VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

MATHEMATICAL MODELING (CO2011)

Assignment

"Cutting stock problem"

Instructor(s): Mai Xuân Toàn

Students: Nguyễn Vương Trung Nam - 2212153 *(Group DT01 - Team 07, **Leader**)*
Nguyễn Anh Lâm - 2211795
Bùi Xuân Bách - 2210179
Ngô Đức Luân - 2211949
Lê Quang Minh - 2212047

HO CHI MINH CITY, August 2024

Contents

Member list & Workload

No.	Fullname	Student ID	Problems	% done
1	Nguyễn Vương Trung Nam	2212153	- Mô hình hóa bài toán - Trình bày đánh giá và kết quả chạy thử - Tìm kiếm giải thuật - Viết báo cáo Latex	100%
2	Nguyễn Anh Lâm	2211795	- Thực thi giải thuật - Tối ưu giải thuật và thiết kế chương trình - Đánh giá kết quả chạy thử	100%
3	Lê Quang Minh	2212047	- Mô hình hóa bài toán - Giới thiệu vấn đề	100%
4	Ngô Đức Luân	2211949	- Literature Review	90%
5	Bùi Xuân Bách	2210179	- Literature Review	90%

Table 1: Member list & workload

1 Giới thiệu vấn đề

Bài toán Cut Stock Problem (CSP) được Kantorovich đưa ra lần đầu tiên vào năm 1939. Nguồn gốc của vấn đề là trong ngành công nghiệp giấy. Với các cuộn giấy có chiều rộng cố định và một tập hợp các đơn đặt hàng cho các cuộn nhỏ hơn, mục tiêu của vấn đề là xác định cách cắt các cuộn thành các chiều rộng nhỏ hơn để hoàn thành các đơn đặt hàng theo cách giảm thiểu lượng phế liệu. Năm 1951, trước khi máy tính được sử dụng rộng rãi, L.V.Kantorovich và V.A.Zalgaller đã đề xuất giải quyết vấn đề sử dụng tiết kiệm vật liệu ở giai đoạn cắt với sự trợ giúp của lập trình tuyến tính (Linear Programming). Kỹ thuật đó sau này được đề xuất gọi là phương pháp tạo cột (column generation).

Đây là một bài toán tối ưu hóa trong toán học phát sinh từ các ứng dụng trong công nghiệp. CSP có thể được mở rộng ra thành các bài toán The one-dimensional cutting stock (1D-CSP) problem, 2D-CSP, 3D-CSP,... Tuy nhiên ở đây ta chỉ xét bài toán 1D-CSP, đây là một bài toán NP-hard nổi tiếng trên thế giới trong những năm gần đây, xảy ra trong quá trình sản xuất ở nhiều ngành công nghiệp như ngành thép, may mặc và nhôm.

1D-CSP là một bài toán lập trình tuyến tính số nguyên (ILP) được dùng để cắt vật liệu theo một số lượng và một chiều nhất định sao cho giảm tổn thất và chi phí cắt và vật liệu dư thừa có thể được tái sử dụng trong công đoạn tiếp theo. Bài toán 1D-CSP được sử dụng trong nền công nghiệp vật liệu khi mà các công ty, xí nghiệp trên thế giới đang cạnh tranh nguồn nguyên vật liệu và sử dụng để sản xuất sao cho ít tốn kém nhất, từ đó việc áp dụng các phương pháp tối ưu nhằm tạo ra lợi thế cạnh tranh trên thị trường.

Đối với bài tập lớn lần này nhóm sẽ tập trung giải quyết vấn đề **1D-CSP**

with multiple stock nhưng không có sự ràng buộc về vật liệu.

2 Acknowledge

2.1 Tổng quan lịch sử các giải pháp đề ra để giải quyết bài toán CSP

2.1.1 Phương pháp thuật toán (algorithm)

- Một trong những tiến bộ quan trọng nhất đến từ Gilmore và Gomory (1961), họ đề xuất phương pháp giải quyết bài toán bằng quy hoạch tuyến tính (linear programming). Phương pháp này vẫn có ảnh hưởng lớn và là nền tảng trong lĩnh vực này.
- Waescher và Gau (1994) kết luận rằng giải pháp nguyên tối ưu (optimal integer) có thể giải quyết cho hầu hết các trường hợp của CSP. Công trình nghiên cứu của họ giúp cải thiện trong việc tìm ra các giải pháp cho CSP.
- Scheithauer và một số nhà nghiên cứu khác (1997) giới thiệu một thuật toán cutting plane để giải quyết chính xác 1D-CSP. Phương pháp này tập trung vào việc tinh chỉnh không gian nghiệm để tìm ra các mẫu tối ưu một cách hiệu quả. Mukhacheva và một số nhà nghiên cứu khác (2003) đề xuất một phương pháp nhánh-cận cải tiến dành riêng cho 1D-CSP, cải thiện hiệu suất tính toán cho một số trường hợp bài toán.
- Belov và Scheithauer (2007) phát triển thuật toán branch-and-cut-and-price giải quyết 1D-CSP và 2D-two-stage CSP. Phương pháp giải quyết bài toán của họ tập trung vào việc kết hợp các thuật toán chính xác và heuristics để tìm ra các giải pháp hiệu quả cho các bài toán CSP phức tạp.

