ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH

MÔ HÌNH HOÁ TOÁN HỌC

Xử lý tối ưu bài toán

Hospitals & Residents trên Matlab

GVHD: Lê Hồng Trang
SV: Chìu Tuấn Bình - 1510221
Mai Đức Tú - 1513924
Phồng Quang Tuấn - 1513865
Lê Duy Hiển - 1511057
Nguyễn Đỗ Đức Anh - 1510062

TP. HỒ CHÍ MINH, THÁNG 4/2017

Mục lục

Bài báo cáo này trình bày những nghiên cứu của nhóm về bài toán Hospitals & Residents bao gồm: quá trình tìm hiểu bài toán - ứng dụng, tìm hiểu lý thuyết - phần mềm dùng để xử lý tính toán, phân tích hướng tiếp cận - tìm hiểu giải thuật, hiện thực giải thuật trên các ứng dụng. Bài báo cáo được gửi kèm: file nhật ký - lưu lại tiến độ làm việc của nhóm, file mã nguồn báo cáo .tex, file chứa source code Matlab .m.

1 Giới thiệu bài toán [1]

1.1 Mô tả bài toán

Cho một tập các bệnh viện và một tập người dân muốn đăng ký khám bệnh ở một trong số bệnh viện đã cho. Bài toán Hospitals/Residents (HR) tìm kiếm một cách sắp xếp danh sách khám bệnh của người dân với các bệnh viện sao cho đạt được sự tốt nhất theo mong muốn của người dân cũng như các ràng buộc và ưu tiên của bệnh viện. Một phát biểu hình thức của bài toán được cho như dưới đây.

Một minh dụ I của bài toán HR bao gồm một tập $R = \{r_1, r_2, ..., r_n\}$ các người dân và một tập $H = \{h_1, h_2, ..., h_m\}$ các bệnh viện. Khả năng phục vụ mỗi bệnh viện $h_j \in H$ là có hạn, ký hiệu bởi c_j, với $j = 1, 2, ..., m$. Mỗi người dân $r_i \in R$ sắp xếp thứ tự ưu tiên các bệnh viện theo một danh sách. Một cặp $\{r_i, h_j\} \in HxH$ được gọi là chấp nhận được nếu r_i xếp h_j trong danh sách ưu tiên của mình, khi này ta có hiểu rằng r_i đang tìm kiếm một bệnh viện. Tương tự, mỗi bệnh viện $h_j \in H$ cũng có một danh sách ưu tiên chấp nhận, trong đó bệnh viện sắp thứ tự người dân đang tìm kiếm bệnh viện. Cho $(x, y, z) \in R \cup H$, x được gọi ưu tiên y hơn z nếu y, z thuộc danh sách ưu tiên của x và y được xuất hiện trước z trong danh sách đó.

Đặt A là tập các cặp đôi chấp nhận được trong I, và $L = \|A\|$. Một phép gán M được định nghĩa là một tập con của A. Nếu $(r_i, h_j), r_i$ được gọi là được gán với h_j và ngược lại h_j cũng được gán với r_i. Với mỗi $q \in R \cup H$, ký hiệu $M(q)$ là tập tất cả các phép gán của q trong M. Nếu $r_i \in R$ và $M(r_i) \neq \emptyset, r_i$ được gọi là được gán, ngược lại thì gọi là không được gán. Tương tự, ta định nghĩa một $h_j \in H$ được đăng ký, đăng ký đầy, hoặc quá tải nếu tương ứng $\|M(h_j)\|$ nhỏ hơn, bằng, hoặc lớn hơn c_j, với $j = 1, 2, ...m$.

Một ghép cặp (matching) M là một phép gán sao cho không có người dân nào được gán cho một bệnh viện không chấp nhận người đó, và mỗi người dân chỉ được gán nhiều nhất là một bệnh viện, và không có bệnh viện nào là quá tải.

Một cặp $(r_i, h_j) \in A \setminus M$ hạn chế (block) một ghép cặp M, được gọi là cặp hạn chế (blocking pair) với M, nếu: (i) r_i không được gán hoặc ưu tiên chọn h_j trong $M(r_i)$); (ii) h_j được đăng ký hoặc ưu tiên r_i với tối thiểu một thành viên của $M(h_j)$, hoặc cả hai.

Một ghép cặp M được gọi là ổn định (stable) nếu nó không chứa một cặp hạn chế nào cả.

1.2 Yêu cầu

Cho một minh dụ I của HR, tìm một ghép cặp ổn định của I.

1.3 Ứng dụng

Bài toán HR có nhiêu ứng dụng trong thực tế, xuất hiện trong ngữ cảnh cần xây dựng các lược đồ ghép cặp tự động giữa các phần tử giữa một tập ứng viên và một tập đối tượng nào đó, ví dụ như sắp xếp sinh viên thực tập với các công ty, phân công đồ án giữa sinh viên và giáo viên, hỗ trợ tuyển sinh trong các trường học, bài toán lập lịch sử dụng tài nguyên hệ thống,... và nhiều bài toán kinh tế (phát biểu dưới dạng lý thuyết trò chơi, ví dụ như lý thuyết thiết kế cơ chế (mehanism design).

Một trong số các dự án tiêu biểu trong thực tế của mô hình là bài toán "National Resident Matching Program", có ở nhiều nước như Mỹ, Canada, Scotland, hay Nhật Bản.

2 Cơ sở lý thuyết

2.1 Bài toán Quy hoạch tuyến tính

Bài toán quy hoạch tuyến tính là bài toán tìm biến (hoặc phương án) thoả mãn các ràng buộc sao cho làm hàm mục tiêu đạt cực đại hoặc cực tiểu. Với cả hàm mục tiêu và các ràng buộc đều tuyến tính theo biến.

