí tối ưu của bài toán (4). Thuật toán kết thúc sau $O(\log n)$ bước. Để tìm một vị trí tối ưu ρ trên cạnh e^*,ta tìm kiếm giá trị nhỏ nhất của các đường bao trên $f^i(\rho)$ với $i = 1, 2, \dots, n$ trong thời gian tuyến tính. Ta đạt được kết quả sau:

Định lý 1. *Bài toán (P) có thể được giải trong thời gian* $O(n^2)$.

Chứng minh. Thời gian chạy của thuật toán chủ yếu được quyết định bởi việc tính toán hàm mục tiêu $z(\rho)$ trong thời gian $O(n^2)$ tại một điểm ρ bất kỳ trên cây $\bar{T}$. ■

3. BÀI TOÁN LẬP LỊCH-VỊ TRÍ NGƯỢC VỚI ĐỘ TRỄ TỐI ĐA TRÊN CÂY

Trong phần này, máy đã được đặt sẵn tại một điểm cho trước trên cây. Không mất tính tổng quát, chúng ta có thể giả sử rằng điểm cho trước này là một đỉnh của cây *T*. Ngược lại, ta có thể chia cạnh chứa điểm này thành hai cạnh mới với một đầu mút chính là điểm cho trước. Trong *bài toán lập lịch-vị trí ngược với độ trễ tối đa trên cây (gọi tắt là bài toán (P'))*, chúng ta xét cây $T = (V, E)$ và một đỉnh cho trước v^*. Dữ liệu đầu vào vẫn liên quan đến thời gian chờ σ_i, tốc độ di chuyển ν_i và thời gian đến hẹn d_i cho mỗi công việc $i \in \{1, 2, \dots, n\}$. Thời gian xử lý của mỗi công việc được tính toán như ở Phần 2. Mục tiêu của *bài toán (P')* là điều chỉnh độ dài các cạnh trong một ngân sách giới hạn sao cho đỉnh v^* có giá trị độ trễ là nhỏ nhất so với các điểm còn lại trên cây. Một cách cụ thể, độ dài của mỗi cạnh $e \in E$ có thể được giảm bởi một lượng $x(e)$ không vượt quá chặn trên $\bar{x}(e)$, tức là $0 \le x(e) \le \bar{x}(e)$. Độ dài cạnh sau khi điều chỉnh là $\tilde{l}(e) = l(e) - x(e)$, với mọi $e \in E$. Chú ý rằng độ dài cạnh luôn là không âm mặc dù đã điều chỉnh, do vậy $\bar{x}(e) < l(e)$. Khi đó, thời gian xử lý và thời gian hoàn thành của công việc i tại đỉnh v^* sau khi điều chỉnh lần lượt là $\tilde{t}_i(v^*)$ và $\tilde{C}_i(v^*)$. Hơn nữa, tổng lượng điều chỉnh độ dài của các cạnh không vượt quá ngân sách một cho trước B. Bài toán có thể được mô hình hoá như sau:

$$\min_{x(e)} \quad \max_{1 \le i \le n} \{0, \tilde{C}_i(v^*) - d_i\}$$

$$\text{s.t. } \tilde{C}_i(v^*) = \sum_{s=1}^{i} \left(\tau_s \tilde{d}(v_s, v^*) + \sigma_i\right), i = 1,2, \dots, n$$

$$\tilde{d}(v_s, v^*) = d(v_s, v^*) - \textstyle\sum_{e \in P(v_s, v^*)} x(e), \quad (5)$$

$$\sum\nolimits_{e \in E} x(e) \le B,$$

$$0 \le x(e) \le \bar{x}(e), \forall e \in E.$$

Ngoài ra,

$$\begin{aligned}\tilde{C}_i(v^*) &= \sum_{s=1}^{i} \tau_s \left[d(v_s, v^*) - \sum\nolimits_{e \in P(v_s, v^*)} x(e)\right] + \omega_i \\ &= \sum_{s=1}^{i} \tau_s d(v_s, v^*) + \omega_i \\ &\quad - \sum_{s=1}^{i} \sum\nolimits_{e \in P(v_s, v^*)} \tau_s x(e).\end{aligned}$$

