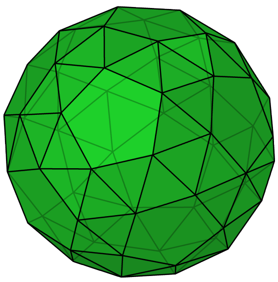
Projects in Mathematics and Applications

COMBINATORIAL OPTIMIZATION WITH DOMATIC NUMBER AND INTEGER PROGRAMMING

Ngày 10 tháng 12 năm 2024

Nguyễn Lê Quốc Bảo [*] [†]Hồ Ngọc Huy
Trịnh Võ Nam Kiệt [‡] [§]Phạm Thị Thu Huế

Mentor: Võ Minh Quân [¶]



[*]Trường THPT Chuyên Lê Hồng Phong, TPHCM
[†]Trường THPT Chuyên Nguyễn Bỉnh Khiêm, Quảng Nam
[‡]Trường THPT Chuyên Lê Quý Đôn, Khánh Hòa
[§]Trường THPT Chuyên Lê Hồng Phong, Nam Định
[¶]Đại học bang Illinois tại Urbana–Champaign

Lời cảm ơn

Lời nói đầu tiên, chúng em xin gửi lời cảm ơn sâu sắc đến Ban tổ chức cũng như các đơn vị tài trợ trại hè Toán học và Ứng dụng PiMA 2021. Dù tình hình dịch bệnh diễn biến phức tạp, trại hè diễn ra trực tuyến nhưng chúng em đã có cơ hội tiếp cận với các bài giảng kiến thức mới cũng như cải thiện các kĩ năng cần thiết như làm việc nhóm, nghiên cứu,... một cách hiệu quả nhất. Đồng thời, PiMA cũng tổ chức các buổi trò chuyện cực kỳ bổ ích với các anh, chị và thầy có kinh nghiệm trong ngành. Thông qua những cơ hội đó, chúng em đã có thêm được nhiều góc nhìn mới đối với việc học môn Toán trong môi trường giáo dục cao hơn, cũng như việc tìm kiếm định hướng cho bản thân trong tương lai. Chúng em muốn gửi lời cảm ơn chân thành đến các anh chị mentor, đặc biệt là anh Minh Quân và Thế Anh, đã luôn sát sao, tận tình hướng dẫn, giúp đỡ nhóm trong quá trình nghiên cứu và hoàn thành dự án này. Chúng mình cũng xin cảm ơn các bạn trại sinh đã tích cực tham gia, góp phần vào thành công của PiMA 2021. Trại hè PiMA 2021 tuy đã kết thúc nhưng những trải nghiệm trong 2 tuần qua sẽ theo chân chúng em trên con đường tương lai sắp tới. Chúc PiMA tiếp tục thành công trong những năm sắp tới. Mặc dù đã có nhiều cố gắng nhưng không thể tránh khỏi sai sót, chúng em mong nhận được sự góp ý của quý bạn đọc để hoàn thiện bản báo cáo này.

Tóm tắt nội dung

Tóm tắt sơ lược nội dung bài báo cáo

Mục lục

1 Giới thiệu chung

1.1 Đặt vấn đề

Khi chúng ta làm việc với các hệ thống phức tạp, ví dụ như mạng lưới máy tính, một trong những thách thức chính là đảm bảo sự kết nối và tối ưu hóa khả năng phân phối tài nguyên và công việc. Trong lý thuyết đồ thị, một khái niệm quan trọng giúp giải quyết vấn đề này là **số domatic**. Một mạng lưới truyền thông (communication network) giữa các thành phố có thể được biểu diễn bằng một đồ thị vô hướng, trong đó các cạnh đại diện cho các liên kết truyền thông và các đỉnh đại diện cho các thành phố. Một nhóm phát (transmitting group) là một tập hợp các thành phố, hoạt động như các trạm phát, có thể truyền tin đến mọi thành phố trong mạng. Theo nghĩa đó, một nhóm phát chính là một tập thống trị (dominating set) trong lý thuyết đồ thị, là một tập hợp các đỉnh D của đồ thị G với tính chất: mọi đỉnh không thuộc D đều có ít nhất một đỉnh kề với nó trong D. Trong thực tế, nếu một mạng lưới truyền thông có từ hai nhóm phát độc lập trở lên, một trong các nhóm phát còn lại vẫn có thể tiếp tục nhiệm vụ khi có một nhóm phát phải ngừng hoạt động do các sự cố kĩ thuật. Vì các nhóm phát độc lập nói trên là các tập thống trị rời nhau của đồ thị, ở dự án này, chúng ta quan tâm đến bài toán sau: cho trước một đồ thị liên thông G, hãy xác định số lượng tối đa các tập thống trị đôi một rời nhau có thể có của nó (số domatic).

1.2 Quy ước toán học

Để đọc báo cáo này, người đọc cần có kiến thức cơ bản về đại số tuyến tính và lý thuyết đồ thị. Trong bài báo cáo này, nếu không nói gì thêm, một đồ thị mặc định được hiểu là một đồ thị đơn, vô hướng.

