

Lời nói đầu

Như chúng ta đã biết, cuộc cách mạng công nghiệp 4.0 đã chính thức nổ ra, đánh dấu một kỷ nguyên mới – kỷ nguyên của công nghệ thông tin và dữ liệu lớn. Tại thời điểm này, những cỗ máy, với những công cụ toán học cũng như công cụ từ những ngành khoa học phụ trợ, đã thực sự có thể độc lập “tư duy”, tự mình tìm kiếm những thông tin có giá trị từ một mớ thông tin khổng lồ, hỗn độn. Những tư tưởng về học máy, về trí tuệ nhân tạo giờ đây đã len lỏi vào từng ngõ ngách trong cuộc sống. Từ những thứ đơn giản như những cỗ máy bán nước, việc mua sắm gia đình tới những quy trình phức tạp như phẫu thuật nội soi, quản lý xuất nhập cảnh, ... tất cả đều dần dần sử dụng trí tuệ nhân tạo như là một cách thức để nâng cao tối đa tính hiệu quả cũng như độ chính xác.

Tuy vậy, đa số vẫn không chú ý rằng, ẩn sau những khái niệm tưởng như cao siêu ấy, đơn thuần chỉ là những công thức toán học. Những mô hình học máy phức tạp kia, thực chất chỉ là những lần lặp với mục tiêu tối giản hàm mục tiêu, ... Sau cùng, mọi thứ chúng ta đang có ở đây cũng chỉ là những kiến thức toán học đã được xây dựng từ trước.

Toán học cũng có nhiều lĩnh vực. Dù vậy, có thể khẳng định rằng, Đại số tuyến tính đóng một vai trò quan trọng nhất trong Khoa học máy tính. Từ việc xử lý ảnh, dựng hình đến việc dựng các mô hình tensors liên kết với nhau tạo thành mạng neuron, ... tất cả chúng về bản chất đều là những thao tác trên ma trận số. Điều này càng được khẳng định, khi mà mọi chương trình đào tạo chính quy về các ngành liên quan đến tin học, Đại số tuyến tính luôn luôn là một trong những học phần cơ bản mà sinh viên phải vượt qua đầu tiên.

Có thể khái quát được, việc nghiên cứu Đại số tuyến tính gói gọn trong việc xem xét những hệ phương trình tuyến tính. Những xem xét này đã có từ rất lâu đời: Những nhà toán học Trung Hoa đã nhắc đến nó trong tập Cửu chương về nghệ thuật

toán học (The Nine Chapters on the Mathematical Art). Hay toán học cận đại cũng ghi nhận những nghiên cứu của Decartes hay Leibniz về những hệ phương trình tuyến tính này.

Hiểu được tầm quan trọng của của vấn đề này, trong báo cáo lần này, nhóm 8-Hoàng Minh Đức và Phạm Nhật Quang sẽ đi tìm hiểu và trình bày việc giải hệ phương trình tuyến tính bằng việc sử dụng phương pháp khử Gauss nhằm giải ra nghiệm đúng của hệ phương trình.

Xin cảm ơn TS Hà Thị Ngọc Yến đã giúp chúng tôi chỉnh sửa và hoàn thành báo cáo lần này. Do hạn chế về thời gian cũng như về kiến thức nên báo cáo ắt hẳn sẽ không tránh khỏi những thiếu sót. Quý bạn đọc có góp ý, xin vui lòng gửi đến địa chỉ duc.hmbk@gmail.com Mọi đóng góp mang tính xây dựng đều được hoan nghênh!

Chương I. Kiến Thức Cơ bản

1.1. Hệ phương trình tuyến tính – Các khái niệm cơ bản

1.1.1. Hệ phương trình tuyến tính

Hệ phương trình tuyến tính m phương trình n ẩn có dạng:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases} \quad (1.1)$$

Trong đó, các hệ số a_{ij}, b_i thuộc trường K. Trong phạm vi báo cáo ta coi $K \equiv \mathbb{R}$

Từ đây ta có các định nghĩa sau:

a. Các hệ số của hệ phương trình lập thành một ma trận A, gọi là **ma trận của các hệ số**.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

b. Ma trận \bar{A} thu được từ ma trận A bằng cách ghép thêm số cột hạng tự do gọi là **ma trận mở rộng của hệ**:

$$\bar{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} & b_n \end{bmatrix}$$

c. Ma trận B thu được từ cột hệ số tự do, gọi là **ma trận hệ số tự do của hệ**.

$$B = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix}$$

Như vậy, nếu gọi X là ma trận cột biểu diễn các ẩn của hệ phương trình, thì hệ phương trình từ đây có thể viết được dưới dạng:

$$AX = B \quad (1.2)$$

Từ đây về sau, ta sử dụng dạng trên để biểu diễn hệ phương trình là chủ yếu.