- Johnston và Sadinlija (2007) phát triển một mô hình giải quyết tính phi tuyến của 1D-CSP, giữa pattern variables và pattern run-lengths bằng cách áp dụng các biến 0-1.
- Belov và Scheithauer (2006) kết hợp các thuật toán cutting plane với column generation cho 1D-CSP với đa stock có chiều dài khác nhau. Họ tiếp tục phát triển thuật toán branch-and-cut-and-price giải quyết 1D-CSP và 2D-two-stage CSP (2007). Phương pháp giải quyết bài toán của họ tập trung vào việc kết hợp các thuật toán chính xác và heuristics để tìm ra các giải pháp hiệu quả cho các bài toán CSP phức tạp.

2.1.2 Phương pháp gần đúng (heuristics)

- Gradisar cùng một số nhà nghiên cứu khác (1999) đã phát triển phương thức heuristic tuần hoàn (sequential heuristic procedure), phương pháp dùng để tối ưu hóa 1D - CSP khi tất cả chiều dài của stock khác nhau. Umetani và những nhà nghiên cứu khác (2003) đề ra hướng tiếp cận dựa vào metaheuristics và kết hợp với phương pháp tạo mẫu tương thích (adaptive pattern generation method).
- Dikili và những nhà nghiên cứu khác (2011) đề ra phương pháp loại bỏ để giải quyết 1D-CSP, trong sản xuất tàu thủy, bằng cách sử dụng các cutting patterns thu được bằng việc phương pháp phân tích ở giai đoạn mô hình toán học.

2.1.3 Phương pháp kết hợp (algorithm và heuristics)

- Gradisar cùng một số nhà nghiên cứu khác (1999) còn đề ra phương pháp kết hợp phân loại theo kiểu mẫu của phương pháp quy hoạch tuyến tính (pattern-oriented of linear programing) và phân loại theo vật phẩm

của phương pháp heuristic tuần tự (item-oriented of sequential heuristic procedure).

- Các nhà nghiên cứu như Vance et al. (1994, 1998), Vanderbeck (1996, 2000), và Valério de Carvalho (1999) đã đưa ra một số khả năng về việc kết hợp 2 phương pháp column generation và phương pháp nhánh cận. Họ đã tìm được giải pháp cho một phần lớn các bài toán CSP.
- Phương pháp của Dyckhoff (1990) chia giải pháp cho CSP thành 2 nhóm: theo kiểu mẫu (pattern-oriented) và theo loại vật phẩm (item-oriented). Ngoài ra, ông còn đề xuất phương pháp phân loại theo 4 đặc điểm: theo chiều hướng (dimensionality), theo loại (kind of assignment), phân loại của đối tượng lớn (assortment of large objects), phân loại của đối tượng nhỏ (assortment of small objects).

3 Giải thuật và Heuristic đề xuất

3.1 Constructive Heuristic

3.1.1 Exhaustive Repetition Heuristic - Algorithm

Các bước hiện thực heuristic

- Step 1: Xây dựng một mẫu tốt cho từng vật liệu (stock) loại k = 1,... h.
- Step 2: Lựa chọn mẫu cắt.
- Step 3: Áp dụng mẫu cắt đã chọn.
- Step 4: Cập nhật nhu cầu cho từng loại vật thể (item) dựa trên các mẫu cắt đã chọn.

- Step 5: Kiểm tra điều kiện dừng (khi nhu cầu của vật thể đã thõa mãn), nếu điều kiện không thõa quay lại Step 1.

Phân tích các bước giải trên và đề ra các giải thuật để tối ưu Heuristic

Dựa trên các bước của Heuristic trên nhóm nhận thấy có sự xuất hiện bài toán ***One - Dimensional Bin Packing***.

Hình trên cho thấy sự tương đồng giữa bài toán **One - Dimensional**

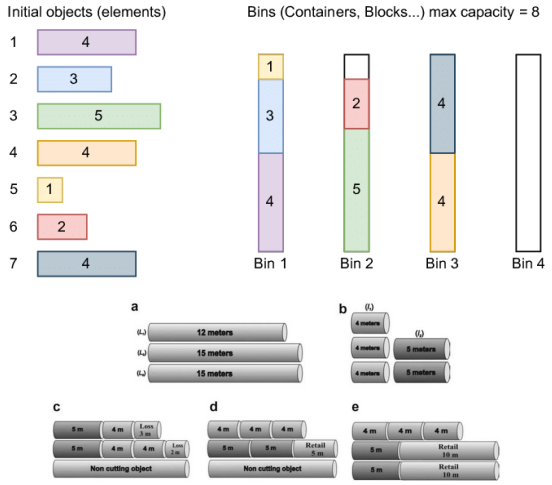

Figure 1: Mô phỏng bài toán Bin Packing (Hình trên) và Mô phỏng bài toán Cutting Stock Problem ((a) vật liệu (b) sản phẩm yêu cầu (c) - (e) các giải pháp)

Bin Packing với việc xây dựng các mẫu cắt cho thuật toán (là bài toán **Pattern Generation**) khi ta cần sắp sếp một số loại vật thể cố định vào một khuôn mẫu nhất định và tối ưu các mẫu đó để giảm thiểu phần dư thừa nhiều nhất có thể. Chính vì thế nhóm đề xuất các thuật toán khá phổ biến khi giải quyết bài toán **One - Dimensional Bin Packing** để tối ưu **Exhaustive Repetition Heuristic - Algorithm** mà nhóm đã sử dụng.