1.3 Danh pháp tiếng Anh

1. Điều kiện ràng buộc: constraint
2. Miền nghiệm, miền chấp nhận được, miền ràng buộc: feasible region
3. Nghiệm (phương án) chấp nhận được: feasible solution
4. Thuật toán đơn hình: simplex algorithm, simplex method
5. Quy hoạch tuyến tính: linear programming
6. Quy hoạch tuyến tính nguyên: integer linear programming
7. Quy hoạch tuyến tính nhị phân: binary linear programming, linear 0-1 programming
8. Quy hoạch tuyến tính nguyên bộ phận: mixed-integer linear programming
9. Phương pháp mặt phẳng cắt: cutting-plane method
10. Thuật toán phán đoán: heuristic algorithm
11. Phương pháp nhánh cận: branch-and-bound method
12. Phương pháp nhánh cắt: branch-and-cut method
13. Bài toán nới lỏng: LP-relaxation, linear programming relaxation
14. Thuật toán nhánh cắt: branch and cut algorithm
15. Cây tìm kiếm: search tree
16. Nghiệm kỉ lục: incumbent solution
17. Giá trị tối ưu: optimal value
18. Đồ thị: graph
19. Liên thông: connected
20. Bậc nhỏ nhất: minimum degree
21. Lân cận (của một đỉnh hay một tập các đỉnh): neigborhood
22. Ma trận kề: adjacency matrix
23. Tập thống trị: dominating set
24. Phân hoạch domatic: domatic partition
25. Số domatic: domatic number
26. Đồ thị siêu khối: hypercube graph
27. Mạng cảm biến không dây: wireless sensor nnetworks
28. Mạng lưới truyền thông: communication network
29. Phân bố tài nguyên: resource allocation

2 Bài toán quy hoạch tuyến tính nguyên

2.1 Định nghĩa

Bài toán quy hoạch tuyến tính nguyên là trường hợp đặc biệt của bài toán quy hoạch tuyến tính với các biến được giới hạn chỉ nhận các giá trị nguyên. Cụ thể, một bài toán quy hoạch tuyến tính nguyên tổng quát có dạng như sau:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^{n} c_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j \le b_i, \quad i = 1, \ldots, m, \\ & x_j \ge 0, \quad j = 1, \ldots, n., \\ & x_j \in \mathbb{Z}, \quad j = 1, \ldots, n. \end{aligned}$$

Hay bài toán có thể được đưa về dạng ma trận như sau:

$$\begin{aligned} \text{maximize} \quad & c^\top x \\ \text{subject to} \quad & Ax \le b, \\ & x \ge 0, \\ & x \in \mathbb{Z}^n. \end{aligned}$$

Nếu $n^* = n$ thì ta có bài toán nguyên toàn phần (Pure Integer Programming), còn nếu $n^* < n$ thì ràng buộc chỉ còn một số nguyên nhất định, ta gọi bài toán này là quy hoạch nguyên một phần (Mixed Integer Programming). Ở đây ta chỉ xét hàm mục tiêu $f(x)$ và điều kiện $g(x) = \sum_{j=1}^{n} a_{ij} x_j$ là các hàm số tuyến tính. Để bài toán quy hoạch tuyến tính nguyên có nghiệm thì bài toán quy hoạch tuyến tính nới lỏng (bỏ đi điều kiện nguyên) cũng phải có nghiệm (bounded). Tức nghiệm thuộc trong vùng không gian giới hạn bởi các siêu mặt phẳng (hyperlane) cắt nhau trong không gian tạo thành khối đa diện. Xét bài toán:

$$\begin{aligned} \text{maximize} \quad & f(x) = c^T x \\ \text{subject to} \quad & x \in D \end{aligned}$$

trong đó $c \in \mathbb{R}^n\{0\}$ và $D \subset R^n$ là tập hữu hạn các vector $\{x_1, x_2, \ldots, x_n\} \mid x_i \in \mathbb{R}^n$ và $x_{ij}, j = 1, \ldots, n$ chỉ nhận các giá trị nguyên. Do D có hữu hạn phần tử nên bài toán IP sẽ luôn có nghiệm tối ưu, tức tồn tại một vector nghiệm $x* \in D$ sao cho $f(x^*) \ge f(x) \forall x \in D$.

2.2 Khái niệm không gian con

Ý tưởng chính của bài toán IP là ta sẽ phân hoạch vùng không gian nghiệm D thành các vùng nhỏ hơn D_i để giải quyết các bài toán nhỏ hơn trên các tập không gian con. Cách phân hoạch tuân theo phương pháp mặt phẳng cắt (cutting planes). Ta ký hiệu cách phân hoạch là $P := \{D_i \subseteq D \mid i \in I\}$ với I là tập hữu hạn các chữ số.

$$D = \bigcup_{i \in I} D_i \text{ và } D_i \cap D_j \neq \emptyset, \forall i \neq j$$

Ta gọi mỗi $D_i \subseteq D \mid i \in I$ là một tập không gian con của D. Thông thường $I = 2$, tức tạo ra cây nhị phân khi thực hiện nhánh cận, nói cách khác ta đang chia cắt đa giác D thành hai phần. Một phân hoạch $P' := \{D'_j \subseteq D \mid j \in I'\}$ được gọi là mịn hơn hơn phân hoạch P ở trên nếu (1) với mọi $i \in I$ tồn tại ít nhất một $j \in I'$ sao cho $D'_j \subseteq D_i$; (2) tồn tại $i_0 \in I, j_0 \in I'$ sao cho $D'_{j0} \subset D_{i0}$.