1.1.2. Nghiệm của hệ phương trình tuyến tính:

Mỗi nghiệm của hệ phương trình (1.2) là một bộ sắp thứ tự (c_1, c_2, \dots, c_n) thỏa mãn tất cả các phương trình của hệ (1.2) được thỏa mãn khi thay các ẩn x_i tương ứng bởi c_i

Một hệ phương trình có thể có duy nhất nghiệm, vô số nghiệm hoặc không có nghiệm.

1.1.3. Biến đổi tương đương

Hai hệ phương trình được gọi là tương đương nếu hai hệ phương trình đó có cùng không gian nghiệm.

Các phép biến đổi sơ cấp của hệ phương trình là các phép biến đổi:

- a. Nhân một phương trình nào đó của hệ với một số khác 0
- b. Cộng vào một phương trình nào đó của hệ với một phương trình khác đã nhân với một số khác 0
- c. Đổi chỗ hai phương trình của hệ

Từ đây, ta có thể rút ra một số nhận xét sau:

- Các phép biến đổi sơ cấp của hệ phương trình tương ứng với các phép biến

đổi trên hàng của ma trận mở rộng của hệ

- Khi thực hiện các phép biến đổi sơ cấp, ta thu được một hệ phương trình mới tương đương với hệ phương trình ban đầu

1.1.4. Các kết quả quan trọng

a. Hệ phương trình Cramer - Định lý Cramer

Một hệ phương trình Cramer là một hệ phương trình mà số ẩn bằng số phương trình và định thức của ma trận hệ số khác không.

Từ đây, ta có thể phát biểu định lý Cramer.

Định lý 1.1 (Cramer): Hệ phương trình Cramer luôn có duy nhất nghiệm.

Chứng minh:

Xét hệ phương trình Cramer:

$$AX=B$$

Do đây là hệ phương trình Cramer nên ta có:

$$\det(A) \neq 0$$

Như vậy tồn tại ma trận nghịch đảo của A là A^{-1} . Nhân cả hai vế của hệ phương trình ban đầu với A^{-1} ta được:

$$X = A^{-1} \cdot B$$

Như vậy hệ phương trình đã cho có nghiệm duy nhất là: $X = A^{-1} \cdot B$

b. Định lý Kronecker – Capelli và các hệ quả

Trong định lý này, ta sẽ nêu ra điều kiện để một hệ phương trình tổng quát có nghiệm, cũng như các hệ quả xung quanh.

Định lý 1.2 (Kronecker – Capelli): Hệ phương trình tuyến tính:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \dots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases}$$

Có nghiệm khi và chỉ khi hạng của ma trận hệ số bằng hạng của ma trận mở rộng

Từ đây, ta rút ra hai hệ quả quan trọng sau:

Hệ quả 1.2.1: Nếu hệ phương trình (1.1) với n ẩn có $r_A = r_{\bar{A}} = n$ thì hệ có nghiệm duy nhất.

Chứng minh:

Giả sử hệ có $r_A = r_{\bar{A}} = n$. Khi đó tồn tại định thức con cơ sở của của M (định thức con khác 0), cả A và \bar{A} . Bây giờ chúng ta sẽ chứng minh rằng mỗi dòng của \bar{A} khác với n dòng đầu tiên là tổ hợp tuyến tính của n dòng này. Khi đó hệ phương trình đã cho tương đương với hệ phương trình:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n \end{cases}$$

Do $M \neq 0$ nên hệ này là một hệ Cramer, vì vậy nó có nghiệm duy nhất theo định lý Cramer.

Hệ quả 1.2.2:

Giả sử $r_A = r_{\bar{A}} < n$ thì hệ có vô số nghiệm phụ thuộc vào $n - r_A$ tham số.