Cụ thể ở đây là hai giải thuật sau:

- Giải thuật First-Fit Decrease (FFD)

- Giải thuật Greedy

3.1.2 First-Fit Decrease Algorithm

Trong ***Cutting Stock Problem***, một trong những thách thức quan trọng là tối ưu hóa việc sử dụng vật liệu bằng cách tạo ra các mẫu cắt hiệu quả. Thuật toán First Fit Decreasing (FFD) là một trong những phương pháp phổ biến được sử dụng để tạo mẫu cắt. Thuật toán này được biết đến với khả năng tìm ra giải pháp hợp lý trong thời gian ngắn và dễ dàng triển khai.

Hướng tiếp cận

Thuật toán ***First Fit Decreasing (FFD)*** sắp xếp các đối tượng cần cắt (items) theo thứ tự giảm dần về kích thước, sau đó lần lượt xếp chúng vào các không gian trống (bins) có sẵn theo nguyên tắc *"first fit"* (vừa khít đầu tiên). Trong ngữ cảnh bài toán cắt tấm, các đối tượng này là các kích thước của các phần tử cần cắt và các không gian trống là các tấm vật liệu có kích thước nhất định.

3.1.3 Greedy Algorithm

Trong ***Cutting Stock Problem***, mục tiêu chính là tối ưu hóa việc sử dụng vật liệu bằng cách tạo ra các mẫu cắt sao cho lượng phế liệu bị lãng phí là nhỏ nhất. Thuật toán Greedy là một trong những phương pháp tiếp cận cơ bản và phổ biến nhất, được sử dụng để giải quyết bài toán này. Phương pháp này đặc biệt hữu ích trong các bài toán yêu cầu giải pháp nhanh chóng và đơn giản.

Hướng tiếp cận

Thuật toán ***Greedy*** hoạt động dựa trên nguyên tắc chọn lựa cục bộ tối

ưu tại mỗi bước với hy vọng rằng sự lựa chọn này sẽ dẫn đến giải pháp toàn cục tối ưu. Trong ngữ cảnh bài toán cắt tấm, tại mỗi bước, thuật toán chọn phần tử có kích thước lớn nhất hoặc phần tử dễ xếp nhất (phù hợp nhất) vào không gian trống hiện tại của tấm vật liệu.

3.2 Mô hình 1

Ở mô hình 1 nhóm sẽ xây dựng phương phảp giải quyết bài toán **CSP** theo **Exhaustive Repetition Heuristic** với **Step 1** của heuristic được tối ưu bằng thuật toán **First Fit Decrease (FFD)**

Flowchart

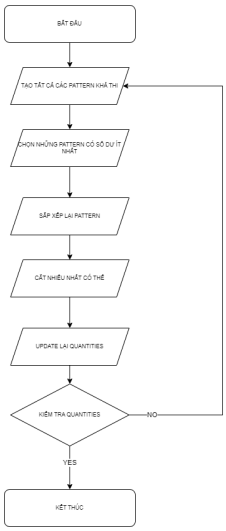

Figure 2: Mô hình 1 flowchart

3.3 Mô hình 2

Ở mô hình 2 nhóm sẽ xây dựng phương phảp giải quyết bài toán **CSP** theo **Exhaustive Repetition Heuristic** với **Step 1** của heuristic được tối ưu bằng thuật toán **Greedy**

Flowchart

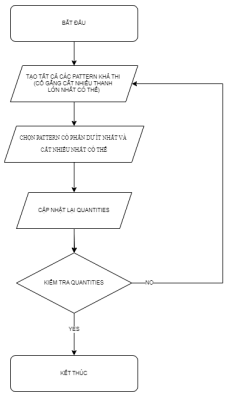

Figure 3: Mô hình 2 flowchart

3.4 Thiết kế chương trình thực thi giải thuật

Chương trình được nhóm hiện thực hoàn toàn bằng ngôn ngữ **Python** với đầu vào (input) và đầu ra (output) được thiết kế như sau:

Input:

Input là một file .txt với các thông số

- **dòng đầu tiên** là một dãy số nguyên cách nhau bởi dấu phẩy tượng trưng cho một list độ dài của các thanh cần cắt.
- **dòng thứ hai** là một dãy số nguyên cách nhau bởi dấu phẩy tượng trưng cho nhu cầu tương ứng của mỗi thanh so với dòng đầu tiên.
- **dòng thứ ba** là một dãy số nguyên cách nhau bởi dấu phẩy tượng trưng cho một list các thanh được cắt

```
2, 3, 4
6, 4, 4
5, 6, 9
```

Listing 1: Input file

Output:

```
All patterns need to use:
{'stock': 9, 'cuts': {0: 1, 1: 1, 2: 1}, 'leftover': 0}
{'stock': 9, 'cuts': {0: 1, 1: 1, 2: 1}, 'leftover': 0}
{'stock': 9, 'cuts': {0: 1, 1: 1, 2: 1}, 'leftover': 0}
{'stock': 9, 'cuts': {0: 1, 1: 1, 2: 1}, 'leftover': 0}
{'stock': 5, 'cuts': {0: 2, 1: 0, 2: 0}, 'leftover': 1}

Summarize:
use {'stock': 9, 'cuts': {0: 1, 1: 1, 2: 1}, 'leftover': 0} 4 times
use {'stock': 5, 'cuts': {0: 2, 1: 0, 2: 0}, 'leftover': 1} 1 times

Remaining quantities: [0, 0, 0]
Sum of leftover: 1
Total time: 0.0 sec
Total stock: 5
```