2.3 Cận trên và cận dưới

Gọi cận dưới của bài toán IP là $f_\alpha \in \mathbb{R}$ nếu: $f_\alpha \leq f(x^*)$ với $x^* \in \mathbb{R}^n$ là nghiệm là cho hàm mục tiêu $f(\cdot)$ tối ưu. Một điều hiển nhiên là trong quá trình truy tìm nghiệm tối ưu, ta sẽ có được có nghiệm $\bar{x} \in D$ là nghiệm chấp nhận được nên $f(\bar{x}) \leq f(x^*)$; vì vậy ta có thể cập nhật cận dưới $f_\alpha = \lfloor f(\bar{x}) \rfloor$. Cận trên của bài toán IP là $f_\beta \in \mathbb{R}$ nếu $f_\beta \geq f(x^*)$. Thông thường khi giải bài toán IP nới lỏng (Relaxed Linear Programming) thì ta sẽ tìm được $f_\beta = f(\bar{x})$ với $\bar{x} \in \mathbb{R}$ và giá trị cận trên này sẽ được cập nhật khi điều kiện của bài toán được thỏa mãn, tức tìm ra được nghiệm x nguyên $\in \mathbb{R}$.

2.4 Kỹ thuật chia để trị

Như đã đề cập, thay vì cố gắng giải bài toán IP trên tập không gian lớn D, ta sẽ cố gắng giải những bài toán con (LP_i) trên các không gian con D_i theo quy hoạch P:

$$\max\{f(x) = c^T x \mid x \in D_i\}$$

Mục đích của việc phân cắt như vậy nằm ở định lý: nghiệm tối ưu của một bài toán LP luôn đạt được tại ít nhất một đỉnh của đa giác D. Nếu ta chọn cách phân hoạch $P := \{D_i \subseteq D \mid i \in I\}$ là cắt theo các đường giá trị nguyên thì ta sẽ thu các đa giác D_i mới có thể có các cạnh và đỉnh nằm trên các đường giá trị nguyên đó. Lúc này, ta chỉ cần các sử dụng thuật toán đơn giản để giải bài toán LP như thuật toán đơn hình (Simplex) hay bài phương pháp điểm trong (Interior Point Method). Và một điểm mấu chốt là ta cần tìm cách gộp kết quả của các bài toán con để đạt được kết quả cuối cùng. Giả sử, ta tính được kết quả các bài toán con hợp lệ và có được cận trên $f_\beta(D_i)$ của các bài toán con (IP_i) thì kết quả cuối cùng sẽ là:

$$f_\beta^* = \max\{f_\beta(D_i) \mid i \in I\}$$

Còn cận dưới, ban đầu ta sẽ đặt $f_\alpha = -\inf$. Trong quá trình tính toán, nếu tìm được nghiệm x_i tối ưu cho bài toán con (IP_i) bất kỳ thì ta sẽ cập nhật $\alpha = \max\{\alpha, \lfloor f(x^i) \rfloor\}$. Với mọi thuật toán branch and bound ta cần tìm điều kiện dừng, tức loại bỏ nhánh/vùng không gian/tập D_i nếu tập $D_i \subset D$ thỏa mãn ba điều kiện sau: (1) Tập D_i rỗng, (2) Tìm được nghiệm nguyên tối ưu $x_i \in D_i \subset D$ của bài toán con (IP_i), (3) cận trên $f_\beta(D_i) \leq \bar{f}_\beta$ với $\bar{f}_\beta$ là giá trị tối ưu tới thời điểm hiện tại.

3 Phương pháp nhánh và cận

Thuật toán nhánh cận là một thuật toán tổng quát được sử dụng để giải quyết các bài toán tối ưu hóa, đặc biệt là các bài toán quy hoạch tuyến tính nguyên, được đề xuất bởi Land và Doig vào năm 1960 [12]. Ý tưởng của hai ông được quan tâm và phát triển rộng rãi thành một hướng mới của tối ưu rời rạc, gọi là phương pháp nhánh cận. Có rất nhiều thuật toán được ra đời dựa trên phương pháp nhánh và cận để tối ưu các bước đi rẽ nhánh trong cách chọn biến và chọn đỉnh. Vì vậy, nhóm tác giả không thể trình bày hết thuật toán, chỉ trình bày ý tưởng tổng quát và chi tiết thuật toán cổ điển đầu tiên là Land-Doig cho bạn đọc.

3.1 Ý tưởng tổng quát

Một bài toán tối ưu $\mathcal{P}$ (giả sử cực đại hóa) được định nghĩa bởi $\mathcal{P} = (\mathcal{D}, f)$, trong đó $\mathcal{D}$ là miền ràng buộc và $f : \mathcal{D} \to \mathbb{R}$ là hàm mục tiêu. Mục tiêu của $\mathcal{P}$ là tìm một nghiệm tối ưu $x^* \in \mathrm{argmax}_{x \in \mathcal{D}} f(x)$. Các thuật toán nhánh cận xây dựng một cây tìm kiếm $\mathcal{T}$ với gốc là $\mathcal{P}$