Chứng minh:

Giả sử $r_A = r_{\bar{A}} < n$. Kí hiệu M là định thức con cơ sở của A và \bar{A} . Hơn nữa giả thiết rằng định thức này nằm ở góc phía trên bên trái của A. Do đó:

$$M = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1r} \\ a_{21} & a_{22} & \dots & a_{2r} \\ \dots & \dots & \dots & \dots \\ a_{r1} & a_{r2} & \dots & a_{rr} \end{vmatrix}$$

Bởi vì mỗi dòng thứ i của \bar{A} , là tổ hợp tuyến tính của r dòng đầu tiên nên hệ đã cho tương đương với:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 - a_{1(r+1)}x_{(r+1)} - \dots - a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 - a_{2(r+1)}x_{(r+1)} - \dots - a_{2n}x_n \\ \dots \\ a_{r1}x_1 + a_{r2}x_2 + \dots + a_{rr}x_r = b_r - a_{r(r+1)}x_{(r+1)} - \dots - a_{rn}x_n \end{cases} \quad (1.3)$$

Đây là một hệ phương trình Cramer r phương trình r ẩn $x_1, x_2, x_3, \dots, x_r$. Với mỗi bộ giá trị (c_{r+1}, \dots, c_n) của các ẩn (x_{r+1}, \dots, x_n) ta được một nghiệm của (1.3) là (c_1, \dots, c_r) . Khi đó $(c_1, c_2, \dots, c_r, c_{r+1}, \dots, c_n)$ là nghiệm của (1.1)

***Nhận xét:** Những hệ quả của định lý Kronecker Capelli rất quan trọng, nó là cơ sở giúp cho chúng ta thực hiện phương pháp khử Gauss khi cài đặt phương pháp giải quyết những trường hợp phương trình vô nghiệm hoặc có vô số nghiệm.

Chương II. Phương pháp Gauss (Sử dụng cho ma trận vuông)

2.1. Ý tưởng phương pháp

Phương pháp Gauss (Gauss Elimination) là một phương pháp giải đúng hệ phương trình tuyến tính. Nó sử dụng những phép biến đổi cơ bản, biến ma trận mở rộng của hệ phương trình tuyến tính về dạng ma trận hình thang (Row echelon form), sau đó sẽ giải lần lượt từng phương trình của hệ theo thứ tự từ dưới lên, khi mà các phương trình ở dưới sẽ đơn giản hơn (ít ẩn hơn), và khi đó việc tìm từng giá trị của ẩn thỏa mãn hệ phương trình sẽ dễ dàng hơn.

2.2. Nội dung phương pháp

Như đã nói ở trên, phương pháp Gauss là việc thực hiện những phép biến đổi tương đương trên ma trận mở rộng của hệ phương trình tuyến tính, hoặc phép thế để giải hệ phương trình này. Và ở đây, nhóm em xin được phép trình bày phương pháp Gauss cho ma trận vuông cấp n (n phương trình – n ẩn):

*Cho ma trận:

$$\begin{cases} E_1: a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ E_2: a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots \\ E_n: a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n \end{cases} \quad (1)$$

- Ta tách hệ phương trình trên thành dạng ma trận:

$$AX = B$$

trong đó:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}; \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}; \quad B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

- Từ các ma trận trên, ta xây dựng ma trận bổ sung:

$$\bar{A} = \left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array} \right]$$

- Tiếp theo, chúng ta sẽ thực hiện phương pháp Gauss theo 2 quá trình

+) Quá trình thuận:

Giả sử $a_{11} \neq 0$, ta sẽ thực hiện thao tác sau:

$$((E_j) - \frac{a_{ji}}{a_{ii}} \times (E_i)) \rightarrow (E_j) \quad \text{với mỗi } j = i + 1, i + 2, \dots, n$$

Các thao tác này sẽ khử hệ số của biến x_i trong mỗi hàng dưới hàng thứ i với mỗi $i = 1, 2, \dots, n - 1$. Ma trận cuối cùng sẽ có dạng như sau:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = a_{1.n+1}$$

$$a_{22}x_2 + \cdots + a_{2n}x_n = a_{2.n+1}$$

.....

$$a_{nn}x_n = a_{n.n+1}$$

Trong đó các phần tử ở cột $n+1$ tương ứng với b_1, b_2, \dots, b_n .

+) Quá trình nghịch:

Bước 1: Giải phương trình hàng E_n ta được $x_n = \frac{a_{n.n+1}}{a_{n.n}}$

(Lưu ý: Nếu hàng E_n gồm toàn những số 0 thì bỏ qua bước này và đặt $x_n = t$ để tiếp tục giải những bước tiếp theo)

Bước 2: Sử dụng nghiệm đã thu được từ việc giải phương trình hàng E_n ta giải phương trình ở hàng E_{n-1} và sẽ có $x_{n-1} = \frac{b_{n-1} - a_{n-1,n} \times x_n}{a_{n-1,n-1}}$