Nhận xét, $max\left(z\right) = -min\left(-z\right)$.

Có ba cách dùng để phát biểu bài toán quy hoạch tuyến tính:

- Dạng Tổng quát:

$$\begin{aligned} f\left(x\right) = \sum_{j=1}^{n} c_j x_j \to min \\ x_j \begin{bmatrix} \leq \\ \geq \end{bmatrix} 0 \\ \sum_{j=1}^{n} a_i j \begin{bmatrix} \leq \\ \geq \\ = \end{bmatrix} b_i, i \in I_k, k = 1, 2, 3 \end{aligned}$$

- Dạng Chính tắc:

$$\begin{aligned} f\left(x\right) = sum_{j=1}^{n} c_j x_j \to min \\ \sum_{j=1}^{n} = b_j, i = 1, ...m \\ x_j \geq 0, j = 1, 2, ..., n \end{aligned}$$

- Dạng ma trận chính tắc

$$\begin{aligned} f\left(x\right) = \sum_{j=1}^{n} c_j x_j \to min \\ Ax = b \\ x \geq 0 \end{aligned}$$

Trong đó, véc tơ x thoả các ràng buộc được gọi là phương án. Phương án là hàm mục tiêu $f\left(x\right)$ đạt giá trị cực trị theo yêu cầu gọi là phương án tối ưu. Giải quy hoạch tuyến tính là tìm phương án tối ưu của bài toán

2.2 Phương pháp đơn hình

Phương pháp đơn hình do G.B;Dantzig đề xuất năm 1947 cho đến hiện nay vẫn là phương pháp được sử dụng nhiều nhất trong việc giải các bài toán qui hoạch tuyến tính.

Đối với các bài toán cỡ lớn (có thể đến hàng nghìn biến và hàng trăm ràng buộc) phải dùng đến máy tính, phương pháp đơn hình cũng đã được kiểm nghiệm qua mấy chục năm áp dụng là rất hiệu quả, với thời gian tính toán khá ngắn.

2.2.1 Cơ sở của phương pháp đơn hình

Phương pháp đơn hình giải bài toán QHTT dựa trên hai tính chất quan trọng sau đây của bài toán QHTT:

- Nếu bài toán quy hoạch tuyến tính chính tắc có phương án tối ưu thì cũng có phương án cực biên tối ưu, nghĩa là có ít nhất một đỉnh của miền ràng buộc là lời giải của bài toán.
- Mỗi điểm cực tiểu địa phương của hàm tuyến tính trên miền ràng buộc D (một tập hợp lồi) là một điểm cực tiểu tuyệt đối.

Phương pháp đơn hình bắt đầu từ một phương án cực biên nào đó (tuỳ ý) của bài toán (tức là một đỉnh của miền ràng buộc). Tiếp đó kiểm tra xem phương án hiện có đã phải là phương án tối ưu hay chưa, bằng cách so sánh giá trị hàm mục tiêu tại đỉnh đó với giá trị hàm mục tiêu tại các đỉnh kề với nó. Nếu đúng thì dừng quá trình tính toán. Trái lại, phương pháp sẽ cho cách tìm một phương án cực biên mới tốt hơn.

2.2.2 Thuật toán đơn hình

Quy trình dưới đây nêu tóm tắt các bước để giải một bài toán QHTT bằng phương pháp đơn hình bằng tay.

- Bước chuẩn bị: Đưa điều kiện về dạng chính tắc.
- Bước 1: Xác định phương án cơ sở xuất phát, chỉ ra các biến và hệ số cơ sở
- Bước 2: Lập bảng đơn hình, tính giá trị hàm mục tiêu và các ước lượng δ
- Bước 3: Kiểm tra điều kiện tối ưu
- Bước 4: Nếu nghiệm vẫn chưa tối ưu ta thay đổi ma trận cơ cơ sở để tìm được nghiệm tốt hơn.

3 Phần mềm tính toán

Để xử lý những bài toán QHTT ở quy mô lớn bằng phương pháp đơn hình là một công việc tốn rất nhiều tài nguyên, dễ sai sót và gần như không thể thực hiện được chỉ với các công cụ thô sơ như giấy bút và máy tính cầm tay. Để đáp ứng nhu cầu xử lý tính toán đó, trên thế giới đã có rất nhiều công cụ mạnh được nghiên cứu và ra mắt. Trong số đó, phần mềm tính toán nâng cao Matlab cùng với bộ công cụ hỗ trợ cvx giúp xử lý dữ liệu cho thuật toán đơn hình cũng như các solver đã được thầy Lê Hồng Trang hướng dẫn để giúp nhóm hoàn thành bài toán QHTT Hospitals/Residents này.