Listing 2: Terminal Console

Output của chương trình mà nhóm đã được thiết kế thành 6 dữ kiện:

- **All patterns need to use** đây là thứ tự sử dụng các pattern được chọn ra trong vùng feasible pattern để giải quyết trường hợp.
- **Summarize** Thống kê lại các loại pattern đã được sử dụng
- **Remaining quantities** cập nhật lại demand khi chương trình đã chạy xong, list này giá trị 0 tức nghĩa chương trình đã hoàn thành demand được đề bài đưa ra.
- **Sum of leftover** tổng số lượng dư.

- **Total time** thời gian thực thi.
- **Total stock** tổng số lượng các thanh đã được sử dụng(không phân biệt từng loại thanh).

Xử lý ngoại lệ:

Những trường hợp sau đây chương trình sẽ thông báo lỗi:

- Số lượng yêu cầu của các vật thể khác với số lượng vật thể chương trình sẽ báo lỗi sau: **Input Error: the length of item demand list are different from the lenght of item list**

- Độ dài các vật thể yêu cầu lớn hơn so với các vật liệu được cắt chương trình sẽ báo lỗi sau: **Input Error: One or more item is longer than the maximum length of the stock.**

4 Mô hình hóa vấn đề

4.1 Phân tích vấn đề

Trong hầu hết các ngành công nghiệp, chi phí nguyên liệu chiếm tỷ lệ cao trong tổng giá thành (hơn 80%). Bài toán cắt nguyên liệu (CSP) là một trong những bài toán nổi tiếng trong nghiên cứu hoạt động, được định nghĩa nhằm cải thiện chất lượng sử dụng nguyên liệu. Bài toán CSP này có thể được mô tả như sau: Có một tập hợp các vật liệu hình chữ nhật với các chiều rộng khác nhau (nhưng có cùng chiều dài $\boldsymbol{L}$) và chúng ta sẽ chia chúng thành các mảnh nhỏ hơn có chiều rộng mong muốn (với cùng chiều dài $\boldsymbol{L}$).

Trước tiên, các chiều rộng mong muốn được sắp xếp theo thứ tự giảm dần $(w_1, w_2, ...)$ và sau đó chia các vật liệu ban đầu thành các mảnh nhỏ có chiều rộng w_1 nhiều nhất có thể. Nếu còn lại phần nguyên liệu thừa, ta tiếp

tục với chiều rộng mong muốn tiếp theo w_2 và chia phần thừa đó thành các mảnh nhỏ có chiều rộng w_2. Bằng cách tiếp tục quy trình tương tự cho đến chiều rộng nhỏ nhất yêu cầu, ta có thể tạo ra tất cả các mẫu cắt và cuối cùng tính toán phần thừa nguyên liệu tối thiểu.

Đặc biệt đối với cách tiếp cận lần này trường hợp sẽ đơn giản hơn do sẽ không có ràng buộc về số lượng thanh được sử dụng hoặc số lượng dư thừa tối đa, tuy nhiên mô hình bài toán cũng phải đáp ứng được việc giải quyết các vấn đề khi có nhiều hơn 1 vật liệu.

4.2 Mô hình bài toán

4.2.1 Hệ thống ký hiệu

Ký hiệu	Định nghĩa
h	Số loại vật liệu được cắt (với các kích cỡ khác nhau)
n	Số lượng các vật thể yêu cầu
m	Số lượng mẫu
a_{ikj}	Số lần xuất hiện vật thể thứ i trên vật liệu thứ k trong mẫu thứ j
c_{kj}	Hệ số chi phí (hoặc hệ số tương tự) liên quan đến việc sử dụng mẫu cắt thứ j cho vật liệu loại k.
d_i	Yêu cầu của vật thể thứ i
x_{kj}	Số lần mẫu cắt thứ j của vật liệu loại k được sử dụng.

Table 2: Hệ thống các ký hiệu

4.2.2 Hàm mục tiêu

Mục tiêu chính của chúng ta đối với mô hình bài toán này là tối thiểu hóa tổng lãng phí khi ta sử dụng các mẫu cắt. Đây là một trong nhiều cách dùng để mô hình hóa bài toán trên, chúng ta có thể thay thế hàm mục tiêu bằng việc tối thiểu hóa số lượng mẫu cắt được sử dụng thay vì tối thiểu tổng số lãng phí khi ta sử dụng các mẫu cắt. Tuy nhiên, ở đây nhóm quyết

định chọn hướng mô hình hóa với mục tiêu tối thiểu hóa tổng số lượng lãng phí.

$$\text{Min} \sum_{k=1}^{h} \sum_{j=1}^{m} c_{kj}.x_{kj} \text{ (Total Cutloss)} \tag{1}$$

Với x_{kj} là số lần mẫu cắt thứ j của vật liệu loại k được sử dụng và c_{kj} Tổn thất trong mẫu thứ j vật liệu loại k, vì vậy tích $x_{kj}.c_{kj}$ chính là tổng tổn thất của của mẫu thứ j của vật liệu loại k.