và các đỉnh còn lại là các bài toán con. Ở mỗi vòng lặp, thuật toán chọn một tập con $\mathcal{S}$ của $\mathcal{D}$ từ hàng đợi $\mathcal{L}$ gồm các tập con chưa kiểm duyệt. Một nghiệm $\hat{x} \in \mathcal{D}$ được gọi là nghiệm kỉ lục nếu nó cho giá trị tối ưu tốt nhất đến hiện tại. Nếu một nghiệm $\hat{x}' \in \mathcal{S}$ (nghiệm tối ưu của bài toán con $(\mathcal{S}, f)$) có giá trị tốt hơn $\hat{x}$, ta cập nhật lại nghiệm kỉ lục. Ngược lại, tập $\mathcal{S}$ bị loại nếu không chứa nghiệm tốt hơn $\hat{x}$. Nếu không, $\mathcal{S}$ sẽ được chia thành các tập con nhỏ hơn để thêm vào hàng đợi $\mathcal{L}$. Khi không còn tập nào trong $\mathcal{L}$, nghiệm kỉ lục hiện tại được trả về và thuật toán kết thúc.

Algorithm 1: Phương pháp nhánh cận tổng quát

```
Objective function: Khởi tạo x̂
L ← {D}
Result: x̂
while L ≠ ∅ do
    Solve: D_i^{ip} = max{f(x) | x ∈ D_i^{ip}}
    if D_i^{ip} = ∅ then
        flag = 0  and stop
    else
        Chọn một tập S từ hàng đợi L
        if một nghiệm x̂' ∈ {x ∈ S | f(x) > f(x̂)} được tìm thấy then
            x̂ ← x̂'
        end
        if S không thể bị loại bỏ then
            Phân hoạch S thành S_1, , . . . , S_r
            Thêm S_1, . . . , S_r vào L
        end
        L ← L \ {S}
    end
end
```

3.2 Thuật toán Land - Doig

Xét bài toán quy hoạch nguyên tuyến tính (IP):

$$\max\{f(x) = c^T x \mid x \in D_0\}$$

trong đó $D_0 \subset R^n$ được xác định bởi:

$$D_0^{ip} := \{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0 \text{ và nguyên}\}$$
$$A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, c \in \mathbb{R}^n \ \{0\}$$

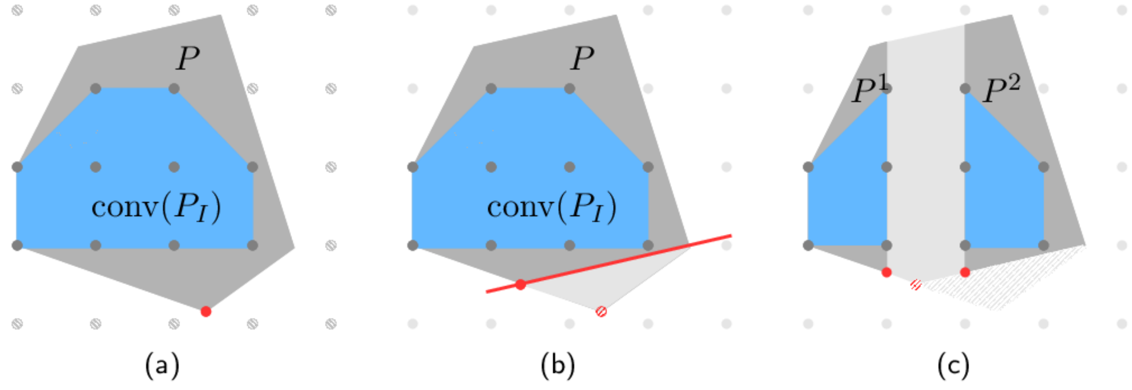
Bước 1: Tìm nghiệm của bài toán quy hoạch tuyến tính nới lỏng LP_0 với:

$$D_0^{lp} := \{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0\}$$

là tập lồi tạo thành khối đa diện có cùng ràng buộc như tập D_0 nhưng bỏ đi tính nguyên của biến x. Việc tìm nghiệm này có thể sử dụng đơn hình hoặc phương pháp điểm trong.

Bước 2: Cập nhật cận trên và cận dưới của bài toán

$$\bar{f}_\beta = f_\beta(D_0) = f_\beta(D_0^{lp})$$



Hình 1: (a) Quy hoạch tuyến tính nới lỏng (b) Mặt phẳng cắt (c) Chia nhánh

Ở bước khởi tạo chạy đầu, mặc dù có thể nghiệm x_0 không nguyên nhưng ta vẫn cập nhật cận trên $\bar{f}_\beta$ vì ta biết rằng đây sẽ là cận trên (có nghiệm không nguyên) lớn nhất cho đến khi kết thúc thuật toán vì $D_0 \subset D_0^{lp}$ và các bài toán con sau $\{D_i \subset D_0 \mid \forall i \in I\{0\}\}$ nên $f_\beta(D_i) \leq f_\beta(D_0^{lp})$ với mọi $i \in I$.