Lấy $\mathcal{L}_i(v^*) = \max\{0, C_i(v^*) - d_i\}$ là độ trễ của công việc i và $\mathcal{L}_{\max}(v^*) = \max_{1 \le i \le n} \mathcal{L}_i(v^*)$ là độ trễ tối đa tại đỉnh v^*. Ta kí hiệu $x = \left(x(e)\right)_{e \in E}$ là véc tơ nghiệm của bài toán (5). Khi đó,

$$\mathcal{F} = \left\{x \in \mathbb{R}^{|E|} : \sum_{e \in E} x(e) \le B \text{ và } 0 \le x(e) \le \bar{x}(e), \forall e \in E\right\}$$

là tập hợp tất cả các nghiệm khả thi. Bài toán (5) tương đương với:

$$\min \max_{1 \le i \le n} \left\{0, \mathcal{L}_i(v^*) - \sum_{s=1}^{i} \sum_{e \in P(v_s, v^*)} \tau_s x(e)\right\}.$$

Một thuật toán kiểu tham lam được đề xuất để giải quyết cho *bài toán (P')* trên cây T có gốc tại đỉnh v^*. Xét một cạnh bất kỳ $e = (u, v)$ với u là đỉnh gần v^* hơn v, lấy T_e là cây con được sinh ra bởi v và các đỉnh bên dưới nó. Tập hợp tất cả các đỉnh có chỉ số cao nhất là i được kí hiệu bởi tập hợp $V^i = \{v_1, v_2, \dots v_i\}$. Hơn nữa, tập chỉ số

$$I^i(e) = \{s : v_s \in V(T_e) \cap V^i\}$$

bao gồm tất cả các chỉ số mà đỉnh tương ứng vừa nằm trong cây con T_e và vừa nằm trong tập V^i. Khi đó, việc giảm độ dài của một cạnh $e \in E$ bởi một lượng $x(e) = \varepsilon < \bar{x}(e)$ sẽ mang lại độ trễ của công việc i như sau:

$$\tilde{\mathcal{L}}_i(v^*) = \mathcal{L}_i(v^*) - \left(\sum_{s \in I^i(e)} \tau_s\right)\varepsilon,$$

với $i = 1,2, \dots, n$. Để đơn giản kí hiệu, ta đặt $\Delta_i(e) \coloneqq \sum_{s \in I^i(e)} \tau_s$. Việc tính toán $\Delta_i(e)$ với mọi $i = 1,2, \dots, n$ và $e \in E$ tốn thời gian bậc hai bằng thuật toán tìm kiếm đầu tiên theo chiều rộng cơ sở (Awerbuch & Gallager, 1987). Ta có thể nói rằng một cạnh e là gần với v^* hơn cạnh e', nếu e' nằm trong cây con T_e. Kết quả sau đây giúp chúng ta xác định được cạnh nào là cạnh cần giảm để cải thiện độ trễ tối đa nhiều nhất có thể với một lượng giảm đủ nhỏ ε.

Mệnh đề 3. *Nếu cạnh e gần với v^* hơn cạnh e', thì việc giảm độ dài cạnh e' có thể được chuyển sang việc giảm độ dài cạnh e với cùng lượng giảm mà không làm tăng độ trễ tối đa.*

Chứng minh. Với bất kỳ một chỉ số nào, ta có $I^i(e') \subseteq I^i(e)$. Vì thế, $\Delta_i(e') \leq \Delta_i(e)$ với mọi i. ■

Dựa theo Mệnh đề 3, chúng ta tập trung vào việc giảm độ dài những cạnh gần với v^* hơn những cạnh khác. Kí hiệu $\mathcal{E}$ là tập của những cạnh có $\bar{x}(e) > 0$ sao cho không tồn tại bất kỳ một cạnh e' nào gần v^* hơn e và $\bar{x}(e') > 0$. Hơn nữa, lấy $\mathcal{M}$ là tập hợp của tất cả các chỉ số tương ứng xác định độ trễ tối đa tại đỉnh v^*.