3.1 Môi trường MATLAB

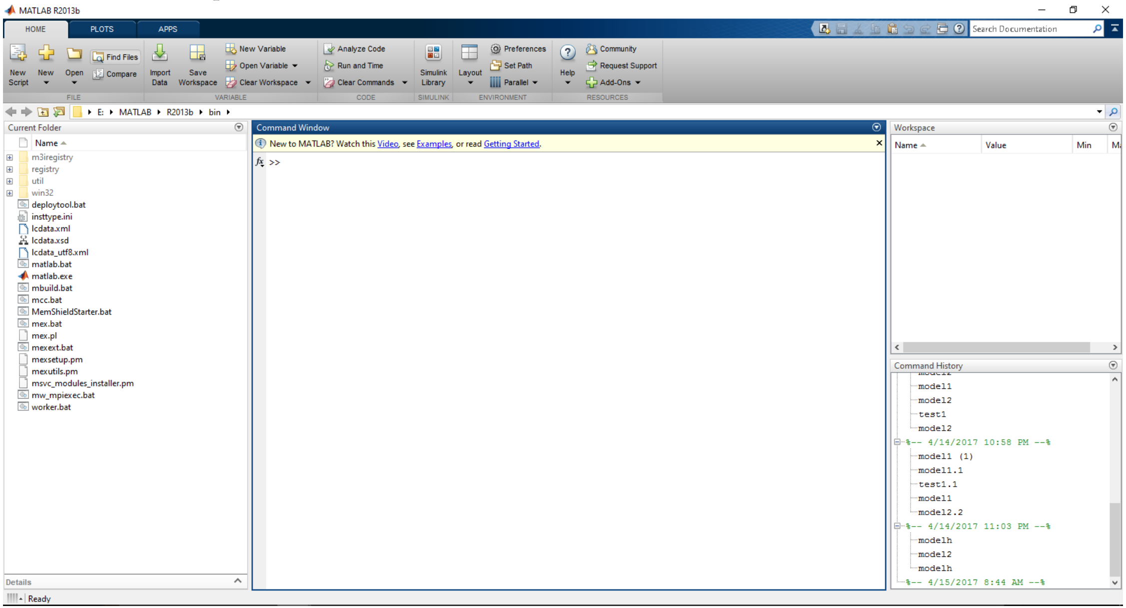
MATLAB là viết tắt của Matrix Laboratory, là một phần mềm toán học của hãng Mathwork để lập trình, tính toán số và có tính trực quan rất cao.

MATLAB làm việc chủ yếu với ma trận. Ma trận cỡ mxn là bảng chữ nhật gồm mxn số được xếp thành m hàng và n cột. MATLAB có thể làm việc với nhiều kiểu dữ liệu khác nhau. Với chuỗi kí tự MATLAB cũng xem là một dãy các ký tự hay dãy mã số của các ký tự.

MATLAB cho phép tính toán số với ma trận, vẽ đồ thị hàm số hay biểu đồ thông tin, thực hiện thuật toán, tạo các giao diện người dùng và liên kết với những chương trình máy tính viết trên nhiều ngôn ngữ lập trình khác. MATLAB giúp đơn giản hóa việc giải quyết các bài toán tính toán kĩ thuật so với các ngôn ngữ lập trình truyền thống như C, C++, và Fortran.

Hiện nay, MATLAB có đến hàng ngàn lệnh và hàm tiện ích. Ngoài các hàm cài sẵn trong chính ngôn ngữ, MATLAB còn có các lệnh và hàm ứng dụng chuyên biệt trong các Toolbox, đểmở rộng môi trường MATLAB nhằm giải quyết các bài toán thuộc các phạm trù riêng. Các Toolbox khá quan trọng và tiện ích cho người dùng như toán sơ cấp, xử lý tín hiệu số, xử lý ảnh,xử lý âm thanh, ma trận thưa, logic mờ,. . .

Giao diện của phần mềm MATLAB:



3.2 Công cụ CVX [2]

CVX là một hệ thống mô hình hoá dùng để xây dựng và giải các bài toán quy hoạch lồi (disciplined convex programs). CVX hỗ trợ nhiều chuẩn dạng bài toán khác nhau bao gồm: linear and quadratic programs ($LPs \setminus QPs$), second-order cone programs (SOCPs), semidefinite programs (SDPs).

CVX được hiện thực trên môi trường MATLAB và đã chuyển MATLAB thành ngôn ngữ giải quyết bài toán tối ưu một cách hiệu quả. CVX sử dụng các hàm và toán tử phổ biến của MATLAB, người dùng có thể thoải mái kết hợp các câu lệnh của MATLAB và dùng CVX như một thư viện đã có các hàm sẵn.

Các phiên bản cũ của CVX hỗ trợ các solver miễn phí gồm SeDuMi [Stu99] và SDPT3 [sTTO3]. Những solver này luôn có trong các bản phân phối của CVX. Bắt đầu từ phiên bản 2.0, CVX hỗ trợ 2 solvers thương mại hoá là Gurobi và MOSEK. Để sử dụng các solver thương mại này, người dùng cần phải có bản quyền CVX Professional hoặc phải có Academic license. Để có được Academic license từ CVX khá đơn giản, chỉ cần yêu cầu từ trang web của công cụ bằng email sinh viên.

4 Làm rõ các điều kiện của bài toán

Theo đề ra thì ta có A gọi là tập tất cả các cặp chấp nhận được (acceptable) trong I. Mà theo định nghĩa

1. Cặp (r_i, h_j) chỉ được coi là chấp nhận được khi mà h_j thuộc danh sách ưu tiên của r_i. Điều đó có nghĩa là trong A sẽ không chứa các cặp (r_x, h_y) mà trong danh sách của r_x không chứa h_y.

2. Bệnh viện h_j có một danh sách ưu tiên dành cho những người dân đang tìm kiếm bệnh viện, mà người dân tìm kiếm bệnh viện được định nghĩa là người dân có xếp h_j vào danh sách ưu tiên của mình. Theo lẽ thông thường, bệnh viện sẽ chỉ xếp danh sách những người dân có nhu cầu vào bệnh viện đó, và trong bài toán này ta cũng chỉ xét trường hợp này mà thôi

Để thể hiện rõ, cụ thể hơn yêu cầu đề bài, ta chuyển dữ liệu đầu vào về dạng biểu diễn toán học

- Theo đề ra ta có R = $\{r_1, r_2, ..., r_n\}$ là tập các người dân, H = $\{h_1, h_2, ..., h_m\}$ là tập các bệnh viện, C = $\{c_1, c_2, ..., c_m\}$ là tập các sức chứa (khả năng tiếp nhận) của các bệnh viện, các con số này không liên quan gì tới độ lớn danh sách ưu tiên tiếp nhận của bệnh viện.