4.2.3 Các ràng buộc

4.2.4 Ràng buộc về nhu cầu

Đây là ràng buộc nhu cầu, đảm bảo rằng tổng số vật thể $\boldsymbol{i}$ được cắt từ tất cả các vật liệu theo các mẫu khác nhau phải lớn hơn hoặc bằng nhu cầu d_i của vật thể đó.

$$\sum_{k=1}^{h} \sum_{j=1}^{m} a_{ikj}.x_{kj} \geq d_i, \text{ i = 1,... m} \tag{2}$$

4.2.5 Ràng buộc về giới hạn vật liệu

Ràng buộc này là phần mở rộng thêm, tuy nhiên trong nghiên cứu của nhóm sẽ các giải thuật và heuristic được xây dựng sẽ không giải quyết vấn đề có chứa ràng buộc này.

$$\sum_{j=1}^{m} \leq e_k, \text{ k = 1, 2,... h} \tag{3}$$

Ràng buộc này sẽ đảm bảo số lượng vật liệu $\boldsymbol{k}$ được cắt theo mẫu $\boldsymbol{j}$ sẽ không vượt quá số lượng vật liệu $\boldsymbol{k}$ có sẵn.

4.2.6 Ràng buộc nguyên và không âm

Ràng buộc để đảm bảo biến quyết định x_{jk} và biến c_{jk} của hàm mục tiêu phải là số nguyên và không âm.

$$\begin{cases} x_{jk} \geq 0 \\ c_{jk} \geq 0 \end{cases}, \text{ j = 1, 2,... m; k = 1, 2,... h} \tag{4}$$

5 Phân tích kết quả thực thi mô hình trên các bộ dữ liệu

Ở phần này nhóm sẽ đánh giá các mô hình để giải bài toán bằng các giải thuật và heuristic đã trình bài ở trên thông qua hai phần, phần 1 sẽ đánh giá và so sánh kết quả giữa hai mô hình mà nhóm đã xây dựng dựa trên bộ dữ liệu tự tạo của các nhà nghiên cứu đi trước đồng thời sẽ so sánh hiệu quả của giải thuật mà nhóm nghiên cứu với các giải thuật mà các nhà nghiên cứu trước đã xây dựng trên cùng một bộ dữ liệu, phần 2 nhóm sẽ áp dụng các mô hình đã xây dựng được vào một case-study thực tế và đánh giá khả năng ứng dụng của mô hình.

5.1 The Case Study (Real World Data)

5.1.1 Mô tả trường hợp

Để minh họa tính ứng dụng của mô hình mà nhóm đã phát triển, nghiên cứu trường hợp liên quan đến vấn đề cắt gặp phải tại một công ty sản xuất pallet gỗ quy mô vừa và nhỏ (SME) ở Thái Lan sẽ được trình bày và thảo luận.

Gỗ thông được nhập khẩu từ New Zealand, vận chuyển dưới dạng container. Gỗ thông nhập khẩu được cắt thành các thanh hình chữ nhật tiêu

chuẩn có kích thước 150 × 100 mm với các độ dài (Lk) lần lượt là 4,2 m, 4,8 m, 5,4 m, và 6,0 m một cách ngẫu nhiên. Chi phí trung bình của gỗ thông nhập khẩu (bao gồm cả chi phí vận chuyển) là 265 Baht/foot^3. Gỗ thông phải được cắt thành các sản phẩm có kích thước yêu cầu tùy thuộc vào mẫu pallet cần sản xuất. Độ dài của mỗi sản phẩm (li) được thể hiện trong Bảng bên dưới.

Figure 4: Tấm gỗ(Stock) và phần dư thừa của nó(off-cuts)

Item (mm)	2316	2200	1650	1460	1140	1120	980
Demand	789	254	468	554	4686	1179	969

Table 3: Bảng nhu cầu của Case Study

5.1.2 Kết quả thực thi

Kết quả sau khi chạy ***mô hình 1*** để giải quyết ***Case-Study*** trên như sau:

```
Remaining quantities: [0, 0, 0, 0, 0, 0, 0]
Sum of leftover: 361296
Total time: 0.015075445175170898 sec
Total stock: 2448
```

Listing 3: Terminal Console

Kết quả sau khi chạy ***mô hình 2*** để giải quyết ***Case-Study*** trên như sau:

```
Remaining quantities: [0, 0, 0, 0, 0, 0, 0]
Sum of leftover: 475896
Total time (greedy): 0.003785848617553711 sec
Total stock (greedy): 2452
```

Listing 4: Terminal Console

Stock length (mm)	**Required pattern (piece)**	**No. of item to be cut with the length of (mm)**						
		2316	**2200**	**1650**	**1460**	**1140**	**1120**	**980**
4200	217	0	0	0	0	0	2	2
4800	254	0	1	0	1	1	0	0
	394	2	0	0	0	0	0	0
	1	1	0	0	0	2	0	0
	1107	0	0	0	0	4	0	0
	1	0	0	0	0	2	2	0
	89	0	0	0	0	0	4	0
	1	0	0	0	0	0	3	1
5400	234	0	0	2	0	0	1	1
6000	150	0	0	0	2	0	1	2
Number of each item		789	254	468	554	4686	1179	969