Nếu chúng ta giảm độ dài cạnh $e \in \mathcal{E}$ bởi một lượng đủ nhỏ ε (sao cho tập $\mathcal{M}$ là không thay đổi), thì $\tilde{\mathcal{L}}_i(v^*) = \mathcal{L}_i(v^*) - \Delta_i(e)\varepsilon$ với mọi $i \in \mathcal{M}$. Lấy $\delta(e) = \min\limits_{1 \leq i \leq n} \Delta_i(e)$. Ta thu được kết quả sau:

Mệnh đề 4. *Lấy cạnh $e_0 \coloneqq arg\max\limits_{e \in \mathcal{E}} \delta(e)$. Việc giảm độ dài cạnh e_0 bởi một lượng đủ nhỏ ε mà không làm tập $\mathcal{M}$ thay đổi sẽ dẫn đến giảm độ trễ tối đa nhiều nhất có thể.*

Chứng minh. Độ trễ tối đa đã điều chỉnh với việc giảm độ dài cạnh e_0 là $\tilde{\mathcal{L}}_i(v^*) = \mathcal{L}_i(v^*) - \Delta_i(e_0)\varepsilon$. Giá trị này rõ ràng là nhỏ nhất trong tất cả các độ trễ tối đa đã điều chỉnh độ dài của những cạnh khác trong tập $\mathcal{E}$. ■

Ta xác định một cạnh e_0 và tìm lượng giảm đủ nhỏ ε sao cho tập $\mathcal{M}$ không thay đổi. Kí hiệu tập chỉ số $\mathcal{N} = \{1,2, \dots, n\} \backslash \mathcal{M}$. Khi đó, với $i \in \mathcal{N}$ ta giải phương trình:

$$\mathcal{L}_i(v^*) - \Delta_i(e_0)\varepsilon_i = \mathcal{L}_{\max}(v^*) - \delta(e_0)\varepsilon_i.$$

Ta thu được:

$$\varepsilon_i = \frac{\mathcal{L}_{\max}(v^*) - \mathcal{L}_i(v^*)}{\delta(e_0) - \Delta_i(e_0)}.$$

Vì $\mathcal{L}_{\max}(v^*) > \mathcal{L}_i(v^*)$ nên ta xét chỉ số $i \in \mathcal{N}$ sao cho $\delta(e_0) > \Delta_i(e_0)$ hoặc giới hạn tới những chỉ số $i < \min\{s : s \in \mathcal{M}\}$. Đặt

$$\mathcal{N}' = \{i \in \mathcal{N} : i < \min\{s : s \in \mathcal{M}\}\}.$$

Ta đạt được kết quả sau:

Mệnh đề 5. *Tập $\mathcal{M}$ chỉ thay đổi khi độ dài của cạnh e được giảm bởi một lượng bằng $\varepsilon = \min\limits_{i \in \mathcal{N}'} \varepsilon_i$.*

Tiếp theo, chúng ta phát triển một thuật toán để giải *bài toán (P')*. Ý tưởng của thuật toán là tìm một cạnh thoả tính chất trong Mệnh đề 4 và giảm độ dài cạnh đó cho đến khi hoặc tập $\mathcal{M}$ hoặc tập $\mathcal{E}$ thay đổi. Sau đó, chúng ta cập nhật lại dữ liệu về độ dài cạnh của tất cả các cạnh. Thuật toán kết thúc khi ngân sách B cạn kiệt hoặc độ trễ tối đa tại đỉnh v^* không thể cải thiện được nữa. Thuật toán sau đây thể hiện chi tiết ý tưởng này.