- Đặt

$$D = \begin{bmatrix} d_{11} & d_{12} & \dots & d_{1m} \\ d_{21} & d_{22} & \dots & d_{2m} \\ \dots & \dots & \dots & \dots \\ d_{n1} & d_{n2} & \dots & d_{nm} \end{bmatrix}$$

  Với d_{ij} là một trọng số thể hiện độ ưu tiên của **người dân** r_i **cho bệnh viện** h_j. Với độ ưu tiên càng cao, ta có số d càng nhỏ, với độ ưu tiên cao nhất là 1, giảm dần đến cuối danh sách ưu tiên và độ ưu của bệnh viện không xếp trong danh sách ưu tiên là m+n+1

- Đặt

$$B = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1m} \\ b_{21} & b_{22} & \dots & b_{2m} \\ \dots & \dots & \dots & \dots \\ b_{n1} & b_{n2} & \dots & b_{nm} \end{bmatrix}$$

  Với b_{ij} là trọng số thể hiện độ ưu tiên của **bệnh viện** h_j **cho người dân** r_i. Độ ưu tiên càng cao, b càng nhỏ, độ ưu tiên cao nhất là 1, giảm dần đến cuối danh sách ưu tiên tiếp nhận, và tất cả các người dân không xếp trong danh sách ưu tiên tiếp nhận sẽ có trọng số ưu tiên m+n+1

- $y \underset{x}{<} z$ tức là x có cả y và z trong danh sách ưu tiên của mình, và y có thứ tự ưu tiên cao hơn z (vì nhóm mình đánh trọng số ưu tiên là từ nhỏ đến lớn, nên mình viết kiểu này cho đỡ lẫn).

- Đặt

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{bmatrix}$$

$x_i, j = 1$ nếu (r_i, h_j) là 1 cặp được ghép với nhau, còn nếu $x_i, j = 0$ thì (r_i, h_j) không được ghép với nhau.

- Nhưng vì ta đã có đánh trọng số ưu tiên cho từng cặp rồi, nên cũng với ý câu trên, nếu r_x có cả h_y, h_z trong danh sách ưu tiên của mình, và h_y có thứ tự ưu tiên cao hơn h_z, thì ta biểu diễn thành $d_{x,y} < d_{x,z}$.
Tương tự, nếu h_x có cả r_y, r_z trong danh sách ưu tiên của mình, và r_y có thứ tự ưu tiên cao hơn r_z, thì ta biểu diễn thành $b_{y,x} < b_{z,x}$. Để tiện biểu diễn sự so sánh này trong mô hình cho đơn giản, ta định nghĩa hàm P(x,y) như sau:

$$P(x,y) = \begin{cases} 1, x < y \\ 0, x \geq y \end{cases}$$

- Khi đó, ta biểu diễn r_i thích h_a hơn h_b là $P(d_{ia}, d_{ib}) = 1$. Tương tự với bệnh viện, nếu h_j thích r_a hơn r_b là $P(b_{aj}, b_{bj}) = 1$

5 Mô hình đề xuất

Bài toán Hospitals/Residents thuộc dạng bài toán ghép cặp bền vững (stable matching). Bài toán dạng này với cách tiếp cận quy hoạch nguyên thì chỉ cần đảm bảo thỏa mãn các điều kiện để kết quả là ghép cặp bền vững (stable). Ta xét trường hợp ưu tiên ngặt trước

5.1 Điều kiện để kết quả là ghép cặp

Một phép gán được coi là một ghép cặp nếu thỏa mãn các điều kiện sau:

- Không có người dân nào được gán cho một bệnh viện không chấp nhận người đó. Điều kiện này có thể được đảm bảo bằng cách chỉ xét các cặp nằm trong tập A.
- Mỗi người dân chỉ được gán nhiều nhất là một bệnh viện. Điều kiện này có thể được thể hiện bằng công thức:

$$\sum_{j}^{m} x_{ij} \leq 1 \text{ với i =1,2,...,n}$$

- Không có bệnh viện nào là quá tải.

$$\sum_{i}^{n} x_{ij} \leq c_j \text{ với j = 1,2,...,m}$$

5.2 Điều kiện để ghép cặp ổn định

Để ghép cặp là ổn đinh, trước hết nó phải là một ghép cặp, đồng thời nó phải không để xảy ra một cặp chặn nào.

5.2.1 Định nghĩa cặp chặn

Một cặp chặn (blocking pair) xảy ra khi có một cặp (r_u, h_v) nào đó mà cả 3 đk sau thỏa
Cặp đó không được ghép với nhau

$$x_{uv} = 0$$

Bệnh viện h_v chưa đăng ký đầy, hoặc trong số những người ghép với bệnh viện này có tồn tại ít nhất 1 người có độ ưu tiên thấp hơn r_u, hoặc cả 2

$$\sum_{i=1}^{n} x_{iv} P(b_{iv}, b_{uv}) < c_v$$

Người dân r_u chưa được xếp với bệnh viện nào cả, hoặc nếu có thì người dân này không thích bệnh viện đó bằng h_v