Bước 3: Thực hiện chia nhánh
Quy tắc chia nhánh được áp dụng cho cả nhánh đầu và nhánh dưới. Lưu ý ký hiệu tập con $D_k \subset D$ và ký hiệu tập cha $D = \Omega(D_k)$. Tổng quát: chọn $D_k \in D$ là tập chấp nhận được của bài toán con $(IP)_k$. Các cách chọn tập D_k khác nhau tạo ra các thuật toán branch and bound khác nhau. Thông thường, ta chọn tập D_k có cận trên lớn nhất $\max f_\beta(D_k)$ trong các bài toán con tương ứng với tập con thuộc D để tiếp tục rẽ nhánh. Gọi $x^{(k)} \in \mathbb{R}^n$ là nghiệm tối ưu của bài toán nới lỏng $(LP)_k$ tương ứng của $(IP)_k$. Giả sử $x_j^{(k)}$ là số đầu tiên không nguyên. Phân hoạch D_k thành hai tập như sau:

$$D_{k+1} := \left\{x \in D_k \mid x_j \leq \left\lceil x_j^{(k)} \right\rceil \right\} \quad D_{k+2} := \left\{x \in D_k \mid x_j \geq \left\lfloor x_j^{(k)} \right\rfloor \right\}$$

Thực hiện phép cắt (cutting planes) khối đa diện K thành $k+1$ và $k+2$ bằng cách cập nhật không gian tìm kiếm bằng cách: $D := (D \setminus \{D_k\}) \cup \{D_{k+1}, D_{k+2}\}$. Như vậy, quay trở lại bài toán, hiện tại ta đang có D_0 chính là một D_k. Và vì tập cha $\Omega(D_0)$ của D_0 không có nên dĩ nhiên D_0 là tập có cận trên max trong các tập $D_k \in \Omega(D_0)$. Vì vậy, bây giờ ta chỉ cần chia nhánh D_{0_1} và D_{0_2} từ D_0. Lưu ý, ở trên ta đang chọn $x_j^{(k)}$ là số không nguyên đầu tiên trong nghiệm $x^{(k)} \in \mathbb{R}^n$. Tuy nhiên các cách chọn $x_j^{(k)}$ khác nhau sẽ dẫn đến các thuật toán nhánh cận khác nhau. Một cách chọn như thế ta gọi là một policy $\pi(D_k)$. Ở đây ta sẽ chọn theo chiến lược:

$$\arg\max \left\{\phi(x_j^{(k)}) \mid j = 1, \ldots, n\right\}$$

với $\phi(x_j^{(k)})$ là số phần thập phân của số thực $x_j^{(k)}$. Như vậy sau bước trên ta đã có D_{0_1} và D_{0_2} từ D_0 qua policy π. Lúc này ta thực hiện phép cắt: $D := (D \setminus \{D_0\}) \cup \{D_{0_1}, D_{0_2}\}$.

Bước 4: Tính toán trên các tập con

Ta thực hiện tính toán trên bài toán quy hoạch tuyến tính nới lỏng D_{k1}^{lp} và D_{k2}^{lp}. Với cách thực hiện truy quét như vậy, ta có thể có đến 2^n kết quả cần tính toán (độ phức tạp $O(2n)$). Tuy nhiên ta không cần đi hết đến tận cùng mà có thể đặt ra điều kiện dừng để quay lui (backtracking). Khi thực hiện di chuyển xuống nhánh mới (branching), ta cần xác định điều kiện dừng (bounding). Với $i \in \{1, 2\}$ ta có:

- Bài toán không chấp nhận được, tức $D_{k+i} = \varnothing$. Loại D_{k+i}: $D := D \setminus \{D_{k_i}\}$ và quay lui.

- Tìm được phương án tối ưu $x^{(ki)}$ nguyên, cập nhật cận trên và quay lui:

$$\bar{f}_\beta = \max\left(\bar{f}_\beta, f(x^{(ki)})\right)$$

- Tìm được phương án tối ưu $x^{(ki)}$ không nguyên, cập nhật cận dưới:

$$\bar{f}_\alpha = \max\left(\bar{f}_\alpha, f(\lfloor x^{(ki)} \rfloor)\right)$$

  Nếu mà $f(x^{(ki)}) < \bar{f}_\beta$ thì ta trả về (quay lui) vì nếu đi rẽ nhánh tiếp thì các cận trên của các tập con sau cũng sẽ không vượt qua cận trên tối ưu hiện tại.

Quay trở lại bước 3 thực hiện cho đến khi nào $D = \varnothing$, dừng thuật toán và phương án tối ưu chính là $\bar{f}_\beta$ với nghiệm $x^* \in \mathbb{R}^n$ có phần tử là các số nguyên. Giải thuật được trình bày tóm tắt qua sơ đồ giải thuật sau:

Algorithm 2: Phương pháp nhánh và cận cho quy hoạch tuyến tính nguyên

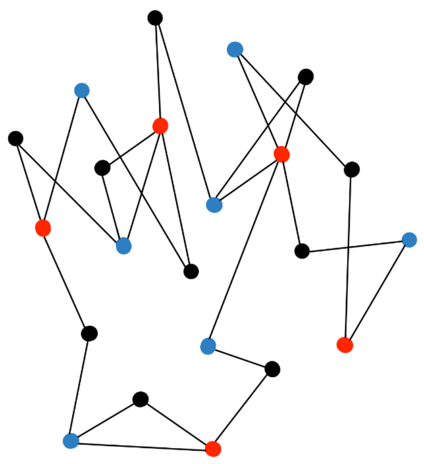
---

Objective function: $\max\{f(x) = c^T x \mid x \in D_0\}$
Subject to: $Ax \leq b$
Result: $f^*_\beta = \max\{f_\beta(D_i) \mid i \in I\}$
$f_\alpha \leftarrow -\infty$; $i \leftarrow 0$; $flag \leftarrow 1$;
$D_0^{ip} := \{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0 \text{ and integer}\}$;
while *flag* $\neq$ *0* **do**
  Solve: $D_i^{ip} = \max\{f(x) \mid x \in D_i^{ip}\}$
  if $D_i^{ip} = \varnothing$ **then**
    flag $= 0$ **and stop**
  else
    Solve: $D_i^{lp} := \{x \in \mathbb{R}^n \mid Ax \leq b, x \geq 0\}$
    Find: $x^{(i)} \in D_i^{lp}$
    if $x^{(i)}$ *is integer* **then**
      $\bar{f}_\beta = \max\left(\bar{f}_\beta, f(x^{(i)})\right)$
    end
    else
      $\bar{f}_\alpha = \max\left(\bar{f}_\alpha, f(\lfloor x^{(i)} \rfloor)\right)$
      if $f(x^{(i)}) < \bar{f}_\beta$ **then**
        flag $= 0$ **and stop**
      end
      Choose candidate: $j \in \arg\max\left\{\phi(x_k^{(i)}) \mid k = 1, \ldots, n\right\}$
      Branching:
      $i \leftarrow i + 1 \rightarrow D_{i+1} := \left\{x \in D_i \mid x_j \leq \lfloor x_j^{(i)} \rfloor\right\}$
      $i \leftarrow i + 2 \rightarrow D_{i+2} := \left\{x \in D_i \mid x_j \geq \lceil x_j^{(i)} \rceil\right\}$
    end
  end
end

---

4 Bài toán số Domatic của đồ thị

4.1 Giới thiệu bài toán



Hình 2: Đồ thị với ba tập thống trị rời nhau (đỏ, xanh, đen)

Trong lý thuyết đồ thị, một tập thống trị của đồ thị $G = (V, E)$ là một tập hợp con các đỉnh $v \in S \subseteq V$ sao cho mọi đỉnh $\forall v \in V$ hoặc là thuộc tập S hoặc là có một đỉnh kề $u \in S$, hoặc ta cũng có thể hiểu bài toán chính là phân tách tất cả các đỉnh $v \in V$ thành các một tập hợp các tập hợp con $S = (S_1, S_2, \ldots, S_m)$ sao cho với $u \in S_i$ thì sẽ có ít nhất tập các cạnh $\{\epsilon_j(u, v) \mid 1 <= j <= m - 1\}$ và $v \in S_j \mid i \neq j$. Số Domatic của một đồ thị G, kí hiệu $d(G)$, là số lượng tập thống trị (dominating sets) lớn nhất của một cách quy hoạch phân vùng. Theo định nghĩa trên, ta dễ dàng xác định hai cận trên và dưới của số Domatic:

$$2 \leq d(G) \leq \min(\delta(G)) + 1$$

trong đó $\delta(G)$ đại diện cho số bậc của các đỉnh $v \in V$ của đồ thị $G(V, E)$. Điều kiện chặn dưới (lower bound) chỉ có thể xác định khi đồ thị phải liên thông, tức là $\min \delta(G) \geq 1$

4.2 Một số kết quả lý thuyết của số domatic

Đầu tiên ta có kết quả sau về chặn trên của số domatic.

Định lý 4.1. *Với mọi đồ thị G, ta có* $\operatorname{dom}(G) \leq \delta(G) + 1$.

Chứng minh. Gọi v_0 là đỉnh có bậc nhỏ nhất trong G và $v_1, v_2, \ldots, v_\delta$ là các đỉnh kề với v_0.. Xét $S = \{v_0, v_1, \ldots, v_\delta\}$, giả sử có một tập thống trị nào đó, gọi là V', sao cho $V' \cap S = \varnothing$. Khi đó, xét đỉnh u bất kì trong V' thì v_0 không kề u nên v_0 phải kề một đỉnh nào đó nằm trong cùng tập thống trị với u (theo định nghĩa về tập thống trị).

Mà v_0 chỉ kề với $v_1, \ldots, v_\delta$ nên phải tồn tại $i \in \{1, 2, \ldots, \delta\}$ để v_i nằm cùng tập thống trị với u. Khi đó, V' chứa v_i (mâu thuẫn với điều giả sử) $\rightarrow$ Một tập thống trị chứa ít nhất 1 đỉnh trong $S \rightarrow \operatorname{dom}(G) \leq |S| = \delta + 1$. □

Định lý 4.2. *Nếu G liên thông thì* $\operatorname{dom}(G) \geq 2$ *(weak 2-coloring).*

Chứng minh. Lấy 1 đỉnh s bất kì trong G và duyệt BFS từ s. Định nghĩa khoảng cách của 2 đỉnh u, v trên đồ thị là số cạnh ít nhất cần duyệt qua để đi từ u đến v.
Ta phân hoạch V thành 2 tập:

- V_1 là tập các đỉnh có khoảng cách đến s lẻ.
- V_2 là tập các đỉnh có khoảng cách đến s chẵn.