Table 4: Kế hoạch cắt các thanh gỗ được tạo ra bởi mô hình 1

Stock length (mm)	**Required pattern (piece)**	**No. of item to be cut with the length of (mm)**						
		2316	**2200**	**1650**	**1460**	**1140**	**1120**	**980**
4200	1	0	0	1	1	0	0	1
	1	0	0	0	0	0	1	3
	1	0	0	0	0	0	0	1
4800	394	2	0	0	0	0	0	0
	233	0	0	2	1	0	0	0
	1171	0	0	0	0	4	0	0
	1	0	0	0	0	2	2	0
	294	0	0	0	0	0	4	0
5400	126	0	2	0	0	0	0	1
	1	0	1	1	1	0	0	0
	106	0	0	0	3	0	0	1
6000	1	1	1	0	1	0	0	0
	122	0	0	2	1	0	0	6
Number of each item		789	254	712	676	4686	1179	969

Table 5: Kế hoạch cắt các thanh gỗ được tạo ra bởi mô hình 2

5.1.3 Kết luận

Table 6: So sánh hai mô hình

	Mô hình 1	Mô hình 2	Change
Volume of stock required (foot^3)	6,327.42	6,379.2	-51.78
Cost of stock (Baht/$foot^3$)	1,676,766.3	1,690,488	-13721.7
Volume of waste (foot^3)	191.31	252.16	-60.85
Cost of waste (Baht/$foot^3$)	50,697	66,822.4	-16,125.25

Nhìn chung đối với bài toán này ta sử dụng Mô hình 1 với cách tối ưu bằng thuật toán FFD sẽ là giải pháp tốt hơn so với Mô hình 1 tối ưu bằng thuật toán Greedy.

Về thời gian thực thi: sự chênh lệch thời gian thực thi của hai mô hình trên là không đáng kể, việc này sẽ không ảnh hưởng gì khi áp dụng vào thực tế, tuy nhiên trong trường hợp này thời gian thực thi của mô hình 2 lại có phần nhỉnh hơn.

Về việc áp dụng thực tế: Khi xét về mặt dư thừa và chi phí mô hình 2 sẽ tốt hơn so với mô hình 1, nhưng nếu xét về khía cạnh môi trường thì doanh nghiệp có thể sẽ ưu tiên sử dụng mô hình 1 do dựa trên kết quả khi thực thi trên máy tính thì ***Total stock*** của mô hình 1 ít hơn so với mô hình 2 4 thanh. Việc chênh lệch như vậy sẽ giảm bớt việc tiêu thụ điện năng, việc khai thác gỗ sử dụng, việc vận chuyển các thanh nguyên liệu.

So sánh với giải pháp của một vài nhà nghiên cứu: Ở đây nhóm sẽ so sánh với nhóm nghiên cứu của P Wattanasiriseth A Krairit [4] cũng thực hiện nghiên cứu trên Case Study tương tự, nhưng tiếp cận vấn đề với một phương pháp khác. Cụ thể, nhóm nghiên cứu sau giải quyết Case-Study bằng **Giải thuật Cutting Plan** và **Giải thuật Optimal Cutting Plan**. Kết quả đạt được của nhóm được thể hiện như sau:

Dựa theo kết quả của nhóm nghiên cứu trên thì thuật toán Cutting Plans

Table 3. The comparison of current cutting plan and optimal cutting plan.

List	Current cutting plan	Optimal cutting plan	Change
Volume of stock required (foot^3)	6564.56	6257.22	-307.34
Cost of stock (Baht/month)	1,739,609	1,658,163	-81,446 *(4.68%)*
Volume of waste (foot^3)	423.38	123.37	-300.01
Cost of waste (Baht/month)	112,195	32,693	-79,502 *(70.86%)*

Figure 5: Tiếp cận Case Study bằng giải thuật Cutting Plans

có thể dễ dàng nhận thấy kém hiệu quả hơn so với Mô hình 1 và Mô hình 2 mà nhóm đã xây dựng.

5.2 Synthetic Data

Mặc dù có nhiều tài liệu phong phú về CSPs, rất ít người cung cấp dữ liệu thực tế cho các bài toán của họ. So sánh trực tiếp với họ là rất khó, nếu không muốn nói là không thể. Trong các nghiên cứu thực nghiệm của nhóm, 10 bài toán thử nghiệm được sử dụng (mười bài toán chuẩn được lấy từ Hinterdingand Khan [1]). Mô tả chi tiết về các bài toán này được cung cấp bên dưới. Tổng số mục được yêu cầu thay đổi từ 20 đến 126. Các bài toán 1-5 là các bài toán CSP có nhiều độ dài vật liệu và các bài toán 1a đến 5a là các bài toán CSP có một độ dài tấm. Các bài toán có cùng chỉ số có cùng số lượng mục được yêu cầu, ví dụ như các bài toán 4 và 4a có cùng số lượng mục được yêu cầu.