$$\sum_{j=1}^{m} x_{uj} P(d_{uj}, d_{uv}) = 0$$

5.2.2 Để cặp chặn không xảy ra

Như đã định nghĩa, cặp chặn chỉ tồn tại ra khi cả 3 điều kiện nói trên đều thỏa với một cặp nào đó. Điều đó có nghĩa là để không tồn tại cặp chặn nào, với mỗi cặp chấp nhận được phải vi phạm ít nhất 1 trong 3 điều kiện trên. Để vi phạm ít nhất 1 trong 3 điều kiện đã nêu ở phần trên, ít nhất 1 trong 3 điều kiện sau phải được thỏa: Cặp đó được ghép với nhau

$$x_{uv} = 1$$

Bệnh viện h_v đăng ký đầy với tất cả những người trong danh sách ghép cặp đều có độ ưu tiên cao hơn r_u

$$\sum_{i=1}^{n} x_{iv} P(b_{iv}, b_{uv}) = c_v$$

tương đương

$$\sum_{i=1}^{n} x_{iv} P(b_{iv}, b_{uv}) \frac{1}{c_v} = 1$$

Người dân r_u đã được ghép với một bệnh viện mà người đó thích hơn h_u

$$\sum_{j=1}^{m} x_{uj} P(d_{uj}, d_{uv}) = 1$$

Tổng hợp 3 điều kiện nói trên, ta có công thức đảm bảo ghép cặp đã cho là ghép cặp ổn đinh là :

$$\sum_{j=1}^{m} P(d_{u,j}, d_{u,v}) x_{u,j} + \sum_{i=1}^{n} P(b_{i,v}, b_{u,v}) x_{i,v} \frac{1}{c_v} + x_{u,v} \geq 1, \qquad \forall (r_u, h_v) \in A$$

5.3 Mô hình đề xuất cho trường hợp ưu tiên ngặt

Như đã phân tích, mô hình lập được chỉ cần thỏa mãn điều kiện ghép cặp ổn định, hàm mục tiêu là không cần thiết. Theo tinh thần đó, mô hình lập được đối là:

$$
\begin{aligned}
\max_{x_i} \quad & 0^T X && (1)\\
s.t \quad & \sum_{i=1}^{n} x_{ij} \le c_j, \text{ for j = 0...m} && (2)\\
& \sum_{j=1}^{m} x_{ij} \le 1, \text{ for i = 0...n} && (3)\\
& \sum_{j=1}^{m} P(d_{uj}, d_{uv})x_{uj} + \sum_{i=1}^{n} P(b_{i,v}, b_{u,v})x_{iv}\frac{1}{c_v} + x_{uv} \ge 1, \qquad \forall(r_u, h_v) \in A && (4)\\
& x_{ij} = \{0, 1\} && (5)
\end{aligned}
$$

5.4 Mô hình đề xuất cho trường hợp ưu tiên không ngặt

Đối với trường hợp ưu tiên không ngặt, ta chỉnh sửa hàm P(x,y) cho nhận cả giá trị bằng, đồng thời chỉnh sửa hàm điều kiện ổn định (4) cho phù hợp với hàm P đã chỉnh sửa :

$$
P(x, y) = \begin{cases} 1, x \le y \\ 0, x > y \end{cases}
$$

$$
\begin{aligned}
\max_{x_i} \quad & 0^T X && (6)\\
s.t \quad & \sum_{i=1}^{n} x_{ij} \le c_j, \text{ for j = 0...m} && (7)\\
& \sum_{j=1}^{m} x_{ij} \le 1, \text{ for i = 0...n} && (8)\\
& \sum_{j=1}^{m} P(d_{uj}, d_{uv})x_{uj} + \sum_{i=1}^{n} P(b_{i,v}, b_{u,v})x_{iv}\frac{1}{c_v} \ge 1, \qquad \forall(r_u, h_v) \in A && (9)\\
& x_{ij} = \{0, 1\} && (10)
\end{aligned}
$$

Cụ thể, hàm điều kiện ổn định đã được bỏ bớt đi số hạng x_{uv}. Ta có thể làm điều này bởi vì bản thân hàm $\sum_{j=1}^{m} P(d_{uj}, d_{uv})x_{uj} + \sum_{i=1}^{n} P(b_{i,v}, b_{u,v})x_{iv}\frac{1}{c_v}$ đã bao gồm cả trường hợp r_u ghép với h_v $(x_{uv} = 1)$.

6 Hiện thực mô hình trên MATLAB

Bản chất mô hình đã lập là dựa trên cấu trúc ma trận, do đó việc chuyển đổi từ mô hình trên giấy sang mô hình hiện thực trên MATLAB khá là đơn giản.

6.1 Chương trình cho mô hình ngặt

Đối với mô hình ngặt (trong phần 5.3). Ta cần thiết lập một vài thông số ban đầu như :

1. n: số người dân
2. m: số bệnh viện
3. D: ma trận nxm thể hiện danh sách ưu tiên của người dân (như đã quy ước trong mục 4)
4. B: ma trận nxm thể hiện danh sách ưu tiên của bệnh viện (như đã quy ước trong mục 4)
5. c: ma trận 1xm thể hiện khả năng tiếp nhận của bệnh viện (như đã quy ước trong mục 4)

Ngoài ra ta tạo thêm 1 ma trận a kích thước nx1 với tất cả các phần tử đều bằng 0 để hỗ trợ quá trình tính toán.
Thực hiện tính toán, ta khởi động công cụ cvx và tiến hành thiết lập các biến cũng như các hàm:

1. Chọn solver mosek (để giải bài toán quy hoạch nguyên)
2. Khai báo biến x(n,m) với chế độ binary (tức quy hoạch nguyên nhị phân)
3. Đặt hàm mục tiêu, ở đây ta không có hàm mục tiêu nên đặt là maximize (0)
4. Lần lượt đưa các hàm điều kiện vào trong solver.
    - Hàm (2), dùng hàm sum(x,1) để tính tổng từng cột trong ma trận x rồi kiểm tra với ma trận c để đảm bảo số bệnh nhân mỗi bệnh viện tiếp nhận không vượt quá khả năng của bệnh viện đó
    - Hàm (3), dùng hàm sum(x,2) để tính tổng từng hàng trong ma trận x rồi kiểm tra với ma trận a để đảm bảo mỗi bệnh nhân chỉ được 1 bệnh viện tiếp nhận
    - Hàm (4), không có hàm nào của MATLAB cho phép ta tính được kết quả của hàm này, nên ta sử dụng lồng ghép các vòng lặp để tính được các số hạng cần thiết. Trong MATLAB đã có hàm lt(x,y) có chức năng tương tự như P(x,y) nên ta sử dụng trực tiếp, không định nghĩa lại.

Nội dung cụ thể của chương trình được trình bày trong file code Ngat.m. Các thông số đầu vào tạm thời được để trống.

```
% ------------------------------- %
%     XOA MAN HINH VA CAC BIEN %
% ------------------------------- %
clear
clc

% ------------------------------- %
%      NHAP DU LIEU BAI TOAN    %
% ------------------------------- %
n = ...;     % So nguoi dan
m = ...;     % So benh vien
% Ma tran D bieu dien thu tu uu tien cua benh vien doi voi benh nhan
% ung voi tung hang
D = [...];
% Ma tran B bieu dien thu tu uu tien cua benh nhan doi voi benh vien
% ung voi tung cot
B = [...];
% Ma tran c bieu dien suc chua cua tung benh vien
c = [...];
```

```
    % Ma tran a bieu dien moi benh nhan chi duoc chon lua mot benh vien
    a = ones(n,1);

    % ------------------------------- %
    % GIAI BAI TOAN BANG SOLVER MOSEK %
    % ------------------------------- %
    cvx_begin
       cvx_solver mosek
       % Bien x(i,j) chi nhan gia tri 0 hoac 1
       % ung voi su ghep goi benh nhan r_i voi benh vien h_j
       variable x(n,m) binary
       % Toi da tong cac bien x(i,j)
       % tuc la cang nhieu cap duoc ghep doi cang tot
       maximize( 0 )
       subject to
          % Tong cac hang trong cung mot cot (so benh nhan duoc chon)
          % nho hon hoac bang suc chua cua benh vien
          sum(x,1) <= c;
          % Tong cac cot trong cung mot cot (so benh vien duoc chon)
          % nho hon hoac bang 1
          sum(x,2) <= a;
       % Bao dam khong co cac cap chan
       for u = 1:n
          for v = 1:m
             %Tinh so hang dau tien trong ham dieu kien on dinh
             t1 = 0;
             for j = 1:m
                t1 = t1 + lt(D(u,j),D(u,v)) * x(u,j);
             end
             %Tinh so hang thu hai trong ham dieu kien on dinh
             t2 = 0;
             for i = 1:n
                t2 = t2 + lt(B(i,v),B(u,v)) * x(i,v) / c(v);
             end
             %Xac lap ham dieu kien on dinh
             t1 + t2 + x(u,v) >= 1;
             %Ham dam bao cac cap (r_u,h_v) duoc xet nam trong A, neu
             %cap do khong nam trong A thi x_uv = 0
             if D(u,v) == m+n+1 || B(u,v) == m+n+1
                (eq(D(u,v),m+n+1) + eq(B(u,v),m+n+1)) * x(u,v) == 0;
             end
          end
       end
    cvx_end

    % ------------------------------- %
    %  HIEN THI KET QUA RA MAN HINH %
    % ------------------------------- %
    D
    B
    c
    x       % Cac cap duoc ghep doi
```

6.2 Chương trình cho mô hình không ngặt

Đối với mô hình không ngặt (5.4), việc xây dựng chương trình được tiến hành gần như tương tự so với mô hình ngặt. Điểm khác biệt duy nhất là hàm P(x,y) và hàm điều kiện ổn đinh. Đối với hàm P(x,y) ta sử dụng hàm le(x,y) thay vì lt(x,y) như trong mô hình ngặt. Đối với hàm điều kiện ổn định, ta loại bỏ số hạng (x_{uv}) ra khỏi hàm.
Nội dung cụ thể của chương trình được trình bày trong file code Khong_ngat.m. Các thông số đầu vào tạm thời được để trống.

```
% ------------------------------ %
%    XOA MAN HINH VA CAC BIEN %
% ------------------------------ %
clear
clc

% ------------------------------ %
%     NHAP DU LIEU BAI TOAN    %
% ------------------------------ %
n = ...;     % So nguoi dan
m = ...;     % So benh vien
% Ma tran D bieu dien thu tu uu tien cua benh vien doi voi benh nhan
% ung voi tung hang
D = [...];
% Ma tran B bieu dien thu tu uu tien cua benh nhan doi voi benh vien
% ung voi tung cot
B = [...];
% Ma tran c bieu dien suc chua cua tung benh vien
c = [...];
% Ma tran a bieu dien moi benh nhan chi duoc chon lua mot benh vien
a = ones(n,1);

% ------------------------------ %
% GIAI BAI TOAN BANG SOLVER MOSEK %
% ------------------------------ %
cvx_begin
   cvx_solver mosek
   % Bien x(i,j) chi nhan gia tri 0 hoac 1
   % ung voi su ghep goi benh nhan r_i voi benh vien h_j
   variable x(n,m) binary
   % Toi da tong cac bien x(i,j)
   % tuc la cang nhieu cap duoc ghep doi cang tot
   maximize( 0 )
   subject to
      % Tong cac hang trong cung mot cot (so benh nhan duoc chon)
      % nho hon hoac bang suc chua cua benh vien
      sum(x,1) <= c;
      % Tong cac cot trong cung mot cot (so benh vien duoc chon)
      % nho hon hoac bang 1
      sum(x,2) <= a;
      % Bao dam khong co cac cap chan
      for u = 1:n
         for v = 1:m
```

```
                %Tinh so hang dau tien trong ham dieu kien on dinh
                t1 = 0;
                for j = 1:m
                   t1 = t1 + le(D(u,j),D(u,v)) * x(u,j);
                end
                %Tinh so hang thu hai trong ham dieu kien on dinh
                t2 = 0;
                for i = 1:n
                   t2 = t2 + le(B(i,v),B(u,v)) * x(i,v) / c(v);
                end
                %Xac lap ham dieu kien on dinh
                t1 + t2 >= 1;
                %Ham dam bao cac cap (r_u,h_v) duoc xet nam trong A, neu
                %cap do khong nam trong A thi x_uv = 0
                if D(u,v) == m+n+1 || B(u,v) == m+n+1
                   (eq(D(u,v),m+n+1) + eq(B(u,v),m+n+1)) * x(u,v) == 0;
                end
             end
          end
    cvx_end

    % ------------------------------ %
    %  HIEN THI KET QUA RA MAN HINH %
    % ------------------------------ %
    D
    B
    c
    x
```