Khi đó, V_1 và V_2 là 2 tập thống trị nên $\mathrm{dom}(G) \geq 2$. □

4.3 Xây dựng mô hình quy hoạch tuyến tính nguyên cho bài toán số domatic

Một tập hợp D gồm các đỉnh trong đồ thị G được gọi là một tập thống trị nếu mỗi đỉnh của G hoặc thuộc D hoặc có một đỉnh kề trong D. Một phân hoạch domatic của đồ thị G là một phân hoạch của tập đỉnh $V(G)$ thành các tập thống trị. Số lượng tối đa các tập hợp trong một phân hoạch domatic của đồ thị G được gọi là số domatic của G, kí hiệu là $\mathrm{dom}(G)$.

4.4 Mô hình hóa bài toán

4.4.1 Mô hình 1

Cho k là một số nguyên dương. Ta hỏi rằng liệu có tồn tại cách phân hoạch tập đỉnh của một đồ thị cho trước G thành k tập thống trị đôi một rời nhau hay không. Câu hỏi có thể được mô hình hóa thành một bài toán quy hoạch tuyến tính nguyên như sau.
Với $N = |V(G)|$, ta xét kN biến nhị phân $x_{u,i}$, với $u \in V(G)$ và $i \in [k]$, nhận lần lượt các giá trị 1 và 0 nếu đỉnh u được phân hoạch vào tập thứ i và ngược lại. Vì đôi một các tập là không giao nhau, mỗi đỉnh chỉ nên thuộc duy nhất một trong k tập. Do đó, ta có ràng buộc tuyến tính sau đúng với mọi $u \in V(G)$,

$$\sum_{i=1}^{k} x_{u,i} = 1.$$

Vì mỗi tập trong k tập trên là một tập thống trị, ta lại phải có ràng buộc tuyến tính sau đúng với mọi $i \in [k]$ và mọi $u \in V(G)$,

$$x_{u,i} + \sum_{v \in N(u)} x_{v,i} \geq 1.$$

Vậy ta xét bài toán sau:

$$\begin{aligned}
\text{maximize} \quad & 1 \\
\text{subject to} \quad & \sum_{i=1}^{k} x_{u,i} = 1, && u \in V(G), \\
& x_{u,i} + \sum_{v \in N(u)} x_{v,i} \geq 1, && i = 1, \dots, k, \\
& x_{u,i} \in \{0, 1\}, && u \in V(G),\ i = 1, \dots, k.
\end{aligned}$$

4.4.2 Mô hình 2

Trong phần này, ta xét bài toán xác định dom(G), số lượng tối đa các tập thống trị đôi một rời nhau có thể có của một graph G cho trước. Trước hết ta có kết quả sau. Xét $k := \delta(G) + 1$ tập $V_1, \ldots, V_k$. Đặt y_i là biến nhị phân nhận giá trị 1 nếu tập V_i được chọn trong phân hoạch cần tìm, và nhận giá trị 0 trong trường hợp còn lại. Hàm mục tiêu của bài toán được xác định bởi $h = \sum_{i=1}^{k} y_i$. Với $N = |V(G)|$, ta xét kN biến nhị phân $x_{u,i}$, với $u \in V(G)$ và $i \in [k]$, nhận lần lượt các giá trị 1 và 0 nếu đỉnh u được phân hoạch vào tập thứ i và ngược lại. Vì đôi một các tập là không giao nhau, mỗi đỉnh chỉ nên thuộc duy nhất một trong k tập (có thể có tập không chứa đỉnh nào). Do đó, ta có ràng buộc tuyến tính sau đúng với mọi $u \in V(G)$,

$$\sum_{i=1}^{k} x_{u,i} = 1.$$

Vì mỗi tập trong k tập trên là một tập thống trị, ta lại phải có ràng buộc tuyến tính sau đúng với mọi $i \in [k]$ và mọi $u \in V(G)$,

$$x_{u,i} + \sum_{v \in N(u)} x_{v,i} - y_i \geq 0.$$

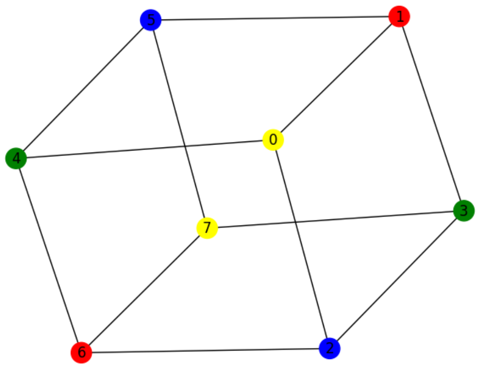
Vậy ta xét bài toán sau:

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{k} y_i \\
\text{subject to} \quad & \sum_{i=1}^{k} x_{u,i} = 1, && u \in V(G), \\
& x_{u,i} + \sum_{v \in N(u)} x_{v,i} - y_i \geq 0, && u \in V(G),\ i = 1, \ldots, k, \\
& x_{u,i} \in \{0, 1\}, && u \in V(G),\ i = 1, \ldots, k, \\
& y_i \in \{0, 1\}, && i = 1, \ldots, k.
\end{aligned}$$