Thuật toán. Giải *bài toán (P')*

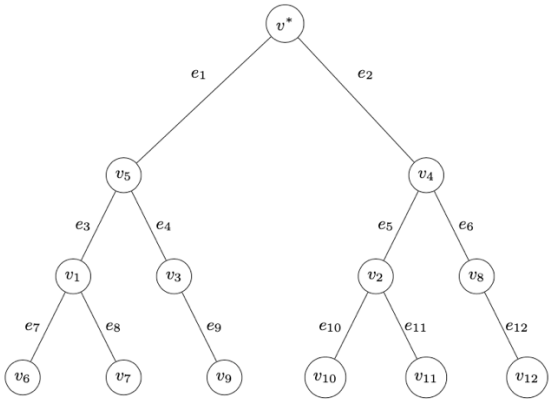
Input: Cây $T = (V, E)$ với độ dài cạnh dương và các thông số của các công việc tại các đỉnh.
Tìm $\mathcal{E}, \mathcal{M}, \mathcal{L}_{\max}, \mathcal{L}_i$ với mọi $i = 1,2, \dots, n$
while $B > 0$ do
  Tìm $e_0 = \arg\max\limits_{e \in \mathcal{E}} \delta(e)$
  Tính toán $x(e_0) = \min\{\bar{x}(e_0), \varepsilon\}$
  $\mathcal{L}_i = \mathcal{L}_i - \Delta_i(e_0)x(e_0)$ với mọi $i \in \mathcal{N}$
  $\mathcal{L}_{\max} \coloneqq \mathcal{L}_{\max} - \delta(e_0)x(e_0)$
  if $x(e_0) = \bar{x}(e_0)$ **then**
    Cập nhật $\mathcal{E}$
  end if
  if $x(e_0) = \varepsilon$ **then**
    Cập nhật $\mathcal{M}$
  end if
  Đặt $B \coloneqq B - x(e)$
end while
Output: Nghiệm tối ưu $x(e), e \in E$.

Chúng ta phân tích độ phức tạp thời gian của Thuật toán trên như sau: Việc tính toán các $\Delta_i(e)$ với mọi $e \in \mathcal{E}$ và $i = 1,2,\dots,n$ được thực hiện trong thời gian bậc hai. Tại mỗi vòng lặp, một cạnh e_0 được giảm bởi một lượng $\min\{\bar{x}(e_0), \varepsilon\}$. Có nhiều nhất $O(n)$ vòng lặp như vậy sao cho một cạnh được giảm bởi một lượng bằng chặn trên của nó. Hơn nữa, tập $\mathcal{M}$ thay đổi nhiều nhất n lần tương ứng với một cạnh e_0. Điều này ngụ ý rằng số lượng vòng lặp trong thuật toán nhiều nhất là trong thời gian bậc hai. Khi mà độ phức tạp về thời gian trong mỗi vòng lặp là hằng số, ta thu được kết quả chính như sau:

Định lý 2. Bài toán (P') với điều chỉnh độ dài cạnh có thể giải được trong thời gian bậc hai.

Chúng ta minh hoạ Thuật toán trên bởi ví dụ sau:

Ví dụ 1. Cho cây $T = (V, E)$ được mô tả như trong Hình 1. Độ dài của tất cả các cạnh bằng 3 và cận trên cho việc điều chỉnh độ dài cạnh là $\bar{x}(e) = 2$ với mọi $e \in E$. Thời gian chờ $\sigma_i = 0$, tốc độ di chuyển $v_i = 1$ với mọi công việc $i = 1, 2,\dots, 12$. Thời gian đến hẹn của tất cả công việc được cho trong Bảng 1. Ngân sách là $B = 8$.



Hình 1. Một ví dụ cho cây trong bài toán

Bảng 1. Thời gian đến hẹn của các công việc

i	1	2	3	4	5	6	7	8	9	10	11	12
d_i	2	3	4	5	6	7	12	18	30	40	60	65

Bảng 2. Thời gian hoàn thành và độ trễ của các công việc tại đỉnh v^*

i	1	2	3	4	5	6	7	8	9	10	11	12
C_i	6	12	18	21	24	33	42	48	57	66	75	84
$\mathcal{L}_i$	4	9	14	16	18	26	30	30	27	26	15	19

Chúng ta tính toán thời gian hoàn thành và độ trễ của các công việc tại đỉnh v^* dựa theo lịch của các công việc được sắp xếp theo chiều không giảm của thời gian đến hẹn như trong Bảng 2.