Table 7: The benchmark problems

Problem 1: 10, 13, 15								
Problem 1a: 14	20 items							
Item length	3	4	5	6	7	8	9	10
No. required	5	2	1	2	4	2	1	3
Problem 2: 10, 13, 15								
Problem 2a: 15	50 items							
Item length	3	4	5	6	7	8	9	10
No. required	4	8	5	7	8	5	5	8
Problem 3: 10, 13, 15, 20, 22, 25								
Problem 3a: 25	60 items							
Item length	3	4	5	6	7	8	9	10
No. required	6	12	6	5	15	6	4	6
Problem 4: 13, 20, 25								
Problem 4a: 25	60 items							
Item length	3	4	5	6	7	8	9	10
No. required	7	12	15	7	4	6	8	1

Problem 5: 4300, 4250, 4150, 3950, 3800, 3700, 3550, 3500										
Problem 5a: 4300	126 items									
Item length	2350	2250	2200	2100	2050	2000	1950	1900	1850	1700
No. required	2	4	4	15	6	11	6	15	13	5
Item length	1650	1350	1300	1250	1200	1150	1100	1050	-	-
No. required	2	9	3	6	10	4	8	3	-	-

Problem 6: 86, 83, 79, 76, 72										
Problem 6a: 86	200 items									
Item length	21	23	24	25	26	27	28	29	31	-
No. required	10	14	10	7	14	4	13	9	5	-
Item length	33	34	35	37	38	41	42	44	47	-
No. required	10	13	10	11	15	12	15	15	13	-

Problem 7: 120, 115, 110, 105, 100												
Problem 7a: 120 200 items												
Item length	22	26	27	28	29	30	31	32	34	36	37	38
No. required	12	8	27	15	25	7	10	22	5	16	19	21
Item length	39	46	47	48	52	53	54	56	58	60	63	64
No. required	26	16	12	26	20	25	9	17	22	14	17	9

Problem 8: 120, 115, 110												
Problem 8a: 120 400 items												
Item length	22	23	24	26	27	28	29	30	31	36	39	41
No. required	12	8	27	15	25	7	10	22	5	16	19	21
Item length	42	48	49	50	51	54	55	56	59	60	66	67
No. required	26	16	12	26	20	25	9	17	22	14	17	9

Problem 9: 120, 115, 110, 105, 100												
Problem 9a: 120 400 items												
Item length	21	22	24	25	27	29	30	31	32	33	34	35
No. required	13	15	7	5	9	9	3	15	18	17	4	17
Item length	38	39	42	44	45	46	47	48	49	50	51	52
No. required	20	9	4	19	9	12	15	3	20	14	15	6
Item length	53	54	55	56	57	59	60	61	63	65	66	67
No. required	4	7	5	19	19	6	3	7	20	5	10	7

Problem 10: 120, 115, 110												
Problem 10a: 120 600 items												
Item length	21	22	23	24	27	28	29	30	31	33	35	
No. required	13	19	24	20	23	24	15	5 7 24	16	12	24	
Item length	36	39	40	41	42	43	44	45	46	47	48	50
No. required	16	4	20	24	6	14	21	20	24	2	11	26
Item length	51	54	56	57	58	61	62	63	64	65	66	67
No. required	23	25	8	16	10	14	6	19	18	11	27	16

Table 8: Caption

5.2.1 Kết quả thực thi

Với bảng dữ liệu trên, từ problem 1 - 10 sẽ là dạng **Multiple Stock** và từ problem 1a - 10a sẽ là dạng **Single Stock**.

Prob. no.	Exhaustive Repetion Heuristic with FFD			Exhaustive Repetion Heuristic with Greedy		
	Stocks used	Total wastage	Execute time	Stocks used	Total wastage	Execute time
1	9	1	0.0058	12	8	0.00
2	23	1	0.00	29	0	0.00
3	16	2	0.0523	30	0	0.00
4	21	14	0.00	30	10	0.00
5	56	2950	0.0458	56	2950	0.00
6	84	25	0.0185	87	70	0.00
7	73	4	0.4333	74	94	0.00
8	146	92	0.2831	151	257	0.00
9	156	12	2.8308	160	102	0.00
10	220	135	1.9229	224	275	0.0157
1a	9	3	0.0155	9	3	0.00
2a	23	13	0.00	23	13	0.00
3a	15	0	0.0157	16	25	0.00
4a	20	36	0.00	20	36	0.00
5a	53	11450	0.0156	56	24350	0.00
6a	82	361	0.0156	85	619	0.00
7a	68	84	0.0950	71	444	0.00
8a	145	332	0.0947	150	932	0.0019
9a	152	382	0.6348	153	502	0.00
10a	216	130	0.7169	221	730	0.00

Table 9: So sánh giải thuật FFD và Greedy trong Exhaustive Repetion Heuristic

5.2.2 Nhận xét kết quả thực thi

*** Hiệu quả của mô hình 1 (Sử dụng thuật toán FFD)**

FFD tỏ ra có hiệu quả hơn trong việc giảm thiểu lượng vật liệu dư thừa (wastage) nhờ vào khả năng sắp xếp lại các phần tử sau mỗi lần tạo mẫu (pattern generation). Việc sắp xếp theo thứ tự giảm dần giúp tối ưu hóa không gian sử dụng trên các tấm vật liệu, từ đó giảm thiểu lượng vật liệu

không sử dụng được.

Tuy nhiên, nhóm nhận thấy thời gian thực thi của FFD khá dài do quá trình sắp xếp và kiểm tra liên tục. Điều này có thể gây ra sự chậm trễ trong các ứng dụng thực tế yêu cầu xử lý nhanh.

*** Hiệu quả của mô hình 2 (Sử dụng thuật toán Greedy)**

Greedy có ưu điểm là chạy nhanh hơn nhiều so với FFD và dễ dàng triển khai hơn. Điều này đặc biệt hữu ích trong các trường hợp yêu cầu giải pháp nhanh chóng mà không cần quá nhiều xử lý phức tạp.