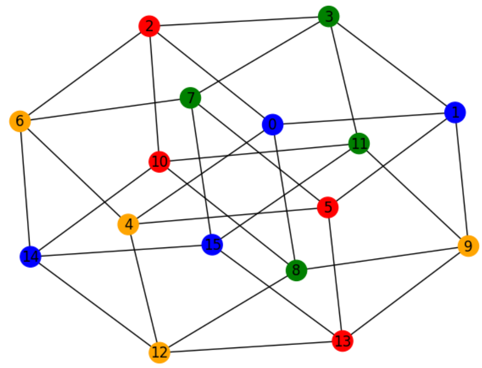
4.5 Cài đặt và thực nghiệm

Nhóm đã thực hiện bài toán tìm số domatic, không phải trên một đồ thị thông thường, mà trên một siêu khối lập phương (hypercube). Trong hình học, hypercube là một đối tượng hình học n chiều, mở rộng từ hình vuông ($n = 2$) hoặc khối lập phương ($n = 3$). Trường hợp đặc biệt với $n = 4$ được gọi là tesseract. Đây là một hình khối đóng, đặc, và lồi (convex), với đặc điểm bao gồm các đoạn thẳng song song, được sắp xếp theo từng chiều không gian, vuông góc với nhau và có cùng độ dài.

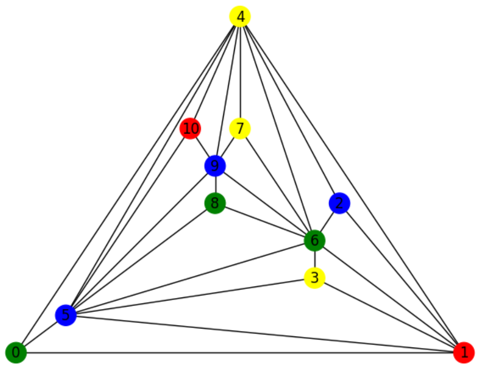
Giả sử mỗi đỉnh của siêu khối lập phương được xem là một đỉnh trong đồ thị, ta có thể áp dụng mô hình bài toán số domatic để phân chia màu sắc cho các đỉnh của siêu khối này, tạo ra một cách tiếp cận độc đáo và thách thức trong việc phân tích đồ thị.



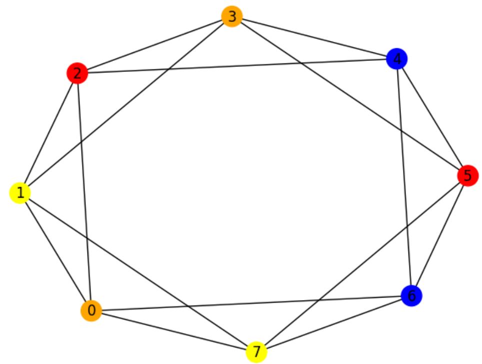
Hình 3:



Hình 4:



Hình 5:



Hình 6:

Tài liệu

[1] E. Balas. An additive algorithm for solving linear programs with zero-one variables. *Operations Research*, 13(4):517–546, August 1965.

[2] L. Caccetta and S. P. Hill. Branch and cut methods for network optimization. *Mathematical and Computer Modelling*, 33(4–5):517–532, February 2001.

[3] E. Cockayne and S. Hedetniemi. Optimal domination in graphs. *IEEE Transactions on Circuits and Systems*, 22(11):855–857, November 1975.

[4] E. J. Cockayne and S. T. Hedetniemi. Towards a theory of domination in graphs. *Networks*, 7(3):247–261, September 1977.

[5] H. Crowder, E. L. Johnson, and M. Padberg. Solving large-scale zero-one linear programming problems. *Operations Research*, 31(5):803–834, October 1983.

[6] M V Devyaterikova and A A Kolokolov. On the stability of some integer programming algorithms. *Oper. Res. Lett.*, 34(2):149–154, March 2006.

[7] Satoshi Fujita, Masafumi Yamashita, and Tiko Kameda. A study on r-configurations—a resource assignment problem on graphs. *SIAM Journal on Discrete Mathematics*, 13(2):227–254, January 2000.

[8] J Glover, Vinh Quan, and Saeed Zolfaghari. Some new perspectives for solving 0–1 integer programming problems using balas method. *Computational Management Science*, 18:177–193, 2021.

[9] R. E. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Society*, 64(5):275–278, 1958.

[10] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024.

[11] Miroslav Karamanov. *Branch and cut: an empirical study*. PhD thesis, Carnegie Mellon University, 2006.

[12] A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497, July 1960.

[13] T. Moscibroda and R. Wattenhofer. Maximizing the lifetime of dominating sets. In *19th IEEE International Parallel and Distributed Processing Symposium*. IEEE.

[14] O. Ore. *Theory of Graphs*. American Mathematical Society, December 1962.

[15] M. Padberg and G. Rinaldi. Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters*, 6(1):1–7, March 1987.

[16] Leonardo Taccari. Integer programming formulations for the elementary shortest path problem. *European Journal of Operational Research*, 252(1):122–130, July 2016.

[17] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, Andrew R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[18] J. Yu, Q. Zhang, D. Yu, C. Chen, and G. Wang. Domatic partition in homogeneous wireless sensor networks. *Journal of Network and Computer Applications*, 37:186–193, January 2014.

[19] B. Zelinka. Domatic numbers of cube graphs. *Mathematica Slovaca*, 32(2):117–119, February 1982.

[20] B. Zelinka. Domatic numbers of cube graphs. *Mathematica Slovaca*, 32(2):117–119, 1982.