Ta xác định được $\mathcal{L}_{\max} = 30$ và các tập $\mathcal{E} = \{e_1, e_2\}$ và $\mathcal{M} = \{7, 8\}$.

Vòng lặp 1. Tính toán được $\delta(e_1) = 5$ và $\delta(e_2) = 2$, cạnh $e_1 = \arg\max\{\delta(e_1), \delta(e_2)\}$. Các giá trị ε_i, $i = 1, 2,\dots, 6$ được tính trong Bảng 3.

Ta được $\varepsilon = 2$. Vì $\bar{x}(e_1) = 2$, nên $x(e_1) = 2$ và cập nhật $B = 6$. Cạnh e_1 được cập nhật với $l(e_1) = 1$, $\bar{x}(e_1) = 0$. Các cạnh khác vẫn giữ nguyên độ dài và chặn trên của sự điều chỉnh. Tính toán lại độ trễ tối đa tại đỉnh v^*, ta thu được $\mathcal{L}_{\max} = 20$, do đó $\mathcal{E} = \{e_2, e_3, e_4\}$ và $\mathcal{M} = \{6, 7, 8\}$. Tiếp tục thực hiện các bước tính toán như vòng lặp 1, ta thu được bảng sau:

Bảng 3. Các giá trị ε_i với $i = 1, 2, \dots, 6$

i	1	2	3	4	5	6
ε_i	$\frac{13}{2}$	$\frac{21}{4}$	$\frac{16}{3}$	6	2	3

Bảng 4. Dữ liệu cập nhật tại mỗi vòng lặp

Vòng lặp	Cạnh giảm e_0	Lượng giảm ε	$\mathcal{L}_{\max}$	B	$\mathcal{E}$	$\mathcal{M}$
0			30	8	$\{e_1, e_2\}$	$\{7, 8\}$
1	e_1	2	20	6	$\{e_2, e_3, e_4\}$	$\{6, 7, 8\}$
2	e_2	2	16	4	$\{e_3, e_4, e_5, e_6\}$	$\{6, 7\}$
3	e_3	2	12	2	$\{e_4, e_5, e_6, e_7, e_8\}$	$\{6\}$
4	e_5	2	10	0		

Thuật toán dừng lại sau 4 vòng lặp do ngân sách $B = 0$. Ta thu được nghiệm của *bài toán (P')* là độ dài của các cạnh e_1, e_2, e_3 và e_5 giảm 2 đơn vị, trong khi các cạnh còn lại vẫn giữ nguyên độ dài. Khi đó, độ trễ tối đa tại đỉnh v^* là nhỏ nhất trên cây đã điều chỉnh độ dài.

4. KẾT LUẬN

Bài báo đề cập đến bài toán lập lịch-vị trí ngược trên cây với việc điều chỉnh độ dài cạnh. Bài toán tổ hợp và cách tiếp cận ngược đã được kết hợp phụ thuộc nhau vào một loạt các mô hình tối ưu hóa nổi bật, bao gồm các bài toán lập lịch, điều chỉnh mạng và phân tích vị trí. Những mô hình tích hợp này đóng một vai trò then chốt trong việc thúc đẩy cả sự hiểu biết về lý thuyết và ứng dụng thực tế. Các lĩnh vực nghiên cứu trong tương lai tập trung vào các vấn đề tối ưu hóa cho các bài toán ngược mang tính tổ hợp. Ví dụ như các vấn đề về cân bằng lập lịch-vị trí ngược trên cây, cũng như các vấn đề về phân bổ tài nguyên ngược, cùng nhiều vấn đề khác.

LỜI CẢM TẠ

Nghiên cứu này được tài trợ bởi Trường Đại học Văn Lang với mã số đề tài: VLU-RP-202302. Tác giả xin cảm ơn Trường Đại học Văn Lang, Việt Nam đã tài trợ cho nghiên cứu này.