7 Thử nghiệm chương trình

Trên thực tế, phạm vi của dữ liệu đầu vào cho bài toán này rất đa dạng, có thể dao động từ vài bệnh viện, vài người dân cho đến vài trăm, vài nghìn bệnh viện, người dân. Trong báo cáo này, chúng ta chỉ xem xét các ví dụ nhỏ để có thể đánh giá tính chính xác của chương trình, đối với các tập dữ liệu lớn hơn, việc đánh giá trở nên khó khăn hơn rất nhiều. Các dữ liệu thử nghiệm được lưu lại trong các file test.txt đính kèm báo cáo

7.1 Ví dụ 1: dữ liệu nhỏ, đơn giản

Trong trường hợp này, ta sử dụng dữ liệu do nhóm tự lập và tính toán. Nội dung của ví dụ là:

- n = 4, m = 5
- $D = \begin{bmatrix} 2 & 3 & 1 & 10 \\ 1 & 2 & 10 & 10 \\ 10 & 10 & 1 & 10 \\ 10 & 3 & 2 & 1 \\ 3 & 2 & 4 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 1 & 10 & 3 & 10 \\ 2 & 2 & 10 & 10 \\ 10 & 10 & 1 & 10 \\ 10 & 1 & 2 & 2 \\ 3 & 3 & 4 & 1 \end{bmatrix}$, $c = \begin{bmatrix} 3 & 1 & 2 & 1 \end{bmatrix}$

Trong ví dụ này, kết quả kì vọng đạt được là $X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Qua chạy thử nghiệp với MATLAB, kết quả thu được từ mô hình ngặt là:

```
x =
(1,1)    1
(2,1)    1
(3,3)    1
(4,3)    1
(5,4)    1
```

Kết quả là chính xác so với kì vọng.
Toàn bộ nội dung của ví dụ (dữ liệu, quá trình chạy và kết quả) được lưu lại trong file **test0.txt**.

7.2 Ví dụ 2: dữ liệu từ tài liệu nghiên cứu[3]

Ví dụ này được trích từ tài liệu của Jay Sethuraman, Chung-Piaw Teo và Liwen Quian. Dữ liệu đầu vào:

- n = 11, m = 5
- $D = \begin{bmatrix} 2 & 17 & 1 & 4 & 3 \\ 1 & 4 & 2 & 3 & 5 \\ 4 & 5 & 3 & 1 & 2 \\ 3 & 17 & 1 & 2 & 4 \\ 1 & 3 & 17 & 2 & 17 \\ 4 & 3 & 2 & 1 & 5 \\ 3 & 1 & 4 & 17 & 2 \\ 1 & 3 & 2 & 5 & 4 \\ 2 & 17 & 17 & 1 & 3 \\ 2 & 4 & 1 & 5 & 3 \\ 3 & 5 & 4 & 2 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 10 & 17 & 8 & 2 & 6 \\ 11 & 6 & 5 & 3 & 1 \\ 1 & 7 & 4 & 8 & 8 \\ 6 & 17 & 6 & 5 & 2 \\ 5 & 1 & 17 & 7 & 17 \\ 9 & 4 & 2 & 9 & 5 \\ 2 & 2 & 7 & 17 & 4 \\ 8 & 5 & 3 & 10 & 7 \\ 3 & 17 & 17 & 6 & 10 \\ 7 & 3 & 9 & 1 & 3 \\ 4 & 8 & 1 & 4 & 9 \end{bmatrix}$, $c = \begin{bmatrix} 4 & 1 & 3 & 2 & 1 \end{bmatrix}$

Theo tác giả của ví dụ, tất cả các trường hợp ghép cặp ổn định là

Matching	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}
M_1	h_3	h_1	h_4	h_3	h_1	h_3	h_2	h_1	h_4	h_1	h_5
M_2	h_1	h_3	h_4	h_3	h_1	h_3	h_2	h_1	h_4	h_1	h_5
M_3	h_3	h_1	h_5	h_3	h_1	h_3	h_2	h_1	h_4	h_1	h_4
M_4	h_1	h_3	h_5	h_3	h_1	h_3	h_2	h_1	h_4	h_1	h_5
M_5	h_5	h_3	h_3	h_4	h_1	h_3	h_2	h_1	h_1	h_1	h_4
M_6	h_5	h_4	h_3	h_1	h_1	h_3	h_2	h_3	h_1	h_1	h_4
M_7	h_4	h_4	h_3	h_1	h_1	h_3	h_2	h_3	h_1	h_5	h_1

Qua chạy thử nghiệm với mô hình ngặt, nhóm nhận được kết quả:

x =
(5,1) 1
(8,1) 1
(9,1) 1
(10,1) 1
(7,2) 1
(2,3) 1
(3,3) 1
(6,3) 1
(4,4) 1
(11,4) 1
(1,5) 1

Kết quả nhận được là cố định qua nhiều lần thử. Kết quả này ứng với Matching M_5. Nội dung dữ liệu cũng như kết quả được ghi lại trong file **test1.txt**.