Bước 3: Tương tự như trên, ta lần lượt giải ngược từ dưới lên các x_{n-2}, x_{n-3}, \dots và có công thức tổng quát như sau:

$$x_i = \frac{a_{i,n+1} - \sum_{j=i+1}^n a_{ij} \times x_j}{a_{ii}} \text{ với mỗi } i = n-1, n-2, \dots, 2, 1$$

Để hiểu rõ hơn, ta đến với một số ví dụ như sau:

Ví dụ 1:

$$-2x + 3y + 3z = -9$$

$$3x - 4y + z = 5$$

$$-5x + 7y + 2z = -14$$

- Ta tạo ra ma trận bổ sung \bar{A} : $\left(\begin{array}{ccc|c} -2 & 3 & 3 & -9 \\ 3 & 4 & 1 & 5 \\ -5 & 7 & 2 & -14 \end{array} \right)$
- Biến đổi trên hàng để đưa về ma trận bậc thang \bar{A}' :

$$\left(\begin{array}{ccc|c} 1 & -1 & 4 & -4 \\ 0 & -1 & -11 & 17 \\ 0 & 0 & 0 & 0 \end{array} \right)$$

- Lần lượt giải từ dưới lên ta sẽ thu được $\begin{cases} x = -15t - 21 \\ y = -11t - 17 \\ z = t \end{cases}$

Ví dụ 2:

$$x + y + z = 2$$

$$3x - y - 2z = 4$$

$$5x + y - z = 10$$

- Ta tạo ra ma trận bổ sung \bar{A} : $\left(\begin{array}{ccc|c} 1 & 1 & 1 & 2 \\ 3 & -1 & -2 & 4 \\ 5 & 1 & -1 & 10 \end{array} \right)$

- Biến đổi trên hàng để đưa về ma trận bậc thang \bar{A}' : $\left(\begin{array}{ccc|c} 1 & 1 & 1 & 2 \\ 0 & 4 & 5 & 2 \\ 0 & 0 & -1 & 2 \end{array}\right)$

- Lần lượt giải từ dưới lên ta sẽ thu được $\begin{cases} x = 1 \\ y = 3 \\ z = -2 \end{cases}$

2.3. Thuật toán:

Input: Ma trận mở rộng của hệ phương trình tuyến tính n ẩn n phương trình

Output: Nghiệm của hệ phương trình tuyến tính (hoặc hệ phương trình không có nghiệm duy nhất)

Phần 1: Đưa về ma trận bậc thang

Với $i=1$:

B1: Chọn số p nhỏ nhất thỏa mãn $i \leq p \leq n$ và $a_{pi} \neq 0$, nếu không có p thỏa mãn thì tăng i lên 1 đơn vị

B2: Nếu $p \neq i$ thì đổi chỗ hàng p và hàng i

B3: Với $j = i + 1, i + 2, \dots, n$

Xét $m_{ij} = \frac{a_{ij}}{a_{ii}}$ thực hiện phép gán $(E_j - E_i \cdot M_{ij}) \rightarrow E_j$

B4: Kết thúc vòng lặp và tăng i lên 1 đơn vị, nếu i nhỏ hơn n và lặp lại bước 1

Phần 2: Kiểm tra các hàng

B5: Nếu $a_{nn} \neq 0$:

$$x_n = \frac{a_{n,n+1}}{a_{nn}}$$

B6: Với mỗi $i = n - 1, n - 2, \dots, 1$ thì:

$$x_i = \frac{a_{i,n+1} - \sum_{j=i+1}^n a_{ij} \cdot x_j}{a_{ii}}$$

Xuất output: nghiệm của hệ phương trình (x_1, \dots, x_n)

B7: Nếu $\begin{cases} a_{nj} = 0 \forall j = \overline{1, n} \\ a_{n,n+1} \neq 0 \end{cases}$

Xuất output: “hệ phương trình vô nghiệm”

Xét hàng E_n

B8: Nếu $a_{nj} = 0 \quad \forall j = \overline{1, n+1} \rightarrow$ Xóa hàng E_n

gán $n = n - 1$ và thực hiện B9 đến khi các hàng có ít nhất 1 phần tử $\neq 0$.

Phần 3: Trường hợp hệ có vô số nghiệm

B9: Với $i = 1, j = i$, kiểm tra $i \leq m$. Nếu sai chuyển tới bước cuối

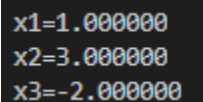
B10: Với hàng i chọn số j nhỏ nhất sao cho $a_{ij} \neq 0$, lưu j vào mảng $a_1[i]=j$ và tiếp tục tăng j thêm 1 đơn vị.

B11: Lưu vị trí j ứng với hàng i vào mảng $a[i]=j$

B12: Tăng ij thêm 1 đơn vị.