TÀI LIỆU THAM KHẢO

Alizadeh, B., & Burkard, R. E. (2011). Uniform-cost inverse absolute and vertex center location problems with edge length variations on trees. *Discrete Appl. Math*, *159*(8), 706-716. https://doi.org/10.1016/j.dam.2011.01.009

Awerbuch, B. & Gallager, R. (1987). A new distributed algorithm to find breadth first search trees. *IEEE Transactions on Information Theory*, *33*(3), 315-322. https://doi.org/10.1109/TIT.1987.1057314

Berman, O., Ingco, D. I., & Odoni, A. (1992). Improving the location of minisum facilities through network modification. *Annals of Operations Research*, *40*(1), 1-16. https://doi.org/10.1007/BF02060467

Berman, O., Ingco, D. I., & Odoni, A. (1994). Improving the location of minimax facilities through network modification. *Networks*, *24*(1), 31-41. https://doi.org/10.1002/net.3230240105

Bonab, B. F., Burkard, R. E., & Alizadeh, B. (2010). Inverse median location problems with variable coordinates. *Central European Journal of Operations Research*, *18*(3), 365-381. https://doi.org/10.1007/s10100-009-0114-2

Brucker, P. (2007). *Scheduling Algorithms* (5th ed.). Springer Verlag, Heidelberg.

Burkard, R. E., Pleschiutschnig, C., & Zhang, J. Z. (2004). Inverse median problems. *Discrete Optimization*, *1*(1), 23-39. https://doi.org/10.1016/j.disopt.2004.03.003

Etemad, R., & Alizadeh, B. (2017). Combinatorial algorithms for reverse selective undesirable center location problems on cycle graphs. *Journal of the Operations Research Society of China*, *5*(1), 347-361. https://doi.org/10.1007/s40305-016-0144-0

Goldman, A. J. (1971). Optimal Center Location in Simple Networks. *Transportation Science*, *5*, 212-221. https://doi.org/10.1287/trsc.5.2.212

Handler, G. Y., & Mirchandani, P. B. (1979). *Location on Networks: Theory and Algorithms*. MIT Press, Cambridge.

Hennes, H. (2005). *Integrated scheduling and location models.* Shaker Verlag, Aachen.

Hennes H., & Hamacher, H. W. (2001). Integrated scheduling and location models: Single machine makespan problem. *Stud. Locat. Anal.*, *16*, 77-90.

Hessler, C. J. (2016). *Scheduling-location algorithms with application in evacuation planning*. Dr. Hut Verlag, Müchen.

Kalsch M. T., & Drezner, Z. (2010). Solving scheduling and location problems in the plane simultaneously. *Computers and Operations Research*, *37*, 256-264. https://doi.org/10.1016/j.cor.2009.04.014

Krumke S. O., & Le, H. M. (2022). 2-approximation algorithm for minmax absolute maximum lateness scheduling-location problem, *Operations Research Letters*, *50*(6), 732-737. https://doi.org/10.1016/j.orl.2022.11.001

Nazari, M., & Fathali, J. (2023). Inverse and Reverse 2-facility Location Problems with Equality Measures on a Network. *Iranian Journal of Mathematical Sciences and Informatics*, *18*(1), 211-225. https://doi.org/10.52547/ijmsi.18.1.211

Sepasian, A. R., & Rahbarnia, F. (2015). An O(nlogn) algorithm for the inverse 1-median problem on trees with variable vertex weights and edge reductions. *Optimization*, *64*(3), 595-602.

Sepasian, A. R., & Tayyebi, J. (2020). Further study on reverse 1-center problem on trees. *Asia-Pacific Journal of Operational Research*, *37*(6), 1-18. https://doi.org/10.1142/S0217595920500347

Tayyebi, J., & Sepasian, A. R. (2023). Reverse 1-centre problem on trees under convex piecewise-

linear cost function. *Optimization*, *72*(3), 843-860. https://doi.org/10.1080/02331934.2021.1995730

Yim, S., Hong, S. P., Park, M. J., & Chung, Y. (2022). Inverse interval scheduling via reduction on a single machine. *European Journal of Operational Research*, *303*(2), 541-549. https://doi.org/10.1016/j.ejor.2022.02.046