7.3 Ví dụ 3: dữ liệu cho trường hợp không ngặt

Trong ví dụ này, trong cả danh sách ưu tiên của người dân và bệnh viện đều có ưu tiên không ngặt. Dữ liệu đầu vào như sau:

- n = 8, m = 5
- $D = \begin{bmatrix} 1 & 14 & 2 & 14 & 14 \\ 1 & 14 & 3 & 2 & 2 \\ 1 & 2 & 14 & 14 & 3 \\ 1 & 2 & 14 & 2 & 14 \\ 2 & 3 & 1 & 14 & 14 \\ 2 & 1 & 1 & 14 & 3 \\ 4 & 14 & 1 & 2 & 3 \\ 14 & 14 & 14 & 2 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 5 & 14 & 3 & 14 & 14 \\ 2 & 14 & 1 & 2 & 3 \\ 1 & 3 & 14 & 14 & 1 \\ 3 & 3 & 14 & 3 & 14 \\ 2 & 1 & 1 & 14 & 14 \\ 4 & 2 & 2 & 14 & 2 \\ 2 & 14 & 3 & 4 & 2 \\ 14 & 14 & 14 & 1 & 2 \end{bmatrix}$, $c = \begin{bmatrix} 2 & 3 & 2 & 2 & 2 \end{bmatrix}$

Kết quả thu được từ việc chạy chương trình là:

x =
(2,1) 1
(3,1) 1
(4,2) 1
(6,2) 1
(1,3) 1
(5,3) 1
(7,4) 1
(8,5) 1

Dữ liệu đầu vào và kết quả chạy chương trình được lưu trong file **test2.txt**

7.4 Các ví dụ khác:

Một số ví dụ khác được nhóm chạy thử nghiệm được lưu lại trong các file **test3.txt, text4.txt, text5.txt**. Các dữ liệu này đều có kích thước ở mức trung bình nhỏ. Chỉ có ở mức này nhóm mới có thể đảm bảo kiểm tra được độ chính xác.

8 Nhận xét, đánh giá

Sau khi hoàn thành bài tập, nhóm đã họp và đưa ra một số nhận xét, đánh giá về công việc cũng như quá trình làm việc chung.

8.1 Nhận xét

- Hiệu quả thuật toán không cao do đã sử dụng vòng lặp lồng 3 trong việc kiểm tra điều kiện ổn định (độ phức tạp O(n^3)). Có thể có phương án khác hiệu quả hơn trong việc hình thành công thức này, tuy nhiên nhóm chưa có cách khắc phục. Đối với các mẫu dữ liệu kích thước lớn, chương trình chạy chậm và có dấu hiệu không phản hồi.

- Đối với các trường hợp người dân và bệnh viện xếp danh sách hoàn toàn độc lập, tức là người dân có thể thích bệnh viện nhưng bệnh viện lại không thích bệnh nhân. Điều này có nghĩa là $d_{ij} \neq m+n+1$ nhưng $b_{ij} = m+n+1$. Nếu trường hợp này xảy ra, mô hình giải có khả năng rất cao không thể đưa ra một đáp án khả thi nào. Điều ngược lại cũng xảy ra. Vì vậy, tất cả các dữ liệu đầu vào của bài toán đều phải đảm bảo một điều kiện đó là $(d_{ij} = m+n+1) \leftrightarrow (b_{ij} = m+n+1)$

- Mô hình giải bài toán ưu tiên không ngặt là trường hợp tổng quát hơn của bài toán ưu tiên ngặt. Vì lý do đó mà trong tất cả các bài toán ưu tiên ngặt, mô hình ưu tiên không ngặt cũng có thể giải được và cho kết quả tương tự. Còn mô hình ngặt lại không thể nào giải được các bài toán ưu tiên không ngặt.

- Các bài toán dạng này

8.2 Đánh giá

- Mục tiêu ban đầu đặt ra của nhóm đã hoàn thành: lập được mô hình, hiện thực được trên MATLAB.

- Có các thành viên: Đức Anh, Hiển, Bình có tích cực tham gia vào quá trình tiến hành bài tập lớn. Tú, Tuấn hoàn thành nhiệm vụ được giao.

- Việc thực hiện gặp một số trục trặc do một số thành viên không đảm bảo được deadline chung đề ra.

- Nhóm thống nhất chia đều điểm số

Tài liệu

[1] Thầy Lê Hồng Trang "**MM-HRP-project_spring2017-TNMT**", *ngày viết :* 9/03/2017.

[2] CVX Introduction "**link: http://cvxr.com/cvx/doc/intro.html/**", *What is CVX*, lần truy cập cuối: 15/04/2017.

[3] Jay Sethuraman - Chung-Piaw Teo - Liwen Qian. "**Many-To-One Stable Matching: Geometry and Fairness**" , March 2004; revised January 2005, January 2006