Tuy nhiên, nhược điểm lớn của Greedy là nó có thể tạo ra nhiều vật liệu dư thừa hơn do không có cơ chế sắp xếp lại các phần tử như FFD. Khi gặp các trường hợp dư thừa bằng nhau, thuật toán Greedy hiện tại của nhóm vẫn chưa được tối ưu, dẫn đến việc không thể giảm thiểu lượng vật liệu dư thừa một cách hiệu quả.

5.2.3 Đề xuất cải tiến và phát triển

Để cải thiện kết quả, nhóm cần xem xét cải tiến quá trình lựa chọn mẫu cắt (pattern selection) của thuật toán Greedy. Cụ thể, trong các trường hợp có lượng dư thừa bằng nhau, nên áp dụng các tiêu chí bổ sung để chọn mẫu cắt sao cho lượng dư thừa cuối cùng được giảm thiểu tối đa.

Đồng thời, nhóm cũng cần tìm cách tối ưu hóa quá trình tạo mẫu của FFD để giảm thời gian thực thi. Điều này có thể thực hiện bằng cách cải tiến thuật toán hoặc áp dụng các kỹ thuật tối ưu hóa khác để giảm thiểu thời gian tính toán mà vẫn đảm bảo được hiệu quả giảm lượng dư thừa.

6 Kết luận

Thông qua bài tập lớn lần này nhóm cũng đã tìm hiểu được rất nhiều vấn đề về một bài toán NP-hard cụ thể ở đây là bài toán **One-dimensional Cutting Stock Problem**, từ khâu mô hình hóa vấn đề thành một bài toán **Quy hoạch tuyền tính** đến khâu tiếp cận giải pháp tối ưu bằng các giải thuật (algorithm) hoặc Heurisitic.

Nhờ đó, nhóm cũng đã đúc kết được nhiều kinh nghiệm cụ thể. Thông qua quá trình tìm hiểu nhóm nhận thầy bài toán **One-Dimensional Cutting Stock Problem** có rất nhiều cách để đưa về một bài toán tuy theo cách tiếp cận sẽ có những mô hình khác nhau, nhưng mục đích chung vẫn là tối ưu số lượng thanh gỗ (stock), hơn thế nữa nhóm nhận thấy bài toán cũng có nhiều "biến thể" khác nhau chẳng hạn như *Single Stock*, *Multiple Stock*, hoặc có thể giới hạn về số lượng thanh (stock) được sử dụng, lúc đó khi mô hình bài toán số lượng ràng buộc cũng sẽ tăng lên. Về giải pháp mà nhóm đề ra, trong quá trình làm việc nhóm nhận thấy bài toán có rất nhiều giải thuật cũng như Heurisitc để tiếp cận và mỗi cách tiếp cận như vậy sẽ phù hợp với từng dạng bài khác nhau, với thuật toán **Column Generation** một thuật toán khá phổ biến và cổ điển để giải quyết bài toán trên nhưng chưa thực sự phù hợp đối với dạng bài *Multiple Stock*, trong quá trình tìm kiếm các giải thuật nhóm cũng đã phải cân nhắc rất nhiều về hiệu quả của giải thuật cũng như tính khả thi khi thực thi giải thuật để cân đối với năng lực của nhóm.

Lời cuối nhóm xin gửi lời cảm ơn đến giảng viên bộ môn Ths. Mai Xuân Toàn đã tạo điều kiện để nhóm có thể nghiên cứu và hiểu thêm về các phương pháp tiếp cận để giải quyết một bài toán NP-hard.

7 Thông tin các file bài nộp

Bài nộp bao gồm các file sau:

- Folder Assignment-CO2011-CSE233-2212153-LOG.txt: file nhật kí làm bài tập lớn của nhóm
- Assignment-CO2011-CSE233-2212153-Report.pdf: file pdf báo cáo của nhóm.
- Folder Assignment-CO2011-CSE233-2212153-Latex: folder báo cáo bằng latex của nhóm
- Folder Assignment-CO2011-CSE233-2212153-ProgramFolder: folder bao gồm các file **input**, **output** mà nhóm đã chạy và file **FFD.py** là file code python thực thi mô hình 1, **Greedy.py** là file code python thực thi mô hình 2.

8 Tài liệu tham khảo

References

[1] Liang, K.-H., Yao, X., Newton, C., & Hoffman, D. (1999). A new evolutionary approach to cutting stock problems with and without contiguity. Received 1 August 1998; received in revised form 1 July 1999.

[2] Tanir, D., Ugurlu, O., Guler, A., & Nuriyev, U. (2015). One-dimensional Cutting Stock Problem with Divisible Items. Received 1 June 2014; accepted 15 January 2015.

[3] Cherri, A. C., Arenales, M. N., & Yanasse, H. H. (2020). The one-dimensional cutting stock problem with usable leftover - A heuristic approach. Received 1 April 2019; received in revised form 30 September 2019; accepted 12 October 2019.

[4] Wattanasiriseth, P., & Krairitl, A. (2020). An Application of Cutting-Stock Problem in Green Manufacturing: A Case Study of Wooden Pallet Industry. Received 22 June 2020; accepted 23 August 2020.

[5] AMPL. (n.d.). Hands-On Optimization with AMPL in Python. Retrieved from https://shorturl.at/0YrXd.