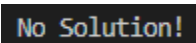
2.3. Một số ví dụ với code:

1. Giải hệ phương trình:
$$\begin{cases} x_1 + x_2 + x_3 = 2 \\ 3x_1 - x_2 - 2x_3 = 4 \\ 5x_1 + x_2 - x_3 = 10 \end{cases}$$

Kết quả: 

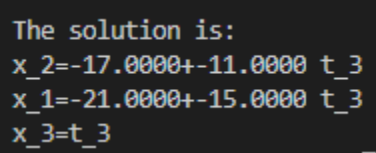
Đánh giá ví dụ: Đây là một ví dụ cơ bản, dễ thấy khi đưa về dạng ma trận thì $(rank(A) = rank(\bar{A}))$ nên hệ có nghiệm duy nhất. Điều đó chứng tỏ trong điều kiện tốt code hoạt động ổn.

2. Giải hệ phương trình:
$$\begin{cases} x_1 + x_2 + 2x_3 = 5 \\ 0x_1 + 0x_2 + 0x_3 = 4 \\ 5x_1 + x_2 - 2x_3 = 7 \end{cases}$$

Kết quả: 

Đánh giá ví dụ: Dễ thấy hệ trên vô nghiệm, code đưa về kết quả đúng là do đã kiểm tra như bước B7 của thuật toán.

3. Giải hệ phương trình:
$$\begin{cases} -2x_1 + 3x_2 + 3x_3 = -9 \\ 3x_1 - 4x_2 + x_3 = 5 \\ -5x_1 + 7x_2 + 2x_3 = -14 \end{cases}$$

Kết quả: 

Đánh giá ví dụ: Hệ phương trình trên lấy từ [Ví dụ 1](#), và code cho kết quả tương tự khi ta giải bằng tay.

***Phân công và tự đánh giá nhiệm vụ của nhóm:**

***Phân công:**

Lí thuyết: Cả hai cùng tìm hiểu lí thuyết của phương pháp

Thuật toán: Đức là người chịu trách nhiệm nắm vững lí thuyết. Từ đó viết thuật toán cho phương pháp.

Code: Quang kiểm tra phần thuật toán Đức viết và trình bày code theo thuật toán. Thêm đó Quang sẽ chuẩn bị slide báo cáo.

***Đánh giá:**

Ở đây ta nêu ra một số đặc điểm của phương pháp gauss:

-Là một phương pháp giải đúng nghiệm của phương trình

-Chia ra làm hai bước: Bước thuận và bước nghịch

-Ý tưởng đơn giản, tuy nhiên việc cài đặt hơi phức tạp bởi các trường hợp đặc biệt có khá nhiều.

*So sánh với các phương pháp khác:

-Sai số tính toán khuếch đại hơn so với phương pháp Gauss-Jordan do phương pháp Gauss-Jordan có áp dụng kĩ thuật chọn phần tử trội.

Ưu điểm:

-Biết cách phân chia phần việc như code, thuật toán, báo cáo cho từng thành viên của nhóm.

-Biết cách phân bổ thời gian hoàn thành công việc được giao

-Có trách nhiệm với công việc được giao

Nhược điểm:

-Có những phần việc như thuật toán còn sơ sài

-Chưa thảo luận với thành viên trong nhóm được tốt.

-Trình bày còn những thiếu sót về mặt kiến thức.

Chấm điểm:7/10

Lời kết

Trên đây là phần trình bày về phương pháp Gauss mà nhóm mình đã tìm hiểu. Đây là một phương pháp đơn giản nhưng khá hữu hiệu để giải đúng những hệ phương trình tuyến tính có kích thước nhỏ. Tuy vậy, nó cũng có một số hạn chế về tốc độ, khả năng thực hiện song song, ... Do vậy, nó thường được sử dụng ở những hệ có hàng ngàn phương trình. Với những hệ có hàng triệu phương trình, những phương pháp lặp như lặp Jacobi, lặp Gauss Seidel... thường được lựa chọn vì những lợi ích về tốc độ mà nó mang lại.

Do hạn chế về thời gian và kiến thức nên phần trình bày của chúng mình có thể một vài kiến thức còn bị thiếu sót. Rất mong quý bạn đọc có thể thông cảm, hoặc hơn nữa, có thể góp ý sửa đổi bổ sung giúp cho báo cáo hoàn chỉnh hơn.

Cảm ơn TS. Hà Thị Ngọc Yến đã giúp nhóm mình nhận xét, sửa đổi, đánh giá bài báo cáo lần này!